



Informatica® B2B Data Transformation
10.5

Guia do Usuário

© Copyright Informatica LLC 2001, 2021

Este software e documentação contêm informações de propriedade da Informatica LLC, são fornecidos sob um contrato de licença que contém restrições quanto a seu uso e divulgação, e são protegidos por leis de copyright. A engenharia reversa do software é proibida. Não está permitida de forma alguma a reprodução ou a transmissão de qualquer parte deste documento (seja por meio eletrônico, fotocópia, gravação ou quaisquer outros) sem o consentimento prévio da Informatica LLC. Este Software pode estar protegido por patentes dos EUA e/ou internacionais e outras patentes pendentes.

O uso, duplicação ou divulgação do Software pelo Governo dos Estados Unidos estão sujeitos às restrições estipuladas no contrato de licença de software aplicável e como estabelecido em DFARS 227.7202-1(a) e 227.7702-3(a) (1995), DFARS 252.227-7013®(1)(ii) (OCT 1988), FAR 12.212(a) (1995), FAR 52.227-19 ou FAR 52.227-14 (ALT III), conforme aplicável.

As informações contidas neste produto ou documentação estão sujeitas a alteração sem aviso prévio. Informe-nos por escrito caso encontre quaisquer problemas neste produto ou documentação.

Informatica, Informatica Platform, Informatica Data Services, PowerCenter, PowerCenterRT, PowerCenter Connect, PowerCenter Data Analyzer, PowerExchange, PowerMart, Metadata Manager, Informatica Data Quality, Informatica Data Explorer, Informatica B2B Data Transformation, Informatica B2B Data Exchange Informatica On Demand, Informatica Identity Resolution, Informatica Application Information Lifecycle Management, Informatica Complex Event Processing, Ultra Messaging, Informatica Master Data Management e Live Data Map são marcas comerciais ou marcas registradas da Informatica LLC nos Estados Unidos e em jurisdições pelo mundo. Todos os outros nomes de outras companhias e produtos podem ser nomes ou marcas comerciais de seus respectivos proprietários.

Partes desta documentação e/ou software estão sujeitas a copyright de terceiros, incluindo sem limitação: Copyright DataDirect Technologies. Todos os direitos reservados. Copyright © Sun Microsystems. Todos os direitos reservados. Copyright © RSA Security Inc. Todos os direitos reservados. Copyright © Ordinal Technology Corp. Todos os direitos reservados. Copyright © Aandacht c.v. Todos os direitos reservados. Copyright Genivia, Inc. Todos os direitos reservados. Copyright Isomorphic Software. Todos os direitos reservados. Copyright © Meta Integration Technology, Inc. Todos os direitos reservados. Copyright © Intalio. Todos os direitos reservados. Copyright © Oracle. Todos os direitos reservados. Copyright © Adobe Systems Incorporated. Todos os direitos reservados. Copyright © DataArt, Inc. Todos os direitos reservados. Copyright © ComponentSource. Todos os direitos reservados. Copyright © Microsoft Corporation. Todos os direitos reservados. Copyright © Rogue Wave Software, Inc. Todos os direitos reservados. Copyright © Teradata Corporation. Todos os direitos reservados. Copyright © Yahoo! Inc. Todos os direitos reservados. Copyright © Glyph & Cog, LLC. Todos os direitos reservados. Copyright © Thinkmap, Inc. Todos os direitos reservados. Copyright © Clearpace Software Limited. Todos os direitos reservados. Copyright © Information Builders, Inc. Todos os direitos reservados. Copyright © OSS Nokalva, Inc. Todos os direitos reservados. Copyright Edifecs, Inc. Todos os direitos reservados. Copyright Cleo Communications, Inc. Todos os direitos reservados. Copyright © International Organization for Standardization 1986. Todos os direitos reservados. Copyright © ej-technologies GmbH. Todos os direitos reservados. Copyright © Jaspersoft Corporation. Todos os direitos reservados. Copyright © International Business Machines Corporation. Todos os direitos reservados. Copyright © yWorks GmbH. Todos os direitos reservados. Copyright © Lucent Technologies. Todos os direitos reservados. Copyright © University of Toronto. Todos os direitos reservados. Copyright © Daniel Veillard. Todos os direitos reservados. Copyright © Unicode, Inc. Copyright IBM Corp. Todos os direitos reservados. Copyright © MicroQuill Software Publishing, Inc. Todos os direitos reservados. Copyright © PassMark Software Pty Ltd. Todos os direitos reservados. Copyright © LogiXML, Inc. Todos os direitos reservados. Copyright © 2003-2010 Lorenzi Davide, todos os direitos reservados. Copyright © Red Hat, Inc. Todos os direitos reservados. Copyright © The Board of Trustees of the Leland Stanford Junior University. Todos os direitos reservados. Copyright © EMC Corporation. Todos os direitos reservados. Copyright © Flexera Software. Todos os direitos reservados. Copyright © Jinfonet Software. Todos os direitos reservados. Copyright © Apple Inc. Todos os direitos reservados. Copyright © Telerik Inc. Todos os direitos reservados. Copyright © BEA Systems. Todos os direitos reservados. Copyright © PDFlib GmbH. Todos os direitos reservados. Copyright © Orientation in Objects GmbH. Todos os direitos reservados. Copyright © Tanuki Software, Ltd. Todos os direitos reservados. Copyright © Ricebridge. Todos os direitos reservados. Copyright © Sencha, Inc. Todos os direitos reservados. Copyright © Scalable Systems, Inc. Todos os direitos reservados. Copyright © jqWidgets. Todos os direitos reservados. Copyright © Tableau Software, Inc. Todos os direitos reservados. Copyright © MaxMind, Inc. Todos os direitos reservados. Copyright © TMate Software s.r.o. Todos os direitos reservados. Copyright © MapR Technologies Inc. Todos os direitos reservados. Copyright © Amazon Corporate LLC. Todos os direitos reservados. Copyright © Highsoft. Todos os direitos reservados. Copyright © Python Software Foundation. Todos os direitos reservados. Copyright © BeOpen.com. Todos os direitos reservados. Copyright © CNRI. Todos os direitos reservados.

Este produto inclui software desenvolvido pela Apache Software Foundation (<http://www.apache.org/>) e/ou outros softwares licenciados nas várias versões da Licença Apache (a "Licença"). Você pode obter uma cópia dessas Licenças em <http://www.apache.org/licenses/>. A menos que exigido pela legislação aplicável ou concordado por escrito, o software distribuído em conformidade com estas Licenças é fornecido "NO ESTADO EM QUE SE ENCONTRA", SEM GARANTIA OU CONDIÇÃO DE QUALQUER TIPO, seja expressa ou implícita. Consulte as Licenças para conhecer as limitações e as permissões que regulam o idioma específico de acordo com as Licenças.

Este produto inclui software desenvolvido pela Mozilla (<http://www.mozilla.org/>), direitos autorais de software de The JBoss Group, LLC; todos os direitos reservados; software copyright © 1999-2006 de Bruno Lowagie e Paulo Soares e outros produtos de software licenciados sob a Licença Pública GNU Lesser General Public License Agreement, que pode ser encontrada em <http://www.gnu.org/licenses/lgpl.html>. Os materiais são fornecidos gratuitamente pela Informatica, no estado em que se encontram, sem garantia de qualquer tipo, explícita nem implícita, incluindo, mas não limitando-se, as garantias implicadas de comerciabilidade e adequação a um determinado propósito.

O produto inclui software ACE(TM) e TAO(TM) com copyright de Douglas C. Schmidt e seu grupo de pesquisa na Washington University, University of California, Irvine e Vanderbilt University, Copyright (©) 1993-2006, todos os direitos reservados.

Este produto inclui o software desenvolvido pelo OpenSSL Project para ser usado no kit de ferramentas OpenSSL (copyright The OpenSSL Project. Todos os direitos reservados) e a redistribuição deste software está sujeita aos termos disponíveis em <http://www.openssl.org> e <http://www.openssl.org/source/license.html>.

Este produto inclui o software Curl com o Copyright 1996-2013, Daniel Stenberg, <daniel@haxx.se>. Todos os direitos reservados. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://curl.haxx.se/docs/copyright.html>. É permitido usar, copiar, modificar e distribuir este software com qualquer objetivo, com ou sem taxa, desde que a nota de direitos autorais acima e esta nota de permissão apareçam em todas as cópias.

O produto inclui software copyright 2001-2005 (©) MetaStuff, Ltd. Todos os direitos reservados. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://www.dom4j.org/license.html>.

O produto inclui o copyright de software © 2004-2007, The Dojo Foundation. Todos os direitos reservados. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://dojotoolkit.org/license>.

Este produto inclui o software ICU com o copyright International Business Machines Corporation e outros. Todos os direitos reservados. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://source.icu-project.org/repos/icu/icu/trunk/license.html>.

Este produto inclui o copyright de software © 1996-2006 Per Bothner. Todos os direitos reservados. O direito de usar tais materiais é estabelecido na licença que pode ser encontrada em <http://www.gnu.org/software/kawa/Software-License.html>.

Este produto inclui o software OSSP UUID com Copyright © 2002 Ralf S. Engelschall, Copyright © 2002 e OSSP Project Copyright © 2002 Cable & Wireless Deutschland. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://www.opensource.org/licenses/mit-license.php>.

Este produto inclui software desenvolvido pela Boost (<http://www.boost.org/>) ou sob a licença de software Boost. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em http://www.boost.org/LICENSE_1_0.txt.

Este produto inclui software copyright © 1997-2007 University of Cambridge. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://www.pcre.org/license.txt>.

Este produto inclui o copyright de software © 2007 The Eclipse Foundation. Todos os direitos reservados. As permissões e as limitações relativas a este software estão sujeitas aos termos disponíveis em <http://www.eclipse.org/org/documents/epl-v10.php> e em <http://www.eclipse.org/org/documents/edl-v10.php>.

Este produto inclui softwares licenciados de acordo com os termos disponíveis em <http://www.tcl.tk/software/tcltk/license.html>, <http://www.bosrup.com/web/overlib/?License>, <http://www.stlport.org/doc/license.html>, <http://asm.ow2.org/license.html>, <http://www.cryptix.org/LICENSE.TXT>, <http://hsqldb.org/web/hsqldbLicense.html>, <http://httpunit.sourceforge.net/doc/license.html>, <http://jung.sourceforge.net/license.txt>, http://www.gzip.org/zlib/zlib_license.html, <http://www.openldap.org/software/release/license.html>, <http://www.libssh2.org>, <http://slf4j.org/license.html>, <http://www.sente.ch/software/OpenSourceLicense.html>, <http://fusesource.com/downloads/license-agreements/fuse-message-broker-v-5-3-license-agreement>, <http://antlr.org/license.html>, <http://aopalliance.sourceforge.net/>, <http://www.bouncycastle.org/license.html>, <http://www.jgraph.com/jgraphdownload.html>, <http://www.jcraft.com/jsch/LICENSE.txt>, http://jotm.objectweb.org/bsd_license.html, <http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231>, <http://www.slf4j.org/license.html>, <http://nanoxml.sourceforge.net/orig/copyright.html>, <http://www.json.org/license.html>, <http://forge.ow2.org/projects/jaservice/>, <http://www.postgresql.org/about/license.html>, <http://www.sqlite.org/copyright.html>, <http://www.tcl.tk/software/tcltk/license.html>, <http://www.jaxen.org/faq.html>, <http://www.jdom.org/docs/faq.html>, <http://www.slf4j.org/license.html>, <http://www.iodbc.org/dataspace/iodbc/wiki/iodbc/License>, <http://www.keplerproject.org/md5/license.html>, <http://www.toedter.com/en/jcalendar/license.html>, <http://www.edankert.com/bounce/index.html>, <http://www.net-snmp.org/about/license.html>, <http://www.openmdx.org/#FAQ>, http://www.php.net/license/3_01.txt, <http://srp.stanford.edu/license.txt>, <http://www.schneier.com/blowfish.html>, <http://www.jmock.org/license.html>, <http://xsom.java.net>, <http://benalman.com/about/license/>, <https://github.com/CreateJS/EaselJS/blob/master/src/easeljs/display/Bitmap.js>, <http://www.h2database.com/html/license.html#summary>, <http://jsoncpp.sourceforge.net/LICENSE>, <http://jdbc.postgresql.org/license.html>, <http://protobuf.googlecode.com/svn/trunk/src/google/protobuf/descriptor.proto>, <https://github.com/rantav/hector/blob/master/LICENSE>, <http://web.mit.edu/Kerberos/krb5-current/doc/mitK5license.html>, <http://jibx.sourceforge.net/jibx-license.html>, <https://github.com/lyokato/libgeohash/blob/master/LICENSE>, <https://github.com/hjiang/jsonxx/blob/master/LICENSE>, <https://code.google.com/p/lz4/>, <https://github.com/jedisct1/libsodium/blob/master/LICENSE>, <http://one-jar.sourceforge.net/index.php?page=documents&file=license>, <https://github.com/EsotericSoftware/kryo/blob/master/license.txt>, <http://www.scala-lang.org/license.html>, <https://github.com/tinkerpop/blueprints/blob/master/LICENSE.txt>, <http://gee.cs.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html>, <https://aws.amazon.com/asl/>, <https://github.com/twbs/bootstrap/blob/master/LICENSE>, <https://sourceforge.net/p/xmlunit/code/HEAD/tree/trunk/LICENSE.txt>.

Este produto inclui software licenciado de acordo com a Academic Free License (<http://www.opensource.org/licenses/afl-3.0.php>), a Common Development and Distribution License (<http://www.opensource.org/licenses/cddl1.php>), a Common Public License (<http://www.opensource.org/licenses/cpl1.0.php>), a Sun Binary Code License Agreement Supplemental License Terms, a BSD License (<http://www.opensource.org/licenses/bsd-license.php>), a nova BSD License (<http://opensource.org/licenses/BSD-3-Clause>), a MIT License (<http://www.opensource.org/licenses/mit-license.php>), a Artistic License (<http://www.opensource.org/licenses/artistic-license-1.0>) e a Initial Developer's Public License Version 1.0 (<http://www.firebirdsql.org/en/initial-developer-s-public-license-version-1-0/>).

Este produto inclui copyright do software © 2003-2006 Joe Walnes, 2006-2007 XStream Committers. Todos os direitos reservados. Permissões e limitações relativas a este software estão sujeitas aos termos disponíveis em <http://xstream.codehaus.org/license.html>. Este produto inclui software desenvolvido pelo Indiana University Extreme! Lab. Para obter mais informações, visite <http://www.extreme.indiana.edu/>.

Este produto inclui software Copyright © 2013 Frank Balluffi e Markus Moeller. Todos os direitos reservados. As permissões e limitações relativas a este software estão sujeitas aos termos da licença MIT.

Consulte as patentes em <https://www.informatica.com/legal/patents.html>.

ISENÇÃO DE RESPONSABILIDADE: a Informatica LLC fornece esta documentação no estado em que se encontra, sem garantia de qualquer tipo, expressa ou implícita, incluindo, mas não limitando-se, as garantias implícitas de não infração, comercialização ou uso para um determinado propósito. A Informatica LLC não garante que este software ou documentação não contenha erros. As informações fornecidas neste software ou documentação podem incluir imprecisões técnicas ou erros tipográficos. As informações deste software e documentação estão sujeitas a alterações a qualquer momento sem aviso prévio.

AVISOS

Este produto da Informatica (o "Software") traz determinados drivers (os "drivers da DataDirect") da DataDirect Technologies, uma empresa em funcionamento da Progress Software Corporation ("DataDirect"), que estão sujeitos aos seguintes termos e condições:

1. OS DRIVERS DA DATADIRECT SÃO FORNECIDOS NO ESTADO EM QUE SE ENCONTRAM, SEM GARANTIA DE QUALQUER TIPO, EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO LIMITANDO-SE, AS GARANTIAS IMPLÍCITAS DE COMERCIALIZAÇÃO, ADEQUAÇÃO A UMA FINALIDADE ESPECÍFICA E NÃO INFRAÇÃO.
2. EM NENHUM CASO, A DATADIRECT OU SEUS FORNECEDORES TERCEIRIZADOS SERÃO RESPONSÁVEIS, EM RELAÇÃO AO CLIENTE FINAL, POR QUALQUER DANOS DIRETOS, INDIRETOS, INCIDENTAIS, ESPECIAIS, CONSEQUENCIAIS OU DEMAIS QUE POSSAM ADVIR DO USO DE DRIVERS ODBC, SENDO OU NÃO ANTERIORMENTE INFORMADOS DAS POSSIBILIDADES DE TAIS DANOS. ESTAS LIMITAÇÕES SE APLICAM A TODAS AS CAUSAS DE AÇÃO, INCLUINDO, SEM LIMITAÇÕES, QUEBRA DE CONTRATO, QUEBRA DE GARANTIA, NEGLIGÊNCIA, RESPONSABILIDADE RIGOROSA, DETURPAÇÃO E OUTROS ATOS ILÍCITOS.

Data da Publicação: 2021-04-28

Conteúdo

Prefácio.....	19
Recursos da Informatica.	19
Rede da Informatica.	19
Base de Dados de Conhecimento da Informatica.	19
Documentação da Informatica.	20
Matrizes de Disponibilidade de Produto da Informatica.	20
Informatica Velocity.	20
Informatica Marketplace.	20
Suporte Global a Clientes da Informatica.	20
 Capítulo 1: Introdução à Transformação de Dados.....	21
Visão Geral da Data Transformation.	21
Arquitetura do Processo de Data Transformation.	22
Componentes da Data Transformation.	23
 Capítulo 2: Transformação do Processador de Dados.....	24
Visão geral da Transformação do Processador de Dados.	24
Exibições da Transformação do Processador de Dados.	25
Portas da Transformação do Processador de Dados.	26
Portas de Entrada da Transformação do Processador de Dados.	26
Portas de Saída de Transformação do Processador de Dados.	27
Portas de Passagem.	28
Componente de Inicialização.	28
Referências.	29
Configurações da Transformação do Processador de Dados.	30
Codificação de Caracteres.	30
Regras e Diretrizes da Codificação de Caracteres.	32
Configurações de Saída.	32
Configurações de Processamento.	34
Configurações de XMap.	35
Configuração de Saída XML.	35
Eventos.	37
Tipos de Eventos.	37
Exibição de Eventos do Processador de Dados.	38
Logs.	38
Log de Eventos de Tempo de Design.	39
Log de Eventos de Tempo de Execução.	39
Exibindo um Log de Eventos na Exibição de Eventos do Processador de Dados.	40
Log do Usuário.	40
Desenvolvimento da Transformação do Processador de Dados.	41

Criar a Transformação do Processador de Dados.	41
Selecionar os Objetos de Esquema	42
Criar Objetos em uma Transformação do Processador de Dados em Branco.	42
Criar as Portas.	44
Testando a Transformação.	45
Exportação e Importação da Transformação do Processador de Dados.	45
Exportando a Transformação do Processador de Dados como um Serviço.	45
Importando vários serviços do Data Transformation.	46
Importando um Serviço de Data Transformation	46
Exportando um Mapeamento com uma Transformação do Processador de Dados para o PowerCenter.	47
Transformação de Processador de Dados Validação.	48
Usando um Mecanismo do Data Transformation com Velocidade Aprimorada para Validações VRL.	48
Transformação de Processador de Dados em um ambiente não nativo.	48
Capítulo 3: Formatos de Entrada e Saída do Assistente.....	50
Visão Geral dos Formatos de Entrada e de Saída do Assistente.	50
Avro.	51
Entrada Avro e Leitor de Arquivos Complexos.	51
Compactação de Dados Avro com o Codec Snappy.	52
Configurar uma Transformação com a Entrada Avro.	52
Configurar uma Transformação com a Saída Avro.	55
Biblioteca de Processamento COBOL.	56
Criando uma Transformação para COBOL.	56
Definições de Dados COBOL.	56
Procedimentos de Teste.	57
Editando uma Transformação para COBOL.	58
Otimizando o processamento do arquivo COBOL de grande volume no ambiente do Hadoop.	58
JSON.	59
Esquemas JSON.	59
Exemplo de Esquema JSON.	60
Criando uma Transformação com JSON.	61
Parquet.	62
Criando uma Transformação com Entrada ou Saída Parquet.	62
Configurar o leitor de arquivos complexos da entrada do parquet.	63
Configurar uma transformação com saída Parquet.	63
XML.	64
Criando uma Transformação que Transforma XML.	64
Capítulo 4: Entrada e Saída Relacionais.....	66
Visão Geral de Entradas e Saídas Relacionais.	66
Entrada Relacional.	66

Configuração de Portas de Entrada Relacionais.	67
Diretrizes para Vincular Portas de Entrada.	68
Definir Portas Relacionais de Entrada com a Exibição Visão Geral.	68
Portas Clustering_Key.	69
Entrada Relacional Normalizada.	70
Entrada Relacional Dinamizada.	71
Entrada Relacional Desnormalizada.	71
Mapeando Portas Relacionais para Nós Hierárquicos.	72
Saída Relacional.	73
Configuração de Portas de Saída Relacionais.	73
Definir Portas Relacionais de Saída com a Exibição Visão Geral.	74
Saída Relacional Normalizada.	75
Saída Relacional Dinâmica.	75
Saída Relacional Desordenada.	76
Capítulo 5: Usando o editor do IntelliScript.	77
Visão geral do Editor do IntelliScript.	77
Criando um Script.	77
Abrindo um editor do IntelliScript.	78
IntelliScript e Visualizador de Dados.	78
Localizando âncoras.	78
Componentes e propriedades.	78
Propriedades básicas e avançadas.	79
Procedimentos de edição.	79
Procedimento básico para edição.	79
Copiar e colar.	80
Arrastar e soltar.	80
Localizar e substituir.	80
Inserindo componentes no IntelliScript.	80
Editando as propriedades de um componente.	80
Inserindo tabulações, novas linhas e outros caracteres especiais.	81
Definindo um componente global.	81
Exibindo a ajuda sobre um componente.	82
Ícones do IntelliScript.	82
Salvando o IntelliScript.	83
Menus do editor do IntelliScript.	83
Capítulo 6: XMap.	85
Visão Geral do XMap.	85
Esquemas do XMap.	86
Instruções de Mapeamento.	87
Tipos de Instrução de Mapeamento.	87
Instruções de Mapa.	89

Instruções de Grupo.	90
Instruções de Grupo de Repetição.	92
Instruções de Roteador.	94
Instruções de Opção.	95
Instruções Padrão.	97
Instruções Executar XMap.	98
Instrução RunMaplet.	99
Instrução MapletInput.	100
Instrução MapletOutput.	101
Criando um XMap.	102
Usando a Grade do Editor XMap.	102
Criando Instruções de Mapeamento.	103
Interface de Grade de Instruções de Mapeamento.	103
Expressões XPath.	104
Predicados.	105
Editor de Expressão do XPath.	108
Funções do Processador de Dados.	109
Exemplo de Expressões do XPath.	110
Criando uma Expressão.	111
Variáveis XMap.	111
Criando uma Variável no Editor de XMap.	112
Exemplo de XMap.	112
Exemplo de Esquema de Entrada XML.	112
Exemplo de Esquema de Saída XML.	113
Dados de Entrada XML.	114
Hierarquias de XML de Entrada e Saída.	115
Instruções de Mapeamento no Exemplo.	115
Exemplo de Instruções de Grupo.	117
Capítulo 7: Bibliotecas.	118
Visão Geral de Bibliotecas.	118
Estrutura da Biblioteca.	119
Propriedades do Elemento.	119
Gerenciamento de Biblioteca.	119
Editar as Bibliotecas com o Editor de Biblioteca.	120
Adicionando um Elemento com o Editor de Biblioteca.	121
Editando as Propriedades do Elemento com o Editor de Biblioteca.	121
Testando uma Biblioteca.	121
Gerando os Objetos de Biblioteca.	122
Descartando os Objetos de Biblioteca.	122
Editar as Bibliotecas com o Editor do IntelliScript.	122

Capítulo 8: Objeto de Esquema.....	123
Visão Geral de Objetos de Esquema.	123
Arquivos de Esquema.	123
Exibição Visão Geral de Objetos de Esquema.	124
Exibição Esquema de Objetos de Esquema.	125
Propriedades de Espaço de Nome.	126
Propriedades do Elemento.	126
Propriedades de Tipos Simples.	128
Propriedades de Tipos Complexos	129
Propriedades do Atributo.	130
Exibição Avançado de Objetos de Esquema.	130
Criando um objeto de esquema.	131
Atualizações de Esquema.	132
Sincronização de Esquema.	132
Edições de Arquivos de Esquema.	133
 Capítulo 9: Interface da Linha de Comando.....	 136
Visão Geral da Interface da Linha de Comando.	136
CM_console.	136
 Capítulo 10: Scripts.....	 139
Visão Geral de Scripts.	139
Componentes de Script.	140
Tipos de Componentes.	140
Nomes de Componentes.	141
Adicionando um Componente Global.	141
Adicionando um Componente Local.	141
Propriedades de um Componente de Script.	142
Propriedades Simples.	142
Propriedades Avançadas.	142
Valores de Propriedade de Componente.	143
Componentes de Inicialização de Script.	144
Definindo o Componente de Inicialização com o Editor do IntelliScript.	144
Origens de Exemplo.	144
Exemplo de Realce de Origem.	145
Definindo uma Origem de Exemplo no Editor do IntelliScript.	145
Exibindo uma Origem de Exemplo.	146
Editor do IntelliScript.	146
Validar um script.	147
Scripts de Amostra.	147
Importando uma Amostra de Script.	148

Capítulo 11: Analisadores.....	150
Visão Geral de Analisadores.	150
Analisadores Independentes de Plataforma.	150
Marcadores de Nova Linha.	150
Caminhos de Arquivo.	150
Referência de Componente de Analisador.	151
Analisador.	151
 Capítulo 12: Portas de Script.....	 154
Visão Geral de Portas de Script.	154
Referência de Componentes de Porta de Script.	154
AdditionalInputPort.	154
AdditionalOutputPort.	156
DocList.	159
FileSearch.	159
InputPort.	160
LocalFile.	160
OutputPort.	160
Texto.	160
URL.	161
 Capítulo 13: Processadores de Documentos.....	 162
Visão Geral de Processadores de Documentos.	162
Definindo um Processador de Documentos.	162
Exibição da Saída do Processador de Documento.	163
Referência de Componente do Processador de Documentos.	163
AsnToXml.	163
ExcelToDataXml.	164
ExcelToXml.	164
ExcelToXml_03_07_10.	165
ExpandFrameSet.	165
ExternalJavaPreProcessor.	166
HIPAAValidator.	166
PdfFormToXml_1_00.	167
PdfToTxt_3_02.	167
PdfToTxt_4.	168
PowerpointToTextML.	168
ProcessByTransformers.	168
ProcessorPipeline.	169
RtfToTextML.	169
WordToXml.	169
XmlToDocument_372.	169

XmlToDocument_45.	170
XmlToExcel.	171
XmlToXlsx.	171
Esquema XML TextML.	172
Editor de Configuração de Tabela PdfToTxt_4.	173
Opções do Editor.	174
Exemplo de Conversão de PDF.	175
Capítulo 14: Formatos.	177
Visão Geral de Formatos.	177
Propriedades do Formato Padrão.	178
Referência de Componente de Formato.	178
BinaryFormat.	179
CustomFormat.	180
HtmlFormat.	181
RtfFormat.	182
TextFormat.	182
XmlFormat.	183
Referência de Componente de Delimitadores.	184
CommaDelimited.	185
Delimitador.	186
DelimiterHierarchy.	186
EnclosingDelimiters.	187
HL7.	187
Posicional.	187
PostScript.	188
RTF.	188
SGML.	188
SpaceDelimited.	188
TabDelimited.	189
Referência de Componentes de Pré-processador de Formato.	189
HtmlProcessor.	190
RtfProcessor.	190
Capítulo 15: Recipientes de Dados.	191
Visão Geral de Recipientes de Dados.	191
Esquemas XML.	191
Codificação de Esquema.	192
Arquivos com Esquema Incluído.	192
Espaços de Nome.	192
Conteúdo Misto.	192
Recursos de Esquema sem Suporte.	192
Precisão de Dados Numéricos.	193

Usando um Esquema para Mapear Âncoras.	194
Representação de Recipientes de Dados do IntelliScript.	194
Mapeando Conteúdo Misto.	194
Mapeando Tipos XSL.	195
Gerando um XML Válido.	195
Função de Esquemas na Análise.	195
Função de Esquemas na Serialização e no Mapeamento.	196
Variáveis.	197
Criando uma Variável Definida pelo Usuário.	197
Variáveis de Sistema.	197
Âncoras de Mapeamento para Variáveis.	200
Usando Variáveis em Ações.	200
Inicializando Variáveis em Tempo de Execução.	200
Referência de Componentes de Variável.	200
Variável.	201
Recipientes de Dados de Ocorrência Múltipla.	201
Atributos.	202
Indexação.	202
Destruindo as Ocorrências.	202
Capítulo 16: Âncoras.	204
Visão Geral de Âncoras.	204
Âncoras de Marcador e Conteúdo.	204
Outros Tipos de Âncoras.	204
Como Âncoras e Delimitadores Trabalham Juntos.	205
Mapeando Âncoras de Conteúdo para Recipientes de Dados.	205
Mapeando para Variáveis.	206
Mapeando para Recipientes de Dados de Ocorrência Múltipla.	206
Mapeando para Elementos de Conteúdo Misto.	206
Definindo Âncoras.	206
Onde Definir Âncoras.	207
Sequência de Âncoras.	207
Adicionando uma Âncora Marcador ou Conteúdo.	207
Definindo uma Âncora.	208
Propriedades de Âncoras Padrão.	208
Como um Analisador Procura Âncoras.	209
Fases de Pesquisa.	209
Escopo da Pesquisa e Critérios de Pesquisa.	210
Ajustar a Fase de Pesquisa.	211
Ajustando o Escopo de Pesquisa.	211
Ajustando os Critérios de Pesquisa.	213
Usando Tipos de Dados para Restringir os Critérios de Pesquisa.	214
Âncoras que Contêm Âncoras Aninhadas.	215

Referência do Componente de Âncora.	216
Alternativas.	216
Conteúdo.	218
DelimitedSections.	221
EmbeddedParser.	224
EnclosedGroup.	226
ExtractRecord.	228
FindReplaceAnchor.	228
Grupo.	231
Marcador.	234
RepeatingGroup.	236
StructureDefinition.	241
Referência de Componentes de Pesquisador.	245
AttributeSearch.	245
LearnByExample.	246
NewlineSearch.	246
OffsetSearch.	246
PatternSearch.	247
SegmentSearch.	247
TextSearch.	248
TypeSearch.	249
Referência do Subcomponente de Âncora.	249
AllStructure.	249
AllStructureLocal.	250
ChoiceStructure.	251
ChoiceStructureLocal.	251
Conectar.	252
EmbeddedStructure.	253
RecordStructure.	253
RecordStructureLocal.	254
SequenceStructure.	255
SequenceStructureLocal.	255
Capítulo 17: Transformadores.....	257
Visão Geral de Transformadores.	257
Definindo Transformadores.	257
Usando Transformadores em Âncoras.	257
Sequências de Transformadores.	258
Transformadores padrão.	258
Usando Transformadores como Processadores de Documento.	258
Usando Transformadores em Âncoras de Serialização.	259
Usando Transformadores em Ações.	259
Propriedades de Transformadores Padrão.	259

Referência de Componente Transformador.	260
AbsURL.	260
AddEmptyTagsTransformer.	260
AddString.	261
Base64Decode.	262
Base64Encode.	262
BidiConvert.	263
CDATADecode.	263
CDATAEncode.	264
ChangeCase.	265
CreateGuid.	265
CreateUUID.	265
DateFormatICU.	266
Dos96HebToAscii.	268
DynamicTable.	269
EbcdicToAscii.	269
EDIFACTValidation.	269
EncodeAsUrl.	270
Codificador.	270
FormatNumber.	271
FromFloat.	272
FromInteger.	273
FromPackDecimal.	274
FromSignedDecimal.	274
hebrewBidi.	275
HebrewDosToWindows.	275
HebrewEBCDICOldCodeToWindows.	275
hebUniToAscii.	275
hebUtf8ToAscii.	275
HtmlEntitiesToASCII.	275
HtmlProcessor.	276
InjectFP.	276
InjectString.	277
InlineTable.	277
JavaTransformer.	278
LookupTransformer.	278
NormalizeClosingTags.	280
RegularExpression.	281
RemoveMarginSpace.	283
RemoveRtfFormatting.	283
RemoveTags.	283
Substituir.	284

Redimensionar.	285
ReverseTransformer.	285
RtfProcessor.	286
RtfToASCII.	286
SubString.	286
ToFloat.	287
ToInteger.	288
ToPackDecimal.	288
TransformationStartTime.	289
TransformByParser.	290
TransformByProcessor.	291
TransformByService.	291
TransformerPipeline.	292
XMLLookupTable.	293
XSLTTransformer.	293

Capítulo 18: Ações..... 295

Visão Geral de Ações.	295
Como as Ações Funcionam.	295
Comparação entre Ações e Transformadores.	296
Definindo Ações.	296
Propriedades de Ação Padrão.	296
Referência do Componente da Ação.	297
AddEventAction.	297
AggregateValues.	298
AppendListItems.	300
AppendValues.	301
CalculateValue.	303
CombineValues.	305
CreateList.	306
CustomLog.	307
DateAddICU.	307
DateDiffICU.	308
DownloadFileToDataHolder.	309
DumpValues.	310
EnsureCondition.	311
ExcludeItems.	314
Mapa.	315
Notificar.	317
ResetVisitedPages.	317
RunMapper.	318
RunMapplet.	320
RunParser.	320

RunPCWebService.	322
RunSerializer.	323
RunXMap.	324
SetValue.	325
Classificar.	326
ValidateValue.	327
WriteValue.	328
XSLTMap.	330
Referência do Subcomponente da Ação.	331
OutputDataHolder.	331
OutputFile.	331
ResultFile.	332
StandardErrorLog.	332

Capítulo 19: Serializadores..... 333

Visão Geral de Serializadores.	333
Controlando como o Comando Criar Serializador Funciona.	333
Solução de Problemas do Serializador Gerado Automaticamente.	335
Criando um Serializador por meio da Edição do Script.	336
Criando um Serializador dentro de uma Ação RunSerializer.	336
Âncoras de Serialização.	336
Exemplo de Âncoras de Serialização.	337
Sequência de Âncoras de Serialização.	337
Propriedades do Serializador Padrão.	338
Referência de Componente Serializador.	338
Serializador.	338
Referência de Componente de Âncora de Serialização.	340
AlternativeSerializers.	340
ContentSerializer.	341
DelimitedSectionsSerializer.	342
EmbeddedSerializer.	344
GroupSerializer.	346
RepeatingGroupSerializer.	346
StringSerializer.	349

Capítulo 20: Mapeadores..... 350

Criando um Mapeador.	350
Componentes Aninhados Dentro de um Mapeador.	350
Exemplo de Mapeador.	351
XML de Origem.	351
XML de Saída.	351
Configuração do Mapeador.	352
Propriedades do Mapeador Padrão.	352

Referência de Componente de Mapeador.	353
Mapeador.	353
Referência do Componente de Âncora Mapeadora.	355
AlternativeMappings.	355
EmbeddedMapper.	356
GroupMapping.	357
RepeatingGroupMapping.	358
Capítulo 21: Localizadores, Chaves e Indexação.	360
Visão Geral de Localizadores, Chaves e Indexação.	360
Exemplo de Localizadores.	361
Entrada e Saída.	361
Solução Incorreta.	362
Solução Correta.	362
Exemplo de Indexação por Chave.	363
Entrada.	363
Saída.	363
Estrutura de Tópicos da Abordagem da Transformação.	363
Configuração do Mapeador.	364
Uso de Indexação.	365
Propriedades de Origem e Destino.	365
Propriedade de Origem.	366
Propriedade de Destino.	369
Localizador Padrão e Propriedades de Chaves.	371
Referência de Componentes de Localizador e Chave.	371
Chave.	371
Localizador.	374
LocatorByKey.	374
LocatorByOccurrence.	375
Capítulo 22: Streamers.	377
Visão Geral de Streamers.	377
Streamers de Texto.	378
Segmentos.	378
Segmentos Simples.	378
Segmentos Complexos.	378
Exemplo.	379
Concatenação de Cabeçalho.	379
Saída de um Streamer.	380
Usando Marcadores e Variáveis em Streamers.	380
Criando um Streamer.	380
XML Streamers.	382
Propriedades Padrão do Streamer.	384

Referência de Componente de Streamer.	385
ComplexSegment.	385
ComplexXmlSegment.	385
JsonStreamer.	386
MarkerStreamer.	387
SimpleSegment.	388
SimpleXmlSegment.	389
Streamer.	390
StreamerVariable.	391
XmlSegment.	392
XmlStreamer.	393
Referência de Subcomponente de Streamer.	394
AddHeaderModifier.	394
AddStringModifier.	395
DoNothingModifier.	395
WellFormedModifier.	395
WriteSegment.	396
 Capítulo 23: Validadores, Notificações e Tratamento de Falhas.....	398
Visão Geral de Validadores, Notificações e Tratamento de Falhas.	398
Tratamento de Falhas.	399
Usando a Propriedade Opcional para o Tratamento de Falhas.	399
Gravando uma Mensagem de Falha no Log do Usuário.	400
Validadores.	402
Propriedades do Validador Padrão.	403
Referência de Componente do Validador.	403
AlternativeValidators.	404
EDIFACTValidation.	405
Enumeração.	405
LengthEquals.	406
MaxLength.	408
MaxNumber.	408
MinLength.	409
MinNumber.	410
NumberEquals.	411
ValidateByExpression.	412
ValidateByPattern.	413
ValidateByTransformer.	414
ValidateByType.	415
ValidateDate.	416
ValidatorPipeline.	417
Notificações.	418
Referência de Componente de Notificação.	419

Notificação.	419
NotificationGroup.	419
NotificationHandler.	420
NotifyFailure.	421
Capítulo 24: Regras de Validação.	422
Visão Geral das Regras de Validação.	422
Referência do Elemento Regras de Validação.	423
Atributos do Elemento Declaração.	423
Atributos do Elemento Lista.	424
Atributos do Elemento Pesquisa.	424
Atributos do Elemento Regra.	425
Atributos do Elemento Rastreamento.	425
Atributos do Elemento Variável.	426
Editor de XPath.	426
Extensões de XPath.	426
Editar as Regras de Validação em um Editor Externo.	429
Criar um Objeto Regras de Validação.	429
Importar um Serviço do Data Transformation com Regras de Validação.	430
Capítulo 25: Componentes de Script Personalizados.	431
Visão Geral de Componentes de Script Personalizados.	431
Exemplo de Componente Personalizado.	431
Propriedades de Componentes Personalizados.	432
Desenvolvendo um Componente Personalizado.	432
Exemplo de Interface Java.	433
Exemplo de Componentes Java Personalizados.	433
Configurando um Componente Personalizado.	434
Amostras de Script que Contêm Componentes Personalizados.	435
Índice.	436

Prefácio

Use o *Guia de Usuário do Data Transformation* para aprender a projetar, configurar, testar e implantar a transformação do Processador de Dados. Este guia contém seções de referência detalhadas que documentam os componentes e propriedades da transformação.

O *Guia do Usuário do Data Transformation* foi desenvolvido para desenvolvedores, analistas e outros usuários do Data Transformation que projetam e implementam transformações. Ele pressupõe que você tenha um conhecimento básico sobre o uso do Informatica Developer. Ele também pressupõe que você conheça XML, esquemas e tenha conhecimentos básicos de programação.

Recursos da Informatica

A Informatica oferece uma variedade de recursos de produtos através da Rede da Informatica e outros portais on-line. Use os recursos para obter o máximo de seus produtos e soluções da Informatica e para aprender com outros usuários da Informatica e especialistas no assunto.

Rede da Informatica

A Rede da Informatica é a porta de entrada para muitos recursos, incluindo a Base de Dados de Conhecimento da Informatica e o Suporte Global a Clientes da Informatica. Para acessar a Rede da Informatica, visite <https://network.informatica.com>.

Como membro da Rede da Informatica, você tem as seguintes opções:

- Pesquisar por recursos do produto na Base de Dados de Conhecimento.
- Visualizar informações sobre disponibilidade de produtos.
- Criar e revisar seus casos de suporte.
- Encontrar a sua Rede de Grupo de Usuários da Informatica local e colaborar com seus colegas.

Base de Dados de Conhecimento da Informatica

Use a Base de Dados de Conhecimento da Informatica para encontrar recursos de produtos, como artigos de instruções, práticas recomendadas, tutoriais em vídeo e respostas a perguntas frequentes.

Para pesquisar na Base de Dados de Conhecimento, visite <https://search.informatica.com>. Em caso de dúvidas, comentários ou ideias sobre a Base de Dados de Conhecimento, entre em contato com a equipe da Base de Dados de Conhecimento da Informatica em KB_Feedback@informatica.com.

Documentação da Informatica

Use o Portal de Documentação da Informatica para explorar uma extensa biblioteca de documentação para versões de produtos atuais e recentes. Para explorar o Portal de Documentação, visite <https://docs.informatica.com>.

Em caso de dúvidas, comentários ou ideias sobre a documentação do produto, entre em contato com a equipe da Documentação da Informatica em infa_documentation@informatica.com.

Matrizes de Disponibilidade de Produto da Informatica

As Matrizes de Disponibilidade de Produto (PAMs) indicam as versões dos sistemas operacionais, os bancos de dados e tipos de fontes e destinos de dados com os quais uma versão de produto é compatível. Veja as PAMs da Informatica em <https://network.informatica.com/community/informatica-network/product-availability-matrices>.

Informatica Velocity

O Informatica Velocity é uma coleção de dicas e práticas recomendadas desenvolvidas pelos Serviços Profissionais da Informatica e baseada em experiências reais de centenas de projetos de gerenciamento de dados. O Informatica Velocity representa o conhecimento coletivo dos consultores da Informatica que trabalham com organizações em todo o mundo para planejar, desenvolver, implantar e manter soluções de gerenciamento de dados bem-sucedidas.

Encontre os recursos do Informatica Velocity em <http://velocity.informatica.com>. Se você tiver dúvidas, comentários ou ideias sobre o Informatica Velocity, entre em contato com os Serviços Profissionais da Informatica em ips@informatica.com.

Informatica Marketplace

O Informatica Marketplace é um fórum onde você pode encontrar soluções que ampliam e aprimoram suas implementações da Informatica. Aproveite as centenas de soluções dos desenvolvedores e parceiros da Informatica no Marketplace para melhorar sua produtividade e agilizar o tempo de implementação em seus projetos. Encontre o Informatica Marketplace em <https://marketplace.informatica.com>.

Suporte Global a Clientes da Informatica

Você pode entrar em contato com um Centro de Suporte Global por telefone ou por meio da Rede da Informatica.

Para descobrir o número de telefone local do Suporte Global a Clientes da Informatica, visite o site da Informatica no seguinte link: <https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>.

Para encontrar recursos de suporte on-line na Rede da Informatica, visite <https://network.informatica.com> e selecione a opção eSupport.

CAPÍTULO 1

Introdução à Transformação de Dados

Este capítulo inclui os seguintes tópicos:

- [Visão Geral da Data Transformation, 21](#)
- [Arquitetura do Processo de Data Transformation, 22](#)
- [Componentes da Data Transformation, 23](#)

Visão Geral da Data Transformation

A Data Transformation é um aplicativo que processa arquivos complexos, como formatos de mensagens, páginas HTML e documentos PDF. A Data Transformation também transforma formatos estruturados, como ACORD, HIPAA, HL7, EDI-X12, EDIFACT, e SWIFT.

A Data Transformation é instalada por padrão quando você instala o Informatica Developer (a Developer tool). Você pode definir uma transformação do Processador de Dados para transformar arquivos complexos em um mapeamento. Quando você executa um mapeamento com a transformação do Processador de Dados, o Serviço de Integração de Dados chama o Mecanismo de Data Transformation para processar os dados.

O aplicativo Data Transformation tem os seguintes elementos:

Transformação do Processador de Dados

Uma transformação que processa os arquivos complexos em um mapeamento. Defina um objeto Script, XMap, Biblioteca ou Regras de Validação no Data Transformation na Developer tool para processar os dados. Você pode incluir a transformação em um mapeamento da serviço de dados SQL, serviço da Web ou perfil de mapeamento.

Serviço de Data Transformation

Um conjunto de objetos da Data Transformation que você pode exportar da transformação do Processador de Dados e executar de forma autônoma. Você exporta um serviço para um repositório da Data Transformation e executa o serviço de lá.

Repositório de Data Transformation

Um diretório que armazena os serviços executáveis que você exportar de uma transformação do Processador de Dados. O nome do diretório do repositório é ServiceDB.

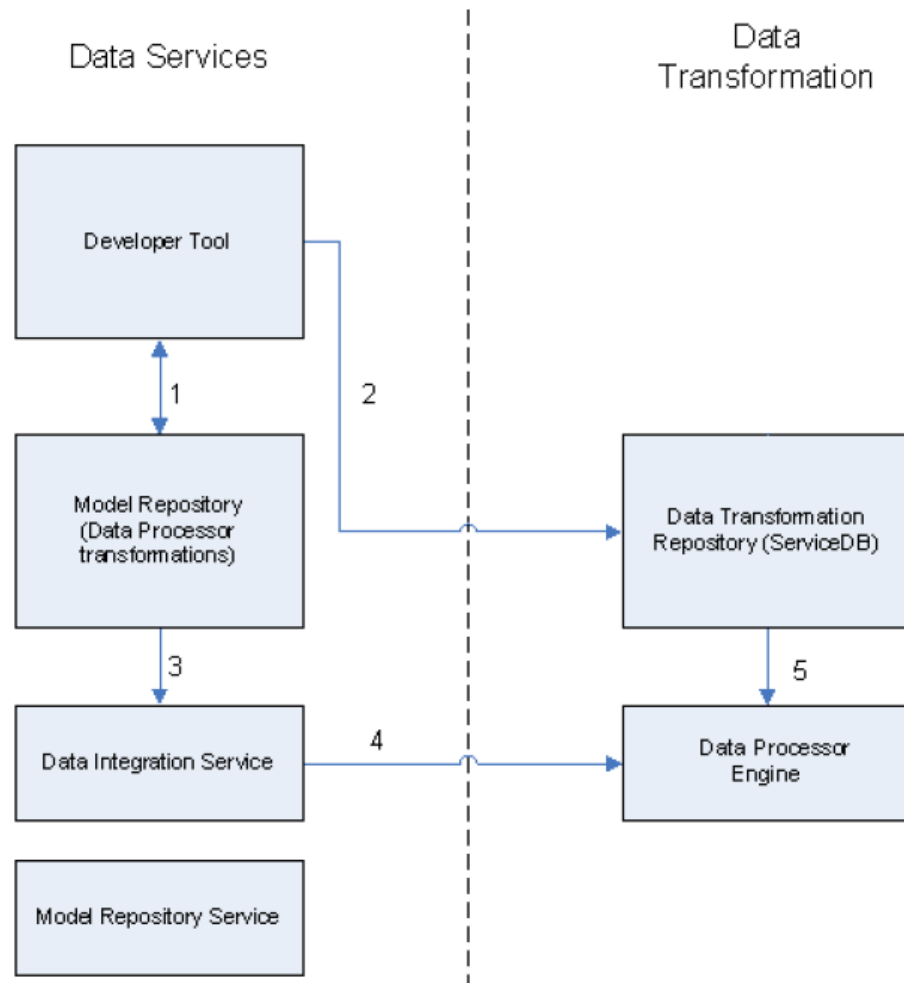
Mecanismo do Processador de Dados

Um processador que executa objetos na transformação do Processador de Dados ou serviços que você cria na Developer tool.

Arquitetura do Processo de Data Transformation

Você deve instalar a Data Transformation para configurar e executar uma transformação do Processador de Dados na ferramenta Developer. A transformação do Processador de Dados pode conter vários objetos de Scripts ou XMap para transformar arquivos complexos. O Mecanismo de Data Transformation executa Scripts, Bibliotecas ou XMaps para transformar os dados. É possível usar uma transformação do Processador de Dados em um serviço de dados, serviço da Web ou perfil.

A seguinte figura mostra os componentes no aplicativo Data Transformation e os componentes que você utiliza para criar a mesma funcionalidade na ferramenta Developer:



1. Crie uma transformação do Processador de Dados na ferramenta Developer. Salve a transformação no repositório do Modelo.

2. Exporte a transformação do Processador de Dados como um serviço de Data Transformation. Exporte o serviço para o repositório de Data Transformation. Você pode executar o serviço a partir do repositório.
3. É possível implantar um aplicativo que contém uma transformação do Processador de Dados em um Serviço de Integração de Dados.
4. O Serviço de Integração de Dados executa o aplicativo e chama o Mecanismo do Processador de Dados para processar a lógica de transformação.
5. O Mecanismo do Processador de Dados também executa serviços a partir do repositório de Data Transformation.

Componentes da Data Transformation

Ao definir um serviço de Data Transformation ou uma transformação do Processador de Dados, você pode combinar vários componentes para transformar os dados.

A Data Transformation tem os seguintes tipos de componentes que transformam dados:

Biblioteca

Transforma uma entrada de tipo de mensagem de indústria em outros formatos.

Mapeador

Converte um documento de origem XML em outro documento XML.

Analizador

Converte documentos de origem em XML. A entrada pode ter qualquer formato. A saída de um Analizador é XML.

Serializador

Converte um arquivo XML em outro documento. A saída pode ser de qualquer formato.

Streamer

Divide documentos de entrada grandes, como fluxos de dados de multigigabytes, em segmentos. O Streamer divide os documentos com várias mensagens ou vários registros.

Transformador

Modifica dados em qualquer formato. Adiciona, remove, converte ou altera o texto. Use Transformadores com um Analizador, um Mapeador ou um Serializador. Você também pode executar um Transformador como componente autônomo.

XMap

Converte uma origem hierárquica em outra estrutura hierárquica. O XMap tem a mesma funcionalidade que o Mapeador, mas você pode usar uma grade na Developer tool para definir o mapeamento.

CAPÍTULO 2

Transformação do Processador de Dados

Este capítulo inclui os seguintes tópicos:

- [Visão geral da Transformação do Processador de Dados, 24](#)
- [Exibições da Transformação do Processador de Dados, 25](#)
- [Portas da Transformação do Processador de Dados, 26](#)
- [Componente de Inicialização, 28](#)
- [Referências, 29](#)
- [Configurações da Transformação do Processador de Dados, 30](#)
- [Eventos, 37](#)
- [Logs, 38](#)
- [Desenvolvimento da Transformação do Processador de Dados, 41](#)
- [Exportação e Importação da Transformação do Processador de Dados, 45](#)
- [Transformação de Processador de Dados Validação, 48](#)
- [Transformação de Processador de Dados em um ambiente não nativo, 48](#)

Visão geral da Transformação do Processador de Dados

A transformação do Processador de Dados processa formatos de arquivo não estruturados e semi-estruturados em um mapeamento. Configure a transformação para processar formatos de mensagens, páginas HTML, XML, JSON e documentos PDF. Você também pode converter formatos estruturados como ACORD, HIPAA, HL7, EDI-X12, EDIFACT e SWIFT.

Um mapeamento usa uma transformação do Processador de Dados para alterar os documentos de um formato para outro. A transformação do Processador de Dados processa arquivos de qualquer formato em um mapeamento. Quando cria uma transformação do Processador de Dados, você define os componentes que convertem os dados.

Uma transformação do Processador de Dados pode conter vários componentes para processar dados. Cada componente pode conter outros componentes.

Por exemplo, você poderá receber faturas do cliente em arquivos do Microsoft Word. Configure uma transformação do Processador de Dados para analisar os dados de cada arquivo do Word. Extraia os dados do cliente para uma tabela de Cliente. Extraia informações sobre o pedido para uma tabela de Pedidos.

Quando cria uma transformação do Processador de Dados, você define um XMap, um Script ou uma Biblioteca. Um XMap converte um arquivo hierárquico de entrada em um arquivo hierárquico de saída de outra estrutura. Uma Biblioteca transforma um tipo de mensagem de indústria em um documento XML com uma estrutura hierárquica, ou de XML em um formato padrão da indústria. Um Script pode analisar documentos de origem em formato hierárquico, converter do formato hierárquico em outros formatos de arquivo ou mapear um documento hierárquico para outro formato hierárquico.

Defina Scripts na transformação do Processador de Dados no Editor do IntelliScript. Você pode definir os seguintes tipos de Scripts:

- **Analisador.** Converte documentos de origem em XML. A saída de um Analisador é sempre XML. A entrada pode ter qualquer formato, como texto, HTML, Word, PDF ou HL7.
- **Serializador.** Converte um arquivo XML em um documento de saída de qualquer formato. A saída de um Serializador pode ter qualquer formato, como documento de texto, documento HTML ou PDF.
- **Mapeador.** Converte um documento de origem XML em outra estrutura ou esquema XML. Você pode transformar os mesmos documentos XML como em um XMap.
- **Transformador.** Modifica os dados em qualquer formato. Adiciona, remove, converte ou altera o texto. Use Transformadores com um Analisador, um Mapeador ou um Serializador. Também você pode executar um Transformador como um componente autônomo.
- **Streamer.** Divide grandes documentos de entrada, como fluxos de dados multigigabytes, em segmentos. O Streamer processa documentos que contêm várias mensagens ou registros, como arquivos HIPAA ou EDI.

Exibições da Transformação do Processador de Dados

A transformação do Processador de Dados tem várias exibições que você pode acessar ao configurar essa transformação e executá-la na ferramenta Developer.

Algumas das exibições da transformação do Processador de Dados não aparecem na ferramenta Developer por padrão. Para alterar as exibições da transformação, clique em **Janela > Mostrar Exibição > Outros > Informatica**. Selecione as exibições que você deseja ver.

A transformação do Processador de Dados tem as seguintes exibições fixas:

Exibição Visão Geral

Configure portas e defina o componente de inicialização.

Exibição de referência

Adicione ou remova esquemas da transformação.

Exibição de Configurações

Defina configurações de transformação para codificação, controle de saída e geração de XML.

Exibição Objetos

Adicione, modifique ou exclua objetos de Script, XMap e Biblioteca da transformação.

Também é possível acessar as seguintes exibições para a transformação do Processador de Dados:

Exibição Origem Hexadecimal do Processador de Dados

Mostra um documento de entrada em formato hexadecimal.

Exibição de Eventos do Processador de Dados

Mostra informações sobre eventos que ocorrem quando você executa a transformação na ferramenta Developer. Mostra eventos de inicialização, execução e resumo.

Exibição Ajuda de Scripts

Mostra a ajuda contextual para o editor de Script.

Exibição Visualizador de Dados

Mostra dados de entrada de exemplo, executa a transformação e exibe os resultados da saída.

Portas da Transformação do Processador de Dados

Defina as portas da transformação do Processador de Dados na exibição **Visão Geral** da transformação.

Uma transformação do Processador de Dados pode ler dados de um arquivo, um buffer ou um buffer transmitidos de um leitor de arquivos complexos. Você pode usar um leitor de arquivo simples como um buffer para ler um arquivo inteiro de uma vez. Você também pode ler um arquivo de entrada a partir de um banco de dados.

Uma transformação do Processador de Dados pode ler dados de um arquivo, um buffer ou um buffer transmitidos de um leitor de arquivos complexos. Você pode usar um leitor de arquivo simples como um buffer para ler um arquivo inteiro de uma vez. Você também pode ler um arquivo de entrada a partir de um banco de dados.

As portas de saída que você cria dependem de se você deseja retornar uma string, arquivos complexos ou linhas de dados relacionais a partir da transformação.

Portas de Entrada da Transformação do Processador de Dados

Quando você cria uma transformação do Processador de Dados, a ferramenta Developer cria uma porta de entrada padrão. Quando você define uma porta de entrada adicional em um componente de inicialização do Script, a ferramenta Developer cria uma porta de entrada adicional na transformação.

O tipo de entrada determina o tipo de dados que o Serviço de Integração de Dados transmite para a transformação do Processador de Dados. O tipo da entrada determina se a entrada é um caminho de arquivo de origem ou dados.

Configure um dos seguintes tipos de entrada:

Buffer

A transformação do Processador de Dados recebe linhas de dados de origem na porta de Entrada. Use o tipo de entrada de buffer ao configurar a transformação para receber dados de um arquivo simples ou de uma transformação Informatica.

Arquivo

A transformação do Processador de Dados recebe o caminho do arquivo de origem na porta de Entrada. O componente de inicialização do Processador de Dados abre o arquivo de origem. Use o tipo de entrada de arquivo para analisar arquivos binários, como arquivos do Microsoft Excel ou do Microsoft Word. Também é possível usar o tipo de entrada de Arquivo para arquivos extensos que podem exigir muita memória do sistema para processamento com uma porta de entrada de buffer.

Parâmetro de Serviço

A transformação do Processador de Dados recebe valores a serem aplicados a variáveis nas portas de parâmetros de serviço. Quando você escolhe as variáveis para receber dados de entrada, a ferramenta Developer cria uma porta de parâmetro de serviço para cada variável.

Output_Filename

Quando você configura a porta de saída padrão para retornar um nome de arquivo em vez de dados de linha, a ferramenta Developer cria uma porta Output_Filename. É possível transmitir um nome de arquivo para a porta Output_Filename a partir de um mapeamento.

Ao definir uma porta de entrada, você pode definir a localização do arquivo de entrada de exemplo para essa porta. Um arquivo de entrada de exemplo é uma pequena amostra do arquivo de entrada. Faça referência a um arquivo de entrada de exemplo quando você criar Scripts. Também é possível usar o arquivo de entrada de exemplo ao testar a transformação na exibição **Visualizador de Dados**. Defina o arquivo de entrada de exemplo no campo **Localização de Entrada**.

Portas de Parâmetro de Serviço

Você pode criar portas de entrada que recebem os valores de variáveis. As variáveis podem conter qualquer tipo de dados, como uma string, uma data ou um número. Uma variável também pode conter uma localização para um documento de origem. Você pode fazer referência às variáveis em um componente do Processador de Dados.

Quando você cria uma porta de entrada para uma variável, a ferramenta Desenvolvedor exibe uma lista de variáveis para escolha.

Criando Portas de Parâmetro de Serviço

Você pode criar portas de entrada que recebem os valores de variáveis. Você também pode remover as portas que cria a partir de variáveis.

1. Abra a exibição **Visão Geral** da transformação do Processador de Dados.
2. Clique em **Escolher**.
A ferramenta Developer exibe uma lista de variáveis e indica quais variáveis já têm várias portas.
3. Selecione uma ou mais variáveis.
A ferramenta Developer cria uma porta de entrada de buffer para cada variável que você seleciona. Não é possível modificar a porta.
4. Para remover uma porta que você cria a partir de uma variável, desmarque a seleção de lista de variável. Quando você desmarca a seleção, a ferramenta Developer remove a porta de entrada.

Portas de Saída de Transformação do Processador de Dados

A transformação do Processador de Dados tem uma porta de saída por padrão. Se você definir portas de saída adicionais a um Script, a ferramenta Developer adicionará as portas à transformação do Processador de Dados. Você pode criar grupos de portas se configurar a transformação para retornar dados relacionais. Você também pode criar portas de parâmetros de serviço e portas de passagem.

Porta de Saída Padrão

A transformação do Processador de Dados tem uma porta de saída por padrão. Ao criar uma saída relacional, você pode definir grupos de portas de saída relacionadas em vez da porta de saída padrão.

Quando você define uma porta de saída adicional em um componente de Script, a ferramenta Developer acrescenta essa porta à transformação.

Configure um dos seguintes tipos de saída para uma porta de saída padrão:

Buffer

A transformação do Processador de Dados retorna o XML através da porta de Saída. Escolha o tipo de arquivo de Buffer ao analisar documentos ou ao mapear um XML para outros documentos XML na transformação do Processador de Dados.

Arquivo

O Serviço de Integração de Dados retorna um nome de arquivo de saída na porta de Saída para cada linha ou instância de origem. O componente da transformação do Processador de Dados grava o arquivo de saída em vez de retornar dados através das portas de saída da transformação do Processador de Dados.

Quando você seleciona uma porta de saída de Arquivo, a ferramenta Developer cria uma porta de entrada Output_Filename. É possível transmitir um nome de arquivo para a porta de nome de arquivo de Saída. A transformação do Processador de Dados cria o arquivo de saída com um nome que ela recebe nessa porta.

Se o nome do arquivo de saída estiver em branco, o Serviço de Integração de Dados retornará um erro de linha. Quando um erro ocorre, o Serviço de Integração de Dados grava um valor nulo na porta de Saída e retorna um erro de linha.

Escolha o tipo de saída de Arquivo ao transformar XML em um arquivo de dados binários, como um arquivo PDF ou um arquivo do Microsoft Excel.

Portas de Passagem

Você pode configurar portas de passagem para qualquer transformação do Processador de Dados. As portas de passagem são portas de entrada e saída que recebem dados de entrada e retornam os mesmos dados para um mapeamento sem alterá-los.

Você pode configurar portas de passagem em uma instância de transformação do Processador de Dados que está em um mapeamento.

Para adicionar uma porta de passagem, arraste uma porta de outra transformação no mapeamento. Você também pode adicionar portas na guia **Portas** da exibição **Propriedades**. Clique em **Novo** para adicionar uma porta de passagem.

Nota: Quando você adicionar portas de passagem em uma transformação do Processador de Dados com entrada relacional e saída hierárquica, adicione as portas ao grupo raiz da estrutura relacional.

As transformações do Processador de Dados podem incluir portas de passagem com tipos de dados personalizados.

Componente de Inicialização

Um componente de inicialização define o componente que inicia o processamento na transformação do Processador de Dados. Configure o componente de inicialização na exibição **Visão Geral**.

Uma transformação do Processador de Dados pode conter vários componentes para processar dados. Cada componente pode conter outros componentes. É necessário identificar qual componente é o ponto de entrada para a transformação.

Ao configurar o componente de inicialização em uma transformação do Processador de Dados, você pode escolher um componente de XMap, Biblioteca ou Script como o componente de inicialização. Em termos de Scripts, você pode selecionar um dos seguintes tipos de componentes:

- Analisador. Converte documentos de origem em XML. A entrada pode ter qualquer formato, como texto, HTML, Word, PDF ou HL7.
- Mapeador. Converte um documento de origem XML em outra estrutura ou esquema XML.
- Serializador. Converte um arquivo XML em um documento de saída de qualquer formato.
- Streamer. Divide grandes documentos de entrada, como fluxos de dados multigigabytes, em segmentos.
- Transformador. Modifica os dados em qualquer formato. Adiciona, remove, converte ou altera o texto. Use Transformadores com um Analisador, um Mapeador ou um Serializador. Também você pode executar um Transformador como um componente autônomo.

Nota: Se o componente de inicialização não for um XMap ou uma Biblioteca, você também poderá configurá-lo em um Script, em vez de na exibição **Visão Geral**.

Referências

Você pode definir as referências da transformação, como as referência de esquema ou de mapplet, selecionando um esquema ou um mapplet para servir como uma referência. Algumas transformações de Processador de Dados exigem um esquema hierárquico para definir a hierarquia de entrada ou de saída de componentes relevantes na transformação. Para usar o esquema na transformação, defina uma referência de esquema para a transformação. Você também pode usar uma ação especializada denominada ação RunMapplet para chamar um mapplet de uma transformação do Processador de Dados. Para chamar um mapplet, primeiro é necessário definir uma referência de mapplet para a transformação.

Você pode definir as referências da transformação, como as referências de esquema ou de mapplet, na exibição **Referências** da transformação.

Referências de Esquema

A transformação do Processador de Dados faz referência aos objetos de esquema no repositório do Modelo. Os objetos de esquema podem existir no repositório antes de você criar a transformação. Você também pode importar os esquemas da exibição **Referências** da transformação.

A codificação do esquema deve corresponder à codificação de entrada dos objetos Serializador ou Mapeador. A codificação do esquema deve corresponder à codificação de saída dos objetos Analisador ou Mapeador. Configure a codificação de trabalho na exibição **Configurações** da transformação.

Um esquema pode fazer referência a esquemas adicionais. A ferramenta Developer mostra o espaço de nome e o prefixo de cada esquema ao qual a transformação do Processador de Dados faz referência. Quando você faz referência a vários esquemas com espaços de nome vazios, a transformação não é válida.

Referências de Mapplet

Você pode chamar um mapplet de uma transformação do Processador de Dados com a ação RunMapplet. Antes de adicionar a ação RunMapplet a um componente da transformação do Processador de Dados, primeiro defina uma referência para o mapplet que você deseja chamar.

Configurações da Transformação do Processador de Dados

Configure as páginas de código, as opções de processamento de XML e as configurações de log na exibição **Configurações** da transformação do Processador de Dados.

Codificação de Caracteres

Uma codificação de caracteres é um mapeamento de caracteres de um idioma ou grupo de idiomas para código hexadecimal.

Ao criar um Script, você define a codificação dos documentos de entrada e a codificação dos documentos de saída. Defina a codificação de trabalho para especificar como o editor do IntelliScript exibe caracteres e como a transformação do Processador de Dados processa esses caracteres.

Codificação de Trabalho

A codificação de trabalho é a página de código para os dados na memória e a página de código para os dados que aparecem na interface do usuário e em arquivos de trabalho. É necessário selecionar uma codificação de trabalho que seja compatível com a codificação dos esquemas aos quais você faz referência na transformação do Processador de Dados.

A seguinte tabela mostra as configurações da codificação de trabalho:

Configuração	Descrição
Usar a Página de Código Padrão do Processador de Dados	Usa a codificação padrão da transformação do Processador de Dados.
Outros	Selecione a codificação na lista.
Codificação de Caracteres Especiais XML	<p>Determina a representação de caracteres especiais XML. Você pode selecionar Nenhuma ou XML.</p> <ul style="list-style-type: none">- Nenhuma. Deixar como <code>&amp;</code>, <code>&lt;</code>, <code>&gt;</code>, <code>&quot;</code>, <code>&apos;</code>; Referências de entidade para caracteres especiais XML são interpretadas como texto. Por exemplo, o caractere <code>></code> é exibido como <code>&gt;</code>; O padrão é nenhuma.- XML. Converter em <code>&</code> e <code><</code> e <code>"</code> e <code>'</code> Referências de entidade para caracteres especiais XML são interpretadas como caracteres regulares. Por exemplo, <code>&gt;</code> aparece como o caractere a seguir: <code>></code>

Codificação de Entrada

A codificação de entrada determina como os dados de caracteres são codificados em documentos de entrada. É possível configurar a codificação para portas de entrada adicionais em um Script.

A seguinte tabela descreve as configurações de codificação na área **Entrada**:

Configuração	Descrição
Usar a Codificação Especificada no Documento de Entrada	Use a página de código definida pelo documento de origem, como o atributo de codificação de um documento XML. Se o documento de origem não tiver uma especificação de codificação, a transformação do Processador de Dados usará as configurações de codificação da exibição Configurações .
Usar Codificação de Trabalho	Use a mesma codificação que a codificação de trabalho.
Outros	Selecione a codificação de entrada em uma lista suspensa.
Codificação de Caracteres Especiais XML	Determina a representação de caracteres especiais XML. É possível selecionar Nenhuma ou XML . - Nenhuma. Deixar como & < > " ' Referências de entidade para caracteres especiais XML são interpretadas como texto, por exemplo, o caractere > aparece como > O padrão é Nenhuma. - XML. Converter em & < > " ' Referências de entidade para caracteres especiais XML são interpretadas como caracteres regulares. Por exemplo, > aparece como o caractere a seguir: >
Ordem de Bytes	Descreve como os caracteres de vários bytes aparecem no documento de entrada. Você pode selecionar as seguintes opções: - Little-endian. O byte menos significativo aparece em primeiro lugar. Padrão. - Big-endian. O byte mais significativo aparece em primeiro lugar. - Nenhuma conversão binária.

Codificação de Saída

A codificação de saída determina como os dados de caracteres são codificados no documento de saída principal.

A seguinte tabela descreve as configurações de codificação na área **Saída**:

Configuração	Descrição
Usar Codificação de Trabalho	A codificação de saída é igual à codificação de trabalho.
Outros	O usuário seleciona a codificação de saída na lista.

Configuração	Descrição
Codificação de Caracteres Especiais XML	<p>Determina a representação de caracteres especiais XML. Você pode selecionar Nenhuma ou XML.</p> <ul style="list-style-type: none"> - Nenhuma. Deixar como &amp; &lt; &gt; &quot; &apos; Referências de entidade para caracteres especiais XML são interpretadas como texto, por exemplo, o caractere > aparece como &gt;; Padrão. - XML. Converter em & < > " ' " Referências de entidade para caracteres especiais XML são interpretadas como caracteres regulares. Por exemplo, &gt; aparece como o caractere a seguir: >
Igual à Codificação de Entrada	A codificação de saída é igual à codificação de entrada.
Ordem de bytes	<p>Descreve como os caracteres de vários bytes aparecem no documento de entrada. Você pode selecionar as seguintes opções:</p> <ul style="list-style-type: none"> - Little-endian. O byte menos significativo aparece em primeiro lugar. Padrão. - Big-endian. O byte mais significativo aparece em primeiro lugar. - Nenhuma conversão binária.

Regras e Diretrizes da Codificação de Caracteres

Use as seguintes regras e diretrizes ao configurar codificações:

- Para aumentar o desempenho, defina a codificação de trabalho para ser igual à codificação do documento de saída.
- Defina a codificação de entrada para a mesma codificação do documento de entrada.
- Defina a codificação de saída para a mesma codificação do documento de saída.
- Para idiomas que têm caracteres de bytes múltiplos, defina a codificação de trabalho para UTF-8. Para a codificação de entrada e saída, você pode usar uma codificação Unicode, como UTF-8, ou uma página de código de byte duplo, como Big5 ou Shift_JIS.

Configurações de Saída

Defina configurações de controle de saída para controlar se a transformação do Processador de Dados cria logs de eventos e salva documentos de saída.

É possível controlar os tipos de mensagens que a transformação do Processador de Dados grava no log de eventos em tempo de design. Se você salvar os documentos de entrada analisados com os logs de eventos, poderá visualizar o contexto em que o erro ocorreu na exibição **Evento**.

A seguinte tabela descreve as configurações na área **Eventos de Tempo de Design**:

Configuração	Descrição
Registrar eventos de tempo de design	Determina se um log de eventos em tempo de design deve ou não ser criado. Por padrão, a transformação do Processador de Dados registra notificações, avisos e falhas no log de eventos em tempo de design. É possível excluir os seguintes tipos de eventos: <ul style="list-style-type: none"> - Notificações - Avisos - Falhas
Salvar documentos analisados	Determina quando a transformação do Processador de Dados salva um documento de entrada analisado. Você pode selecionar as seguintes opções: <ul style="list-style-type: none"> - Sempre. - Nunca - Em caso de falha O padrão é sempre.

A seguinte tabela descreve as configurações na área **Eventos de Tempo de Execução**:

Configuração	Descrição
Registrar eventos de tempo de execução	Determina se um log de eventos é criado quando você executa a transformação por meio de um mapeamento. <ul style="list-style-type: none"> - Nunca. - Em caso de falha O padrão é Nunca.

A seguinte tabela descreve as configurações na área **Saída**:

Configuração	Descrição
Desativar saída automática	Determina se a transformação do Processador de Dados grava a saída no arquivo de saída padrão. Desabilite a saída padrão nas seguintes situações: <ul style="list-style-type: none"> - Você transmite a saída de um Analisador para a entrada de outro componente antes de a transformação criar um arquivo de saída. - Você usa uma ação WriteValue para gravar dados diretamente na saída de um Script, em vez de transmitir dados através das portas de saída.
Desabilitar compactação de valor	Determina se a transformação do Processador de Dados usa a compactação de valores para otimizar o uso da memória. Importante: Não desabilite a compactação de valores, exceto quando o Suporte Global a Clientes da Informatica o aconselhar a fazer isso.

A tabela a seguir descreve as configurações na área **Modo de coleta de porta de saída binária**. Você pode selecionar uma dessas opções de saída binária para uma transformação de relacional para hierárquica com

saída XML, Avro ou Parquet ou para um Analisador de transformação de Processador de Dados com saída Avro ou Parquet.

Configuração	Descrição
Coletar linhas de entrada para uma única saída	Determina se a transformação de Processador de Dados acumula a entrada relacional em uma única porta de saída binária.
Dividir saída quando o tamanho exceder	Determina se a transformação de Processador de Dados divide a saída em partes com base em um tamanho máximo de saída declarado.
Linha de saída para cada linha (não coletar)	Determina se a transformação de Processador de Dados transmite a saída em linhas separadas.

Configurações de Processamento

As configurações de processamento definem como a transformação do Processador de Dados processa um elemento sem um tipo de dados definido. Essas configurações afetam Scripts. Elas não afetam os elementos que são processados por um XMap.

A seguinte tabela descreve as configurações de processamento que afetam o processamento do XML em Scripts:

Configuração	Descrição
Tratar como xs:string	A transformação do Processador de Dados trata um elemento sem tipo como uma cadeia. Na caixa de diálogo Escolher XPath , o elemento ou o atributo aparece como um único nó.
Tratar como xs:anyType	A transformação do Processador de Dados trata um elemento sem tipo como anyType. Na caixa de diálogo Escolher XPath , o elemento ou atributo aparece como uma árvore de nós. Um nó é do tipo xs:string, e todos os tipos de dados complexos nomeados aparecem como nós de árvore.

A seguinte tabela descreve uma configuração de processamento que afeta o processamento de Streamer:

Configuração	Descrição
Tamanho do bloco do streamer	Essa configuração define a quantidade de dados que o Streamer lê a cada vez de um fluxo de arquivo de entrada. A transformação do Processador de Dados aplica essa configuração a um Streamer com um arquivo de entrada.

A seguinte tabela descreve uma configuração de processamento que afeta o processamento da transformação de hierárquico para relacional:

Configuração	Descrição
Impor validação estrita	Essa configuração determina se a transformação de Processador de Dados realiza uma validação estrita para a entrada hierárquica. Quando a validação estrita é aplicada, o arquivo de entrada hierárquica deve estar em conformidade estrita com seu esquema. Essa opção pode ser aplicada quando o modo do Processador de Dados é definido como Mapeamento de Saída , o que cria portas de saída para saída relacional. Esta opção não é aplicada a mapeamentos com entrada JSON de versões anteriores à versão 10.2.1.
Normalizar entrada XML	Essa configuração determina se a transformação do Processador de Dados normaliza a entrada XML. Por padrão, a transformação realiza a normalização da entrada XML. Em alguns casos, você pode optar por ignorar a normalização automática para melhorar o desempenho.

Configurações de XMap

A configuração de XMap define como os processos de transformação do Processador de Dados processam os elementos de entrada XMap que não são transformados em elementos de saída. Os elementos não lidos são transmitidos para uma porta denominada **XMap_Unread_Input_Values**. A configuração entra em vigor somente quando o XMap é selecionado como o componente de inicialização. A configuração não afeta os elementos que o XMap processa.

Para transmitir elementos XMap não lidos para uma porta dedicada, ative a configuração **Gravar elementos não lidos em uma porta de saída adicional**.

Configuração de Saída XML

As configurações de geração de XML definem as características de documentos de saída XML.

A seguinte tabela descreve as configurações de geração de XML na área **Título do Esquema**:

Configuração	Descrição
Localização do esquema	Define a schemaLocation para o elemento raiz do principal documento de saída.
Nenhuma localização de esquema de espaço de nome	Define o atributo xsi:noNamespaceSchemaLocation do elemento raiz do principal documento de saída.

Defina as configurações do Modo de Saída XML para determinar como a transformação do Processador de Dados lida com elementos ou atributos ausentes no documento XML de entrada. A seguinte tabela descreve as configurações de geração de XML na área **Modo de Saída XML**:

Configuração	Descrição
No estado em que se encontra	Não adicione ou remova elementos vazios. O padrão é ativada.
Completo	Todos os elementos obrigatórios e opcionais definidos no esquema de saída são gravados na saída. Os elementos que não têm conteúdo são gravados como elementos vazios.
Compacto	Remove os elementos vazios da saída. Se Adicionar a Elementos estiver ativada, a transformação do Processador de Dados removerá apenas os elementos opcionais. Se Adicionar a Elementos estiver desativada, a transformação do Processador de Dados removerá todos os elementos vazios. A saída XML pode não ser válida.

A seguinte tabela descreve as configurações de geração de XML na área **Valores Padrão para Nós Obrigatórios**:

Configuração	Descrição
Adicionar a elementos	Quando o esquema de saída define um valor padrão para um elemento necessário, a saída inclui o elemento com um valor padrão. O padrão é ativada.
Adicionar a atributos	Quando o esquema de saída define um valor padrão para um atributo necessário, a saída inclui o atributo com seu valor padrão. O padrão é ativada.
Validar valores adicionados	Determina se a transformação do Processador de Dados valida os elementos vazios que são adicionados pela saída do modo Completo. O padrão é desativada. Se a opção Validar valores adicionados estiver ativada e o esquema não permitir elementos vazios, a saída XML poderá não ser válida.

A seguinte tabela descreve as configurações de geração de XML na área **Instruções de Processamento**:

Configuração	Descrição
Adicionar instruções de processamento XML	Define a codificação de caractere e versão XML do documento de saída. O padrão é marcado.
Versão XML	Define a versão XML. A definição de versão XML tem as seguintes opções: - 1.0 - 1.1 O padrão é 1.0.

Configuração	Descrição
Codificação	Define a codificação de caracteres especificada na instrução de processamento. A configuração de Codificação tem as seguintes opções: <ul style="list-style-type: none"> - Igual à codificação de saída. A codificação de saída na instrução de processamento é igual à codificação de saída definida nas configurações de transformação do Processador de Dados. - Personalizar. Define a codificação de saída na instrução de processamento. O usuário digita o valor no campo.
Adicionar instruções personalizadas de processamento	Adiciona outras instruções de processamento ao documento de saída. Digite a instrução de processamento exatamente conforme ela aparece no documento de saída. O padrão é desativada.

A seguinte tabela descreve as configurações de geração de XML na área **Raiz XML**:

Configuração	Descrição
Adicionar o elemento raiz XML	Adiciona um elemento raiz ao documento de saída. Use esta opção quando o documento de saída tiver mais de uma ocorrência do elemento raiz definido no esquema de saída. O padrão é desativada.
Nome do elemento raiz	Define um nome para o elemento raiz para adicionar ao documento de saída.

Eventos

Um evento é um registro de uma etapa de processamento a partir de um componente na transformação do Processador de Dados. Em um Script ou Biblioteca, cada âncora, ação ou transformador gera um evento. Em um XMap, cada instrução de mapeamento gera um evento.

É possível mostrar eventos na exibição **Eventos do Processador de Dados**.

Tipos de Eventos

A transformação do Processador de Dados grava os eventos em arquivos de log. Cada evento tem um tipo de evento que indica se o evento foi bem-sucedido, se o evento falhou ou se o evento foi executado com erros.

Um componente pode gerar um ou mais eventos. O componente pode passar ou falhar, dependendo se os eventos obtiveram êxito ou falharam. Se um evento falhar, um componente falhará.

A tabela a seguir descreve os tipos de eventos que a transformação do Processador de Dados gera:

Tipo de Evento	Descrição
Notificação	Operação normal.
Aviso	A transformação do Processador de Dados foi executada, mas ocorreu uma condição inesperada. Por exemplo, a transformação do Processador de Dados gravou dados várias vezes no mesmo elemento. Cada vez que o elemento é substituído, a transformação do Processador de Dados gera um aviso.

Tipo de Evento	Descrição
Falha	A transformação do Processador de Dados foi executada, mas um componente falhou. Por exemplo, um elemento de entrada obrigatório estava vazio.
Falha Opcional	A transformação do Processador de Dados foi executada, mas um componente opcional falhou. Por exemplo, uma âncora opcional não foi encontrada no documento de origem.
Erro Fatal	A transformação do Processador de Dados falhou devido a um erro grave. Por exemplo, o documento de entrada não existia.

Exibição de Eventos do Processador de Dados

A exibição **Eventos do Processador de Dados** mostra eventos quando você executa uma transformação do Processador de Dados a partir da ferramenta Developer.

A exibição **Eventos do Processador de Dados** possui um painel **Navegação** e um painel **Detalhes**. O painel **Navegação** contém uma árvore de navegação. A árvore de navegação lista os componentes que a transformação executou, em ordem cronológica. Cada nó tem um ícone que representa o evento mais grave abaixo dele na árvore. Quando você seleciona um nó no painel **Navegação**, os eventos aparecem no painel **Detalhes**.

A árvore de navegação contém os seguintes nós de nível superior:

- Inicialização de Serviço. Descreve os arquivos e as variáveis que a transformação do Processador de Dados inicializa.
- Execução. Lista os componentes que o Script, a Biblioteca ou o XMap executou.
- Resumo. Mostra estatísticas sobre o processamento.

Quando você executa um XMap, cada nome de nó no painel de navegação tem um número entre colchetes, como [5]. Para identificar a instrução que gerou os eventos do nó, clique com o botão direito na grade de instruções e selecione Ir para Número de Linha. Insira o número do nó.

Quando você executa um Script e clica duas vezes em um evento no painel **Navegação** ou no painel **Detalhes**, o editor de Script realça o componente de Script que gerou esse evento. O painel **Entrada** da exibição **Visualizador de Dados** realça a parte do exemplo de documento de origem que gerou o evento.

Logs

Um log contém um registro da transformação do Processador de Dados. A transformação do Processador de Dados grava eventos em logs.

A transformação do Processador de Dados cria os seguintes tipos de logs:

Log de eventos de tempo de design

O log de eventos de tempo de design contém eventos que ocorrem quando você executa a transformação do Processador de Dados na exibição **Visualizador de Dados**. Visualize o log de tempo de design na exibição **Eventos**.

Log de eventos de tempo de execução

O log de eventos de tempo de execução contém eventos que ocorrem quando você executa a transformação do Processador de Dados em um mapeamento. É possível visualizar esse log em um editor de texto ou arrastá-lo até a exibição **Eventos** da transformação do Processador de Dados.

Log do usuário

O log do usuário contém eventos que você configura para componentes em um Script. A transformação do Processador de Dados grava no log do usuário ao ser executada a partir da exibição **Visualizador de Dados** e quando você a executa em um mapeamento. É possível exibir o log do usuário em um editor de texto.

Log de Eventos de Tempo de Design

O log de eventos de tempo de design contém eventos que ocorrem quando você executa a transformação do Processador de Dados na exibição do **Visualizador de Dados** da ferramenta Developer.

Quando você executar uma transformação do Processador de Dados na exibição do **Visualizador de Dados** o log de eventos de tempo de design será mostrado na exibição de **Eventos do Processador**. Por padrão, o log de eventos de tempo de design contém notificações, avisos e falhas. Nas configurações de transformação, você pode configurar a transformação do Processador de Dados para excluir um ou mais tipos de eventos do log.

Ao salvar os documentos de entrada com o log, você poderá clicar em um evento na exibição de **Eventos do Processador de Dados** para achar a localização no documento de entrada que gerou o evento. Ao configurar a transformação do Processador de Dados, você poderá optar por salvar os arquivos de entrada de cada execução ou somente em falha.

O log de eventos de tempo de design é denominado `events.cme`. Você pode localizar o log de eventos de tempo de design da última execução da transformação do Processador de Dados no seguinte diretório:

```
C:\<Installation_directory>\clients\DT\CMReports\Init\events.cme
```

A transformação do Processador de Dados substituirá o log de eventos de tempo de design sempre que executar a transformação no **Visualizador de Dados**. Renomeie log de eventos de tempo de design se desejar exibi-lo após a execução mais recente da transformação ou se desejar comparar os logs de diferentes execuções. Quando você fechar a ferramenta Developer, ela não salvará os arquivos em

Log de Eventos de Tempo de Execução

O log de evento de tempo de execução grava os eventos que ocorrem quando você executa a transformação do Processador de Dados em um mapeamento.

Se a transformação do Processador de Dados conclui a execução sem falhas, ele não grava um log de eventos. Se houver falhas na execução, a transformação do Processador de Dados executa uma segunda vez e grava um log de eventos durante a segunda execução. O log de eventos de tempo de execução é denominado `events.cme`.

Em uma máquina Windows, o log de eventos no tempo de execução está no seguinte diretório:

```
C:\<Installation_Directory>\clients\DT\CMReports\Tmp\
```

Em uma máquina Linux ou UNIX, o log de eventos de tempo de execução para um usuário raiz no seguinte diretório:

```
/root/<Installation_Dirctory>/clients/DT/CMReports/Tmp
```

Em uma máquina Linux ou UNIX, você pode localizar o log de eventos de tempo de execução para um usuário que não é raiz no seguinte diretório:

```
/home/[UserName]/<Installation_Directory>/DT/CMReports/Tmp
```

Use o editor de configuração para alterar a localização do log de eventos de tempo de execução.

Exibindo um Log de Eventos na Exibição de Eventos do Processador de Dados

Use a exibição **Eventos do Processador de Dados** para exibir um log de eventos de tempo de criação ou um log de eventos de tempo de execução.

Abra o Windows Explorer e navegue para o arquivo de log de eventos que você deseja exibir. Arraste o log da janela do Windows Explorer para a exibição **Eventos do Processador de Dados**. Clique com o botão direito do mouse na exibição **Eventos do Processador de Dados** e selecione **Localizar** para pesquisar no log.

Nota: Para recarregar o log de eventos de tempo de design mais recente, clique com o botão direito do mouse na exibição **Eventos do Processador de Dados** e selecione **Recarregar Eventos do Projeto**.

Log do Usuário

O log do usuário contém mensagens personalizadas que você configura sobre falhas de componentes em um Script.

A transformação do Processador de Dados grava mensagens no log do usuário quando um Script é executado a partir da exibição **Visualizador de Dados** e quando você a executa em um mapeamento.

Quando um componente de Script tem a propriedade **on_fail**, você poderá configurá-lo para gravar uma mensagem no log do usuário se ele falhar. No Script, defina a propriedade **on_fail** como um dos seguintes valores:

- LogInfo
- LogWarning
- LogError

Cada execução do Script gera um novo log do usuário. O nome de arquivo do log do usuário contém o nome da transformação com um GUID exclusivo:

```
<Transformation_Name>_<GUID>.log
```

Por exemplo, CalculateValue_Aa93a9d14-a01f-442a-b9cb-c9ba5541b538.log

Em uma máquina Windows, o log do usuário está localizado no seguinte diretório:

```
c:\Users\[UserName]\AppData\Roaming\Informatica\DataTransformation\UserLogs
```

Em uma máquina Linux ou UNIX, o log para o usuário root está localizado no seguinte diretório:

```
/<Installation_Directory>/DataTransformation/UserLogs
```

Em uma máquina Linux ou UNIX, o log para o usuário não root está localizado no seguinte diretório:

```
home/<Installation_Directory>/DataTransformation/UserLogs
```


Desenvolvimento da Transformação do Processador de Dados

Use o assistente de Nova Transformação para gerar automaticamente uma transformação do Processador de Dados ou criar uma transformação do Processador de Dados em branco e para configuração posterior. Se você criar uma transformação do Processador de Dados em branco, deverá selecionar para criar um objeto Script, XMap, Biblioteca ou Regras de Validação na transformação. Um Script pode analisar documentos de origem em formato hierárquico, converter do formato hierárquico em outros formatos de arquivo ou mapear um documento hierárquico para outro formato hierárquico. Um XMap converte um arquivo hierárquico de entrada em um arquivo hierárquico de saída de outra estrutura. Uma Biblioteca transforma um tipo de mensagem de indústria em um documento XML com uma estrutura hierárquica, ou de XML em um formato padrão da indústria. Escolha os esquemas que definam as hierarquias de entrada ou saída.

1. Crie a transformação na Developer tool.
2. Para uma transformação do Processador de Dados em branco, execute as seguintes etapas adicionais:
 - a. Adicione referências de esquema que definem as hierarquias XML de entrada ou saída.
 - b. Crie um objeto Script, XMap, Biblioteca ou Regras de Validação.
3. Configure as portas de entrada e de saída.
4. Teste a transformação.

Criar a Transformação do Processador de Dados

Crie uma transformação do Processador de Dados na Developer tool. Se você criar uma transformação do Processador de Dados em branco, deverá criar um objeto Script, XMap, Biblioteca ou Regras de Validação na transformação. Como alternativa, você pode usar o assistente de Nova Transformação para gerar automaticamente uma transformação do Processador de Dados.

1. Na Developer tool, clique em **Arquivo > Nova > Transformação**.
2. Selecione a transformação do Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do repositório do Modelo para inserir a transformação.
4. Selecione se você deseja criar a transformação do Processador de Dados usando um assistente ou criar uma transformação do Processador de Dados em branco.
5. Se você preferir criar uma transformação do Processador de Dados em branco, clique em **Concluir**.
A Developer tool cria a transformação em branco no repositório. A exibição **Visão Geral** aparece na Developer tool.
6. Se você preferir criar uma transformação do Processador de Dados usando um assistente, realize as seguintes etapas:
 - a. Clique em **Avançar**.
 - b. Selecione um formato de entrada.
 - c. Navegue para selecionar um esquema, copybook, arquivo de exemplo, ou arquivo de especificação, se necessário para determinados formatos de entrada, como COBOL, JSON ou ASN.1.
 - d. Selecione um formato de saída.
 - e. Navegue para selecionar um esquema, copybook, arquivo de exemplo, ou arquivo de especificação, se necessário para o formato de saída.

f. Clique em **Concluir**. O assistente cria a transformação no repositório.

A transformação pode conter um Analisador, um Serializador, um Mapeador ou um objeto com componentes comuns. Se você selecionar um esquema, um copybook, um arquivo de exemplo ou um arquivo de especificação, o assistente também criará um esquema no repositório que seja equivalente à hierarquia no arquivo.

Selecionar os Objetos de Esquema

Escolha os objetos de esquema que definem as hierarquias de entrada ou de saída para cada componente de XMap ou de Script que você planeja criar.

É possível adicionar referências de esquema na exibição **Referências** ou adicionar essas referências ao criar objetos de Script ou XMap. Um objeto de esquema deve existir no repositório do Modelo para que você possa fazer referência a ele em um Script ou XMap.

1. Na exibição **Referências** da transformação do Processador de Dados, clique em **Adicionar**.
2. Se o objeto de esquema existir no repositório do Modelo, navegue até ele e selecione o esquema.
3. Se o esquema não existir no repositório do modelo, clique em **Criar um novo objeto de esquema** e importe um objeto de esquema de um arquivo de esquema hierárquico.
4. Clique em **Concluir** para adicionar a referência de esquema à transformação do Processador de Dados.

Criar Objetos em uma Transformação do Processador de Dados em Branco

Crie um objeto Script, Biblioteca, XMap ou Regras de Validação na exibição **Objetos** da transformação do Processador de Dados. Após criar o objeto, você pode abrir o objeto na exibição **Objetos** para configurá-lo.

Criando um Script

Crie um objeto de Script e defina o tipo de componente de Script a ser criado. Opcionalmente, é possível definir uma referência de esquema e um arquivo de origem de exemplo.

1. Na exibição **Objetos** da transformação do Processador de Dados, clique em **Novo**.
2. Insira um nome para o Script e clique em **Avançar**.
3. Escolha a criação de um Analisador ou de um Serializador. Selecione Outros para criar um componente Mapeador, Transformador ou Streamer.
4. Insira um nome para o componente.
5. Se o componente for o primeiro a processar dados na transformação, habilite **Definir como componente de inicialização**.
6. Clique em **Avançar** se quiser inserir uma referência de esquema para esse Script. Clique em **Concluir** se não quiser inserir a referência de esquema.
7. Se você optar por criar uma referência de esquema, selecione **Adicionar referência a um objeto de esquema** e procure o objeto de Esquema no repositório do Modelo. Clique em **Criar um novo objeto de esquema** para criar o objeto de Esquema no repositório do Modelo.
8. Clique em **Avançar** para inserir uma referência de origem de exemplo ou para inserir um texto de exemplo. Clique em **Concluir** se não quiser definir uma origem de exemplo.
Use uma origem de exemplo para definir dados de amostra e para testar o Script.
9. Se você optar por selecionar uma origem de exemplo, selecione **Arquivo** e procure o arquivo de amostra.

Também é possível inserir um texto de amostra na área **Texto**. A Developer tool usa esse texto para testar um Script.

10. Clique em **Concluir**.

A exibição **Script** aparece no editor da Developer tool.

Criando um XMap

Crie um XMap na exibição de **Objetos** da Data Transformation. Ao criar um XMap, você deve ter um esquema que descreve a entrada e a saída dos documentos hierárquicos. Selecione o elemento no esquema que é o elemento raiz da hierarquia de entrada.

1. Na exibição **Objetos** da transformação do Processador de Dados, clique em **Novo**.
2. Selecione XMap e clique em **Avançar**.
3. Insira um nome para o XMap.
4. Se o componente XMap for o primeiro componente a processar dados na transformação, ative **Definir como componente de inicialização**.

Clique em **Avançar**.

5. Se você optar por criar uma referência de esquema, selecione **Adicionar referência a um Objeto de Esquema** e procure o objeto de Esquema no repositório do Modelo.

Para importar um novo objeto de Esquema, clique em **Criar um novo objeto de esquema**.

6. Se você tiver um arquivo hierárquico de amostra que possa usar para testar o XMap, procure e selecione o arquivo no sistema de arquivos.

Você pode alterar o arquivo hierárquico de amostra.

7. Escolha a raiz da hierarquia de entrada.

Na caixa de diálogo **Seleção de Elemento Raiz**, selecione o elemento no esquema que é o elemento raiz do arquivo hierárquico de entrada. Você pode pesquisar um elemento no esquema. Você pode usar pesquisa padrão. Digite `*<string>` para fazer a correspondência de qualquer número de caracteres na string. Digite `?<character>` para fazer a correspondência de um único caractere.

8. Clique em **Concluir**.

A ferramenta Developer cria uma exibição para cada XMap que você criar. Clique em **exibir** para configurar o mapeamento.

Criando uma Biblioteca

Crie um objeto Biblioteca na exibição **Objetos** da Data Transformation. Selecione o tipo de mensagem, o componente e o nome. Opcionalmente, você pode definir um arquivo de origem de tipo de mensagem de exemplo que pode usar para testar o objeto Biblioteca.

Antes de criar uma Biblioteca na transformação do Processador de Dados, instale o pacote de software da biblioteca no seu computador.

1. Na exibição **Objetos** da transformação do Processador de Dados, clique em **Novo**.
2. Selecione a Biblioteca e clique em **Avançar**.
3. Procure e selecione o tipo de mensagem.
4. Escolha a criação de um Analisador ou de um Serializador.

Crie um Analisador se a entrada do objeto de Biblioteca for um tipo de mensagem e a saída for XML.

Crie um Serializador se a entrada do objeto de Biblioteca for XML e a saída for um tipo de mensagem.

5. Se a Biblioteca for o primeiro componente a processar dados na transformação do Processador de Dados, habilite **Definir como componente de inicialização**.
Clique em **Avançar**.
6. Se você tiver um exemplo de tipo de mensagem do arquivo de origem, pode usá-lo para testar a Biblioteca, procurar e selecionar o arquivo do sistema de arquivos.
Você pode alterar o arquivo de exemplo.
7. Clique em **Concluir**.
A Developer tool cria uma exibição para cada tipo de mensagem que você criar. Clique na exibição para acessar o mapeamento.

Criando Regras de Validação

Crie um objeto Regras de Validação na exibição **Objetos** da transformação do Processador de Dados.

1. Na exibição **Objetos** da transformação do Processador de Dados, clique em **Novo**.
2. Selecione as Regras de Validação e clique em **Avançar**.
3. Insira um nome para as Regras de Validação.
4. Se você tiver um arquivo XML de amostra que possa usar para testar as Regras de Validação, procure e selecione o arquivo no sistema de arquivos.
Você pode alterar o arquivo XML de exemplo.
5. Clique em **Concluir**.
A ferramenta Developer cria um objeto Regras de Validação e abre-o no editor de Regras de Validação.

Adicionando um Exemplo de Origem

Escolha a origem de exemplo para testar o Script, o XMap, a Biblioteca ou as Regras de Validação que você planeja criar.

Você pode adicionar uma origem de exemplo ao criar um Script, um XMap, uma Biblioteca ou Regras de Validação. Depois de ser selecionada, a origem de exemplo será adicionada ao repositório do Modelo. Devido às limitações do repositório do Modelo, o tamanho do arquivo de origem de exemplo é limitado a 5 MB.

Você pode alterar a origem de exemplo.

Criar as Portas

Configure as portas de entrada e saída na exibição **Visão Geral**.

Quando você configura portas de entrada ou saída adicionais em um Script, a Developer tool acrescenta portas de entrada e de saída adicionais à transformação por padrão. Você não adiciona portas de entrada na exibição **Visão Geral**.

1. Se quiser retornar linhas de dados de saída em vez de XML, habilite a **Saída Relacional**.
Quando você habilita a saída relacional, a Developer tool remove a porta de saída padrão.
2. Selecione o tipo de dados da porta de entrada, o tipo de porta, a precisão e a escala.
3. Se você não estiver definindo portas de saída relacionais, defina o tipo de dados da porta de saída, o tipo de porta, a precisão e a escala.

4. Se um Script tiver portas de entrada adicionais, será possível definir a localização do exemplo de arquivo de entrada para essas portas. Clique no botão **Abrir** no campo **Localização de Entrada** para procurar o arquivo.
5. Se você tiver habilitado a saída relacional, clique em **Mapeamento de Saída** para criar as portas de saída.
6. Na exibição Portas, mapeie nós da área **Saída Hierárquica** para campos na área **Portas Relacionais**.

Testando a Transformação

Testar a transformação do Processador de Dados na exibição **Visualizador de Dados**.

Antes de testar a transformação, verifique se você definiu o componente de inicialização. É possível definir o componente de inicialização em um Script ou selecioná-lo na guia **Visão Geral**. Você também precisa ter escolhido um arquivo de entrada de exemplo para realizar testes.

1. Abra a exibição **Visualizador de Dados**.
 2. Clique em **Executar**.

A Developer tool valida a transformação. Se não houver erros, a Developer tool mostrará o arquivo de exemplo na área **Entrada**. Os resultados da saída aparecem no painel Saída.
 3. Clique em **Mostrar Eventos** para mostrar a exibição **Eventos do Processador de Dados**.
 4. Clique duas vezes em um evento na exibição **Eventos do Processador de Dados** para depurar esse evento no editor de Script.
 5. Clique em **Sincronizar com Editor** para alterar o arquivo de entrada quando você estiver testando vários componentes, cada uma com um arquivo de entrada de exemplo diferente.
- Se você modificar o conteúdo do arquivo de exemplo no sistema de arquivos, as alterações aparecerão na área **Entrada**.

Exportação e Importação da Transformação do Processador de Dados

Você pode exportar uma transformação do Processador de Dados como serviço e executá-la a partir do repositório do Data Transformation. Você também pode importar um serviço da Data Transformation para a ferramenta Developer. Quando você importa um serviço de Data Transformation, a ferramenta Developer cria uma transformação do Processador de Dados a partir do serviço.

Nota: Quando você importa um serviço de Data Transformation para o repositório do Modelo, a ferramenta Developer importa os esquemas associados para o repositório. Se você modificar o esquema no repositório, as alterações poderão não ser exibidas nas referências do esquema da transformação. Você pode fechar e abrir a conexão do repositório do Modelo ou fechar e abrir a Developer tool para fazer com que as alterações do esquema apareçam na transformação.

Exportando a Transformação do Processador de Dados como um Serviço

Você pode exportar a transformação do Processador de Dados como um serviço de Data Transformation. Exporte o serviço para o arquivo de repositório do sistema da máquina onde você deseja executar o serviço.

Você pode executar o serviço com o PowerCenter, com aplicativos definidos pelo usuário ou com o comando CM_console da Data Transformation.

1. Na exibição **Explorador de Objetos**, clique com o botão direito na transformação do Processador de Dados que você deseja exportar e selecione **Exportar**.
A caixa de diálogo **Exportar** será exibida.
2. Selecione **Informatica > Exportar Transformação do Processador de Dados** e clique em **Avançar**.
A página **Selecionar** será exibida.
3. Clique em **Avançar**.
A página **Selecione o Nome do Serviço e a Pasta de Destino** será exibida.
4. Escolha uma pasta de destino:
 - Para exportar o serviço na máquina que hospeda a Developer tool, clique em **Pasta de Serviço**.
 - Para implantar o serviço em outra máquina, clique em **Pasta**. Navegue até o diretório \ServiceDB na máquina onde você deseja implantar o serviço.
5. Clique em **Concluir**.

Importando vários serviços do Data Transformation

É possível importar um diretório de serviços do Data Transformation da máquina em que diretório foi salvo. Quando você importa serviços do Data Transformation para o repositório do Modelo do Developer, a Developer tool importa as transformações, os esquemas e os dados de exemplo com os arquivos .cmw. Se for necessário importar muitos serviços, importe um diretório de serviços em vez de um serviço de cada vez.

1. Clique em **Arquivo > Importar**.
A caixa de diálogo **Importar** é exibida.
2. Selecione **Informatica Importar Serviços do Data Transformation (Pasta)** e clique em **Avançar**.
A página **Importar Serviço de Data Transformation** será exibida.
3. Navegue até o diretório que você deseja importar.
4. Procure uma localização no Repositório na qual você queira salvar as transformações e clique em **Concluir**.
A Developer tool importa as transformações, os esquemas e os dados de exemplo com o arquivo .cmw.

Importando um Serviço de Data Transformation

É possível importar um arquivo .cmw do serviço do Data Transformation para o repositório do Modelo para criar uma transformação de Processador de Dados. A ferramenta Developer importa a transformação, os esquemas e os dados do exemplo com o arquivo .cmw.

1. Clique em **Arquivo > Importar**.
A caixa de diálogo **Importar** é exibida.
2. Selecione **Informatica Importar Serviço do Data Transformation (Único)** e clique em **Avançar**.
A página **Importar Serviço de Data Transformation** será exibida.
3. Navegue até o arquivo .cmw de serviço que você deseja importar.
A ferramenta Developer nomeia a transformação de acordo com o nome de arquivo do serviço. É possível alterar o nome.

4. Procure uma localização no Repositório onde você deseja salvar a transformação e clique em **Concluir**.
A ferramenta Developer importa a transformação, os esquemas e os dados de exemplo com o arquivo .cmw.
5. Para editar a transformação, clique duas vezes na transformação no **Explorador de Objetos**.

Exportando um Mapeamento com uma Transformação do Processador de Dados para o PowerCenter

Quando você exporta um mapeamento com uma transformação do Processador de Dados para o PowerCenter, pode exportar os objetos para um arquivo local e importar o mapeamento para o PowerCenter. Como alternativa, você pode exportar o mapeamento diretamente para o repositório do PowerCenter.

1. Na exibição **Object Explorer**, selecione o mapeamento a ser exportado. Clique duas vezes e selecione **Exportar**.
A caixa de diálogo **Exportar** é exibida.
2. Selecione **Informatica > PowerCenter**.
3. Clique em **Avançar**.
A caixa de diálogo **Exportar para o PowerCenter** é exibida.
4. Selecione o projeto.
5. Selecione a versão do PowerCenter.
6. Escolha a localização de exportação, um arquivo XML de importação do PowerCenter ou um repositório do PowerCenter.
7. Especifique as opções de exportação.
8. Clique em **Avançar**.
A ferramenta Developer solicita que você selecione os objetos a serem exportados.
9. Selecione os objetos a serem exportados e clique em **Concluir**.
A ferramenta Developer exporta os objetos para a localização selecionada. Se você exportou o mapeamento para uma localização, a ferramenta Developer também exporta as transformações de Processador de Dados no mapeamento, como serviços, para uma pasta na localização especificada.
10. Se você exportou o mapeamento para um repositório do PowerCenter, os serviços são exportados para o seguinte caminho de diretório: %temp%\DTServiceExport2PC\
A função de exportação cria uma pasta separada para cada serviço com o seguinte nome:
<date><serviceFullName>
Se a transformação incluir o mapeamento relacional, uma pasta será criada para o mapeamento relacional para hierárquico e uma pasta separada para o mapeamento hierárquico para relacional.
11. Copie a pasta ou as pastas com serviços de transformação do Processador de Dados da localização local onde você exportou os arquivos para a pasta ServiceDB do PowerCenter.
12. Se você exportou o mapeamento para um arquivo XML de importação do PowerCenter, importe o mapeamento para o PowerCenter. Para obter mais informações sobre como importar um objeto para o PowerCenter, consulte *Guia do Repositório do PowerCenter 9.6.0*.

Transformação de Processador de Dados Validação

Depois de exportar uma transformação do Processador de Dados como um serviço, você poderá executar validações VRL no serviço do repositório do Data Transformation.

Você pode usar um mecanismo do Data Transformation com velocidade aprimorada para validações VRL. O mecanismo do Data Transformation com velocidade aprimorada oferece suporte às seguintes funções VRL:

- `dt:exist`
- `dt:empty`
- `dt:date-valid`
- `dt:next-sequence`
- `dt:all-equal`
- `dt:lookup`
- `dt:regex-match`

O mecanismo do Data Transformation com velocidade aprimorada produz a saída **ValidateValue**.

ValidateValue contém a propriedade `max_error_count`, com um padrão de 200 erros. Quando o número de erros excede o `max_error_count`, a validação para.

Nota: A sintaxe VRL do mecanismo do Data Transformation com velocidade aprimorada não suporta a marcação `<list>`.

Usando um Mecanismo do Data Transformation com Velocidade Aprimorada para Validações VRL

Depois de exportar um serviço do Data Transformation, você pode usar um mecanismo do Data Transformation com velocidade aprimorada para validações VRL com o serviço.

- ▶ Defina a seguinte sinalização no arquivo `.cmw` de serviço `optimize_vrl`.

Adicione a sinalização `optimize_vrl` para a instância do `ServiceConfigProf`, conforme mostrado no seguinte exemplo:

```
instance ServiceConfig = ServiceConfigProf<add_required_xml_elements,  
add_required_xml_attributes, optimize_vrl>
```

Transformação de Processador de Dados em um ambiente não nativo

O processamento da transformação de Processador de dados em um ambiente não nativo depende do mecanismo que executa a transformação.

Considere o suporte para os seguintes mecanismos de tempo de execução não nativos:

- Mecanismo Blaze. Suportado sem restrições.
- Mecanismo Spark. Com suporte com restrições em mapeamentos em lote. Sem suporte em mapeamentos de fluxo.*
- Mecanismo Databricks Spark. Sem suporte.

** Para obter informações sobre o suporte à transformação de Processador de Dados no mecanismo Spark, consulte o [KB article](#).*

CAPÍTULO 3

Formatos de Entrada e Saída do Assistente

Este capítulo inclui os seguintes tópicos:

- [Visão Geral dos Formatos de Entrada e de Saída do Assistente, 50](#)
- [Avro, 51](#)
- [Biblioteca de Processamento COBOL, 56](#)
- [JSON, 59](#)
- [Parquet, 62](#)
- [XML, 64](#)

Visão Geral dos Formatos de Entrada e de Saída do Assistente

Você pode usar um assistente para criar uma transformação do Processador de Dados gerada automaticamente com os formatos de entrada e de saída, como COBOL, XML, relacional ou JSON. Você também pode usar o assistente para transformar formatos definidos pelo usuário.

Crie uma transformação de Processador de Dados e selecione os formatos de entrada e saída por meio do assistente de transformação de Processador de Dados. Selecione um formato existente ou crie formatos definidos pelo usuário. Para determinados formatos, como XML, JSON ou COBOL, adicione um esquema, um arquivo de especificações, um arquivo de exemplo ou um copybook que defina a estrutura esperada para a entrada ou a saída.

O assistente cria uma transformação com objetos Script, XMap ou Library relevantes que servem como modelos para transformar o formato de entrada no formato de saída. A transformação de Processador de Dados cria uma solução de transformação de acordo com os formatos selecionados e o arquivo de especificações, o arquivo de exemplo, ou o copybook. A transformação talvez não seja completa, mas contém componentes que você conecta e personaliza para concluir a definição de transformação.

Avro

Use o assistente para criar uma transformação com a entrada ou a saída Avro. Quando você criar uma transformação do Processador de Dados para transformar o formato Avro, selecione um esquema Avro ou um arquivo de exemplo que defina a estrutura esperada dos dados Avro. O assistente cria componentes que transformam o formato Avro em outros formatos ou vice-versa. Esses componentes podem incluir um mapeamento relacional a hierárquico, um mapeamento hierárquico a relacional e um XMap. Depois que o assistente criar a transformação, você poderá configurar melhor a transformação para determinar a lógica de mapeamento.

O Apache Avro é um sistema de serialização de dados em binário ou outros formatos de dados. Os dados Avro estão em um formato que pode não ser compreendido diretamente pelo usuário. Para obter mais informações sobre o Avro, consulte <http://avro.apache.org/>

Nota: Use o Avro com codificação binária para criar uma transformação com a entrada ou a saída Avro. A entrada ou a saída Avro não é processada em outros formatos.

Uma transformação que lê a entrada ou a saída Avro se baseia em um esquema. Quando a transformação lê ou grava dados Avro, a transformação usa o esquema para interpretar a hierarquia.

Quando você seleciona um arquivo de exemplo para definir a hierarquia Avro, o assistente também salva o primeiro registro no arquivo como um arquivo de teste separado. Você pode usar esse arquivo para testar a transformação. Para localizar o arquivo, no painel **Portas** da exibição **Visão Geral**, verifique o caminho de arquivo listado no campo **Localização de Entrada**.

Quando você cria uma transformação do Processador de Dados que transforma o Avro em formato hierárquico ou o formato hierárquico em Avro, o assistente cria um componente XMap na transformação. O editor de XMap mostra os nós de esquema hierárquicos e os nós de esquema Avro. Use o editor de XMap para vincular os nós e definir a lógica de transformação. Para obter mais informações sobre o objeto XMap e o editor, consulte [“Visão Geral do XMap” na página 85](#).

Quando você cria uma transformação que transforma Avro em formato relacional ou o formato relacional em Avro, o assistente cria um mapeamento relacional. O painel **Portas** na exibição **Visão Geral** mostra os nós de esquema hierárquicos e as portas relacionais. Use o painel **Portas** para vincular os elementos hierárquicos às portas e aos grupos relacionais. Para obter mais informações sobre a transformação de dados relacionais, consulte [“Visão Geral de Entradas e Saídas Relacionais” na página 66](#).

Depois de criar uma transformação do Processador de Dados para a entrada Avro, adicione-a a um mapeamento com um leitor de arquivos complexos. O leitor de arquivos complexos transfere a entrada Avro para a transformação. Para uma transformação do Processador de Dados com a saída Avro, adicione um gravador de arquivos complexos ao mapeamento para receber a saída da transformação.

Entrada Avro e Leitor de Arquivos Complexos

Para que a transformação do Processador de Dados transforme a entrada Avro, a transformação terá de receber entrada de dados de um objeto do leitor de arquivos complexos. Depois de criar e configurar a transformação, adicione a transformação a um mapeamento e conecte a porta de entrada à porta de saída do leitor de arquivos complexos.

O leitor de arquivos complexos fornece entrada a um componente Streamer que o assistente de transformação do Processador de Dados cria como parte da transformação. Se a transformação transformar a entrada Avro em um formato personalizado ou relacional, não haverá a necessidade de alterar as configurações do Streamer. Especifique um formato de saída personalizado selecionando **Outros** como o formato de saída no assistente.

Se a transformação transformar a entrada Avro no formato JSON, XML ou Avro, o assistente criará um XMap para mapear o formato de saída. Você deve identificar o XMap para o Streamer, para que a transformação processe os dados corretamente.

Você também deve configurar o leitor de arquivos complexos para processar a entrada Avro. A porta de saída do leitor de arquivos complexos deve ser definida em formato binário. Da mesma forma, a porta de entrada da transformação do Processador de Dados deve ser definida como entrada binária.

Compactação de Dados Avro com o Codec Snappy

Você pode compactar dados Avro com o leitor de arquivos complexos. Se você usar o codec Snappy para a compactação de dados Avro, deverá atualizar o arquivo .jar do codec Snappy antes de testar ou executar a transformação.

Para usar o codec Snappy, substitua o arquivo .jar padrão do Snappy na instalação do servidor Informatica e no ambiente Hadoop pela versão atualizada. O arquivo `snappy-java-1.0.4.1.jar` atualizado está disponível no seguinte link: <http://mvnrepository.com/artifact/org.xerial.snappy/snappy-java/1.1.0.1>

Atualizando o Codec Snappy para Ativar a Compactação de Dados Avro

Para ativar a compactação de dados Avro com o codec Snappy, substitua o arquivo .jar Snappy padrão pela versão atualizada.

1. Na máquina onde você instalou o Informatica server, substitua o arquivo `snappy-java-1.0.4.1.jar` na instalação do servidor pelo arquivo `snappy-java-1.1.0.1.jar`. Substitua o .jar no seguinte caminho: `<Server_Installation>\services\shared\hadoop\<Hadoop_Distribution>\lib`
2. Nas máquinas onde você instalou e executa o Hadoop, substitua o arquivo `snappy-java-1.0.4.1.jar` pelo arquivo `snappy-java-1.1.0.1.jar`. Substitua o .jar no seguinte caminho: `<Hadoop_rpm>\services\shared\hadoop\<Hadoop_Distribution>\lib`

Configurar uma Transformação com a Entrada Avro

Para criar uma transformação do Processador de Dados para uma entrada Avro, use o assistente de transformação do Processador de Dados. O assistente cria a transformação no repositório do Modelo com os componentes necessários para transformar a entrada Avro. Use o Editor do IntelliScript para editar o Streamer e o editor de XMap para editar um XMap, se estiver incluído na transformação. Adicione a transformação a um mapeamento com um leitor de arquivos complexos.

1. Crie a transformação do Processador de Dados com o assistente de Nova Transformação. Adicione um esquema Avro ou um arquivo de exemplo que defina a estrutura de entrada esperada.
2. Se a transformação tiver XML, JSON ou outro formato estruturado como saída, use o editor de XMap para editar o XMap na transformação.
3. Use o Editor do IntelliScript para editar e personalizar o Streamer na transformação. Se a transformação tiver XML, JSON ou outro formato estruturado de saída, edite o Streamer para identificar o XMap na transformação.
4. Adicione a transformação do Processador de Dados a um mapeamento com um leitor de arquivos complexos. A transformação deve permanecer definida como entrada binária, a configuração padrão para a entrada Avro. Configure o leitor de arquivos complexos para processar o Avro. A configuração de saída permanece definida como binária, a configuração padrão. Vincule a porta de saída do leitor de arquivos complexos à porta de entrada da transformação do Processador de Dados para fornecer a entrada Avro à transformação.

Etapa 1. Criar uma Transformação que Transforma o Avro

Crie uma transformação do Processador de Dados com a entrada Avro, a saída Avro ou ambos.

1. Na ferramenta Developer, clique em **Arquivo > Nova > Transformação**.
2. Selecione a transformação de Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do Repositório do Modelo para inserir a transformação.
4. Selecione **Criar um processador de dados usando um assistente** e clique em **Avançar**.
5. Selecione Avro ou outro formato de entrada e clique em **Avançar**.
6. Se você tiver selecionado Avro como o formato de entrada, procure para selecionar um arquivo de esquema .xsd ou um arquivo de exemplo Avro. Clique em **Avançar**.

O Developer adiciona um arquivo de esquema .xsd que representa a hierarquia Avro para o Repositório do Modelo. Se você selecionar um arquivo de exemplo, o Developer criará um arquivo de teste do primeiro registro no arquivo de exemplo. Você pode usar esse arquivo para testar a transformação. Para localizar o arquivo, no painel **Portas** da exibição **Visão Geral**, verifique o caminho de arquivo listado no campo **Localização de Entrada**.
7. Selecione Avro ou outro formato de saída e clique em **Avançar**.
8. Se você selecionar Avro como um formato de saída, procure para selecionar um esquema relacionado ou um arquivo de exemplo Avro. Clique em **Avançar**.
9. Clique em **Concluir**.

A ferramenta Developer cria a transformação no repositório com os componentes relevantes, como um XMap para transformar a hierarquia Avro em outro formato de hierarquia. A exibição **Visão Geral** aparece na ferramenta Developer.
10. Para editar os componentes na transformação, na exibição **Objetos**, clique duas vezes no componente de transformação para abri-lo no editor relevante.

Etapa 2. Editar o XMap

Para configurar um objeto XMap da transformação do Processador de Dados, adicione instruções de mapeamento.

1. Para abrir o editor de XMap, nos **Objetos** da transformação do Processador de Dados, clique no objeto XMap.
2. Para criar uma instrução de mapeamento de Mapa, de Grupo ou de Grupo de Repetição, no editor de XMap, arraste e solte de um nó no esquema hierárquico de entrada para um nó no esquema hierárquico de saída.

O Editor de XMap cria um vínculo de mapa entre os nós. A instrução de mapeamento é exibida na grade. O Editor de XMap preenche automaticamente os campos de instrução de mapeamento.
3. Para criar a lógica condicional na grade, adicione uma instrução de mapeamento de Roteador da seguinte maneira:
 - a. Na instrução de mapeamento de Roteador, crie as instruções de mapeamento de Opção. Arraste e solte os nós de esquema de entrada e saída para os campos da instrução Opção na grade.
 - b. Na instrução de mapeamento de Roteador, crie a instrução de mapeamento de Padrão para especificar o que acontece se nenhuma instrução de mapeamento Opção se aplicar.
 - c. Com as instruções de mapeamento de Opção, crie instruções de mapeamento de Mapa para especificar condições para mapear o nó de entrada para o nó de saída.

4. Para fornecer um contexto comum para um grupo de instruções, adicione uma instrução de mapeamento de Grupo. Aninhe as instruções de mapeamento de Mapa abaixo da instrução de mapeamento de Grupo.
5. Para chamar outro objeto XMap, adicione uma instrução Executar XMap.
6. Para alterar o contexto e a lógica de mapeamento de uma instrução de mapeamento, edite as propriedades da instrução de mapeamento da seguinte maneira:
 - a. Rebaixe instruções para instruções filho ou promova instruções para instruções pai.
 - b. Crie expressões XPath para alterar o contexto ou adicionar predicados usando o editor de XPath.

Etapa 3. Configurar o Streamer

Se a transformação tiver saída XML, JSON ou outro formato estruturado selecionado como saída, o assistente criará um componente Streamer e objeto XMap no assistente. Edite o Streamer para identificar o XMap na transformação.

1. Na exibição **Objetos**, clique duas vezes no Streamer, um objeto de Script, para abri-lo no Editor do IntelliScript.
2. Para configurar o Streamer para identificar o XMap na transformação, execute as seguintes etapas:
 - a. Para configurar o elemento necessário, expanda o elemento **contém**, no Editor do IntelliScript. Clique duas vezes na seta dupla para a direita >> próximo ao elemento.
 - b. Expanda o elemento **repeating_segment**. Clique duas vezes na seta dupla para a direita >> próximo ao elemento.
 - c. No elemento **run_component**, selecione o objeto XMap relevante de lista de componentes.
3. Para configurar o Streamer para ler dados de localizações no documento de origem e gravar os dados em XML, clique com o botão direito
4. Para configurar mais o Streamer, no elemento **Streamer**, aninhe os componentes **ComplexSegment** e **SimpleSegment** de acordo com a estrutura de origem.
5. Para cada **SimpleSegment**, defina o marcador de abertura e o marcador de fechamento, se necessário. Defina a transformação que processa o segmento.

Etapa 4. Configurar o Leitor de Arquivos Complexos

Adicione uma transformação do Processador de Dados que transforma o Avro em um mapeamento com um leitor de arquivos complexos. Configure o leitor de arquivos complexos para processar a entrada Avro.

1. No editor de Mapeamento, crie um objeto de leitor de arquivos complexos.
2. Para configurar o leitor de arquivos complexos, realize as seguintes etapas:
 - a. Na guia **Avançado** da exibição **Propriedades**, selecione a propriedade **Formato de Arquivo** e escolha **Entrada Personalizada**.
 - b. Selecione a propriedade **Formato de Entrada** e, em seguida, digite `com.informatica.avro.AvroToXML`.
 - c. Para otimizar o desempenho, use a propriedade **Parâmetros do Formato de Entrada** para ajustar o parâmetro `MaxOutputAccumulation`. Por padrão, o parâmetro `MaxOutputAccumulation`, que define o número esperado de registros de saída, é definido como 50.000. Para alterar a configuração para 250.000 por exemplo, insira `"MaxOutputAccumulation"="250000"`.

- d. Por padrão, o leitor de arquivos complexos adiciona o esquema à saída do leitor de arquivos complexos em um único elemento diretamente depois do elemento raiz. Para não adicionar o esquema à saída, selecione a propriedade **Parâmetros do Formato de Entrada** e, em seguida, digite `"InjectSchema"="false"`.

Use um ponto e vírgula para separar vários parâmetros, por exemplo

`"MaxOutputAccumulation"="250000";"InjectSchema"="false"`.

3. Adicione a transformação do Processador de Dados ao mapeamento. A porta de entrada da transformação deve permanecer definida como entrada binária, a configuração padrão para a entrada Avro.
4. Vincule a porta de saída do leitor de arquivos complexos à porta de entrada da transformação do Processador de Dados. A porta de saída do leitor de arquivos complexos deve permanecer definida como saída binária.

Configurar uma Transformação com a Saída Avro

Para criar uma transformação do Processador de Dados para uma saída Avro, use o assistente de transformação do Processador de Dados. O assistente cria a transformação no repositório do Modelo com os componentes necessários para transformar a saída Avro. Use o editor de XMap para editar o XMap, se estiver incluído. Adicione a transformação a um mapeamento com um gravador de arquivos complexos.

1. Crie a transformação do Processador de Dados com o assistente de Nova Transformação. Adicione um esquema Avro ou um arquivo de exemplo que defina a estrutura de saída esperada.
2. Se a transformação tiver XML, JSON ou outro formato estruturado como entrada, use o editor de XMap para editar e personalizar o XMap que o assistente criou na transformação.
3. Para configurar as configurações de saída para a transformação do Processador de Dados, no painel **Controle de Saída** da exibição **Configurações**, selecione a opção de saída relevante na área **Modo de coleta de porta de saída binária**.

Nota: Para configurar a saída para enviar um registro de cada vez com o esquema, selecione a opção **Linha de saída para cada linha**. Para coletar todos os registros com o esquema em um fluxo de saída, selecione a opção **Coletar linhas de entrada em uma única saída**.

4. Adicione uma transformação do Processador de Dados a um mapeamento. A transformação permanece definida como saída binária, a configuração padrão para a saída Avro.
5. Crie e vincule um gravador de arquivos complexos para a transformação do Processador de Dados no mapeamento para receber a saída Avro da transformação.

Nota: A configuração de entrada permanece definida como binária, a configuração padrão. O gravador de arquivos complexos não oferece suporte de compactação para a saída Avro.

6. Configure o Gravador de Arquivos Complexos. Na Operação de Objeto de Dados, na guia **Avançado**, para o **Formato do Arquivo**, selecione **Saída Personalizada**. Para **Formato de Saída**, insira `com.informatica.avro.XMLToAvro`.

Biblioteca de Processamento COBOL

A biblioteca COBOL transforma dados COBOL em XML e dele. Ao usar o assistente para criar uma transformação com entrada ou saída COBOL, você seleciona um copybook COBOL para definir a estrutura esperada dos dados de entrada ou saída.

Quando você cria uma transformação de Processador de Dados com entrada ou saída COBOL com o assistente, o Developer adiciona os seguintes objetos à transformação:

- Um objeto de esquema que define uma representação XML da estrutura de dados COBOL.
- Para a entrada COBOL, o Developer adiciona um Analisador que transforma dados de entrada da definição de dados COBOL em XML.
- Para saída COBOL, o Developer adiciona um Serializador que transforma XML em COBOL.

Nota: Você pode criar uma transformação de Processador de Dados que usa entrada ou saída COBOL, mas não ambos. Para processar o COBOL codificado em EBCDIC, certifique-se de alterar as configurações de codificação da transformação de Processador de Dados para EBCDIC.

Criando uma Transformação para COBOL

Use o assistente de transformação do Processador de Dados para criar uma transformação do Processador de Dados com entrada ou saída COBOL.

1. Na Developer tool, clique em **Arquivo > Nova > Transformação**.
2. Selecione a transformação do Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do Repositório do Modelo para inserir a transformação.
4. Selecione **Criar um processador de dados usando um assistente** e clique em **Avançar**.
5. Selecione um formato de entrada e clique em **Avançar**.
6. Se você selecionar COBOL como um formato de entrada, procure até selecionar um copybook COBOL. Clique em **Avançar**.
O arquivo de especificação de copybook geralmente tem uma extensão *.txt. O Developer adiciona um arquivo de esquema XSD que representa o copybook para o repositório do Modelo.
7. Selecione um formato de saída e clique em **Avançar**.
8. Se você selecionar COBOL como um formato de saída, procure até selecionar um copybook COBOL. Clique em **Avançar**.
Se você selecionou COBOL como o formato de entrada, não tem a opção para selecionar COBOL como o formato de saída.
9. Clique em **Concluir**.
A Developer tool cria a transformação no repositório. A exibição **Visão Geral** aparece na Developer tool.
10. Na exibição **Objetos**, clique duas vezes no Analisador para abri-lo no Editor do IntelliScript.
11. Se os dados COBOL forem codificados em EBCDIC, na exibição **Configurações**, altere a codificação de entrada ou saída para a página de códigos EBCDIC relevante.

Definições de Dados COBOL

O copybook COBOL que você usa para criar uma transformação de Processador de Dados pode conter definições de dados de qualquer complexidade. O copybook e a entrada COBOL devem estar de acordo com as regras de definição de dados descritas nesta seção.

Definições de Dados com Suporte

A importação COBOL dá suporte a definições de dados de qualquer complexidade. Por exemplo, as definições de dados podem usar os tipos de dados decimal compactado (COMP-3), binário (COMP-1, COMP-2 ou COMP-4) e ponto decimal lógico (99v99). Elas podem conter recursos, como cláusulas REDEFINES, OCCURS e OCCURS DEPENDING ON.

Regras de Definição de Dados

Uma definição de dados COBOL deve estar de acordo com as seguintes regras:

- Não há mais de 72 caracteres para cada linha e não há texto após a coluna 72
- A primeira linha deve ser um comentário, com um * na coluna 7, ou deve começar com um número de nível
- O primeiro número de nível deve estar na coluna 1 ou 8.

Definições de Dados sem Suporte

A transformação de Processador de Dados não dá suporte às seguintes definições de dados COBOL:

- Os números de nível especiais 66, 77 e 88
- Cláusulas USAGE em um nível de grupo
- Cláusulas INDEXED BY
- POINTER e PROCEDURE-POINTER

Procedimentos de Teste

Quando você testa o Analisador de COBOL, você pode transformar os dados COBOL de exemplo em XML e verificar a saída. Depois de testar o Analisador, você pode executar o serializador COBOL na saída do Analisador.

Testando um Analisador de COBOL

Para testar o Analisador de COBOL, você precisa de um arquivo de entrada que contenha os dados COBOL de exemplo. A estrutura de dados deve estar em conformidade com a definição de dados que você importou. O Analisador transforma os dados COBOL de exemplo em XML e você verifica a saída.

1. Na exibição **Objeto**, clique duas vezes no Analisador de COBOL.
O Analisador é exibido no painel de script do Editor do IntelliScript.
2. Clique com o botão direito no nome do Analisador e clique em **Definir como Componente de Inicialização**.
3. Expanda a árvore do IntelliScript e edite a propriedade `example_source` do Analisador. Altere o respectivo valor de `Text` para `LocalFile`.
O assistente configura o Analisador de COBOL de uma maneira que não exige um exemplo de documento de origem. Quando você conclui o teste, pode remover a origem de exemplo. A origem de exemplo não tem efeito sobre a transformação no tempo de execução.
4. Para atribuir um arquivo de exemplo, expanda o componente `LocalFile` clicando nas setas duplas à direita >>. Clique duas vezes na propriedade `file_name` e navegue para o arquivo de entrada que contém os dados COBOL de exemplo.
5. Na exibição **Visualizador de Dados**, no painel **Entrada**, você pode examinar o arquivo de exemplo. Se o documento não for exibido, clique com o botão direito no nome do Analisador no IntelliScript e clique em **Abrir Origem de Exemplo**.

6. Clique em **Executar > Executar Visualizador de Dados** para testar o Analisador.
7. Na exibição **Visualizador de Dados** do painel **Saída**, examine a saída do Analisador.
8. Para confirmar que o Analisador foi executado sem erros, na exibição **Visualizador de Dados** do painel **Saída**, clique no botão **Mostrar Eventos**. Examine o log de execução na exibição **Eventos do Processador de Dados**.

Testando um Serializador COBOL

Depois de testar um Analisador de COBOL, você pode executar o Serializador COBOL na saída do Analisador.

1. No Explorador de Data Transformation, clique duas vezes no arquivo de script TGP do Serializador. O Serializador será exibido no painel de script do Editor do IntelliScript.
 2. Clique com o botão direito no nome do Serializador e clique em **Definir como Componente de Inicialização**.
 3. Clique em **Executar > Executar** para ativar o Serializador. No prompt, navegue até o arquivo `Results\output.xml` que você gerou quando executou o Analisador.
 4. Examine o log de execução na exibição **Eventos**. Confirme que o Serializador foi executado sem erros.
 5. Para exibir a saída do Serializador, clique duas vezes no arquivo `Results\output.xml` no Explorador do Data Transformation.
- A exibição deve ser a mesma da entrada original na qual você executou o Analisador.

Editando uma Transformação para COBOL

Você pode editar uma transformação para COBOL gerada com o assistente de transformação de Processador de Dados.

Se você fizer isso, documente sua edição. A documentação pode ser essencial quando você consulta posteriormente a definição de dados COBOL, reimporta-a para uma nova transformação e precisa reproduzir a edição.

Otimizando o processamento do arquivo COBOL de grande volume no ambiente do Hadoop

Você pode otimizar como um mapeamento com um leitor de arquivos complexos e uma transformação do Processador de Dados processa arquivos COBOL de grande volume no ambiente do Hadoop.

Para otimizar o processamento do arquivo COBOL de grande volume, você deve ser capaz de usar uma expressão regular para dividir os registros. Se o arquivo COBOL puder ser dividido com uma expressão regular, você poderá definir um parâmetro de entrada para o leitor de arquivos complexos que fornece uma expressão regular que determina como dividir o processamento de registro no ambiente do Hadoop.

Configurando o leitor de arquivos complexos para COBOL

Adicione uma transformação do Processador de Dados que transforma a entrada de COBOL em um mapeamento com um leitor de arquivos complexos. Configure o leitor de arquivos complexos para otimizar como o mapeamento processa a entrada de COBOL em um ambiente de Hive. A codificação de entrada para o leitor de arquivos complexos deve ser EBCDIC.

1. No editor de Mapeamento, crie um objeto de leitor de arquivos complexos.

2. Para configurar o leitor de arquivos complexos, realize as seguintes etapas:
 - a. Na guia **Avançado** da exibição **Propriedades**, selecione a propriedade **Formato de Arquivo** e escolha **Formato de Entrada**.
 - b. Selecione a propriedade **Formato de Entrada** e digite
`com.informatica.hadoop.reader.RegexInputFormat`.
 - c. Para otimizar o desempenho, use a propriedade **Parâmetros do Formato de Entrada** para definir a expressão regular no formato `Regex="<expressão regular>"`.
3. Crie uma transformação do Processador de Dados para a entrada de COBOL usando o assistente.
4. Adicione a transformação do Processador de Dados ao mapeamento. A porta de entrada da transformação deve permanecer definida como entrada binária, a configuração padrão para a entrada de COBOL.
5. Vincule a porta de saída do leitor de arquivos complexos à porta de entrada da transformação do Processador de Dados. A porta de saída do leitor de arquivos complexos deve permanecer definida como saída binária.

JSON

Quando você cria uma transformação de Processador de Dados com entrada ou saída JSON, pode selecionar um esquema JSON ou um arquivo de amostra para a transformação. O arquivo de esquema ou de exemplo define a estrutura esperada da hierarquia de dados de entrada ou de saída.

JavaScript Object Notation (JSON) é um formato de intercâmbio hierárquico de dados semelhante ao XML. O formato JSON é normalmente usado para transmitir dados estruturados por meio de uma conexão de rede. Um Mapeador ou um Serializador usa um esquema de entrada ou um documento de entrada JSON da mesma maneira que um esquema de entrada e um documento de entrada XML para definir a hierarquia de dados de entrada esperada. Um Analisador usa um esquema de saída ou documento de saída JSON para definir a hierarquia de dados de saída esperada.

Quando você usa o assistente de transformação do Processador de Dados para criar uma transformação com entrada ou saída JSON, a transformação pode conter um Analisador, um Mapeador, um Transformador ou um Serializador associado à hierarquia JSON. O esquema JSON é convertido em um arquivo .xsd que define a estrutura hierárquica do arquivo JSON. Você pode usar esse esquema .xsd com qualquer documento JSON que tenha o mesmo formato hierárquico.

Esquemas JSON

Quando você cria uma transformação de Processador de Dados por meio do assistente, pode usar um esquema JSON ou uma origem de exemplo para definir a hierarquia de entrada ou saída JSON.

O assistente de transformação de Processador de Dados gera um esquema XML no repositório do Modelo que especifica a estrutura JSON utilizada pelos componentes da transformação. A transformação contém um transformador associado ao esquema e pode conter outros componentes, dependendo da entrada ou saída selecionada no assistente.

Os scripts utilizam esquemas para definir as estruturas hierárquicas de entrada e saída. Um esquema de entrada JSON deve estar de acordo com o Internet Draft do Esquema JSON, publicado pela Internet Engineering Task Force.

Para obter mais informações sobre a sintaxe do esquema JSON, consulte os seguintes sites:

- Introdução ao JSON: <http://www.json.org>
- Converte um documento JSON em um Esquema JSON: <http://jsonschema.net>

Exemplo de Esquema JSON

A seguir está um exemplo de esquema JSON:

```
{ "type": "object",
  "$schema": "http://json-schema.org/draft-03/schema",
  "id": "#",
  "required": false,
  "properties": {
    "OrgId": {
      "type": "string",
      "id": "OrgId",
      "required": false
    },
    "metrics": {
      "type": "array",
      "id": "metrics",
      "required": false,
      "items": {
        "type": "object",
        "id": "0",
        "required": false,
        "properties": {
          "name": {
            "type": "string",
            "id": "name",
            "required": false
          },
          "valueTrend": {
            "type": "array",
            "id": "valueTrend",
            "required": false,
            "items": {
              "type": "object",
              "id": "0",
              "required": false,
              "properties": {
                "date": {
                  "type": "string",
                  "id": "date",
                  "required": false
                },
                "val": {
                  "type": "string",
                  "id": "val",
                  "required": false
                }
              }
            }
          }
        }
      }
    }
  }
}
```

O esquema define os elementos e atributos que podem ocorrer em um documento JSON. O esquema usa a sintaxe JSON para especificar a hierarquia e a sequência de elementos, se os elementos são obrigatórios, o tipo de elemento e os valores possíveis.

O exemplo de esquema anterior define o seguinte documento de entrada JSON:

```
{ "OrgId": "ORG0000000000001",
  "metrics": [
```

```

{
  "name": "COL1",
  "valueTrend": [
    {
      "date": "2011-11-01",
      "val": "122.456"
    },
    {
      "date": "2011-11-02",
      "val": "215.1"
    }
  ]
},
{
  "name": "COL2",
  "valueTrend": [
    {
      "date": "2011-11-01",
      "val": "122.456"
    },
    {
      "date": "2011-11-02",
      "val": "215.1"
    }
  ]
}
]
}

```

Se você rastrear pelo esquema, poderá determinar o relacionamento entre os elementos do esquema e o documento de entrada.

A hierarquia de esquemas contém o objeto `metrics` que aninha a matriz `valueTrend`. A matriz contém os campos `date` e `val` que são do tipo de dados `string`.

Criando uma Transformação com JSON

Crie uma transformação de Processador de Dados com entrada ou saída JSON.

1. Na ferramenta Developer, clique em **Arquivo > Nova > Transformação**.
2. Selecione a transformação de Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do Repositório do Modelo para inserir a transformação.
4. Selecione **Criar um processador de dados usando um assistente** e clique em **Avançar**.
5. Selecione um formato de entrada e clique em **Avançar**.
6. Se você selecionar JSON como um formato de entrada, procure até selecionar um esquema JSON ou arquivo JSON de exemplo. Clique em **Avançar**.

Normalmente, o arquivo JSON tem uma extensão `*.json`. O Developer adiciona um arquivo de Esquema XSD que representa a hierarquia JSON para o Repositório do Modelo.

7. Selecione um formato de saída e clique em **Avançar**.
8. Se você selecionar JSON como um formato de saída, procure até selecionar um esquema JSON ou arquivo JSON de exemplo. Clique em **Avançar**.

Se você selecionou JSON como o formato de entrada, não terá a opção para selecionar JSON como o formato de saída.

9. Clique em **Concluir**.

A ferramenta Developer cria a transformação no repositório. A exibição **Visão Geral** aparece na ferramenta Developer.

10. Na exibição **Objetos**, clique duas vezes no componente de transformação para abri-lo no Editor do IntelliScript.

Parquet

Use o assistente para criar uma transformação com a entrada ou a saída Parquet. Quando você cria uma transformação do Processador de Dados para transformar o formato Parquet, selecione um esquema Parquet ou um arquivo de exemplo que defina a estrutura esperada dos dados Parquet. O assistente cria componentes que transformam o formato Parquet em outros formatos ou vice-versa. Depois que o assistente criar a transformação, você poderá configurar melhor a transformação para determinar a lógica de mapeamento.

O Apache Parquet é um formato de armazenamento em colunas que pode ser processado em um ambiente Hadoop. O Parquet é implementado para lidar com estruturas de dados complexos aninhados e usar um algoritmo de retalhamento e montagem de registros. Para obter mais informações sobre Parquet, consulte <http://parquet.incubator.apache.org/documentation/latest/>.

Uma transformação que lê a entrada ou a saída Parquet se conta com um esquema. Quando a transformação lê ou grava dados Parquet, a transformação usa o esquema para interpretar a hierarquia.

Depois de criar uma transformação do Processador de Dados para a entrada Parquet, adicione-a a um mapeamento com um leitor de arquivos complexos. O leitor de arquivos complexos transfere a entrada Parquet para a transformação. Para uma transformação do Processador de Dados com saída Parquet, adicione um gravador de arquivos complexos ao mapeamento para receber a saída da transformação.

Criando uma Transformação com Entrada ou Saída Parquet

Crie uma transformação do Processador de Dados com entrada ou saída Parquet.

1. Na ferramenta Developer, clique em **Arquivo > Novo > Transformação**.
2. Selecione a transformação do Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do repositório do Modelo para inserir a transformação.
4. Selecione a opção **Criar um processador de dados usando um assistente** e clique em **Avançar**.
5. Selecione um formato de entrada e clique em **Avançar**.
6. Se você selecionar Parquet como um formato de entrada, procure para selecionar um esquema Parquet ou um arquivo Parquet de amostra. Clique em **Avançar**.

A ferramenta Developer adiciona um arquivo de objeto de esquema que representa a hierarquia Parquet ao repositório do Modelo.

7. Selecione um formato de saída e clique em **Avançar**.
8. Se você selecionar Parquet como um formato de saída, procure e selecione um esquema Parquet ou um arquivo Parquet de amostra. Clique em **Avançar**.

Se você tiver selecionado Parquet como o formato de entrada, não terá a opção de selecionar Parquet como o formato de saída.

9. Clique em **Concluir**.

A ferramenta Developer cria a transformação no repositório. A exibição **Visão Geral** aparece na ferramenta Developer.

10. Na exibição **Objetos**, clique duas vezes no componente de transformação para abri-lo no editor do IntelliScript.

Para configurar o componente de transformação, adicione instruções de mapeamento.

Configurar o leitor de arquivos complexos da entrada do parquet

Depois de criar uma transformação do Processador de Dados que converte a entrada Parquet, adicione a transformação a um mapeamento com um leitor de arquivos complexos. Configure o leitor de arquivos complexos para processar a entrada Parquet.

1. No editor de Mapeamento, crie um objeto de leitor de arquivos complexos.
2. Para configurar o leitor de arquivos complexos, realize as seguintes etapas:
 - a. Na guia **Avançado** da exibição **Propriedades**, selecione a propriedade **Formato de Arquivo** e escolha **Formato de Entrada**.
 - b. Na guia **Avançado**, selecione a propriedade **Formato de Entrada** e digite `com.informatica.parquet.ParquetToXML`.
 - c. Para otimizar o desempenho, use a propriedade **Parâmetros do Formato de Entrada** para ajustar o parâmetro `MaxOutputAccumulation`. Por padrão, o parâmetro `MaxOutputAccumulation`, que define o número esperado de registros de saída, é definido como 50.000. Para alterar a configuração para 250.000 por exemplo, insira `"MaxOutputAccumulation"="250000"`.
 - d. Por padrão, o leitor de arquivos complexos adiciona o esquema à saída do leitor de arquivos complexos em um único elemento diretamente depois do elemento raiz. Para não adicionar o esquema à saída, selecione a propriedade **Parâmetros do Formato de Entrada** e, em seguida, digite `"InjectSchema"="false"`.

Use um ponto e vírgula para separar vários parâmetros, por exemplo `"MaxOutputAccumulation"="250000"; "InjectSchema"="false"`.
3. Adicione a transformação do Processador de Dados ao mapeamento. A porta de entrada da transformação deve permanecer definida como entrada binária, a configuração padrão para a entrada Parquet.
4. Vincule a porta de saída do leitor de arquivos complexos à porta de entrada da transformação do Processador de Dados. A porta de saída do leitor de arquivos complexos deve permanecer definida como saída binária.

Configurar uma transformação com saída Parquet

Para criar uma transformação de Processador de Dados para uma saída Parquet, use o assistente de transformação de Processador de Dados. Esse assistente cria a transformação no repositório do Modelo com os componentes necessários para transformar a saída Parquet. Use o editor de XMap para editar o XMap, se estiver incluído. Adicione a transformação a um mapeamento com um gravador de arquivos complexos.

1. Crie a transformação do Processador de Dados com o assistente de Nova Transformação. Adicione um esquema Parquet ou um arquivo de exemplo que defina a estrutura de saída esperada.
2. Se a transformação tiver XML, JSON ou outro formato estruturado como entrada, use o editor de XMap para editar e personalizar o XMap que o assistente criou na transformação.
3. Para configurar as configurações de saída para a transformação do Processador de Dados, no painel **Controle de Saída** da exibição **Configurações**, selecione a opção de saída relevante na área **Modo de coleta de porta de saída binária**.

Nota: Para configurar a saída para enviar um registro de cada vez com o esquema, selecione a opção **Linha de saída para cada linha**. Para coletar todos os registros com o esquema em um fluxo de saída, selecione a opção **Coletar linhas de entrada em uma única saída**.

4. Adicione uma transformação do Processador de Dados a um mapeamento. A transformação permanece definida como saída binária, que é a configuração padrão para a saída Parquet.
5. Crie um gravador de arquivos complexos e vincule-o à transformação de Processador de Dados no mapeamento para receber a saída Parquet desse transformação.

Nota: A configuração de entrada permanece definida como binária, a configuração padrão. O gravador de arquivos complexos não oferece suporte a compactações para a saída Parquet.

6. Configure o Gravador de Arquivos Complexos. Na Operação de Objeto de Dados, na guia **Avançado**, para o **Formato do Arquivo**, selecione **Saída Personalizada**. Para **Formato de Saída**, insira `com.informatica.parquet.XMLToParquet`.

XML

Use o assistente para criar uma transformação com a entrada ou a saída XML. Quando você criar uma transformação do Processador de Dados com uma entrada ou uma saída XML, selecione um esquema .xsd ou um arquivo de exemplo XML para a transformação. O arquivo de esquema ou de exemplo define a estrutura esperada da hierarquia de dados de entrada ou de saída. Se você selecionar um arquivo de exemplo, o assistente criará um esquema a partir da hierarquia de dados do arquivo de exemplo.

O assistente cria uma transformação do Processador de Dados que pode conter um Analisador, um Mapeador, um XMap ou um Serializador associado à hierarquia XML. Um Mapeador, um XMap ou um Serializador usa um esquema de entrada para definir a hierarquia de dados de entrada esperada. Um XMap, um Mapeador ou um Analisador usa um esquema de saída para definir a hierarquia de dados de saída esperada. Para obter mais informações sobre a sintaxe do esquema, consulte <http://www.w3.org>.

O assistente cria os componentes básicos ou o mapeamento relacional que a transformação necessita com base nos tipos de entrada e de saída. A transformação pode ser concluída ou pode servir como um ponto de partida para fazer mais configurações.

Se a transformação tiver a entrada e a saída estruturadas, o assistente poderá criar um XMap que você configura para transformar dados de uma hierarquia para outra. Use o editor de XMap para vincular os nós de hierarquia do esquema de entrada e de saída, e para definir a lógica de transformação. Para obter mais informações sobre o objeto XMap e o editor de XMap, consulte [“Visão Geral do XMap” na página 85](#).

Se a transformação tiver uma entrada ou uma saída relacional que você deseja transformar de ou para dados estruturados, o assistente criará um mapeamento relacional. O painel **Portas** na exibição **Visão Geral** mostra a os nós de esquema hierárquico e as portas relacionais. Use o painel **Portas** para vincular os elementos hierárquicos às portas e aos grupos relacionais. Para obter mais informações sobre a transformação de dados relacionais, consulte [“Visão Geral de Entradas e Saídas Relacionais” na página 66](#).

Criando uma Transformação que Transforma XML

Crie uma transformação do Processador de Dados com entrada XML, saída XML ou ambos.

1. Na ferramenta Developer, clique em **Arquivo > Nova > Transformação**.
2. Selecione a transformação de Processador de Dados e clique em **Avançar**.
3. Insira um nome para a transformação e procure uma localização do Repositório do Modelo para inserir a transformação.

4. Selecione **Criar um processador de dados usando um assistente** e clique em **Avançar**.
5. Selecione XML ou outro formato de entrada e clique em **Avançar**.
6. Se você tiver selecionado XML como o formato de entrada, procure para selecionar um esquema ou arquivo XML de exemplo. Clique em **Avançar**.
O Developer adiciona um arquivo de esquema .xsd que representa a hierarquia do Repositório do Modelo.
7. Selecione XML ou outro formato de saída e clique em **Avançar**.
8. Se você selecionar XML como um formato de saída, procure para selecionar um esquema ou um arquivo XML de exemplo. Clique em **Avançar**.
9. Clique em **Concluir**.
A ferramenta Developer cria a transformação no repositório. A exibição **Visão Geral** aparece na ferramenta Developer.
10. Para editar um componente específico na transformação, na exibição **Objetos**, clique duas vezes no componente de transformação para abri-lo no Editor do IntelliScript.

CAPÍTULO 4

Entrada e Saída Relacionais

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Entradas e Saídas Relacionais, 66](#)
- [Entrada Relacional, 66](#)
- [Saída Relacional, 73](#)

Visão Geral de Entradas e Saídas Relacionais

Uma transformação de Processador de Dados pode ler uma entrada de banco de dados relacional em portas de entrada e transformá-la em outros formatos. Uma transformação pode gerar linhas de dados relacionais para portas de saída. Você pode definir o mapeamento com portas de transformação de Processador de Dados na exibição Visão Geral da transformação. Outra alternativa é usar o assistente de transformação de Processador de Dados para mapear dados relacionais automaticamente.

Você pode transformar dados relacionais em dados hierárquicos. Para transformar grupos de entrada em dados hierárquicos, mapeie os nós do grupo de portas relacionais para as portas hierárquicas. Você pode transmitir os dados das portas de saída hierárquicas para outra transformação no mapeamento.

Você pode retornar saída relacional da transformação de Processador de Dados. Se um componente retornar dados relacionais, crie grupos de portas de saída mapeando nós da entrada hierárquica para grupos de portas relacionais. Você pode transmitir os dados das portas de saída relacionais para outra transformação em um mapeamento.

Entrada Relacional

Uma transformação do Processador de Dados pode converter uma entrada relacional em uma saída hierárquica. Um banco de dados relacional contém uma coleção de tabelas de dados, organizadas de acordo com um modelo relacional. Tabelas podem ter relacionamentos adicionais entre si.

No modelo relacional, cada esquema da tabela identifica uma coluna chamada chave primária, para identificar exclusivamente cada linha. Você identifica o relacionamento entre cada linha da tabela e uma linha em outra tabela com uma chave externa. Uma chave externa é uma coluna em uma tabela que aponta para a chave primária de outra tabela.

Para converter dados de porta relacional em dados hierárquicos, você deve definir a estrutura do mapeamento com base em um esquema hierárquico. Importe um esquema para o repositório do Modelo.

Após a importação desse esquema, é possível exibir seus componentes na Developer tool. Se o esquema hierárquico tiver vários elementos que podem ser um elemento raiz, escolha um nó para ser o elemento raiz.

No painel **Portas** da exibição **Visão Geral**, é possível mapear as portas de entrada relacionais para nós de esquema. O lado esquerdo do painel exibe a área **Entrada da transformação**, enquanto o lado direito exibe a área **Entrada do serviço**.

Ao arrastar um nó de um nó de **Entrada do serviço** para uma porta de **Entrada da transformação**, você mapeia de um nó de esquema para um nó de entrada relacional. A Developer tool cria as portas de entrada para mapear os dados. É possível definir grupos, definir portas e mapear nós das portas de entrada para as portas de saída.

Uma transformação do Processador de Dados com entrada relacional pode conter portas de passagem. Adicione portas de passagem ao grupo raiz da estrutura relacional.

Configuração de Portas de Entrada Relacionais

Para mapear portas de entrada relacionais para a saída hierárquica, selecione um esquema para definir a saída. Você também deve definir as portas de entrada relacionais. No painel **Portas**, crie e defina grupos de portas e mapeie os nós dos nós hierárquicos para essas portas.

Para definir um mapeamento de entrada, selecione o **Modo de processador de dados** e defina-o como **Mapeamento de Entrada** ou **Mapeamento de Entrada e Serviço**. O mapeamento usa um esquema para definir a saída. Se o esquema tiver mais de um elemento que possa ser um elemento raiz, você poderá escolher um nó do esquema para ser o elemento raiz.

Para definir um nó como uma raiz, clique em **Escolher Hierarquia**. A ferramenta Developer exibe apenas os nós do nível raiz e abaixo dele na área **Saída da transformação**. Clique em **Mostrar como Hierarquia** para exibir os nós de saída em uma hierarquia. Cada grupo filho é exibido embaixo do grupo pai.

Crie e defina portas de entrada relacionais com um dos seguintes métodos:

Arrastar nós até portas

Arraste nós da área **Saída da transformação** até a área **Entrada da transformação**. Se você arrastar um nó para um grupo, a ferramenta Developer adicionará uma porta ao grupo. Caso contrário, ela criará um grupo com a porta.

Criar manualmente as portas

Para criar uma porta, selecione um campo vazio na área **Entrada da transformação** e clique em **Novo > Campo**. Se você não selecionar um campo dentro de um grupo, a ferramenta Developer criará um grupo e adicionará a porta ao grupo.

Criar automaticamente as portas

Clique em **Mapear Automaticamente**. A ferramenta Developer cria grupos de entrada e adiciona portas a esses grupos com base na hierarquia de saída.

Quando você arrasta nós à área **Entrada da transformação**, a ferramenta Developer atualiza o campo de localização com a localização do nó na hierarquia. Se você criar portas manualmente, deverá mapear um nó para a porta. Clique na coluna **Localização** e selecione um nó da lista.

Quando você arrastar um nó com ocorrências múltiplas para um grupo que contém o elemento pai, poderá configurar o número de ocorrências do elemento filho para incluir. Ou você poderá substituir o grupo pai pelo grupo filho com ocorrências múltiplas na saída de transformação.

Para criar um grupo relacional, arraste um nó de saída até uma coluna vazia na área **Entrada da transformação**. Se você arrastar um nó filho de ocorrência múltipla até uma coluna de entrada vazia, a ferramenta Developer solicitará que você relacione o grupo a outros grupos. Quando você seleciona um grupo, a ferramenta Developer cria chaves para relacionar os grupos.

Você também pode criar um novo grupo relacional clicando em **Novo > Grupo** na área **Entrada da transformação**. Insira um nome para o grupo. Configure grupos relacionados de portas de entrada na área **Entrada da transformação**. Quando a ferramenta Developer solicita que você relacione grupos de saída, ela adiciona as chaves aos grupos. Você também pode adicionar portas manualmente para representar chaves.

Para exibir as linhas que conectam as portas aos nós hierárquicos, clique em **Mostrar Linhas**. Selecione para exibir todas as linhas de conexão ou as linhas de portas selecionadas.

Diretrizes para Vincular Portas de Entrada

Quando você usar o assistente de Nova Transformação para gerar automaticamente uma transformação do Processador de Dados, a Developer tool criará portas relacionais com base na hierarquia de esquema e as vinculará aos nós hierárquicos.

Considere as seguintes regras e diretrizes quando você vincular portas de entrada relacionais a nós de saída hierárquicos:

- Você pode vincular uma porta relacional de entrada a um nó na hierarquia.
- Vincule uma chave primária do elemento ou atributo relevante na hierarquia a um grupo relacional na entrada. A chave primária identifica cada linha nas tabelas relacionais.
- Vincule uma chave externa do elemento ou atributo relevante na hierarquia a um grupo relacional na entrada. Uma chave externa na entrada relacional é uma coluna em uma tabela que aponta para a chave primária de outra tabela.
- A porta relacional de entrada e o nó hierárquico devem ter tipos de dados compatíveis.
- As Chaves podem ser do tipo string ou do tipo integer.

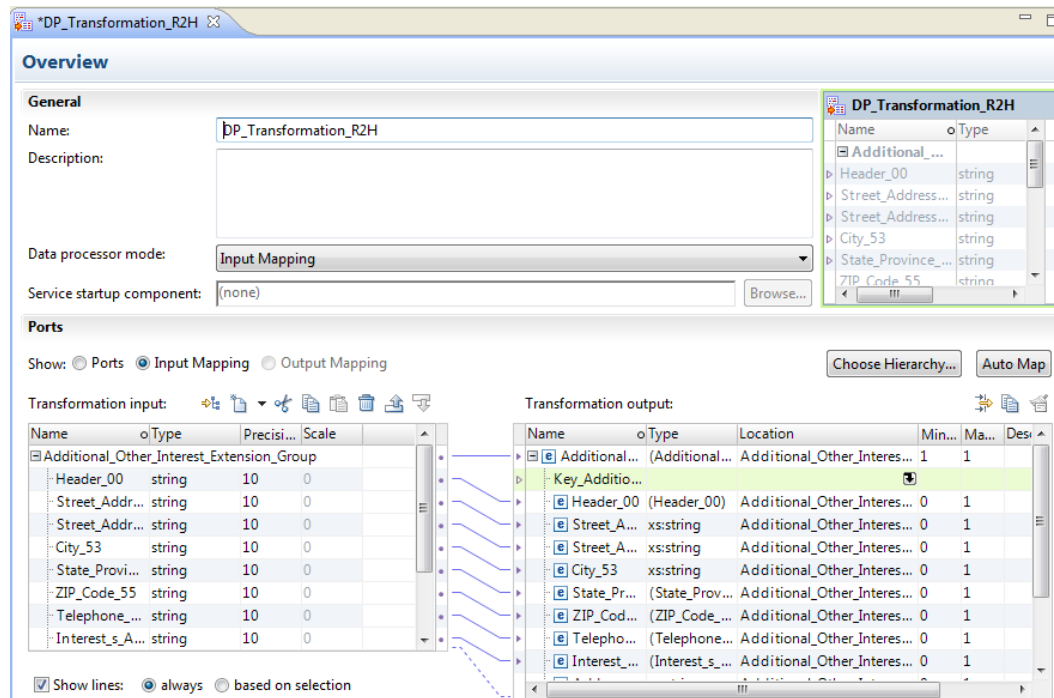
Definir Portas Relacionais de Entrada com a Exibição Visão Geral

Para transformar dados relacionais em dados hierárquicos na transformação do Processador de Dados, mapeie os nós do grupo de portas relacionais para as portas hierárquicas. Use a exibição Visão Geral para vincular portas relacionais a portas hierárquicas.

Para transformar a entrada relacional em saída hierárquica, ative a entrada relacional na exibição **Visão Geral**. A ferramenta Developer remove a porta de entrada padrão da exibição.

Selecione **Mapeamento de Entrada**. O painel **Portas** aparece na exibição **Visão Geral**.

A imagem mostra o painel **Portas**:



À esquerda do painel **Portas** se encontra a área **saída de Transformação**, que contém os nós de esquema hierárquico. À direita, está a área **Entrada da transformação**, que contém os grupos e elementos relacionais.

Você pode criar portas na área **Entrada da transformação** e vincular elementos relacionais aos nós de esquema. Também é possível arrastar o ponteiro de um nó no esquema até um campo vazio na área **Entrada da transformação** para criar uma porta. Quando você conecta uma porta relacional a um nó do esquema, a ferramenta Developer mostra um link entre eles.

Portas Clustering_Key

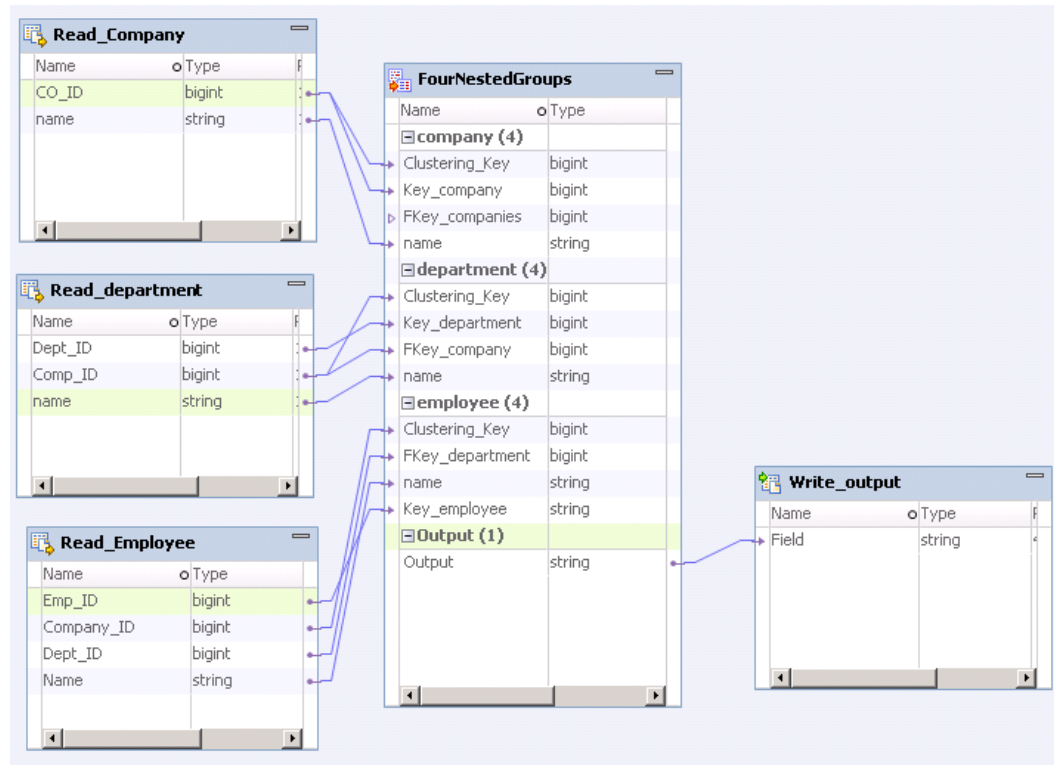
Quando você criar uma transformação do Processador de Dados de relacional para hierárquica com vários grupos no ambiente Hive, ative o particionamento dos dados de entrada para garantir que os dados de cada linha sejam processados corretamente. O Sistema de Integração de Dados particiona as linhas de entrada de acordo com uma porta que funciona como chave de particionamento de nome Clustering_Key.

Para particionar dados de entrada para uma transformação do Processador de Dados em um mapeamento, selecione a transformação no mapeamento e, na guia **Avançado** da exibição **Propriedades**, selecione para ativar o particionamento. Quando você ativa o particionamento, o Developer cria uma porta Clustering_Key na transformação do Processador de Dados para cada grupo de entrada.

Cada grupo de entrada deve usar a mesma chave externa para o grupo raiz de entrada para ajudar no particionamento. Para classificar dados de acordo com uma chave, conecte a porta de entrada relacional da chave externa de cada objeto de Dados à porta Clustering_Key relevante na transformação do Processador de Dados. O Serviço de Integração de Dados usa a Clustering_Key para particionar e processar os dados.

Você deve usar a mesma chave em todos os grupos de entrada relacional. Se necessário, você pode usar uma transformação de Associador para adicionar a chave a um grupo de entrada relacional que não tem essa chave.

A seguinte imagem mostra um mapeamento com a chave externa Company_ID nos grupos de entrada relacional associados às portas Clustering_Key na transformação do Processador de Dados:



Entrada Relacional Normalizada

Quando você normaliza os dados de entrada relacional na saída hierárquica, os valores de dados não se repetem no grupo hierárquico. Crie um relacionamento um para um entre os níveis de hierarquia nos dados de saída hierárquicos e nos grupos de entrada das portas.

Exemplo de Entrada Relacional Normalizada

Você deseja transformar uma entrada relacional com um grupo que contém detalhes de gerentes de várias empresas em hierarquias XML separadas. Na entrada, cada registro de gerente contém detalhes da empresa. Na saída, uma hierarquia XML contém detalhes de empresas, e uma hierarquia XML separada contém detalhes de gerentes.

Na entrada relacional, os elementos Company_ID e Company_Name se repetem para cada gerente na empresa:

Company_ID	Company_Name	Manager_ID	Manager_Name
100	Percy Accounting	76500	Cindy Jacques
100	Percy Accounting	46501	Tom Jorry
100	Percy Accounting	86509	Delilah Smith

Se você definir a saída XML para conter um nível de hierarquia pai Empresa e um nível de hierarquia filho Gerentes, será possível usar os seguintes grupos de hierarquias:

```
Companies
  Company_Key
  Company_ID
```

Company_Name

Managers

Company_Key
Manager_ID
Manager_Name

O elemento Company_Key relaciona a hierarquia Gerentes com a hierarquia Empresas.

Entrada Relacional Dinamizada

Você pode incluir um número específico de elementos com ocorrência múltipla em um grupo de saída.

Para dinamizar elementos de ocorrência múltipla, mapeie o elemento de porta de ocorrência múltipla para nós de saída específicos.

Exemplo de Entrada Relacional Dinamizada

Você deseja transformar uma entrada relacional que contém uma tabela de números de telefone em uma hierarquia XML com elementos separados para diferentes tipos de números de telefone.

Na entrada relacional, o elemento Telephone_type define o tipo de número de telefone listado para cada pessoa:

Telephone_Number	Telephone_Type	Last_Name	First_Name
9173327437	Celular	Sandrine	Jacques
9174562342	Celular	Race	Tom
8484526471	Home	Race	Tom
7023847265	Trabalho	Smith	Delilah
9174596725	Celular	Smith	Delilah

Na hierarquia XML de saída de Telefones, diferentes tipos de números de telefone no grupo pai têm elementos separados:

Telephones

Telephone_Number
Last_Name
First_Name
Work_Telephone
Mobile_Telephone
Home_Telephone

Entrada Relacional Desnormalizada

Você pode desnormalizar a saída hierárquica da entrada relacional. Quando os dados de entrada são desnormalizados, os valores de elementos do grupo pai se repetem em cada elemento filho.

Para desnormalizar dados de entrada, mapeie nós de um grupo de portas para um nível de hierarquia filho. Todos os elementos se repetem no nível de hierarquia filho.

Exemplo de Entrada Relacional Desnormalizada

Você deseja transformar uma entrada relacional com grupos separados para detalhes de gerentes e detalhes de empresas em uma hierarquia JSON que contenha detalhes tanto de gerentes quanto de empresas.

O elemento Company_Name não aparece no grupo com detalhes de gerentes. O elemento Company_ID é a chave externa no primeiro grupo relacional.

Company_ID	Manager_ID	Manager_Name
100	56673	Kathy Jason
100	23501	Jackie Lyons
100	44509	Bob Terrence

O segundo grupo relacional contém detalhes da empresa.

Company_ID	Company_Name
100	Percy Accounting
102	Sandy Auto Sales
410	Movers Inc.

O elemento de Gerentes na saída JSON contém ambos os elementos; Company_ID e Company_Name:

```
Managers
  Company_ID
  Company_Name
  Manager_ID
  Manager_Name
```

Os elementos Company_ID e Company_Name se repetem para cada gerente do departamento.

Mapeando Portas Relacionais para Nós Hierárquicos

No painel Portas, defina grupos de portas e mapeie os nós do esquema hierárquico para essas portas.

1. Para adicionar um esquema, na exibição **Referências**, clique em **Adicionar**. Procure e selecione um esquema.
2. Para criar um mapeamento de entrada, na área **Geral** da exibição **Visão Geral**, selecione **Mapeamento de Entrada** para o Modo do Processador de Dados.
3. Na área **Portas**, selecione **Mapeamento de Entrada**.
4. Selecione um nó como raiz.
5. Para definir um mapeamento automaticamente, clique em **Mapear Automaticamente**. Para definir manualmente um mapeamento, defina uma chave primária para o grupo de entrada raiz e depois defina portas e grupos de entrada.
6. Para adicionar uma porta ou um grupo de entrada à área **Entrada da transformação**, use um dos seguintes métodos:
 - Arraste um nó de grupo hierárquico ou um nó filho na área **Saída da transformação** até uma coluna vazia na área **Entrada da transformação**. Se o nó for um nó de grupo, a ferramenta Developer adicionará um grupo relacional sem portas.
 - Para adicionar um grupo relacional, selecione uma linha e clique com o botão direito do mouse para selecionar **Novo > Grupo**.
 - Para adicionar uma porta relacional, clique com o botão direito do mouse para selecionar **Novo > Campo**.
7. Para mapear nós como uma chave primária, use um dos seguintes métodos:
 - Selecione dois ou mais nós de hierarquia e arraste-os até uma chave na área **Entrada da transformação**.
 - Clique na coluna **Localização** de uma chave na área **Saída da transformação** e selecione a porta de entrada relacional na área **Entrada da transformação**.

8. Para limpar as configurações de nós hierárquicos para localizações de portas, use um dos seguintes métodos:
 - Selecione um ou mais nós na área **Saída da transformação**, clique com o botão direito do mouse e selecione **Limpar**.
 - Selecione uma ou mais linhas que conectem as portas de entrada relacionais aos nós hierárquicos na área **Saída da transformação**, clique com o botão direito do mouse e selecione **Excluir**.

Saída Relacional

Ao configurar a saída relacional, você pode configurar um grupo de saída separado para cada nó de entrada com várias ocorrências. Você também pode criar grupos que contêm dados desordenados. Você pode dinamizar elementos com várias ocorrências e limitar o número de ocorrências em um grupo de saída.

Configuração de Portas de Saída Relacionais

Para transformar uma entrada hierárquica em saída relacional, selecione um esquema para definir os dados hierárquicos. Você também deve definir as portas de saída relacionais. No painel Portas, defina grupos de portas e mapeie os nós do esquema hierárquico para essas portas.

Para definir um mapeamento de saída, selecione o **Modo do processador de dados** e defina-o como **Mapeamento de Saída** ou **Mapeamento de Saída e Serviço**.

O mapeamento usa um esquema para definir a entrada hierárquica. Se o esquema tiver mais de um elemento que possa ser um elemento raiz, escolha um nó para ser o elemento raiz. Para definir um nó como uma raiz, clique em **Escolher Hierarquia**. A ferramenta Developer exibe apenas os nós do nível raiz e abaixo dele na área **Entrada da transformação**.

Clique em **Mostrar Como Hierarquia** para exibir as portas de saída em uma hierarquia. Cada grupo filho é exibido embaixo do grupo pai.

Crie portas com um dos seguintes métodos:

Arrastar nós até portas

Arraste nós da área **Entrada da transformação** até a área **Saída da transformação**. Se você arrastar um nó para um grupo, a ferramenta Developer adicionará uma porta ao grupo. Caso contrário, ela criará um grupo com a porta.

Criar manualmente as portas

Para criar uma porta, selecione um campo vazio na área **Saída da transformação** e clique em **Novo > Campo**. Se você não selecionar um campo dentro de um grupo, a ferramenta Developer criará um grupo e adicionará a porta ao grupo.

Quando você arrasta nós à área **Saída de transformação**, a ferramenta Developer atualiza o campo de localização com a localização do nó na hierarquia. Se você criar portas manualmente, deverá mapear um nó para a porta. Clique na coluna **Localização** e selecione um nó da lista.

Quando você arrastar um nó com ocorrências múltiplas para um grupo que contém o elemento pai, poderá configurar o número de ocorrências do elemento filho para incluir. Ou você poderá substituir o grupo pai pelo grupo filho com ocorrências múltiplas na saída de transformação.

Para criar um grupo, arraste um nó até uma coluna vazia na área **Saída da transformação**. Se você arrastar um nó filho de ocorrência múltipla até uma coluna de entrada ou saída vazia, a ferramenta Developer

solicitará que você relacione o grupo a outros grupos de saída. Quando você seleciona um grupo, a ferramenta Developer cria chaves para relacionar os grupos.

Você também pode criar um novo grupo clicando em **Novo > Grupo**. Insira um nome para o grupo.

Configure grupos relacionados de portas de saída na área **Saída da transformação**. Quando a ferramenta Developer solicita que você relacione grupos de saída, ela adiciona as chaves aos grupos. Você também pode adicionar portas manualmente para representar chaves.

Para exibir linhas que conectam as portas aos nós hierárquicos, clique em **Mostrar Linhas**. Selecione para exibir todas as linhas de conexão ou apenas as linhas das portas selecionadas.

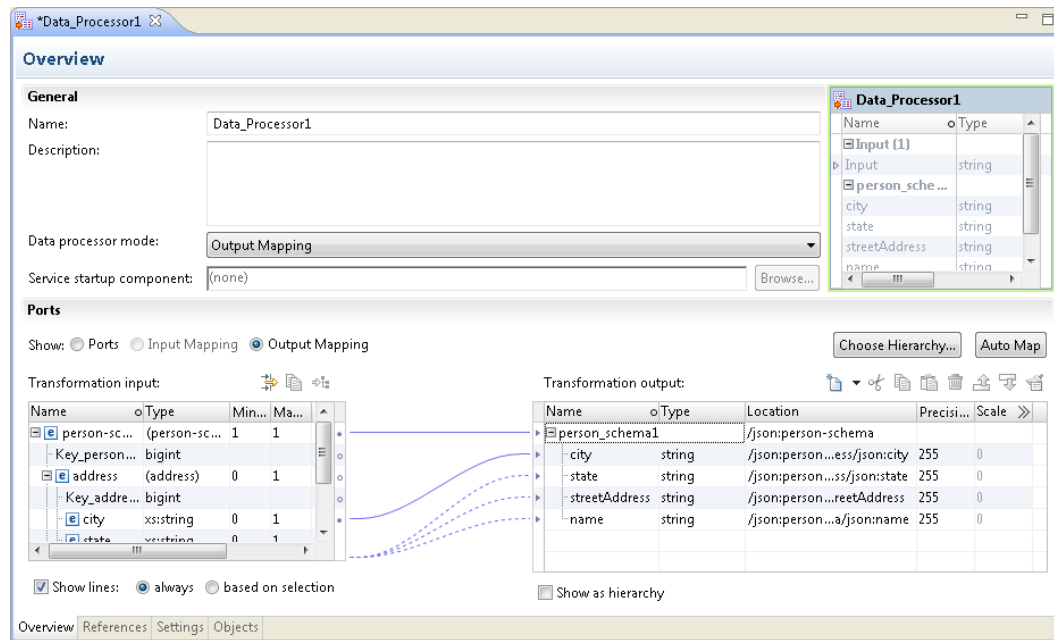
Definir Portas Relacionais de Saída com a Exibição Visão Geral

Para converter dados hierárquicos em saída relacional na transformação do Processador de Dados, vincule os nós hierárquicos às portas relacionais. Use a exibição Visão Geral para vincular portas relacionais a portas hierárquicas. Você cria grupos de portas de saída vinculando nós da saída hierárquica a grupos de portas.

Para retornar portas de grupos relacionais, ative a saída relacional a partir da exibição **Visão Geral**. A ferramenta Developer remove a porta de saída padrão da exibição.

Selecione **Mapeamento de Saída**. O painel **Portas** aparece na exibição **Visão Geral**.

A seguinte imagem mostra o painel **Portas**:



A área **Entrada da transformação**, a qual mostra o esquema de saída, está à esquerda. A área **Saída da transformação**, a qual mostra as portas de saída relacionais, está à direita.

Defina portas de saída relacionais na área **Saída da transformação** e vincule nós do esquema a essas portas. Também é possível arrastar o ponteiro de um nó no esquema até um campo vazio na área **Saída da transformação** para criar uma porta. Quando você arrasta um nó do esquema de saída até uma porta, a ferramenta Developer mostra um link entre eles.

Saída Relacional Normalizada

Quando você criar normalizada dados de saída, os valores de dados não se repetirão em um grupo de saída. Crie um relacionamento um para um entre os níveis de hierarquia nos dados hierárquicos de entrada e nos grupos de saída das portas.

Exemplo de Saída Relacional Normalizada

Você deseja transformar uma hierarquia JSON que define detalhes do departamento e dos funcionários em uma saída relacional com grupos separados para os detalhes do departamento e dos funcionários.

A entrada JSON contém uma hierarquia de Equipe com elementos que contêm os detalhes de departamento e de funcionário.

```
Staff
  Department_ID
  Department_Name
  Employee_ID
  Employee_Name
```

Os seguintes grupos de portas relacionais podem ser criados:

Department_Key	Employee_ID	Employee_Name
100	2673	Jason Stuart
100	1501	Lila Rose
100	4309	Sarah Jacobs

Department_Key	Department_ID	Department_Name
100	1982	Contabilidade
102	3297	Vendas
410	8276	Logística

A Department_Key é uma chave gerada que relaciona o grupo Funcionários a um grupo Departamento na saída.

Saída Relacional Dinâmica

Você pode incluir um número específico de elementos com ocorrência múltipla em um grupo de saída.

Para dinamizar elementos com ocorrências múltiplas, mapeie o elemento filho de ocorrência múltipla para o grupo pai de portas de saída. A ferramenta Developer solicita que você defina o número de elementos filhos para incluir no pai.

Exemplo de Saída Relacional Dinamizada

Você deseja transformar uma hierarquia XML que contém detalhes de funcionários com vários IDs em um grupo de saída relacional com elementos de saída separados para os IDs de funcionários.

O exemplo a seguir mostra duas instâncias de Employee_ID no grupo de saída relacional Departments:

```
Departments
  Department_ID
  Department_Name
  Employee_ID1
  Employee_ID2
```

Saída Relacional Desordenada

Você pode desfazer normalização da saída relacional. Quando você desfaz a normalização dos dados de saída, os valores de elemento do grupo pai se repetem em cada elemento filho.

Para desnormalizar dados de saída, vincule nós do nível de hierarquia pai ao grupo filho de portas de saída.

Exemplo de Saída Relacional Desnormalizada

Você deseja transformar uma hierarquia XML com grupos separados para detalhes de funcionários e detalhes de departamentos em um grupo relacional que contém detalhes tanto de funcionários quanto de departamentos.

As hierarquias XML separam os detalhes do Departamento dos detalhes do Funcionário:

```
Department
  Department_ID
  Department_Name

Employees
  Department_ID
  Employee_ID
  Employee_Name
```

O exemplo a seguir mostra Department_ID e Department_Name no grupo de saída Employees:

Department_ID	Department_Name	Employee_ID	Employee_Name
100	Contabilidade	56500	Kathy Jones
100	Contabilidade	56501	Tom Lyons
100	Contabilidade	56509	Bob Smith

Os elementos Department_ID e Department_Name se repetem para cada funcionário do departamento.

CAPÍTULO 5

Usando o editor do IntelliScript

Este capítulo inclui os seguintes tópicos:

- [Visão geral do Editor do IntelliScript, 77](#)
- [Abrindo um editor do IntelliScript, 78](#)
- [Procedimentos de edição, 79](#)
- [Menus do editor do IntelliScript, 83](#)

Visão geral do Editor do IntelliScript

O editor do IntelliScript é o editor principal no qual é possível configurar um componente de Script, como um Analisador, Mapeador ou Serializador.

O editor do IntelliScript tem uma estrutura em árvore. O nível superior global da árvore mostra os componentes de transformação executáveis, como Analisadores. Esses componentes se chamam executáveis porque podem ser definidos como os componentes de inicialização da transformação.

Também é possível definir componentes não executáveis, como variáveis ou ações, em nível global. Isso permite que eles sejam usados em outros locais do projeto.

Nos níveis aninhados do editor do IntelliScript, é possível definir componentes específicos, como âncoras e ações. Você pode clicar nos símbolos de + ou – para expandir a árvore do editor de IntelliScript e exibir esses níveis aninhados.

Criando um Script

Crie um objeto de Script e defina o tipo de componente de Script a ser criado. Opcionalmente, é possível definir uma referência de esquema e um arquivo de origem de exemplo.

1. Na exibição **Objetos** da transformação do Processador de Dados, clique em **Novo**.
2. Insira um nome para o Script e clique em **Avançar**.
3. Escolha a criação de um Analisador ou de um Serializador. Selecione Outros para criar um componente Mapeador, Transformador ou Streamer.
4. Insira um nome para o componente.
5. Se o componente for o primeiro a processar dados na transformação, habilite **Definir como componente de inicialização**.
6. Clique em **Avançar** se quiser inserir uma referência de esquema para esse Script. Clique em **Concluir** se não quiser inserir a referência de esquema.

7. Se você optar por criar uma referência de esquema, selecione **Adicionar referência a um objeto de esquema** e procure o objeto de Esquema no repositório do Modelo. Clique em **Criar um novo objeto de esquema** para criar o objeto de Esquema no repositório do Modelo.
8. Clique em **Avançar** para inserir uma referência de origem de exemplo ou para inserir um texto de exemplo. Clique em **Concluir** se não quiser definir uma origem de exemplo.
Use uma origem de exemplo para definir dados de amostra e para testar o Script.
9. Se você optar por selecionar uma origem de exemplo, selecione **Arquivo** e procure o arquivo de amostra. Também é possível inserir um texto de amostra na área **Texto**. A Developer tool usa esse texto para testar um Script.
10. Clique em **Concluir**.
A exibição **Script** aparece no editor da Developer tool.

Abrindo um editor do IntelliScript

O editor do IntelliScript exibe um componente de Script na transformação de Processador de Dados. Para abrir um editor do IntelliScript, clique duas vezes em um componente de Script na transformação de Processador de Dados. É possível visualizar os componentes de transformação de Processador de Dados na guia **Estrutura de Tópicos**.

IntelliScript e Visualizador de Dados

O editor do IntelliScript exibe um ou mais componentes de Script. É aqui que você define os componentes da transformação.

É possível visualizar o exemplo de documento de origem de uma transformação na guia **Visualizador de Dados**. Esse painel pode ser usado para ajudar a visualizar, configurar ou testar uma transformação

O painel de exemplo do **Visualizador de Dados** é somente leitura. Não é possível editar o exemplo de documento de origem no Data Transformation Studio.

É possível visualizar o Script no modo de script, com uma representação de código do Script, ou no IntelliMode, usando o Editor do IntelliScript. Por padrão, você visualiza o script no IntelliMode.

Localizando âncoras

Quando você edita um analisador, é fácil encontrar as âncoras correspondentes no IntelliScript e nos painéis de exemplo.

- Clique com o botão direito do mouse em uma âncora no IntelliScript e depois clique em **Exibir Marcação**. Isso localiza a âncora no painel de exemplo.
- Clique em uma âncora no painel de exemplo. A âncora é selecionada automaticamente no IntelliScript.

Componentes e propriedades

O IntelliScript contém dois tipos de itens:

- Componentes. Itens que você pode inserir e excluir na árvore do IntelliScript. Alguns exemplos de componentes são analisadores, serializadores, âncoras, ações e transformadores.

- Propriedades. Itens que você pode editar, mas que não pode inserir ou excluir, exceto inserindo ou excluindo um componente que os contenha. Alguns exemplos são a propriedade `example_source` de um analisador ou a propriedade `search` de uma âncora de marcador.

O editor do IntelliScript exibe os componentes e as propriedades em cores diferentes.

Propriedades básicas e avançadas

Para ajudar a simplificar a exibição, o IntelliScript organiza as propriedades em duas categorias:

- Propriedades básicas. Propriedades que são importantes na maioria dos usos do componente. Em geral, você precisa atribuir essas propriedades para usar o componente.
- Propriedades avançadas. Propriedades que você geralmente não precisa atribuir. Essas propriedades podem ter valores padrão que normalmente não precisam ser alteradas ou podem implementar opções que muitas vezes não precisam ser usadas.

O IntelliScript sempre exibe as propriedades básicas. Ele oculta as propriedades avançadas até você optar por exibi-las.

A distinção é apenas para fins de exibição. Propriedades avançadas são tão fáceis de usar quanto propriedades básicas. Não hesite em usá-las se elas forem necessárias nos seus projetos.

Exibindo as propriedades avançadas de um componente

- Clique no ícone **>>** à direita do nome do componente.
O ícone **>>** muda para **<<**, e as propriedades avançadas são exibidas.

Ocultando as propriedades avançadas

- Clique no ícone **<<** à direita do nome do componente.
O ícone **<<** muda novamente para **>>**, e as propriedades avançadas desaparecem. No entanto, se você tiver atribuído um valor não padrão a uma propriedade avançada, ele permanecerá visível.

Procedimentos de edição

Para editar o IntelliScript, siga os procedimentos descritos nesta seção.

Procedimento básico para edição

Para editar o IntelliScript:

1. Clique no componente ou na propriedade que você deseja editar.
2. Pressione **ENTER** para entrar no modo de edição. Na maioria dos locais, você também pode clicar duas vezes em vez de pressionar **ENTER**.
3. Atribua o nome do componente ou o valor da propriedade.
4. Pressione **ENTER** novamente para concluir a operação de edição.

Copiar e colar

É possível copiar e colar componentes no IntelliScript.

Para copiar vários componentes ao mesmo tempo, pressione **CTRL** ou **SHIFT** enquanto seleciona com o mouse. Os componentes devem estar todos no mesmo nível de aninhamento no IntelliScript.

É possível colar componentes apenas em locais onde eles fazem sentido. Por exemplo, você pode colar âncoras de serialização em um serializador, mas não em um analisador.

Arrastar e soltar

É possível mover componentes de um local para outro arrastando com o mouse. Por exemplo, você pode usar esse método para alterar a sequência de âncoras em um analisador.

Para mover vários componentes, pressione **CTRL** ou **SHIFT** enquanto seleciona com o mouse. Solte **CTRL** ou **SHIFT** e arraste com o mouse.

Para copiar os componentes selecionados, em vez de movê-los, mantenha pressionada a tecla **SHIFT** enquanto arrasta.

Localizar e substituir

Para localizar ou substituir texto no IntelliScript, selecione **Editar > Localizar** ou **Editar > Substituir**.

Inserindo componentes no IntelliScript

Em muitos componentes, o IntelliScript exibe uma linha horizontal, geralmente com um rótulo, como `contém`. Essa linha é seguida por uma seta e três pontos (. . .). É possível inserir componentes aninhados nesses três pontos.

Inserindo um componente

1. Selecione os três pontos e pressione **ENTER**.
Uma lista dos componentes é exibida
2. Selecione um componente na lista.
Como alternativa, comece digitando o nome do componente. O nome será preenchido automaticamente depois que você digitar as primeiras letras.
3. Pressione **ENTER** novamente para concluir a inserção.

Excluindo um componente

- Selecione o componente e pressione **DELETE**.

Editando as propriedades de um componente

1. Selecione o valor de uma propriedade e pressione **ENTER**.
Dependendo do tipo de propriedade, uma caixa de texto, lista ou caixa de diálogo é exibida.
2. Digite ou selecione o novo valor da propriedade.
3. Pressione **ENTER** para concluir a tarefa.

Inserindo tabulações, novas linhas e outros caracteres especiais

Ao atribuir uma propriedade textual, pode inserir caracteres especiais digitando seus códigos numéricos ASCII.

1. Selecione uma propriedade e pressione **ENTER** para iniciar a edição.

2. Pressione **CTRL+A**.

Um pequeno ponto aparece, indicando que o caractere é um código ASCII.

3. Digite o código ASCII de três dígitos. Por exemplo, você pode digitar:

Código ASCII	Caractere
009	Guia
010	Nova linha
013	Retorno de carro

4. Para inserir uma cadeia de códigos ASCII, repita as etapas 2 e 3.

É possível intercalar códigos ASCII e texto regular.

5. Pressione **ENTER** para concluir a edição.

Uma tabulação aparece como um símbolo «. Outros caracteres aparecem como seus códigos de caractere ASCII.

Definindo um componente global

Você pode inserir componentes em um escopo global ou local.

Escopo	Descrição
Escopo global	O componente é definido no nível superior do IntelliScript. Ele pode ser acessado ou usado em qualquer local do projeto.
Escopo local	O componente é definido em um nível aninhado do IntelliScript. Ele pode ser acessado ou usado somente no local aninhado específico.

A maioria dos componentes do Data Transformation pode ser global ou local.

Por exemplo, âncoras são geralmente definidas localmente. Porém, você pode definir uma âncora como um componente global caso queira usar a mesma configuração de âncora em vários analisadores ou várias vezes no mesmo analisador. Em cada local desejado, você pode fazer referência à âncora definida globalmente usando seu identificador.

Dessa forma, um analisador pode usar `MyMarker`, em vez de repetir a configuração da âncora de marcador todas as vezes que ela for necessária. Você pode selecionar `MyMarker` na lista de componentes, no local apropriado dentro do analisador, ou pode arrastar `MyMarker` até o local.

Restrições de nomenclatura

Os nomes que você atribui aos componentes do IntelliScript devem conter somente caracteres ingleses (A-Z, a-z), números (0-9) e sublinhados (_). Eles devem começar com uma letra. Eles podem conter até 127 caracteres.

Exibindo a ajuda sobre um componente














Você pode exibir os tópicos de ajuda online que descrevem um componente ou uma propriedade enquanto edita o IntelliScript.

1. Mostra a exibição **Ajuda**.
2. Selecione o componente ou a propriedade.

A ajuda acessa a localização que descreve o item selecionado.

Ícones do IntelliScript

O IntelliScript exibe cada tipo de componente com um ícone característico. A tabela a seguir descreve os ícones mais comuns que aparecem no IntelliScript.

Ícone	Componente
	Parser
	Serializer
	Mapper
	Transformer
	Marker
	Content ContentSerializer
	Group
	RepeatingGroup
	StringSerializer
	Handle
	Key
	Outras âncoras
	Ações
	Ícone padrão, usado quando não há um ícone específico para um componente

Salvando o IntelliScript

Se um asterisco (*) aparecer na guia de título de um editor do IntelliScript, significa que existem alterações não salvas no editor. Se você tentar fechar o editor ou sair com alterações não salvas, será solicitado que você salve o Script.

Menus do editor do IntelliScript

Clique com o botão direito do mouse em um editor do IntelliScript para exibir um menu. As opções de menu dependem do contexto em que você clica.

Tabela 1. Menu do painel do IntelliScript

Opção	Descrição
Exibir Marcação	Destaca a âncora selecionada no exemplo de origem.
Definir como Componente de Inicialização	Define o componente selecionado como o componente de inicialização do projeto.
Recortar	Permite recortar componentes no IntelliScript.
Copiar	Permite copiar componentes no IntelliScript.
Colar	Permite colar componentes no IntelliScript.
Inserir	Permite inserir componentes no IntelliScript.
Excluir	Permite excluir componentes no IntelliScript.
Tornar Opcional	Seleciona a propriedade <code>optional</code> de um componente. Se um componente for opcional, uma falha nele não causará uma falha em seu componente pai. Para obter mais informações, consulte o <i>Data Transformation Studio</i> .
Tornar Obrigatório	Anula a propriedade <code>optional</code> de um componente. Se um componente for opcional, uma falha nele não causará uma falha em seu componente pai. Para obter mais informações, consulte o <i>Data Transformation Studio</i> .
Ativar	Seleciona a propriedade <code>disabled</code> de um componente. Um componente desativado é ignorado. Esse recurso é útil para desativar um componente temporariamente para testes e depuração.
Desativar	Desmarca a propriedade <code>disabled</code> de um componente. Um componente desativado é ignorado. Esse recurso é útil para desativar um componente temporariamente para testes e depuração.
Modo de Script	O modo de Script exibe o conteúdo bruto do arquivo *.tgp. Esse modo destina-se apenas à resolução avançada de problemas.
IntelliMode	O IntelliMode exibe o IntelliScript em uma representação gráfica legível. Este é o modo ilustrado neste manual e nos outros manuais que fazem parte da documentação do Data Transformation.

Opção	Descrição
Abrir Exemplo de Origem	Abre o exemplo de arquivo de origem do analisador, serializador ou mapeador selecionado.
Criar Serializador	Cria um serializador com base no analisador selecionado. O serializador e o analisador realizam transformações inversas.

Tabela 2. Menu do painel de exemplo

Opção	Descrição
Copiar	Copia uma cadeia para a área de transferência.
Inserir Marcador	Define o texto selecionado como uma âncora de <code>Marcador</code> . A âncora é adicionada ao IntelliScript.
Inserir Conteúdo	Define o texto selecionado como uma âncora de <code>Conteúdo</code> . A âncora é adicionada ao IntelliScript.
Inserir Conteúdo de Deslocamento	Define a localização selecionada como uma âncora de <code>Conteúdo</code> . A âncora é adicionada ao IntelliScript.
Inserir RepeatingGroup	Define o texto selecionado como o separador de uma âncora <code>RepeatingGroup</code> . A âncora é adicionada ao IntelliScript.
Exibir Instância	Localiza a âncora selecionada no IntelliScript.
Exibir Evento	Localiza o evento correspondente na exibição Eventos .
Localizar	Localiza uma cadeia no exemplo de documento de origem.
Codificação Lógica	Se o exemplo de origem contiver texto em um idioma da direita para a esquerda, como o hebraico ou o árabe, esse comando transformará a exibição da esquerda para a direita em uma exibição da direita para a esquerda.
Quebra de Linha	Quebra linhas longas.
Salvar Origem como	Salva o exemplo de origem em uma localização especificada com um nome especificado.

CAPÍTULO 6

XMap

Este capítulo inclui os seguintes tópicos:

- [Visão Geral do XMap, 85](#)
- [Esquemas do XMap, 86](#)
- [Instruções de Mapeamento, 87](#)
- [Expressões XPath, 104](#)
- [Variáveis XMap, 111](#)
- [Exemplo de XMap, 112](#)

Visão Geral do XMap

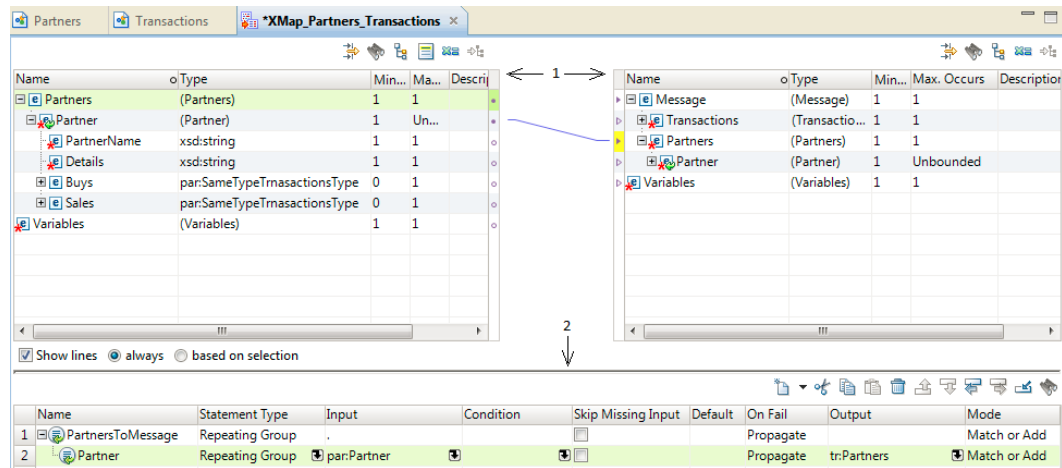
Um XMap é um objeto de transformação do Processador de Dados que altera um documento de entrada hierárquico para outro documento hierárquico com uma estrutura de hierarquia diferente.

Um XMap usa os esquemas de entrada e saída para definir a hierarquia esperada de documentos de entrada e de saída. Use o editor do XMap para definir e gerenciar instruções de mapeamento. O editor de XMap contém a hierarquia de esquema de entrada e a hierarquia de esquema de saída. As instruções de mapeamento vinculam elementos de esquema de entrada a elementos de esquema de saída.

Um XMap pode transformar qualquer documento hierárquico de entrada cujos elementos correspondam à hierarquia de esquema de entrada em um documento de saída com a hierarquia do esquema de saída.

Por exemplo, um XMap pode transformar faturas de cliente em uma lista de pedidos de clientes filtrada pelo mês do ano. A entrada é um documento JSON que contém uma hierarquia de elementos de cliente. A saída é um documento XML que contém uma hierarquia de elementos de mês. O documento XML de saída agrupa os pedidos de clientes por mês e inclui as informações de contato e os totais de pedidos.

A imagem a seguir mostra o editor do XMap.



1. O editor de XMap contém os esquemas hierárquicos de entrada e de saída. Arraste e solte entre os elementos de esquema para criar instruções de mapeamento.
2. A grade do editor do XMap mostra instruções de mapeamento. Use a grade para gerenciar e editar instruções de mapeamento.

Um XMap usa instruções de mapeamento para definir como transformar um elemento de esquema de entrada em um elemento de esquema de saída. Você pode arrastar a partir de um nó no esquema de entrada para um nó no esquema de saída para criar um link. Quando você cria links, eles são instruções de mapeamento. O editor do XMap mostra a instrução de mapeamento na grade.

Você pode editar as instruções de mapeamento na grade. Você pode definir as condições para transformar e filtrar os dados de acordo com diferentes tipos de instruções de mapeamento e expressões XPath. O XPath é uma linguagem de consulta usada para selecionar nós em um documento hierárquico e executar computações.

Você pode usar as expressões do XPath para definir o contexto de uma instrução de mapeamento. Você também pode adicionar vários cálculos aritméticos a uma instrução de mapeamento usando expressões do XPath.

Esquemas do XMap

Um XMap necessita de esquemas hierárquicos que definam as hierarquias hierárquicas de entrada e de saída. Um XMap usa os esquemas hierárquicos de entrada e de saída para determinar qual tipo de dados é esperado no documento de origem de entrada e no documento de saída. Vincule os nós de esquema de entrada e saída para criar instruções de mapeamento.

Um elemento de esquema é o bloco de construção básico de uma instrução de mapeamento. Quando você define uma instrução de mapeamento, pode usar uma série de elementos de esquema de entrada na instrução ou adicionar uma variável. Você pode alterar a entrada e a saída raiz, adicionar variáveis e personalizar a exibição do esquema, mas não pode editar o esquema.

Você pode usar as seguintes opções para gerenciar os esquemas e pesquisar por elementos:

Personalizar exibição

Altera a forma pela qual o esquema é exibido para que você possa pesquisar rapidamente os elementos relevantes. Você pode pesquisar uma sequência de nós, todos os nós ou uma seleção de nós. Exiba estes nós para entender a lógica do esquema. Esta exibição não afeta o mapeamento.

Pesquisar

Pesquise por elementos no esquema. Você pode pesquisar pelos esquemas de entrada e saída separadamente.

Variáveis

Defina variáveis para armazenar dados. Você pode mapear nós em variáveis e mapear as variáveis para nós. Você também pode mapear variáveis para variáveis. Quando você cria uma variável, a variável é exibida na parte final de ambos os esquemas.

Selecionar Entrada ou Saída Raiz

Altere o elemento raiz no esquema de entrada ou saída. Você pode alterar o elemento raiz para fazer referência a uma parte diferente do esquema.

Escolha o exemplo de origem

Defina o arquivo de origem de exemplo. Use um exemplo de origem para testar a transformação e para testar as expressões do XPath.

Instruções de Mapeamento

Uma instrução de Mapeamento determina como mapear dados do documento hierárquico de entrada para o documento hierárquico de saída. Quando você arrasta um nó do esquema de entrada para o esquema de saída, o editor XMap cria instruções de mapeamento na grade.

Você pode usar a grade para criar instruções de mapeamento simples ou detalhadas. Você pode arrastar elementos dos esquemas de entrada ou saída para campos na grade para incluí-los nas instruções de mapeamento.

Você pode verificar o elemento para o qual uma instrução de mapeamento faz referência. Quando você clicar em uma instrução de mapeamento na grade, o Editor de XMap realça os nós no esquemas.

Adicione expressões XPath para determinar o contexto ou adicionar computações para uma instrução de mapeamento. Quando uma expressão XPath identifica o contexto de uma instrução de mapeamento, a transformação do Processador de Dados executa a instrução do mapeamento para cada ocorrência do elemento de entrada ou expressão no documento de entrada.

Aninhe as instruções de mapeamento para torná-las dependentes de outras instruções de mapeamento. Uma instrução de mapeamento pode ser um pai para um grupo de instruções filhas. Sempre que a transformação do Processador de Dados executar a instrução pai, ela executará as instruções filhas também. As instruções filhas são exibidas recuadas do pai no editor de XMap.

Tipos de Instrução de Mapeamento

Os tipos de instrução de mapeamento definem a lógica de mapeamento XMap. Defina o tipo de instrução de mapeamento com base em se você deseja mapear um valor de entrada simples para um valor de saída, iterar sobre um elemento ou executar o mapeamento com base em uma condição.

Crie uma instrução arrastando um elemento do esquema de entrada para um elemento do esquema de saída ou adicionando uma instrução de mapeamento a uma grade. Quando você cria uma instrução, a

transformação do Data Processor identifica um tipo de instrução de mapeamento com base em se o elemento é um elemento simples, um elemento complexo ou um elemento de repetição.

O tipo de instrução básicas de mapeamento é um Mapa, que mapeia um valor de entrada simples para um valor de saída simples. Outras instruções de mapeamento identificam as condições ou alternativas para a lógica de mapeamento, ou agrupam de um conjunto de instruções lógicas.

Você pode definir os seguintes tipos de instruções de mapeamento na grade:

Mapa

Mapeia um elemento de entrada simples para um elemento de saída simples. Uma instrução de Mapa é o bloco básico de criação do XMap.

Grupo

Um grupo lógico de instruções. Outros tipos de instrução de mapeamento são aninhados na instrução Grupo.

Grupo de Repetição

Uma instrução de grupo que a transformação do Data Processor executa toda vez que o elemento de entrada é exibido no documento de entrada. O Grupo de Repetição contém instruções de Mapa que são iteradas. O Grupo de Repetição identifica o elemento usado para iterar o grupo.

Roteador

Contém um grupo de instruções de Opção e seleciona apenas a instrução de Opção cujo critério de condição corresponde à entrada. Se nenhuma das Opções se aplicar, uma ação Padrão será realizada, se houver uma instrução Padrão. Se nenhuma das Opções se aplicar e não houver uma instrução Padrão, o Roteador falhará.

Opção

Um ou mais instruções de Opção são aninhadas na instrução do Roteador. A instrução de Opção é como a instrução de Grupo e contém um grupo lógico de instruções. A instrução de Opção define uma condição para mapear o elemento de entrada para o elemento de saída.

Padrão

Uma instrução Padrão pode ser aninhada na instrução do Roteador. A instrução Padrão é realizada quando nenhuma instrução de Opção se aplica. Se todas as instruções de Opção falharem e não houver uma instrução Padrão, o Roteador falhará.

Executar XMap

Chama outro objeto XMap na transformação do Data Processor.

RunMapplet

Chama um mapplet da transformação do Processador de Dados.

MappletInput

Uma ou mais instruções MappletInput podem ser aninhadas sob a instrução RunMapplet. Os valores são mapeados para as portas de entrada do mapplet na mesma ordem em que estão listados nas instruções MappletInput.

MappletOutput

Uma ou mais instruções MappletOutput podem ser aninhadas sob a instrução RunMapplet. Os valores nas portas de saída de mapplet são mapeados para a instrução MappletOutput na mesma ordem em que estão listados nas portas de mapplet.

As instruções de mapeamento contêm campos que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e uma condição para mapear um elemento de entrada para um elemento de saída.

Configure se é para ignorar uma instrução de mapeamento quando ela falhar ou não houver entrada.
Configure se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento.

Instruções de Mapa

Uma instrução de Mapa é o bloco de criação básico de um objeto XMap e mapeia um valor de entrada simples para um valor de saída simples. A entrada deve ser um valor único ou um valor constante. Você deve definir a entrada e a saída em uma instrução de Mapa.

Quando você arrasta e solta entre um nó de esquema de entrada simples não repetido e um nó de esquema de saída não repetido, a instrução de Mapa é criada automaticamente.

Uma instrução de Grupo, Grupo de Repetição, Opção ou Padrão pode conter uma ou mais instruções de Mapa filhas.

Propriedades da Instrução de Mapa

A instrução de Mapa contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução de Mapa tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define uma condição para mapear o elemento. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Padrão

Opcional. Valor padrão a ser usado quando um elemento está ausente na entrada. Por exemplo, você pode definir um valor padrão para inicializar um contador.

Entrada

Obrigatório. Uma expressão XPath que define um elemento de entrada. A expressão pode avaliar para um nó ou valor.

Modo

Obrigatório. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Obrigatório. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Obrigatório. Uma expressão XPath que define o valor do elemento na saída hierárquica com base nos resultados da expressão XPath de Entrada.

Ignorar Entrada Ausente

Opcional. Determina se a instrução deverá ser ignorada se não houver correspondência para o valor de Entrada. Escolha uma das seguintes opções:

- Habilitado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desabilitado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução de Mapa.

Instruções de Grupo

Uma instrução de Grupo contém um grupo lógico de instruções. Uma instrução de Grupo pai contém instruções filho. As instruções filho são aninhadas embaixo da instrução de Grupo na grade do editor de XMap.

Você pode usar uma instrução de Grupo para fornecer um contexto ou uma condição comum de êxito ou falha para um grupo de instruções. Você pode usar uma instrução de mapeamento de Grupo se desejar que um conjunto de instruções inteiro seja aprovado ou reprovado. Você pode usar uma instrução de mapeamento de Grupo para agrupar um conjunto de instruções para organizar e simplificar uma grade de XMap.

Quando você vincula um elemento complexo de ocorrência única a um elemento complexo de ocorrência única ou a um elemento de várias ocorrências, o editor de XMap cria uma instrução de Grupo. Um elemento de ocorrência única tem um valor de **Máx. de Ocorrências** de 1.

Exemplo de Instrução de Grupo

Você deseja mapear de um documento hierárquico de entrada com dados do funcionário gerente para um documento hierárquico de saída com dados do trabalhador. Há somente um gerente, então o elemento do funcionário de entrada é uma ocorrência única.

Uma instrução de mapeamento de Grupo é usada quando um elemento é um elemento complexo de ocorrência única. O esquema tem um elemento Funcionário de ocorrência única. O Funcionário tem os elementos filho FirstName e LastName:

```
Employee
  FirstName
  LastName
```

Crie uma instrução de mapeamento de Grupo e configure Funcionário como a entrada. Cada instrução de mapeamento que você incluir no grupo está dentro do contexto de Funcionário.

Na figura a seguir, a instrução 1 é a instrução de Grupo:

	Name	Statement...	Input
1	Employee to Worker	Group	Employee
2	FirstName to FirstName	Map	ns0:FirstName
3	LastName to LastName	Map	ns0:LastName

A coluna de Entrada da instrução de Grupo mostra que a entrada é o elemento pai de FirstName e LastName. A instrução 2 e a instrução 3 são instruções filho da instrução 1. As instruções filho são exibidas recuadas da instrução pai. Para cada entrada do elemento Funcionário, mapeie o elemento FirstName e o elemento LastName para a saída.

Propriedades da Instrução de Grupo

A instrução de Grupo contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução de Grupo tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define a condição de entrada para a instrução de Grupo e todas as instruções filho. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Entrada

Opcional. Uma expressão XPath que avalia como zero ou um elemento ou valor. Se for deixada em branco, a instrução usará o contexto atual. Se a expressão avaliar para mais de um valor, o primeiro será usado.

Modo

Obrigatório. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Correspondender. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Correspondender ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Opcional. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Opcional. Uma expressão XPath que define o valor do elemento na saída hierárquica com base nos resultados da expressão XPath de Entrada. Se for deixada em branco, a instrução usará o contexto atual.

Ignorar Entrada Ausente

Opcional. Determina se a instrução deverá ser ignorada se não houver correspondência para o valor de Entrada. Escolha uma das seguintes opções:

- Habilitado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desabilitado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução de Grupo.

Instruções de Grupo de Repetição

Uma instrução de Grupo de Repetição é uma instrução de grupo que pode ocorrer várias vezes. A entrada é uma expressão XPath que pode avaliar para uma sequência de elementos ou valores.

A transformação do Processador de Dados executa a instrução de Grupo de Repetição para cada elemento ou valor que seja um resultado da expressão XPath de Entrada.

Quando você vincula um elemento do esquema de entrada de repetição a um elemento do esquema de saída de repetição, o editor de XMap cria uma instrução de Grupo de Repetição na grade. Um elemento se repetirá se o valor **Max. Occurs** do elemento for maior do que 1.

Exemplo de Instrução de Grupo de Repetição

Um esquema de entrada tem a seguinte hierarquia:

```
Employees
  Employee (Unbounded)
    LastName
    FirstName
```

Quando você arrasta Funcionário para um elemento de saída no Editor de XMap, a ferramenta Developer cria uma instrução de mapeamento de Grupo de Repetição por padrão. Um grupo de repetição pode conter instruções de mapeamento para retornar o LastName e o FirstName de cada Funcionário no documento hierárquico de entrada.

Propriedades da Instrução de Grupo de Repetição

A instrução de Grupo de Repetição contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução de Grupo de Repetição tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define uma condição para mapear o elemento. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Entrada

Obrigatório. Uma expressão XPath que avalia para uma sequência de nós ou valores.

Modo

Obrigatório. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Opcional. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar Iteração. Se uma instrução aninhada dentro do Grupo de Repetição falhar e a sua propriedade Em Falha estiver definida como **Propagar**, a iteração atual do Grupo de Repetição será ignorada.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Opcional. Uma expressão XPath que define o valor do nó na saída hierárquica com base nos resultados da expressão XPath de Entrada.

Ignorar Entrada Ausente

Opcional. Determina se a instrução deverá ser ignorada se não houver correspondência para o valor de Entrada. Escolha uma das seguintes opções:

- Habilitado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desabilitado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução de Grupo de Repetição.

Instruções de Roteador

Uma instrução de Roteador fornece alternativas para a lógica de mapeamento com base em condições no documento de entrada.

A instrução de Roteador contém uma ou mais instruções de Opção e pode conter uma instrução Padrão. Quando a transformação do Data Processor executa a instrução de Roteador, ela testa cada Opção aninhada na instrução de Roteador.

A primeira instrução de Opção que corresponde é executada. A instrução de Opção pode conter uma ou mais instruções filho de qualquer tipo. Se nenhuma instrução de Opção corresponder, a instrução Padrão é executada. Se não houver uma instrução Padrão, a instrução de Roteador falhará.

Você pode configurar várias instruções de Opção no mesmo grupo do Roteador. A transformação do Data Processor executará a instrução Padrão quando nenhuma das instruções de Opção se aplicarem. A transformação do Data Processor não executará qualquer instrução de Opção abaixo dela no grupo. Quando a transformação do Data Processor não aceitar uma instrução de Opção, ela testará a instrução de Opção seguinte.

Quando a transformação não aceitar nenhuma instrução de Opção e não houver uma instrução Padrão, a instrução de Roteador falhará. Se a instrução de Opção tiver uma condição que é verdadeira, mas as instruções de mapeamento dentro dela falharem e propagarem a falha, o Roteador falhará. Você pode configurar o mapeamento para ignorar o Roteador se o Roteador da falhar.

Se um Roteador não tiver nenhuma instrução de Opção, mas tiver uma instrução Padrão, a instrução Padrão sempre será executada.

Exemplo de Instrução de Roteador

Um XMap contém um grupo de repetição sob o contexto de Funcionário. A primeira instrução filho no grupo é uma instrução de Roteador. A instrução de Roteador tem uma instrução de Opção. A instrução de Opção contém uma condição que verifica se o valor da Função (Function) é igual ao valor de Gerente (Manager). Se a função for igual a gerente, a instrução de Opção será considerada verdadeira. O mapeamento avalia a instrução Executar XMap aninhada sob a instrução Opção. A transformação do Processador de Dados chama XMap EmployeeToWorker para mapear elementos para Gerente.

Se a função não for igual à gerente, a instrução Padrão será verdadeira. O mapeamento avalia a próxima instrução para a opção Padrão. A instrução de mapeamento padrão chama o XMap EmployeeToWorker para mapear elementos para o Funcionário.

A figura a seguir mostra a instrução de Roteador com uma instrução de Opção e uma instrução Padrão:

Employee to Worker	Repeating Group	ns0:Employee		<input type="checkbox"/>	Skip Iterati...
Employee	Router			<input type="checkbox"/>	Skip
Employee to Manager	Option	ns0:Role = "Manager"		<input checked="" type="checkbox"/>	Propagate
EmployeeToWorker	EmployeeToWorker			<input type="checkbox"/>	Propagate ns0:Manager
Employee to Worker	Default			<input type="checkbox"/>	Propagate
EmployeeToWorker	EmployeeToWorker			<input checked="" type="checkbox"/>	Propagate ns0:Worker

Propriedades da Instrução de Roteador

A instrução de Roteador contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução de Roteador tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define uma condição para mapear o elemento. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath

de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Padrão

Obrigatório. Valor padrão a ser usado quando um elemento está ausente na entrada. Por exemplo, você pode definir um valor padrão para inicializar um contador.

Entrada

Obrigatório. Uma expressão XPath que avalia para uma sequência de nós ou valores.

Modo

Obrigatório. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Opcional. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Obrigatório. Uma expressão XPath que define o valor do elemento na saída hierárquica com base nos resultados da expressão XPath de Entrada. O campo de Saída fornece o contexto para as instruções filho.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução de Roteador.

Instruções de Opção

A instrução de Opção oferece a condição para mapear o nó de entrada para o nó de saída. Uma instrução de Opção deve ser aninhada abaixo de uma instrução de Roteador. A instrução de Roteador deve ter uma expressão XPath de Entrada ou uma expressão XPath de Condição. Ou a instrução de Opção pode incluir uma expressão XPath de Entrada e uma expressão XPath de Condição.

A transformação do Processador de Dados aceita uma instrução de Opção quando os resultados da expressão de campo de Entrada e uma expressão de campo de Condição avaliam para um único nó. Se você não definir uma expressão de campo de Entrada, a transformação do Processador de Dados aceitará a instrução de Opção quando a expressão de campo de Condição for avaliada como verdadeira.

A instrução de Opção pode conter uma ou mais instruções filho de qualquer tipo incluindo Mapa, Grupo, Grupo de Repetição, Executar XMap e outras instruções de Roteador. Por exemplo, uma instrução de Opção pode conter a seguinte condição:

```
EmployeeID="100"
```

Quando o EmployeeID for 100, a condição será verdadeira. A instrução filho na grade define a instrução de mapeamento para avaliar quando a condição for verdadeira.

Propriedades da Instrução de Opção

A instrução de Opção contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução de Opção tem as seguintes propriedades:

Condição

Obrigatório se a Entrada não estiver definida. Uma expressão XPath que define uma condição para mapear o elemento. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. A transformação do Data Processor aplica o XPath de Condição ao resultado do XPath de Entrada.

Entrada

Obrigatório se a Condição não estiver definida. Uma expressão XPath que define um elemento de entrada. A expressão pode avaliar para um nó ou valor.

Modo

Opcional. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Opcional. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Opcional. Uma expressão XPath que define o valor do elemento na saída hierárquica com base nos resultados da expressão XPath de Entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução de Opção.

Instruções Padrão

Uma instrução Padrão é uma instrução filho de uma instrução de Roteador. A instrução de Roteador contém uma ou mais instruções de Opção e pode conter uma instrução Padrão. A transformação do Processador de Dados executa a instrução Padrão quando nenhuma das instruções de Opção se aplicam.

Você pode definir somente uma instrução Padrão em um grupo de instruções de Roteador. A instrução Padrão deve ser a última instrução do grupo de instruções de Roteador. A instrução Padrão não pode ter expressões de Entrada ou Condição XPath.

Propriedades da Instrução Padrão

A instrução Padrão contém propriedades que você pode configurar para personalizar a instrução. Você poderá configurar o valor padrão se um elemento de entrada estiver ausente.

A instrução Padrão tem as seguintes propriedades:

Padrão

Obrigatório. Valor padrão a ser usado quando um elemento está ausente na entrada. Por exemplo, você pode definir um valor padrão para inicializar um contador.

Modo

Opcional. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Saída

Opcional. Uma expressão XPath que define o valor do nó na saída hierárquica com base nos resultados da expressão XPath de Entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução Padrão.

Instruções Executar XMap

Uma instrução Executar XMap chama outro XMap.

Quando você cria uma instrução de mapeamento Executar XMap, a ferramenta Desenvolvedor lista os objetos XMap na transformação. Selecione o XMap para chamar. A ferramenta Desenvolvedor cria uma instrução de mapeamento com o nome XMap no campo **Tipo de Instrução**.

Os elementos raiz de entrada e saída no XMap chamado devem ser do mesmo tipo que os valores de entrada e saída transmitidos para ele do XMap de chamada. Você pode chamar um XMap para executar a lógica de mapeamento que é repetida.

Propriedades da Instrução Executar XMap

A instrução Executar XMap contém as propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e a condição para mapear um elemento de entrada para um elemento de saída.

A instrução Executar XMap tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define uma condição para mapear o elemento. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Entrada

Opcional. Uma expressão XPath que avalia para uma sequência de nós ou valores. O tipo de instrução de mapeamento determina como a transformação do Processador de Dados usa os nós ou os valores no mapeamento.

Modo

Obrigatório. Determina se a transformação do Data Processor adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera achar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento correspondente existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Obrigatório. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.

- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Opcional. Uma expressão XPath que define o valor do nó na saída hierárquica com base nos resultados da expressão XPath de Entrada.

Ignorar Entrada Ausente

Obrigatório. Determina se a instrução deverá ser ignorada se não houver correspondência para o valor de Entrada. Escolha uma das seguintes opções:

- Habilitado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desabilitado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução Executar XMap.

Instrução RunMapplet

Uma instrução RunMapplet chama um mapplet.

Quando você cria uma instrução de mapeamento RunMapplet, a Developer tool lista os objetos de referência do mapplet associados à transformação. Selecione o mapplet a ser chamado. A Developer tool cria uma instrução XMap com o nome do mapplet no campo **Tipo de Instrução**.

As portas de entrada e saída no mapplet chamado devem ser do mesmo tipo que os valores transmitidos para ele do XMap de chamada. Você pode chamar um mapplet para realizar tarefas como mascaramento de dados, qualidade de dados, pesquisa de dados e outras atividades normalmente relacionadas à transformação relacional, sem a necessidade de converter dados no formato relacional e voltar a converter no formato hierárquico.

Nota: A ação RunMapplet pode ser usada para chamar somente mapplets passivos.

Propriedades da Instrução RunMapplet

A instrução RunMapplet contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada, a saída e uma condição para executar a instrução RunMapplet. A instrução RunMapplet pode conter a instrução MappletInput e a instrução MappletOutput.

A instrução RunMapplet tem as seguintes propriedades:

Condição

Opcional. Uma expressão XPath que define uma condição para executar a instrução RunMapplet. Uma condição é semelhante a uma expressão de predicado na coluna de Entrada. Se você definir uma expressão XPath de Entrada e uma expressão XPath de Condição para a mesma instrução de mapeamento, a transformação do Processador de Dados aplicará o XPath de Condição ao resultado do XPath de Entrada.

Entrada

Opcional. Uma expressão XPath que é avaliada como uma sequência de nós ou valores. O tipo de instrução de mapeamento determina como a transformação do Processador de Dados usa os nós ou os valores no mapeamento.

Modo

Obrigatório. Determina se a transformação do Processador de Dados adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento.

Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera localizar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.
- Corresponder ou Adicionar. Se um elemento de correspondência existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Em Falha

Opcional. Determina a ação realizada se a instrução falhar. Escolha uma das seguintes opções:

- Ignorar. Se a instrução falhar, ignore a instrução.
- Propagar. Se a instrução falhar, force a instrução pai a falhar também.

Saída

Obrigatório. Uma expressão XPath que define o valor do nó na hierarquia de saída com base nos resultados da expressão XPath de Entrada.

Ignorar Entrada Ausente

Opcional. Determina se a instrução deverá ser ignorada se não houver correspondência para o valor de Entrada. Escolha uma das seguintes opções:

- Ativado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desativado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Tipo de Instrução

Obrigatório. Identifica a instrução como uma instrução RunMapplet. O campo identifica o nome do mapplet referenciado.

Instrução MappletInput

A instrução MappletInput contém propriedades que você pode configurar para personalizar a instrução. Você pode configurar a entrada para mapear um elemento de entrada para a porta de entrada do mapplet. Uma ou mais instruções MappletInput podem ser aninhadas sob a instrução RunMapplet.

A instrução MappletInput é executada para fornecer um valor a ser transmitido para a entrada de mapplet. Os valores na instrução RunMapplet são transmitidos para as portas de mapplet na mesma ordem em que estão listados na instrução RunMapplet. Se uma instrução for ignorada, um valor nulo será transmitido para a porta de entrada do mapplet.

Uma instrução MappletInput transmite um valor único. Quando o RunMapplet é executado, os valores das instruções MappletInput aninhadas são coletados e transmitidos para o mapplet na mesma ordem das instruções MappletInput.

Propriedades da Instrução MappletInput

A instrução MappletInput contém propriedades que você pode configurar para personalizar a instrução.

A instrução RunMapplet tem as seguintes propriedades:

Padrão

Opcional. O valor padrão a ser usado quando um elemento está ausente na entrada da porta de mapplet.

Entrada

Obrigatório. Uma expressão XPath que é avaliada como uma sequência de nós ou valores. Os valores são transmitidos para as portas de entrada do mapplet.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Ignorar Entrada Ausente

Opcional. Determina se a instrução deverá ser ignorada se não houver uma correspondência da porta de entrada do mapplet para o valor de entrada. Escolha uma das seguintes opções:

- Ativado. Se o elemento não estiver no documento hierárquico de entrada, a transformação do Processador de Dados ignorará a instrução sem erro.
- Desativado. A instrução falha quando o elemento não está no documento hierárquico de entrada.

Instrução MappletOutput

A instrução MappletOutput é executada para obter um valor transmitido da saída de mapplet. Os valores são transmitidos das portas de mapplet para as portas de saída RunMapplet na mesma ordem em que estão listados na instrução RunMapplet. Uma ou mais instruções MappletOutput podem ser aninhadas sob a instrução RunMapplet.

Uma instrução MappletOutput obtém um único valor. Quando o RunMapplet é executado, os valores do mapplet são coletados para as instruções MappletOutput aninhadas na mesma ordem das instruções MappletOutput.

Propriedades da Instrução MappletOutput

A instrução MappletOutput contém propriedades que você pode configurar para personalizar a instrução.

A instrução RunMapplet tem as seguintes propriedades:

Padrão

Opcional. O valor padrão a ser usado ao testar a transformação do Processador de Dados sem a entrada real de mapeamento. O valor padrão não será usado quando houver entrada de mapeamento.

Modo

Opcional. Determina se a transformação adiciona um elemento de saída ou faz a correspondência de um elemento existente com um valor de uma instrução de mapeamento. Escolha uma das seguintes opções:

- Adicionar. Cria um elemento no documento hierárquico de saída. Se o elemento não ocorrer várias vezes e o mesmo valor existir na saída, a instrução de mapeamento falhará.
- Corresponder. A instrução espera localizar uma correspondência para o elemento nos elementos de saída. A instrução falhará se o elemento não existir no documento hierárquico de saída.

- Corresponder ou Adicionar. Se um elemento de correspondência existir no documento hierárquico de saída, a transformação do Processador de Dados não adicionará um elemento de saída. Se o elemento não existir no documento hierárquico de saída, a transformação criará um elemento de saída.

Nome

Opcional. Um nome para a instrução. Você pode alterar o nome a qualquer momento. O nome identifica as instruções para que seja possível encontrá-las na grade de mapeamento ou em um log de eventos. Os nomes de instruções não precisam ser exclusivos.

Saída

Obrigatório. Uma expressão XPath que é avaliada como uma sequência de nós ou valores. Os valores de porta de saída de mapplet são transmitidos para a sequência de nós.

Criando um XMap

Para criar um objeto XMap do Processador de Dados, escolha os esquemas de entrada e saída e adicione instruções de mapeamento.

1. A exibição de **Objetos** da transformação do Processador de Dados cria um XMap. Selecione um esquema de entrada, um exemplo de origem e um esquema de saída.
2. Para abrir o editor de XMap, clique no objeto XMap.
3. Para criar uma instrução de mapeamento de Mapa, de Grupo ou de Grupo de Repetição, no editor de XMap, arraste e solte de um nó no esquema hierárquico de entrada para um nó no esquema hierárquico de saída.
O Editor de XMap cria um vínculo de mapa entre os nós. A instrução de mapeamento é exibida na grade. O Editor de XMap preenche automaticamente os campos de instrução de mapeamento.
4. Para criar a lógica condicional na grade, adicione uma instrução de mapeamento de Roteador da seguinte maneira:
 - a. Na instrução de mapeamento de Roteador, crie as instruções de mapeamento de Opção. Arraste e solte os nós de esquema de entrada e saída para os campos da instrução Opção na grade.
 - b. Na instrução de mapeamento de Roteador, crie a instrução de mapeamento de Padrão para especificar o que acontece se nenhuma instrução de mapeamento Opção se aplicar.
 - c. Com as instruções de mapeamento de Opção, crie instruções de mapeamento de Mapa para especificar condições para mapear o nó de entrada para o nó de saída.
5. Para fornecer um contexto comum para um grupo de instruções, adicione uma instrução de mapeamento de Grupo. Aninhe as instruções de mapeamento de Mapa abaixo da instrução de mapeamento de Grupo.
6. Para chamar outro objeto XMap, adicione uma instrução Executar XMap.
7. Para alterar o contexto e a lógica de mapeamento de uma instrução de mapeamento, edite as propriedades da instrução de mapeamento da seguinte maneira:
 - a. Rebaixe instruções para instruções filho ou promova instruções para instruções pai.
 - b. Crie expressões XPath para alterar o contexto ou adicionar predicados usando o editor de XPath.

Usando a Grade do Editor XMap

Quando você arrasta um nó do esquema de entrada para o esquema de saída, a ferramenta Desenvolvedor cria um Mapa, Grupo ou instrução de mapeamento de Grupo de Repetição na grade. É possível atualizar a instrução de mapeamento. Os campos de Entrada e de Saída contêm os elementos do esquema. Se você

deseja criar instruções de mapeamento para fornecer contexto ou para definir as opções de Roteador, você pode digitar as instruções na grade.

Quando você seleciona uma instrução de mapeamento na grade, o editor XMap realça os nós dos esquemas de entrada e de saída que estão na grade.

É possível copiar uma instrução de uma linha para outra linha na grade. Se a linha não for válida para a localização na qual você copiá-la, o editor de XMap exibirá uma caixa de diálogo com um erro de validação. Você pode alterar as instruções do XPath de entrada e de saída na caixa de diálogo para ajustar o contexto da instrução de mapeamento ou pode alterar os campos XPath de entrada e de saída na grade.

Crindo Instruções de Mapeamento

Crie instruções de mapeamento no editor de XMap. Você pode criar instruções de mapeamento arrastando nós do esquema de entrada para o esquema de saída, e você pode definir as instruções na grade de instrução do mapeamento.

Siga as seguintes etapas para definir instruções de mapeamento na grade:

1. Para criar uma instrução de mapeamento, nas opções de grade, clique em **Novo**.
2. Selecione o tipo de instrução de mapeamento na lista.
Se você escolher **Executar XMap**, a ferramenta Developer mostrará uma lista dos objetos XMap na transformação.
3. Digite um nome para a instrução de mapeamento.
4. Para definir a entrada da instrução de mapeamento, arraste um elemento do esquema de entrada para o campo de Entrada. Você também pode configurar uma expressão XPath ou uma constante no campo de Entrada.
5. Para selecionar o nó de saída da instrução de mapeamento, arraste um elemento do esquema de saída para o campo de Saída. Você também pode configurar uma expressão XPath no campo de Saída.
6. Para criar uma expressão XPath com o editor de Expressão XPath para um campo de Entrada, de Saída ou de Condição, clique no botão **Abrir** no campo.
7. Para alterar o tipo de instrução de mapeamento, clique no botão **Abrir** no campo **Tipo de Instrução**. Escolha o tipo de instrução de mapeamento na lista.

Interface de Grade de Instruções de Mapeamento

Edite instruções de mapeamento na grade da instrução de mapeamento do editor de XMap. Você deve criar uma instrução para poder modificar os campos na grade da instrução de mapeamento.

Você pode realizar as tarefas a seguir na grade de instrução de mapeamento:

- Arraste nós do esquema de entrada ou saída para campos nas instruções de mapeamento.
- Copie os elementos para os campos da instrução de mapeamento ou digite os valores nos campos.
- Você pode mover uma instrução de mapeamento para cima ou para baixo da grade. Selecione a linha e clique nas opções de seta **Para cima** ou **Para baixo**.
- Você pode clicar em **Rebaixar** para recuar uma instrução de mapeamento sob outra instrução. Clique em **Promover** se desejar mover uma instrução para cima na hierarquia.
- Clique em **Ir para Número de Linha** para navegar para uma linha. Insira a linha para navegar até ela. A ferramenta Developer destaca a linha para você.

Vinculando Nós de Esquema

Você pode vincular um nó do esquema de entrada a um nó do esquema de saída arrastando com o mouse. Você pode arrastar e soltar para mapear um nó simples para um nó simples, um nó simples para um nó complexo, um nó complexo para um nó simples ou um nó complexo para um nó complexo.

O editor de XMap cria um vínculo de mapa entre o nó do esquema de entrada e o nó do esquema de saída. O Editor de XMap também cria uma instrução de mapeamento na grade.

Você pode mover uma instrução de mapeamento arrastando e soltando de uma linha na grade para outra linha para alterar a lógica de XMap. Por exemplo, você pode usar esse método para alterar a sequência das instruções de Opção dentro de um Roteador.

Recortar e Colar Instruções de Mapeamento

Você pode recortar e colar instruções de mapeamento na grade da instrução de mapeamento do editor de XMap.

Você pode mover as instruções para cima e para baixo. Você pode colar instruções de mapeamento mesmo dentro de instruções aninhadas. Você pode promover uma instrução para ser uma instrução pai ou rebaixar uma instrução para ser um filho.

Normalmente, as instruções coladas devem ser corrigidas, pois sua lógica geralmente ficará fora do contexto. Depois de colar uma instrução, você poderá modificar os campos na grade da instrução de mapeamento.

Expressões XPath

As expressões XPath identificam elementos específicos ou nós em documentos hierárquicos ou verificam se há condições nos dados. Use expressões XPath para definir os campos de Entrada, de Condição ou de Saída de uma instrução de mapeamento.

O XPath é uma sintaxe que define partes de um documento hierárquico. Use o XPath para selecionar as sequências de nós ou os valores em um documento hierárquico. O XPath inclui uma biblioteca de funções padrão que você pode usar para selecionar dados.

Você pode definir expressões XPath 2.0 na transformação do Data Processor. Quando você configurar expressões XPath de Saída, poderá usar um subconjunto da sintaxe XPath 2.0 ao definir instruções de mapeamento para o modo Adicionar ou Corresponder ou Adicionar.

Para obter mais informações sobre o XPath, consulte a documentação do XPath.

A tabela a seguir descreve algumas expressões XPath:

Expressão XPath	Descrição
nodename	Seleciona todos os nós filhos do nome determinado no contexto.
. (ponto)	Seleciona o nó atual.
..	Seleciona o pai do nó atual.
@	Seleciona o atributo.

Expressão XPath	Descrição
/	Seleciona do nó raiz ou filho do nó atual se precedido pelo nó. Quando o caminho começa com uma barra (/), ele representa um caminho absoluto para um elemento.
//	Seleciona nós em qualquer lugar no documento ou descendentes do nó atual se precedido pelo nó.

A tabela a seguir lista algumas expressões XPath e o resultado de cada expressão:

Expressão XPath	Resultado
/livraria	Seleciona o nó raiz da livraria.
livraria/livro	Seleciona os nós de livro que são filhos de todos os nós de livraria.
//livro	Seleciona os nós de livro no documento em todas as localizações.
livraria//livro	Seleciona todos os nós de livro que são descendentes dos nós de livraria.
/livraria/*	Seleciona todos os nós filhos do elemento raiz da livraria.
//*	Retorna uma sequência de todos os elementos no documento.

Predicados

Um predicado é uma expressão que você configura para localizar um nó em um documento hierárquico. É possível configurar a expressão para localizar um valor específico. Crie um predicado em um campo de Entrada, Condição ou Saída de uma instrução de mapeamento.

Ao definir um predicado, coloque a expressão entre colchetes [] após o nó.

```
/<nó>[expressão]
```

Por exemplo, a seguinte expressão seleciona os elementos book que são filhos de bookstore e possuem um elemento price com um valor maior que 55.00:

```
/bookstore/book[price>55.00]
```

A seguinte expressão seleciona os elementos title dos elementos book que são filhos de bookstore com um valor de elemento price maior que 55.00:

```
/bookstore/book[price>55.00]/title
```

A seguinte expressão seleciona os elementos title que possuem um atributo lang com um valor de "eng":

```
//title[@lang="eng"]
```

Nota: A transformação de Processador de Dados não pode aceitar todas as instruções XPath no campo de Saída quando você configura uma instrução de mapeamento com o modo Adicionar ou com o modo Corresponder ou Adicionar.

Referência de Predicados do XPath

Os predicados do XPath localizam um nó ou uma sequência de nós correspondentes em um documento hierárquico. Uma expressão de predicado define o valor de um campo de Entrada, de Condição ou de Saída

de uma instrução de mapeamento. A expressão do XPath determina o contexto no qual um mapeamento é executado.

Use as seguintes expressões do XPath em predicados para selecionar um nó ou uma sequência de nós:

ancestral

Seleciona todos os ancestrais, como pai ou avô, do nó atual. Por exemplo, a expressão de predicado "ancestor::book" seleciona todas os ancestrais de livro do nó atual.

ancestor-or-self

Seleciona todos os ancestrais, como pai ou avô, do nó atual, bem como o nó atual. Por exemplo, a expressão de predicado "ancestor-or-self::book" seleciona todos os ancestrais de livro do nó atual, e também o nó atual.

atributo

Seleciona todos os atributos do nó atual. Por exemplo, a expressão de predicado "attribute::lang" seleciona o atributo lang do nó atual.

filho

Seleciona todos os filhos do nó atual. Por exemplo, a expressão de predicado "child::book" seleciona todos os nós de livro que são filhos do nó atual.

descendente

Seleciona todos os descendentes, como filhos ou netos, do nó atual. Por exemplo, a expressão de predicado "descendant::book" seleciona todos os descendentes de livro do nó atual.

descendant-or-self

Seleciona todos os descendentes, como filhos ou netos, do nó atual, bem como o nó atual. Por exemplo, a expressão de predicado "descendant-or-self::book" seleciona todos os descendentes de livro do nó atual e o nó atual se ele for um nó de livro.

seguinte

Seleciona tudo no documento após a marca de fechamento do nó atual. Por exemplo, a expressão de predicado "following::book" selecionar tudo no documento após a marca de fechamento do nó de livro.

following-sibling

Seleciona todos os irmãos após o nó atual. Por exemplo, a expressão de predicado "following-sibling::book" seleciona todos os irmãos no documento após o nó de livro.

espaço de nome

Seleciona todos os nós de espaço de nome do nó atual.

pai

Seleciona o pai do nó atual. Por exemplo, a expressão de predicado "parent::book" seleciona o atributo lang do nó atual.

anterior

Seleciona todos os nós que aparecem antes do nó atual no documento, exceto ancestrais, nós de atributo e nós de espaço de nome. Por exemplo, a expressão de predicado "preceding::book" seleciona os nós antes do nó de livro.

preceding-sibling

Seleciona todos os irmãos que são exibidos antes do nó atual no documento. Por exemplo, a expressão de predicado "preceding-sibling::book" seleciona os nós de irmãos antes do nó de livro.

si mesmo

Seleciona o nó atual. Por exemplo, a expressão de predicado "self::book" seleciona o nó de livro atual.

Operadores Aritméticos do XPath

Para executar cálculos, adicione operadores aritméticos que avaliem nós do documento hierárquico. Você pode adicionar operadores aritméticos para expressões XPath nos campos de Entrada, de Condição ou de Saída de uma instrução de mapeamento.

A tabela a seguir descreve os operadores aritméticos do XPath que você pode usar em expressões do XMap:

Expressão XPath	Descrição
	Seleciona dois conjuntos de nós no contexto. Por exemplo, a expressão de predicado <code>//book //cd</code> retorna um conjunto de nós com todos os elementos de livro e cd.
+	Adiciona os elementos. Por exemplo, a expressão de predicado <code>"1+2"</code> retorna 3.
-	Subtrai os elementos. Por exemplo, a expressão de predicado <code>"2-1"</code> retorna 1.
*	Multiplica os elementos. Por exemplo, a expressão de predicado <code>"2*1"</code> retorna 2.
div	Divide os elementos. Por exemplo, a expressão de predicado <code>"6 div 3"</code> retorna 2.
=	Seleciona os elementos que são iguais à expressão. Por exemplo, a expressão de predicado <code>"cost=1.50"</code> retorna true quando o custo é 1,50 e retorna false quando o custo é 1,60.
!=	Seleciona os elementos que não são iguais à expressão. Por exemplo, a expressão de predicado <code>"cost!=1.50"</code> retorna true quando o custo é 1,60 e false quando o custo é 1,50.
<	Seleciona os elementos que são menores do que a expressão. Por exemplo, a expressão de predicado <code>"tax<1.50"</code> retorna true quando o imposto é 1,00 e false quando o imposto é 1,50.
<=	Seleciona os elementos que são iguais ou menores que a expressão. Por exemplo, a expressão de predicado <code>"tax<=1.50"</code> retorna true quando o imposto é 1,50 e false quando o imposto é 1,80.
>	Seleciona os elementos que são maiores que a expressão. Por exemplo, a expressão de predicado <code>"tax>1.50"</code> retorna true quando o imposto é 1,90 e false quando o imposto é 1,50.
>=	Seleciona os elementos que são iguais ou maiores que a expressão. Por exemplo, a expressão de predicado <code>"tax>=1.50"</code> retorna true quando o imposto é 1,50 e false quando o imposto é 1,00.
ou	Seleciona os elementos que satisfazem uma ou mais condições. Por exemplo, a expressão de predicado <code>"tax=1.50 or tax=1.70"</code> retorna true quando o imposto é 1,50 e false quando o imposto é 1,00.
e	Seleciona os elementos que satisfazem todas as condições fornecidas. Por exemplo, a expressão de predicado <code>"price>1.00 and price<1.90"</code> retorna true quando o preço é 1,50 e false quando o preço é 1,00.
modo	Executa a divisão e fornece o restante. Por exemplo, a expressão de predicado <code>"3 mod 2"</code> retorna 1.

Expressões XPath de Saída

A transformação de Processador de Dados aceita um subconjunto de instruções XPath no campo Saída quando o campo Modo está definido como **Adicionar** ou **Corresponder ou Adicionar**. Quando você seleciona

essas configurações de modo, a transformação de Processador de Dados cria elementos conforme necessário para corresponder à expressão XPath no campo Saída.

É possível usar uma expressão XPath simples no campo Saída. Uma expressão simples tem eixos filho, eixos pai ou variáveis. Expressões simples não têm predicados, funções ou eixos complexos. Por exemplo, você pode usar as seguintes expressões para o campo Saída:

```
person/data  
/root/ceo/name  
$var/name  
person/../ceo
```

É possível usar um predicado simples com cardinalidade para um elemento com várias instâncias. Por exemplo, você pode usar a seguinte expressão do campo Saída:

```
person/phone[4]
```

É possível usar um predicado simples com uma fórmula contendo um sinal de igual, com XPaths simples no lado esquerdo desse sinal de igual. Por exemplo, você pode usar as seguintes expressões para o campo Saída:

```
Person[id=10]  
Person[id=$id]  
Person[id=@dp:input()/ID]  
Company[name=upper-case($compName)]  
Person[role="manager" and id=1]
```

Você também pode usar uma combinação de uma expressão simples com cardinalidade e uma fórmula que usa um XPath simples no lado esquerdo do sinal de igual. Por exemplo, você pode usar as seguintes expressões para o campo Saída:

```
company[4]/details[id=$myid]/phone
```

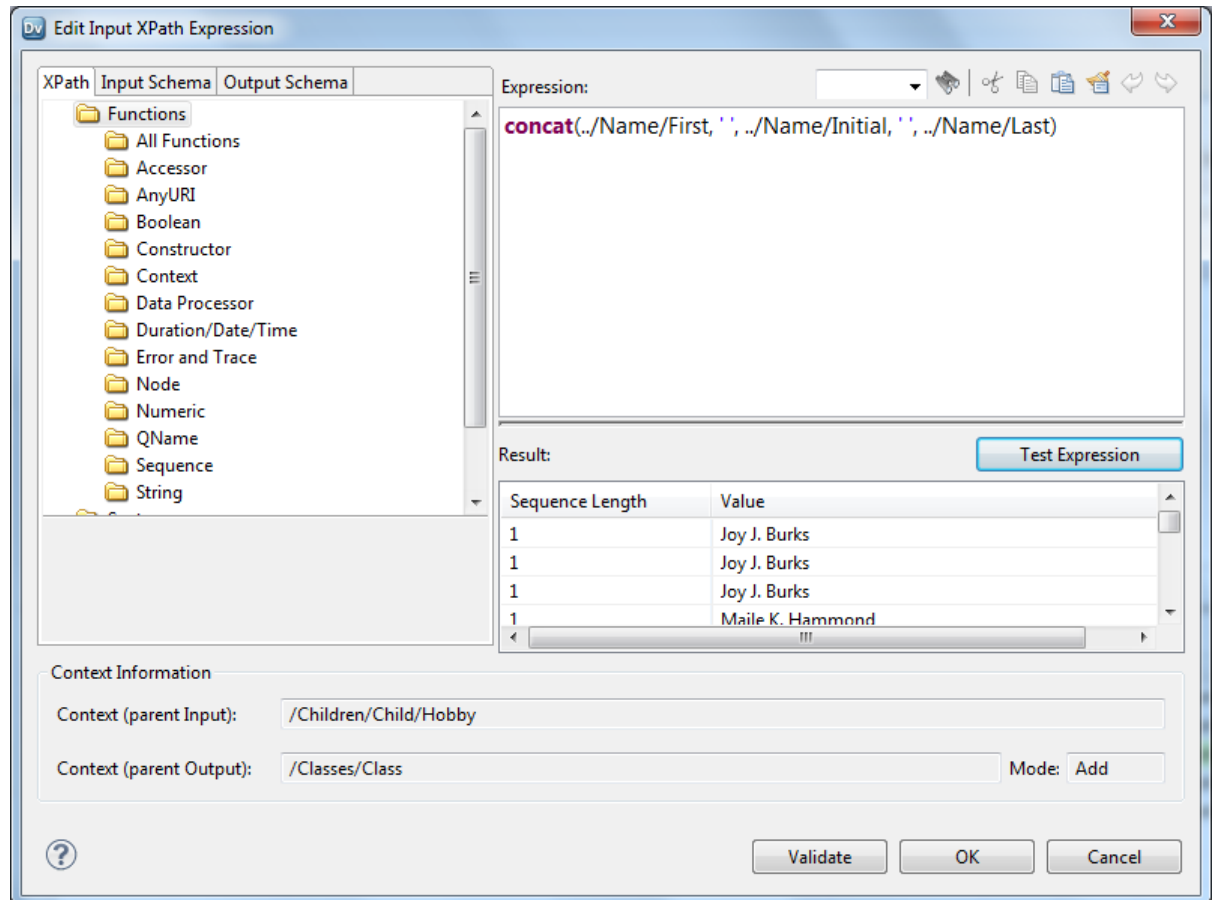
Nota: Quando o campo Modo está definido como **Corresponder**, o campo Saída também pode aceitar expressões XPath complexas.

Editor de Expressão do XPath

Crie expressões no Editor de Expressão do XPath. O XPath é uma linguagem de consulta usada para selecionar nós em um documento hierárquico e executar computações.

Você pode usar as expressões do XPath para definir o contexto de uma instrução de mapeamento. Você pode definir as condições para transformar e filtrar os dados usando diferentes expressões do XPath em propriedades de instrução de mapeamento. Você pode adicionar vários cálculos aritméticos a uma instrução de mapeamento usando expressões do XPath.

Quando você clica no botão Abrir no campo Entrada, Condição ou Saída, o Editor de Expressão é exibido. A figura a seguir mostra o Editor de Expressão XPath:



Crie expressões no painel **Expressão**.

O Editor de Expressão do XPath tem um painel com uma biblioteca de funções que você pode usar para criar expressões do XPath. As funções são padrão para o Caminho de Linguagem XML do W3C. A biblioteca de funções também inclui algumas funções que são específicas à transformação do Processador de Dados.

Funções do Processador de Dados

O Editor de Expressão tem funções de Processador de Dados que podem ser usadas para a transformação de Processador de Dados.

A transformação de Processador de Dados usa as seguintes funções XPath:

dp:as_xml

Recebe um nó como entrada e retorna o valor do nó e o valor de todos os filhos como uma string XML recursivamente. A função as_xml usa a seguinte sintaxe: `dp:as_xml(<nó>)`

dp:get_id

Gera e retorna um ID exclusivo associado a um nó. Esse ID pode ser usado para criar relacionamentos de chave primária/chave externa nos dados. Mapeie o ID para um nó no esquema e mapeie-o para chaves em dados relacionais. A função get_id usa a seguinte sintaxe: `dp:get_id(<nó>)`

dp:input

Retorna o nó que fornece o contexto de entrada atual. Use a função no campo Saída para fazer referência a um nó a partir do esquema de Entrada. A função input usa a seguinte sintaxe: `dp:input()`

dp:transform

Chame um transformador de transformação do Processador de Dados definido em um Script. Você pode executar uma transformação embutida ou externa. A função usa a seguinte sintaxe:

`dp:transform(<transform-name>, <transform-value>)`

A função transform usa os seguintes parâmetros:

- Transform-name. O nome do transformador no Script.
- Transform-value. O valor de transformação que executará a transformação.

Nota: A função lookup está disponível por meio do uso da função **dp:transform**.

dp:output

Retorna o nó que fornece o contexto de saída atual. Use a função no campo Entrada para fazer referência a um nó no esquema de saída. A função output usa a seguinte sintaxe: `dp:output()`

Exemplo de Expressões do XPath

Use as expressões do XPath nas instruções de mapeamento. As expressões do XPath identificam os elementos no documento de entrada para serem mapeados e transformados no documento de saída. Uma expressão do XPath também é usada para executar uma operação aritmética.

A seguinte imagem mostra as expressões do XPath na grade:

The screenshot shows the XMap tool interface with two schemas and a mapping table.

Schema 1: ChildrenToHobbies

Name	Type	Min...	Max. Occurs	Description
Children	(Children)	1	1	
Child	(Child)	1	Unbounded	
Name	(Name)	1	1	
First	xsd:string	1	1	
Last	xsd:string	1	1	
Initial	xsd:string	0	1	
Hobby	xsd:string	1	Unbounded	
Variables	(Variables)	1	1	

Schema 2: Classes

Name	Type	Min...	Max. Occurs	Description
Classes	(Classes)	1	1	
noOfClasses	xsd:integer	0	1	
Class	(Class)	1	Unbounded	
name	xsd:string	0	1	
noOfChildren	xsd:int	0	1	
Child	xsd:string	1	Unbounded	
Variables	(Variables)	1	1	

Mapping Table

Name	Statement...	Input	Condition	Skip...	Default	On Fail	Output	Mode
1	Children to Classes	Repeating...	Child			Propagate		Add
2	Hobby to Class	Repeating...	Hobby			Propagate	Class[@name = dp:...	Match or Add
3	First to Child	Map	concat(../Name/First, ' ', ../Name/Initial, ' ', ../Name/Last)			Propagate	Child	Add
4	Count Children in Class	Map	dp:output()/@noOfChildren + 1		1	Propagate	@noOfChildren	Match or Add
5	Count # of Classes	Map	count(dp:output()/Class)			Propagate	@noOfClasses	Match or Add

O documento de entrada do XMap é uma lista de filhos e seus hobbies. A raiz de entrada é Filhos. Filho é um elemento de múltiplas ocorrências dentro de Filhos. Cada filho tem um Nome e hobbies de múltiplas ocorrências. O Nome consiste nos elementos Primeiro, Inicial e Último.

A saída é uma lista de classes com o número de filhos em cada classe. A raiz de saída é Classes. As classes têm um atributo que contém o número total de classes. Cada elemento de Hobby de entrada aponta para um elemento de Classe de saída. Uma instrução Map concatena os elementos Primeiro, Inicial e Último no elemento de saída de Filho. Outra instrução Map conta o número de filhos em cada classe. Outra instrução conta o número de classes.

O XMap contém as seguintes expressões:

Grid row 2 expression <Class[@name = dp:input()]>

Adiciona um elemento de Classe ou localiza uma Classe que corresponde ao Hobby. A dp:input() é necessária porque a expressão se refere a um elemento de entrada.

Grid row 3 expression <concat(..Name/First,' ',..Name/Initial,' ',..Name/Last)>

Concatena o nome, a inicial do meio e o sobrenome e adiciona espaços entre eles.

Grid row 4 expression <dp:output()/@noOfChildren + 1>

Para cada Hobby que ocorre, adicione 1 ao número de filhos desta classe. A função dp:output() é necessária porque a expressão se refere a um elemento de saída.

Grid row 5 expression <count(dp:output()/Class)>

Conta os elementos de Classe. A função dp:output() é necessária porque a expressão se refere a um elemento de saída.

Criando uma Expressão

Crie expressões XPath no **Editor de Expressão**.

1. Na instrução XMap, clique no botão **Abrir** do campo **Entrada, Condição** ou **Saída**.
O **Editor de Expressão** será exibido.
2. Para adicionar um elemento a uma expressão, clique duas vezes nos elementos no painel de .
3. Clique em **Validar** para validar a expressão.
4. Se a expressão for para o campo de entrada, clique em **Testar Expressão** para testar a expressão com relação ao exemplo de dados.

Os resultados aparecem sempre que a ferramenta Developer avalia a expressão usando o exemplo de dados. A exportação XPath pode retornar uma sequência de zeros, ou mais nós ou valores. O

Comprimento da Sequência indica quantos nós a exportação XPath retorna.

Variáveis XMap

Você pode adicionar variáveis no editor XMap. Você pode mapear valores a variáveis e usar essas variáveis em predicados ou como recipientes temporários para valores. Você pode mapear variáveis para elementos de saída.

Quando você cria uma variável, ela aparece nos esquemas de entrada e saída da exibição do XMap. A ferramenta Developer adiciona um sinal de cifrão (\$) ao nome da variável para indicar que se trata de uma variável.

É possível criar uma variável que consiste em uma lista de vários valores. Uma variável de lista pode ser usada para a mesma finalidade de um elemento de esquema de ocorrência múltipla. Configure uma variável de lista como entrada para um grupo de repetição ou configure um predicado para procurar um valor nessa variável de lista.

Por exemplo, existe um documento XML que contém endereços. Você precisa criar uma lista de todos os países a partir dos endereços. Mapeie o elemento country para uma \$countries que você define como uma lista.

Criando uma Variável no Editor de XMap

Você pode criar variáveis no Editor de XMap.

1. Clique em **Variáveis** acima do esquema de entrada ou saída no editor de XMap.
A caixa de diálogo **Variáveis** será exibida.
2. Para criar uma variável, clique em **Novo**.
3. Insira um nome de variável e um tipo de dados.
4. Ative a opção de **Lista** para criar uma variável de ocorrência múltipla.

Exemplo de XMap

Um documento XML contém dados de funcionários que incluem a função do funcionário na empresa. Você precisa criar um documento XML que tenha os gerentes e funcionários em grupos separados. Crie dois objetos XMap na transformação do Processador de Dados para reestruturar o documento XML.

O componente de inicialização é um objeto XMap que contém uma instrução de Roteador. Uma instrução de Opção verifica se a função do funcionário função é "Gerente". Se a função for gerente, o XMap mapeará os elementos do funcionário para um grupo de saída de gerente. Caso contrário, o XMap mapeará os elementos do funcionários para um grupo de trabalhadores na saída XML.

O componente de inicialização XMap chama outro XMap para mapear os elementos do Funcionário para elementos de saída.

Exemplo de Esquema de Entrada XML

O esquema de Entrada XML para o exemplo de XMap tem a seguinte estrutura:

```
<?xml version="1.0" encoding="UTF-16LE"?>
<!-- edited with XMLSpy v2012 sp1 (x64) (http://www.altova.com) by Informatica
Corporation (Informatica Corporation) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="company"
targetNamespace="company" elementFormDefault="qualified"
attributeFormDefault="unqualified">

  <xs:element name="Employee" type="EmployeeType"/>

  <xs:element name="Input">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Company" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="Company">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Name" type="xs:string"/>
        <xs:element ref="Department" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="Department">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Name" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```



```

        <xs:element ref="Employee" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
</xs:element>

<xs:complexType name="EmployeeType">
    <xs:sequence>
        <xs:element name="FirstName" type="xs:string" minOccurs="0"/>
        <xs:element name="LastName" type="xs:string" minOccurs="0"/>
        <xs:element name="Role" type="xs:string" minOccurs="0"/>
        <xs:element name="StartDate" type="xs:date" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"/>
</xs:complexType>

</xs:schema>

```

A raiz do esquema é a Entrada. A Entrada tem Empresas de múltiplas ocorrências. Em cada Empresa, há diversos Departamentos. Cada Departamento tem Funcionários. O Funcionário tem uma Função que determina se o funcionário é um gerente.

Exemplo de Esquema de Saída XML

O esquema de Saída XML para o exemplo de XMap tem a seguinte estrutura:

```

<?xml version="1.0" encoding="UTF-16LE"?>
<!-- edited with XMLSpy v2012 sp1 (x64) (http://www.altova.com) by Informatica
Corporation (Informatica Corporation) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="organization"
targetNamespace="organization" elementFormDefault="qualified"
attributeFormDefault="unqualified">

    <xs:element name="Worker" type="WorkerType"/>

    <xs:element name="Output">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Organization" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="Organization">
        <xs:complexType>
            <xs:sequence>
                <xs:choice maxOccurs="unbounded">
                    <xs:element name="Worker" type="WorkerType"/>
                    <xs:element name="Manager" type="WorkerType"/>
                </xs:choice>
                <xs:element name="Department" type="xs:string" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="noOfEmployees" type="xs:int"/>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="WorkerType">
        <xs:sequence>
            <xs:element name="FirstName" type="xs:string" minOccurs="0"/>
            <xs:element name="LastName" type="xs:string" minOccurs="0"/>
            <xs:element name="FullName" type="xs:string" minOccurs="0"/>
            <xs:element name="Id" type="xs:string"/>
            <xs:element name="YearsOfService" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>

```

O esquema raiz é a Saída. A Saída tem Organizações de ocorrência múltipla. Dentro de cada Organização há Funcionários e Gerentes. Funcionários e Gerentes são WorkerTypes. Um Funcionário e um Gerente contêm os mesmos elementos. O WorkerType inclui um elemento YearsOfService que o XMap calcula com base na StartDate.

Dados de Entrada XML

O seguinte texto mostra os dados de amostra do documento XML de entrada:

```
<?xml version="1.0" encoding="windows-1252"?>
<Input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="company
Company.xsd" xmlns="company">
  <Company>
    <Name>Hypostores</Name>

    <Department>
      <Name>Customer Service</Name>

      <Employee id="25721195">
        <FirstName>Blair</FirstName>
        <LastName>Conner</LastName>
        <Role>Manager</Role>
        <StartDate>1993-04-21</StartDate>
      </Employee>

      <Employee id="238036220">
        <FirstName>Karina</FirstName>
        <LastName>Rasmussen</LastName>
        <Role>Worker</Role>
        <StartDate>1993-08-15</StartDate>
      </Employee>
    </Department>

    <Department>
      <Name>Research and Development</Name>

      <Employee id="259089785">
        <FirstName>Thaddeus</FirstName>
        <LastName>Burt</LastName>
        <Role>Consultant</Role>
        <StartDate>1998-02-26</StartDate>
      </Employee>

      <Employee id="289021615">
        <FirstName>Christen</FirstName>
        <LastName>Fulton</LastName>
        <Role>Worker</Role>
        <StartDate>1997-11-16</StartDate>
      </Employee>

      <Employee id="761338290">
        <FirstName>Felix</FirstName>
        <LastName>Boyd</LastName>
        <Role>Worker</Role>
        <StartDate>2009-12-29</StartDate>
      </Employee>
    </Department>
  </Company>
```

Os dados podem incluir várias empresas. Cada empresa tem diversos departamentos. Cada funcionário em um departamento tem uma função que é um gerente ou outro tipo de funcionário.

Hierarquias de XML de Entrada e Saída

O Editor de XMap exibe a hierarquia de entrada na área esquerda da exibição e a hierarquia de XML de saída na área direita da exibição.

A seguinte figura mostra algumas hierarquias de XML de entrada e saída:

The figure consists of two side-by-side screenshots of the XMap editor. The left screenshot shows the 'Input' hierarchy for a project named 'Company'. It lists elements like 'Company', 'Name', 'Department', 'Employee', 'id', 'FirstName', 'LastName', 'Role', 'StartDate', and 'Variables' with their respective types and cardinalities. The right screenshot shows the 'Output' hierarchy for a project named 'Department_Organization'. It lists elements like 'Organization', 'noOfEmployees', 'Worker', 'FirstName', 'LastName', 'FullName', 'Id', 'YearsOfService', 'Manager', and 'Variables' with their respective types and cardinalities.

Instruções de Mapeamento no Exemplo

Use a área de grade do editor de XMap para definir as instruções que mapeiam os elementos XML de entrada para os elementos XML de saída. Crie e edite instruções de mapeamento na grade. Defina o contexto, a condição, e a entrada e saída esperadas para um instrução de mapeamento. Você pode adicionar variáveis às instruções de mapeamento.

A figura seguinte mostra as instruções de mapeamento na grade:

Name	Statement Type	Input	Condition	Skip...	Default	On Fail	Output	Mode
1 Company	Repeating Group	tns0:Company				Propagate		Add
2 Department	Repeating Group	tns0:Department				Propagate		Add
3 NameToDepartment	Map	tns0:Name				Propagate	\$deptName	Match or Add
4 MatchOrganization	Group	-				Propagate	tns0:Organization[tns0:Department=\$d...	Match or Add
5 Employee to Worker	Repeating Group	tns0:Employee				Skip Iterati...		Match or Add
6 Employee	Router					Skip		Match or Add
7 Employee to Manager	Option		tns0:Roles="Manager"			Propagate		Match or Add
8 EmployeeToWorker	EmployeeToWorker					Propagate	tns0:Manager	Add
9 Employee to Worker	Default					Propagate		Match or Add
10 EmployeeToWorker	EmployeeToWorker					Propagate	tns0:Worker	Add
11 Increment employee count	Map	dp:output()/@noOfEmployees + 1			1	Propagate	@noOfEmployees	Match or Add

A grade contém as seguintes intruções de mapeamento:

Linha de grade 1, instrução de Grupo de Repetição chamada Empresa

A instrução Empresa é uma instrução de Grupo de Repetição. Ela se repete para cada elemento de Empresa. A instrução oferece um contexto para o resto das instruções na grade. Para cada empresa, a transformação do Data Processor avalia as instruções filho.

Linha de grade 2, instrução de Grupo de Repetição chamado Departamento

A instrução Departamento é uma instrução de Grupo de Repetição. Ela se repete para cada elemento de Departamento. A instrução oferece um contexto para o resto das instruções na grade. Para cada departamento, a transformação do Data Processor avalia as instruções filho.

Linha de grade 3, instrução de Mapeamento chamada NametoDepartment

A instrução NametoDepartment é uma instrução de Mapeamento. Ele mapeia o elemento Nome para uma variável \$deptName.

Linha de grade 4, instrução de Grupo de Repetição chamada MatchOrganization

A instrução MatchOrganization é uma instrução de Grupo de Repetição. Ela tem uma expressão de saída:

```
tns0:Organization[tns0:Department=$deptName]
```

A instrução localiza o elemento Organização na saída que contém um elemento Departamento com o valor em \$deptName. Ou, se o elemento de Departamento não existir, o elemento será criado.

Linha de grade 5, instrução de Grupo de Repetição chamada EmployeeToWorker

A instrução EmployeeToWorker é uma instrução de Grupo de Repetição. Ela se repete para cada elemento de Funcionário.

Linha de grade 6, instrução de Roteador chamada Funcionário

A instrução Funcionário é uma instrução de Roteador. A instrução não tem entrada ou saída.

Linha de grade 7, instrução de Opção chamada EmployeeToMgr

A instrução EmployeeToMgr é uma instrução de Opção. A instrução de Opção contém a seguinte condição:

```
tns0:Role="Gerente".
```

Quando a função é Gerente, a instrução é verdadeira e a transformação do Data Processor avalia as instruções aninhadas dentro da opção Instrução.

Linha de grade 8, instrução Executar XMap chamada EmployeeToWorker

A instrução EmployeeToWorker é uma instrução Executar XMap. Ele chama o XMap XMap_EmployeesToRoles para transmitir os elementos de Funcionário para o tipo Gerente.

Linha de grade 9, instrução Padrão chamada EmployeeToWorker

A instrução EmployeeToWorker é uma instrução Padrão. A transformação do Data Processor executa as instruções filho quando a função funcionário não é um gerente.

Linha de grade 10, instrução Executar XMap chamada EmployeeToWorker

A instrução EmployeeToWorker é uma instrução Executar XMap. Ele chama o XMap XMap_EmployeesToRoles para transmitir os elementos de Funcionário para o tipo Trabalhador.

Linha de grade 11, instrução de Mapeamento chamada IncrementEmployeeCount

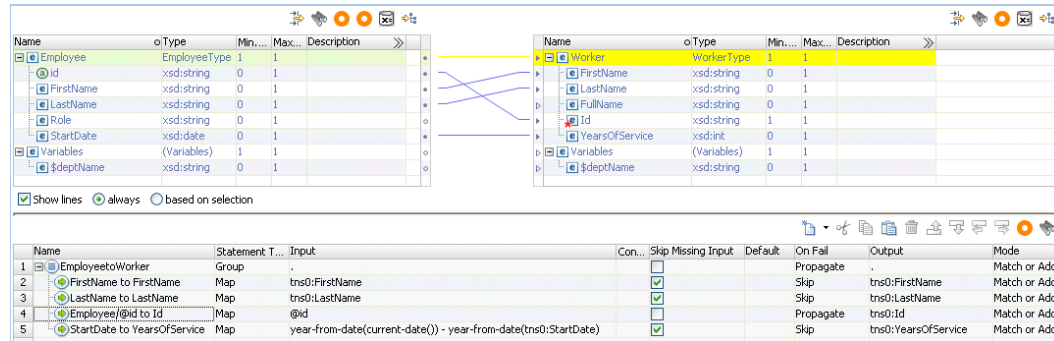
A instrução IncrementEmployeeCount é uma instrução Mapeamento. Ela chama a transformação do Data Processor para adicionar 1 a @noOfEmployees para cada Funcionário que ele itera. A instrução de Mapa contém a seguinte expressão de entrada:

```
dp:output()/@ noOfEmployees + 1.
```

Exemplo de Instruções de Grupo

O XMap EmployeeToWorker move elementos de um funcionário para um trabalhador. O XMap processa um funcionário.

A figura a seguir mostra o XMap EmployeeToWorker no editor de XMap:



A grade contém as seguintes instruções de mapeamento:

Linha de grade 1, instrução de Grupo chamada EmployeeToWorker

A instrução EmployeeToWorker é uma instrução de Grupo. Ela fornece contexto para o restante das instruções de mapeamento.

Linha de grade 2, instrução de Mapeamento chamada FirstNametoFirstName

A instrução FirstNametoFirstName é uma instrução Mapeamento. Ele mapeia o nome para o nome.

Linha de grade 3, instrução de Mapeamento chamada LastNametoLastName

A instrução LastNametoLastName é uma instrução de Mapeamento. Ele mapeia o sobrenome para o sobrenome.

Linha de grade 4, instrução de Mapeamento chamada Employee/@IDtoID

A instrução Employee/@IDtoID é uma instrução Mapeamento. Ele mapeia a ID do funcionário para a ID do funcionário.

Linha de grade 5, instrução de Mapeamento chamada StartDatetoYearsofService

A instrução StartDatetoYearsofService é uma instrução Mapeamento. Ele determina o número de anos de serviço subtraindo uma data inicial de uma data atual.

CAPÍTULO 7

Bibliotecas

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Bibliotecas, 118](#)
- [Estrutura da Biblioteca, 119](#)
- [Propriedades do Elemento, 119](#)
- [Gerenciamento de Biblioteca, 119](#)
- [Editar as Bibliotecas com o Editor de Biblioteca, 120](#)
- [Editar as Bibliotecas com o Editor do IntelliScript, 122](#)

Visão Geral de Bibliotecas

Uma Biblioteca é um objeto de transformação do Processador de Dados que contém componentes predefinidos usados para transformar uma faixa de padrões de mensagem da indústria. Uma transformação do Processador de Dados usa uma Biblioteca para transformar uma entrada de tipo de mensagem da indústria em outros formatos. Você pode criar objetos de Biblioteca para todas as bibliotecas.

Uma Biblioteca contém um grande número de objetos e componentes, como Analisadores, Serializadores e esquemas XML, que transformam a entrada padrão do setor e mensagens de aplicativos específicos em uma saída XML. Uma Biblioteca pode conter objetos para validação de mensagens, reconhecimentos e exibições de diagnósticos. Uma Biblioteca usa objetos para transformar o tipo de mensagem de entrada padrão da indústria em XML e XML em outros formatos.

Você pode criar objetos de biblioteca para bibliotecas ACORD, BAI, CREST, DTCC-NSCC, EDIFACT, EDIT-UCS & WINS, EDI_VICS, EDI-X12, FIX, FpML, HIPAA, HIX, HL7, IATA, IDS, MDM Mapping, NACHA, NCPDP, SEPA, SWIFT e Telekurs.

Você pode usar um editor de Biblioteca dedicada para editar as especificações de biblioteca das bibliotecas DTCC-NSCC, EDIFACT, EDI-X12, HIPAA, HL7, and SWIFT. Uma mensagem de Biblioteca contém um elemento raiz, elementos contêiner e elementos de dados. Os tipos de elementos de contêiner e de dados variam de acordo com tipo de mensagem. Você pode adicionar e excluir elementos, assim como configurar as propriedades de elementos, para alterar as configurações de validação.

Para obter mais informações sobre os tipos de mensagem da indústria, consulte o *Guia de Bibliotecas do Data Transformation*.

Estrutura da Biblioteca

Uma Biblioteca é um conjunto de transformações. Todas as Bibliotecas contêm Analisadores, Serializadores e esquemas XML. Algumas Bibliotecas contêm componentes adicionais para validação de mensagens, confirmações e exibições de diagnósticos. Um objeto de Biblioteca é um conjunto de componentes que convertem um tipo específico de mensagem do setor.

Ao criar uma transformação de Processador de Dados, você pode incluir um objeto de Biblioteca em vez de criar os seus próprios Scripts para transformar um tipo de mensagem padrão do setor. Você pode usar uma Biblioteca sem modificação ou editá-la com base nos seus requisitos.

Cada objeto de Biblioteca transforma um determinado padrão do setor. Por exemplo, a Biblioteca HL7 contém componentes para cada um dos tipos de mensagens ou estruturas de dados disponíveis no padrão de mensagens HL7 para sistemas médicos de informações.

A Biblioteca contém tipos específicos de mensagens. Por exemplo, a Biblioteca HL7 contém mensagens como:

```
ADT_A01_Admit_a_Patient
ADT_A02_Transfer_a_Patient
```

Propriedades do Elemento

Uma mensagem de Biblioteca contém um elemento raiz, elementos contêiner e elementos de dados. Os tipos de elementos de contêiner e de dados variam de acordo com tipo de mensagem. Você pode configurar as propriedades do elemento de contêiner e do elemento de dados com o editor de Bibliotecas.

As propriedades têm as seguintes categorias:

Configurações Globais

As propriedades têm o mesmo valor para um elemento com mais de uma instância, cada um ocupando uma posição diferente na hierarquia. Qualquer alteração na propriedade de Configurações Globais será exibida em todas as instâncias do elemento.

Configurações Posicionais

Propriedades cujo valor pode ser diferente para cada instância do elemento na hierarquia.

As propriedades do elemento são nomeadas de acordo com os padrões da indústria. O editor de Biblioteca exibe propriedades diferentes para Bibliotecas diferentes.

Gerenciamento de Biblioteca

Você pode usar o editor de Biblioteca para personalizar as estruturas e os elementos do tipo de mensagem de bibliotecas DTCC-NSCC, EDIFACT, EDI-X12, HIPAA, HL7 e SWIFT. Para alterar a estrutura do tipo de mensagem, use o editor de Biblioteca para adicionar, editar e excluir elementos da mensagem de biblioteca.

Talvez você queira alterar a forma como um tipo de mensagem é transformada. Uma transformação de Biblioteca contém um grande número de objetos e componentes, como Scripts com Analisadores, Serializadores e esquemas XML, os quais definem a transformação. Uma transformação de Biblioteca pode conter objetos para validação de mensagens, confirmações e exibições de diagnóstico.

Uma transformação de Biblioteca usa seus objetos para transformar o tipo de mensagens de entrada padrão do setor em XML e de XML em outros formatos. Para acessar e editar os Scripts, o XMaps e os esquemas criados com o editor de Biblioteca, você deve gerar os objetos de Biblioteca. Gere objetos de Biblioteca somente se você precisar alterar os objetos de Biblioteca que você não podem ser alterados com o editor de Biblioteca.

Depois de gerar os objetos de Biblioteca, use o editor do IntelliScript para editar os Analisadores, os Serializadores e os Mapeadores. Por exemplo, você deseja alterar a estrutura de saída para atender aos seus requisitos. Gere os objetos de biblioteca e edite os Scripts com o editor do IntelliScript.

Quando você cria uma Biblioteca para os pacotes de biblioteca que não são compatíveis com o editor de Biblioteca, a objetos de biblioteca são gerados previamente. Você não precisa gerar os objetos de Biblioteca. Você pode usar o editor do IntelliScript para editar diretamente os componentes de Script para essas bibliotecas.

Depois de gerar objetos de Biblioteca, ou se estes tiverem sido pré-gerados, você não poderá editar os elementos da Biblioteca com o editor de Biblioteca. Para usar o editor de Biblioteca novamente, você deve descartar os objetos de Biblioteca gerados para essas bibliotecas que podem ser editadas com o editor de Biblioteca. Por exemplo, talvez você decida que deseja adicionar campos de entrada, mas não deseja alterar a estrutura da saída. Descarte os objetos de Biblioteca e, depois disso, adicione elementos personalizados com o editor de Biblioteca.

Nota: Qualquer alteração feita nos objetos de Biblioteca gerados será perdida quando você descartar os objetos gerados.

Exemplo da Biblioteca para EDI-X12

Você e seus fornecedores usam o Pedido de Compra X12 850 padrão para documentar e processar pedidos. Você deseja alterar os padrões do setor para especificações de tipos de mensagem de biblioteca de forma a oferecer suporte para os seus processos internos.

O setor de fabricação de móveis precisa de elementos padrão como modelo, cor e tamanho, além de um elemento personalizado para o tipo de tecido. Ele não precisa de um número de lote ou de uma data de expiração. Use o editor de Biblioteca para remover elementos e adicionar elementos personalizados.

Você também pode usar o editor de Biblioteca para alterar propriedades de elementos, adicionar segmentos e copiar elementos. Por exemplo, altere a propriedade `element`, que define a lista de cores, para adicionar novas cores. Para adicionar um elemento de código de fabricante ao segmento de tecido e ao segmento de utensílios, você pode copiar esse elemento.

Você executa uma transformação de Processador de Dados que contém sua Biblioteca personalizada. A entrada de texto dos seus fornecedores é formatada em uma versão modificada do Pedido de Compra X12 850 padrão. O formato de entrada modificado corresponde às alterações que você fez na Biblioteca. O texto é transformado em uma saída hierárquica que pode ser enviada a outras transformações para processamento adicional.

Editar as Bibliotecas com o Editor de Biblioteca

Você pode usar um editor de Biblioteca dedicado para editar as especificações da Biblioteca para as bibliotecas EDI-X12, SWIFT, HL7, HIPAA, DTCC-NSCC e EDIFACT.

O editor de Biblioteca permite que você personalize os elementos e as estruturas do tipo de mensagem com um editor que é personalizado para cada tipo de mensagem. Você pode adicionar, editar e excluir elementos do objeto de Biblioteca com o Editor de Biblioteca.

Adicionando um Elemento com o Editor de Biblioteca

Uma mensagem de Biblioteca contém um elemento raiz, elementos contêiner e elementos de dados. Use o editor de Biblioteca para adicionar um elemento a uma mensagem.

1. Na exibição de **Objetos**, selecione o objeto do editor de Biblioteca e clique em **Abrir**.
O editor de Bibliotecas é exibido.
2. Clique no ícone **Adicionar elemento** e escolha onde você deseja adicionar o elemento.
 - **Novo**. Anexe o elemento ao fim da lista de elementos.
 - **Inserir Acima**. Insira o elemento acima do elemento selecionado no editor de Bibliotecas.
 - **Inserir Abaixo**. Insira o elemento abaixo do elemento selecionado no editor de Bibliotecas.
 - **Inserir Dentro**. Insira o elemento dentro do elemento do recipiente no editor de Bibliotecas.
3. Selecione se deseja copiar um elemento existente ou criar um novo elemento e clique em **OK**.

Editando as Propriedades do Elemento com o Editor de Biblioteca

Use o editor de Biblioteca para editar um elemento de contêiner ou um elemento de dados em uma biblioteca. Ao alterar as propriedades de um elemento, você altera as especificações para o tipo de mensagem.

1. Na janela **Definição de Mensagem** do editor de Biblioteca, selecione o elemento para editar.
A janela **Propriedades** exibirá as propriedades do elemento.
2. Na janela **Propriedades**, selecione uma propriedade e insira o novo valor de propriedade.
Se você digitar um tipo de valor incorreto ou um valor fora da faixa, receberá uma solicitação para corrigir o valor.

Testando uma Biblioteca

Testar o objeto de Biblioteca na exibição **Visualizador de Dados**.

Antes de testar a Biblioteca, selecione um arquivo de entrada de amostra para ser testado.

1. Para selecionar uma arquivo de entrada de amostra, no painel **Geral** do editor de Biblioteca, próximo ao campo **entrada de Amostra**, procure um arquivo de entrada de amostra para ser selecionado.
2. Clique na exibição do **Visualizador de Dados**.
3. Para selecionar o tipo de mensagem de Biblioteca que você editou como o componente para execução, clique em **Sincronizar com Editor**.
4. Clique em **Executar**.
A Developer tool executa o Analisador de objeto de Biblioteca. Os resultados de saída são exibidos na janela **Saída**.
5. Se houver erros de validação, o painel **Erros de Saída** na janela **Saída** lista os erros e as respectivas localizações. Para localizar a origem do erro, clique duas vezes nele.
6. Para exibir os arquivos de erro de validação, clique nos nomes de arquivo no painel **Saídas Adicionais** na janela **Saída**.
Quando você clica no nome de arquivo `ErrorsFound.txt` ou `Errors.xml`, o arquivo é aberto em um navegador externo.

Gerando os Objetos de Biblioteca

Gere os objetos de Biblioteca para que você possa acessá-los diretamente com os editores da transformação de Processador de Dados. Depois de gerar os objetos de Biblioteca, você não pode usar o editor de Biblioteca para editar a Biblioteca.

É necessário gerar objetos de Biblioteca para acessar e editar os analisadores, mapeadores, serializadores e esquemas XML pré-configurados associados à Biblioteca.

1. Na exibição **Objetos**, clique com o botão direito na Biblioteca e selecione **Gerar Objetos de Biblioteca**.
2. Para acessar os objetos de Biblioteca, selecione **Sim**.

A Developer tool cria uma pasta **Objetos de Biblioteca Gerados**, que contém os objetos de Biblioteca. O componente de inicialização é gerado.

Nota: Se você deseja alterar quais objetos de biblioteca são gerados, selecione um componente diferente.

3. Para editar um Script ou esquema, selecione-o e clique em **Abrir**.
Use o editor do IntelliScript para editar o Script.

Descartando os Objetos de Biblioteca

Para editar os elementos da Biblioteca com o editor de Bibliotecas, você deve remover os objetos da Biblioteca que você gerou. Quando você descartar os objetos da Biblioteca, qualquer alteração que tenha feito será perdida.

1. Na exibição de **Objetos**, clique com o botão direito na Biblioteca e selecione **Descartar Gerar Objetos da Biblioteca**.
2. Para descartar os objetos da Biblioteca dos editores de transformação do Processador de Dados, selecione **Sim**.

A pasta **Objetos Gerados da Biblioteca** será descartada. Os componentes e objetos da Biblioteca ainda existem, mas não poderão ser acessados por meio dos editores de transformação do Processador de Dados.

Editar as Bibliotecas com o Editor do IntelliScript

Ao criar uma Biblioteca para as bibliotecas ACORD, BAI, FIX, HIX, IATA, IDS, NACHA, NCPDP, Telekurs, Bloomberg, SEPA, FpML e Thompson Reuters, você pode usar o editor do IntelliScript para editar diretamente os componentes de Script dessas bibliotecas. Você não precisa gerar os objetos de Biblioteca.

Para as bibliotecas EDI-X12, SWIFT, HL7, HIPAA, DTCC-NSCC, Edifact, SEPA e FpML, você deve gerar os objetos de Biblioteca para acessar e editar os Scripts, XMaps e esquemas criados usando o editor de Biblioteca com o editor do IntelliScript.

O editor do IntelliScript é uma ferramenta gráfica que você utiliza para editar Scripts. Use o editor do IntelliScript para adicionar componentes de Script ao Script e configurar as propriedades do componente de Script. Para obter mais informações sobre os Scripts e o editor do IntelliScript, consulte [“Visão Geral de Scripts” na página 139](#).

Se você importou um projeto de Biblioteca de uma versão anterior como um serviço do Data Transformation, poderá editar os componentes de Script com o editor do IntelliScript.

CAPÍTULO 8

Objeto de Esquema

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Objetos de Esquema, 123](#)
- [Arquivos de Esquema, 123](#)
- [Exibição Visão Geral de Objetos de Esquema, 124](#)
- [Exibição Esquema de Objetos de Esquema, 125](#)
- [Exibição Avançado de Objetos de Esquema, 130](#)
- [Criando um objeto de esquema, 131](#)
- [Atualizações de Esquema, 132](#)

Visão Geral de Objetos de Esquema

Um objeto de esquema é um esquema hierárquico que você importa para o repositório do modelo. Após a importação do esquema, você pode exibir os componentes do esquema na Developer tool. Você pode importar um esquema Avro, Parquet, XML ou JSON. A Developer tool converte o esquema em um arquivo .xsd no repositório do modelo.

Ao criar um serviço da Web SOAP, você pode definir a estrutura desse serviço com base em um esquema hierárquico. Ao criar um serviço da Web sem um WSDL, você pode definir as operações, a entrada, a saída e as assinaturas de falha com base nos tipos e elementos que o esquema define.

Ao importar um esquema, é possível editar propriedades de esquema gerais na exibição **Visão Geral**. Edite propriedades avançadas na exibição **Avançado**. Visualize o conteúdo no arquivo de esquema na exibição **Esquema**.

Arquivos de Esquema

É possível adicionar vários arquivos .xsd em nível de raiz a um objeto de esquema. Também é possível remover arquivos .xsd em nível de raiz de um objeto de esquema.

Quando você adiciona um arquivo de esquema, a ferramenta Developer importa todos os arquivos .xsd que são importados ou estão incluídos no arquivo adicionado. A ferramenta Developer valida os arquivos adicionados com base nos arquivos que fazem parte do objeto de esquema. A ferramenta Developer não permitirá a adição de um arquivo se ele estiver em conflito com outro arquivo que faz parte do objeto de esquema.

Por exemplo, um objeto de esquema contém o arquivo de esquema raiz "BostonCust.xsd". Você deseja adicionar o arquivo de esquema raiz "LACust.xsd" ao objeto de esquema. Ambos os arquivos de esquema têm o mesmo espaço de nome de destino e definem um elemento denominado "Customer". Quando você tentar adicionar o arquivo de esquema LACust.xsd ao objeto de esquema, a ferramenta Developer perguntará se você deseja manter o arquivo BostonCust.xsd ou se prefere substituí-lo pelo arquivo LACust.xsd.

Você pode usar o atributo `xsd:nillable` para marcar os elementos XSD como anuláveis. Quando você marca um elemento como anulável, o elemento correspondente no arquivo XML permite valores nulos.

É possível remover qualquer arquivo de esquema em nível de raiz. Se você remover um arquivo de esquema, a ferramenta Developer alterará para `xs:string` o tipo dos elementos que foram definidos por esse arquivo de esquema.

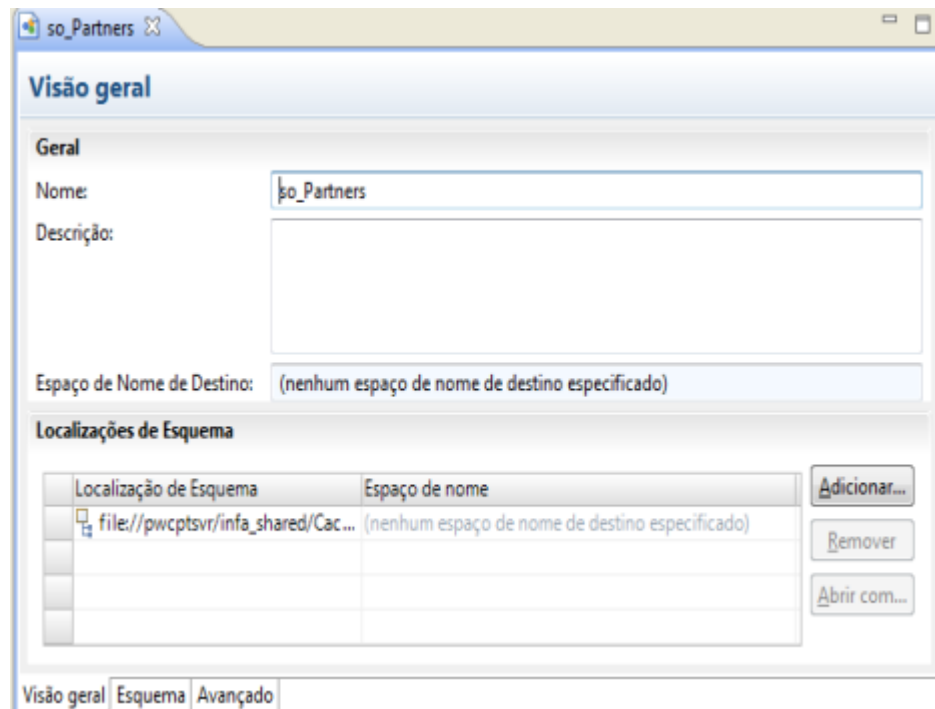
Para adicionar um arquivo de esquema, selecione a exibição **Visão Geral** e clique no botão **Adicionar** ao lado da lista **Localizações de Esquema**. Em seguida, selecione o arquivo de esquema. Para remover um arquivo de esquema, selecione esse arquivo e clique no botão **Remover**.

Exibição Visão Geral de Objetos de Esquema

Selecione a exibição **Visão Geral** para atualizar o nome ou descrição do esquema, exibir espaços de nome e gerenciar arquivos de esquema.

A exibição **Visão Geral** mostra o nome, a descrição e o espaço de nome de destino do esquema. É possível editar o nome e a descrição do esquema. O espaço de nome de destino mostra o espaço de nome ao qual os componentes do esquema pertencem. Se nenhum espaço de nome de destino aparecer, significa que os componentes do esquema não pertencem a um espaço de nome.

A seguinte figura mostra a exibição **Visão Geral** de um objeto de esquema:



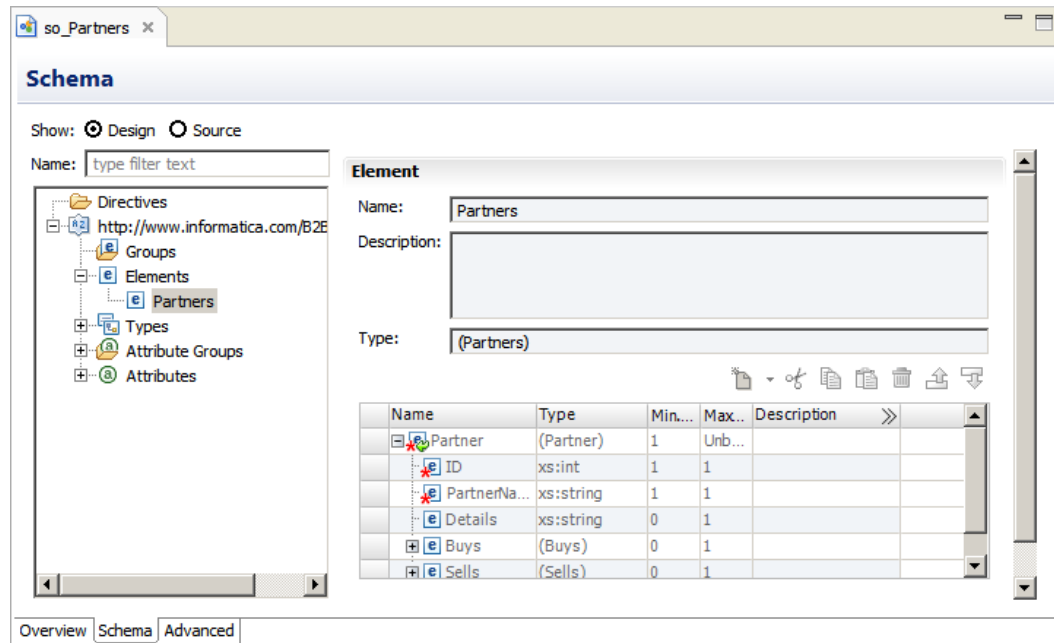
A área **Localizações de Esquema** lista os arquivos de esquema e os espaços de nome. É possível adicionar vários arquivos .xsd raiz. Se um arquivo de esquema incluir ou importar outros arquivos de esquema, a ferramenta Developer incluirá os arquivos .xsd filho no esquema.

Exibição Esquema de Objetos de Esquema

A exibição mostra uma lista alfabética de grupos, elementos, tipos, grupos de atributos e atributos no esquema. Quando você seleciona um grupo, elemento, tipo, grupo de atributos ou atributo na exibição **Esquema**, as propriedades aparecem no painel direito. Também é possível visualizar cada arquivo .xsd na exibição **Esquema**.

A exibição **Esquema** fornece uma lista dos espaços de nome e arquivos .xsd no objeto de esquema.

A seguinte figura mostra a exibição **Esquema** de um objeto de esquema:



É possível realizar as seguintes ações na exibição **Esquema**:

- Para exibir a lista de construções de esquema, expanda a pasta **Diretivas**. Para exibir o espaço de nome, o prefixo e a localização, selecione uma construção de esquema na lista.
- Para exibir o prefixo do espaço de nome, o prefixo gerado e a localização, selecione um espaço de nome. É possível alterar o prefixo gerado.
- Para exibir o objeto de esquema como um arquivo .xsd, selecione **Origem**. Se o objeto de esquema incluir outros esquemas, você poderá selecionar qual arquivo .xsd deseja exibir.
- Para exibir uma lista alfabética de grupos, elementos, tipos, grupos de atributos e atributos em cada espaço de nome do esquema, selecione **Design**. É possível inserir um ou mais caracteres no campo **Nome** para filtrar os grupos, elementos, tipos, grupos de atributos e atributos por nome.
- Para exibir as propriedades de elementos, selecione um grupo, elemento, tipo, grupo de atributos ou atributo. A ferramenta Developer mostra campos diferentes no painel direito com base no objeto selecionado.

Ao exibir tipos, você pode ver se um tipo é derivado de outro. A interface mostra o tipo pai. Ela também mostra se o elemento filho herdou valores por restrição ou extensão.

Propriedades de Espaço de Nome

A exibição **Espaço de Nome** mostra o prefixo e a localização de um espaço de nome selecionado.

O espaço de nome associado a cada arquivo de esquema faz distinção entre elementos provenientes de diferentes origens, mas que apresentam os mesmos nomes. Uma referência de URI (Identificador de Recurso Uniforme) define a localização do arquivo que contém os elementos e os nomes de atributos.

Quando você importa um esquema que contém mais de um espaço de nome, a Developer tool adiciona os espaços de nome ao objeto de esquema. Quando o arquivo de esquema inclui outros esquemas, os espaços de nome desses esquemas também são incluídos.

A ferramenta Developer cria um prefixo gerado para cada espaço de nome. Quando o esquema não contém um prefixo, a Developer tool gera o prefixo de espaço de nome tns0 e incrementa o número de prefixo para cada prefixo de espaço de nome adicional. A ferramenta Developer reserva o prefixo de espaço de nome xs. Se você importar um esquema que contém o prefixo de espaço de nome xs, a Developer tool criará o prefixo gerado xs1. A ferramenta Developer aumenta o número do prefixo quando o esquema contém o valor do prefixo gerado.

Por exemplo, Customer_Orders.xsd tem um espaço de nome. O esquema inclui outro esquema, Customers.xsd. O esquema Customers tem um espaço de nome diferente. A ferramenta Developer atribui o prefixo tns0 ao espaço de nome Customer_Orders e o prefixo tns1 ao espaço de nome Customers.

Para ver a localização e o prefixo do espaço de nome, selecione um espaço de nome na exibição **Esquema**.

Quando você cria um serviço da Web a partir de mais de um objeto de esquema, cada espaço de nome deve ter um prefixo exclusivo. É possível modificar o prefixo gerado para cada espaço de nome.

Propriedades do Elemento

Um elemento é um tipo simples ou complexo. Um tipo complexo contém outros tipos. Quando você seleciona um elemento na exibição **Esquema**, a ferramenta Developer lista os elementos filho e as propriedades no painel direito da tela.

A seguinte tabela descreve as propriedades de elemento que aparecem quando você seleciona um elemento:

Propriedade	Descrição
Nome	O nome do elemento.
Descrição	Descrição do tipo.
Tipo	O tipo de elemento.

A seguinte tabela descreve as propriedades de elemento filho que aparecem quando você seleciona um elemento:

Propriedade	Descrição
Nome	O nome do elemento.
Tipo	O tipo de elemento.

Propriedade	Descrição
Mínimo de Ocorrências	O número mínimo de vezes que o elemento pode ocorrer em determinado ponto de uma instância.
Máximo de Ocorrências	O número máximo de vezes que o elemento pode ocorrer em determinado ponto de uma instância.
Descrição	Descrição do elemento.

Para exibir propriedades de elementos filho adicionais, clique na seta dupla da coluna Descrição para expandir a janela.

A seguinte tabela descreve as propriedades de elementos filho adicionais que aparecem quando você expande a coluna Descrição:

Propriedade	Descrição
Valor Fixo	Um valor específico para um elemento que não muda.
Anulável	O elemento pode ter valores nulos. Um elemento nulo possui marcas de elemento, mas não tem valor nem conteúdo.
Abstrato	O elemento é um tipo abstrato. Uma instância deve incluir tipos derivados desse tipo. Um tipo abstrato não é um tipo válido sem tipos de elementos derivados.
Valor Mínimo	O valor mínimo para um elemento em uma instância.
Valor Máximo	O valor máximo para um elemento em uma instância.
Comprimento Mínimo	O comprimento mínimo de um elemento. O comprimento é em bytes, caracteres ou itens com base no tipo de elemento.
Comprimento Máximo	O comprimento máximo de um elemento. O comprimento é em bytes, caracteres ou itens com base no tipo de elemento.
Enumeração	Uma lista de todos os valores legais para um elemento.
Padrão	Um padrão de expressão que define valores de elementos válidos.

Propriedades avançadas do elemento

Para visualizar as propriedades avançadas de um elemento, selecione esse elemento na exibição **Esquema**. Clique em **Avançado**.

A seguinte tabela descreve as propriedades avançadas de elementos:

Propriedade	Descrição
Abstrato	O elemento é um tipo abstrato. Uma mensagem SOAP deve incluir tipos derivados desse tipo. Um tipo abstrato não é um tipo válido sem tipos de elementos derivados.
Bloquear	Impede que um elemento derivado apareça na hierarquia no lugar deste elemento. O valor de bloqueio podem conter "#all" ou uma lista que inclui extensão, restrição ou substituição.

Propriedade	Descrição
Final	Impede que o esquema estenda ou restrinja o tipo simples como um tipo derivado.
Grupo de Substituição	O nome de um elemento a ser substituído pelo elemento.
Anulável	O elemento pode ter valores nulos. Um elemento nulo possui marcas de elemento, mas não tem valor nem conteúdo.

Propriedades de Tipos Simples

Um elemento de tipo simples é um elemento que contém texto não estruturado. Quando você seleciona um elemento de tipo simples na exibição **Esquema**, informações sobre esse elemento aparecem no painel direito.

A seguinte tabela descreve as propriedades que você pode exibir para um tipo simples:

Propriedade	Descrição
Tipo	Nome do elemento.
Descrição	Descrição do elemento.
Variedade	Define se o tipo simples é union, list, anyType ou atomic. Um elemento atômico não contém outros elementos ou atributos.
Tipos de membros	Uma lista dos tipos em uma construção UNION.
Tipo de item	O tipo de elemento.
de base.	O tipo base de um elemento atômico, como um inteiro ou uma string.
Comprimento Mínimo	O tamanho mínimo para um elemento. O comprimento é em bytes, caracteres ou itens com base no tipo de elemento.
Comprimento Máximo	O tamanho máximo para um elemento. O comprimento é em bytes, caracteres ou itens com base no tipo de elemento.
Reduzir espaço em branco	Remove espaços em branco à esquerda e à direita. Reduz vários espaços a um único espaço.
Enumerações	Restringem o tipo à lista de valores legais.
Padrões	Restringem o tipo a valores definidos por uma expressão de padrão.

Propriedades Avançadas de Tipos Simples

Para visualizar propriedades avançadas para um tipo simples, selecione esse tipo simples na exibição **Esquema**. Clique em **Avançado**.

As propriedades avançadas aparecem abaixo das propriedades do tipo simples.

A seguinte tabela descreve a propriedade avançada para um tipo simples:

Propriedade	Descrição
Final	Impede que o esquema estenda ou restrinja o tipo simples como um tipo derivado.

Propriedades de Tipos Complexos

Um tipo complexo é um elemento que contém outros elementos e atributos. Um tipo complexo contém elementos que são tipos simples ou complexos. Quando você seleciona um tipo complexo na exibição **Esquema**, a ferramenta Developer lista os elementos filhos e suas propriedades no painel direito da tela.

A seguinte tabela descreve as propriedades de tipos complexos:

Propriedade	Descrição
Nome	O nome do tipo.
Descrição	Descrição do tipo.
Herdar de	Nome do tipo pai.
Herdar por	Restrição ou extensão. Um tipo complexo é derivado de um tipo pai. O tipo complexo pode reduzir os elementos ou atributos do pai. Ou, ele pode adicionar elementos e atributos.

Para exibir propriedades de cada elemento em um tipo complexo, clique na seta dupla na coluna Descrição para expandir a janela.

Propriedades Avançadas de Tipos Complexos

Para visualizar as propriedades avançadas de um tipo complexo, selecione esse elemento na exibição **Esquema**. Clique em **Avançado**.

A seguinte tabela descreve as propriedades avançadas de um tipo ou elemento complexo:

Propriedade	Descrição
Abstrato	O elemento é um tipo abstrato. Uma mensagem SOAP deve incluir tipos derivados desse tipo. Um tipo abstrato não é um tipo válido sem tipos de elementos derivados.
Bloquear	Impede que um elemento derivado apareça no esquema no lugar deste elemento. O valor de bloqueio podem conter "#all" ou uma lista que inclui extensão, restrição ou substituição.
Final	Impede que o esquema estenda ou restrinja o tipo simples como um tipo derivado.
Grupo de Substituição	O nome de um elemento a ser substituído pelo elemento.
Anulável	O elemento pode ter valores nulos. Um elemento nulo possui marcas de elemento, mas não tem valor nem conteúdo.

Propriedades do Atributo

Um atributo é um tipo simples. Elementos e tipos complexos contêm atributos. Atributos globais aparecem como parte do esquema. Quando você seleciona um atributo global na exibição **Esquema**, a ferramenta Developer lista propriedades de atributos e propriedades de tipos relacionados no painel direito da tela.

A seguinte tabela descreve as propriedades de atributos:

Propriedade	Descrição
Nome	O nome do atributo.
Descrição	Descrição do atributo.
Tipo	O tipo de atributo.
Valor	O valor do tipo de atributo. Indica se o valor do tipo de atributo é fixo ou tem um valor padrão. Se nenhum valor for definido, a propriedade exibirá o padrão = 0.

A seguinte tabela descreve as propriedades de tipos:

Propriedade	Descrição
Comprimento Mínimo	O comprimento mínimo do tipo. O comprimento é em bytes, caracteres ou itens com base no tipo.
Comprimento Máximo	O tamanho máximo do tipo. O comprimento é em bytes, caracteres ou itens com base no tipo.
Reduzir Espaço em Branco	Remove espaços em branco à esquerda e à direita. Reduz vários espaços a um único espaço.
Enumerações	Restringem o tipo à lista de valores legais.
Padrões	Restringem o tipo a valores definidos por uma expressão de padrão.

Exibição Avançado de Objetos de Esquema

Exiba propriedades avançadas para o objeto de esquema.

A seguinte tabela descreve propriedades avançadas para um objeto de esquema:

Nome	Valor	Descrição
elementFormDefault	Qualificado ou Não Qualificado	Determina se os elementos devem ou não ter um espaço de nome. O esquema qualifica os elementos com um prefixo ou por uma declaração de espaço de nome de destino. O valor não qualificado significa que os elementos não precisam de um espaço de nome.

Nome	Valor	Descrição
attributeFormDefault	Qualificado ou Não Qualificado	Determina se atributos declarados localmente devem ter um espaço de nome. O esquema qualifica os atributos com um prefixo ou por uma declaração de espaço de nome de destino. O valor não qualificado significa que os atributos não precisam de um espaço de nome.
Localização do arquivo	Caminho completo para o arquivo .xsd	A localização do arquivo .xsd quando você o importou.

Criando um objeto de esquema

Você pode importar um arquivo de esquema hierárquico ou um arquivo de amostra para criar um objeto de esquema no repositório.

1. Selecione um projeto ou uma pasta na exibição **Object Explorer**.
2. Clique em **Arquivo > Novo > Esquema**.
A caixa de diálogo **Novo Esquema** é exibida.
3. Para importar um arquivo de esquema, selecione **Criar do esquema** e, em seguida, procure e selecione um arquivo de esquema hierárquico.

Você pode inserir um URI ou uma localização no sistema de arquivos para procurar. A Developer tool valida o esquema escolhido. Examine as mensagens de validação. Você pode selecionar um arquivo de esquema Avro, Parquet, JSON ou .xsd.
Nota: Se o URI contiver caracteres diferentes do inglês, a importação poderá falhar. Copie o URI para a barra de endereço de qualquer navegador. Copie a localização de volta para o navegador. A Developer tool aceita o URI codificado do navegador.
4. Para criar um esquema de um arquivo de amostra, selecione **Criar de um arquivo de amostra** e, em seguida, procure e selecione um arquivo hierárquico.

Você pode selecionar um arquivo Avro, Parquet, JSON ou XML.
Nota: Se você selecionar um arquivo com uma extensão diferente que contém o conteúdo do Avro, Parquet, JSON ou XML, o assistente reconhecerá o conteúdo do arquivo.
5. Como opção, altere o nome do esquema.
6. Clique em **Avançar** para exibir uma lista de elementos e tipos no esquema.
7. Clique em **Concluir** para importar o esquema.

O esquema aparece em Objetos de Esquema, na exibição **Object Explorer**. A Developer tool armazena o esquema como um arquivo .xsd.
8. Para alterar o prefixo gerado para um espaço de nome de esquema, selecione o espaço de nome na exibição **Object Explorer**. Altere a propriedade **Prefixo Gerado** na exibição **Espaço de Nome**.

Atualizações de Esquema

Você pode atualizar um objeto de esquema quando os elementos, atributos, tipos ou outros componentes do esquema são alterados. Ao atualizar um objeto de esquema, a ferramenta Developer atualiza os objetos que usam o esquema.

Você pode atualizar um objeto de esquema por meio dos seguintes métodos:

Sincronizar o esquema.

Sincronizar um objeto de esquema quando você atualiza os arquivos de esquema fora da ferramenta Developer. Quando você sincroniza um objeto de esquema, a ferramenta Developer reimporta todos os arquivos de esquema .xsd que contêm alterações.

Editar um arquivo de esquema.

Edite um arquivo de esquema quando quiser atualizar um arquivo dentro da ferramenta Developer. Quando você edita um arquivo de esquema, a ferramenta Developer o abre no editor normalmente utilizado para arquivos .xsd. É possível abrir o arquivo em um editor diferente ou definir um editor de padrão para arquivos .xsd na ferramenta Developer.

Você pode usar um esquema para definir tipos de elementos em um serviço da Web. Quando você atualiza um esquema que está incluído no WSDL de um serviço da Web, a ferramenta Developer atualiza esse serviço da Web e o marca como alterado no momento em que ele é aberto. Quando a ferramenta Developer compara o novo esquema com o esquema antigo, ela identifica componentes do esquema por meio do atributos de nome.

Se nenhum atributo de nome mudar, a ferramenta Developer atualizará o serviço da Web com as alterações do esquema. Por exemplo, você edita um arquivo de esquema na ferramenta Developer e altera o atributo maxOccurs para o elemento "Item" de 10 para "não associado". Quando o arquivo é salvo, a ferramenta Developer atualiza o atributo maxOccurs em todos os serviços da Web que fazem referência ao elemento Item.

Se um atributo de nome mudar, a ferramenta Developer marcará o serviço da Web como alterado quando você o abrir. Por exemplo, você edita um esquema fora da ferramenta Developer e altera o nome de um tipo de elemento complexo de "Order" para "CustOrder". Em seguida, sincroniza o esquema. Quando você abre um serviço da Web que faz referência ao elemento, a ferramenta Developer marca o nome desse serviço Web no editor com um asterisco para indicar que ele contém alterações. A ferramenta Developer adiciona o tipo de elemento CustOrder ao serviço da Web, mas não remove o tipo de elemento Order. Como a ferramenta Developer não pode mais determinar o tipo do elemento Order, ele altera o tipo de elemento para xs:string.

Sincronização de Esquema

Você pode sincronizar um objeto de esquema quando os componentes do esquema são alterados. Quando você sincroniza um objeto de esquema, a ferramenta Developer reimporta os metadados desse objeto a partir dos arquivos de esquema.

Use a sincronização de esquemas quando você fizer alterações complexas no objeto de esquema fora da ferramenta Developer. Por exemplo, é possível sincronizar um esquema depois de realizar as seguintes ações:

- Fazer alterações em vários arquivos de esquema.
- Adicionar ou remover arquivos de esquema no esquema.
- Alterar elementos para importação ou inclusão.

A ferramenta Developer valida os arquivos de esquema antes de atualizar o objeto de esquema. Se os arquivos de esquema contiverem erros, a ferramenta Developer não os importará.

Para sincronizar um objeto de esquema, clique nele com o botão direito na exibição **Object Explorer** e selecione **Sincronizar**.

Edições de Arquivos de Esquema

É possível editar um arquivo de esquema dentro da ferramenta Developer para atualizar componentes de esquema.

Edite um arquivo de esquema na ferramenta Developer para fazer atualizações secundárias em um pequeno número de arquivos. Por exemplo, você pode fazer uma das seguintes atualizações secundárias em um arquivo de esquema:

- Alterar os atributos minOccurs ou maxOccurs para um elemento.
- Adicionar um atributo a um tipo complexo.
- Alterar um tipo de objeto simples.

Quando você edita um arquivo de esquema, a ferramenta Developer abre uma cópia temporária desse arquivo em um editor. Você pode editar arquivos de esquema com o editor do sistema utilizado para arquivos .xsd ou pode selecionar outro editor. Você também pode definir o editor padrão da ferramenta Developer para arquivos .xsd. Salvar o arquivo de esquema temporário depois de editá-lo.

A ferramenta Developer valida o arquivo temporário antes de atualizar o objeto de esquema. Se o arquivo de esquema contiver erros ou componentes em conflito com outros arquivos de esquema no objeto de esquema, a ferramenta Developer não importará esse arquivo em questão.

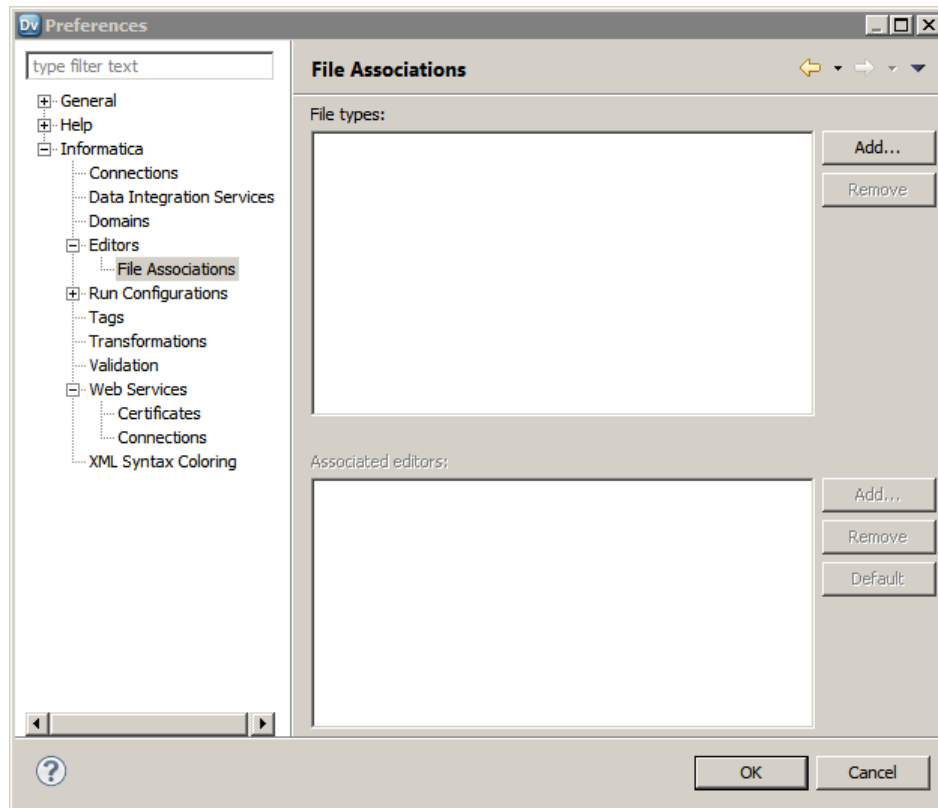
Nota: Quando você edita e salva o arquivo de esquema temporário, a ferramenta Developer não atualiza o arquivo de esquema que aparece na lista **Localizações de Esquema**. Se você sincronizar um objeto de esquema depois de editar um arquivo de esquema na ferramenta Developer, a operação de sincronização substituirá as edições.

Definindo um Editor de Arquivo de Esquema Padrão

É possível definir o editor padrão aberto pela ferramenta Developer quando você edita um arquivo de esquema.

1. Clique em **Janela > Preferências**.
A caixa de diálogo **Preferências** é exibida.
2. Clique em **Editores > Associações de Arquivo**.

A página **Associações de Arquivo** da caixa de diálogo **Preferências** é exibida.



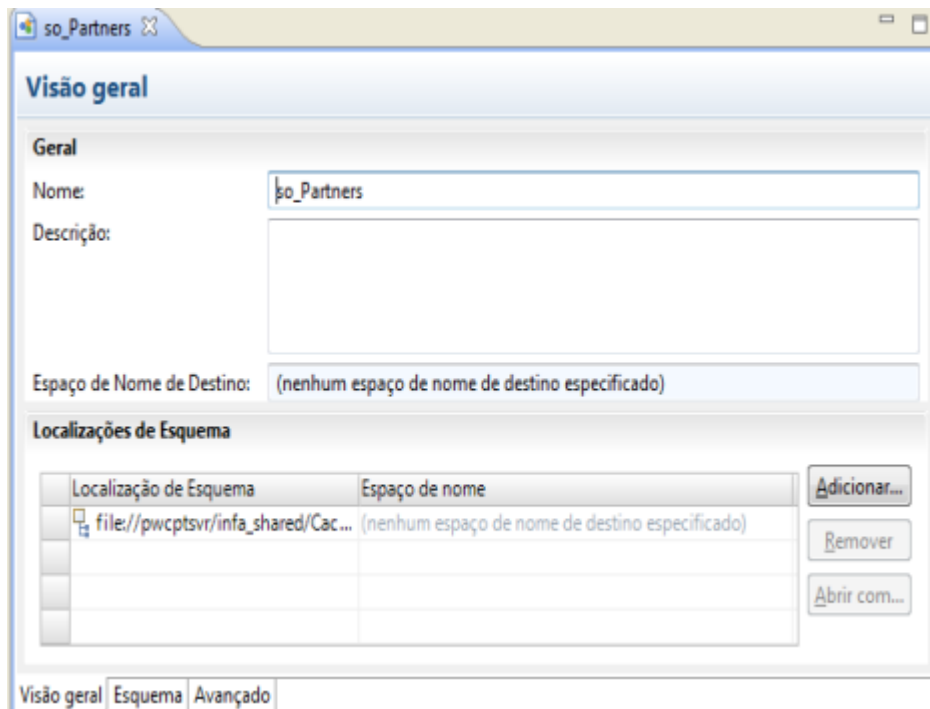
3. Clique em **Adicionar** ao lado da área **Tipos de arquivo**.
A caixa de diálogo **Adicionar Tipo de Arquivo** é exibida.
4. Insira `.xsd` como o tipo de arquivo e clique em **OK**.
5. Clique em **Adicionar** ao lado da área **Editores associados**.
A caixa de diálogo **Seleção de Editor** é exibida.
6. Selecione um editor na lista de editores ou clique em **Procurar** para selecionar um editor diferente e clique em **OK**.
O editor selecionado aparece na lista **Editores associados**.
7. Como opção, adicione outros editores à lista **Editores associados**.
8. Se você adicionar vários editores, poderá alterar o editor padrão. Selecione um editor e clique em **Padrão**.
9. Clique em **OK**.

Editando um Arquivo de Esquema

É possível editar qualquer arquivo de esquema em um objeto de esquema.

1. Abra um objeto de esquema.
2. Selecione a exibição **Visão Geral**.

A exibição **Visão Geral** do objeto de esquema aparece.



3. Selecione um arquivo de esquema na lista **Localizações de Esquema**.
4. Clique em **Abrir com** e selecione uma das seguintes opções:

Opção	Descrição
Editor do Sistema	O arquivo de esquema é aberto no editor usado pelo seu sistema operacional para arquivos .xsd.
Editor Padrão	O arquivo de esquema é aberto no editor de que você define como padrão na ferramenta Developer. Essa opção será exibida se você definir um editor padrão.
Outro	Selecione o editor no qual abrir o arquivo de esquema.

A ferramenta Developer abre uma cópia temporário do arquivo de esquema.

5. Atualize o arquivo de esquema temporário, salve as alterações e feche o editor.

A ferramenta Developer solicita que você atualize o objeto de esquema.

6. Para atualizar o objeto de esquema, clique em **Atualizar Objeto de Esquema**.

A ferramenta Developer atualiza o arquivo de esquema com as alterações feitas.

CAPÍTULO 9

Interface da Linha de Comando

Este capítulo inclui os seguintes tópicos:

- [Visão Geral da Interface da Linha de Comando, 136](#)
- [CM_console, 136](#)

Visão Geral da Interface da Linha de Comando

Você pode executar um serviço de Transformação de Dados da linha de comando da máquina que hospeda o serviço.

Exporte uma transformação do Processador de Dados como um serviço para o diretório `/ServiceDB` da máquina onde você deseja executar o serviço de Transformação de Dados. Execute o comando `CM_console`.

CM_console

Executa um serviço de Transformação de Dados.

O comando `CM_console` usa a seguinte sintaxe:

```
CM_console <ServiceName>
[< -f | -u | -t >InputDocument]
[ -aServiceParameter=InitialValue]
[ -o<[Path]FileName | FileName>]
[ -r<curr | res | spec=OutputDirectory | guid>]
[ -lUserName -pPassword]
[ -v]
[ -S]
[ -x<f | u | t>InputPortName=InputDocument]
[ -xoOutputPortName=OutputDocument]
[ -e]
```

Nota: Não inclua um espaço entre uma opção e o seu argumento.

A seguinte tabela descreve as opções e argumentos de CM_console:

Opção	Argumento	Descrição
-	ServiceName	Obrigatório. Especifica o nome do serviço.
-f	InputDocument	Opcional. Especifica um caminho e nome de arquivo no sistema de arquivos local. Por padrão, o serviço usará o documento definido na propriedade example_source do componente de inicialização.
-t	InputDocument	Opcional. Especifica uma string entre aspas duplas.
-u	InputDocument	Opcional. Especifica uma URL.
-a	ServiceParameter=InitialValue	Opcional. Especifica um parâmetro de entrada para o serviço. ServiceParameter é o nome de uma variável definida no serviço. InitialValue deve ser um tipo de dados válido para a variável definida. Você pode inserir vários parâmetros de entrada separados por espaços.
-o	FileName [Path]FileName	Opcional. Direciona a saída para Path/FileName. Se você inserir somente o Path/FileName, você deve definir o Caminho com a opção -r. Por padrão, o comando CM_console direciona de saída para a tela.
-r	curr	Opcional. Especifica o diretório do qual você executou o comando CM_console.
-r	res	Opcional. Especifica o subdiretório <i>resultados</i> no diretório que contém o serviço no sistema de arquivos do repositório.
-r	spec=OutputDirectory	Opcional. Especifica um diretório no sistema de arquivos local.
-r	guid	Opcional. Especifica um diretório com um nome exclusivo no diretório CMReports/tmp. Você pode usar o editor de configuração para alterar a localização desse diretório.
-l	UserName	Obrigatório quando você usar a autenticação HTTP. Especifica o nome de usuário para autenticação HTTP. Nota: Essa opção é um L minúsculo.
-p	Senha	Obrigatório quando você usar a autenticação HTTP. Especifica a senha para autenticação HTTP.
-v	-	Opcional. Exibe as informações sobre a versão da Transformação de Dados, a versão da sintaxe da Transformação de Dados, o identificador do pacote de configuração, a licença e outras informações.
-S	-	Obrigatório se o componente de inicialização do serviço for um streamer. Você também deve usar a opção -f para definir o arquivo de entrada.
-xf	InputPortName=InputDocument	Opcional. InputPortName especifica o nome de uma AdditionalInputPort definida no serviço. InputDocument especifica um caminho e nome de arquivo no sistema de arquivos local. Você pode inserir várias portas de entrada separadas por espaços.

Opção	Argumento	Descrição
-xt	InputPortName=InputDocument	Opcional. InputPortName especifica o nome de uma AdditionalInputPort definida no serviço. InputDocument especifica uma string entre aspas duplas. Você pode inserir várias portas de entrada separadas por espaços.
-xu	InputPortName=InputDocument	Opcional. InputPortName especifica o nome de uma AdditionalInputPort definida no serviço. InputDocument especifica uma URL. Você pode inserir várias portas de entrada separadas por espaços.
-xo	OutputPortName=OutputDocument	Opcional. OutputPortName especifica o nome de uma AdditionalOutputPort definida no serviço. OutputDocument especifica um caminho e nome de arquivo no sistema de arquivos local. Você pode inserir várias portas de saída separadas por espaços.
-e	-	Opcional. Por padrão, o comando CM_console é encerrado com um código de saída de 1 para êxito e maior do que 1 para erro. Quando você incluir a opção -e, o comando CM_console será encerrado com um código de saída de 0 para êxito e maior do que 1 para erro.

Por exemplo:

```
CM_console XYZparser -fInputFile.txt -aMaxLines=1000 -oResults.xml -rcurr
```

Esse exemplo chama o serviço XYZparser usando `InputFile.txt` como o documento de entrada principal. Ele fornece o valor 1000 para o parâmetro *MaxLines* e grava a saída no arquivo `Results.xml` no diretório do qual você executou o comando `CM_console`.

CAPÍTULO 10

Scripts

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Scripts, 139](#)
- [Componentes de Script, 140](#)
- [Propriedades de um Componente de Script, 142](#)
- [Componentes de Inicialização de Script, 144](#)
- [Origens de Exemplo, 144](#)
- [Editor do IntelliScript, 146](#)
- [Validar um script, 147](#)
- [Scripts de Amostra, 147](#)

Visão Geral de Scripts

Um Script realiza transformações complexas em dados de entrada e grava os dados de saída. Crie um Script na guia **Objetos** da transformação de Processador de Dados. Use o editor do IntelliScript para exibir um Script, adicionar e configurar componentes e definir o componente de inicialização para um Script.

Use um Script para ler um ou mais documentos em qualquer formato, como HL7, PDF, XML ou Word. É possível gravar um ou mais documentos em qualquer formato. É possível gravar a saída de um Script no sistema de arquivos local ou retornar a saída através de portas de saída da transformação de Processador de Dados.

Um Script é formado por componentes que definem documentos de entrada e saída, a lógica comercial, variáveis que armazenam dados temporariamente e definições de configuração. Os componentes são dispostos em uma árvore hierárquica. Quando a transformação executa um Script, ela inicia o processamento no componente que você define como o componente de inicialização.

Ao configurar um Script, você define origens de exemplo que contêm dados de amostra para cada porta de entrada. Quando a transformação é executada na exibição **Visualizador de Dados**, ela faz a leitura dos exemplos de documentos de origem. Quando a transformação é executada em um mapeamento, ela faz a leitura dos documentos que recebe através de suas portas de entrada.

A transformação de Processador de Dados que contém o Script deve fazer referência a um esquema para cada documento XML lido ou gravado pelo Script.

Componentes de Script

Um componente de Script é uma linha ou um grupo de linhas em um Script que define documentos de entrada e saída, a lógica comercial, variáveis que armazenam dados temporariamente e definições de configuração. Os componentes de um Script aparecem em uma árvore hierárquica. Alguns componentes aparecem no nível global do Script, enquanto outros são exibidos como componentes filho.

O nível global do Script contém componentes de inicialização, variáveis e outros componentes, como portas de entrada adicionais e Transformadores. Um componente no nível global deve ter um nome.

Um componente pode ter propriedades que controlam seu comportamento. As propriedades de um componente aparecem aninhadas dentro dele. Uma propriedade pode aparecer em uma linha ou como uma hierarquia de propriedades. Você pode configurar as propriedades de alguns componentes de forma a substituir as configurações padrão que se aplicam à transformação de Processador de Dados.

Alguns componentes, como Analisadores ou Mapeadores, podem conter outros componentes, como Transformadores ou ações **RunParser**. Também existe a opção de configurar a propriedade **name** de um componente filho.

Tipos de Componentes

O contexto do Script determina os tipos de componentes que você pode adicionar.

Por exemplo, as âncoras devem aparecer aninhadas em Analisadores, Mapeadores ou Serializadores. Além disso, portas de entrada e de saída adicionais podem aparecer somente no nível global do Script.

A seguinte tabela descreve os tipos de componentes que você pode adicionar a um Script:

Tipo de Componente	Descrição
Ação	Obtém dados de um recipiente de dados e realiza uma operação neles. Por exemplo, a ação RunParser executa um Analisador.
Âncora	Identifica uma seção do documento de entrada.
Processador de documentos	Realiza uma transformação complexa em um documento de entrada. Por exemplo, o processador de documentos PdfToTxt_4 converte um documento PDF em texto simples.
Formato	Define o formato dos documentos que um Analisador processa.
Localizador	Isola uma única ocorrência de um recipiente de dados de ocorrência múltipla.
Mapeador	Lê e grava documentos XML. Pode ser definido como o componente de inicialização.
Notificação	Grava uma mensagem na saída padrão ou em um log. Por exemplo, a notificação XsdValidationError indica que o documento de entrada não é válido em relação ao esquema que o define.
Analisador	Lê documentos e grava documentos em qualquer formato. Pode ser definido como o componente de inicialização.
Porta de Script	Define um documento de entrada ou saída.
Serializador	Lê documentos XML e grava documentos em qualquer formato. Pode ser definido como o componente de inicialização.

Tipo de Componente	Descrição
Streamer	Divide arquivos de entrada extensos em blocos e os transmite para um Analisador, um Mapeador ou um Serializador. Pode ser definido como o componente de inicialização.
Transformador	Transforma uma string de entrada em uma string de saída. Pode ser definido como o componente de inicialização.
Validador	Determina se os dados de entrada estão em conformidade com uma definição de dados específica.
Variável	Armazena dados que o Script recebe através de um parâmetro de serviço ou armazena dados do componente no Script.

Nomes de Componentes

O nome de um componente o identifica no Script, na exibição **Eventos do Processador de Dados** e no log.

Quando a transformação de Processador de Dados realiza as instruções em um componente, esse componente gera um evento que aparece no log e na exibição **Eventos do Processador de Dados**. Um componente que aparece no nível global do Script deve ter um nome. Um componente que aparece como filho de outro componente pode ter um nome que você configura com a propriedade **name**.

O nome de um componente deve começar com uma letra, deve conter somente caracteres em inglês (A-Z, a-z), numerais (0-9) e sublinhados (_) e não deve conter mais de 127 caracteres.

Adicionando um Componente Global

Defina um componente globalmente quando precisar usá-lo em dois ou mais locais no Script ou quando o componente só puder aparecer no nível global.

1. No final do nível global do Script, clique duas vezes no sinal de reticências à esquerda (...).
Uma caixa de texto é exibida.
2. Insira o nome do componente e pressione **ENTER**.
3. Clique duas vezes no sinal de reticências à direita.
Uma caixa de listagem é exibida.
4. Clique na seta para baixo e selecione o tipo de componente que você deseja adicionar.
O componente global aparece no Script.
5. Defina as propriedades do componente, se houver.

Adicionando um Componente Local

Defina um componente localmente quando você planeja utilizá-lo em apenas uma localização no Script ou quando esse componente só pode aparecer como um componente filho.

1. No lugar do Script onde você deseja inserir um componente, clique duas vezes no sinal de reticências.
Uma caixa de listagem é exibida.
2. Clique na seta para baixo à direita da caixa de listagem.
Uma lista de componentes disponíveis é exibida, incluindo componentes globais.

3. Selecione um componente.
O componente é exibido no Script.
4. Defina as propriedades do componente, se houver.

Propriedades de um Componente de Script

As propriedades de um componente de Script definem a funcionalidade desse componente. Um componente pode ter uma ou mais propriedades. Essas propriedades aparecem aninhadas dentro do componente. Todos os componentes do mesmo tipo têm as mesmas propriedades.

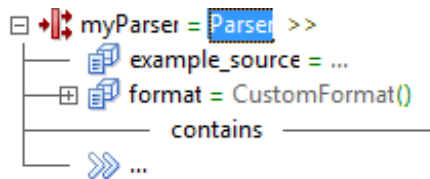
Por exemplo, a propriedade **example_source** de um Analisador define o texto de exemplo que o Analisador usa quando você executa a transformação na exibição **Visualizador de Dados**.

Propriedades Simples

As propriedades simples de um componente são as propriedades que o editor do IntelliScript sempre exibem.

A maioria dos usuários precisa modificar somente as propriedades simples.

A seguinte figura mostra as propriedades simples de um componente de **Analisador**:



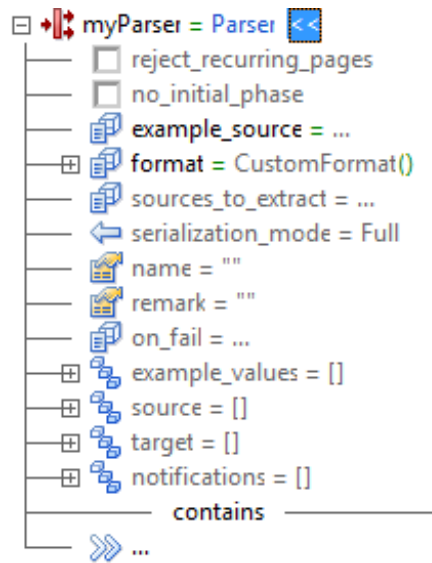
Propriedades Avançadas

As propriedades avançadas de um componente normalmente são definidas como valores padrão que os usuários normalmente não alteram.

O editor do IntelliScript normalmente exibe apenas as propriedades avançadas que você define para um valor não padrão.

Para mostrar as propriedades que não são exibidas, clique na seta direita dupla na primeira linha.

A figura a seguir mostra todas as propriedades de um componente **Analisador**:



Valores de Propriedade de Componente

Defina os valores das propriedades de um componente.

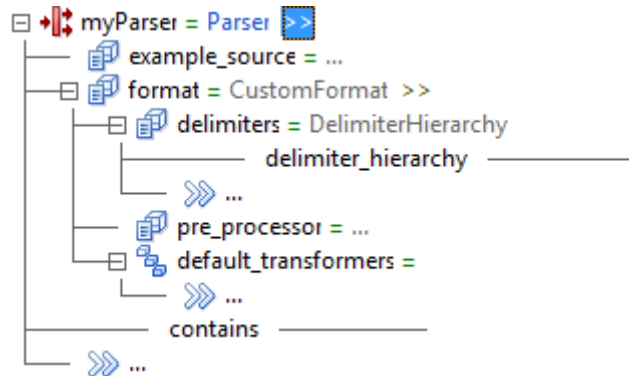
Quando o valor de uma propriedade for booleano, ele aparecerá como uma caixa de seleção à esquerda do nome da propriedade. Por exemplo, a propriedade **opcional** de um componente de **Conteúdo** é Booleano.

Quando o valor for uma string, ele aparecerá à direita do nome da propriedade entre aspas duplas. Os valores válidos são strings válidas de caracteres alfanuméricos, símbolos ou caracteres de controle, mas não inclui caracteres nulos. Para inserir um caractere que não existe no teclado em um campo de texto, pressione **CTRL+A** e, em seguida, digite o código decimal de três dígitos do caractere. Por exemplo, digite **CTRL+A 010** para um avanço de linha ou **CTRL+A 255** para a letra Islandesa "thorn" (þ). Por exemplo, o valor da propriedade da **expressão** de um componente **CalculateValue** é uma string.

Quando o valor for uma seleção, ele aparecerá à direita do nome da propriedade. Quando você for editar o valor, uma caixa de listagem será exibida. Por exemplo, a propriedade **val_type** de um componente de **Variável** é uma seleção.

Quando o valor for uma árvore hierárquica de propriedades, ele aparecerá à direita e abaixo do nome da propriedade. Por exemplo, quando você definir a propriedade de **formato** de um componente **Analisador** como CustomFormat, uma árvore de propriedades adicionais será exibida.

A figura a seguir mostra a propriedade de **formato** de um componente **Analisador**, que é exibida como uma árvore:



Componentes de Inicialização de Script

O componente de inicialização de um Script define o ponto de entrada em que a transformação de Processador de Dados começa a processar o Script. O componente de inicialização deve aparecer no nível global do Script.

Você pode definir um Analisador, um Mapeador, um Serializador, um Streamer ou um Transformador como o componente de inicialização.

É possível definir o componente de inicialização na guia **Visão Geral** de uma transformação de Processador de Dados. Quando você usa o editor do IntelliScript para definir o componente de inicialização de um Script, esse componente se torna o componente de inicialização da transformação de Processador de Dados.

Definindo o Componente de Inicialização com o Editor do IntelliScript

Você pode usar o editor do IntelliScript para definir um componente de Script como o componente de inicialização da transformação de Processador de Dados. É necessário definir o componente de inicialização para executar o Script. É necessário definir o componente de inicialização para exibir a origem de exemplo no painel **Entrada** da exibição **Visualizador de Dados**.

1. Abra um Script no editor do IntelliScript.
2. Clique com o botão direito em um componente que apareça no nível global do Script e selecione **Definir como Componente de Inicialização**.

Origens de Exemplo

Uma origem de exemplo é um documento que contém dados de entrada de amostra para o Script processar em tempo de design. Você configura uma origem de exemplo para cada Analisador, Mapeador, Serializador

ou porta de entrada adicional. A origem de exemplo contém o mesmo tipo de dados que a transformação de Processador de Dados recebe de uma porta de entrada.

Por padrão, a exibição **Visualizador de Dados** mostra a origem de exemplo definida para o componente de inicialização. Você também pode exibir a origem de exemplo de qualquer outro componente que define uma origem de exemplo. Quando um Script é executado a partir da exibição **Visualizador de Dados**, a transformação de Processador de Dados lê os exemplos de documentos de origem.

É possível configurar os seguintes tipos de exemplos de documentos de origem:

- LocalFile. Um arquivo no sistema de arquivos local.
- Text. Uma string definida em código fixo no Script.
- URL. Um arquivo na rede local ou na Internet.

Nota: Quando você executa um Script em um mapeamento e um documento de entrada está ausente, a transformação usa a origem de exemplo. Se nenhuma origem de exemplo estiver configurada e não houver um documento de entrada, a transformação de Processador de Dados será interrompida e gerará um erro fatal.

Exemplo de uma Origem de Exemplo

O seguinte texto de amostra ilustra uma parte de um arquivo local que você pode usar para uma origem de exemplo ao analisar documentos HL7:

```
MSH|^~\&|ADT1|MCM|FINGER|MCM|198808181126|SECURITY|ADT^A01|MSG00001|P|2.3.1
EVN|A01|198808181123
PID|1||PATID1234^5^M11^ADT1^MR^MCM~123456789^^^USSSA^SS||
SMITH^WILLIAM^A^III||19610615|M||C|1200 N ELM STREET^^JERUSALEM^TN^999999?
1020|GL|(999)999?1212|(999)999?3333||S||PATID12345001^2^M10^ADT1^AN^A|
123456789|987654^NC
NK1|1|SMITH^OREGANO^K|WI^WIFE|||NK^NEXT OF KIN
PV1|1|I|2000^2012^01|||004777^CASTRO^FRANK^J.|||SUR|||ADM|AO
```

Exemplo de Realce de Origem

O painel **Entrada** da exibição do **Visualizador de Dados** realça partes do exemplo de documento de origem.

A exibição do **Visualizador de Dados** usa diferentes cores para realçar âncoras de conteúdo do exemplo de origem, âncoras de marcador que definem onde a transformação encontra conteúdo e repetição de grupos de âncoras.

Definindo uma Origem de Exemplo no Editor do IntelliScript

Ao executar um Script na exibição **Visualizador de Dados**, você deve ter uma origem de exemplo para a entrada principal e para cada porta de entrada adicional. Defina a origem de exemplo no editor do IntelliScript. Você também pode selecionar a origem de exemplo ao criar um Script na transformação de Processador de Dados.

1. Selecione o componente para o qual você deseja definir uma origem de exemplo e expanda-o para mostrar suas propriedades.
2. Ao lado da propriedade **example_source**, clique duas vezes no sinal de reticências.

3. Selecione um formato de entrada. A tabela a seguir descreve as opções de formato de entrada:

Opção	Descrição
LocalFile	A propriedade file_name aparece abaixo da propriedade example_source . Clique duas vezes no sinal de reticências e navegue até um arquivo no sistema de arquivos local.
Text	A propriedade quote aparece abaixo da propriedade example_source . Insira uma string.
URL	A propriedade stable_url aparece abaixo da propriedade example_source . Insira uma string.

Exibindo uma Origem de Exemplo

Você pode visualizar a origem de exemplo de um Analisador, um Mapeador, um Serializador ou uma porta de entrada adicional no painel **Entrada** da exibição **Visualizador de Dados**.

1. Abra um Script no editor do IntelliScript.
2. Defina um dos componentes do Script como o componente de inicialização.
3. No editor do IntelliScript, selecione o componente com a origem de exemplo que você deseja exibir.
4. Na exibição **Visualizador de Dados**, clique em **Sincronizar com Editor**.

Editor do IntelliScript

O editor do IntelliScript é uma ferramenta gráfica que você utiliza para editar Scripts. Use o editor do IntelliScript para adicionar componentes ao Script, configurar as propriedades do componente e definir o componente de inicialização.

Quando você abre um Script, o editor do IntelliScript é exibido na área no editor localizada no centro da interface da ferramenta Developer. Por padrão, o editor do IntelliScript exibe Scripts no Modo Inteli, que mostra o Script em um formato de árvore hierárquica ampliável, ou no Modo de Script, que mostra o Script como texto. É possível exibir ou editar um Script no Modo Inteli. Algumas propriedades avançadas ficam ocultas por padrão, mas você pode mostrá-las clicando em uma seta gráfica dupla na primeira linha do componente.

É possível inserir somente componentes que são válidos para o contexto. Você pode arrastar um componente para movê-lo ou pode recortá-lo e colá-lo com **CTRL+C** e **CTRL+V**. É possível selecionar vários componentes com cliques do mouse e com as teclas **CTRL** e **SHIFT**.

Quando você usa o editor do IntelliScript, as seguintes exibições mostram informações relevantes:

- Visualizador de Dados, painel Entrada. Mostra a origem de exemplo para o componente de inicialização ou o componente selecionado no editor do IntelliScript.
- Visualizador de Dados, painel Saída. Mostra a saída quando você executa a transformação de Processador de Dados a partir da exibição **Visualizador de Dados**.
- Eventos do Processador de Dados. Mostra os eventos que ocorrem quando você executa uma transformação de Processador de Dados. Use a exibição **Eventos do Processador de Dados** para solução de problemas.
- Ajuda de Scripts do Processador de Dados. Mostra a documentação relevante para o componente ou propriedade atualmente selecionado no editor do IntelliScript.

- Origem Hexadecimal do Processador de Dados. Mostra o exemplo de documento de origem em formato hexadecimal. Use a exibição **Origem Hexadecimal do Processador de Dados** para localizar caracteres não imprimíveis, como tabulações.

Para exibir a origem de um Script, clique com o botão direito no editor do IntelliScript e selecione **Modo de Script**. Para retornar ao Modo Intelli, clique com o botão direito no editor do IntelliScript e selecione **Modo Intelli**.

Validar um script

Ao criar um Script, você pode validá-lo antes de o executar. Quando você valida um Script, a Developer tool verifica se há falhas que possam impedir que um componente processe dados da maneira esperada.

Para validar um Script, na exibição **Estrutura de Tópicos** da Developer tool, selecione o Script, clique nele com o botão direito do mouse e selecione **Validar**. Se houver erros, a exibição **Log de Validação** aparecerá mostrando erros ou avisos sobre falhas que o processador de validação descobriu.

Se você clicar duas vezes em um erro na exibição **Log de Validação**, a linha relevante no Script será destacada no editor do IntelliScript.

Scripts de Amostra

A Informatica fornece Scripts de amostra como exemplos de tarefas que você pode realizar com um Script.

Os Scripts de amostra estão localizados no seguinte subdiretório do diretório de instalação:

```
\DataTransformation\samples\Projects
```

Para exibir, modificar ou copiar um Script de amostra, primeiro é necessário importá-lo.

A seguinte tabela descreve os Scripts de amostra:

Nome do Script	Descrição
Alternativas	Demonstra o processo de ramificação e a âncora de Alternativas .
AppendListItems	Concatena strings em um recipiente de dados de ocorrência múltipla e demonstra a ação AppendListItems .
CalculateValue	Realiza um cálculo numérico complexo e demonstra a ação CalculateValue .
CombineValues	Concatena strings e demonstra as ações CombineValues e DumpValue .
Conteúdo	Demonstra a âncora de Conteúdo e o processo de localização no documento de origem procurando uma string específica, calculando um deslocamento a partir da última âncora e procurando um atributo em um par de nome=valor.
CopyValue	Copia um elemento XML complexo inteiro com a ação Map .
DelimitedSections	Demonstra a âncora DelimitedSections em um Analisador.

Nome do Script	Descrição
DocumentOrder	Demonstra o processo de ramificação e a âncora de Alternativas , com a opção selector definida como DocumentOrder.
Dynamic_And_RepeatingGroup	Reitera sobre as linhas de um documento e demonstra a âncora RepeatingGroup . Lê dados de outra localização no documento com base no conteúdo no escopo atual.
EmbeddedParser	Usa um Analisador secundário incorporado para analisar o conteúdo do Analisador principal e demonstra a âncora EmbeddedParser .
EnsureCondition	Avalia uma expressão Booleana JavaScript para selecionar alternativas e demonstra a ação EnsureCondition .
ManualSerializer	Demonstra um Serializador personalizado.
Marcadores	Demonstra âncoras de Marcador que usam as opções TextSearch, OffsetSearch, TypeSearch e PatternSearch.
Marking_Mode	Demonstra vários métodos de configuração de âncoras de Marcador .
NonMarker	Demonstra um Analisador que usa somente âncoras de Conteúdo e pesquisa o documento de entrada de maneira retroativa.
Pattern	Demonstra a extração de dados que correspondem a uma restrição definida no esquema.
persistent_search	Demonstra a propriedade on_partial_match de um Grupo e a propriedade adjacent de um Marcador .
ResetListVariable	Redefine uma variável de lista usando um targetLocator.
RunSerializer	Demonstra um Analisador que chama um Serializador secundário.
HL7	Converte um arquivo HL7 em XML.
TabDelimited	Converte um arquivo HL7 delimitado por tabulação em XML.
Splitter	Divide um arquivo em dois e demonstra a ação WriteValue .
TransformByParser	Usa um Analisador para transformar o texto específico em alimentação de linha de retorno de carro e demonstrar a ação TransformByParser .
Transformers_Example	Demonstra a âncora de Conteúdo com a propriedade value definida como LearnByExample .

Importando uma Amostra de Script

Importe uma amostra de Script para exibir esse Script ou copiar partes para outro Script.

1. Clique em **Arquivo > Importar**.
A caixa de diálogo **Importar** é exibida.
2. Selecione **Informatica > Importar Serviço de DT** e clique em **Avançar**.
A página **Importar Serviço de DT** é exibida.

3. Ao lado do campo **Arquivo de serviço**, clique no botão **Procurar** e navegue até o arquivo CMW do serviço.
4. Clique em **Concluir**.
A amostra de Script aparece em uma transformação de Processador de Dados.

CAPÍTULO 11

Analísadores

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Analísadores, 150](#)
- [Analísadores Independentes de Plataforma, 150](#)
- [Referência de Componente de Analísador, 151](#)

Visão Geral de Analísadores

Analísadores são componentes de Script que fazem a leitura de documentos de origem em qualquer formato.

A saída de um Analísador é sempre XML. A entrada pode ter qualquer formato, como texto, HTML, Word, PDF ou HL7. A entrada pode ser um documento XML que o Analísador processa como dados de string.

Analísadores Independentes de Plataforma

Scripts Analísadores são executados em sistemas Microsoft Windows e UNIX. A maioria dos recursos de Analísador tem o mesmo bom desempenho de execução em ambas as plataformas.

Porém, há algumas exceções para essa regra. Se você planeja executar um Analísador no Windows e no UNIX, veja algumas dicas que podem ajudar a garantir a independência de plataforma.

Marcadores de Nova Linha

Evite definir âncoras de **Marcador** que pesquisem um caractere de nova linha seguido por um caractere de retorno de carro (`\n\r`). Essa combinação é comumente usada no Windows, mas normalmente não é usada no UNIX.

Em vez disso, configure um **Marcador** com o componente interno **NewlineSearch**, que pesquisa pela sequência `\n\r` e pelo caracteres `\n` ou `\r` sozinho.

Caminhos de Arquivo

Use caminhos de arquivos relativos, em vez de absolutos. Lembre-se de que os caminhos de arquivos no UNIX fazem diferença entre maiúsculas e minúsculas.

Referência de Componente de Analisador

Um componente **Analisador** converte um documento de origem em XML.

Analisador

Um Analisador lê um documento de origem em qualquer formato. É possível adicionar componentes filho para realizar transformações nos dados.

Defina Analisadores no nível global do Script. Defina um Analisador principal como o componente de inicialização. Chame um Analisador secundário com a ação **RunParser**. Para obter mais informações, consulte [“RunParser” na página 320](#).

As propriedades do **Analisador** aparecem acima da linha **contains**. Abaixo da linha, você pode inserir componentes filho, como âncoras e ações.

A seguinte tabela descreve as propriedades do componente **Analisador**:

Propriedade	Descrição
example_source	Define um documento de origem de amostra a ser processado no ambiente de desenvolvimento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Vazio. A ferramenta Developer solicita um documento de origem quando você executa o Analisador.- InputPort. Define uma porta de entrada.- LocalFile. Define um arquivo no sistema de arquivos local.- Text. Define uma string.- URL. Define uma URL. O padrão é vazio. Nota: Se a propriedade sources_to_extract estiver definida, a propriedade example_values será ignorada no ambiente de design.
example_values	Define os valores simulados que outra transformação pode transmitir para o Analisador. Use essa propriedade para criar um Analisador que seja chamado por outro Analisador. Um Analisador usa a propriedade example_values somente quando processa a origem de exemplo. Ele ignora a propriedade ao analisar um documento de origem. Nos componentes ExampleValue aninhados, especifique os recipientes de dados que o Analisador de chamada transmite para esse Analisador e seus valores simulados.
ExampleValue	Define um valor de exemplo na propriedade example_values .
formato	Define o formato do documento de origem. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- BinaryFormat- CustomFormat- HtmlFormat- Rtf Format- TextFormat- XmlFormat O padrão é CustomFormat. Para obter mais informações, consulte “Referência de Componente de Formato” na página 178 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
no_initial_phase	Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais. - Selected. Procura âncoras aninhadas na fase principal. O padrão é Desmarcado.
notificações	Define uma lista de componentes NotificationHandler que o Analisador executa em notificações disparadas por componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
reject_recurring_pages	Determina o número de vezes que o Analisador analisa a mesma página. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Seleccionado. O Analisador analisa uma página somente uma vez. - Limpo. O Analisador analisa uma página sempre que seguir um link para essa página. Use reject_recurring_pages quando um site tiver muitos links para a mesma página. Nota: A ação ResetVisitedPages redefine a lista de histórico e permite que um Analisador processe uma página novamente, mesmo quando a propriedade reject_recurring_pages estiver selecionada.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
serialization_mode	Define como o Script processa partes da origem de exemplo que o Analisador não gera em XML quando você cria um serializador de um Analisador. Para obter mais informações, consulte “Controlando como o Comando Criar Serializador Funciona” na página 333 . É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Completo. Faz com que o comando Criar Serializador copie o texto não XML para a configuração do serializador. - Estrutura de Tópicos. Faz com que o comando Criar Serializador copie apenas os delimitadores do texto não XML para a configuração do serializador. Quando a opção Estrutura de Tópicos está selecionada, você pode definir a propriedade use_markers.
origem	Define uma sequência de recipientes de dados para a entrada no Analisador. Cada recipiente de dados é identificado por uma das seguintes propriedades: <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração acessa uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. Em um Analisador secundário, defina Analisador > origem > Localizador > data_holder como o recipiente de dados definido no AdditionalInputPort > data_holder associado. Para obter mais informações, consulte “Propriedade de Origem” na página 366 .

Propriedade	Descrição
sources_to_extract	<p>Define uma lista embutida em código de documentos de origem que o Analisador processa. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - DocList. Define uma lista de componentes LocalFile, Text e URL. - Vazio. O Analisador processa example_source. - FileSearch. Define uma pasta no sistema de arquivos local e um filtro de nomes de arquivo. - InputPort. Define uma porta de entrada. Não use esta opção. - LocalFile. Define um arquivo no sistema de arquivos local. - Text. Define uma string. - URL. Define uma URL. <p>O padrão é vazio.</p> <p>Nota: Use a propriedade sources_to_extract apenas no ambiente de design.</p>
destino	<p>Define uma sequência de recipientes de dados para saída do Analisador. Se o recipiente de dados ainda não existir, o Analisador o criará. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração cria uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade destino quando a saída do Analisador for usada por outro componente. Para obter mais informações, consulte "Propriedade de Destino" na página 369.</p>
use_markers	<p>Determina se o comando Criar Serializador copia o conteúdo das âncoras Marcador, mas somente os delimitadores de outros textos não XML. use_markers é uma opção na propriedade serialization_mode quando a opção Estrutura de Tópicos está selecionada. O padrão é selecionado.</p>

CAPÍTULO 12

Portas de Script

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Portas de Script, 154](#)
- [Referência de Componentes de Porta de Script, 154](#)

Visão Geral de Portas de Script

Uma porta de Script especifica a entrada ou a saída de um Script, como um documento de origem ou um documento de saída.

Por exemplo, em um componente **Analisador**, os valores das propriedades **example_source** e **sources_to_extract** são portas de entrada.

Em alguns componentes, as portas de Script são definidas implicitamente. Por exemplo, o arquivo de saída padrão de um Analisador é o arquivo `output.xml`. Não é necessário definir uma porta de saída que faça referência ao arquivo `output.xml`.

Por padrão, cada Script tem uma porta de entrada e uma porta de saída. É possível configurar portas de entrada e de saída adicionais. Quando você cria portas de entrada ou de saída adicionais em um Script, a ferramenta Developer acrescenta portas adicionais na transformação de Processador de Dados.

Referência de Componentes de Porta de Script

Um componente de porta de Script especifica a entrada ou a saída de uma transformação, como um documento de origem ou um documento de saída.

AdditionalInputPort

A porta **AdditionalInputPort** define uma porta de entrada adicional.

A seguinte tabela descreve as propriedades da porta **AdditionalInputPort**:

Propriedade	Descrição
code_page	Determina a codificação de entrada para a porta. Quando nenhum valor está definido, AdditionalInputPort usa a codificação de entrada definida nas configurações da transformação de Processador de Dados. O padrão é em branco.
data_holder	Define um recipiente de dados no qual a porta armazena o conteúdo do documento de entrada. Use o mesmo recipiente de dados na propriedade Analizador > origem > Localizador > data_holder do Analisador, Mapeador ou Serializador secundário associado.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
encode_as_xml	Determina se caracteres especiais são convertidos em entidades XML. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Caracteres especiais são convertidos em entidades XML.- Limpo. Caracteres especiais não são convertidos. A propriedade encode_as_xml é um filho da propriedade input_encoding quando definida como PortEncoding. O padrão é limpo.
example_source	Define a localização de uma origem a ser processada durante testes. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- LocalFile. Define um arquivo no computador local.- Text. Define uma string.- URL. Define a URL de uma página da Web. CUIDADO: Não defina um processador de documento na propriedade AdditionalInputPort > example_source > pre_processor . Defina-o em AdditionalInputPort > pre_processor .
input_encoding	Define a codificação da entrada.
PortEncoding	Define configurações personalizadas para a entrada adicional. A propriedade PortEncoding tem as seguintes opções: <ul style="list-style-type: none">- code_page- encode_as_xml
pre_processor	Define o nome de um processador de documentos a ser aplicado à entrada antes do processador de documentos definido em RunParser > pre_processor . Para obter mais informações, consulte "Referência de Componente do Processador de Documentos" na página 163 . CUIDADO: Não defina um processador de documento na propriedade AdditionalInputPort > example_source > pre_processor . Defina-o em AdditionalInputPort > pre_processor .

Defina a porta **AdditionalInputPort** no nível global do Script e atribua um nome a ela.

Exemplo de AdditionalInputPort

Suponha que você tenha dois arquivos de texto:

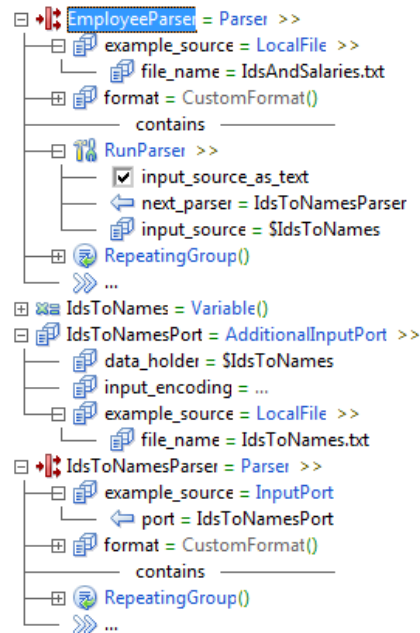
- `IdsAndSalaries.txt` é uma tabela de IDs e salários de funcionários.
- `IdsAndNames.txt` é uma tabela de IDs e nomes de funcionários.

Você deseja analisar esses arquivos em conjunto, gerando um arquivo de saída XML que contém os nomes e os salários dos funcionários. É possível configurar a transformação da seguinte maneira:

- O Analisador principal, denominado `EmployeeParser`, processa `IdsAndSalaries.txt`.

- O Analisador principal ativa um Analisador secundário, denominado `IdsToNamesParser`, que processa `IdsAndNames.txt` e armazena o resultado em uma tabela XML.
- O Analisador principal usa um `LookupTransformer` para converter os IDs em nomes. A tabela de pesquisa é a saída do Analisador secundário.

A seguinte figura mostra um exemplo de um Script com um Analisador secundário que menciona uma `AdditionalInputPort` para recuperar o arquivo `IdsAndNames.txt`:



AdditionalOutputPort

A porta **AdditionalOutputPort** define uma porta de saída adicional. Use esse componente para definir a saída em várias localizações ou em vários documentos.

A tabela a seguir descreve as propriedades da porta **AdditionalOutputPort**:

Propriedade	Descrição
<code>add_BOM_prefix</code>	Adiciona um prefixo de marca de ordem de byte (BOM) à saída. O tipo de prefixo BOM é determinado pela codificação de saída definida na propriedade output_encoding . O padrão é desmarcado.
<code>code_page</code>	Define o atributo de codificação da saída adicional. Se essa propriedade não estiver definida, a saída adicional será gerada com a codificação definida nas configurações da transformação do Processador de Dados. A propriedade code_page será um filho da propriedade output_encoding quando estiver definida como PortEncoding . O padrão é desmarcado.
<code>desativado</code>	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.

Propriedade	Descrição
encode_as_xml	Determina se os caracteres especiais são convertidos em entidades XML. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. Os caracteres especiais são convertidos em entidades XML. - Desmarcado. Os caracteres especiais não são convertidos. A propriedade encode_as_xml será um filho da propriedade output_encoding quando estiver definida como PortEncoding. O padrão é desmarcado.
file_extension	Define a extensão de arquivo para o arquivo de saída adicional no ambiente de design. O nome do arquivo é o nome atribuído ao componente AdditionalOutputPort . Essa configuração não tem efeito no ambiente de produção. O padrão é .xml.
other_properties	Define as propriedades de codificação quando é definido como XmlHeader . É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - XmlHeader. Define o cabeçalho de XML. A propriedade XmlHeader tem as seguintes opções: <ul style="list-style-type: none"> - add_BOM_prefix - process_instruction - process_instruction_string - root_element - xml_version - XSLT_stylesheet_name - Desmarcado. As propriedades de saída são determinadas pelas configurações de transformação do Processador de Dados. O padrão é desmarcado.
output_encoding	Define as propriedades de codificação quando é definido como PortEncoding . É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - PortEncoding. A saída adicional tem configurações personalizadas para code_page e encode_as_xml. - Desmarcado. As configurações de transformação do Processador de Dados controla a codificação de saída e a conversão de entidades XML. O padrão é desmarcado.
PortEncoding	Define as configurações personalizadas para a saída adicional. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - code_page - encode_as_xml
process_instruction	Define uma instrução de processamento no arquivo XML de saída. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Nenhum. Não grava uma instrução de processamento na saída XML. - UseOutputCodePage. Retorna a página de código definida na propriedade output_encoding. - FreeEncodingString. Retorna a string definida na propriedade process_instruction_string. O padrão é UseOutputCodePage.
process_instruction_string	Define uma instrução de processamento definida pelo usuário. A propriedade process_instruction_string tem efeito somente quando a propriedade process_instruction está definida como FreeEncodingString.
root_element	Define o nome do elemento raiz que está ajustado ao redor da saída inteira.
xml_version	Define o atributo da versão da instrução de processamento. O padrão é 1.0.
XSLT_stylesheet_name	Define uma folha de estilos XSLT que é gravada na instrução de processamento.

Definindo uma Porta de Saída Adicional

1. No nível global do Script, insira um componente **AdditionalOutputPort** e atribua um nome a ele.
2. Aninhada no componente de inicialização da transformação de Processador de Dados, insira uma ação **WriteValue**, defina a propriedade **output** como **OutputPort** e defina **port** como o nome da porta de saída adicional.
3. Nas configurações da transformação de Processador de Dados, selecione **Controle de Saída** e marque **Desabilitar Saída Automática**.

Nome do Arquivo de Saída Adicional

Quando você executa a transformação na ferramenta Developer, o sistema define um nome de arquivo para a saída adicional e armazena o arquivo na pasta de resultados do projeto. Por exemplo, se a porta se chamar **MyOutputPort**, o nome de arquivo poderá ser `output_MyOutputPort.xml`.

Para determinar o nome de arquivo:

1. Clique em **Executar > Executar**.
2. Clique em **Detalhes** para exibir a tabela de **Portas de E/S**.

A tabela exibe o nome de cada **AdditionalOutputPort** e o arquivo de saída.

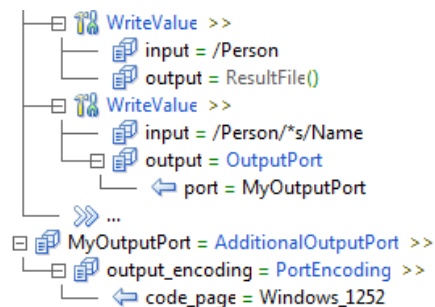
Quando você implantar a transformação como um serviço, um aplicativo que executa o serviço poderá passar a localização de saída adicional como um parâmetro. Por exemplo, a localização pode ser um buffer.

Exemplo de AdditionalOutputPort

Um Analisador gera a seguinte estrutura XML:

```
<Person gender="M">
  <Name>
    <First>Ron</First>
    <Last>Lehrer</Last>
  </Name>
  <Id>547329876</Id>
  <Age>27</Age>
</Person>
```

A figura a seguir mostra um Analisador que usa duas ações `WriteValue` para gerar saída.



O primeiro `WriteValue` grava o elemento `<Person>` no arquivo de resultados padrão.

```
<Person gender="M">
  <Name>
    <First>Ron</First>
    <Last>Lehrer</Last>
  </Name>
  <Id>547329876</Id>
  <Age>27</Age>
</Person>
```

O segundo `WriteValue` faz referência a um `AdditionalOutputPort` para gravar o elemento `<Name>` aninhado a outro arquivo.

```
<Name>
  <First>Ron</First>
  <Last>Lehrer</Last>
</Name>
```

DocList

A porta **DocList** define uma lista dos seguintes tipos de portas de entrada:

- LocalFile
- Texto
- URL

A seguinte tabela descreve as propriedades da porta **DocList**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
pre_processor	Define o nome do pré-processador a ser aplicado aos arquivos de entrada. Para obter mais informações, consulte "Referência de Componente do Processador de Documentos" na página 163 .

FileSearch

A porta **FileSearch** define arquivos de entrada em um computador na rede local. Use a porta **FileSearch** na propriedade `sources_to_extract` de um **Analísador**.

A seguinte tabela descreve as propriedades da porta **FileSearch**:

Propriedade	Descrição
diretório	Define uma pasta que contém os arquivos de entrada.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
pre_processor	Define o nome de um processador de documentos a ser aplicado aos arquivos de entrada. Para obter mais informações, consulte "Referência de Componente do Processador de Documentos" na página 163 .
recursivo	Determina se os arquivos de entrada podem ocorrer em subpastas da pasta especificada. O padrão é limpo.
wildcard	Define um critério para a filtragem dos arquivos na pasta especificada. Use * como um caractere curinga. Por exemplo, *.txt localiza todos os arquivos TXT. O padrão é *.*.

InputPort

A porta **InputPort** define uma porta de entrada nomeadas que é definida com o componente **AdditionalInputPort**.

A tabela a seguir descreve as propriedades da porta **InputPort**:

Propriedade	Descrição
entrada	Define o nome do componente AdditionalInputPort que define a entrada.

LocalFile

A porta **LocalFile** define um arquivo na rede local.

A seguinte tabela descreve as propriedades da porta **LocalFile**:

Propriedade	Descrição
file_name	Define o caminho e o nome de arquivo na rede local.
pre_processor	Define o nome de um processador de documentos a ser aplicado ao arquivo. Para obter mais informações, consulte "Referência de Componente do Processador de Documentos" na página 163 .
simulated_url	Define uma URL a ser atribuída ao arquivo. Essa propriedade faz com que o Analisador trate o arquivo como se ele estivesse localizado em um servidor da Web. Se o arquivo contiver links relativos, o Analisador resolverá esses links relativos à URL. A parte de nome de host da URL não faz distinção entre maiúsculas e minúsculas.

OutputPort

A porta **OutputPort** define uma porta de saída nomeada que é definida com o componente **AdditionalOutputPort**. Você pode usar uma porta **OutputPort** em uma ação **WriteValue**.

A tabela a seguir descreve as propriedades da porta **OutputPort**:

Propriedade	Descrição
porta	Determina o nome da AdditionalOutputPort .

Texto

A porta de **Texto** define uma string de texto que é usada como entrada de uma transformação.

A seguinte tabela descreve as propriedades da porta de **Texto**:

Propriedade	Descrição
pre_processor	Define o nome de um processador de documentos a ser aplicado à string. Para obter mais informações, consulte “Referência de Componente do Processador de Documentos” na página 163 .
quote	Define uma string de texto.
simulated_url	Define uma URL a ser atribuída à string. Esta propriedade faz com que o Analisador trate a string como se fosse um arquivo localizado em um servidor da Web. Se a string contiver links relativos, o Analisador resolverá esses links em relação à URL.
size	Define um tamanho estático para o buffer de texto. Use a propriedade size com origens binárias. O padrão é -1, o que significa que o buffer é dimensionado dinamicamente.

URL

A porta **URL** define o URL de um documento que está disponível em um servidor da Web.

A seguinte tabela descreve as propriedades da porta URL:

Propriedade	Descrição
post_data	Define os dados que a transformação publica no URL.
pre_processor	Define o nome de um processador de documento a ser aplicado aos arquivos.
tentativas	Define o número de tentativas que o Analisador executa antes de relatar uma falha. O padrão é 0.
seconds_to_wait	Define o número de segundos de espera entre as tentativas. O padrão é 60.
stable_url	Define um endereço de URL que contém um documento de entrada.

Nota: Este componente é fornecido para compatibilidade com projetos criados em versões anteriores do Data Transformation. Ele está sendo gradualmente removido do sistema Data Transformation. Não o use ao desenvolver transformações. Observe que o Data Transformation não consegue processar uma URL HTTPS em um ambiente Linux.

CAPÍTULO 13

Processadores de Documentos

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Processadores de Documentos, 162](#)
- [Definindo um Processador de Documentos, 162](#)
- [Referência de Componente do Processador de Documentos, 163](#)
- [Esquema XML TextML, 172](#)
- [Editor de Configuração de Tabela PdfToTxt_4, 173](#)

Visão Geral de Processadores de Documentos

Processadores de documentos são componentes que convertem o formato de um documento completo em outro formato para processamento.

É possível usar um processador de documentos como um pré-processador que converte o formato de um documento de origem antes de uma transformação. Por exemplo, se o documento de origem de um analisador estiver no formato PDF, você poderá aplicar o processador **PdfToTxt_4**. Isso converte o documento de origem em texto, que é muito mais fácil de analisar do que o formato PDF binário.

Não confunda processadores de documentos com pré-processadores de formato. Para obter mais informações sobre pré-processadores de formato, consulte [“Visão Geral de Formatos” na página 177](#).

Definindo um Processador de Documentos

É possível pré-processar o documento de origem com qualquer processador de documentos.

1. Atribua a propriedade **example_source** da transformação. O valor de **example_source** é uma porta de entrada, como **LocalFile** ou **Text**.
2. Atribua a propriedade **pre_processor** da porta de entrada.

O Script aplica o processador que você define em **example_source** a todas as origens nas quais a transformação é executada.

Nota: Você também pode definir um pré-processador na propriedade **sources_to_extract** de um Analisador. O processador definido nesse ponto somente se aplica aos documentos de origem que você define em **sources_to_extract** e não aos outros documentos processados pelo Analisador.

Exibição da Saída do Processador de Documento

Se você atribuir um processador de documentos ao exemplo de origem, o painel de origem da exibição do **Visualizador de Dados** mostrará a saída do processador.

Referência de Componente do Processador de Documentos

Os processadores de documentos convertem um documento completo de um formato para outro antes que ele seja processado por um Analisador, um Mapeador ou um Serializador.

AsnToXml

O processador de documentos **AsnToXml** converte um arquivo binário de arquivo ASN.1 em XML.

A tabela a seguir descreve as propriedades do processador de documentos **AsnToXml**:

Propriedade	Descrição
asn_file	Define um arquivo de especificação ASN.1.
cabeçalho	Define um cabeçalho para excluir a partir do XML. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- NewlineSearch. O cabeçalho é uma nova linha.- OffsetSearch. O cabeçalho é definido pelo número de caracteres do início do arquivo.- PatternSearch. O cabeçalho é definido por uma expressão regular.- TextSearch. O cabeçalho é definido por uma string explícita ou uma string que você recupera dinamicamente do documento de origem.
no_constraints	Determina se o arquivo ASN é processado com restrições. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- verdadeiro. O arquivo ASN é processado sem restrições.- falso. O arquivo ASN é processado com restrições. O padrão é Falso.
pdu_type	Define o tipo de PDU. Use essa propriedade para explicar um ambiguidade.
process_first_message	Determina se o arquivo CDR inteiro foi processado. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- verdadeiro. Somente o primeiro registro é processado.- falso. O arquivo CDR inteiro é processado. O padrão é Falso.
separador	Define o texto a ser ignorado entre os registros. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- NewlineSearch. O separador é uma nova linha.- OffsetSearch. O separador é definido pelo número de caracteres a partir do fim do registro anterior.- PatternSearch. O separador é definido por uma expressão regular.- TextSearch. O separador é definido por uma string explícita ou uma string que você recupera dinamicamente do documento de origem.

ExcelToDataXml

O processador de documentos **ExcelToDataXml** converte documentos do Microsoft Excel em XML.

A tabela a seguir descreve as propriedades do processador de documentos **ExcelToDataXml**:

Propriedade	Descrição
ativado	Determina o conteúdo da saída. A propriedade ativado tem as seguintes opções: <ul style="list-style-type: none">- Marcado. A saída contém dados brutos e dados formatados.- Desmarcado. A saída contém somente dados formatados. O padrão é marcado.
param1	Determina se os dados brutos serão exibidos na saída do processador de documentos quando forem diferentes dos dados formatados. param1 é denominado Display_raw_data_when_different e tem somente uma propriedade ativada .

O XML contém os dados e os resultados das fórmulas que existiam no documento original do Excel. Ele não preserva as próprias fórmulas, informações de formatação ou macros de código. Se você precisar usar código de macro, use **ExcelToXml** em vez de **ExcelToDataXml**.

A representação XML está em conformidade com um subconjunto do esquema `ExcelToXml.xsd`, que você pode localizar no subdiretório `doc` do diretório de instalação.

A saída do processador será na codificação UTF-8. Se uma transformação receber entrada do processador, você deverá definir a codificação de entrada para UTF-8.

O processador suporta Excel versão 97 e posterior. Ele acessa a entrada diretamente, e não por meio do aplicativo do Excel. Não é necessário instalar o Excel no computador. O processador suporta os formatos XLS e XLSX.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

ExcelToXml

O processador de documentos **ExcelToXml** converte documentos do Microsoft Excel em XML.

A tabela a seguir descreve as propriedades do processador de documentos **ExcelToXml**:

Propriedade	Descrição
ativado	Define o valor de param2 ou param3 .
param1	Define as planilhas da pasta de trabalho do Excel para incluir no XML. Na saída XML, cada planilha é representada por um elemento <code><sheet></code> . param1 é denominado include_sheets e tem a propriedade value .
param2	Determina se o processador de documentos inclui células vazias no XML de saída quando as células são formatadas ou mescladas. param2 é denominado include_empty_cells e tem a propriedade ativada , com as seguintes opções: <ul style="list-style-type: none">- Marcado. A saída inclui células vazias.- Desmarcado. A saída omite células vazias. O padrão é selecionado.

Propriedade	Descrição
param3	Determina se o processador de documentos inclui código de macro do Excel na saída XML. param3 é denominado include_macro_information e tem apenas uma propriedade ativada , com as seguintes opções: - Marcado. O processador de documentos inclui códigos de macro. - Desmarcado. O processador de documentos omite códigos de macro. O padrão é limpo.
param4	Determina se o processador de documentos inclui células vazias não formatadas no XML de saída para células. param4 é denominado include_empty_non_formatted_cells e tem a propriedade ativada , com as seguintes opções: - Marcado. A saída inclui células vazias. - Desmarcado. A saída omite células vazias. O padrão é limpo.
valor	Define uma lista das seguintes opções: - A string "Tudo". A saída inclui todas as planilhas. - Os recipientes de dados que contêm os nomes das planilhas. A saída inclui somente as planilhas nomeadas. Se você listar uma planilha que não existe na pasta de trabalho, o processador vai gerar um elemento <code><sheet></code> contendo uma mensagem de erro. As outras planilhas serão processadas normalmente. O padrão é Tudo.

O XML preserva os dados, fórmulas, formatação e macro de código que existia no documento original do Excel. Se somente os dados forem necessários, use o processador **ExcelToDataXml**, que oferece menor saída e melhor desempenho.

A representação XML está em conformidade com o esquema `ExcelToXml.xsd`, que está no subdiretório `doc` do diretório de instalação.

A saída do processador será na codificação UTF-8. Se uma transformação receber entrada do processador, você deverá definir a codificação de entrada para UTF-8.

O processador suporta Excel versão 97-2003. O processador acessa a entrada diretamente e não por meio do Excel. Não é necessário instalar o Excel no computador.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

ExcelToXml_03_07_10

O processador de documentos **ExcelToXml_03_07_10** converte os seguintes arquivos em XML.

- Arquivos XLSX criados com o Microsoft Excel 2007, 2010 ou 2013
- Arquivos XLS criados com o Microsoft Excel 2003, 2007, 2010 ou 2013

ExpandFrameSet

O processador de documentos **ExpandFrameSet** abre todos os quadros de um documento HTML. Use esse processador de documentos quando o documento de origem de um Analisador for um conjunto de quadros HTML. O Analisador é executado no conteúdo de todos os quadros.

ExternalJavaPreProcessor

O processador de documentos **ExternalJavaPreProcessor** executa um processador de documentos definido pelo usuário que é implementado em Java.

A seguinte tabela descreve as propriedades processador de documentos **ExternalJavaPreProcessor**:

Propriedade	Descrição
jclass	Define o caminho da classe Java.
jmethod	Define o método a ser executado.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

Nota: Esse componente foi preterido. O editor do IntelliScript o exibe apenas para scripts herdados. Não o use em novos Scripts. Em vez disso, crie um processador de documentos Java personalizado. Para obter mais informações, consulte [“Desenvolvendo um Componente Personalizado” na página 432](#).

HIPAAValidator

O processador de documentos **HIPAAValidator** valida mensagens HIPAA e gera confirmações HIPAA. O projeto **HIPAA_Validation** da biblioteca HIPAA usa esse processador.

A tabela a seguir descreve as propriedades do processador de documentos **HIPAAValidator**:

Propriedade	Descrição
param1	A propriedade param1 é denominada validation_params e tem somente uma propriedade valor , que tem as seguintes opções: <ul style="list-style-type: none">- LDNSB- Validador
param2	Define o tipo de validação. A propriedade param2 é denominada types_to_validate e tem somente uma propriedade valor . Os valores válidos são entre 1 e 7.
param3	Define o formato de saída do relatório de erro. A propriedade param3 é denominada report_formats e tem somente uma propriedade valor , que tem as seguintes opções: <ul style="list-style-type: none">- HTML. Use para exibição na ferramenta Developer.- XML. Use para processamento adicional.
param4	Define o tipo de reconhecimento. A propriedade param4 é denominada generate_acknowledgments e tem somente uma propriedade valor , que tem as seguintes opções: <ul style="list-style-type: none">- 277- 824- 997- 999- TA1
valor	Define o valor de param1 , param2 , param3 ou param4 .

Nota: Esse processador de documentos opera nas plataformas Windows e Linux x64. Antes de usar, você deve instalar e configurar o pacote do complemento de validação HIPAA em cada máquina em que **HIPAAValidator** é executado.

PdfFormToXml_1_00

O processador de documentos **PdfFormToXml_1_00** converte formatos PDF em XML. O processador suporta formatos que estão em conformidade com o padrão Adobe AcroForms.

PdfToTxt_3_02

O processador de documentos **PdfToTxt_3_02** converte arquivos PDF em texto.

A seguinte tabela descreve as propriedades processador de documentos **PdfToTxt_3_02**:

Propriedade	Descrição
enabled	Define o valor de param2 ou param4 .
param1	Define uma string ou variável que contém o fator de espaçamento entre palavras. A propriedade param1 se chama WordSpacingFactor e tem apenas uma propriedade, value , que contém a string ou a variável. O padrão é 1.8.
param2	Determina se o documento de saída é otimizado para tabelas. A propriedade param2 se chama OptimizeForTables e tem apenas uma propriedade, enabled , que possui as seguintes opções: <ul style="list-style-type: none">- Selected. O documento de saída é otimizado para tabelas.- Cleared. O documento de saída não é otimizado para tabelas. O padrão é Desmarcado.
param3	Define uma string ou variável que contém a senha. A propriedade param3 se chama Password e tem apenas uma propriedade, value , que contém a string ou a variável.
param4	A propriedade param4 se chama HideNewPageChar e tem apenas uma propriedade, enabled , que possui as seguintes opções: <ul style="list-style-type: none">- Selected. Novos caracteres de página ficam ocultos.- Cleared. Novos caracteres de página não ficam ocultos. O padrão é Desmarcado.
param5	Define uma string ou variável que contém otimizações avançadas. A propriedade param5 se chama AdvancedOptimizations e tem apenas uma propriedade, value , que contém a string ou a variável.
value	Define o valor de param1 , param3 ou param5 .

O pré-processador PdfToTxt pode não dar suporte a determinados PDFs com fontes incorporadas. Se o pré-processador falhar, copie o texto do PDF de entrada no Bloco de Notas para verificar se há fontes incorporadas. Se você não consegue colar o texto ou se ele está corrompido, o PDF provavelmente tem fontes incorporadas.

Nota: Esse componente foi preterido. O editor do IntelliScript o exibe apenas para projetos herdados. Não o use em novos Scripts.

PdfToTxt_4

O processador de documentos **PdfToTxt_4** converte arquivos PDF em texto ou XML.

A tabela a seguir descreve as propriedades do processador de documentos **PdfToTxt_4**:

Propriedade	Descrição
param1	Define o layout de tabela PDF. A propriedade param1 tem somente uma opção: PdfLayout
valor	Define o layout de tabela PDF. Clique duas vezes na propriedade valor para abrir o editor de configuração de tabelas.

O editor de configuração de tabelas personaliza a maneira como as tabelas são lidas. Use-o para corrigir problemas com alinhamento de coluna, quebra automática de linha, espaçamento de linha e estouro de uma célula para outra. Para obter mais informações, consulte [“Editor de Configuração de Tabela PdfToTxt_4” na página 173](#).

O processador de documentos **PdfToTxt_4** gera saída de texto por padrão. Use o editor de configuração de tabelas para selecionar a saída XML. O XML está em conformidade com o esquema `PDF4.xsd`, que você pode localizar no seguinte diretório:

```
<INSTALL_DIR>\DataTransformation\doc
```

Quando você usar o processador de documentos **PdfToTxt_4**, defina a codificação de entrada como UTF-8 para permitir que o Analisador, o Mapeador ou o Serializador leia o documento corretamente.

Nota: O pré-processador PdfToTxt pode não dar suporte a determinados PDFs com fontes incorporadas. Se o pré-processador falhar, copie o texto do PDF de entrada no Bloco de Notas para verificar se há fontes incorporadas. Se você não consegue colar o texto ou se ele está corrompido, o PDF provavelmente tem fontes incorporadas.

PowerpointToTextML

O processador de documentos **PowerpointToTextML** converte apresentações do Microsoft PowerPoint (PPT) em esquema XML TextML. Para obter mais informações, consulte [“Esquema XML TextML” na página 172](#).

Esse componente suporta PowerPoint versão 97 e superior. Ele acessa a entrada diretamente, e não por meio do PowerPoint. Não é necessário instalar o PowerPoint no computador.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

ProcessByTransformers

O processador de documentos **ProcessByTransformers** executa um transformador ou uma sequência de transformadores em todo o documento. Uma transformação pode, então, ser executada na saída do transformadores.

Defina a lista de transformadores sob a linha de **transformadores**.

ProcessorPipeline

O processador de documentos **ProcessorPipeline** define uma sequência de processadores de documentos para executar em um documento. Use esse componente quando você precisar executar dois ou mais processadores de documentos.

Defina a lista de processadores de documentos sob a linha **pre_processor_list**.

RtfToTextML

O processador de documento **RtfToTextML** converte arquivos RTF no esquema XML de TextML. Para obter mais informações, consulte [“Esquema XML TextML” na página 172](#).

A saída do processador será na codificação UTF-8. Se uma transformação receber entrada do processador, você deverá definir a codificação de entrada para UTF-8.

WordToXml

O processador de documento **WordToXml** converte documentos do Microsoft Word em XML.

A saída do processador será na codificação UTF-8. Se uma transformação receber entrada do processador, você deverá definir a codificação de entrada para UTF-8.

Esse componente oferece suporte ao Word, versão 97 e posterior. Ele acessa sua entrada diretamente, e não por meio do Microsoft Word. Não é necessário instalar o Word no computador.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

XmlToDocument_372

O processador de documentos **XmlToDocument_372** converte dados XML em formatos de documento, como PDF ou Excel. Você pode usá-lo como pós-processador para converter a saída de um Analisador ou um Mapeador em vários tipos de documentos.

Esse componente usa o complemento Eclipse **Business Intelligence and Reporting Tool** (BIRT) para gerar os documentos de saída. No BIRT, é necessário configurar um relatório que converte o XML no formato de documento desejado. O processador **XmlToDocument_372** executa o relatório.

É possível fazer download do BIRT no local mencionado no arquivo `readme_BIRT.txt` em <diretorio de instalação do mecanismo do Data Transformation>/`readme_Birt.txt`.

Para obter mais informações sobre o BIRT, consulte <http://www.eclipse.org/birt>.

Nota: Para usar o BIRT versão 4.5, use o pré-processador **XmlToDocument_45** em vez do pré-processador **XmlToDocument_372**.

A seguinte tabela descreve as propriedades do processador de documentos **XmlToDocument_372**:

Propriedade	Descrição
param1	O caminho e o nome do arquivo *.rptdesign do BIRT. A propriedade param1 é nomeada report_file e contém a propriedade value , que inclui o caminho e o nome de arquivo.
param2	O formato do documento de saída. A propriedade param2 é nomeada output_format e contém a propriedade value , que possui as seguintes opções: <ul style="list-style-type: none">- pdf. Documento PDF.- doc. Documento do Microsoft Word.- xls. Pasta de trabalho do Microsoft Excel.- ppt. Apresentação da Microsoft PowerPoint.- html. Página da Web HTML.- ps. Documento PostScript. O padrão é PDF.
param3	Uma variável que contém a localização do arquivo *.rptdesign. A propriedade param3 é nomeada report_location e contém a propriedade value , que aponta para a variável. O padrão é \$VarServiceInfo/*s/ServiceLocation.
value	Contém o valor de param1 , param2 ou param3 .

Nota: O processador **XmlToDocument** foi preterido a partir da versão 9.5.1. O editor do IntelliScript ainda exibe o pré-processador **XmlToDocument** em Scripts existentes, mas não pode mais adicioná-lo a novos Scripts. Em vez disso, use o pré-processador **XmlToDocument_372**.

XmlToDocument_45

O processador de documentos **XmlToDocument_45** converte dados XML em formatos de documento, como PDF ou Excel. Você pode usá-lo como pós-processador para converter a saída de um Analisador ou um Mapeador em vários tipos de documentos.

Esse componente usa o complemento Eclipse versão 4.5 da **Ferramenta Inteligência Comercial e Relatórios** (BIRT) para gerar os documentos de saída. No BIRT, configure um relatório que converte o XML no formato de documento desejado. O processador **XmlToDocument_45** executa o relatório.

É possível fazer download do BIRT no local mencionado no arquivo `readme_BIRT.txt` em <diretorio de instalação do mecanismo do Data Transformation>/readme_Birt.txt.

Para obter mais informações sobre o BIRT, consulte <http://www.eclipse.org/birt>.

A seguinte tabela descreve as propriedades do processador de documentos **XmlToDocument_45**:

Propriedade	Descrição
param1	O caminho e o nome do arquivo *.rptdesign do BIRT. A propriedade param1 é nomeada report_file e contém a propriedade value , que inclui o caminho e o nome de arquivo.
param2	O formato do documento de saída. A propriedade param2 é nomeada output_format e contém a propriedade value , que possui as seguintes opções: <ul style="list-style-type: none">- pdf. Documento PDF.- doc. Documento do Microsoft Word.- xls. Pasta de trabalho do Microsoft Excel.- ppt. Apresentação da Microsoft PowerPoint.- html. Página da Web HTML.- ps. Documento PostScript. O padrão é PDF.
param3	Uma variável que contém a localização do arquivo *.rptdesign. A propriedade param3 é nomeada report_location e contém a propriedade value , que aponta para a variável. O padrão é \$VarServiceInfo/*s/ServiceLocation.
value	Contém o valor de param1 , param2 ou param3 .

XmlToExcel

O processador de documento **XmlToExcel** converte documentos XML para o formato do Microsoft Excel.

O processador opera em uma representação XML de uma pasta de trabalho do Excel. A representação XML deve estar na codificação UTF-8 e deve corresponder ao esquema `ExcelToXml.xsd`. É possível localizar o esquema no subdiretório `doc` do diretório de instalação. O arquivo de esquema é fornecido para a sua informação. Você pode usar o processador sem adicionar o esquema ao seu projeto.

O processador reverte a operação do **ExcelToXml**. Por exemplo, você pode usar o **ExcelToXml** para converter uma pasta de trabalho do Excel em XML. Você pode alterar alguns dados XML e usar o **XmlToExcel** para converter os dados de volta para uma pasta de trabalho do Excel.

Esse componente oferece suporte ao Excel, versão 97 e posterior. Ele grava sua saída diretamente, e não por meio do Microsoft Excel. Não é necessário instalar o Excel no computador.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

XmlToXlsx

O processador de documentos **XmlToXlsx** converte documentos XML no formato .xlsx do Microsoft Excel. O processador de documentos **XmlToXlsx** pode, opcionalmente, usar um modelo .xlsx para gerar o documento .xlsx.

O processador opera em uma representação XML de uma pasta de trabalho do Excel. A representação XML deve estar na codificação UTF-8 e deve corresponder ao esquema `ExcelToXml_03_07_10.xsd`. É possível localizar o esquema no subdiretório `doc` do diretório de instalação. O arquivo de esquema é fornecido para a sua informação.

O processador reverte a operação do **ExcelToXml_03_07_10**. Use o processador **ExcelToXml_03_07_10** em uma transformação de Processador de Dados para transformar uma pasta de trabalho do Excel em XML. Depois de processar os dados XML, use o processador **XmlToXlsx** para transformar os dados novamente em uma pasta de trabalho do Excel.

Esse componente dá suporte à versão 2007 do Excel e posterior. Ele grava sua saída diretamente, e não por meio do Microsoft Excel. Não é necessário instalar o Excel no computador.

Esse componente é implementado em Java e requer configuração correta do Java Runtime Environment (JRE).

A seguinte tabela descreve as propriedades do processador de documentos **XmlToXlsx**:

Propriedade	Descrição
param1	Uma variável que mantém o nome do arquivo de modelo *.xlsx. A propriedade param1 tem a propriedade template_file , a qual especifica o nome do arquivo de modelo.
param2	Uma variável que mantém a localização do arquivo de modelo *.xlsx. A propriedade param2 tem a propriedade template_location , a qual especifica o caminho do arquivo de modelo.

Nota: As planilhas do Excel que existem exclusivamente no modelo são representadas como no modelo na saída .xlsx. As planilhas do Excel que são definidas no modelo e no XML recebem estilos de célula do modelo e valores de célula do XML.

Para aplicar um estilo de célula de uma célula diferente no modelo, você pode usar o atributo `style_as` no XML como parte do elemento da célula.

Exemplo

Para usar o atributo `style_as` para aplicar um estilo de célula de uma célula na planilha atual, defina o atributo `style_as` igual ao número da célula que contém o estilo. No exemplo a seguir, o estilo do campo A1 na planilha atual se aplica à célula definida pelo contexto de XML, ou seja, linha 3 e célula 2.

```
<row rownumber="3" firstcol="1" lastcol="12" height="405" zeroheight="false" >
  <cell number="2" type="string" style_index="3" font_index="3" style_as="A1">
    <data>Informatica</data>
  </cell>
</row>
```

Para aplicar um estilo de célula de uma célula em uma planilha diferente, defina o atributo `style_as` igual à planilha e ao número de célula que contém o estilo. No exemplo a seguir, o estilo é aplicado do campo A1 na planilha da pasta de trabalho denominada `Resumo`.

```
<row rownumber="3" firstcol="1" lastcol="12" height="405" zeroheight="false" >
  <cell number="2" type="string" style_index="3" font_index="3"
  style_as="Summary:A1">
    <data>Informatica</data>
  </cell>
</row>
```

Esquema XML TextML

Alguns dos processadores de documento convertem documentos em um vocabulário XML chamado TextML. Este é um vocabulário XML simples para salvar o conteúdo do documento sem layout.

O esquema TextML, `textML.xsd`, está disponível na subpasta `\doc` da pasta de instalação.

Este é um documento TextML de amostra.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<document>
  <docinfo>
    <title>TextML Sample</title>
```

```

<author>Tex Tomiller</author>
<company>Acme Gizmos, Inc.</company>
<modified>2004-03-14T14:39:00</modified>
<created>2004-03-12T09:15:00</created>
<last_author>Tex Tomiller</last_author>
<word_count>16</word_count>
<char_count>105</char_count>
<version>2</version>
</docinfo>
<docbody>
  <p>This is a sample of the TextML XML vocabulary.</p>
  <p>TextML saves document content without layout information.<p>
</docbody>
</document>

```

Editor de Configuração de Tabela PdfToTxt_4

O editor de configuração de tabela personaliza a maneira como o processador de documentos **PdfToTxt_4** converte tabelas em documentos PDF.

Use o editor de configuração de tabela quando as configurações padrão do processador de documentos **PdfToTxt_4** não renderizarem corretamente o alinhamento de colunas, a quebra de palavras, o espaçamento entre linhas ou o estouro de uma célula para outra.

Nota: A interface de usuário do editor de configuração de tabela é exibida somente em inglês.

1. Adicione um Analisador, um Mapeador, um Serializador ou um **AdditionalInputPort** ao Script.
2. Na propriedade **example_source**, defina a propriedade **pre_processor** como PdfToTxt_4.
3. Na propriedade **pre_processor**, clique duas vezes na propriedade **value**.

O editor de configuração de tabela é exibido. O painel superior exibe o documento PDF de entrada, enquanto o painel inferior exibe a saída de **PdfToTxt_4**.

Os comandos de edição de tabela aparecem na barra de ferramentas localizada na parte superior da janela. É possível clicar com o botão direito para exibir um menu de edição.

4. Procure uma tabela no documento PDF e clique em **Adicionar Tabela**.
O nome da tabela aparece no campo **Tables** e no campo **Name**.
5. Selecione **Use Regular Expressions**. No campo **Table Start**, insira uma expressão regular que define o canto superior esquerdo da tabela.

Sugestão: Use os títulos das duas primeiras colunas como a expressão regular. Adicione mais títulos de coluna conforme necessário para tornar **Table Start** exclusivo. Separe os títulos com um único caractere de espaço, mesmo que as colunas estejam amplamente separadas.

6. No campo **Table End**, insira uma expressão regular que defina o texto logo após a tabela.

Nota: O valor de **Table End** deve aparecer no corpo do documento, e não em um rodapé de página.

7. Clique em **Process**.

O editor exibe a configuração de tabela detectada por **PdfToTxt_4**. As partes superior e inferior da tabela aparecem como linhas azuis horizontais. As bordas de coluna padrão aparecem como linhas vermelhas verticais.

8. Para editar as bordas de coluna, realize uma ou mais das etapas a seguir:
 - Arraste uma borda de coluna para a direita ou esquerda para alterar sua posição.

- Clique em **Add Column** para adicionar uma coluna.
- Clique em **Remove Column** e selecione uma borda de coluna para excluir uma coluna.

Nota: Se a tabela contiver células mescladas horizontalmente, o **PdfToTxt_4** poderá truncar as entradas.

- Examine a janela de saída para confirmar se a tabela foi convertida corretamente. Em caso negativo, corrija as definições da tabela.
- Repita as etapas de 1 a 9 para cada tabela no documento PDF.
- Clique em **OK** para retornar à ferramenta Developer.

Uma string XML que define a configuração da tabela é exibida na propriedade **value** do processador de documentos **PdfToTxt_4**.

Opções do Editor

A tabela a seguir descreve os controles e campos do editor de configuração de tabelas **PdfToTxt_4**:

Controle ou Campo	Descrição
Ampliar Zoom	Faz o PDF ser exibido em tamanho maior.
Reduzir Zoom	Faz o PDF ser exibido em tamanho menor.
Ajustar Largura	Exibe o documento PDF de acordo com a largura da janela.
Página Anterior	Vai para a página anterior.
Próxima Página	Vai para a próxima página.
Encontrar	Pesquisa uma string no PDF.
Adicionar Tabela	Adiciona uma tabela à configuração.
Rem. Tabela	Remove uma tabela da configuração.
Adicionar Coluna	Adiciona uma borda de coluna à tabela atual.
Rem. Coluna	Exclui a borda da coluna selecionada no momento.
Processar	Aplica as definições da tabela atual. Clique em Processar depois de cada tabela e ação relacionada à coluna para aplicar essa ação.
Tabelas	Uma lista de tabelas definidas no PDF de entrada. Você pode selecionar uma tabela clicando nela.
Nome	Nome da tabela selecionada no momento.
Início da Tabela	Uma expressão definindo o canto superior esquerdo da tabela.
Fim da Tabela	Uma expressão definindo o primeiro texto após a tabela.
Cabeçalho de Página	Uma expressão definindo o fim do cabeçalho da página. Use essa opção para excluir o cabeçalho do processamento da tabela.
Rodapé da Página	Uma expressão definindo o fim do rodapé da página. Use essa opção para excluir o rodapé do processamento da tabela.

Controle ou Campo	Descrição
Usar Expressões Regulares	Se selecionado, o processador interpretará o Início da Tabela , Fim da Tabela , Cabeçalho da Página e Rodapé da Página como expressões regulares e pesquisará por texto correspondente. Se não for selecionado, o processador interpretará esses campos como texto literal.
Recalcular no Tempo de Execução	Se você selecionar essa opção, PdfToTxt_4 ignorará as configurações de tabela que você especificou usando o editor de configuração de tabelas. Esse recurso é útil se as tabelas em um PDF forem simples suficientes para o PdfToTxt_4 processar sem configuração especial. Por exemplo, suponha que um demonstrativo financeiro em PDF simples contenha uma tabela com colunas que podem variar um pouco de mês para mês. Selecione a opção Recalcular no Tempo de Execução para que o PdfToTxt_4 ajuste as larguras das colunas no tempo de execução.
Recalcular Agora	Se você tiver alterado a definição da tabela, por exemplo, alterando as bordas da coluna ou adicionando um Cabeçalho de Página ou Rodapé de Página , clique em Recalcular Agora para atualizar a definição da tabela.
Página	Número da página de PDF que está exibida atualmente.
Gerar Saída como XML	Gera a saída de PdfToTxt_4 como XML em vez de texto.
Delimitador	Insira um caractere para usar como separador de coluna na saída de texto. O padrão é uma barra vertical ().
OK	Clique para salvar a configuração da tabela e retornar para a ferramenta Developer.
Cancelar	Clique para retornar para a ferramenta Developer sem salvar a configuração da tabela.
Auxílio de Navegação de Tabela	O auxílio de navegação de tabela exibe o número de vezes que uma tabela foi encontrada no documento PDF. Um exemplo de auxílio de navegação é Tabela `Tabela 1' encontrada 2 vezes. As setas ao lado dessas informações permitem que você avance e volte entre as instâncias da mesma estrutura de tabela.

Exemplo de Conversão de PDF

Este exemplo ilustra o procedimento de configuração da tabela **PdfToTxt_4** usando um exemplo de projeto de Analisador e um exemplo de documento PDF.

A entrada do processador é uma pequeno relatório financeiro em formato PDF. O relatório contém alguns textos e duas tabelas. Use o editor de configuração de tabelas para garantir que o processador converta as tabelas corretamente em texto.

Configurando a Primeira Tabela

1. Configure um Analisador e atribua o documento PDF como **example_source**. Clique duas vezes na propriedade **valor** para abrir o editor de configuração de tabelas.
2. Na exibição de PDF, navegue para a primeira tabela.
3. Configure o Início da Tabela = GID RMS ID (os cabeçalhos das primeiras duas colunas da tabela). Observe que a expressão faz distinção entre maiúsculas e minúsculas.
4. Defina o Fim da Tabela = Encaminhar transações de câmbio (o primeiro texto após a tabela). O editor exibirá a configuração da tabela.

5. Se necessário, ajuste a definição da tabela e as colunas. Você pode arrastar, adicionar ou remover as bordas da coluna.

Configurando a Segunda Tabela

A segunda tabela se estende por várias páginas.

1. Clique em **Adicionar Tabela**.
O sistema exibe a Tabela 2 nos campo **Tabelas e Nome**.
2. Defina **Início da Tabela** = **Ações Ticker Negociadas**.
3. Defina **Fim da Tabela** = **Conclusão** (o primeiro o texto do corpo depois da tabela).
4. Clique em **Processar** para configurar a tabela.
5. Ajuste as bordas direitas de **Ações Negociadas** e as colunas **Moeda**.
6. Execute as etapas a seguir para eliminar o cabeçalho e o rodapé da página do documento de saída:
 - a. Defina **Cabeçalho da Página** = **Lucro/Prejuízo**.
 - b. Defina **Rodapé da Página** = **Página [1-9]**.
 - c. Clique em **Processar**.

CAPÍTULO 14

Formatos

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Formatos, 177](#)
- [Propriedades do Formato Padrão, 178](#)
- [Referência de Componente de Formato, 178](#)
- [Referência de Componente de Delimitadores, 184](#)
- [Referência de Componentes de Pré-processador de Formato, 189](#)

Visão Geral de Formatos

A propriedade `formato` de um Analisador define o formato dos documentos a serem processados pela transformação. O valor da propriedade é um dos seguintes componentes de formato:

```
BinaryFormat  
CustomFormat  
HtmlFormat  
RtfFormat  
TextFormat  
XmlFormat
```

O formato tem propriedades próprias, que também definem melhor como o Analisador interpreta e processa a entrada.

A seguinte tabela descreve os subcomponentes que você pode aninhar em um formato:

Subcomponente	Descrição
Delimitador	Define uma hierarquia de caracteres ou strings que organiza as informações no documento, como noval linhas e tabulações.
Pré-processador de formato	Limpa a origem antes que o Analisador comece a procurar as âncoras.
Transformador padrão	Executa operações pré-definidas na saída de cada âncora.

Propriedades do Formato Padrão

A seguinte tabela descreve as propriedades padrão dos componentes de formatação:

Propriedade	Descrição
default_Transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo.
delimitadores	<p>Define a estrutura de informações do documento. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. <p>Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184.</p>
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	<p>Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. <p>O padrão é em branco.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componente de Formato

Os componentes de formato definem o formato dos documentos de entrada. Defina os componentes de formato na propriedade **formato** de um **Analisador**.

BinaryFormat

O formato **BinaryFormat** processa arquivos binários e arquivos de texto que você deseja tratar como um buffer de bytes binários.

A tabela a seguir descreve as propriedades do formato **BinaryFormat**:

Propriedade	Descrição
default_transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo. O padrão é vazio.
delimitadores	Define a estrutura de informações do documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184 . O padrão é Posicional.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. O padrão é vazio.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

CustomFormat

O formato **CustomFormat** é um formato definido pelo usuário para o processamento de qualquer tipo de documento de origem.

A tabela a seguir descreve as propriedades do formato **CustomFormat**:

Propriedade	Descrição
default_transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo. O padrão é vazio.
delimitadores	Define a estrutura de informações do documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184 . O padrão é DelimiterHierarchy.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. O padrão é vazio.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo

Um documento de origem tem a seguinte estrutura:

```
Ron      Lehrer && 547329876:27
Evelyn   Kern   &&    9875424: 53
```

Cada linha do documento é um registro que contém um nome da pessoa, número de ID e idade. Os campos estão separados pelos símbolos **&&** e **:**. Os campos contêm vários caracteres de espaço em locais aleatórios.

Uma maneira de analisar esse documento é usando **CustomFormat**. Na propriedade **delimitadores** do formato, atribua um **DelimiterHierarchy** que contém o símbolos:

```
newline
&&
:
```

Na propriedade **default_transformers**, atribua o **HtmlProcessor**, que remove os espaços extras da saída.

HtmlFormat

O formato **HtmlFormat** define o formato dos arquivos HTML.

A tabela a seguir descreve as propriedades do formato **HtmlFormat**:

Propriedade	Descrição
default_transformers	<p>Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo.</p> <p>O padrão é a seguinte lista de Transformadores:</p> <ul style="list-style-type: none">- RemoveTags. Remove as marcas de HTML.- HtmlEntitiesToASCII. Converte entidades HTML em equivalentes ASCII.- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço.- RemoveMarginSpace. Remove espaço à esquerda e à direita.
delimitadores	<p>Define a estrutura de informações do documento. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. <p>Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184.</p> <p>O padrão é SGML.</p>
nome	<p>Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos. Use a propriedade name para identificar o componente que causou o evento.</p>
pre_processor	<p>Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. <p>O padrão é HtmlProcessor.</p>
comentário	<p>Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.</p>

RtfFormat

O formato **RtfFormat** define o formato dos arquivos RTF.

A seguinte tabela descreve as propriedades do formato **RtfFormat**:

Propriedade	Descrição
default_Transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo. O Padrão é a seguinte lista de Transformadores: <ul style="list-style-type: none">- RtfToASCII. Remove palavras de controle RTF da saída.- RemoveRtfFormatting. Remove as instruções de formatação RTF do texto.- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço.- RemoveMarginSpace. Remove espaço à esquerda e à direita.
delimitadores	Define a estrutura de informações do documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184 . O padrão é RTF.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. O padrão é RtfProcessor.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

TextFormat

O formato **TextFormat** define o formato dos arquivos de texto.

Use esse formato em combinação com um processador de documento para processar outros tipos de documentos. Por exemplo, você pode usá-lo com o processador de documento **PdfToTxt_4** para processar documentos PDF.

A seguinte tabela descreve as propriedades do formato **TextFormat**:

Propriedade	Descrição
default_transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo. O Padrão é a seguinte lista de Transformadores: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço.- RemoveMarginSpace. Remove espaço à esquerda e à direita.
delimitadores	Define a estrutura de informações do documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184 . O padrão é DelimiterHierarchy.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. O padrão é vazio.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

XmlFormat

O formato **XmlFormat** define o formato dos arquivos XML.

O Analisador trata o documento de entrada XML como texto comum. Você pode definir delimitadores, âncoras e outros componentes da mesma forma que faria para um documento de texto regular.

A seguinte tabela descreve as propriedades do formato **XmlFormat**:

Propriedade	Descrição
default_transformers	Define uma lista de Transformadores que o Analisador aplica à saída de cada âncora de conteúdo. O Padrão é a seguinte lista de Transformadores: <ul style="list-style-type: none">- RemoveTags. Remove as marcas XML da saída.- HtmlEntitiesToASCII. Converte entidades XML em seus equivalentes ASCII.- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço.- RemoveMarginSpace. Remove espaço à esquerda e à direita.
delimitadores	Define a estrutura de informações do documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- CommaDelimited. Os campos de dados são separados por vírgulas.- DelimiterHierarchy. Os campos de dados são separados ou rodeados por caracteres de texto.- HL7. Os campos de dados são separados conforme definido no padrão HL7.- Posicional. Os campos de dados são definidos pelo número de caracteres entre eles.- PostScript. Os campos de dados são definidos de acordo com o formato PostScript.- RTF. Os campos de dados são definidos de acordo com o formato RTF.- SGML. Os campos de dados são definidos de acordo com o formato SGML.- SpaceDelimited. Os campos de dados são separados por espaços.- TabDelimited. Os campos de dados são separados por marcas de tabulação. Para obter mais informações, consulte "Referência de Componente de Delimitadores" na página 184 . O padrão é SGML.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pre_processor	Define um pré-processador de formato que processa a entrada depois de qualquer processador de documentos que você tenha definido para a propriedade pre_processor do example_source . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- HtmlProcessor. Converte todas as combinações de tabulação, espaço ou nova linha para um único caractere de espaço. Ele não é restrito a documentos HTML.- RtfProcessor. Normaliza arquivos RTF. O padrão é HtmlProcessor.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componente de Delimitadores

Um componente de delimitadores define uma hierarquia de caracteres ou strings que organiza as informações em um documento, como noval linhas, tabulações, vírgulas ou barras verticais. Você também pode usar um padrão de caractere curinga para definir os delimitadores.

O conceito do delimitador é aplicável tanto para documentos estruturados rigidamente que usam caracteres delimitadores pré-definidos para separar os campos de dados, quanto para texto estruturados de modo flexível ou documentos HTML delimitados por novas linhas ou marcação sintática. O conceito do delimitador também engloba dados estruturados por posição, onde os campos estão localizados em deslocamentos fixos uns dos outros.

O Analisador usa os delimitadores para determinar os critérios de pesquisa das âncoras de **Conteúdo** configuradas com a opção `LearnByExample`.

Por exemplo, suponha que você configure um formato com o componente de delimitadores `TabDelimited`. Isso define uma hierarquia usando os seguintes caracteres como delimitadores:

```
Newline
Tab
```

Você pode definir uma âncora de **Conteúdo** localizada dois caracteres de tabulação após a âncora de **Marcador** precedente no exemplo de origem, como essa:

```
MARKER<tab>abc<tab>CONTENT
```

Quando um Analisador processa um documento de origem, ele procura o **Conteúdo** duas guias depois do **Marcador**.

Em um segundo exemplo, você pode definir uma âncora de **Conteúdo** localizada três novas linhas e uma tabulação após a âncora de **Marcador** no exemplo de origem.

```
MARKER
abc<tab>de
fghi<tab>jkl<tab>mnop
pqrst<tab>CONTENT
```

Dentro das linhas intermediárias, as guias não são contadas porque as novas linhas são mais altas na hierarquia.

Muitos dos componentes de delimitadores, como `TabDelimited` ou `CommaDelimited`, exibem uma hierarquia pré-definida de delimitadores, que você pode editar conforme necessário.

O componente `DelimiterHierarchy` não tem uma hierarquia pré-definida. Você pode inserir todos os delimitadores que precisar.

CommaDelimited

O componente de delimitadores **CommaDelimited** define a seguinte hierarquia de delimitador:

```
Newline
Comma
```

Use **CommaDelimited** quando cada linha de um arquivo de texto contiver um registro e cada registro contiver campos de dados separados por vírgulas.

Você pode adicionar mais delimitadores ou editar a hierarquia pré-definida. Use o mesmo procedimento que você usa para editar o componente **DelimiterHierarchy**.

Exemplo

No documento de origem, uma âncora de **Conteúdo** está duas linhas depois de uma âncora de **Marcador**. Na terceira linha, há três vírgulas, além de qualquer outro texto, antes da âncora de **Conteúdo**:

```
MARKER
abcdef, ghij
abc, def,ghi,CONTENT
```

Se você atribuir o componente **CommaDelimited**, o Analisador aprenderá pelo exemplo de origem que a âncora de **Conteúdo** sempre segue o **Marcador** com duas novas linhas e três vírgulas. Em outro documento de origem, o Analisador encontrará com êxito a seguinte âncora de **Conteúdo**:

```
MARKER
    xyz, uvw, rst
,, ,CONTENT
```

Delimitador

O subcomponente **Delimitador** define um caractere delimitador ou uma string que separa âncoras. Você pode adicionar subcomponentes de **Delimitador** dentro de uma hierarquia de delimitador.

A tabela a seguir descreve as propriedades do subcomponente de **Delimitador**:

Propriedade	Descrição
pesquisar	Define o delimitador. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- NewlineSearch. O delimitador é uma nova linha.- PatternSearch. O delimitador é definido por uma expressão regular.- TextSearch. O delimitador é uma string explícita ou uma string que você recupera dinamicamente do documento de origem. Para obter mais informações, consulte "Referência de Componentes de Pesquisador" na página 245 .

Exemplo

O componente `TabDelimited` contém dois subcomponentes de `Delimitador`. O primeiro usa `NewlineSearch` para definir o caractere de nova linha como um delimitador. O segundo usa `TextSearch` para definir o caractere de tabulação como um delimitador. A tabulação é graficamente representada como um caractere «.

O componente `SpaceDelimited` contém dois subcomponentes de `Delimitador`. O primeiro é idêntico ao do `TabDelimited`. O segundo usa um `PatternSearch` para definir qualquer string de um ou mais espaços como um delimitador. A expressão regular `[]+` significa "um ou mais caracteres de espaço". Observe o espaço entre colchetes.

DelimiterHierarchy

O componente de delimitadores **DelimiterHierarchy** permite que você defina uma hierarquia de delimitador personalizada.

Sob **DelimiterHierarchy**, é possível aninhar qualquer número de componentes de **Delimitador** ou **EnclosingDelimiters**.

Exemplo

No exemplo de documento de origem, suponha que as âncoras são separadas por vírgulas e rodeadas por colchetes, como essa:

```
MARKER,, [CONTENT]
```

Você pode definir um **DelimiterHierarchy** que contém:

```
comma //definido como um componente Delimitador  
[]    //definido como um componente EnclosingDelimiters
```

Neste exemplo, o Analisador aprende que a âncora de **Conteúdo** segue o **Marcador** com duas vírgulas e é rodeado por colchetes. Em outro documento de origem, o Analisador encontrará a seguinte âncora de **Conteúdo**:

```
MARKER,abc,def[CONTENT]
```

Exemplo Online

Para ver um exemplo online, consulte `samples\Projects\EDI\EDI.cmw`. O exemplo usa **DelimiterHierarchy** para definir a nova linha e caracteres de asterisco (*) como delimitadores em um documento de origem EDI.

EnclosingDelimiters

O subcomponente **EnclosingDelimiters** define um par caracteres de delimitador ou uma string que separa âncoras. Você pode adicionar subcomponentes de **EnclosingDelimiters** em uma hierarquia de delimitador.

Você pode usar esse componente para definir delimitadores de chaves ({}), que são colocados ao redor de blocos de códigos de programa C.

A tabela a seguir descreve as propriedades do subcomponente de **EnclosingDelimiters**:

Propriedade	Descrição
abertura	Define o delimitador de abertura.
fechamento	Define o delimitador de fechamento.
escape_sequence	Define um prefixo que faz com que o Analisador ignore uma instância de delimitador de abertura ou de fechamento no documento de origem.

HL7

O componente de delimitadores **HL7** define a seguinte hierarquia de delimitadores para analisar mensagens HL7:

```
newline
vertical bar (|)
caret (^) or tab
```

Você pode adicionar mais delimitadores ou editar a hierarquia pré-definida. O procedimento é o mesmo para o componente **DelimiterHierarchy**.

O padrão de mensagens HL7 permite a uma mensagem definir os seus próprios delimitadores. Você pode analisar a declaração do delimitador de uma mensagem HL7 e criar uma definição de delimitador dinâmica da seguinte maneira:

1. Use âncoras de **Conteúdo** para recuperar os caracteres do delimitador do cabeçalho da mensagem HL7. Armazene os caracteres em variáveis.
2. Adicione componentes do **Delimitador** no componente **HL7**.
3. Para cada componente do **Delimitador**, atribua um **TextSearch**.
4. Sob o componente **TextSearch**, atribua uma das variáveis para a propriedade de **texto**.

Posicional

O componente de delimitadores **Posicionais** faz com que o Analisador localize âncoras de conteúdo por meio da contagem dos caracteres a partir do início do escopo de pesquisa. Para obter mais informações sobre o escopo de pesquisa, consulte [“Visão Geral de Âncoras” na página 204](#).

Exemplo

No exemplo de documento de origem, suponha que uma âncora de **Conteúdo** siga uma âncora de **Marcador** por cinco caracteres, possivelmente incluindo espaços, tabulações e assim por diante.

```
MARKERab cdCONTENTefg
```

Se você atribuir o componente **Posicional**, o Analisador aprenderá pelo exemplo de origem que a âncora de **Conteúdo** sempre segue o **Marcador** com cinco caracteres e tem sete caracteres de comprimento. Em outro documento de origem, o Analisador encontrará com êxito a seguinte âncora de **Conteúdo**:

```
MARKERd<tab>cbaCONTENTzy,xwv
```

Usando a Análise Posicional Junto com os Delimitadores

Não é possível adicionar delimitadores ao componente .

Às vezes você pode desejar definir um Analisador que use os delimitadores para localizar algumas âncoras e use uma definição de posição para outras âncoras. Para fazer isso, selecione um dos outros delimitadores componentes. Não use . Para definir a localização de uma âncora posicionamente, você pode atribuir a opção **OffsetSearch** nas propriedades da âncora.

PostScript

O componente de delimitadores **PostScript** define uma hierarquia de delimitadores para analisar documentos Adobe PostScript:

Não é possível editar a hierarquia de delimitador do componente **PostScript**.

RTF

O componente de delimitadores **RTF** define uma hierarquia de delimitadores analisando documentos RTF:

Não é possível editar a hierarquia de delimitador do componente **RTF**.

SGML

O componente de delimitadores **SGML** define uma hierarquia de delimitadores para analisar documentos SGML, HTML e XML.

Não é possível editar a hierarquia de delimitadores do componente **SGML**.

SpaceDelimited

O componente de delimitadores **SpaceDelimited** define a seguinte hierarquia de delimitadores:

```
Newline
String of one or more space characters
```

SpaceDelimited é usado quando cada linha de um arquivo de texto contém um registro e cada registro contém campos de dados separados por espaços.

Você pode adicionar mais delimitadores ou editar a hierarquia pré-definida. O procedimento é o mesmo para o componente **DelimiterHierarchy** .

Exemplo

No exemplo de documento de origem, suponha que uma âncora de **Conteúdo** siga uma âncora de **Marcador** por duas linhas. Na terceira linha, há dois caracteres de espaço e uma string que contém vários espaços antes da âncora de **Conteúdo**, como por exemplo:

```
MARKER
abcdef
abc def ghi          CONTENT
```

Se você atribuir o componente **SpaceDelimited**, o Analisador aprenderá pelo exemplo de origem que a âncora de **Conteúdo** sempre segue o **Marcador** com duas linhas e três strings de espaços. Em outro documento de origem, o Analisador encontrará com êxito a seguinte âncora de **Conteúdo**:

```
MARKER
      xyz
ghi    def abc CONTENT
```

TabDelimited

O componente de delimitadores **TabDelimited** define a seguinte hierarquia de delimitadores:

```
Newline
Tab
```

TabDelimited é usado quando cada linha de um arquivo de texto contém um registro e cada registro contém campos de dados separados por tabulações.

Você pode adicionar mais delimitadores ou editar a hierarquia pré-definida. O procedimento é o mesmo para o componente **DelimiterHierarchy**.

Exemplo

No exemplo de documento de origem, suponha que uma âncora de **Conteúdo** siga uma âncora de **Marcador** por duas linhas. Na terceira linha, há três caracteres de guia e mais algum texto antes da âncora de **Conteúdo**, como por exemplo:

```
MARKER
abcdef
abc<tab> de,f<tab>ghi<tab>CONTENT
```

Se você atribuir o componente **TabDelimited**, o Analisador aprenderá pelo exemplo de origem que a âncora de **Conteúdo** sempre segue o **Marcador** com duas linhas e três guias. Em outro documento de origem, o Analisador encontrará com êxito a seguinte âncora de **Conteúdo**:

```
MARKER
      xyz
<tab><tab><tab>CONTENT
```

Referência de Componentes de Pré-processador de Formato

A seguinte lista descreve as diferenças entre pré-processadores de formato e processadores de documentos:

- Você pode atribuir um processador de documentos à propriedade **pre_processor** de uma porta de entrada, localizada na propriedade **example_source** ou **sources_to_extract** de um Analisador. Um pré-processador de formato só pode ser atribuído à propriedade **pre_processor** de um formato.
- Um processador de documentos é executado no documento de origem antes de executar qualquer outra operação.
- Um pré-processador de formato é executado no texto antes de procurar âncoras. A saída do pré-processador de formato não é exibida.

Para obter mais informações, consulte [“Visão Geral de Processadores de Documentos” na página 162](#).

HtmlProcessor

O pré-processador de formato **HtmlProcessor**, que também funciona como um transformador, normaliza um espaço em branco de acordo com as convenções de HTML. Ele reduz qualquer combinação de tabulações, quebras de linha e caracteres de espaço em um único caractere de espaço.

Use esse pré-processador para normalizar um espaço em branco em qualquer tipo de texto. Ele não é restrito a documentos HTML.

RtfProcessor

O pré-processador de formato **RtfProcessor** normaliza o código dos arquivos RTF.

CAPÍTULO 15

Recipientes de Dados

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Recipientes de Dados, 191](#)
- [Esquemas XML, 191](#)
- [Usando um Esquema para Mapear Âncoras, 194](#)
- [Gerando um XML Válido, 195](#)
- [Variáveis, 197](#)
- [Referência de Componentes de Variável, 200](#)
- [Recipientes de Dados de Ocorrência Múltipla, 201](#)

Visão Geral de Recipientes de Dados

Um recipiente de dados é um objeto que tem um dos seguintes tipos:

- Um elemento XML
- Um atributo XML
- Uma variável

Os elementos e atributos XML são normalmente usados para armazenamento permanente. Um Analisador, por exemplo, armazena a sua saída em recipientes de dados desses tipos.

As variáveis são usadas para armazenamento temporário. Por exemplo, um Analisador pode armazenar dados que extrai de um documento de origem em uma variável. Ele pode processar mais os dados antes de criar a saída.

Cada recipiente de dados tem um tipo de dados. No caso de elementos e atributos, os recipientes de dados são definidos em um esquema XML que você deve fornecer. As variáveis são definidas em um esquema interno, que você pode personalizar adicionando variáveis definidas pelo usuário.

Esquemas XML

Quando você cria um Analisador, um Serializador, um XMap ou um Mapeador, deve fornecer um ou mais esquemas XML que definem a estrutura do XML. O esquema define os elementos e atributos que a transformação pode usar.

Adicione o esquema ao repositório do Modelo. Você pode mapear o conteúdo de um documento para os elementos e atributos que são definidos no esquema.

Codificação de Esquema

Salve o esquema em uma das codificações de entrada com suporte.

A codificação de esquema deve ser compatível com a codificação de trabalho que você utiliza no editor do IntelliScript. Isso significa que:

- A codificação de esquema é idêntica à codificação de trabalho ou
- Cada caractere no esquema tem um equivalente na codificação de trabalho. Por exemplo, se o esquema utilizar a codificação UTF-8, e a codificação de trabalho for Windows-1252, o esquema não deverá conter caracteres Unicode sem um equivalente em Windows-1252.

Quando você adiciona um esquema a um projeto a partir de uma localização externa, o Script converte a cópia do projeto desse esquema na codificação de trabalho.

Arquivos com Esquema Incluído

Um esquema pode fazer referência a outros arquivos de esquema. Esse recurso permite que você mantenha um grande esquema de forma modular.

Espaços de Nome

Se você planeja trabalhar com espaços de nome de XML, use o atributo **targetNamespace** do esquema. Você pode editar o alias atribuído ao espaço de nome.

Não é possível adicionar dois esquemas que usem um alias vazio ou o mesmo alias para diferentes espaços de nome.

Conteúdo Misto

Elementos podem conter dados de caracteres e elementos aninhados. É possível usar o atributo **mixed** em um esquema.

O Script faz distinção entre dados de caracteres antes e depois de cada elemento. Para obter mais informações, consulte ["Mapeando Conteúdo Misto" na página 194](#).

Recursos de Esquema sem Suporte

A versão atual não oferece suporte para determinados usos de recursos de esquema. A seguinte tabela lista as limitações conhecidas:

Recurso	Limitação
Restrições de exclusividade	Os elementos unique , key e keyref são ignorados. O log de eventos inclui um aviso.
Valores padrão para elementos de tipo misto	O Script ignora o padrão. O log de eventos inclui um aviso.

Recurso	Limitação
Tipo de dados padrão	Se o tipo de um elemento for indefinido, o Script o processará como <code>xs:string</code> . O log de eventos inclui um aviso. É possível alterar o padrão para <code>xs:anyType</code> .
Expressões regulares	Existem pequenas discrepâncias entre o processador de expressões regulares e o padrão do esquema.
Sequência definindo vários elementos que possuem o mesmo nome	Se um <code>xs:sequence</code> contiver várias definições <code>xs:element</code> com o mesmo nome, o Script processará somente o primeiro <code>xs:element</code> . O log de eventos inclui um aviso. Para resolver o problema, envolva cada <code>xs:element</code> em um <code>xs:sequence</code> independente.
Datas mínimas e máximas	Se uma faceta definir um valor mínimo ou máximo para um elemento <code>xs:date</code> , a transformação falhará.
Negligenciar ou ignorar opções de validação	Em um elemento <code>xs:any</code> ou <code>xs:anyAttribute</code> , o Script ignora um valor processContents de <code>lax</code> ou <code>skip</code> . Ele se comporta como se o valor fosse <code>strict</code> .
Grupo de substituição	O Script permite um substitutionGroup , mesmo que um atributo block ou blockDefault proíba substituições.
Tipo XSI	O Script permite um atributo <code>xsi:type</code> mesmo quando um atributo block de o proíbe.
Tipos internos	Alguns tipos internos não têm padrões corretos, por exemplo, quando incluem caracteres acima de ASCII 127.
Grupo de substituição sem um tipo	O Script às vezes falha quando um grupo de substituição não tem um tipo.
Espaço de nome vazio	Quando o espaço de nome está vazio, o Script adiciona um alias a todos os elementos no arquivo de origem, mas o alias não aparece em Locator , e Locator falha.
Lista	O Script lê um <code>xs:list</code> separado por espaço como um único item, que poderá falhar se o comprimento exceder o limite informado para itens individuais na lista.
Flutuantes e duplos	<code>xs:float</code> e <code>xs:double</code> não aceitam valores válidos de INF , -INF ou NaN .
Elemento com atributos fixos e mistos	O Script não lê todas as partes de um elemento que possui atributos fixos e mistos.
max_occurs=0	O Script cria uma saída mesmo quando <code>max_occurs=0</code> .
Token	O Script não analisa um <code>xs:token</code> que contém tabulações, retornos de carro ou avanços de linha.
String normalizada	O Script não carrega um XML quando um <code>xs:normalizedString</code> contém tabulações, retornos de carro ou avanços de linha.

Precisão de Dados Numéricos

O Script armazena dados `xs:decimal` e `xs:float` como strings, preservando a precisão dos dados.

Em cálculos, o Script converte dados decimais e flutuantes em um ponto flutuante de precisão dupla e arredonda o resultado para 15 dígitos decimais. Isso significa que dados decimais podem perder uma certa precisão. Por exemplo, o resultado de `xs:decimal 5,28 * 1` pode ser exibido como 5,280000000000001.

O Script normaliza valores `xs: decimal`. Por exemplo, ele armazena 0004 como 4, -0 como 0 e 1,200 como 1,2.

Usando um Esquema para Mapear Âncoras

Quando você define um Analisador, deve mapear as âncoras de `Conteúdo` para os recipientes de dados de saída. Quando você define um Serializador, deve mapear os recipientes de dados de entrada para as âncoras de serialização `ContentSerializer`.

Representação de Recipientes de Dados do IntelliScript

No Script, recipientes de dados são identificados por uma expressão XPath modificada, como:

```
data_holder = /Person/*s/Name/*s/First
```

Para alterar esse valor, selecione a propriedade **data_holder** e pressione **ENTER**. Isso abre uma caixa de diálogo **Escolher XPath**, na qual você pode selecionar o novo valor.

A sintaxe de XPath é um pouco diferente da sintaxe de XPath padrão, que é `Person/Name/First`. O Script insere `*s`, `*c` e `*a`, que fazem referência aos termos de esquema **sequence**, **choice** e **all**. As modificações resolvem ambiguidades quando o Script usa o esquema para ajudar a construir a saída XML.

Mapeando Conteúdo Misto

Se o esquema suportar conteúdo misto, cada elemento terá recipientes de dados **antes** e **depois**. Por exemplo, considere o seguinte conteúdo misto:

```
<Deal>
  We are pleased to offer you a price of
  <Price>34</Price>
  dollars. This is a special price for
  <Partner>
    <Name>Acme Gizmos, Inc.</Name>
    <ID>98765</ID>
  </Partner>
  valid only until December 31.
</Deal>
```

Essa estrutura contém recipientes de dados nas seguintes localizações:

- Imediatamente após a marca `<Deal>`, antes de qualquer dos subelementos.
- Antes do elemento `Price`
- No elemento `Price`
- Depois do elemento `Price`
- Antes do elemento `Partner`
- Nos elementos `Partner/Name` e `Partner/ID`
- Depois do elemento `Partner`
- Imediatamente antes da marca `</Deal>`, depois de qualquer dos subelementos.

Você pode mapear o texto "We are pleased to offer you a price of" para o recipiente de dados antes do elemento `Price`. Você pode mapear "dollars. " para o recipiente de dados após `Price` e "This is a special price for " para o recipiente de dados antes de `Partner`.

O exemplo a seguir mostra conteúdo misto:

```
data_holder = /Deal/*s/Price/$text_before
```

Mapeando Tipos XSI

Um esquema pode definir tipos de dados derivados que podem ser usados no lugar de um tipo base. Nesses casos, um documento XML pode definir o tipo de dados real de um elemento especificando um atributo `xsi:type`.

Por exemplo, um esquema define um elemento `Person` como tendo um tipo `PersonT1` e um conteúdo de string. Ele define um tipo denominado `PersonT2` que estende `PersonT1` adicionando um atributo `Id`. Os seguintes elementos `Person` são válidos:

```
<!-- base type PersonT1 -->
<Person>Ron Lehrer</Person>

<!-- derived type PersonT2 -->
<Person Id="547329876" xsi:type="PersonT2">Ron Lehrer</Person>
```

O Script interpreta atributos `xsi:type` em documentos XML de entrada. Ele adiciona atributos `xsi:type` onde for necessário para gerar a saída de documentos XML.

Selecione o tipo apropriado de acordo com os dados que a transformação processa. Por exemplo, se você quiser que uma âncora de **Conteúdo** armazene dados em um elemento `Person` que possui um tipo `PersonT2`, selecione `xsi:type=PersonT2`. A seleção é exibida no Script da seguinte maneira:

```
data_holder=/Person/*c/xsi:type=PersonT2
```

Em casos nos quais o conteúdo pode exigir um recipiente de dados `PersonT1` ou `PersonT2`, é possível configurar uma âncora de **Alternativas** que contém duas âncoras de **Conteúdo**. Uma das âncoras de **Conteúdo** é mapeada para `PersonT1`, enquanto a outra é mapeada para `PersonT2`. Para obter mais informações, consulte ["Alternativas" na página 216](#).

Se você mapear um recipiente de dados para o elemento `Person` não qualificados, o recipiente de dados assumirá como padrão o tipo base `PersonT1`. Portanto, as seguintes mapeamentos são equivalentes:

```
data_holder=/Person
data_holder=/Person/*c/xsi:type=PersonT1
```

Gerando um XML Válido

O Script gera um XML que é válido de acordo com o esquema de saída que você definiu.

Esse esquema é usado como guia enquanto o XML está sendo gerado. Ele é aplicado durante a geração, e não depois. Essa abordagem auxilia na conclusão bem-sucedida de transformações. Ela garante a validade continuamente à medida que a transformação prossegue.

Função de Esquemas na Análise

Esta seção explica algumas das maneiras pelas quais um Analisador usa o esquema para garantir que ele gere um XML válido.

A discussão apresenta exemplos do comportamento.

Sequência de Elementos

Quando o Script executa um Analisador, ele organiza a saída na sequência necessária para o esquema.

Por exemplo, um esquema pode exigir que um elemento **LastName** preceda um elemento **FirstName**. O Script cria a saída nas localizações definidas pelo esquema, mesmo quando as âncoras que produzem a saída estão definidas na sequência oposta.

Número de Ocorrências

Um Analisador pode tentar inserir várias instâncias de um elemento no XML de saída. O Script usa o esquema para determinar se novas instâncias devem ser acrescentadas ou se elementos existentes devem ser substituídos. O Analisador exclui quaisquer elementos em excesso além dos permitidos pelo esquema e grava avisos no log de eventos.

Em outro exemplo, suponha que o esquema defina um elemento sem especificar um atributo **minOccurs** ou **maxOccurs**. Os valores padrão de **minOccurs** e de **maxOccurs** são 1, o que significa que o elemento deve ocorrer exatamente uma vez na saída do Analisador. Se o elemento estiver ausente da saída, o Analisador poderá adicioná-lo.

Para obter mais informações, consulte [“Recipientes de Dados de Ocorrência Múltipla” na página 201](#).

Elementos Ausentes ou Vazios

Nas configuração de transformação do Processador de Dados, você pode configurar se um Analisador inserirá elementos vazios para estar em conformidade com um esquema.

Tipos de Dados

O Script garante que o texto que ele armazena em um recipiente de dados possui o tipo de dados necessário. Por exemplo, se uma âncora de **Conteúdo** recuperar a string "oranges 5 for a dollar", e o tipo do recipiente de dados for `xs:integer`, essa âncora armazenará somente o inteiro 5 no recipiente de dados.

Para obter mais informações, consulte [“Usando Tipos de Dados para Restringir os Critérios de Pesquisa” na página 214](#).

Função de Esquemas na Serialização e no Mapeamento

Um serializador ou mapeador verifica se a sua entrada é válida de acordo com o esquema XML. Há dois modos de validação:

- Validação parcial. Alguns desvios são permitidos entre o documento de origem XML e o esquema. Padrão.
- Validação estrita. O documento de origem XML deve obedecer estritamente ao esquema.

Para definir o nível de validação, atribua a propriedade **validate_source_document** do componente **Serializador** ou **Mapeador**.

Se você usar o modo estrito, um erro de validação fará com que o serializador ou mapeador falhe. A exibição **Eventos** mostra os erros.

Se você usar o modo parcial, a transformação poderá continuar, apesar de certos erros de validação. Por exemplo, se houver mais ocorrências de um elemento do que o esquema permite, um serializador ignorará os elementos em excesso e processará os elementos válidos, gravando um aviso no log de eventos. Da mesma forma, ele poderá ignorar um elemento que contém um tipo de dados inválido.

O Script usa o Analisador XML Xerces C, versão 3.1, para realizar a validação.

Variáveis

Variáveis são recipientes de dados temporários que podem ser usados no lugar de atributos ou elementos XML. Variáveis são úteis quando você precisa armazenar um valor temporariamente durante a operação de uma transformação, mas não precisa gerar a saída desse valor no XML.

Por exemplo, suponha que você queira que o Analisador leia duas âncoras de **Conteúdo** e concatene seus valores. É possível mapear cada âncora de **Conteúdo** para uma variável definida pelo usuário. Em seguida, é possível usar uma ação para concatenar as variáveis e gerar a saída do resultado em um elemento XML.

O Script também usa variáveis do sistema predefinidas para armazenar informações que são necessárias em determinadas operações.

Criando uma Variável Definida pelo Usuário

1. Adicione um componente de **Variável** no nível global do Script.
2. Insira um nome para a variável e pressione **ENTER**.
3. Selecione o tipo de dados que a variável pode armazenar.

É possível selecionar um tipo padrão, como `xs:string` ou `xs:integer`, ou um tipo global definido em um esquema referenciado no projeto.

Variáveis de Sistema

Os seguintes parágrafos descrevem as variáveis do sistema e as formas pelas quais elas são usadas.

Variáveis Usadas para Acessar Documentos de Origem

Muitas das variáveis de sistema armazenam dados que podem ser usados por ações para acessar documentos de origem. Por exemplo, a ação **RunParser** pode usar `VarLinkURL`, que contém um caminho de arquivo.

A seguinte variável é usada no processador **XmlToDocument**:

Variável	Descrição
VarServiceInfo > <i>ServiceLocation</i>	O caminho de diretório do Script ou serviço que está em execução.

Variáveis de Acesso Somente Leitura

As variáveis a seguir são somente leitura. Uma transformação pode usá-las para visitar um documento de origem mais de uma vez.

Variável	Descrição
<i>VarRequestedURL</i>	O caminho do documento de origem que um Analisador está processando.
<i>VarCurrentURL</i>	O caminho do arquivo atual que um Analisador está processando. Normalmente, ele é o mesmo que VarRequestedURL . Se o Analisador estiver configurado com determinados pré-processadores, VarCurrentURL poderá apontar para um arquivo temporário em vez do documento de origem original. VarRequestedURL sempre aponta para o documento de origem.
<i>VarCurrentPost</i>	Os dados de formulário que um Analisador enviou para recuperar a página atual.

Variáveis de Tempo de Sistema Somente Leitura

VarSystem é uma variável de somente leitura que retorna as informações do sistema. É uma estrutura que contém várias variáveis aninhadas:

Variável	Descrição
VarSystem > ExecStartTime > <i>Ano</i>	Ano em que a transformação iniciou a execução
VarSystem > ExecStartTime > <i>Mês</i>	Mês numérico
VarSystem > ExecStartTime > <i>MonthName</i>	Nome do mês
VarSystem > ExecStartTime > <i>Dia</i>	Dia do mês
VarSystem > ExecStartTime > <i>DayName</i>	Dia da semana
VarSystem > ExecStartTime > <i>Hora</i>	Hora
VarSystem > ExecStartTime > <i>Minuto</i>	Minuto
VarSystem > ExecStartTime > <i>Segundo</i>	Segundo
VarSystem > ExecStartTime > <i>Milissegundos</i>	Milissegundos

Você pode usar a **VarSystem** para inserir um registro de data/hora na saída de uma transformação.

Variáveis Usadas para Manipulação de Falhas

VarLastFailure armazena a falha de componente mais recente que ocorreu em uma transformação. Por exemplo, ele pode registrar uma instância de uma âncora de **Marcador** que falhou ao localizar o texto de marcador. É possível configurar um componente para gravar **VarLastFailure** em um log de usuário quando uma falha ocorrer. Para obter mais informações, consulte [“Tratamento de Falhas” na página 399](#).

Nota: Quando você usa **VarLastFailure**, o serviço é executado no modo especial, que exige cerca de três vezes mais tempo de CPU.

VarServiceInfo armazena o nome do serviço, a localização no diretório do log do usuário e o nome de arquivo do log do usuário.

VarLastFailure e **VarServiceInfo** são estruturas que contêm as seguintes variáveis aninhadas:

Variável	Descrição
VarLastFailure > <i>InternalId</i>	Identificador de falha
VarLastFailure > <i>Text</i>	Descrição da falha
VarLastFailure > <i>Location</i>	Localização da falha no Script
VarLastFailure > <i>AnchorName</i>	Nome do componente que falhou
VarLastFailure > <i>Data</i>	Informações adicionais sobre a falha
VarServiceInfo > <i>ServiceName</i>	Nome do serviço
VarServiceInfo > StandardError > <i>StandardErrorDir</i>	Caminho de diretório do log do usuário
VarServiceInfo > StandardError > <i>StandardErrorName</i>	Nome de arquivo do log do usuário

Variáveis Usadas para Análise Estruturada

VarStructureDetails mantém o rastreamento do registro atual que uma âncora **StructureDefinition** está analisando. Ele contém as seguintes variáveis aninhadas:

Variável	Descrição
VarStructureDetails > <i>Nome</i>	A propriedade nome do subelemento que corresponde ao registro.
VarStructureDetails > <i>Repetições</i>	O número de iteração do registro.
VarStructureDetails > <i>RecordIndex</i>	O número de índice do registro na entrada geral de StructureDefinition .
VarStructureDetails > <i>RecordId</i>	O identificador recorfd. Se houver vários identificadores, a variável conterá uma lista separada por vírgula.
VarStructureDetails > <i>InternalPath</i>	Informações internas, não usar.

Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

Variáveis Usadas em Notificações

VarNotificationDetails armazena informações sobre a notificação mais recente que foi disparada em uma transformação:

Variável	Descrição
VarNotificationDetails > <i>Nome</i>	O nome da notificação.
VarNotificationDetails > <i>Caminho</i>	O XPath do recipiente de dados ao qual a notificação se aplica. Por exemplo, se um validador disparar uma notificação em uma âncora de Conteúdo , o Caminho será o recipiente de dados no qual a âncora de Conteúdo armazena sua saída.
VarNotificationDetails > <i>Valor</i>	O valor de entrada que causou a notificação. Se um validador disparar a notificação, o Valor corresponderá aos dados de entrada inválidos. Se uma ação Notificar disparar a notificação, você poderá especificar o Valor na configuração de Notificar .
VarNotificationDetails > <i>Criador</i>	A localização no Script que disparou a notificação.

Para obter mais informações, consulte [“Notificações” na página 418](#).

Âncoras de Mapeamento para Variáveis

Você pode mapear uma âncora de **Conteúdo** para uma variável da mesma maneira que você mapeia qualquer outro recipiente de dados.

Não mapeie uma âncora para uma variável de sistema somente leitura.

Usando Variáveis em Ações

Variáveis são frequentemente usadas como entrada de ações. Você pode usar uma variável da mesma maneira que utiliza outros recipientes de dados. Para obter mais informações, consulte [“Visão Geral de Ações” na página 295](#).

Iniciando Variáveis em Tempo de Execução

Você pode inicializar os valores de variáveis das seguintes maneiras:

- No Script, é possível configurar a propriedade **initialization** da **Variável**.
Os valores iniciais definidos com essa abordagem são usados quando você executa a transformação na ferramenta Developer ou como um serviço.
- Um aplicativo pode transmitir os valores iniciais como parâmetros de serviço a um serviço em tempo de execução.
Os parâmetros de serviço substituem a propriedade **initialization** das variáveis. Se o Script especificar um valor inicial, e você também transmitir um valor de um aplicativo, o último valor será usado.

Referência de Componentes de Variável

Uma componente de **Variável** é uma variável definida pelo usuário.

Para obter mais informações sobre variáveis de sistema, consulte [“Variáveis de Sistema” na página 197](#).

Variável

Uma componente **Variável** é uma variável definida pelo usuário que você usa em um Script.

Use variáveis para armazenamento temporário da mesma maneira que você usa um atributo ou elemento XML. Por exemplo, você pode mapear uma âncora **Conteúdo** para uma variável e pode usar uma variável como entrada de uma ação.

Variáveis são exibidas no nível global do Script. Uma variável pode ter qualquer tipo de dados definido nos esquemas associados ao projeto, incluindo tipos padrão e tipos personalizados. Um tipo personalizado pode ser simples ou complexo. Uma variável complexa é uma estrutura que contém campos aninhados. Não há suporte para a inicialização de variáveis complexas. Para obter mais informações, consulte [“Inicializando Variáveis em Tempo de Execução” na página 200](#).

A seguinte tabela descreve as propriedades do componente **Variável**:

Propriedade	Descrição
initialization	Define um valor inicial para a variável. É possível inicializar variáveis que possuem tipos de dados simples. O padrão é vazio.
InitialValue	Define um valor inicial para a variável. InitialValue tem uma propriedade, value .
list	Determina se a variável é de ocorrência única ou de ocorrência múltipla. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selected. Determina uma variável de ocorrência múltipla.- Cleared. Determina uma variável de ocorrência única. O padrão é Desmarcado. Para obter mais informações, consulte “Recipientes de Dados de Ocorrência Múltipla” na página 201 .
val_type	Define o tipo de dados que a variável pode armazenar. Valores legais são definidos no esquema. O padrão é <code>xs:string</code> .
value	Define o valor inicial.

Recipientes de Dados de Ocorrência Múltipla

Em um esquema, é possível usar o atributo **maxOccurs** para definir o número máximo de vezes que elementos de mesmo nível podem ocorrer em um documento XML. Da mesma forma, é possível definir uma variável que pode ocorrer uma ou várias vezes. Um elemento ou variável que pode ocorrer apenas uma vez é chamado de recipiente de dados de ocorrência única. Um elemento ou variável que pode ocorrer mais de uma vez é chamado de recipiente de dados de ocorrência múltipla.

Recipientes de dados de ocorrência única e de ocorrência múltipla apresentam comportamentos diferentes quando o Script armazena dados neles, por exemplo, quando você mapeia âncoras de **Conteúdo** para um recipiente de dados.

- Em um recipiente de dados de ocorrência única, cada atribuição substitui a atribuição precedente.
- Em um recipiente de dados de ocorrência múltipla, cada atribuição gera uma nova ocorrência do recipiente de dados.

Para entender isso, suponha que um esquema defina um elemento XML denominado `<FirstName>`. Se `maxOccurs = 1`, trata-se de um recipiente de dados de ocorrência única. Se um Analisador mapear mais de uma âncora de **Conteúdo** para o elemento `<FirstName>`, a saída conterá apenas o mapeamento final.

Considere o que aconteceria se você analisasse um documento de origem que consiste em uma lista de nomes:

```
Jack Jennie Larissa
```

Suponha que cada nome seja uma âncora de **Conteúdo** mapeada para **FirstName**. Cada nome substitui o valor de **FirstName**. A saída contém apenas o mapeamento:

```
<FirstName>Larissa</FirstName>
```

Agora, suponha que `maxOccurs = unbounded`. Isso significa que **FirstName** é um recipiente de dados de ocorrência múltipla. Se você mapear várias âncoras de **Conteúdo** para o elemento, o Analisador gerará uma lista de nomes. A saída é:

```
<FirstName>Jack</FirstName>
<FirstName>Jennie</FirstName>
<FirstName>Larissa</FirstName>
```

O mesmo princípio se aplica a variáveis. Se você mapear várias âncoras para uma variável de ocorrência múltipla, cada âncora gerará uma nova ocorrência dessa variável. É possível usar esse recurso, por exemplo, para preparar a entrada para as ações **AppendListItems** e **CombineValues**, que concatenam as ocorrências.

Nota: O comportamento descrito aqui pressupõe que o recipiente de dados de ocorrência múltipla tenha um tipo de dados simples. Em determinadas circunstâncias, se o tipo for complexo, cada âncora talvez não gere uma nova ocorrência. Para controlar esse comportamento, é possível usar um localizador. Para obter mais informações, consulte [“Visão Geral de Localizadores, Chaves e Indexação” na página 360](#).

Atributos

Um atributo XML é sempre um recipiente de dados de ocorrência única. Um atributo não pode ser de ocorrência múltipla porque o XML não permite que o mesmo atributo apareça mais de uma vez no mesmo elemento.

Um atributo pode ter um tipo de dados que é uma lista separada por espaços. O atributo `names` no seguinte elemento é um exemplo:

```
<Countries names="USA Canada Mexico"/>
```

O Script trata o atributo como atributo de ocorrência única com um tipo de lista. Para obter mais informações, consulte [“Usando Tipos de Dados para Restringir os Critérios de Pesquisa” na página 214](#).

Indexação

Por padrão, o Script acessa as instâncias de um recipiente de dados de ocorrência múltipla sequencialmente. É possível acessar as instâncias não sequencialmente usando o recurso de indexação. Para obter mais informações, consulte [“Visão Geral de Localizadores, Chaves e Indexação” na página 360](#).

Destruindo as Ocorrências

Em determinadas circunstâncias, talvez você queira destruir todas as ocorrências existentes de um recipiente de dados de ocorrência múltipla e começar a criar novas ocorrências a partir do início da lista. Isso é útil, por exemplo, quando você está analisando uma estrutura iterativa e deseja manter somente a última iteração. É possível destruir as ocorrências que armazenam dados de iterações anteriores.

Esse efeito pode ser obtido definindo um recipiente de dados de ocorrência única que contém um elemento aninhado de ocorrência múltipla. Quando o recipiente de dados de ocorrência única for reutilizado, as ocorrências aninhadas serão destruídas.

O seguinte cenário é um exemplo típico.

1. Adicione o seguinte esquema a um projeto:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:complexType name="MyListType">
    <xs:sequence>
      <xs:element name="item" type="xs:string" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

O esquema define um tipo de dados personalizado denominado **MyListType**. O tipo contém um elemento aninhado de ocorrência múltipla denominado **item**.

2. Defina uma variável de ocorrência única denominada **MyList** e que possui o tipo de dados **MyListType**.
3. Use essa variável como destino de uma estrutura iterativa.

Para obter mais informações, consulte [“Visão Geral de Localizadores, Chaves e Indexação” na página 360](#).

Cada iteração reutiliza a ocorrência única de **MyList**. No início da iteração, os elementos de **item** aninhados são destruídos. Âncoras dentro da estrutura iterativa, como um **RepeatingGroup** aninhado, começam a atribuir os elementos de **item** a partir do início da lista.

Exemplo Online

Para ver um exemplo online de como destruir ocorrências múltiplas de um recipiente de dados, consulte o seguinte exemplo:

```
samples\Projects\ResetListVariable\ResetListVariable.cmw
```

CAPÍTULO 16

Âncoras

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Âncoras, 204](#)
- [Mapeando Âncoras de Conteúdo para Recipientes de Dados, 205](#)
- [Definindo Âncoras, 206](#)
- [Propriedades de Âncoras Padrão, 208](#)
- [Como um Analisador Procura Âncoras, 209](#)
- [Referência do Componente de Âncora, 216](#)
- [Referência de Componentes de Pesquisador, 245](#)
- [Referência do Subcomponente de Âncora, 249](#)

Visão Geral de Âncoras

As âncoras são componentes que permitem que um Analisador se fixe em localizações específicas de um documento de origem com a finalidade de encontrar dados e armazená-los em recipientes de dados. Uma âncora é um ponto de sinalização que você coloca em um documento, indicando a posição dos dados.

Este capítulo explica os diferentes tipos de âncoras e como você pode usá-las em Analisadores.

Âncoras de Marcador e Conteúdo

As âncoras mais comumente usadas são chamadas de âncoras de **Marcador** e **Conteúdo**. Essas âncoras são frequentemente usadas em par:

- Uma âncora de **Marcador** rotula uma localização em um documento.
- Uma âncora de **Conteúdo** recupera o texto da localização.

Para usar essas âncoras, imagine um questionário impresso. A primeira linha normalmente pergunta o sobrenome e o nome da pessoa, com cada rótulo seguido por um espaço em branco para receber as informações. Os rótulos impresso **Sobrenome** e **Nome** são âncoras de **Marcador** e os espaços em branco são âncoras de **Conteúdo**. As âncoras fornecem um meio para encontrar os dados e extraí-los do documento de origem.

Outros Tipos de Âncoras

Além das âncoras de **Marcador** e **Conteúdo**, há muitos outros tipos de âncoras que você pode usar para analisar documentos. Por exemplo, as âncoras de **Grupo** e **RepeatingGroup** ajudam você a especificar a

organização dos campos de dados. Uma âncora de **Alternativas** permite que você especifique vários tipos de dados que podem ocorrer em uma determinada localização de um documento de origem.

Como Âncoras e Delimitadores Trabalham Juntos

É possível definir as âncoras no exemplo de documento de origem. O Analisador aprende a analisar o documento examinando as âncoras e os delimitadores que as separam. Para obter mais informações sobre delimitadores, consulte [“Visão Geral de Formatos” na página 177](#).

Por exemplo, suponha que você tenha especificado que o seu documento utilize um formato delimitado por tabulação. Uma linha na origem de exemplo indica

```
First name:<tab>Ron
```

em que <tab> é um caractere de tabulação.

É possível definir `Nome:` como uma âncora de **Marcador**. É possível definir `Ron` como uma âncora de **Conteúdo**. O Analisador aprende com essas definições que ele deve pesquisar um documento de origem para a string `Nome:`. Em seguida, ele deve pular um único delimitador de tabulação e recuperar o texto após a tabulação.

Suponha que você execute o Analisador em outro documento de origem, que contém o seguinte texto:

```
First name:<tab>Jack
```

O Analisador localiza as âncoras conforme especificado acima e recupera o texto `Jack`.

Agora, suponha que o documento de origem indique:

```
First name:<tab>Jack<tab>Age:<tab>34
```

O Analisador ainda recupera o texto `Jack` em vez de `Jack<tab>Age<tab>34`. Isso funciona porque você definiu o caractere de tabulação como um delimitador. O Script entende que a âncora de **Conteúdo** começa depois da primeira tabulação e termina antes da segunda tabulação. Obviamente, você pode definir âncoras adicionais que recuperam a idade de Jack, que é 34.

Nota: Os exemplos acima descrevem um possível comportamento das âncoras e dos delimitadores. As âncoras possuem muitas propriedades que permitem alterar esse comportamento. Por exemplo, você pode definir uma âncora de **Conteúdo** que ignora tabulações, mesmo em um formato delimitado por tabulação. Para obter mais informações, consulte [“Como um Analisador Procura Âncoras” na página 209](#).

Mapeando Âncoras de Conteúdo para Recipientes de Dados

Uma âncora de **Conteúdo** armazena o texto que ela extrai de um documento de origem em um recipiente de dados. Por exemplo, você pode configurar uma âncora de **Conteúdo** para armazenar seu resultado em um elemento XML chamado **FirstName**. Se a âncora de **Conteúdo** recuperar o texto `Jack`, o Analisador produzirá a seguinte saída:

```
<FirstName>Jack</FirstName>
```

Mais precisamente, você pode especificar que o âncora deve armazenar o texto recuperado no caminho `Person/*s/FirstName`, que se refere a um elemento definido no esquema XML. A saída do Analisador atual seria:

```
<Person>
  <FirstName>Jack</FirstName>
</Person>
```

Por outro lado, suponha que o esquema define **FirstName** como um atributo do elemento **Pessoa**. Você pode mapear a âncora de **Conteúdo** para `/Person/@FirstName`. A saída seria:

```
<Person FirstName="Jack" />
```

Você deve mapear um recipiente de dados que tenha um tipo de dados apropriado. Por exemplo, não mapeie **Jack** para um elemento XML que tem um tipo de dados `xs:integer` ou para um elemento XML que tenha um tipo de dados complexo que contém elementos aninhados. Para obter mais informações sobre essa regra, consulte [“Usando Tipos de Dados para Restringir os Critérios de Pesquisa” na página 214](#).

Mapeando para Variáveis

Você pode mapear uma âncora para um recipiente de dados que é um elemento XML, um atributo XML ou uma variável. A opção variável será útil se você desejar usar os dados em uma etapa de processamento subsequente, mas não incluir os dados brutos na saída do Analisador.

Por exemplo, suponha que você deseja extrair vários números de um documento de origem e gerar saída da soma no XML. Você não quer os números individuais na saída. Você pode mapear as âncoras de **Conteúdo** para recuperar os números para variáveis e usar uma ação `CalculateValue` para computar e gerar saída da soma.

Você também pode mapear para uma variável que você usar em uma âncora subsequentes, por exemplo, para definir um texto de pesquisa para uma âncora de `Marcador`.

Mapeando para Recipientes de Dados de Ocorrência Múltipla

Se você mapear âncoras de **Conteúdo** para um recipiente de dados de ocorrência única, cada atribuição de recipiente de dados substituirá a atribuição anterior.

Se você mapear um recipiente de dados de ocorrência múltipla, cada atribuição gerará uma nova ocorrência do recipiente de dados. Por exemplo, se cada âncora de **Conteúdo** recuperar um nome de pessoa, a saída será uma lista de nomes:

```
<FirstName>Jack</FirstName>
<FirstName>Jennie</FirstName>
<FirstName>Larissa</FirstName>
```

Para obter mais informações, consulte [“Recipientes de Dados de Ocorrência Múltipla” na página 201](#).

Mapeando para Elementos de Conteúdo Misto

O termo conteúdo misto refere-se a um elemento XML que contém os dados de caracteres e os elementos aninhados. Se o esquema permitir que um elemento tenha conteúdo misto, a exibição de Esquema mostrará recipientes de dados antes e depois dos elementos. Isso permite que você mapeie uma âncora de **Conteúdo** para dados de caracteres que estejam localizados antes ou depois de um determinado elemento aninhado. Para obter mais informações, consulte [“Mapeando Conteúdo Misto” na página 194](#).

Definindo Âncoras

Quando você definir um componente `Analisador`, deverá adicionar uma sequência de âncoras. O analisador opera procurando as âncoras no documento de origem e executando as operações que você configurou para serem realizadas pelas âncoras.

Onde Definir Âncoras

No Script, as âncoras estão aninhadas dentro de um **Analisador**.

Se você pressionar **ENTER** na localização indicada, o editor do IntelliScript exibirá uma lista suspensa que inclui as âncoras e outros componentes que você pode adicionar.

Após a adição das âncoras, a ferramenta Developer as realça na origem de exemplo.

Alguns tipos de âncoras podem conter âncoras aninhadas. Por exemplo, é possível aninhar âncoras dentro de uma âncora de **Alternativas**, **Grupo** ou **RepeatingGroup**.

Sequência de Âncoras

A sequência de âncoras deve ser a sequência de texto no documento de origem.

Por exemplo, suponha que o documento de origem seja:

```
First Name: Ron  
Last Name: Lehrer
```

Considerando que você definiu `Nome` e `Sobrenome` como âncoras de `Marcador` e que definiu `Ron` e `Lehrer` como âncoras de `Conteúdo`, a sequência necessária de âncoras na configuração do Analisador é:

Âncora	Texto no Documento de Origem
Marcador	First Name
Conteúdo	Ron
Marcador	Last Name
Conteúdo	Lehrer

Exceção: Sequência de Origem de Variável

Alguns documentos de origem podem ter uma sequência de variável. Por exemplo, suponha que o documento de origem tenha qualquer um dos seguintes formatos:

```
First Name: Ron  
Last Name: Lehrer
```

ou

```
Last Name: Lehrer  
First Name: Ron
```

Nesse casos, você pode usar a propriedade de `marcação` para alterar o escopo de pesquisa das âncoras. Para obter mais informações, consulte [“Como um Analisador Procura Âncoras” na página 209](#).

Adicionando uma Âncora Marcador ou Conteúdo

Você pode adicionar âncoras **Marcador** e **Conteúdo** por meio de uma abordagem selecionar e clicar.

1. Selecione o texto da âncora no exemplo de arquivo de origem.
2. Clique com o botão direito do mouse no texto selecionado e clique em **Inserir Marcador** ou **Inserir Conteúdo**.
3. Defina as propriedades da âncora.

Definindo uma Âncora

É possível criar qualquer tipo de âncora editando o Script. O procedimento é idêntico para a edição de qualquer outro componente.

1. Na localização da âncora desejada, selecione o sinal de reticências (...) e pressione **ENTER**.
2. Selecione ou digite o nome da âncora.
3. Pressione **ENTER** novamente para confirmar sua seleção.
4. Edite as propriedades da âncora.

Propriedades de Âncoras Padrão

A seguinte tabela descreve as propriedades padrão de âncoras:

Propriedade	Descrição
direction	<p>Uma direção de pesquisa para a âncora dentro do escopo de pesquisa. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- backward. Pesquisa a partir do final do escopo de pesquisa e localiza a última instância da âncora.- forward. Pesquisa a partir do início do escopo de pesquisa e localiza a primeira instância da âncora. <p>Para uma âncora de Marcador, é possível modificar esse comportamento usando a propriedade count. Por exemplo, se direction = backward e count = 2, o Script localizará a penúltima instância. O padrão é forward. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p>
disabled	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
marking	<p>Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- begin position. Insere um ponto de referência antes da âncora atual.- end position. Insere um ponto de referência depois da âncora atual.- full. Insere um ponto de referência antes e depois da âncora atual.- none. Não cria um ponto de referência. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p>
name	<p>Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos. Use a propriedade name para identificar o componente que causou o evento.</p>
no_initial_phase	<p>Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none">- Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais.- Selected. Procura âncoras aninhadas na fase principal. <p>O padrão é Desmarcado.</p>
notifications	<p>Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418.</p>

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
phase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p>
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Quando não estiver claro que uma âncora existe em um documento de origem, selecione a propriedade **optional**. Se a âncora não existir, o **Analisador** no qual ela está aninhada continuará.

Se a âncora estiver aninhada dentro de uma âncora de **Grupo**, a propriedade **optional** impedirá que esse **Grupo** falhe. Se a âncora estiver em um **RepeatingGroup**, a propriedade impedirá a falha de uma iteração de **RepeatingGroup**.

Como um Analisador Procura Âncoras

Para criar um Analisador corretamente, é importante que você entenda como o Script procura as âncoras no Analisador. Há três conceitos principais:

- Fase de pesquisa
- Escopo de pesquisa
- Critérios de pesquisa

Esta seção explica os conceitos e como você pode controlar cada um deles definindo as propriedades das âncoras.

Fases de Pesquisa

O Script procura uma sequência de âncoras em três fases:

- Inicial

- Principal
- Final

Por padrão, todas as âncoras de **Marcador** estão na fase inicial e todas as âncoras de **Conteúdo** estão na fase principal. Isso significa que o Script primeiro localiza as âncoras de `Marcador` e, em seguida, localiza as âncoras de `Conteúdo` entre elas.

Para entender isso, considere um Analisador que processa o seguinte documento de origem:

```
First name: Ron    Last name: Lehrer
```

Suponha que você tenha definido as âncoras da seguinte maneira, com propriedades de âncora padrão:

Âncora	Texto no Documento de Origem	Fase
Marcador	Nome:	Inicial
Conteúdo	Ron	Principal
Marcador	Sobrenome:	Inicial
Conteúdo	Lehrer	Principal

Na fase inicial, o Script procura as âncoras de `Marcador`:

- Ele procura `Nome:`.
- Ele procura `Sobrenome:` em uma localização após `Nome:`.

Na fase principal, o Script procura as âncoras de `Conteúdo`:

- Ele procura a âncora `Ron` em uma localização entre `Nome` e `Sobrenome:`.
- Ele procura a âncora `Lehrer` em uma localização depois de `Sobrenome:`.

Fases Aninhadas

As âncoras que têm âncoras aninhadas, como `Grupo`, têm fases aninhadas. Por exemplo, se uma âncora de `Grupo` for executada na fase principal de um Analisador, uma âncora de `Marcador` que esteja aninhada no `Grupo` será executada em uma fase inicial aninhada. A fase inicial aninhada faz parte da fase principal do Analisador, mas está antes das outras âncoras do `Grupo`.

Outro exemplo é uma âncora de `RepeatingGroup`, que pesquisa separadores e âncoras aninhadas. Para identificar as âncoras corretamente, ela procura os separadores antes de procurar as âncoras.

Escopo da Pesquisa e Critérios de Pesquisa

O exemplo acima de fases de pesquisa ilustra os conceitos de escopo de pesquisa e critérios de pesquisa. O escopo da pesquisa é a parte de um documento na qual o Script procura uma âncora. Os critérios de pesquisa são as regras com base nas quais o Script localiza a âncora dentro do escopo de pesquisa.

Na fase inicial, o Script começa procurando a âncora de `Marcador` contendo `Nome:` no início do documento. O escopo de pesquisa para essa âncora é o documento inteiro. O critério de pesquisa é que a âncora deve conter o texto `Nome:`.

O escopo de pesquisa para a âncora `Sobrenome:` começa no final de `Nome:` e se estende até o final do documento. O critério de pesquisa é que a âncora deve conter o texto `Sobrenome:`.

Na fase principal, o Analisador interpola as âncoras de **Conteúdo** entre as âncoras do **Marcador**. O escopo de pesquisa para a âncora **Ron** se estende a partir do final da âncora **Nome**: até o início da âncora **Sobrenome**: . Supondo que o Analisador use um formato delimitado por espaços, os critérios de pesquisa devem recuperar todo o texto no escopo de pesquisa depois do caractere de espaço à esquerda e antes do segundo caractere de espaço.

O escopo de pesquisa para a âncora **Lehrer** é a partir do final de **Sobrenome**: até o final do documento. Os critérios de pesquisa são semelhantes aos da âncora **Ron**.

Podemos adicionar essa análise à tabela de âncoras que apresentamos acima. Agora, a tabela descreve o método completo por meio do qual o Analisador localiza as âncoras.

Âncora	Texto no Documento de Origem	Fase	Escopo de Pesquisa	CrITÉRIOS de Pesquisa
Marcador	First name:	Initial	Documento inteiro	Texto = Nome :
Conteúdo	Ron	Main	Final de Nome: até o início de Sobrenome:	Após o espaço à esquerda Antes do próximo espaço
Marcador	Last name:	Initial	Final de Nome: até o final do documento	Texto = Sobrenome:
Conteúdo	Lehrer	Main	Final de Sobrenome: até o final do documento	Após o espaço à esquerda Antes do próximo espaço

Ajustar a Fase de Pesquisa

Ao atribuir a propriedade **phase** de uma âncora, você pode alterar a fase na qual o Script procura a âncora.

Considere o seguinte documento de origem:

```
CONTENT<10 characters>MARKER
```

Neste exemplo, a âncora **Marcador** está localizada 10 caracteres depois da âncora **Conteúdo**.

Por padrão, o Script procura o **Marcador** na fase de inicial e procura o **Conteúdo** na fase principal. Isso não funcionará aqui porque o Script só poderá localizar o **Marcador** se já tiver localizado o **Conteúdo**!

A solução é alterar a propriedade **phase** de uma das âncoras. É possível alterar o **Conteúdo** para a fase inicial ou o **Marcador** para a fase principal. Em qualquer caso, o Script localiza as âncoras.

Ajustando o Escopo de Pesquisa

Há duas maneiras de ajustar o escopo da pesquisa para uma âncora:

- Definindo a propriedade **phase** da âncora ou das âncoras ao redor
- Definindo a propriedade **marcação** da âncora ou das âncoras ao redor

Propriedade Phase

Se uma âncora de **Conteúdo** estiver entre duas âncoras de **Marcador**, por padrão, o escopo de pesquisa para o **Conteúdo** será o segmento entre as âncoras de **Marcador**.

Se você alterar todas as âncoras para a mesma fase, o escopo de pesquisa do **Conteúdo** deixará de ficar associado ao segundo **Marcador**. Ele vai do final do primeiro **Marcador** até o final do documento.

Como exemplo, considere o seguinte documento de origem:

```
Tree Fig    Date<tab>October 27, 2003 (pruned)
Tree Date Palm Date April 27, 2003<tab>(planted)
```

O exemplo supõe que o documento de origem tenha uma estrutura ampla, contendo números variáveis de espaços, tabulações ou outros símbolos intercalados no texto, e, portanto, não é fácil usar os espaços e as tabulações como delimitadores. Um exemplo como esse pode surgir na análise de documentos de processadores de texto.

Podemos analisar esse documento usando uma âncora `RepeatingGroup`, que contém âncoras de Marcador e Conteúdo aninhadas. As âncoras de Marcador são as strings `Tree` e `Date`. As âncoras de Conteúdo compreendem tudo entre as âncoras de Marcador, incluindo os espaços e as tabulações.

O problema em analisar esse documento está na segunda iteração de `RepeatingGroup`, que analisa a segunda linha. Se deixarmos as âncoras de Marcador na fase inicial, o Script concluirá incorretamente que a primeira instância da palavra `Date` é um Marcador. Na fase principal, ele não conseguirá localizar `Date Palm`, pois o escopo de pesquisa compreende as duas âncoras de Marcador, e não há texto entre elas.

Uma solução possível é mover o Marcador de `Date` para a fase principal e definir a âncora de Conteúdo, `Date Palm`, usando uma expressão que procura um nome de árvore com uma ou duas palavras. Na fase inicial do `RepeatingGroup`, o Script localiza o Marcador de `Tree`. Na fase principal, ele localiza `Date Palm` seguido do Marcador de `Date`.

Com a nova definição de fase, alteramos o escopo de pesquisa para o nome da árvore. Agora, o escopo vai de `Tree` até o final da iteração, e o Script localiza `Date Palm` com êxito.

Propriedade Marking

Considere a seguinte estrutura de documento de origem:

```
MARKER
%%%CONTENT A
^^^CONTENT B
```

Suponha que a sequência de Conteúdo A e Conteúdo B varie entre os documentos de origem. Em alguns documentos, o Conteúdo B precede o Conteúdo A.

Nesse caso, os critérios de pesquisa são:

- Tanto o Conteúdo A quanto o Conteúdo B seguem a âncora Marcador.
- O Conteúdo A começa com %%, enquanto o Conteúdo B começa com ^^.

Por padrão, o escopo de pesquisa para o Conteúdo A vai do final do Marcador até o final do documento. O escopo de pesquisa para o Conteúdo B vai do final do Conteúdo A até o final do documento. Isso não funciona porque, em alguns documentos de origem, o Conteúdo A e o Conteúdo B são invertidos.

A solução é alterar o escopo de pesquisa para o Conteúdo B. Você pode fazer isso definindo a propriedade `marking` do Conteúdo A. A propriedade `marking` especifica onde o Script insere os pontos de referência que determinam o início e o final do escopo de pesquisa.

A configuração padrão é `marking = full`, o que significa que o Script insere pontos de referência antes e depois de cada âncora. O escopo de pesquisa para o Conteúdo B começa no último ponto de referência, que é aquele depois do Conteúdo A. Como já vimos, isso resulta em uma análise incorreta.

Para impedir que o Script insira pontos de referência em torno do Conteúdo A, defina a propriedade `marking` do Conteúdo A como `none`. Como resultado, o escopo de pesquisa para o Conteúdo B começa no final do Marcador. Isso permite que o Script localize o Conteúdo B, mesmo que ele preceda o Conteúdo A.

A seguinte tabela descreve todos os quatro valores possíveis da propriedade `marking`. A coluna **Resultado** pressupõe que você atribua o valor de `marking` ao `Conteúdo A` no exemplo acima.

Propriedade Marking	Explicação	Resultado
full	O Script insere marcas de referência no início e no final da âncora atual. Esse é o comportamento padrão.	O Script procura a próxima âncora após o final da âncora atual. O <code>Conteúdo B</code> segue o <code>Conteúdo A</code> .
begin position	O Script insere uma marca de referência apenas no início da âncora atual.	O Script procura a próxima âncora após o início da âncora atual. O <code>Conteúdo B</code> substitui ou segue o <code>Conteúdo A</code> .
end position	O Script insere uma marca de referência apenas no final da âncora atual.	O Script procura a próxima âncora após o final da âncora atual. O <code>Conteúdo B</code> segue o <code>Conteúdo A</code> .
none	O Script não insere marcas de referência na âncora atual.	O Script procura a próxima âncora após o final da âncora precedente. O <code>Conteúdo B</code> segue o <code>Marcador</code> , sem levar em consideração o <code>ConteúdoA</code> .

Nota: Há algumas circunstâncias em que você deve usar uma âncora que marca um ponto de referência. Um exemplo é o separador de um `RepeatingGroup`. Se o separador não marcar, ele não fará nada. Um aviso será exibido se você tentar usar uma âncora de não marcação em uma localização na qual uma marcação é necessária.

Exemplos Online

Para obter um exemplo online da propriedade de marcação, abra o projeto `samples\Projects\Marking_Mode\Marking_Mode.cmw`. O exemplo usa a propriedade para alterar o escopo da pesquisa de uma âncora de `Conteúdo`.

Para ver outro exemplo, consulte `samples\Projects\NonMarker\NonMarker.cmw`. Esse exemplo usa a opção `marcação = nenhum`, permitindo duas âncoras de `Conteúdo` sobrepostas. O exemplo de também ilustra o uso de `direção = invertida` para pesquisar a partir do fim do escopo.

Ajustando os Critérios de Pesquisa

O Script pode procurar âncoras de acordo com vários critérios de pesquisa, por exemplo:

- De acordo com as localizações de delimitadores, que o Script detecta a partir da origem de exemplo
- De acordo com um deslocamento posicional, em outras palavras, o número de caracteres a partir de um ponto de referência
- Procurando um determinado texto
- Procurando um padrão ou uma expressão regular
- Procurando um tipo de dados especificado
- Procurando um valor de atributo

É possível combinar esses critérios de pesquisa de quase qualquer maneira. Por exemplo, você pode especificar que uma âncora de `Conteúdo` comece duas tabulações depois de uma âncora de `Marcador` e que ela tenha 10 caracteres de comprimento. Se fizer isso, você estará usando um critério de delimitador para definir o início da âncora de `Conteúdo` e um critério de deslocamento para definir o final.

Os componentes que realizam essas pesquisa são chamados de componentes pesquisadores. Para obter mais informações, consulte [“Referência de Componentes de Pesquisador” na página 245](#).

Usando Tipos de Dados para Restringir os Critérios de Pesquisa

Por padrão, além dos outros critérios de pesquisa, o Script procura uma âncora de `Conteúdo` de acordo com o tipo de dados do seu recipiente de dados.

Por exemplo, suponha que o escopo de pesquisa de uma âncora de `Conteúdo` seja a seguinte string:

```
The students' grades were 81, 56, and 95, respectively.
```

Suponha também que você não defina mais nenhum critério de pesquisa para a âncora. Se você mapear a âncora para um recipiente de dados que possui um tipo de `xs:string`, a âncora recuperará a string inteira.

Se o recipiente de dados tiver um tipo de `xs:integer`, o Script procurará a primeiro substring que corresponder ao tipo de dados. Supondo que você configure a âncora com `direction = forward`, ela recuperará o inteiro 81. Se `direction = backward`, a âncora recuperará 95.

Agora, suponha que o recipiente de dados tenha um tipo de `xs:integer`, e o esquema restrinja esse recipiente a valores menores que 60. O Script procura um inteiro que esteja em conformidade com a restrição e recupera 56.

Tipos de Dados em Combinação com Outros Critérios de Pesquisa

Você pode combinar um critério de tipo de dados com outros critérios de pesquisa. No exemplo acima, suponha que você configurou a âncora de `Conteúdo` para pesquisar a seguinte expressão regular:

```
[",.*,"]
```

A expressão procura duas vírgulas, separadas por qualquer caractere diferente de uma nova linha. A pesquisa localiza a sub-string

```
, 56,
```

Se o tipo de recipiente de dados for `xs:integer`, a âncora recuperará 56.

Tipos de Dados de Lista

Um recipiente de dados pode ser uma lista separada por espaços. O Script filtra o texto recuperado pela âncora de `Conteúdo` para que ele corresponda aos tipos dos itens de lista.

Suponha que o esquema defina um atributo denominado `grades`, que é uma lista de itens `xs:integer`. No exemplo acima, se você mapear a âncora de `Conteúdo` para `grades`, a âncora retornará uma lista de inteiros na string:

```
81 56 95
```

Se o atributo `grades` pertencer a um elemento denominado `Students`, a saída XML será:

```
<Students grades="81 56 95" />
```

Se você definir a âncora de `Conteúdo` com `direction = backward`, a lista será invertida:

```
<Students grades="95 56 81" />
```

Tipo Decimal

Se um recipiente de dados tiver o tipo `xs:decimal`, o Script assumirá que o separador de decimal é um ponto. Se a sua definição de localidade usar uma vírgula como separador de decimal, uma pesquisa de `xs:decimal` poderá falhar.

Pesquisa de Tipo com um Marcador de Fechamento

Se uma âncora de **Conteúdo** tiver uma propriedade `closing_marker`, mas não tiver uma propriedade `opening_marker`, o Script retornará a substring mais próxima a `closing_marker` que corresponder ao tipo do recipiente de dados.

No exemplo acima, se você definir a palavra `respectively` como o valor de `closing_marker`, e o recipiente de dados tiver um tipo `xs:integer`, a âncora recuperará 95.

Exemplo Online

Para ver um exemplo online de pesquisa por um tipo de dados, abra o projeto `samples\Projects\Pattern\Pattern.cmw`. O exemplo é um Analisador que contém uma única âncora de **Conteúdo** que está Mapeada para um elemento XML. O esquema usa um `xs:pattern` para restringir o elemento a determinadas sequências de caracteres. A âncora retorna a parte do documento de origem que corresponde ao padrão.

Desabilitando a Pesquisa de Tipo de Dados

É possível desabilitar a pesquisa de tipo de dados selecionando a propriedade **`disable_XSD_type_search`** da âncora de **Conteúdo**. Se você fizer isso, a âncora pesquisará de acordo com outros critérios, sem levar em consideração o tipo do recipiente de dados.

Se o resultado não tiver o tipo de apropriado, ele não poderá ser armazenado no recipiente de dados, e a âncora falhará. Você pode usar transformadores para converter o resultado no tipo adequado e impedir a falha. Para obter mais informações, consulte [“Visão Geral de Transformadores” na página 257](#).

Por exemplo, suponha que o documento de origem contenha uma data no formato `dd-mm-aaaa` e que você queira armazenar essa data em um recipiente de dados `xs:date`. É possível lidar com essa situação da seguinte maneira:

1. Defina uma âncora de **Conteúdo** que recupera a data `dd-mm-aaaa`, ignorando a incompatibilidade com o tipo `xs:date`.
2. Configure a âncora com um transformador **`DateFormatICU`** que converte o resultado em `xs:date`.

Âncoras que Contêm Âncoras Aninhadas

Uma pergunta interessante é como um Analisador busca uma âncora que tem âncoras aninhadas, como uma âncora de **Grupo**.

O Script não procura um **Grupo** e, em seguida, as âncoras aninhadas. Em vez disso, ele procura essas âncoras aninhadas. A extensão do **Grupo** é definida pelas âncoras aninhadas que o Script localiza.

Por exemplo, suponha que um Analisador tenha a sequência de âncoras a seguir. Partimos do princípio de que as âncoras possuem propriedades `phase`, `marking` e `optional` padrão.

```
Marker A
Group
  Marker B
  Content C
  Marker D
Marker E
```

Primeiro, o Script procurará **Marcador A** e **Marcador E**. O escopo de pesquisa do **Grupo** é a região entre o **Marcador A** e o **Marcador E**.

Em seguida, dentro do escopo de pesquisa do **Grupo**, o Script procura o **Marcador B** e o **Marcador D**. A região entre essas âncoras de **Marcador E** é o escopo de pesquisa para o **Conteúdo C**.

Dentro do último escopo de pesquisa, o Script procura o **Conteúdo C**.

Referência do Componente de Âncora

Os componentes de **Âncora** indicam as localizações dos dados nos documentos de origem.

Alternativas

A âncora de **Alternativas** define um conjunto de âncoras aninhadas alternativas. É possível definir um critério para selecionar um alternativa no conjunto.

A seguinte tabela descreve as propriedades da âncora de **Alternativas**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
marking	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- begin position. Insere um ponto de referência antes da âncora atual.- end position. Insere um ponto de referência depois da âncora atual.- full. Insere um ponto de referência antes e depois da âncora atual.- none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
selector	<p>Determina o critério para a seleção de uma âncora entre as âncoras aninhadas abaixo da âncora de Alternativas. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - ScriptOrder. O Analisador testa as âncoras aninhadas na sequência definida no Script. Ele aceita a primeira âncora bem-sucedida. Se todas as âncoras falharem, a âncora de Alternativas também falhará. - DocumentOrder. O Analisador testa todas as âncoras aninhadas. Ele aceita a primeira ou a última âncora bem-sucedida, de acordo com as localizações das âncoras no documento de origem, conforme determinado na propriedade select. Se todas as âncoras falharem, a âncora de Alternativas também falhará. - NameSwitch. O Analisador procura a âncora cuja propriedade nome está especificada no recipiente de dados definido em option_name. Ele ignora as outras âncoras. Se a âncora aninhada nomeada falhar, a âncora de Alternativas também falhará.

Exemplo

Você está analisando um documento no qual uma data pode aparecer em qualquer um dos seguintes padrões:

```
21/10/03
October 21, 2003
```

Para processar essa conteúdo, você pode definir uma âncora de `Alternativas` que contém duas âncoras de `Conteúdo` que armazenam a saída em diferentes elementos XML. Cada elemento XML está restrito a aceitar um dos padrões de data. A âncora de `Alternativas` está configurada com `selector = ScriptOrder`.

Quando o Analisador executa a âncora de `Alternativas`, ele testa a primeira âncora de `Conteúdo`. Se a data corresponder ao padrão da primeira âncora, a primeira âncora de `Conteúdo` obterá êxito. Se a data não corresponder ao padrão, a primeira âncora de `Conteúdo` falhará e a âncora de `Alternativas` testará a segunda âncora de `Conteúdo`. Desta maneira, o Analisador pode processar ambos os padrões de data.

Como Definir uma Âncora de Alternativas

Adicione uma âncora de **Alternativas** editando o Script. Aninhadas dentro da âncora de **Alternativas**, adicione as âncoras alternativas.

Usando Alternativas para Selecionar um Analisador Secundário

Você pode usar uma âncora `Alternativas` para controlar quais dos diversos analisadores secundários processam um documento. O Analisador principal pode usar esse recurso para processar os documentos de origem de vários tipos.

Por exemplo, suponha que a página inicial de um site de jornal tem links para artigos. Após cada link, o artigo é rotulado como `Notícias`, `Economia` ou `Esportes`. É desejável analisar os artigos usando um Analisador diferente para cada tipo, como por exemplo:

```
<a href="PrincessWeds.html">Norwegian Princess Weds</a> - News
<a href="BanksMerge.html">Local Banks to Merge</a> - Business
<a href="HomeTeamWins.html">Bears Trounce Antelopes</a> - Sports
```

Você pode dar suporte a esta situação da seguinte maneira:

1. O Analisador principal recupera o nome de arquivo de um artigo e o armazena em uma variável.
2. O Analisador principal contém uma âncora de `Alternativas` que está configurada com a opção `DocumentOrder`.

3. A âncora `Alternativas` contém âncoras `Grupo` aninhadas.
4. Cada âncora `Grupo` está configurada com uma âncora `Marcador` e uma ação `RunParser`, da seguinte maneira:
 - O primeiro `Grupo` contém um `Marcador` que pesquisa pela string `Notícias`. O `Grupo` está configurado com uma ação `RunParser` que executa um Analisador secundário denominado `NewsParser`.
 - O segundo `Grupo` contém um `Marcador` que pesquisa por `Economia` e executa `BusinessParser`.
 - O terceiro `Grupo` contém um `Marcador` que pesquisa por `Esportes` e executa `SportsParser`.

A âncora `Alternativas` testa todas as três âncoras `Grupo`. Ele aceita o `Grupo` que contém o primeiro `Marcador` que ocorre após o nome de arquivo. O `Grupo` executa o Analisador apropriado no arquivo.

Exemplo Online

Para ver um exemplo online dessa âncora, abra o projeto `samples\Projects\Alternatives\Alternatives.cmw`. O exemplo usa as âncoras `Alternativas` para analisar um nome diferente e formatos de data que podem existir em um documento de origem.

Conteúdo

Uma âncora de **Conteúdo** recupera texto do documento de origem. O Analisador pesquisa em uma região definida de acordo com os critérios de pesquisa especificados e armazena o texto recuperado em um recipiente de dados.

A seguinte tabela descreve as propriedades da âncora de **Conteúdo**:

Propriedade	Descrição
<code>allow_empty_values</code>	Determina se a âncora de Conteúdo âncora pode estar vazia. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. A propriedade <code>data_holder</code> está atribuída com um valor vazio.- Limpo. Valores vazios não são permitidos. A propriedade <code>allow_empty_values</code> deve ser selecionada nas seguintes situações: <ul style="list-style-type: none">- Quando a âncora está configurada com <code>value = LearnByExample</code> e não há nada entre os delimitadores.- Quando não há nada entre <code>opening_marker</code> e <code>closing_marker</code>.
<code>closing_marker</code>	Define o final de uma região na qual o Analisador procura pela âncora de Conteúdo . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- <code>NewlineSearch</code>. O final da âncora de Conteúdo é o próximo caractere de nova linha.- <code>OffsetSearch</code>. O final da âncora de Conteúdo é o número de caracteres especificado em <code>offset</code>.- <code>PatternSearch</code>. O final da âncora de Conteúdo é o primeiro texto que corresponde a uma determinada expressão regular.- <code>TextSearch</code>. O final da âncora de Conteúdo é uma string de texto especificada.
<code>data_holder</code>	Define um recipiente de dados no qual a âncora de Conteúdo armazena o texto recuperado.

Propriedade	Descrição
direction	<p>Uma direção de pesquisa para a âncora dentro do escopo de pesquisa. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - backward. Pesquisa a partir do final do escopo de pesquisa e localiza a última instância da âncora. - forward. Pesquisa a partir do início do escopo de pesquisa e localiza a primeira instância da âncora. <p>Para uma âncora de Marcador, é possível modificar esse comportamento usando a propriedade count. Por exemplo, se direction = backward e count = 2, o Script localizará a penúltima instância.</p> <p>O padrão é forward. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209.</p>
disable_XSD_type_search	<p>Determina se o analisador procura pelo conteúdo que corresponde ao tipo de dados do recipiente de dados. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Analisador pesquisa sem levar em consideração o tipo de dados. Depois que transformadores forem aplicados ao conteúdo, se o resultado não corresponder ao tipo de dados do recipiente de dados, a âncora falhará. - Limpo. O Analisador procura pelo conteúdo que corresponde ao tipo de dados. <p>O padrão é limpo. Para obter mais informações, consulte “Usando Tipos de Dados para Restringir os Critérios de Pesquisa” na página 214.</p>
disabled	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
ignore_default_transformers	<p>Determina se o Analisador aplica os transformadores padrão ao conteúdo. O padrão é limpo.</p> <p>Para obter mais informações, consulte “Visão Geral de Transformadores” na página 257.</p>
marking	<p>Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - begin position. Insere um ponto de referência antes da âncora atual. - end position. Insere um ponto de referência depois da âncora atual. - full. Insere um ponto de referência antes e depois da âncora atual. - none. Não cria um ponto de referência. <p>Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209.</p>
nome	<p>Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos. Use a propriedade name para identificar o componente que causou o evento.</p>
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399.</p>

Propriedade	Descrição
opening_marker	Define o início de uma região na qual o Analisador procura pela âncora de Conteúdo . Os valores possíveis são os seguintes componentes: <ul style="list-style-type: none"> - NewlineSearch. O início da âncora de Conteúdo é o próximo caractere de nova linha. - OffsetSearch. O início da âncora de Conteúdo é o número de caracteres especificado em offset. - PatternSearch. O início da âncora de Conteúdo é o primeiro texto que corresponde a uma determinada expressão regular. - TextSearch. O início da âncora de Conteúdo é uma string de texto especificada.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma sequência de transformadores que o Analisador aplica ao texto recuperado. Para obter mais informações, Capítulo 17, "Transformadores" na página 257 .
validadores	Define uma lista de validadores aplicados aos dados. Para obter mais informações, consulte "Validadores" na página 402 .
valor	Define critérios para uma pesquisa na região definida pelos atributos opening_marker e closing_marker . Se o atributo opening_marker não estiver definido, a pesquisa ocorrerá entre os pontos de referência circundantes. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Vazio. A âncora de Conteúdo recupera todo o escopo da pesquisa. - AttributeSearch. A âncora de Conteúdo recupera o valor de uma expressão do tipo AttributeName=.... Use essa opção para recuperar valores de atributo de um documento de origem XML ou HTML. - LearnByExample. O Analisador aprende o texto a ser recuperado de acordo com o formato do Analisador e a origem do exemplo. Por exemplo, se o Analisador tiver um formato delimitado por tabulação, ele contará o número de guias desde o início do escopo de pesquisa até o texto de exemplo. Ele recupera o texto entre as tabulações correspondentes no documento de origem. - PatternSearch. A âncora de Conteúdo recupera o primeiro texto que corresponde a uma determinada expressão regular. - TypeSearch. A âncora de Conteúdo recupera o primeiro texto que corresponde a um determinado tipo de dados. O padrão é vazio. Para obter mais informações sobre essas opções, consulte o documento "Referência de Componentes de Pesquisador" na página 245 . Além dos componentes de pesquisador, o Analisador usa o tipo de dados do data_holder como um critério de pesquisa. Para obter mais informações, consulte "Usando Tipos de Dados para Restringir os Critérios de Pesquisa" na página 214 .

As propriedades **opening_marker** e **closing_marker** são equivalentes a âncoras de **Marcador** em um componente de **Grupo**.

- Uma âncora de **Conteúdo** com **opening_marker** definido é como um componente de **Grupo** com a seguinte sequência de âncoras:
 1. Marcador
 2. Conteúdo
- Uma âncora de **Conteúdo** com **closing_marker** definido é como um componente de **Grupo** com a seguinte sequência de âncoras:
 1. Conteúdo
 2. Marcador
- Uma âncora de **Conteúdo** com **opening_marker** e **closing_marker** definidos é como um componente de **Grupo** com a seguinte sequência de âncoras:
 1. Marcador
 2. Conteúdo
 3. Marcador

Para obter mais informações, consulte o documento [“Referência de Componentes de Pesquisador” na página 245](#).

Direção da Pesquisa

A propriedade `direction` tem vários efeitos em uma âncora de **Conteúdo**. Se `direction = backward`:

- O Script pesquisará para trás a partir do final do escopo de pesquisa em busca de `opening_marker` e `closing_marker`. `Opening_marker` ainda precede `closing_marker`.
- O componente pesquisador pesquisa para trás a partir do final do escopo de pesquisa.
- Se o componente pesquisador for `LearnByExample`, ele contará os delimitadores para trás a partir do final do escopo de pesquisa.

Exemplo Online

Para ver um exemplo online das âncoras de **Conteúdo**, abra o projeto `samples\Projects\Content\Content.cmw`. O exemplo ilustra vários usos das propriedades `opening_marker`, `closing_marker` e valor para configurar âncoras de **Conteúdo**.

DelimitedSections

A âncora **DelimitedSections** analisa os dados que divididos em seções por um separador. Ela define um grupo de âncoras aninhadas. Cada âncora analisa uma única seção.

A tabela a seguir descreve as propriedades da âncora de **DelimitedSections**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
marcação	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - begin position. Insere um ponto de referência antes da âncora atual. - end position. Insere um ponto de referência depois da âncora atual. - full. Insere um ponto de referência antes e depois da âncora atual. - none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
separador	Define uma âncora que delimita as seções.

Propriedade	Descrição
separator_position	<p>Define o posicionamento do separador com relação às seções. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - após. Há um separador após cada seção, incluindo a última seção. Por exemplo, 1 2 3 4 - ao redor. Há separadores antes e depois de cada seção, incluindo a primeira e a última seção. Por exemplo, 1 2 3 4 - antes. Há um separador antes de cada seção, incluindo a primeira seção. Por exemplo, 1 2 3 4 - entre. Há um separador entre as seções sucessivas, mas não antes da primeira seção e não após a última seção. Por exemplo, 1 2 3 4
using_placeholders	<p>Determina quando a âncora DelimitedSections grava o separador de uma seção opcional que está ausente no documento de origem. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - sempre. O separador de uma seção ausente sempre existe. Por exemplo, 1 3 - nunca. O separador de uma seção ausente nunca existe. Por exemplo, 1 3 - quando necessário. O separador de uma seção interna sempre existe. O separador de uma seção final nunca existe. Por exemplo, 1 3 <p>Nesses exemplos, separator_position está definido como antes e as seções 2 e 4 estão ausentes.</p>

Exemplo

Um formulário de resumo de funcionário contém várias seções, cada qual é precedida por uma linha de hífen:

```

-----
Jane Palmer
Employee ID 123456
-----
Professional Experience
...
-----
Education
...

```

Você pode definir a região seccionada como uma âncora de `DelimitedSections`, com a linha de hífen como o separador. Como a linha de hífen precede cada seção, defina o `separator_position` como `antes`.

Dentro da âncora `DelimitedSections`, aninhe três âncoras de `Group`. O primeiro `Group` analisa a seção `Jane Palmer`, o segundo `Group` analisa a seção `Professional Experience`, e assim por diante.

Seções Opcionais

No exemplo acima, suponha que a segunda seção, `Professional Experience` esteja ausente de alguns documentos de origem. O seu separador, a linha de hífen, está sempre presente.

```

-----
Jane Palmer
Employee ID 123456
-----
-----
Education
...

```

Para lidar com essa situação, configure o `DelimitedSections` da seguinte maneira:

- Na segunda âncora de `Group`, selecione a propriedade `opcional`. Isso significa que se o `Group` falhar, ele não deverá fazer com que o `DelimitedSections` falhe.
- Na âncora `DelimitedSections`, defina `using_placeholders = sempre`. Isso significa que a âncora procura o separador de uma seção opcional, mesmo se a própria seção estiver ausente.

Agora suponha que se a seção `Professional Experience` estiver ausente, seu separador também estará ausente.

```
-----  
Jane Palmer  
Employee ID 123456  
-----  
Education  
...
```

Nesse caso, configure o `DelimitedSections` da seguinte forma:

- Na segunda âncora de `Group`, selecione a propriedade `opcional`.
- Na âncora `DelimitedSections`, defina `using_placeholders = nunca`. Isso significa a âncora não deve procurar o separador de uma seção ausente.

Como Definir uma Âncora `DelimitedSections`

Adicione uma âncora **`DelimitedSections`** editando o Script no editor do IntelliScript. Na âncora **`DelimitedSections`**, adicione uma sequência de âncoras que analisam as seções.

Exemplo Online

Para ver um exemplo online dessa âncora, abra o projeto `samples\Projects\DelimitedSections\DelimitedSections.cmw`. O exemplo ilustra uma âncora de `DelimitedSections` que analisa seções separadas por um símbolo `|`. Cada seção é analisada por uma única âncora de `Conteúdo`.

EmbeddedParser

A âncora **`EmbeddedParser`** usa um Analisador secundário para analisar seu escopo de pesquisa. Ela pode chamar-se a si própria recursivamente.

A tabela a seguir descreve as propriedades da âncora de **`EmbeddedParser`**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .
marcação	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- begin position. Insere um ponto de referência antes da âncora atual.- end position. Insere um ponto de referência depois da âncora atual.- full. Insere um ponto de referência antes e depois da âncora atual.- none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .

Propriedade	Descrição
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
Analizador	Determina o nome de um Analisador secundário que está definido no mesmo projeto.
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
schema_connections	Define uma lista de subcomponentes Conectar que definem a relação entre os recipientes de dados na saída do Analisador principal e do Analisador secundário. Para obter mais informações, consulte "Conectar" na página 252 .
source_transformers	Define uma sequência de transformadores que o Analisador aplica ao escopo de pesquisa antes do Analisador secundário processá-la.

Exemplo

Um documento é delimitado por tabulação, exceto por uma seção que é delimitada por vírgula.

Para analisar o documento, você pode definir um Analisador principal que usa o formato `TabDelimited`.

Defina outro Analisador que use o formato `CommaDelimited`. Use uma âncora `EmbeddedParser` para executar o segundo Analisador na execução do primeiro Analisador.

Exemplo Online

Para ver um exemplo online dessa âncora, abra o projeto `samples\Projects\EmbeddedParser\EmbeddedParser.cmw`. O exemplo usa um Analisador principal para determinar a localização de um endereço. Ele executa um `EmbeddedParser` para analisar o endereço.

EnclosedGroup

A âncora **EnclosedGroup** define uma região limitada que contém âncoras aninhadas. Os limites são especificados por âncoras de **abertura** e de **fechamento**. No caso de limites aninhados, como parênteses ou marcas HTML, **EnclosedGroup** localiza os limites correspondentes.

A seguinte tabela descreve as propriedades da âncora **EnclosedGroup**:

Propriedade	Descrição
fechando	Define a âncora de fechamento de EnclosedGroup .
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
marking	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- begin position. Insere um ponto de referência antes da âncora atual.- end position. Insere um ponto de referência depois da âncora atual.- full. Insere um ponto de referência antes e depois da âncora atual.- none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
no_initial_phase	Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais.- Selected. Procura âncoras aninhadas na fase principal. O padrão é Desmarcado.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
abrindo	Define a âncora de abertura de EnclosedGroup .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
origem	Define uma sequência de recipientes de dados para entrada em EnclosedGroup . Cada recipiente de dados é identificado por uma das seguintes propriedades: <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração acessa uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. Use a propriedade source quando EnclosedGroup for chamado por outro componente. Para obter mais informações, consulte “Propriedade de Origem” na página 366 .
destino	Define uma sequência de recipientes de dados para saída a partir de EnclosedGroup . Se um recipiente de dados ainda não existir, ele será criado. Cada recipiente de dados é identificado por uma das seguintes propriedades: <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração cria uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. Use a propriedade target quando a saída de EnclosedGroup for usada por outro componente. Para obter mais informações, consulte “Propriedade de Destino” na página 369 .

Um **EnclosedGroup** é semelhante a uma âncora de **Conteúdo** âncora com um **opening_marker** e um **closing_marker**. No entanto:

- A âncora de **Conteúdo** recupera todo o conteúdo entre a abertura e o fechamento, sem análise adicional.
- **EnclosedGroup** permite que você analise adicionalmente o conteúdo entre âncoras de **abertura** e **fechamento**.

Exemplo

Você pode definir uma tabela HTML como um **EnclosedGroup**, com as marcas `<table>` e `</table>` como abertura e fechamento. As âncoras analisarão o conteúdo da tabela.

Suponha que o elemento `<table>` contém um elemento `<table>` aninhado. Em outras palavras, uma tabela está aninhadas dentro de uma célula de tabela. A âncora **EnclosedGroup** fará a correspondência da marca `<table>` pai com a marca `</table>` pai. Ela não fará a correspondência da marca `<table>` pai com a marca `</table>` aninhada, que seria uma identificação errada da tabela.

Como Definir uma Âncora EnclosedGroup

É possível definir uma âncora **EnclosedGroup** editando o Script no editor do IntelliScript. Adicione as âncoras aninhadas que analisam o conteúdo.

ExtractRecord

A âncora **ExtractRecord** extrai um registro, atribui os identificadores ao registro e transmite o registro aos subelementos de um **StructureDefinition**. **ExtractRecord** é usada na propriedade **format_definition** de um **StructureDefinition**.

ExtractRecord extrai todo o seu escopo de pesquisa. Por exemplo, se você inserir um **ExtractRecord** entre duas âncoras de **Marcador**, ele extrairá o escopo entre os marcadores.

A tabela a seguir descreve as propriedades da âncora de **ExtractRecord**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
ids	Define uma lista de identificadores anexados ao registro. StructureDefinition usa os identificadores para fazer a correspondência do registro com um subelemento. Em cada entrada da lista, insira um identificador de valor ou procure para um recipiente de dados que contém o valor.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

FindReplaceAnchor

A âncora **FindReplaceAnchor** marca o texto de origem e especifica o texto de substituição para transformação pelo transformador **TransformByParser**.

A seguinte tabela descreve as propriedades da âncora **FindReplaceAnchor**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
marking	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - begin position. Insere um ponto de referência antes da âncora atual. - end position. Insere um ponto de referência depois da âncora atual. - full. Insere um ponto de referência antes e depois da âncora atual. - none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
no_initial_phase	Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais. - Selected. Procura âncoras aninhadas na fase principal. O padrão é Desmarcado.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
on_partial_match	Determina o comportamento quando FindReplaceAnchor não encontra todas as suas âncoras aninhadas não opcionais. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - fail. FindReplaceAnchor falha. Padrão. - skip. FindReplaceAnchor remove de seu escopo de pesquisa a área gerada pelas âncoras aninhadas bem-sucedidas e tenta localizar novamente todas as âncoras aninhadas. Ele repetirá esse processo até localizar as âncoras ou falhar.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
replace_with	Define uma string de substituição literal ou um recipiente de dados que contém essa string de substituição.
origem	<p>Define uma sequência de recipientes de dados para entrada em FindReplaceAnchor. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração acessa uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade source quando FindReplaceAnchor for chamado por outro componente. Para obter mais informações, consulte "Propriedade de Origem" na página 366.</p>
destino	<p>Define uma sequência de recipientes de dados para saída a partir de FindReplaceAnchor. Se um recipiente de dados ainda não existir, ele será criado. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração cria uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade target quando a saída de FindReplaceAnchor for usada por outro componente. Para obter mais informações, consulte "Propriedade de Destino" na página 369.</p>

Se **FindReplaceAnchor** não contiver âncoras aninhadas, ele marcará todo o texto dentro de seu escopo de pesquisa. Por exemplo, se **FindReplaceAnchor** estiver entre duas âncoras de **Marcador**, ele marcará o texto entre elas.

Se **FindReplaceAnchor** contiver uma âncora de **Marcador**, ele marcará o **Marcador** para substituição.

Se **FindReplaceAnchor** contiver duas âncoras de **Marcador**, ele marcará essas âncoras de **Marcador** e o segmento entre elas para substituição.

O texto de substituição pode ser uma string de substituição estática ou uma string recuperada dinamicamente do documento de origem.

Para obter mais informações, consulte ["TransformByParser" na página 290](#).

Exemplo

Você deseja adicionar números de linha a um documento de texto. É possível adicionar os números de linha com base na seguinte abordagem:

1. Crie um Analisador e adicione um `RepeatingGroup` a ele.
2. No `RepeatingGroup`, adicione um `FindReplaceAnchor`.
3. No `FindReplaceAnchor`, adicione uma âncora de **Marcador** e defina sua propriedade `search` como `NewlineSearch`.
Isso faz com que `FindReplaceAnchor` marque cada nova linha no documento.
4. Configure o `RepeatingGroup` de forma a armazenar sua propriedade `current_iteration` em uma variável. Defina a propriedade `replace_with` de `FindReplaceAnchor` como a variável.
5. No nível global do Script, defina um transformador `TransformByParser`. Defina sua propriedade `analisador` para o Analisador.

6. Defina `TransformByParser` como o componente de inicialização da transformação.

O transformador gera a saída de uma versão modificada do arquivo original que contém números de linha.

Como Definir uma âncora `FindReplaceAnchor`

É possível definir uma âncora **FindReplaceAnchor** editando o Script no editor do IntelliScript. Se necessário, adicione âncoras aninhadas marcando uma substring a ser substituída.

Grupo

A âncora de **Grupo** une uma sequência de âncoras e ações.

As propriedades do **Grupo** se aplicam a todos os componentes filho. Use um **Grupo** para definir operações a serem realizadas pelo Script em um conjunto de âncoras ou para controlar a fase das âncoras aninhadas.

A seguinte tabela descreve as propriedades da âncora de **Grupo**:

Propriedade	Descrição
absent	Define o comportamento da âncora de Grupo quando uma de suas ações ou âncoras aninhadas não opcionais falha. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Grupo falha.- Limpo. Comportamento normal. Use esse recurso para testar a ausência de âncoras aninhadas.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
marking	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- begin position. Insere um ponto de referência antes da âncora atual.- end position. Insere um ponto de referência depois da âncora atual.- full. Insere um ponto de referência antes e depois da âncora atual.- none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
no_initial_phase	Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais.- Selected. Procura âncoras aninhadas na fase principal. O padrão é Desmarcado.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
on_partial_match	<p>Determina o comportamento quando o Grupo não encontra todas as suas âncoras aninhadas não opcionais. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - fail. O Grupo falha. Padrão. - skip. O Grupo remove de seu escopo de pesquisa a área gerada pelas âncoras aninhadas bem-sucedidas e tenta localizar novamente todas as âncoras aninhadas. Ele repetirá esse processo até localizar as âncoras ou falhar.
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
phase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é main.</p>
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
search_order	<p>Define a direção de processamento de âncoras aninhadas. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - top-down. As âncoras aninhadas são processadas na sequência definida no Script. - bottom-up. As âncoras aninhadas são processadas em ordem inversa. Use essa opção quando os dados de uma âncora posterior afetarem o processamento de uma âncora anterior.

Propriedade	Descrição
origem	<p>Define uma sequência de recipientes de dados para entrada no Grupo. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração acessa uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade source quando o Grupo for chamado por outro componente. Para obter mais informações, consulte "Propriedade de Origem" na página 366.</p>
destino	<p>Define uma sequência de recipientes de dados para saída a partir de Grupo. Se um recipiente de dados ainda não existir, ele será criado. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração cria uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade target quando a saída de Grupo for usada por outro componente. Para obter mais informações, consulte "Propriedade de Destino" na página 369.</p>

Como Definir uma Âncora de Grupo

É possível definir uma âncora de **Grupo** editando o Script no editor do IntelliScript. Adicione âncoras aninhadas e ações que analisam o conteúdo do **Grupo**.

Grupo Opcional

É possível usar a propriedade **optional** de um **Grupo** para impedir que o Script tente recuperar texto de uma seção ausente de um documento.

Por exemplo, para analisar a origem

```
First name: Ron
```

you pode definir **Nome:** como um **Marcador** e **Ron** como o **Conteúdo**. Se alguns documentos de origem não contiverem os dados de nome, você poderá colocar o **Marcador** e o **Conteúdo** em um **Grupo** e torná-lo opcional. Se o **Nome:** não for encontrado, o **Grupo** falhará imediatamente e o Analisador não procurará a âncora de **Conteúdo**.

Há uma diferença entre tornar o **Grupo** opcional e tornar suas âncoras aninhadas opcionais. Se você tornar tanto o **Marcador** quanto o **Conteúdo** opcionais, em vez do **Grupo**, o Script ignorará a falha do **Marcador** e procurará o **Conteúdo**. Isso pode resultar na recuperação de um texto irrelevante.

Exemplo Online

Para ver um exemplo online dessa âncora, abra o projeto `samples\Projects\persistent_search\persistent_search.cmw`.

O exemplo ilustra um **Grupo** que está configurado com a propriedade `on_partial_match = skip`. O **Grupo** contém duas âncoras de **Marcador**:

- O primeiro **Marcador** procura o texto **A**.
- O segundo **Marcador** pesquisa uma string que contém qualquer número de caracteres *****. Ele tem a propriedade `adjacente`, o que significa que deve estar adjacente ao primeiro **Marcador**.

Na primeira passagem, o Grupo localiza um caractere A no início do documento de origem. No entanto, ele não localiza o segundo Marcador adjacente ao caractere A.

O Grupo reduz o escopo da pesquisa eliminando o primeiro caractere A e pesquisa novamente as duas âncoras de Marcador adjacentes. Ele continua esse procedimento até encontrar uma string A* com êxito, que contém as âncoras de Marcador adjacentes.

Você pode observar o comportamento no log de eventos. O log registrou que o Grupo falhou nas duas primeiras tentativas e obteve êxito na terceira.

Tente experimentar com as configurações `on_partial_match` e `adjacente`. Você pode visualizar o efeito na codificação de cor do exemplo de origem.

Também você pode tentar executar o exemplo, embora o arquivo de resultado esteja vazio porque o Analisador não contém âncoras de Conteúdo. Se você definir `on_partial_match = fail`, poderá observar no log de eventos que o Analisador falhou porque o Grupo não conseguiu localizar as âncoras adjacentes.

Marcador

Uma âncora de **Marcador** define uma localização em um documento de origem. Ela é usada como um ponto de referência a partir do qual o Script procura as âncoras seguintes.

Por padrão, a propriedade **phase** de um **Marcador** é , o que significa que o Script examina um documento em busca de âncoras de **Marcador** antes de procurar âncoras de **Conteúdo**. Para obter mais informações, consulte [“Como um Analisador Procura Âncoras” na página 209](#).

A seguinte tabela descreve as propriedades da âncora de **Marcador**:

Propriedade	Descrição
absent	Determina se o texto ou padrão especificado está ausente no documento. A propriedade absent tem as seguintes opções: <ul style="list-style-type: none">- Selected. Se o texto especificado aparecer no documento, Marcador falhará.- Cleared. Se o texto especificado aparecer no documento, Marcador terá êxito. O padrão é Desmarcado.
adjacent	Se for selecionado, o Marcador deverá ser adjacente à âncora no início de seu escopo de pesquisa. Se a propriedade direction for definida como , o Marcador deverá ser adjacente à âncora no final de seu escopo de pesquisa. Se não for selecionado, o Script poderá ignorar o texto até localizar o Marcador . A propriedade adjacent tem as seguintes opções: <ul style="list-style-type: none">- Selected. O Marcador deve ocorrer logo após o início do escopo de pesquisa quando a propriedade direction está definida como <code>forward</code> ou logo antes do final do escopo de pesquisa quando direction está definida como <code>backward</code>.- Cleared. O Marcador pode ocorrer em qualquer parte dentro do escopo de pesquisa. O padrão é Desmarcado.
count	Define o número de ocorrência a ser localizado. Por exemplo, para definir o Marcador na segunda nova linha após a âncora precedente, defina search como <code>NewlineSearch</code> e count como 2.
direction	Uma direção de pesquisa para a âncora dentro do escopo de pesquisa. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- backward. Pesquisa a partir do final do escopo de pesquisa e localiza a última instância da âncora.- forward. Pesquisa a partir do início do escopo de pesquisa e localiza a primeira instância da âncora. Para uma âncora de Marcador , é possível modificar esse comportamento usando a propriedade count . Por exemplo, se direction = backward e count = 2 , o Script localizará a penúltima instância. O padrão é forward. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 .

Propriedade	Descrição
disabled	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
marking	<p>Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - begin position. Insere um ponto de referência antes da âncora atual. - end position. Insere um ponto de referência depois da âncora atual. - full. Insere um ponto de referência antes e depois da âncora atual. - none. Não cria um ponto de referência. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p>
name	<p>Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos. Use a propriedade name para identificar o componente que causou o evento.</p>
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
phase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é initial.</p>

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
search	<p>Define os critérios de pesquisa para o Marcador. Os critérios de pesquisa determinam onde o Marcador está localizado dentro do escopo de pesquisa. Por exemplo, NewlineSearch localiza o Marcador em um caractere de nova linha. Um TextSearch localiza o Marcador em uma string especificada. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p> <p>O valor dessa propriedade é um dos seguintes componentes pesquisadores:</p> <ul style="list-style-type: none"> - NewlineSearch. Procura um caractere de nova linha. - TextSearch. Procura uma string de texto predefinida ou uma string de texto armazenada em um recipiente de dados. - PatternSearch. Procura uma string que corresponde a uma determinada expressão regular. - OffsetSearch. Ignora um número predefinido de caracteres após o ponto de referência anterior ou um número de caracteres que está armazenado em um recipiente de dados. O Marcador é o ponto após os caracteres ignorados. - TypeSearch. Procura uma string que esteja em conformidade com um tipo de dados especificado. <p>Para obter mais informações, consulte a "Referência de Componentes de Pesquisador" na página 245.</p>

Como Definir uma Âncora de Marcador

É possível definir uma âncora de **Marcador** editando o Script no editor do IntelliScript. Para obter mais informações, consulte ["Definindo Âncoras" na página 206](#).

Exemplo Online

Na pasta de Exemplos Online, abra `Projects\Markers\Markers.cmw`. O exemplo demonstra as âncoras de **Marcador** que pesquisam:

- Uma string de texto predefinido
- Um caractere de nova linha
- Um deslocamento
- Um tipo de dados
- Uma expressão regular

Se você executar o Analisador, observe que o arquivo de resultado estará vazio porque a configuração não tem âncoras de `Conteúdo`.

RepeatingGroup

A âncora **RepeatingGroup** analisa uma região que contém segmentos repetitivos. Cada segmento é chamado de iteração e pode ser delimitado por um **separador**. O **RepeatingGroup** contém uma sequência de âncoras aninhadas e de ações que analisam cada iteração da mesma maneira.

A seguinte tabela descreve as propriedades da âncora **RepeatingGroup**:

Propriedade	Descrição
count	Define um número ou um recipiente de dados que contém o número de iterações a serem executadas. Se estiver em branco, as iterações continuarão até o esgotamento do escopo de pesquisa. Se count for 0, o RepeatingGroup não procurará iterações. Nesse caso, RepeatingGroup terá êxito, mas não produzirá nenhuma saída.
current_iteration	Define um recipiente de dados no qual o RepeatingGroup gera o número da iteração atual.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
iteration_order	Define a ordem na qual as iterações são processadas. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - top-down. As iterações são processadas na sequência definida no Script. - bottom-up. As iterações são processadas em ordem inversa. Use essa opção se os dados de uma iteração posterior afetarem a maneira como você processa uma iteração anterior.
marking	Determina se uma âncora é usada como o início do escopo de pesquisa da âncora seguinte. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - begin position. Insere um ponto de referência antes da âncora atual. - end position. Insere um ponto de referência depois da âncora atual. - full. Insere um ponto de referência antes e depois da âncora atual. - none. Não cria um ponto de referência. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
no_initial_phase	Determina se o script procura âncoras aninhadas na fase principal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Procura âncoras aninhadas de acordo com as suas propriedades individuais. - Selected. Procura âncoras aninhadas na fase principal. O padrão é Desmarcado.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
on_iteration_fail	<p>Define a ação quando uma única iteração falha. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Limpo. Nenhuma ação. - CustomLog. Grava no log do usuário. - LogError. Grava uma mensagem de erro no log do Mecanismo. - LogInfo. Grava uma mensagem de informações no log do Mecanismo. - LogWarning. Grava uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Dispara uma notificação. <p>Use a propriedade on_fail para gravar uma entrada se todo o RepeatingGroup falhar. Para obter mais informações, consulte “Tratamento de Falhas” na página 399.</p>
on_partial_match	<p>Define o comportamento quando algumas das âncoras necessárias aninhadas em RepeatingGroup, mas não todas, aparecem na entrada. A propriedade on_partial_match tem as seguintes opções:</p> <ul style="list-style-type: none"> - fail. A iteração falha. - skip. RepeatingGroup remove de seu escopo de pesquisa a área gerada pelas âncoras aninhadas bem-sucedidas e tenta localizar novamente todas as âncoras aninhadas. O procedimento de remoção e nova tentativa é repetido até que a iteração tenha êxito ou até que não haja mais uma correspondência parcial. Se não houver correspondência parcial, a iteração falhará.
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399.</p>
phase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209. O padrão é main.</p>
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
search_order	<p>Define a ordem de processamento das âncoras aninhadas dentro de cada iteração. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - top-down. As âncoras aninhadas são processadas na sequência definida no Script. - bottom-up. As âncoras aninhadas são processadas em ordem inversa. Selecione essa opção se os dados de uma âncora posterior afetarem a maneira como você processa uma âncora anterior.
separador	<p>Define uma âncora que delimita as seções.</p> <p>Se você deixar a propriedade separator vazia, RepeatingGroup não procurará um delimitador entre as iterações. Em vez disso, ele assumirá que uma iteração está concluída quando tiver localizado todas as âncoras aninhadas. Em seguida, ele começará a analisar a próxima iteração a partir do início da sequência de âncoras aninhadas.</p> <p>Você pode criar um separador complexo inserindo um Grupo na propriedade separator em vez de um Marcador.</p>

Propriedade	Descrição
separator_position	<p>Define o posicionamento de separator relativo às seções. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - after. Há um separador depois cada seção, incluindo a última. Por exemplo, 1 2 3 4 - around. Há separadores antes e depois de cada seção, incluindo a primeira e a última. Por exemplo, 1 2 3 4 - before. Há um separador antes de cada seção, incluindo a primeira. Por exemplo, 1 2 3 4 - between. Há um separador entre as seções sucessivas, mas não antes da primeira, nem depois da última. Por exemplo, 1 2 3 4
skip_failed_iterations	<p>Determina se iterações com falha são ignoradas. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. RepeatingGroup ignora uma iteração com falha e continua com a próxima iteração. Se uma iteração for bem-sucedida, RepeatingGroup terá êxito. - Limpo. RepeatingGroup falhará se qualquer iteração falhar. <p>A propriedade skip_failed_iterations somente terá efeito se a propriedade separator for definida. O padrão é selecionado.</p>
origem	<p>Define uma sequência de recipientes de dados para entrada no RepeatingGroup. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração acessa uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade source quando o RepeatingGroup for chamado por outro componente. Para obter mais informações, consulte "Propriedade de Origem" na página 366.</p>
destino	<p>Define uma sequência de recipientes de dados para saída a partir de RepeatingGroup. Se um recipiente de dados ainda não existir, ele será criado. Cada recipiente de dados é identificado por uma das seguintes propriedades:</p> <ul style="list-style-type: none"> - Localizador. Identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla. Para recipientes de dados de ocorrência múltipla, cada iteração cria uma nova ocorrência. - LocatorByKey. Identifica um recipiente de dados de ocorrência múltipla por chave. - LocatorByOccurrence. Identifica um recipiente de dados de ocorrência múltipla por número de sequência. <p>Use a propriedade target quando a saída de RepeatingGroup for usada por outro componente. Para obter mais informações, consulte "Propriedade de Destino" na página 369.</p>

Nota: Para analisar uma região de seções que exigem tratamento diferente, use uma âncora **DelimitedSections**.

Como Definir uma Âncora RepeatingGroup

É possível definir uma âncora **RepeatingGroup** editando o Script no editor do IntelliScript. Adicione as âncoras aninhadas e as ações que analisam cada iteração de **RepeatingGroup**.

Pesquisar Iterações

Por padrão, um `RepeatingGroup` procura iterações do início até o fim do seu escopo de pesquisa. Para obter mais informações, consulte [“Como um Analisador Procura Âncoras” na página 209](#).

Opcionalmente, você pode definir a propriedade `iteration_order` para uma pesquisa reversa.

Em cada iteração:

- Se o `RepeatingGroup` estiver configurado com um `separador`, ele procurará o próximo separador. Em seguida, ele procurará as âncoras presentes entre um par de separadores.
- Se o `RepeatingGroup` estiver configurado com um `separador`, ele procurará somente as âncoras.

Fim de um RepeatingGroup

Você pode sinalizar o fim de um `RepeatingGroup` da seguinte maneira:

- O `RepeatingGroup` pode continuar até o fim do documento.
- Você pode inserir um Marcador após o `RepeatingGroup`. Por padrão, o Marcador está em uma fase de pesquisa anterior ao `RepeatingGroup`. Isso faz com que o Analisador pesquise o Marcador primeiro e use-o para limitar o escopo de pesquisa do `RepeatingGroup`. Para obter mais informações, consulte [“Ajustar a Fase de Pesquisa” na página 211](#).
- Você pode definir a propriedade de `contagem` limitando a pesquisa para um número máximo de iterações.
- Se o `RepeatingGroup` não tiver um `separador`, ele será encerrado quando o Analisador não puder localizar mais nenhuma iteração.

Sucesso ou Falha de um RepeatingGroup

Se um **RepeatingGroup** não puder localizar as âncoras não opcionais em uma iteração, a iteração falhará.

Quando uma iteração falha, **RepeatingGroup** pode encerrar, reprovar ou ignorar essa iteração. O comportamento é o seguinte:

- Se **RepeatingGroup** não tiver um `separador`, ele será encerrado **RepeatingGroup**. Desde que haja pelo menos uma iteração bem-sucedida antes da iteração com falha, **RepeatingGroup** será bem-sucedido.
- Se **RepeatingGroup** tiver um `separador`, e a propriedade `skip_failed_iterations` não estiver selecionada, **RepeatingGroup** falhará.
- Se **RepeatingGroup** tiver um `separador`, e a propriedade `skip_failed_iterations` estiver selecionada, o Script ignorará a iteração com falha e continuará com a próxima iteração. Desde que pelo menos uma iteração seja bem-sucedida, **RepeatingGroup** terá êxito.

Log de Eventos de um Grupo de Repetição

Os registros do log de eventos para cada iteração de um **RepeatingGroup**.

Se a propriedade `skip_failed_iterations` for selecionada, o **RepeatingGroup** poderá gerar um evento de falha opcional após as iterações bem-sucedidas. Um evento de falha pode estar aninhado dentro da falha opcional. Esses eventos ocorrem porque o **RepeatingGroup** não pode localizar iterações adicionais para analisar. Os eventos são normais e não é necessário se preocupar.

Exemplos Online

Para ver um exemplo online dessa âncora, abra o projeto `samples\Projects\Dynamic_And_RepeatingGroup\Dynamic_And_RepeatingGroup.cmw`. O exemplo usa um `RepeatingGroup` para iterar sobre as linhas de um documento.

Algumas linhas do documento de origem contêm uma referência de nota de rodapé entre parênteses, como "(1)". O **RepeatingGroup** contém um **Grupo**, cuja finalidade é analisar a nota de rodapé e inserir seu conteúdo na saída.

O **Grupo** contém uma âncora de **Conteúdo** que recupera a referência da nota de rodapé e a armazena em uma variável. Em seguida, o **Grupo** ativa uma ação **RunParser** que ativa um Analisador secundário. O Analisador secundário localiza a nota de rodapé mencionada pela variável, analisa-a e insere o resultado na saída.

StructureDefinition

A âncora **StructureDefinition** processa a entrada bem estruturada, como mensagens de texto em conformidade com os protocolos de mensagens padrão do setor. A saída de **StructureDefinition** é uma representação XML dos dados.

Os dados de entrada têm registros delimitados. Você organiza os registros de entrada de maneiras predefinidas. Por exemplo, um registro do tipo A precede um registro do tipo B, seguido por um a três registros do tipo C. Cada registro contém um conjunto organizado de campos.

A seguinte tabela descreve as propriedades da âncora **StructureDefinition**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
format_definition	Define uma lista de âncoras e ações que identificam e extraem os registros. A lista deve conter uma âncora ExtractRecord .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
destino	Define um recipiente de dados no qual o componente armazena sua saída.

Uma **StructureDefinition** tem as seguintes partes:

- A propriedade **format_definition**. Extrai os registros e identifica seus tipos.
- Uma hierarquia de componentes filho. Cada componente filho analisa os registros de um determinado tipo.

A **format_definition** pode conter um **RepeatingGroup** que localiza os registros. No **RepeatingGroup**, uma ou mais âncoras **Conteúdo** recuperam os identificadores de tipo de registro. O **RepeatingGroup** contém uma âncora **ExtractRecord** que passa o registro para a lista de subelementos.

Nota: Quando uma transformação de Biblioteca tem uma âncora **StructureDefinition** com uma propriedade **format_definition** que contém **RepeatingGroup**, a transformação substitui o elemento de repetição, em vez de efetuar sua iteração.

A hierarquia de subelementos reflete a organização necessária dos registros. Você pode configurar sequências, opções, loops de registros e registros obrigatórios ou opcionais.

Os subelementos recebem os registros de **ExtractRecord**. O sistema corresponde cada registro a um subelemento, de acordo com os seguintes critérios:

- Os identificadores `$id` e `$qualifier` do registro devem corresponder aos valores especificados no subelemento.
- A localização do registro na entrada deve corresponder à localização do subelemento na hierarquia.

O subelemento correspondente analisa o registro.

Se não houver subelemento correspondente, a **StructureDefinition** dispara uma notificação. Você pode inserir os componentes **NotificationHandler** que processam a notificação. A transformação usa a âncora **StructureDefinition** para localizar e diagnosticar erros de entrada. Normalmente, a âncora **StructureDefinition** pode continuar a analisar o restante da entrada.

Exemplo

Um Analisador processa a entrada com a seguinte estrutura:

```
ST*12*23~
NM1*12*23*4~
N1*12*23~
NM1*13*23*4~
N2*1*2*3~
SE*12*2:3:4~
```

Os registros são delimitados por caracteres de nova linha. Dentro de cada registro, os campos são delimitados por ~, * e : caracteres.

O primeiro campo de cada registro identifica o tipo de registro. Há cinco tipos de registro principais:

```
ST
NM1
N1
N2
SE
```

Nos registros **NM1**, o segundo campo é um subtipo. Há dois subtipos:

```
NM1*12
NM1*13
```

Os registros devem ocorrer na seguinte sequência:

1. Um registro **ST**.
2. Um registro **NM1*12** seguido por **N1**.
3. Um registro **NM1*13** seguido por **N2**.
4. Um registro **SE**.

Você pode analisar essa entrada configurando uma âncora **StructureDefinition**.

A propriedade **format_definition** contém um **RepeatingGroup** que encontra os registros. O **RepeatingGroup** executa as seguintes operações:

1. Ele encontra o conteúdo do registro até o delimitador de nova linha.
2. Ele extrai o identificador de record-type, como um **ST** ou **NM1** e o armazena na variável `$id`.

3. Se o tipo de registro for `NM1`, ele extrairá o subtipo (12 ou 13) e o armazenará na variável `$qualifier`.
4. Ele executa uma âncora `ExtractRecord` âncora que passa o registro para os subelementos.
`ExtractRecord` anexa os identificadores `$id` e `$qualifier` ao registro.

O elemento é configurado para corresponder a qualquer registro que tenha o identificador `ST`.

O `format_definition` encontra o primeiro registro de entrada e o passa para os subelementos. O primeiro registro corresponde ao primeiro subelemento porque ele tem o identificador `ST`. O primeiro subelemento contém as âncoras `Conteúdo` `Marcador` `Conteúdo` que analisam o registro.

O segundo subelemento define uma sequência de subelementos aninhados. O primeiro subelemento aninhado corresponde a um registro com os identificadores `NM1` e `12`. O segundo subelemento aninhado corresponde a um registro com um identificador `N1`.

O segundo e terceiro registros de entrada são `NM1*12` e `N1`. Eles correspondem à sequência de subelementos. Cada subelemento analisa o registro correspondente.

Suponha que o segundo e terceiro registros sejam `NM1*12` e `N2`. Eles não correspondem à hierarquia de subelementos, portanto, não seriam analisados.

Os próximos registros são `NM1*13` e `N2`. Eles correspondem ao terceiro subelemento, chamado `Loop2000`.

O último registro é `SE`, que corresponde ao último subelemento.

Todos os registros de entrada correspondem à hierarquia de subelementos, portanto a `StructureDefinition` analisa com êxito a entrada completa.

Componentes de Subelemento

Dentro da hierarquia de subelementos, é possível inserir os seguintes componentes:

Subelemento	Descrição
<code>RecordStructureLocal</code>	Corresponde e analisa um único registro.
<code>SequenceStructureLocal</code>	Define uma sequência de subelementos aninhados. Os registros devem ocorrer na mesma sequência que os subelementos aninhados.
<code>ChoiceStructureLocal</code>	Define uma opção de subelementos aninhados. Um registro deve corresponder a um dos subelementos aninhados.
<code>AllStructureLocal</code>	Define um conjunto de subelementos aninhados, sem uma sequência especificada. Os registros podem corresponder aos subelementos aninhados em qualquer ordem.

Os nomes terminam com `Local` porque você pode configurá-los em localizações aninhadas não globais do Script. Há um conjunto correspondente de componentes denominados `RecordStructure`, `SequenceStructure` e assim por diante, sem o sufixo `Local`. É possível configurar esses componentes no nível global do Script e fazer referência a eles sempre que for necessário. Para fazer referência aos componentes globais, insira um subelemento `EmbeddedStructure`.

A lista de subelementos de nível superior de um `StructureDefinition` é equivalente a `SequenceStructureLocal`. Os registros devem ocorrer na mesma sequência que os subelementos de nível superior.

Por padrão, cada subelemento deve ocorrer exatamente uma vez. Para alterar o padrão, defina as propriedades `minOccurs` e `maxOccurs` do subelemento. Por exemplo, se um subelemento puder ficar ausente ou ocorrer até três vezes, defina `minOccurs = 0` e `maxOccurs = 3`. Para permitir ocorrências ilimitadas, defina `maxOccurs = -1`.

Para obter mais informações sobre os componentes de subelementos, consulte a [“Referência do Subcomponente de Âncora” na página 249](#).

Notificações

Se um registro ou um conjunto de registros não corresponder à hierarquia de subelementos, **StructureDefinition** acionará uma notificação.

A seguinte tabela descreve os tipos de notificações:

Notificação	Descrição
MandatoryStructureMissing	Um registro obrigatório não é exibido na entrada.
MismatchIDs	Os IDs do registro e do subelemento correspondem parcialmente. Por exemplo, há dois identificadores de registro, e apenas um deles corresponde.
StructureBelowMinOccurs	Há menos registros correspondentes do subelemento do que o definido em minOccurs .
StructureExceedsMaxOccurs	Há mais registros correspondentes do subelemento do que o definido em maxOccurs .
StructureOutOfSequence	Os registros correspondem aos subelementos, mas não na sequência necessária. Por exemplo, os subelementos definem uma sequência ABC, mas a entrada contém ACB.
UnexpectedRecord	Os registros correspondem aos subelementos, mas não na hierarquia necessária. Por exemplo, o subelemento define uma sequência ABC, e D está definido em outra localização. A entrada contém ABD.
UnrecognizedRecord	Nenhum subelemento corresponde a um dos identificadores de registro.
XsdValidationError	A entrada não corresponde aos requisitos do esquema.

Configure os componentes **NotificationHandler** na propriedade **notifications** de **StructureDefinition** ou de um subelemento. Você também pode configurar manipuladores na propriedade **notifications** de um componente de nível superior, como um **Analizador** ou um **Grupo** que contém **StructureDefinition**. Se existir um manipulador no subelemento em que uma incompatibilidade ocorre, ele processará a notificação. Se nenhum manipulador existir, a notificação será propagada na hierarquia do IntelliScript até ser processada por um manipulador. Se não houver um manipulador para uma notificação, essa notificação será ignorada, e o **StructureDefinition** continuará processando a entrada.

Acompanhando o Andamento

Como o `format_definition` extrai registros, ele atualiza a variável de sistema `VarStructureDetails`. Você pode usar a variável em notificações. Por exemplo, para reportar o identificador de registro, um identificador de notificação poderá inserir `VarStructureDetails/RecordId` na sua saída.

Para obter mais informações, consulte [“Variáveis de Sistema” na página 197](#).

Referência de Componentes de Pesquisador

Componentes de pesquisador são usados para as seguintes finalidades:

- Para definir a localização de âncoras. Para obter mais informações, consulte [“Referência do Componente de Âncora” na página 216](#).
- Para definir strings ou caracteres delimitadores. Para obter mais informações, consulte [“Referência de Componente de Formato” na página 178](#).
- Para definir a string `find_what` de um transformador **Replace**. Para obter mais informações, consulte [“Referência de Componente Transformador” na página 260](#).

AttributeSearch

O componente de pesquisa **AttributeSearch** procura um documento de origem para o valor de um atributo especificado. O componente recupera o valor de uma expressão em um dos seguintes formatos:

- `AttributeName = valor`
- `AttributeName = "valor"`

onde `AttributeName` é o nome do atributo, as aspas podem ser simples ou duplas e os espaços são opcionais.

AttributeSearch é uma das configurações da propriedade **valor** da âncora de **Conteúdo**. Para obter mais informações, consulte [“Conteúdo” na página 218](#).

A tabela a seguir descreve as propriedades do componente **AttributeSearch**:

Propriedade	Descrição
<code>att</code>	Define o nome do atributo.
<code>match_case</code>	Determina se o nome do atributo faz distinção entre maiúsculas e minúsculas. A propriedade match_case tem as seguintes opções: <ul style="list-style-type: none">- Marcado. O nome do atributo faz distinção entre maiúsculas e minúsculas.- Desmarcado. O nome do atributo não faz distinção entre maiúsculas e minúsculas.

Exemplo

Um documento HTML contém o elemento:

```
<img src='MyPicture.gif'>
```

Você pode usar **AttributeSearch** para recuperar o valor do atributo `src`. Ele retornará o texto `MyPicture.gif`.

Sintaxe de Atributo Válida

AttributeSearch lê pares de nome-valor que contêm um sinal de igualdade. O sinal de igualdade pode ser rodeado por espaços. O valor pode ser rodeado por aspas duplas, aspas simples ou sem aspas.

Por exemplo, suponha que **AttributeSearch** esteja configurada para pesquisar um atributo chamado `hora`. Todos os exemplos a seguir têm sintaxe válida e retornam o mesmo valor, `12:55:33`.

```
time = "12:55:33"  
time="12:55:33"  
time = '12:55:33'  
time='12:55:33'
```

```
time = 12:55:33
time=12:55:33
```

Exemplo Online

Para ver um exemplo online desse componente, abra o projeto `samples\Projects\Content\Content.cmw`. O exemplo ilustra o uso de um `AttributeSearch` para analisar um documento de texto que tem uma estrutura `variable = value`.

LearnByExample

O componente de pesquisa **LearnByExample** aprende como para procurar texto examinando a localização do texto no exemplo de documento de origem. Ele usa o formato de Analisador para interpretar o documento de origem.

Por exemplo, se o Analisador tiver um formato delimitado por tabulação, **LearnByExample** contará o número de guias desde o início da pesquisa até o texto de exemplo. Ele procura pelo texto no documento de origem que está no mesmo número de tabulações desde o início do escopo da pesquisa.

LearnByExample é uma das configurações da propriedade **valor** da âncora de **Conteúdo**. Para obter mais informações, consulte [“Conteúdo” na página 218](#).

Se o atributo **direção** da âncora de **Conteúdo** estiver definido como `invertido`, o componente contará os delimitadores a partir do fim do escopo da pesquisa.

A tabela a seguir descreve as propriedades do componente **LearnByExample**:

Propriedade	Descrição
example	Define o texto no documento de origem de exemplo na localização da âncora.

Nota: O componente do pesquisador **LearnByExample** aplica heurística que diferencia maiúsculas de minúsculas. Para usar o componente do pesquisador **LearnByExample**, o exemplo deve ter o mesmo formato da entrada esperada. O formato deve ser uniforme em todo o arquivo (ou arquivos) de entrada. Se o exemplo for diferente da entrada, o componente do pesquisador **LearnByExample** poderá falhar.

NewlineSearch

O componente de pesquisa **NewlineSearch** pesquisa um caractere de nova linha ou avanço de linha (0x0A), um caractere de retorno de carro (0x0D) ou ambos.

A âncora **Marcador** pode usar **NewlineSearch** para achar marcadores de nova linha. Um componente de **Delimitador** pode usar **NewlineSearch** para achar delimitadores de nova linha.

OffsetSearch

O componente de pesquisado **OffsetSearch** define o número de caracteres entre um ponto de referência e uma âncora. Por exemplo, ele pode definir o número de caracteres entre o fim de um **Marcador** e o início de uma âncora de **Conteúdo**.

A tabela a seguir descreve as propriedades do componente **OffsetSearch**:

Propriedade	Descrição
allow_smaller_offset	Determina se um deslocamento que se estende além do escopo da pesquisa é válido. Selecione essa propriedade para permitir um tamanho do campo truncado no fim de um documento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. OffsetSearch obtém êxito quando um deslocamento se estende além do escopo da pesquisa.- Desmarcado. OffsetSearch falha quando um deslocamento se estende além do escopo da pesquisa.
deslocamento	Define o número de caracteres entre o ponto de referência e a âncora. Em algumas localizações onde OffsetSearch foi usado, como em uma âncora de Marcador , o editor do IntelliScript exibe um botão de procurar ao lado da propriedade de deslocamento . Você pode inserir um valor ou procurar um recipiente de dados que contém o valor.

PatternSearch

O componente de pesquisador **PatternSearch** procura uma string que corresponde a uma expressão regular.

Âncoras podem usar **PatternSearch** para localizar marcadores ou conteúdo. O componente **Delimitador** pode usar **PatternSearch** para localizar delimitadores. O transformador **Replace** pode usar **PatternSearch** para localizar o texto a ser substituído.

A seguinte tabela descreve as propriedades do componente de pesquisador **PatternSearch**:

Propriedade	Descrição
escape_sequence	Define um prefixo que faz com que o componente de pesquisa ignore uma instância do padrão no documento de origem.
pattern	Define a expressão regular.

Para obter mais informações sobre a sintaxe de expressões regulares, consulte [“Sintaxe de Expressões Regulares” na página 282](#).

Exemplo

Suponha que você deseja definir a string `%%%`, que contém um ou mais símbolos `%` como um delimitador. Dentro do componente **Delimitador**, você pode usar **PatternSearch** com a seguinte expressão regular:

```
%+
```

Em outro exemplo, suponha que você deseja definir uma vírgula e um ponto-e-vírgula como delimitadores alternativos no mesmo nível da hierarquia do delimitador. Você pode usar a seguinte expressão regular:

```
[, ;]
```

SegmentSearch

O componente pesquisador **SegmentSearch** pesquisa por marcadores de abertura e de fechamento em uma string de texto. Ele retorna o segmento do marcador de abertura para o marcador de fechamento, incluindo os marcadores. **SegmentSearch** é uma das opções para o atributo **find_what** do transformador **Substituir**.

A seguinte tabela descreve as propriedades do componente pesquisador **SegmentSearch**:

Propriedade	Descrição
abertura	Define os critérios de pesquisa para o marcador de abertura. As opções são os seguintes componentes pesquisadores: <ul style="list-style-type: none">- NewlineSearch- OffsetSearch- PatternSearch- TextSearch
fechamento	Define os critérios de pesquisa para o marcador de fechamento. As opções são os seguintes componentes pesquisadores: <ul style="list-style-type: none">- NewlineSearch- OffsetSearch- PatternSearch- TextSearch

TextSearch

O componente pesquisador **TextSearch** pesquisa por uma string explícita.

As âncoras podem usar o **TextSearch** para localizar marcadores. O componente **Delimitador** pode usar o **TextSearch** para localizar os delimitadores. O transformador **Substituir** pode usar o **TextSearch** para localizar o texto a ser substituído.

A seguinte tabela descreve as propriedades do componente pesquisador **TextSearch**:

Propriedades	Descrição
escape_sequence	Define um prefixo que faz com que a pesquisa ignore uma instância da string no documento de origem. Nas localizações onde a pesquisa dinâmica é suportada, você pode procurar um retentor de dados que contém a sequência de escape.
match_case	Determina se o texto definido deve corresponder exatamente, com as mesmas letras em maiúsculas e minúsculas. O padrão é desmarcado.
texto	Define a string a ser localizada. Nas localizações onde a pesquisa dinâmica é suportada, você pode procurar um retentor de dados que contém a string.

Exemplo

Para definir a string percent-percent-tab como um delimitador, crie um componente `Delimitador` e defina a sua propriedade `de_pesquisa` como `TextSearch`. Na propriedade `de_texto` digite:

`%%`

Em seguida, pressione **CTRL+A** e digite `009` (código ASCII de um caractere de tabulação).

Especificando uma String de Pesquisa Dinamicamente

Em algumas localizações onde `TextSearch` é usado, como em um componente `Delimitador` ou uma âncora `Marcador`, um botão procurar aparece à direita da caixa de texto. Navegue até um retentor de dados que contém o texto de pesquisa.

Para localizar instâncias repetidas da primeira palavra em um documento, você pode definir uma âncora **Conteúdo** que recupera a primeira palavra e a armazena em uma variável. Você pode definir âncoras **Marcador** que usam **TextSearch** para localizar outras instâncias da palavra que você armazenou na variável.

Exemplo Online

Para obter um exemplo online desse componente, abra o projeto `samples\Projects\Dynamic_And_RepeatingGroup\Dynamic_And_RepeatingGroup.cmw`.

No componente **GetRemarkParser** desse exemplo, uma âncora de **Marcador** usa **TextSearch** definido dinamicamente para localizar uma nota de rodapé no final do documento de origem. Para obter mais informações sobre esse exemplo, consulte [“RepeatingGroup” na página 236](#).

TypeSearch

O componente pesquisador **TypeSearch** pesquisa por uma âncora de um tipo de dados especificado.

TypeSearch é uma das configurações da propriedade **valor** da âncora **Conteúdo**. Para obter mais informações, consulte [“Conteúdo” na página 218](#).

A seguinte tabela descreve as propriedades do componente pesquisador **TypeSearch**:

Propriedade	Descrição
val_type	Determina o tipo de dados da âncora a ser pesquisada.

Referência do Subcomponente de Âncora

Os subcomponentes de âncora são atribuídos como os valores de determinadas propriedades de âncora.

AllStructure

O componente **AllStructure** define um conjunto de subelementos aninhados sem levar a sequência em consideração. Um conjunto de registros corresponderá a **AllStructure** se corresponder a todos os subelementos em qualquer sequência. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

O componente **AllStructure** aparece no nível global do Script e tem a mesma função que **AllStructureLocal**. É possível fazer referência a ele no atributo **ref** de um **EmbeddedStructure**.

A seguinte tabela descreve as propriedades do componente **AllStructure**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.

Propriedade	Descrição
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
target	Define um recipiente de dados no qual o componente armazena sua saída.

AllStructureLocal

O componente **AllStructureLocal** define um conjunto de subelementos aninhados sem levar a sequência em consideração. Um conjunto de registros corresponderá a **AllStructureLocal** se corresponder a todos os subelementos em qualquer sequência. Para obter mais informações, consulte ["StructureDefinition" na página 241](#).

AllStructureLocal é um subelemento da âncora **StructureDefinition** e tem a mesma função que **AllStructure**. No nível global do Script, use **AllStructure**.

A seguinte tabela descreve as propriedades do componente **AllStructure**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
minOccurs	Define o número mínimo de registros correspondentes. O padrão é 1.
maxOccurs	Define o número máximo de registros correspondentes. O padrão é 1. Use -1 para um número ilimitado.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sub_elements	Define uma lista de subelementos aninhados.
target	Define um recipiente de dados no qual o componente armazena sua saída.

ChoiceStructure

O componente **ChoiceStructure** define um conjunto de subelementos aninhados. Um registro corresponderá a **ChoiceStructure** se corresponder a qualquer subelemento aninhado. Para obter mais informações, consulte ["StructureDefinition" na página 241](#).

O componente **ChoiceStructure** aparece no nível global do Script e tem a mesma função que **ChoiceStructureLocal**. É possível fazer referência a ele no atributo **ref** de um **EmbeddedStructure**.

A seguinte tabela descreve as propriedades do componente **ChoiceStructure**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
target	Define um recipiente de dados no qual o componente armazena sua saída.

ChoiceStructureLocal

O componente **ChoiceStructureLocal** define um conjunto de subelementos aninhados. Um registro corresponderá a **ChoiceStructureLocal** se corresponder a qualquer subelemento aninhado. Para obter mais informações, consulte ["StructureDefinition" na página 241](#).

ChoiceStructureLocal é um subelemento da âncora **StructureDefinition** e tem a mesma função que **ChoiceStructure**. No nível global do Script, use **ChoiceStructure**.

A seguinte tabela descreve as propriedades do componente **ChoiceStructureLocal**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
minOccurs	Define o número mínimo de registros correspondentes. O padrão é 1.

Propriedade	Descrição
maxOccurs	Define o número máximo de registros correspondentes. O padrão é 1. Use -1 para um número ilimitado.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sub_elements	Define uma lista de subelementos aninhados.
target	Define um recipiente de dados no qual o componente armazena sua saída.

Conectar

O componente **Conectar** especifica um link entre os recipientes de dados em dois componentes. Os dois recipientes de dados devem ter o mesmo tipo de dados.

A tabela a seguir descreve as propriedades do componente **Conectar**:

Propriedade	Descrição
data_holder	Define um recipiente de dados que é referenciado no Analisador, Serializador ou Mapeador principal. Nota: Se o recipiente de dados for uma variável, a transformação atribuirá um valor padrão vazio. Se a variável tiver um tipo de dados que não aceita um valor vazio, como <code>xs:boolean</code> , certifique-se de que a variável tem um valor antes de executar a transformação incorporada.
embedded_data_holder	Define um recipiente de dados que é referenciado no Analisador, Serializador ou Mapeador secundário.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

A propriedade **schema_connections** os seguintes componentes pode ter uma ou mais instâncias do componente **Conectar**:

- EmbeddedParser. Especifica onde um Analisador secundário armazena o resultado na saída do Analisador principal.
- EmbeddedSerializer. Especifica um link entre os recipientes de dados de entrada de um Serializador secundário e os recipientes de dados de entrada do Serializador principal.
- EmbeddedMapper. Especifica um link entre os recipientes de dados de entrada e saída.
- EmbeddedStructure. Especifica um link entre os destinos de subelementos globais e locais de **StructureDefinition**.

Exemplo

Um Analisador secundário gera um elemento XML denominado `ID`. É desejável que o Analisador principal armazene este resultado em uma variável chamada `VarID`. Você pode conectar `ID` à `VarID`.

Para um ver um exemplo adicional, consulte [“EmbeddedSerializer” na página 344](#).

EmbeddedStructure

O componente **EmbeddedStructure** ativa componentes definidos no nível global do Script. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

EmbeddedStructure é um subelemento da âncora **StructureDefinition**.

A seguinte tabela descreve as propriedades do componente **EmbeddedStructure**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
minOccurs	Define o número mínimo de registros correspondentes.
maxOccurs	Define o número máximo de registros correspondentes. Use -1 para um número ilimitado.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .
ref	Define o nome do componente globalmente configurado.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
schema_connections	Conecta o destino do subelemento de referências ao destino de EmbeddedStructure . Para obter mais informações, consulte “Conectar” na página 252 .
target	Define um recipiente de dados no qual o componente armazena sua saída.

RecordStructure

O componente **RecordStructure** define um conjunto de componentes. Um conjunto de registros corresponderá **RecordStructure** se ele tiver os mesmos identificadores. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

O componente **RecordStructure** aparece no nível global do Script e tem a mesma função que **RecordStructureLocal**. É possível fazer referência a ele no atributo **ref** de um **EmbeddedStructure**.

A seguinte tabela descreve as propriedades do componente **RecordStructure**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
ids	Define uma ou mais strings.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
target	Define um recipiente de dados no qual o componente armazena sua saída.

RecordStructureLocal

O componente **RecordStructureLocal** define um conjunto de componentes. Um conjunto de registros corresponderá **RecordStructureLocal** se ele tiver os mesmos identificadores. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

RecordStructureLocal é um subelemento da âncora **StructureDefinition** e tem a mesma função que **RecordStructure**. No nível global do Script, use **RecordStructure**.

A seguinte tabela descreve as propriedades do componente **RecordStructureLocal**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
ids	Define uma ou mais strings.
minOccurs	Define o número mínimo de registros correspondentes. O padrão é 1.
maxOccurs	Define o número máximo de registros correspondentes. O padrão é 1. Use -1 para um número ilimitado.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
target	Define um recipiente de dados no qual o componente armazena sua saída.

SequenceStructure

O componente **SequenceStructure** define uma sequência de subelementos aninhados. Um conjunto de registros corresponderá a **SequenceStructure** se corresponder a todos os subelementos aninhados em sequência. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

O componente **SequenceStructure** aparece no nível global do Script e tem a mesma função que **SequenceStructureLocal**. É possível fazer referência a ele no atributo **ref** de um **EmbeddedStructure**.

A seguinte tabela descreve as propriedades do componente **SequenceStructure**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
target	Define um recipiente de dados no qual o componente armazena sua saída.

SequenceStructureLocal

O componente **SequenceStructureLocal** define uma sequência de subelementos aninhados. Um conjunto de registros corresponderá a **SequenceStructureLocal** se corresponder a todos os subelementos aninhados em sequência. Para obter mais informações, consulte [“StructureDefinition” na página 241](#).

SequenceStructureLocal é um subelemento da âncora **StructureDefinition** e tem a mesma função que **SequenceStructure**. No nível global do Script, use **SequenceStructure**.

A seguinte tabela descreve as propriedades do componente **SequenceStructureLocal**:

Propriedade	Descrição
action	Define uma ação que é executada na lista de subcomponentes.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
minOccurs	Define o número mínimo de registros correspondentes. O padrão é 1.
maxOccurs	Define o número máximo de registros correspondentes. O padrão é 1. Use -1 para um número ilimitado.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notifications	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sub_elements	Define uma lista de subelementos aninhados.
target	Define um recipiente de dados no qual o componente armazena sua saída.

CAPÍTULO 17

Transformadores

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Transformadores, 257](#)
- [Definindo Transformadores, 257](#)
- [Propriedades de Transformadores Padrão, 259](#)
- [Referência de Componente Transformador, 260](#)

Visão Geral de Transformadores

Transformadores modificam a saída de outros componentes.

É possível usar transformadores dentro de componentes como âncoras, âncoras de serialização e ações. Por exemplo, se você usar um transformador dentro de uma âncora de `Conteúdo`, ele modificará os dados que a âncora extrai do documento de origem.

É possível usar transformadores como processadores de documentos. Também é possível definir um transformador no nível global de um Script e defini-lo como o componente de inicialização.

Definindo Transformadores

Você pode definir transformadores nas seguintes localizações do Script:

- Na propriedade **transformers** de uma âncora ou de uma âncora de serialização
- Na propriedade **default_transformers** de um formato ou de um serializador
- No processador de documentos **ProcessByTransformers**
- Na propriedade **transformers** de certas ações
- No nível global, como um componente autônomo executável que modifica um documento de origem.

Usando Transformadores em Âncoras

É possível usar transformadores em uma âncora que cria a saída XML, como uma âncora de `Conteúdo`. No Script, aninhe os componentes transformadores dentro da propriedade **transformers** da âncora.

A entrada de um transformador é a saída bruta da âncora, antes de a âncora inserir a saída em um recipiente de dados.

Por exemplo, suponha que você esteja analisando o seguinte documento de origem:

```
First name: Ron
Last name: Lehrer
```

Você deseja criar a saída XML em maiúsculas (ALL CAPS) da seguinte maneira:

```
<Person>
  <FirstName>RON</FirstName>
  <LastName>LEHRER</LastName>
</Person>
```

Para fazer isso, é possível configurar as âncoras de **Conteúdo**, que recuperam as strings `Ron` e `Lehrer`, com o transformador `ChangeCase`.

Sequências de Transformadores

Você pode configurar uma âncora com uma sequência de transformadores. Cada transformador modifica a saída do transformador precedente.

No exemplo `Ron Lehrer`, suponha que você deseja a seguinte saída:

```
<Person>
  <FirstName>- RON -</FirstName>
  <LastName>- LEHRER -</LastName>
</Person>
```

Para fazer isso, você pode configurar as âncoras **Conteúdo** com os transformadores `ChangeCase` e `AddString`. Os transformadores alteram a diferenciação de maiúsculas e minúsculas e adicionam os hífens, em sequência.

Transformadores padrão

Muitas vezes você deseja que os mesmos transformadores sejam executados em todas as âncoras de **Conteúdo** de um Analisador. Você pode configurar o componente de formato do Analisador com transformadores padrão. Isso poupa o trabalho de adicionar os mesmos transformadores a cada âncora do Analisador.

Para fazer isso, aninhe os transformadores na propriedade **default_transformers** do formato. Para obter mais informações, consulte [“Referência de Componente de Formato” na página 178](#).

Muitos dos componentes de formato predefinidos incluem transformadores padrão. Por exemplo, o componente **HtmlFormat** tem transformadores padrão que removem marcas HTML da saída e convertem entidades HTML em texto simples. É possível alterar os transformadores padrão editando a propriedade **default_transformers**.

Se uma âncora tiver os seus próprios transformadores, eles serão executados após os transformadores padrão.

Você pode cancelar os transformadores padrão para âncoras específicas. Para fazer isso, defina a propriedade **ignore_default_transformers** da âncora.

Usando Transformadores como Processadores de Documento

Você pode executar um transformador ou uma sequência de transformadores como um processador de documento.

Por exemplo, você pode executar o transformador `RemoveTags` como um processador em um documento HTML. O transformador remove as marcas de HTML antes de um Analisador iniciar a pesquisa por âncoras no documento.

Para fazer isso, configure o componente de formato de Analisador com o processador de documentos `ProcessByTransformers` e aninhe os transformadores no componente.

Usando Transformadores em Âncoras de Serialização

Você pode usar transformadores em âncoras de serialização que gravam no documento de saída, como `ContentSerializer`. O transformadores modificam os dados antes do serializador os gravar no documento.

Por exemplo, um `ContentSerializer` poderá gravar o conteúdo de um retentor de dados chamado `DoctorName` em um documento de saída. Você pode configurar o `ContentSerializer` com um transformador `AddString` que adiciona o prefixo "Dr. " ao conteúdo. Suponha que a entrada XML tenha o seguinte formato:

```
<DoctorName>Albert Schweitzer</DoctorName>
```

O transformador modifica o conteúdo, resultando na seguinte saída:

```
Dr. Albert Schweitzer
```

Você pode adicionar transformadores à propriedade `default_transformers` de um serializador. Os transformadores que você adiciona aqui são executados em todas as âncoras de serialização `ContentSerializer` antes que gravarem no documento de saída.

Usando Transformadores em Ações

Certas ações, como **SetValue** e **Mapear**, aplicam transformadores às suas saídas. Para obter mais informações, consulte ["Visão Geral de Ações" na página 295](#).

Propriedades de Transformadores Padrão

A seguinte tabela descreve as propriedades padrão de Transformadores:

Propriedade	Definição
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Referência de Componente Transformador

Os transformadores modificam os dados.

AbsURL

O transformador **AbsURL** converte um caminho ou URL de arquivo relativo em um caminho absoluto.

Por exemplo, se a entrada for `test.html` e a URL base for `http://www.example.com`, a saída será `http://www.example.com/test.html`.

Se a entrada for um caminho absoluto, o transformador não será alterado.

A tabela a seguir descreve as propriedades do transformador **AbsURL**:

Propriedade	Descrição
base_URL	Define o caminho ou URL base.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

AddEmptyTagsTransformer

O transformador **AddEmptyTagsTransformer** verifica se todos os elementos definidos no esquema existem na entrada XML. Se não existirem, ele adiciona elementos vazios ao XML. Esse é um transformador XML-para-XML.

A tabela a seguir descreve as propriedades do transformador **AddEmptyTagsTransformer**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
root_element	Define o elemento raiz do XML.

AddString

O transformador **AddString** adiciona uma string antes e depois do texto de entrada.

A tabela a seguir descreve as propriedades do transformador **AddString**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
pré	Define a string a ser adicionada antes do texto.
pós	Define a string a ser adicionada depois do texto.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo Online

Para ver um exemplo online, abra `samples\Projects\Transformers_Example\Transformers_Example.cmw`. A primeira âncora de Conteúdo no Analisador está configurada com um transformador `AddString`.

Base64Decode

O transformador **Base64Decode** converte a codificação MIME Base64 em uma string binária.

A tabela a seguir descreve as propriedades do transformador **Base64Decode**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
tolerância	Controla como o transformador processa caracteres de espaço em branco ou seções não base64 da entrada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- ignore_white_spaces. Processa todos os caracteres, exceto espaços em branco. Padrão.- ignore_none. Processa todos os caracteres.- ignore_non_base64. Processa apenas caracteres base-64.

Base64Encode

O transformador **Base64Encode** converte uma string binária para a codificação MIME Base64. Isso é útil para salvar dados binários em XML.

A tabela a seguir descreve as propriedades do transformador **Base64Encode**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

BidiConvert

O transformador **BidiConvert** reverte uma string gravada em idiomas da direita para a esquerda (RTL), como Hebraico e Árabe. A entrada deve estar no formato RTL. A saída será da esquerda para a direita (LTR).

O transformador **BidiConvert** funciona no Windows onde o idioma padrão é RTL. Se preferir um transformador semelhante que é executado em todas as plataformas, use **hebrewBidi**. Os dois transformadores usam algoritmos um pouco diferentes que ocasionalmente apresentam resultados diferentes.

A tabela a seguir descreve as propriedades do transformador **BidiConvert**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8. Solução: Use **hebrewBidi**.

CDATADecode

O transformador **CDATADecode** decodifica uma seção **CDATA** de um documento XML. Por exemplo, ele converte

```
<![CDATA[100 < 200]]>
```

em

```
100 < 200
```

Nota: Se você gravar o resultado em XML, o Script o recodificará usando a codificação XML padrão:

```
100 &lt; 200
```

A seguinte tabela descreve as propriedades do transformador **CDATADecode**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

CDATAEncode

O transformador **CDATAEncode** converte uma string em uma seção **CDATA** de um documento XML. Por exemplo, ela converte

```
100 < 200
```

em

```
<![CDATA[100 < 200]]>
```

A tabela a seguir descreve as propriedades do transformador **CDATAEncode**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

ChangeCase

O transformador **ChangeCase** altera uma string de texto para todas as letras maiúsculas, todas as letras minúsculas ou apenas a primeira letra em maiúscula. Esse transformador funciona com caracteres em inglês. Ele pode falhar com alguns caracteres que não são em inglês. Por exemplo, ele não converterá minúsculas em alemão ß para maiúsculas SS.

A tabela a seguir descreve as propriedades do transformador **ChangeCase**:

Propriedade	Descrição
case_type	Define se a saída será em maiúsculas ou minúsculas. A propriedade case_type tem as seguintes opções: <ul style="list-style-type: none">- all_caps. A saída tem todas as letras maiúsculas.- all_lower. A saída tem todas as letras minúsculas.- first_cap. A primeira letra da saída é maiúscula e as restantes são minúsculas. O padrão é all_caps.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo Online

Para ver um exemplo online, abra `samples\Projects\Transformers_Example\Transformers_Example.cmw`. A terceira âncora de Conteúdo no Analisador está configurada com um transformador `ChangeCase`.

CreateGuid

O transformador **CreateGuid** gera um identificador de GUID. O GUID resultante é exclusivo sempre que esse transformador é executado.

O GUIDs podem ter um formato não padrão em plataformas Linux e UNIX. Para um transformador totalmente compatível com UNIX, use **CreateUUID**. Para obter mais informações, consulte ["CreateUUID" na página 265](#).

CreateUUID

O transformador **CreateUUID** gera um identificador de UUID compatível com as plataformas Windows, Linux e UNIX. O UUID resultante é exclusivo toda vez que o transformador é executado.

A tabela a seguir descreve as propriedades do transformador **CreateUUID**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

DateFormatICU

O transformador **DateFormatICU** converte uma data ou hora em um formato especificado pelo usuário.

A tabela a seguir descreve as propriedades do transformador **DateFormatICU**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
input_format	Define o formato da data de entrada, por exemplo, d/M/yy. Digite o formato ou selecione um recipiente de dados que contém um formato.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
output_format	Define o formato da data de saída, por exemplo, MM/dd/yyyy. Digite o formato ou selecione um recipiente de dados que contém um formato.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo

Suponha que você configure um transformador `DateFormatICU` com:

```
input_format = "d/M/yy"  
output_format = "MM/dd/yyyy"
```

Se a entrada for

13/3/05

a saída será

03/13/2005

Formatos com Suporte

O transformador **DateFormatICU** usa as convenções de ICU para representar o formato de data e hora. A seguinte tabela lista os símbolos que você pode usar nos padrões de formato. Para obter mais informações, consulte:

<http://icu.sourceforge.net/apiref/icu4c/classSimpleDateFormat.html>

Símbolo de Padrão	Significado	Tipo	Exemplos
G	Designador de era	Texto	AD
y	Ano	Número	1996
u	Ano estendido	Número	-200, ou seja, 201 AC
M	Mês do ano	Texto ou número	Julho 07
d	Dia do mês	Número	10
h	Hora em AM/PM (1-12)	Número	12
H	Hora do dia (0-23)	Número	0
m	Minuto da hora	Número	30
s	Segundo do minuto	Número	55
S	Segundo fracionário	Número	978
E	Dia da semana	Texto	Terça-feira
e	Dia da semana (local 1-7)	Número	2
D	Dia do ano	Número	189
F	Dia da semana do mês	Número	2, ou seja, a segunda quarta-feira de Julho
w	Semana do ano	Número	27
W	Semana do mês	Número	2

Símbolo de Padrão	Significado	Tipo	Exemplos
a	Marcador AM/PM	Texto	PM
k	Hora do dia (1-24)	Número	24
K	Hora em AM/PM (0-11)	Número	0
z	Fuso horário	Hora	Hora Padrão do Pacífico
Z	Fuso horário (RFC 822)	Número	-0800
v	Fuso horário (genérico)	Texto	Hora do Pacífico
g	Dia juliano	Número	2451334
A	Milissegundos no dia	Número	69540000
' '	O texto entre aspas simples é interpretado como uma string literal	Texto	'Hoje é 'dd/MM/aaaa gera saída como Hoje é 15/03/2005
''	Aspas simples literais	Texto	'horas' gera a saída horas

A contagem dos símbolos de padrão determina mais o formato:

- Para texto: Quatro ou mais símbolos de padrão significa usar o formato completo. Menos de quatro significa usar um formato curto ou abreviado, se houver. Por exemplo, se `EEEE` produzir `Segunda-feira`, `EEE` produzirá `Seg`.
- Para números: O número dos símbolos de padrão é o número mínimo de dígitos. Os números mais curtos são preenchidos com zero à esquerda. Por exemplo, se `m` produz `6`, `mm` produzirá `06`.
- Para anos: O ano de dois dígitos é `yy` e o ano com quatro dígitos é `yyyy`. Por exemplo, se `yy` produz `05`, `yyyy` produzirá `2005`.
- Para meses: Se `M` produz `1`, `MM` produzirá `01`, `MMM` produzirá `Jan` e `MMMM` produzirá `Janeiro`.

Todos os caracteres não alfabéticos são interpretados como literais, mesmo se eles não estiverem entre aspas simples. Por exemplo, `dd/MM/yyyy HH:mm` produz `15/03/2005 13:15`.

Dos96HebToAscii

O transformador **Dos96HebToAscii** converte codificação de 7 bits em Hebraico em página de código Windows-1255.

DynamicTable

O componente **DynamicTable** define um recipiente de dados que contém uma tabela de pesquisa. A tabela é usada pelo transformador **LookupTransformer**.

A tabela a seguir descreve as propriedades do componente **DynamicTable**:

Propriedade	Descrição
tabela	Define o recipiente de dados que contém a tabela.

EbcdicToAscii

O transformador **EbcdicToAscii** converte EBCDIC em texto ASCII.

EDIFACTValidation

O validador **EDIFACTValidation** testa se uma string de origem é uma mensagem EDIFACT válida.

A seguinte tabela descreve as propriedades do validador **EDIFACTValidation**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
enabled	Determina a configuração para param1 .
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
param1	Determina se a entrada é opcional. param1 se chama is_optional e tem apenas uma propriedade, enabled . enabled tem as seguintes opções: <ul style="list-style-type: none">- Selected. Os dados de entrada são opcionais.- Cleared. Os dados de entrada são obrigatórios.
param2	Define um tipo de dados EDI. param2 se chama input_type e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados.
param3	Define um intervalo de inteiros. param3 se chama minmax_limits e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados que especifica dois inteiros separados por um hífen.
param4	Define uma lista de valores. param4 se chama enumerations e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados que especifica uma lista de strings ou inteiros separados por vírgula.

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
value	Define um valor para param1 , param2 ou param3

Nota: Esse componente foi preterido. O editor do IntelliScript o exibe apenas para projetos herdados. Não o use em novos Scripts. **Solução:** Use outros componentes de validador.

EncodeAsUrl

O transformador **EncodeAsUrl** codifica espaços e caracteres especiais como obrigatórios em um URL. Os caracteres são codificados como um símbolo de porcentagem (%) seguido por um número hexadecimal.

Por exemplo, o transformador **EncodeAsUrl** converte

```
http://www.example.com?name=John Doe
```

em

```
http://www.example.com?name=John%20Doe
```

Nota: Os caracteres de parênteses não são codificados.

A tabela a seguir descreve as propriedades do transformador **EncodeAsUrl**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo Online

Para ver um exemplo online, abra `samples\Projects\Transformers_Example\Transformers_Example.cmw`. A quarta âncora de Conteúdo do Analisador está configurada com um transformador `EncodeAsUrl`.

Codificador

O transformador **Codificador** converte o texto de uma página de código para outra.

A tabela a seguir descreve as propriedades do transformador **Codificador**:

Propriedade	Descrição
add_prefix	Adiciona um marca BOM (Marca de Ordem de Byte) quando a codificação de saída for UTF-8 ou UTF-16.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
input_code_page	Define a página de código do texto de entrada.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
output_code_page	Define a página de código do texto de saída.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

FormatNumber

O transformador **FormatNumber** formata um número adicionando um sinal, ponto decimal, zeros à esquerda e direita, e unidade.

A tabela a seguir descreve as propriedades do transformador **FormatNumber**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
insert_decimal_point	Define o símbolo do ponto decimal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- vírgula. O decimal é uma vírgula.- nenhum. A saída não tem um decimal.- ponto. O decimal é um ponto. O padrão é nenhum.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
number_of_decimals	Preenche a parte decimal com zeros à direita para o tamanho indicado. O padrão é 0.

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sinal	Determina o sinal do número de saída. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - un_signed. Exclui um sinal, se presente. - leading_sign. Um sinal de mais ou menos é adicionado na frente do número de saída. - trailing_sign. Um sinal de mais ou menos é adicionado após o número de saída. - sinal negativo somente. Um sinal de menos será adicionado ao número, se o número for negativo. - como na origem. Não altera o sinal de entrada. O padrão é un_signed.
size_of_integer_part	Preenche a parte inteira com zeros à direita para o tamanho indicado. O padrão é 0.
unit_type	Define a unidade de medida após o número. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - cm - polegada - metro - mm - indefinido. Nenhuma unidade foi adicionada. O padrão é indefinido.

FromFloat

O transformador **FromFloat** converte um número de ponto flutuante de binário para uma representação de string ASCII. A conversão é realizada na codificação de entrada com a ordem por byte de entrada.

A tabela a seguir descreve as propriedades do transformador **FromFloat**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
tamanho	Determina o tamanho do número de entrada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - single_precision_32_bit - double_precision_64_bit Default is single_precision_32_bit.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

FromInteger

O transformador **FromInteger** converte um inteiro de binário em uma representação de string ASCII em decimal, octal ou hexadecimal. A conversão é realizada na codificação de entrada com a ordem por byte de entrada.

A tabela a seguir descreve as propriedades do transformador **FromInteger**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
com sinal	Determina se o número de saída tem um sinal. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. O número de saída tem um sinal. - Desmarcado. O número de saída não tem um sinal. O padrão é desmarcado.
tamanho	Define o tamanho em bytes da entrada binária. Os valores suportados são de 1 a 8. O padrão é 1.
to_base	Define a base da saída. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - decimal. Base 10. - hexadecimal. Base de 16 usando letras maiúsculas de A à F. - minúsculas hexadecimais. Base de 16 usando letras minúsculas de a à f. - octal. Base 8. O padrão é decimal.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

FromPackDecimal

O transformador **FromPackDecimal** converte um número decimal compactado em uma representação de string ASCII. A conversão é realizada na codificação de entrada com a ordem por byte de entrada.

A tabela a seguir descreve as propriedades do transformador **FromPackDecimal**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

FromSignedDecimal

O transformador **FromSignedDecimal** converte um número decimal com sinal em uma representação de string ASCII. A conversão é realizada na codificação de entrada com a ordem por byte de entrada.

A tabela a seguir descreve as propriedades do transformador **FromSignedDecimal**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
insert_sign_symbol	Define o sinal do número. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- após. Um sinal de mais ou menos é adicionado após o número de saída.- antes. Um sinal mais ou menos é adicionado na frente do número de saída.- nenhum. A saída não tem sinal. O padrão é "nenhum".
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

hebrewBidi

O transformador **hebrewBidi** reverte uma string gravada em idiomas da direita para a esquerda (RTL), como Hebraico e Árabe.

A entrada deve estar no formato RTL. A saída será da esquerda para a direita (LTR).

HebrewDosToWindows

O transformador **HebrewDosToWindows** converte documentos em Hebraico da página de código Hebraico do MS-DOS em página de código em Hebraico do Windows.

HebrewEBCDICOldCodeToWindows

O transformador **HebrewEBCDICOldCodeToWindows** converte texto em Hebraico do EBCDIC em página de código Windows-1255.

hebUniToAscii

O transformador **hebUniToAscii** converte texto em Hebraico em Unicode UTF-16 em página de código Windows-1255.

hebUtf8ToAscii

O transformador **hebUtf8ToAscii** converte texto em Hebraico em Unicode UTF-8 em página de código Windows-1255.

HtmlEntitiesToASCII

O transformador **HtmlEntitiesToASCII** converte entidades HTML em texto simples. Por exemplo, ele converte `©` ou `©` no símbolo de copyright (©).

A tabela a seguir descreve as propriedades do transformador **HtmlEntitiesToASCII**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Entidades Com Suporte

O transformador dá suporte às entidades ISO 8859-1 (Latin-1) que foram definidas na referência de HTML 4.0, <http://www.w3.org/TR/1998/REC-html40-19980424/sgml/entities.html>. As entidades com suporte incluem:

- `&`, `<`, `>`; e `"`; (`&` `<` `>` ", respectivamente)
- Códigos de caractere numéricos `�`; a `ÿ`
- Entidades para caracteres de Latin-1: ` `; = espaço incondicional, `©`; = copyright, etc.

O transformador não dá suporte a caracteres estendidos, isto é, códigos maiores que 255 ou caracteres não Latin-1.

Codificação de Saída para Caracteres ASCII em Maiúsculas

Se a saída do transformador contiver caracteres ASCII em maiúsculas, selecione uma codificação de saída que ofereça suporte aos caracteres, como Windows-1252 ou UTF-8.

Nota: Inclua um atributo de codificação na instrução de processamento de XML. Caso contrário, a ferramenta Developer poderá não exibir os caracteres.

HtmlProcessor

O transformador **HtmlProcessor** normaliza um espaço em branco de acordo com as convenções de HTML. Ele converte qualquer sequência de guias, quebras de linha e caracteres de espaço em um único caractere de espaço. Esse transformador opera em texto HTML e em qualquer outro tipo de texto. Você também pode usá-lo como um pré-processador de formato. Para obter mais informações, consulte [“Referência de Componentes de Pré-processador de Formato” na página 189](#).

InjectFP

O transformador **InjectFP** insere um ponto decimal em uma localização especificada em um número. Por exemplo, o transformador pode converter 12345 para 123.45.

A tabela a seguir descreve as propriedades do transformador **InjectFP**:

Propriedade	Descrição
digits_after_decimal_point	Determina o número de dígitos após o ponto decimal.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

InjectString

O transformador **InjectString** insere uma string no texto.

A tabela a seguir descreve as propriedades do transformador **InjectString**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
injection_place	Define o número de caracteres do início do texto onde a string é inserida.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
string_to_inject	Define a string para inserção.

InlineTable

O componente **InlineTable** define uma tabela de pesquisa no Script. Essa tabela é usada pelo transformador **LookupTransformer**.

A seguinte tabela descreve as propriedades do componente **InlineTable**:

Propriedade	Descrição
Entrada	Define um par de chave e valor .
key	Define uma string de entrada exclusiva.
match_case	Determina se a string key faz distinção entre maiúsculas e minúsculas. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selected. key faz distinção entre maiúsculas e minúsculas.- Cleared. key não faz distinção entre maiúsculas e minúsculas. O padrão é Desmarcado.
table	Define uma lista de componentes Entrada .
value	Define uma string de saída.

JavaTransformer

O transformador **JavaTransformer** executa um transformador personalizado que é implementado em Java.

A seguinte tabela descreve as propriedades do transformador **JavaTransformer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
java_class	Define o caminho da classe Java.
método	Define o método a ser executado.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Nota: Esse componente foi preterido. O editor do IntelliScript o exibe apenas para Scripts herdados. Não o use em novos Scripts. Em vez disso, crie um transformador Java personalizado. Para obter mais informações, consulte ["Desenvolvendo um Componente Personalizado" na página 432](#).

LookupTransformer

O transformador **LookupTransformer** procura um valor em uma tabela.

A seguinte tabela descreve as propriedades do transformador **LookupTransformer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
look_at	Define o tipo de tabela de pesquisa usada pelo transformador. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - DynamicTable. A propriedade table da propriedade look_at define um recipiente de dados que contém a tabela. Para obter mais informações, consulte "DynamicTable" na página 269. - InlineTable. A propriedade table da propriedade look_at define uma lista de componentes de Entrada, cada um contendo uma chave e um valor. Para obter mais informações, consulte "InlineTable" na página 277. - XMLLookupTable. A propriedade xml_file_name da propriedade look_at define o caminho e nome de um arquivo XML que define a tabela. Para obter mais informações, consulte "XMLLookupTable" na página 293. - [TableName]. Um DynamicTable, InlineTable ou XMLLookupTable definido no nível global do Script. O padrão é em branco.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Se você usar a mesma tabela de pesquisa repetidamente, considere definir um **InlineTable** ou um **XMLLookupTable** no nível global do Script. Dessa forma, é possível fazer referência à tabela por nome na propriedade **look_at**.

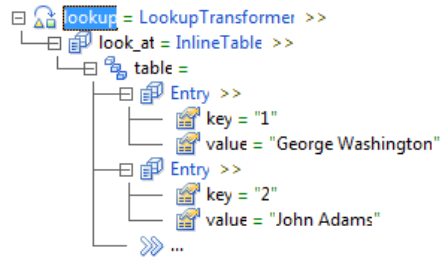
Por exemplo, você pode configurar um **LookupTransformer** para pesquisar valores na seguinte tabela:

Chave	Valor
1	George Washington
2	John Adams
3	Thomas Jefferson
4	James Madison

Se a entrada do transformador for 3, a saída será `Thomas jefferson`.

Definindo uma Tabela em Linha

Para definir uma tabela em linha, configure os pares de chave-valor no Script, como no exemplo a seguir:



Armazenando uma Tabela de Pesquisa XML em um Arquivo

Prepare um arquivo XML em conformidade com o esquema `lookupTableDefinition.xsd`. É possível localizar o esquema no subdiretório `\doc` do diretório de instalação. O seguinte documento XML é um exemplo:

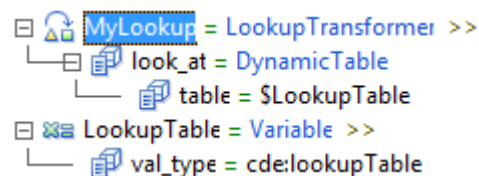
```
<?xml version="1.0" encoding="windows-1252" ?>
<lt:LookupTable xmlns:lt="http://www.itemfield.com/Engine/V4/lookupTable"
matchCase="false">
  <lt:Entry key="1" value="George Washington" />
  <lt:Entry key="2" value="John Adams" />
</lt:LookupTable>
```

Criando uma Tabela de Pesquisa XML Dinamicamente

Uma transformação pode criar uma tabela de pesquisa XML no tempo de execução. Por exemplo, a transformação pode executar um Analisador secundário que gera a estrutura XML.

A transformação deve armazenar a string XML em um recipiente de dados de várias ocorrências do tipo `cde:lookupTable`. Armazene cada par chave-valor em uma ocorrência do recipiente de dados. Por exemplo, você pode configurar um **RepeatingGroup** que contém uma ação **WriteValue**. Cada iteração do **RepeatingGroup** cria uma ocorrência do recipiente de dados e grava um par chave-valor na ocorrência.

Em seguida, configure um **LookupTransformer** com a opção de **DynamicTable** e especifique o recipiente de dados.



NormalizeClosingTags

Em entradas XML, o transformador **NormalizeClosingTags** remove marcas de fechamento abreviadas de elementos vazios. Ele muda `<tag/>` para `<tag></tag>`.

O transformador de XML não corrige XML incorreto. Ele converte XML corretos de um estilo de marca de fechamento para outro.

A tabela a seguir descreve as propriedades do transformador **NormalizeClosingTags**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RegularExpression

O transformador **RegularExpression** realiza uma pesquisa padrão no texto de entrada. Ele substitui instâncias do padrão por uma string especificada.

A tabela a seguir descreve as propriedades do transformador **RegularExpression**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
exp	Define uma expressão regular para o critério da pesquisa.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
substituição	Define o texto de substituição.

Por exemplo, suponha que uma âncora de **Conteúdo** recupere o texto seguinte:

```
transformer
```

Configure a âncora com um transformador **RegularExpression** que pesquisa o padrão `t.+s`. O padrão significa a letra `t`, seguida de um ou mais caracteres, seguidos pela letra `s`. Configure o transformador para substituir o padrão com o caractere `X`.

O padrão corresponde à substring `trans` da entrada. O transformador substitui a substring e as saídas:

```
Xformer
```

Sintaxe de Expressões Regulares

Uma expressão regular define um padrão de pesquisa de acordo com uma sintaxe padrão.

A transformação do Processador de Dados usa a implementação Regex++ de expressões regulares, © 1998-2003 do Dr. John Maddock, Versão 1.33, 18 de abril 2000.

Nota: A Regex++ não suporta localidades.

A tabela a seguir lista alguns caracteres especiais que você pode usar em expressões regulares:

Caractere	Significado	Exemplo
*	Corresponde a zero ou mais instâncias do caractere precedente.	<code>ab*c</code> corresponde a <code>ac</code> , <code>abc</code> ou <code>abbbc</code> .
?	Corresponde a zero ou uma instância do caractere precedente.	<code>ab?c</code> corresponde a <code>ac</code> ou <code>abc</code> .
+	Corresponde a uma ou mais instâncias do caractere precedente.	<code>a+</code> corresponde a <code>a</code> ou <code>aaaa</code> .
{}	Corresponde ao número especificado de instâncias do caractere precedente.	<code>ab{2}c</code> corresponde a <code>abbc</code> .
[]	Corresponde a qualquer item de um conjunto de caracteres.	<code>a[bst]c</code> corresponde a <code>abc</code> , <code>asc</code> ou <code>atc</code> .
-	Define um intervalo de caracteres dentro de colchetes.	<code>[A-Za-z]</code> corresponde a qualquer caractere do alfabeto em inglês. <code>[A-Za-zü]</code> corresponde a qualquer caractere do alfabeto em inglês ou a letra em alemão ü.
.	Corresponde a qualquer caractere.	<code>a.c</code> corresponde a <code>abc</code> , <code>a c</code> ou <code>a1c</code> .
^	Corresponde ao início do texto de entrada.	<code>^P.</code> corresponde a <code>Pe</code> , mas não corresponde a <code>Pi</code> em "Peter Piper".
\$	Corresponde ao fim da entrada de texto.	<code>r.\$</code> corresponde a <code>rs</code> em "Peter Piper's peppers".
	Corresponde a qualquer uma das duas expressões.	<code>abc ded</code> corresponde a <code>abc</code> ou <code>def</code> .
()	Agrupamento	<code>A(abc def)</code> corresponde a <code>Aabc</code> ou <code>Adef</code> .
\	Ignora um dos outros caracteres especiais, tratando-o como um caractere literal.	<code>\.</code> corresponde a um ponto literal, em vez de qualquer caractere.

Preservando Partes do Texto Original

Na propriedade `exp`, você pode colocar partes da expressão regular entre parênteses. Na propriedade `substituição`, você pode usar:

- `$0` para identificar todo o texto que corresponde à expressão regular
- `$1` para identificar a substring que corresponde à primeira parte entre parênteses da expressão regular
- `$2`, `$3`, e assim por diante, para identificar as substrings que correspondem à segunda, terceira, etc. parte entre parênteses

Por exemplo, suponha que você definiu:

```
exp = abc([0-9]+)(def)
replacement = $1
```

Isso substitui `abc5624def` por `5624`.

Como alternativa, suponha que você definiu:

```
exp = abc([0-9]+)(def)
replacement = $2ZYX$1
```

Isso substitui `abc5624def` por `defZYX5624`.

RemoveMarginSpace

O transformador **RemoveMarginSpace** exclui caracteres de espaço à esquerda e à direita do texto.

A tabela a seguir descreve as propriedades do transformador **RemoveMarginSpace**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RemoveRtfFormatting

O transformador **RemoveRtfFormatting** remove as instruções de formatação RTF do texto.

A tabela a seguir descreve as propriedades do transformador **RemoveRtfFormatting**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RemoveTags

O transformador **RemoveTags** remove marcas HTML do texto de entrada. Ele substitui as marcas em localizações internas do texto por uma string do separador, como um caractere de espaço. Ele não insere a

string do separador no início ou no fim do texto. Várias marcas adjacentes são transformadas em um único separador.

A tabela a seguir descreve as propriedades do transformador **RemoveTags**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
replace_with	Define a string do separador. O padrão é " " (espaço).

Substituir

O transformador **Substituir** encontra e substitui strings no texto de entrada. Deixar a propriedade **replace_with** vazia exclui o texto encontrado.

A tabela a seguir descreve as propriedades do transformador **Substituir**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
find_what	Define o texto para localizar. O valor é um dos seguintes componentes do pesquisador: <ul style="list-style-type: none">- NewlineSearch. Localiza um caractere de nova linha.- PatternSearch. Localiza o texto que corresponde a uma expressão regular.- SegmentSearch. Localiza um segmento de um marcador de abertura especificado para um marcador de fechamento.- TextSearch. Localiza uma string especificada. O padrão é TextSearch. Para obter mais informações, consulte "Referência de Componentes de Pesquisador" na página 245 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
ocorrência	Especifica quais ocorrências substituir: <i>todas</i> , <i>primeiras</i> ou <i>últimas</i> .

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
replace_with	Define a string de substituição.

Exemplo Online

Para ver um exemplo online, abra `samples\Projects\Transformers_Example\Transformers_Example.cmw`. A segunda e a quinta âncoras de **Conteúdo do Analisador** estão configuradas com transformadores de Substituição.

Redimensionar

O transformador **Redimensionar** ajusta o texto de entrada para um tamanho especificado. Ele preenche ou trunca o texto conforme necessário.

A tabela a seguir descreve as propriedades do transformador **Redimensionar**:

Propriedade	Descrição
alinhar	Define o alinhamento do texto dentro da string redimensionada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - esquerda. O preenchimento ou corte é feito à direita. - direita. O preenchimento ou corte é feito à esquerda.
allow_smaller_size	Quando selecionado, o transformador de Redimensionamento ajustará o texto de entrada a um tamanho especificado com preenchimento se a string for menor que o tamanho definido pelo parâmetro size . Se não for selecionado e a string de entrada for menor que o tamanho especificado, o transformador falhará.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
padding_character	Define o caractere de preenchimento. Digite o caractere ou selecione um recipiente de dados que contém um caractere.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
tamanho	Define o tamanho do texto de saída. Digite um número inteiro ou selecione um recipiente de dados que contém um número inteiro.

ReverseTransformer

O transformador **ReverseTransformer** reverte uma string. Por exemplo, ele transforma 1234 em 4321.

A tabela a seguir descreve as propriedades do transformador **ReverseTransformer**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RtfProcessor

O transformador **RtfProcessor** normaliza o código RTF. Ele também está disponível como um pré-processador de formato. Para obter mais informações, consulte ["Referência de Componentes de Pré-processador de Formato" na página 189](#).

RtfToASCII

O transformador **RtfToASCII** converte a entrada RTF em texto simples. Ele remove palavras de controle RTF do texto.

A seguinte tabela descreve as propriedades do transformador **RtfToASCII**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

SubString

O transformador **SubString** retorna uma substring de entrada, inicial e final nas localizações especificadas.

A seguinte tabela descreve as propriedades do transformador **SubString**:

Propriedade	Descrição
início	Define a localização inicial. 0 significa começar no início da entrada.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
fim	Define a localização final.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

ToFloat

O transformador **ToFloat** converte um número de ponto flutuante de uma representação de string ASCII para binário. A conversão é realizada na codificação de saída com a ordem de byte de saída.

A seguinte tabela descreve as propriedades do transformador **ToFloat**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
tamanho	Determina o tamanho do número de saída. A propriedade tamanho tem as seguintes opções: <ul style="list-style-type: none">- single_precision_32_bit- double_precision_64_bit Default is single_precision_32_bit.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

ToInteger

O transformador **ToInteger** converte um número de uma representação de string ASCII para um integer binário. A entrada da string pode ser decimal, octal ou hexadecimal. A conversão é realizada na codificação de saída com a ordem de byte de saída.

A seguinte tabela descreve as propriedades do transformador **ToInteger**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
from_base	Define a base da entrada. A propriedade to_base tem as seguintes opções: <ul style="list-style-type: none">- decimal. Base 10.- hexadecimal. Base de 16 usando letras maiúsculas de A à F.- minúsculas hexadecimais. Base de 16 usando letras minúsculas de a à f.- octal. Base 8. O padrão é decimal.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
com sinal	Determina se o número de entrada tem um sinal. A propriedade to_base tem as seguintes opções: <ul style="list-style-type: none">- Marcado. O número de saída tem um sinal.- Desmarcado. O número de saída não tem um sinal. O padrão é desmarcado.
tamanho	Define o tamanho em bytes da representação binária. Os valores suportados são de 1 a 8.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

ToPackDecimal

O transformador **ToPackDecimal** converte um número de uma representação de string ASCII para decimais compactados. A conversão é realizada na codificação de saída com a ordem de byte de saída.

A seguinte tabela descreve as propriedades do transformador **ToPackDecimal**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
não assinado	Determina se o decimal compactado é assinado. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. O decimal compactado não é assinado.- Desmarcado. O decimal compactado é assinado. O padrão é desmarcado.

Nota: Esse componente não oferece suporte à codificação de entrada UTF-8.

TransformationStartTime

O transformador **TransformationStartTime** retorna a data e a hora de início da execução da transformação.

O transformador copia a data e hora da variável *VarSystem* e formata a saída de acordo com a sua especificação.

A seguinte tabela descreve as propriedades do transformador **TransformationStartTime**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
formato	Define o formato de data e hora. Digite o formato ou selecione um recipiente de dados que contém um formato. Para obter mais informações sobre os formatos com suporte, consulte "DateFormatICU" na página 266 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

TransformByParser

O transformador **TransformByParser** executa um Analisador em seu texto de entrada. O Analisador deve conter componentes **FindReplaceAnchor** que marcam segmentos do texto para substituição. Quando o Analisador conclui a execução, o transformador executa as substituições.

A saída do transformador é o texto modificado. O Script ignora qualquer saída XML gerada pelo Analisador.

A seguinte tabela descreve as propriedades do transformador **TransformByParser**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
Analisador	Define o nome do Analisador.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Amostra Online

Para obter uma amostra online, abra <instalação>\client\DT\samples\Projects\TransformByParser\TransformByParser.cmw. A amostra usa **TransformByParser** para substituir cada instância da string ~NL~ por um retorno de carro, seguido por um avanço de linha.

Nota: As amostras ficam disponíveis somente quando você realiza uma instalação de cliente.

Para executar a amostra de TransformByParser:

1. Defina **MyTransformByParser** como o componente de inicialização.

2. Execute o transformador.
3. No prompt, selecione o arquivo de origem `Report.edi`.

O transformador armazena sua saída em `Results\Transformation of Report.edi`. Você pode comparar a saída com a origem no Bloco de Notas.

TransformByProcessor

O transformador **TransformByProcessor** executa um processador de documentos em sua entrada. A saída do transformador é a saída do processador de documentos. Para obter mais informações, consulte [“Visão Geral de Processadores de Documentos” na página 162](#).

A seguinte tabela descreve as propriedades do transformador **TransformByProcessor**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
processador	Define o nome do processador de documentos.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

TransformByService

O transformador **TransformByService** executa um serviço de transformação de Processador de Dados em sua entrada. A saída do transformador é a saída do serviço.

Se você usar o transformador para invocar um serviço de Analisador, a saída do transformador será uma string XML.

Nota: O transformador **TransformByService** oferece suporte para serviços de entrada única. Não o use com um serviço que possua várias portas de entrada.

A seguinte tabela descreve as propriedades do transformador **TransformByService**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
disable_automatic_encoding	Determina se o Script aplica as codificações de entrada e saída que estão definidas no serviço. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selected. O Script ignora as codificações de entrada e saída que estão definidas no serviço. - Cleared. O Script aplica as codificações de entrada e saída que estão definidas no serviço.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
parâmetros	Define uma lista de valores iniciais que o Script atribui a variáveis definidas no serviço. Em cada elemento da lista, especifique o nome de uma variável e seu valor.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
service_name	Define o nome do serviço que é executado na entrada.

TransformerPipeline

O transformador **TransformerPipeline** aplica uma sequência de transformadores aninhados à sua entrada.

A seguinte tabela descreve as propriedades do transformador **TransformerPipeline**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

XMLLookupTable

O componente **XMLLookupTable** especifica um arquivo XML que contém uma tabela de pesquisa. A tabela está em conformidade com o esquema `lookupTableDefinition.xsd` no subdiretório `\doc` do diretório de instalação. A tabela é usada pelo transformador **LookupTransformer**.

A seguinte tabela descreve as propriedades do componente **XMLLookupTable**:

Propriedade	Descrição
xml_file_name	Define o caminho e nome do arquivo XML.

O seguinte documento XML é válido em relação ao esquema:

```
<?xml version="1.0" encoding="windows-1252" ?>
<lt:LookupTable xmlns:lt="http://www.Itemfield.com/Engine/V4/lookupTable"
  matchCase="false">
  <lt:Entry key="1" value="George Washington" />
  <lt:Entry key="2" value="John Adams" />
</lt:LookupTable>
```

Se o atributo opcional **matchCase** for `true`, o atributo **chave** fará distinção entre maiúsculas e minúsculas.

XSLTTransformer

O transformador **XSLTTransformer** aplica uma transformação XSLT para o texto de entrada XML.

Por exemplo, você pode usar um Analisador para extrair dados de um documento XML. Uma âncora **Conteúdo** recupera uma ramificação completa, bem formada da árvore XML. Você pode configurar a âncora **Conteúdo** com um **XSLTTransformer** que executa uma transformação XSLT na ramificação.

A seguinte tabela descreve as propriedades do transformador **XSLTTransformer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
xslt_file	Define o caminho e nome do arquivo XSLT.

CAPÍTULO 18

Ações

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Ações, 295](#)
- [Propriedades de Ação Padrão, 296](#)
- [Referência do Componente da Ação, 297](#)
- [Referência do Subcomponente da Ação, 331](#)

Visão Geral de Ações

Ações são componentes que realizam operações nos dados que o Script extraiu de um documento de origem. Alguns exemplos de ações com suporte são:

- Cálculos aritméticos
- Concatenações de string
- Envio de formulários para um servidor da Web
- Ativando um Analisador, Serializador ou Mapeador secundário
- Consulta de um banco de dados

A transformação de Processador de Dados fornece muitas ações, e você pode definir ações personalizadas.

Como as Ações Funcionam

Uma ação usa a entrada dos recipientes de dados que estão disponíveis no momento. Uma única ação pode ter várias entradas.

Se a ação estiver incorporada em um Analisador, os recipientes de dados disponíveis serão aqueles que o Analisador gerou. Em um Serializador, os recipientes de dados são aqueles que existem na entrada XML, além de quaisquer recipientes de dados adicionais que o Serializador gerou. Para um Mapeador, os recipientes de dados podem estar na entrada ou na saída.

A ação executa operações na entrada e gera a saída. Você pode configurar muitas ações para armazenar a saída nos recipientes de dados.

Na maioria das ações, os recipientes de dados de entrada e saída devem ter tipos de dados simples. Eles não devem conter elementos aninhados. Algumas ações trabalham com recipientes de dados que contêm elementos aninhados, como recipiente de dados de ocorrência múltipla ou outros tipos de especiais.

Uma ação pode ter efeitos adicionais, como gravar em um arquivo, atualizar um banco de dados ou enviar dados para um aplicativo externo.

Comparação entre Ações e Transformadores

Algumas ações realizam operações que são similares aos transformadores, por exemplo, modificar uma string ou consultar um banco de dados. No entanto, as ações são diferentes dos transformadores em algumas formas fundamentais.

A tabela a seguir resume as diferenças:

Operação	Transformadores	Ações
Entrada	A entrada de um transformador é uma string única.	A entrada é implementada pela ação. Uma ação pode ter várias entradas. As entradas podem ser recipientes de dados.
Saída	A saída de um transformador é uma string.	A saída é implementada pela ação. Por exemplo, uma ação pode criar recipientes de dados de saída.
Efeitos colaterais	Um transformador não tem efeitos colaterais, a não ser modificar a string de entrada.	Uma ação pode ter efeitos colaterais, como atualizar um banco de dados.

Definindo Ações

Edite o Script para definir uma ação. Você pode inserir as ações na linha **contains** de componentes, como um **Analisador**, um **Serializador**, um **Mapeador**, um **Grupo** ou um **RepeatingGroup**. Basicamente, você pode inserir ações em qualquer localização na qual pode inserir âncoras, âncoras de serialização ou âncoras Mapeadoras.

As ações são executadas em sequência com as âncoras que você especifica na mesma localização. Em um analisador, é possível definir a propriedade **phase** de uma ação, que determina se ela é executada no estágio inicial, principal ou final do processo de análise. Para obter mais informações, consulte [“Fases de Pesquisa” na página 209](#).

Propriedades de Ação Padrão

A seguinte tabela descreve as propriedades padrão de ações:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência do Componente da Ação

As ações executam operações no sistema, por exemplo, fazem download de um arquivo de um local remoto ou validam um valor.

AddEventAction

A ação **AddEventAction** adiciona uma mensagem a log de eventos.

A tabela a seguir descreve as propriedades da ação **AddEventAction**:

Propriedade	Descrição
desativado	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
mensagem	Define a mensagem da string.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
gravidade	<p>Determina o nível de gravidade da mensagem. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - notificação - aviso - falha - erro fatal <p>O padrão é notificação.</p>

AggregateValues

A ação **AggregateValues** executa uma computação em uma agregação de um recipiente de dados de ocorrência múltipla.

A tabela a seguir descreve as propriedades da ação **AggregateValues**:

Propriedade	Descrição
aggregation_function	<p>Determina a função para executar na agregação. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - AllEqual. Retornará verdadeiro se os valores forem iguais ou falso se não forem iguais. - Contagem. Retorna o número de ocorrências do recipiente de dados. - Associação. Retorna uma lista de todos os valores, separados pelo separador especificado na propriedade separador. - Soma. Retorna a soma dos valores.
AllEqual	Define uma opção sob a propriedade aggregation_function .
Contagem	Define uma opção sob a propriedade aggregation_function .
data_holder	Define o recipiente de dados que armazena a saída.

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
include_empty_values	Determina se a agregação inclui ocorrências que não contêm dados. <ul style="list-style-type: none"> - Marcado. A ação inclui ocorrências vazias. - Desmarcado. A ação ignora as ocorrências vazias. O padrão é marcado.
Associação	Define uma opção sob a propriedade aggregation_function .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
root_element	Determina o recipiente de dados na raiz da ramificação de XML que contém o recipiente de dados de ocorrência múltipla. O root_element pode ser de ocorrência única ou de ocorrência múltipla.
sub_element	Determina o recipiente de dados de ocorrência múltipla para o qual a ação calcula uma agregação. Se sub_element não estiver atribuído, a ação calculará a agregação de root_element .
Soma	Define uma opção sob a propriedade aggregation_function .

Dependendo do **root_element** que você configura, a ação poderá agregar ocorrências em níveis diferentes da ramificação. Por exemplo, um documento XML tem a estrutura:

```
<Company>
  <Division name="America">
    <Employee>...<Employee>
    <Employee>...<Employee>
    <Employee>...<Employee>
  </Division>
  <Division name="Europe">
    <Employee>...<Employee>
    <Employee>...<Employee>
  </Division>
</Company>
```

Se o **root_element** for *Empresa* e você configurar a ação para contar ocorrências de *Funcionário*, a ação contará todos os elementos de *Funcionário* que forem descendentes de *Empresa*. A ação retornará 5.

Se o **root_element** for *Divisão*, a ação contará o número de ocorrências de *Funcionário* na *Divisão* que a transformação está atualmente processando. Quando a ação processar *América*, ela retornará 3. Quando processar *Europa*, ela retornará 2.

AppendListItems

A ação **AppendListItems** concatena strings em um recipiente de dados de ocorrência múltipla.

A tabela a seguir descreve as propriedades da ação **AppendListItems**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
entrada	Determina um recipiente de dados de ocorrência múltipla por entrada. O recipiente de dados deve ter um tipo de dados simples.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Determina o recipiente de dados que armazena a saída. O recipiente de dados deve ter um tipo de dados simples.

Propriedade	Descrição
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Para obter mais informações sobre a preparação de entrada para essa ação, consulte [“Mapeando para Recipientes de Dados de Ocorrência Múltipla” na página 206](#).

Exemplo

Um documento de origem contém o seguinte texto separado por espaço:

H E L L O

Ao analisar o documento, você deseja remover os espaços e armazenar o resultado em um elemento XML chamado *Saudação*.

Crie uma variável de ocorrência múltipla chamada *VarLetter*. Crie várias âncoras de *Conteúdo* que recuperam letras individuais e as armazena em ocorrências de *VarLetter*.

Em seguida, use a ação **AppendListItems** para concatenar as ocorrências de *VarLetter* e armazenar o resultado no elemento *Saudação*. O resultado é:

```
<Greeting>HELLO</Greeting>
```

Exemplo Online

Para obter um exemplo online dessa ação, abra o projeto `samples\Projects\AppendListItems\AppendListItems.cmw`. O exemplo usa um *RepeatingGroup* para armazenar valores em uma variável de ocorrência múltipla. Em seguida, ele usa como uma ação *AppendListItems* para concatenar os valores.

AppendValues

A ação **AppendValues** concatena strings.

A tabela a seguir descreve as propriedades da ação **AppendValues**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
entrada	Determina a lista de recipientes de dados que contém os valores a serem anexados. Os recipientes de dados devem ter um tipo de dados simples.

Propriedade	Descrição
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
saída	Determina o recipiente de dados que armazena a saída. O recipiente de dados deve ter um tipo de dados simples.
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
skip_unfound_values	<p>Determina se a ação continua quando um dos recipientes de dados de entrada estiver ausente.</p> <ul style="list-style-type: none"> - Marcado. A ação continua. - Desmarcado. A ação falha. <p>O padrão é marcado.</p>

Exemplo

Um Analisador gerou o seguinte XML:

```
<Name>
  <First>Ron</First>
  <Last>Lehrer</Last>
</Name>
```

Você pode configurar uma ação **AppendValues** para gerar essas saídas:

```
<FullName>Ron Lehrer</FullName>
```

CalculateValue

Calcula valores numéricos ou concatena valores de string.

Para calcular valores numéricos, use os seguintes operadores entre os parâmetros:

- +
- -
- *
- /

Você pode usar parênteses para explicar a expressão numérica. Você pode usar variáveis dos seguintes tipos de dados:

- xs:anyType
- xs:anySimpleType
- tipos de dados numéricos
- tipos de dados de string

Se os parâmetros forem todos de tipos de dados numéricos ou de strings numéricas, CalculateValue executará um cálculo aritmético. Os resultados de números não inteiros serão arredondado para 14 casas decimais.

Para concatenar strings, use o sinal do operador mais (+) entre parâmetros e strings.

A tabela a seguir descreve as propriedades da ação **CalculateValue**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
expressão	Define uma expressão de JavaScript. Para representar um parâmetro de entrada, use um sinal de dólar (\$) seguido por um número inteiro. Para representar uma string, coloque-a entre aspas simples.
failure_action	Determina o comportamento no caso de falha. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Ignorar. A transformação continua.- HaltExecution. A transformação é interrompida. O padrão é Ignorar.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
params	Define uma lista de recipientes de dados que contêm os parâmetros de entrada.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
resultado	Determina um recipiente de dados que armazena a saída.

Nota: Para obter mais informações sobre a sintaxe de JavaScript que a transformação do Processador de Dados oferece suporte, consulte ["EnsureCondition" na página 311](#). Para obter mais informações sobre a precisão dos valores de `xs:decimal` e `xs:float`, consulte ["Precisão de Dados Numéricos" na página 193](#).

Exemplo

Um Analisador gerou o seguinte XML:

```
<ItemOrdered>
  <Name>Gizmo</Name>
  <Quantity>100</Quantity>
  <Price>25</Price>
</ItemOrdered>
```

Você pode usar uma ação **CalculateValue** para gerar a saída:

```
<ItemOrdered>
  <Name>Gizmo</Name>
  <Quantity>100</Quantity>
  <Price>25</Price>
  <Total>2500</Total>
</ItemOrdered>
```

Defina os elementos `Nome` e `Quantidade` como parâmetros de entrada. Especifique a expressão de JavaScript `$1 * $2` e armazena o resultado no elemento `Total`.

Exemplo Online

Para obter um exemplo online dessa ação, abra o projeto `samples\Projects\CalculateValue\CalculateValue.cmw`. O exemplo recupera três números de um documento de origem e os armazena em variáveis. Ele usa uma ação `CalculateValue` para computar uma função matemática de números.

CombineValues

A ação **CombineValues** concatena strings.

A entrada é uma lista de recipiente de dados e variáveis. A saída é um recipiente de dados de ocorrência múltipla.

Se a entrada for um recipiente de dados de ocorrência múltipla, a ação **CombineValues** gerará um iteração para cada instância do recipiente de dados. Em cada iteração, a ação **CombineValues** combina todos os recipientes de dados de entrada e grava a saída em uma instância do recipiente de dados de saída.

A tabela a seguir descreve as propriedades da ação **CombineValues**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
entrada	Define uma lista de recipientes de dados para entrada. Os recipientes de dados devem ter um tipo de dados simples.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Determina o recipiente de dados de ocorrência múltipla onde a ação armazena a saída. O recipiente de dados deve ter um tipo de dados simples.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo

Em uma variável de ocorrência múltipla chamada *VarDay*, você armazenou a lista Segunda-feira, Terça-feira. Em uma variável de ocorrência múltipla chamada *VarTime*, você armazenou manhã, tarde. Em uma variável de ocorrência única chamada *VarSpace*, você armazenou um caractere de espaço.

Suponha que você execute **CombineValues** em *VarDay*, *VarSpace* e *VarTime* com um recipiente de dados de saída chamado **DayTime**. A saída é

```
<DayTime>Monday morning</DayTime>
<DayTime>Monday afternoon</DayTime>
<DayTime>Tuesday morning</DayTime>
<DayTime>Tuesday afternoon</DayTime>
```

Exemplo Online

Para obter um exemplo online dessa ação, abra o projeto `samples\Projects\CombineValues\CombineValues.cmw`. O exemplo recupera listas de dias, meses e anos de um documento de origem. Ela usa uma ação **CombineValues** para gerar todas as possíveis datas da lista.

CreateList

A ação **CreateList** insere dados em uma lista. A saída é um recipiente de dados de ocorrência múltipla contendo a lista. Para obter mais informações, consulte [“Recipientes de Dados de Ocorrência Múltipla” na página 201](#).

Aninhado nesse componente, digite os valores de dados.

A tabela a seguir descreve as propriedades da ação **CreateList**:

Propriedade	Descrição
data_holder	Define o recipiente de dados de ocorrência múltipla onde a ação armazena a lista. O recipiente de dados deve ter um tipo de dados simples.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .

Propriedade	Descrição
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Exemplo

Se os valores dos dados de entrada valores forem

```
Jack
Jennie
Larissa
```

a ação poderá criar a seguinte saída:

```
<Name>
  <First>Jack</First>
  <First>Jennie</First>
  <First>Larissa</First>
</Name>
```

CustomLog

A ação **CustomLog** pode ser usada como o valor da propriedade **on_fail**. Quando ocorrer uma falha, a ação **CustomLog** executará um serializador que prepara uma mensagem de log. O sistema grava a mensagem em uma localização de saída especificada.

A tabela a seguir descreve as propriedades da ação **CustomLog**:

Propriedade	Descrição
run_serializer	Determina o serializador que prepara a mensagem de log. Defina um serializador nesse local ou digite o nome de um serializador definido globalmente.
saída	Determina a localização da saída. A propriedade saída tem as seguintes opções: <ul style="list-style-type: none"> - OutputDataHolder. Grava em um recipiente de dados. - OutputFile. Grava em um arquivo. - OutputPort. Define o nome de um AdditionalOutputPort onde os dados são gravados. - ResultFile. Grava o arquivo de resultados padrão da transformação. - StandardErrorLog. Grava no log do usuário. O padrão é StandardErrorLog. Para obter mais informações sobre essas opções, consulte "Referência do Subcomponente da Ação" na página 331 e "Tratamento de Falhas" na página 399 .

Para obter mais informações sobre a propriedade **on_fail**, consulte ["Tratamento de Falhas" na página 399](#).

DateAddICU

A ação **DateAddICU** incrementa uma data.

A tabela a seguir descreve as propriedades da ação **DateAddICU**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
input_date	Define a data a ser incrementada.
input_format	Define uma string ou um recipiente de dados que define o formato de data, por exemplo, dd/MM/yy. Para obter mais informações, consulte "DateFormatICU" na página 266 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
num_of_days	Define um inteiro positivo ou negativo ou um recipiente de dados que contém o número de dias para adicionar.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Determina o recipiente de dados que armazena a data de saída.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

DateDiffICU

A ação **DateDiffICU** calcula a diferença entre duas datas.

A tabela a seguir descreve as propriedades da ação **DateDiffICU**:

Propriedade	Descrição
data1	Define uma string ou recipiente de dados que define a primeira data.
date2	Define uma string ou recipiente de dados que define a segunda data.
date_format1	Define uma string ou um recipiente de dados que define o formato da primeira data, por exemplo, dd/MM/yy. Se você omitir o formato, o padrão do sistema será usado. Para obter mais informações, consulte "DateFormatICU" na página 266 .
date_format2	Define uma string ou recipiente de dados que define o formato da segunda data.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Define o recipiente de dados que armazena o resultado.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

DownloadFileToDataHolder

A ação **DownloadFileToDataHolder** baixa um arquivo de um servidor da Web e armazena o seu conteúdo em um recipiente de dados. A ação converte símbolos em entidades XML.

A tabela a seguir descreve as propriedades da ação **DownloadFileToDataHolder**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
file_url	Determina um recipiente de dados que armazena a URL do arquivo.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Determina o recipiente de dados que armazena o conteúdo baixado.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

DumpValues

A ação **DumpValues** é uma ferramenta de depuração. Ela grava dados em um elemento `<DumpValues>...</DumpValues>`. Defina os recipiente de dados que você deseja descartar aninhando-os como componentes filhos.

A tabela a seguir descreve as propriedades da ação **DumpValues**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Determina o recipiente de dados que armazena o resultado. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - OutputFile - ResultFile - StandardErrorLog O padrão é ResultFile.
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

EnsureCondition

A ação **EnsureCondition** avalia uma expressão booliana JavaScript. Se a expressão for *falsa*, a ação falhará.

A tabela a seguir descreve as propriedades da ação **EnsureCondition**:

Propriedade	Descrição
condição	Define uma expressão de JavaScript para avaliação. Na expressão, consulte os parâmetros definidos em params com um sinal de dólar (\$) seguido por um número inteiro. Por exemplo, a seguinte expressão verifica se o primeiro parâmetro tem o valor Ron Lehrer: <pre>\$1 == "Ron Lehrer"</pre>
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.

Propriedade	Descrição
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
params	Define uma lista de recipientes de dados.
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Sintaxe de JavaScript Padrão

O processador de JavaScript dá suporte às expressões JavaScript padrão que contêm os recursos a seguir.

- Os operadores unário e binário:

```
() + - * / % == != < <= > >= && ||
```

- O ternário `?:Operador` .

- Os seguintes métodos:

```
charAt
indexOf
lastIndexOf
length
join
substring
toString
```

Se você aplicar estes métodos a um literal que tem um tipo de dados simples, coloque-o entre parênteses, por exemplo:

```
123.toString();           //Wrong
(123).toString();         //Right

"Hello, World".substring(3,7); //Wrong
("Hello, World").substring(3,7); //Right
```

- As seguintes funções:

```
Math.ceil
Math.floor
Math.max
Math.min
Math.pow
Math.sqrt
```



```
parseFloat  
parseInt
```

O processador de JavaScript não dá suporte a recursos, como os seguintes:

- Os operadores unário e binário:

```
++ -- typeof void >> >>> << === !== ~ & | ^
```

- Operadores de Atribuição:

```
= += -= *= /= >>= >>>= <<= &= |= ^=
```

- O operador de vírgula (,).
- Os valores NaN, nulo, infinito ou -0 (0 negativo).
- Os tipos de dados diferentes de string, número e booleano.
- O objeto `Data`.
- A função `equalsIgnoreCase`.

Extensões JavaScript

O processador de JavaScript implementa os seguintes métodos que não estão definidos no JavaScript padrão. É possível usar essas extensões em qualquer localização na qual o Script aceite uma expressão JavaScript.

A maioria das funções é uma implementação JavaScript de transformadores ou ações.

```
extra.sum(number1, number 2, ...)
```

Retorna a soma dos parâmetros.

```
extra.allSame(param1, param2, ...)
```

Retorna true quando todos os parâmetros têm o mesmo valor.

```
lookup.<nome_pesquisa>(chave)
```

Essa função acessa uma tabela de pesquisa global por nome.

No Script, defina um `InlineTable` ou `XMLLookupTable` global. Em seguida, em uma expressão JavaScript, você pode acessar a tabela. Por exemplo, se você definir um `InlineTable` global denominado `USPresidents`, `lookup.USPresidents(1)` retornará `George Washington`.

Para obter mais informações, consulte [“LookupTransformer” na página 278](#).

```
extra.formatDate(date, input_format, output_format)
```

Formata uma data ou hora. Para obter mais informações, consulte [“DateFormatICU” na página 266](#).

```
extra.substitute(text, oldSubstring, newSubstring, useRegex)
```

Substitui todas as instâncias de uma substring por outra substring. Se `useRegex` for true, o método interpretará `oldSubstring` como uma expressão regular.

```
extra.formatNumber(number, size_of_integer_part, number_of_decimals, sign,  
insert_decimal_point, unit_type)
```

Formata um número. Para obter mais informações, consulte [“FormatNumber” na página 271](#).

```
extra.insertString(text, injections_place, string_to_inject)
```

Insere uma string no texto. Para obter mais informações, consulte [“InjectString” na página 277](#).

```
extra.formatTransformationStartTime(format)
```

Processa a saída da data e/ou da hora em que a transformação começou a ser executada. Para obter mais informações, consulte [“TransformationStartTime” na página 289](#).

```
extra.resize(text, size, padding_character, align)
```

Ajusta o texto de entrada a um tamanho especificado, preenchendo ou truncando conforme necessário. Para obter mais informações, consulte [“Redimensionar” na página 285](#).

```
extra.dateAdd(date_format, date, days_to_add)
```

Incrementa uma data com base em um determinado número de dias. Para obter mais informações sobre `date_format`, consulte [“DateFormatICU” na página 266](#).

```
extra.dateAddMonths(date_format, date, months_to_add)
```

Incrementa uma data com base em um determinado número de meses. Para obter mais informações sobre `date_format`, consulte [“DateFormatICU” na página 266](#).

```
extra.dateAddYears(date_format, date, years_to_add)
```

Incrementa uma data com base em um determinado número de anos. Para obter mais informações sobre `date_format`, consulte [“DateFormatICU” na página 266](#).

```
extra.dateDiff(date_format1, date1, date_format2, date2)
```

Calcula a diferença entre duas datas. Para obter mais informações, consulte [“DateDiffICU” na página 308](#).

```
extra.createGuid(0)
```

Gera um identificador de GUID. Para obter mais informações, consulte [“CreateGuid” na página 265](#). É necessário fornecer um valor de parâmetro como 0.

```
extra.upper(text), extra.lower(text), extra.capitalize(text)
```

Muda o texto para todas as letras maiúsculas, todas as letras minúsculas ou apenas a primeira letra maiúscula. Para obter mais informações, consulte [“ChangeCase” na página 265](#).

```
extra.rtl2ltr(text)
```

Inverte uma string gravada em um idioma da direita para a esquerda, exibindo-a da esquerda para a direita. Para obter mais informações, consulte [“hebrewBidi” na página 275](#).

```
extra.trim(text)
```

Exclui do texto os caracteres de espaço à esquerda e à direita. Para obter mais informações, consulte [“RemoveMarginSpace” na página 283](#).

ExcludeItems

A ação **ExcludeItems** remove valores especificados de um recipiente de dados de ocorrência múltipla. O tipo de recipiente de dados deve ser simples. Para excluir strings específicas do recipiente de dados, defini-as como componentes filho. Para obter mais informações, consulte [“Recipientes de Dados de Ocorrência Múltipla” na página 201](#).

A tabela a seguir descreve as propriedades da ação **ExcludeItems**:

Propriedade	Descrição
<code>data_holder</code>	Define um recipiente de dados de ocorrência múltipla.
<code>desativado</code>	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .

Propriedade	Descrição
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Mapa

A ação **Map** copia um valor de um recipiente de dados para outro.

A tabela a seguir descreve as propriedades da ação **Mapa**:

Propriedade	Descrição
desativado	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209. O padrão é principal.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
origem	Determina o recipiente de dados de origem.
source_default	Determina o recipiente de dados de origem padrão.
destino	Determina o recipiente de dados de destino.
transformadores	Define uma sequência de transformadores que modificam o valor. Não atribua essa propriedade se a origem e o destino forem elementos XML complexos.
validadores	Define uma lista de validadores aplicados aos dados de origem. Para obter mais informações, consulte "Validadores" na página 402 .

Quando você copia um recipiente de dados que tem um tipo de dados simples, a origem e destino devem ter tipos de dados compatíveis. A ação pode aplicar transformadores com o valor copiado.

Quando você copiar um recipiente de dados de ocorrência múltipla que tem um tipo simples e a ação não for localizado dentro de um componente em iteração, como um **RepeatingGroup**, a ação copiará todas as ocorrências do recipiente de dados.

Quando você copia um recipiente de dados que tem um tipo de dados complexos, a origem e destino devem ter estruturas internas idênticas e tipos de dados idênticos. A ação copia os elementos e atributos aninhados.

Exemplo Online

Para obter um exemplo online dessa ação, abra o projeto `samples\Projects\CopyValue\CopyValue.cmw`. O exemplo usa uma ação `Mapa` para copiar um elemento complexo que contém um atributo e elementos aninhados.

Notificar

A ação **Notificar** dispara uma notificação. Use-a para inserir uma mensagem de aviso na saída da transformação.

A seguinte tabela descreve as propriedades da ação **Notificar**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notify	Define uma notificação. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- MandatoryStructureMissing. Um registro obrigatório não é exibido na entrada.- MismatchIDs. Os IDs do registro e do subelemento correspondem parcialmente.- StructureBelowMinOccurs. Há menos registros correspondentes do subelemento do que o definido em minOccurs.- StructureExceedsMaxOccurs. Há mais registros correspondentes do subelemento do que o definido em maxOccurs.- StructureOutOfSequence. Os registros correspondem aos subelementos, mas não na sequência necessária.- UnexpectedRecord. Os registros correspondem aos subelementos, mas não na hierarquia necessária.- UnrecognizedRecord. Nenhum subelemento corresponde a um dos identificadores de registro.- XsdValidationError. A entrada não corresponde aos requisitos do esquema.- Componente de Notificação ou NotificationGroup definido pelo usuário. Mensagem definida pelo usuário. Para obter mais informações, consulte "Notificações" na página 244 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é final.
valor	Define um valor a ser atribuído à variável <i>VarNotificationDetails/Value</i> . Um NotificationHandler pode incluir o valor em sua saída.

ResetVisitedPages

A ação **ResetVisitedPages** limpa a lista de páginas visitadas de analisadores secundários especificados. Ela permite várias visitas à mesma página, mesmo se **reject_recurring_pages** estiver selecionado. Use essa ação ao postar diferentes dados de entrada na mesma página da Web. Essa ação é usada com a propriedade **reject_recurring_pages** componente **Analisador**.

A tabela a seguir descreve as propriedades da ação **ResetVisitedPages**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
analisadores	Define uma lista de Analisadores. A lista de páginas visitadas de cada Analisador é redefinida.
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RunMapper

A ação **RunMapper** executa um Mapeador como um subcomponente de um Analisador, um Mapeador ou um Serializador.

A seguinte tabela descreve as propriedades da ação **RunMapper**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
Entrada	Define um recipiente de dados que contém o texto XML no qual executar o mapeador. O recipiente de dados deve ter um tipo de dados simples, como <code>xs:string</code> . O valor da string pode ser um texto XML de qualquer complexidade. Para obter mais informações sobre como executar um mapeador em um recipiente de dados que possui um tipo complexo, consulte “EmbeddedMapper” na página 356 . Se você omitir essa propriedade, o mapeador usará os recipientes de dados disponíveis no escopo da ação. Por exemplo, se a ação estiver aninhada em um Analisador, o Mapeador será executado na saída desse Analisador. Se a ação estiver dentro de um Grupo , ela será executada na saída do Grupo .
mapeador	Define o Mapeador. Selecione o nome de um componente Mapeador existente ou defina um componente Mapeador nessa localização do Script. Para obter mais informações, consulte “Referência de Componente de Mapeador” na página 353 .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

RunMapplet

A ação **RunMapplet** executa um mapplet. A saída do **RunMapplet** é lida no recipiente de dados especificado na ação RunMapplet.

Use a ação **RunMapplet** para realizar tarefas como mascaramento de dados, qualidade de dados, pesquisa de dados e outras atividades normalmente relacionadas à transformação relacional, sem a necessidade de converter dados no formato relacional e, em seguida, voltar a converter no formato hierárquico.

Nota: A ação RunMapplet somente pode ser usada para chamar mapplets passivos.

Os parâmetros de saída devem ser especificados na mesma ordem em que aparecem nas portas de destino do mapplet.

A seguinte tabela descreve as propriedades da ação **RunMapplet**:

Propriedade	Descrição
inputs	Especifica os valores de entrada a serem transmitidos ao mapplet. Os parâmetros de entrada devem ser especificados na mesma ordem em que aparecem nas portas de destino do mapplet.
mapplet_name	Indica o mapplet a ser executado. Nota: Primeiro adicione uma referência na guia Referências para qualquer mapplet que você desejar chamar com a ação RunMapplet. Para obter mais informações, consulte "Referências" na página 29 .
outputs	Especifica o local para armazenar os valores de saída retornados do mapplet. Os parâmetros de saída devem ser especificados na mesma ordem em que aparecem nas portas de destino do mapplet. No parâmetro OutputLocation , especifique os seguintes valores: <ul style="list-style-type: none">- data_holder: especifica o local para armazenar os valores que o mapplet retorna.- initialization: quando você testa a transformação, a transformação do Processador de Dados não executa o mapplet chamado pela ação RunMapplet. Você pode especificar um valor a ser usado como uma cadeia de saída temporária ao testar a transformação durante o tempo de design.

RunParser

A ação **RunParser** executa um Analisador secundário. A saída de **RunParser** é acrescentada à saída do componente principal que a ativou, como um Analisador ou um Serializador.

Use a ação **RunParser** para seguir os links em um arquivo HTML e executar um Analisador secundário nos destinos do link. Em um serializador, você pode usá-la para analisar partes de dados não estruturados na entrada.

Observe a diferença entre a ação **RunParser** e a âncora **EmbeddedParser**:

- **RunParser** analisa uma nova origem.
- **EmbeddedParser** analisa uma seção de uma origem existente.

A seguinte tabela descreve as propriedades da ação **RunParser**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
exclude_strings	Define as strings que não devem existir em input_source . Se uma string especificada estiver presente, a ação RunParser não acessará a origem nem ativará o Analisador secundário.
include_strings	Define as strings que devem estar presentes no valor input_source . Se uma string especificada estiver ausente, a ação RunParser não acessará a origem nem ativará o Analisador secundário.
input_source	Define um retentor de dados que contém um dos seguintes objetos: <ul style="list-style-type: none"> - Se input_source_as_text estiver selecionada, input_source conterá uma string. - Se input_source_as_text estiver desmarcada, input_source conterá o caminho e o nome de arquivo do documento de entrada. O padrão é a variável de sistema <i>VarLinkURL</i> .
input_source_as_text	Determina o tipo de dados no retentor de dados input_source . <ul style="list-style-type: none"> - Marcado. input_source contém uma string de texto. - Desmarcado. input_source contém um caminho de arquivo. O padrão é limpo.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
next_Parser	Define o nome do Analisador a ser executado. Uma chamada recursiva para o mesmo Analisador é permitida.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.
pre_processor	Define um processador de documento a ser aplicado na origem após o processador de documento definido no AdditionalInputPort > pre_processor associado.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
tentativas	Define o número de tentativas caso a solicitação falhe. O padrão é 0.
seconds_to_wait	Define o intervalo em segundos entre as tentativas. O padrão é 60.

Exemplo

Um arquivo HTML tem um link para um segundo arquivo. Uma âncora de **conteúdo** armazena o caminho do arquivo de destino do link na variável de sistema *VarLinkURL*. A ação **RunParser** acessa o arquivo de destino e executa um Analisador secundário nele.

Em outro exemplo, o Analisador principal contém uma âncora de **Alternativas** que seleciona um Analisador secundário de acordo com texto no documento de origem. Para obter mais informações, consulte ["Alternativas" na página 216](#).

RunPCWebService

A ação **RunPCWebService** executa um applet *_PowerCenter* em uma transformação de Processador de Dados. A saída do **RunPCWebService** é lida no recipiente de dados especificado na ação **RunPCWebService**.

Nota: A ação **RunPCWebService** é usada para chamar os applets passivos.

Os parâmetros de saída devem ser especificados na mesma ordem em que aparecem nas portas de destino do applet.

A seguinte tabela descreve as propriedades da ação **RunPCWebService**:

Propriedade	Descrição
wsdl	Especifique o WSDL para o serviço da Web a ser executado.
operationName	Indica a operação a ser executada. Selecione uma operação.
inputs	Especifica os valores de entrada a serem transmitidos ao applet. Os parâmetros de entrada devem ser especificados na mesma ordem em que aparecem nas portas de destino do applet.
outputs	Especifica o local para armazenar os valores de saída retornados do applet. Os parâmetros de saída devem ser especificados na mesma ordem em que aparecem nas portas de destino do applet.

RunSerializer

A ação **RunSerializer** executa um Serializador como subcomponente de um Analisador, um Mapeador ou um Serializador. A saída do serializador é armazenada em um recipiente de dados.

A seguinte tabela descreve as propriedades da ação **RunSerializer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
Entrada	Define um recipiente de dados que contenha o texto XML no qual o Serializador é executado. O recipiente de dados deve ter um tipo de dados simples, como <code>xs:string</code> . O valor da string pode ser um texto XML de qualquer complexidade. Para obter mais informações sobre como executar um Serializador em um recipiente de dados que tenha um tipo complexo, consulte “EmbeddedSerializer” na página 344 . Se você omitir essa propriedade, o Serializador usará os recipientes de dados disponíveis no escopo da ação. Por exemplo, se a ação estiver aninhada em um Analisador, o Serializador será executado na saída desse Analisador. Se a ação estiver dentro de um Grupo , ela será executada na saída do Grupo .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
output	Define um recipiente de dados para a saída do Serializador.
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 . O padrão é main.

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
serializador	Define um Serializador. Selecione o nome de um componente Serializador existente ou defina um Serializador nesta localização do Script. Para obter mais informações, consulte "Referência de Componente Serializador" na página 338 .

Exemplo Online

Para obter um exemplo online dessa ação, abra o projeto `samples\Projects\RunSerializer\RunSerializer.cmw`.

Para observar como o exemplo funciona, defina `MainParser` como o componente de inicialização e execute-o. `MainParser` contém um `RepeatingGroup` que analisa pares de nomes e os armazena em variáveis. Depois de cada iteração, `RepeatingGroup` executa uma ação de `RunSerializer` que concatena as variáveis com alguns textos pré-definidos. A ação armazena sua saída em um elemento XML que é adicionado à saída do Analisador.

RunXMap

A ação **RunXMap** executa um objeto XMap como um subcomponente de um analisador, um mapeador ou um serializador.

A seguinte tabela descreve as propriedades da ação **RunXMap**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
input	A entrada do objeto XMap. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Retentor de Dados. Use um tipo complexo que deve corresponder ao tipo de entrada do XMap. - String XML. A string XML é a entrada ou um retentor de dados do tipo simples como <code>xs:string</code>. O tipo de raiz XML deve corresponder ao tipo de entrada XMap. Se você não usar essa propriedade, o objeto XMap usará a entrada padrão.
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
output	Um recipiente de dados de um tipo complexo que deve corresponder ao tipo de saída do XMap. Se você omitir esta propriedade, o objeto XMap gravará em sua definição de saída padrão.
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
xmap	Nome do XMap. Você pode escolher o nome de um objeto XMap existente.

SetValue

A ação **SetValue** coloca conteúdo predefinido em um retentor de dados. Se o retentor de dados for um retentor de dados de ocorrência única, o conteúdo existente será substituído. Se o retentor de dados for um retentor de dados de múltiplas ocorrências, o definido conteúdo será anexado ao final.

A seguinte tabela descreve as propriedades da ação **SetValue**:

Propriedade	Descrição
data_holder	Define um retentor de dados que recebe a saída.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
aspas	Define uma string para o conteúdo do retentor de dados.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que são aplicados ao conteúdo antes dele ser salvo no retentor de dados.

Classificar

A ação **Classificar** classifica as ocorrências de um retentor de dados de múltiplas ocorrências. A saída do conteúdo original do retentor de dados. A classificação diferencia maiúsculas de minúsculas.

Se você executar a ação em um elemento XML que contém atributos ou elementos aninhados, você poderá usá-los como chaves de classificação.

A seguinte tabela descreve as propriedades da ação **Classificar**:

Propriedade	Descrição
by_fields	Define uma lista de chaves de classificação em ordem decrescente de precedência. Para cada campo, selecione o retentor de dados e uma classificação crecente ou decrescente . Você pode selecionar retentor de dados de múltiplas ocorrências em si ou seus elementos ou atributos aninhados. Para classificar numericamente, uma chave de classificação deve ter um tipo de dados numéricos, como <code>xs:integer</code> .
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399.</p>
fase	<p>Determina quando o Script processa o componente. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. <p>Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209. O padrão é principal.</p>
recurring_element	Define um retentor de dados de múltiplas ocorrências para ser classificado.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Limitação

Você não poderá usar a ação **Classificar** se uma **Chave** estiver definida no recipiente de dados de ocorrência múltipla. Para obter mais informações, consulte [“Visão Geral de Localizadores, Chaves e Indexação” na página 360](#).

ValidateValue

A ação **ValidateValue** valida os dados XML de acordo com um conjunto de regras definido em um objeto Regras de Validação. Se os dados violarem as regras, a ação salvará um relatório de validação em um retentor de dados.

A entrada da ação é um retentor de dados. Se o retentor de dados for a raiz de uma ramificação XML, a ação analisará toda a ramificação.

A seguinte tabela descreve as propriedades da ação **ValidateValue**:

Propriedade	Descrição
disabled	<p>Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. <p>O padrão é Cleared.</p>
errors_found	Define um recipiente de dados que conta o número de violações de regras de validação na entrada.

Propriedade	Descrição
errors_output	Define um recipiente de dados no qual a ação armazena o relatório de erros de validação de XML. Se o recipiente de dados tiver o tipo <code>xs:string</code> , a saída será uma cadeia que contém marcas XML. Se o recipiente de dados tiver o tipo <code>cde:validationErrors</code> , a saída será uma estrutura que contém os recipientes de dados aninhados.
input	Define o recipiente de dados de entrada cuja conformidade com as Regras de Validação a ação analisa.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

WriteValue

A ação **WriteValue** grava o valor de um retentor de dados para uma localização, como um arquivo, ou para um retentor de dados de tipo de string.

Se o retentor de dados de entrada for um elemento XML, a ação gravará o elemento e quaisquer elementos e atributos aninhados.

A seguinte tabela descreve as propriedades da ação **WriteValue**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
entrada	Define o retentor de dados do qual será gravado.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
no_tags	Determina se o resultado está cercado por marcas XML. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. As marcas XML são omitidas. Isso é apropriado apenas se a entrada for um retentor de dados simples, que não contém elementos ou atributos aninhados. - Desmarcado. O resultado está rodeado por marcas XML. Este é o padrão. O padrão é desmarcado.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
saída	Define a localização de saída. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - OutputDataHolder. Grava em um recipiente de dados. - OutputFile. Grava em um arquivo. Selecione Sincronizar para bloquear o arquivo de forma a anexar dados para o mesmo arquivo. - OutputPort. Define o nome de um AdditionalOutputPort onde os dados são gravados. - ResultFile. Grava o arquivo de resultados padrão da transformação. - StandardErrorLog. Grava no log do usuário. Para obter mais informações, consulte "Tratamento de Falhas" na página 399. O padrão é ResultFile. Para obter mais informações sobre essas opções, consulte "Referência do Subcomponente da Ação" na página 331 .
fase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - initial. O Script processa o componente durante a fase inicial. - main. O Script processa o componente durante a fase principal. - final. O Script processa o componente durante a fase final. Para obter mais informações, consulte "Como um Analisador Procura Âncoras" na página 209 . O padrão é principal.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que modificam o valor antes de gravar. A entrada para os transformadores é o retentor de dados de entrada completo, incluindo marcas XML.

Amostra Online

Para obter uma amostra online dessa ação, abra o projeto `samples\Projects\Splitter\Splitter.cmw`.

A amostra demonstra como dividir um arquivo em dois. Um Analisador usa um **RepeatingGroup** para recuperar os registros de um arquivo HL7. Ele usa uma ação **Mapear** para criar nomes de arquivos exclusivos para cada registro e uma ação **WriteValue** para gravar os registros nesses arquivos. Os arquivos de saída, `MyOutput1.txt` e `MyOutput2.txt`, são armazenados na pasta `Results` do projeto.

Nota: É possível usar um streamer para dividir entradas grandes. Para obter mais informações, consulte [“Visão Geral de Streamers” na página 377](#).

XSLTMap

A ação **XSLTMap** executa uma transformação XSLT. A entrada e a saída são ramificações de um documento XML. Elas podem ser a saída de um Analisador ou a entrada de um Serializador.

A seguinte tabela descreve as propriedades da ação **XSLTMap**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
Entrada	Define um retentor de data que contém o elemento XML para a transformação.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
output	Define um retentor de dados para armazenar a saída.
phase	Determina quando o Script processa o componente. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- initial. O Script processa o componente durante a fase inicial.- main. O Script processa o componente durante a fase principal.- final. O Script processa o componente durante a fase final. Para obter mais informações, consulte “Como um Analisador Procura Âncoras” na página 209 . O padrão é main.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
xslt_file	Define o arquivo XSLT.

Exemplo

Suponha que o seguinte XML seja o resultado de um Analisador:

```
<Person>
  <First>Ron</First>
  <Last>Lehrer</Last>
</Person>
```

Com um arquivo a XSLT apropriado, você pode usar a ação **XSLTMap** para convertê-lo em:

```
<Person Name="Lehrer, Ron" />
```

Referência do Subcomponente da Ação

Os subcomponentes de ação servem como os valores de determinadas propriedades de ações.

OutputDataHolder

O subcomponente **OutputDataHolder** direciona a saída para um recipiente de dados. Ele é usado na propriedade de **saída** da ação **WriteValue**.

A tabela a seguir descreve as propriedades do subcomponente de **OutputDataHolder**:

Propriedade	Descrição
data_holder	Define o recipiente de dados de saída.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma sequência de transformadores que modificam o fluxo antes de gravar.

OutputFile

O subcomponente **OutputFile** direciona a saída para um arquivo. Ele é usado na propriedade **output** das ações **DumpValues** e **WriteValue**.

A seguinte tabela descreve as propriedades do subcomponente **OutputFile**:

Propriedade	Descrição
append	Determina se os dados são acrescentados ao conteúdo existente do arquivo. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selected. Os dados são acrescentados ao conteúdo existente.- Cleared. Os dados substituem o conteúdo existente. O padrão é Desmarcado.
file	Define uma string ou um recipiente de dados que define o caminho e o nome do arquivo. O caminho pode ser absoluto ou relativo. Se o caminho for relativo, o Script o resolverá relativo à pasta de saída da transformação.

Propriedade	Descrição
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

ResultFile

O subcomponente **ResultFile** especifica que a saída é o arquivo de saída normal de uma transformação do Processador de Dados. Ele é usado nas ações **DumpValues** e **WriteValue**.

StandardErrorLog

O subcomponente **StandardErrorLog** especifica que a saída é o log do usuário.

CAPÍTULO 19

Serializadores

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Serializadores, 333](#)
- [Âncoras de Serialização, 336](#)
- [Propriedades do Serializador Padrão, 338](#)
- [Referência de Componente Serializador, 338](#)
- [Referência de Componente de Âncora de Serialização, 340](#)

Visão Geral de Serializadores

Um Serializador converte um arquivo XML ou JSON em um documento de saída em qualquer formato. O conceito de serialização é oposto ao de análise. Por exemplo, a saída de um Serializador pode ser um documento de texto, um documento HTML ou até mesmo outro documento XML.

Você pode criar um Serializador com base nos seguintes métodos:

- Inverter a configuração de um Analisador existente
- Editar o Script e inserir um componente **Serializer**

Você pode combinar e também inverter um Analisador e editar o Script do Serializador resultante.

Normalmente, é mais fácil criar um Serializador do que um Analisador porque a entrada XML ou JSON é completamente estruturada. A estrutura facilita a identificação dos dados necessários e a sua gravação, em um procedimento sequencial, na saída. Por outro lado, um Analisador talvez precise processar uma entrada não estruturada ou semiestruturada, uma tarefa que geralmente é complexa.

Os principais componentes aninhados em um Serializador são as âncoras de serialização. A função das âncoras de serialização é identificar os dados XML ou JSON e gravá-los na saída. As âncoras de Serialização são análogas às âncoras em um Analisador, com a diferença de que elas funcionam na direção oposta.

Controlando como o Comando Criar Serializador Funciona

Quando você executa o comando **Criar Serializador**, a ferramenta Developer converte as âncoras de **Conteúdo** do Analisador em âncoras de serialização **ContentSerializer**.

Por padrão, o comando converte todos os outros textos do exemplo de origem em âncoras de serialização **StringSerializer**. Se o outro texto tiver um conteúdo de texto clichê, a saída do Serializador conterá todos os textos clichê que estavam no exemplo de origem original.

Por exemplo, suponha que o Analisador seja executado em documentos de origem delimitados por tabulação com a seguinte estrutura:

```
Name (first and last):<tab>Ron Lehrer
```

Suponha que as âncoras sejam definidas da seguinte maneira:

Texto de origem	Âncora
Nome	Marcador
(primeiro e último):<tab>	Não está marcado como uma âncora
Ron Lehrer	Conteúdo

A saída XML do Analisador é:

```
<FullName>Ron Lehrer<FullName>
```

Agora, gere um Serializador desse Analisador e execute o Serializador nas seguintes entradas:

```
<FullName>Larissa Chan<FullName>
```

A saída do Serializador é:

```
Name (first and last):<tab>Larissa Chan
```

Modo de Serialização

A origem de exemplo pode conter o texto que você não deseja na saída do Serializador. Nesse caso, você pode modificar o comportamento do comando **Criar Serializador** em uma maneira que não gera as âncoras de serialização **StringSerializer**.

Para fazer isso, defina a propriedade **serialization_mode** do componente **Analisador**. Os valores possíveis da **serialization_mode** são explicados na seguinte tabela:

Valor	Descrição
Completo	O comando Criar Serializador copia o texto não XML para a configuração do Serializador. Esse é o comportamento padrão.
Estrutura de Tópicos	O comando Criar Serializador copia somente os delimitadores do texto não XML para a configuração do Serializador. Na opção Estrutura de Tópicos , você pode selecionar a opção use_markers . Isso faz com que o comando Criar Serializador copie o conteúdo das âncoras Marcador , mas somente os delimitadores de outros textos não XML.

A seguinte tabela ilustra os resultados das configurações de **serialization_mode**:

serialization_mode	Comportamento	Saída de Serializador de Amostra
estrutura de tópicos Com a opção use_markers desmarcada	O comando Criar Serializador converte: <ul style="list-style-type: none">- As âncoras Conteúdo em âncoras de serialização ContentSerializer- Os delimitadores de outros textos no exemplo de origem em âncoras de serialização StringSerializer	<tab>Larissa Chan
estrutura de tópicos Com a opção use_markers marcada	O comando Criar Serializador converte: <ul style="list-style-type: none">- As âncoras Conteúdo em âncoras de serialização ContentSerializer- O texto completo das âncoras Marcador em âncoras de serialização StringSerializer- Os delimitadores de outros textos no exemplo de origem em âncoras de serialização StringSerializer	Nome<tab>Larissa Chan
completo	O comando Criar Serializador converte: <ul style="list-style-type: none">- As âncoras Conteúdo em âncoras de serialização ContentSerializer- Todos os outros textos no exemplo de origem em âncoras de serialização StringSerializer	Nome (primeiro e sobrenome):<tab>Larissa Chan

Solução de Problemas do Serializador Gerado Automaticamente

Geralmente, você pode usar diretamente um Serializador gerado automaticamente. Se necessário, você pode editar o Serializador gerado automaticamente para corrigir as limitações ou os problemas que você localizar nele.

Os parágrafos a seguir listam algumas situações típicas nas quais você precisa editar o Serializador, além das etapas de edição sugeridas.

Marca Raiz

Na guia Geração de XML das configurações da transformação de Processador de Dados, é possível configurar o Script de forma a delimitar um elemento raiz no elemento raiz definido no esquema de saída.

Se você usar o XML de saída como a entrada de um Serializador gerado automaticamente, deverá definir a propriedade **root_tag** do Serializador como o nome do elemento raiz definido nas configurações de Geração de XML.

Variáveis

Se o Analisador usar uma variável para armazenar os resultados intermediários, um Serializador gerado automaticamente poderá falhar. Para resolver o problema, examine a lógica do Serializador e remova a variável, se necessário.

Componentes Adicionais

O comando Criar Serializador inverte as âncoras de um Analisador. Ele não inverte componentes, como processadores de documentos, transformadores ou ações.

Por exemplo, suponha que um Analisador use um processador de documentos PdfToTxt_4 para converter documentos de origem em PDF para texto. O Analisador contém âncoras que transformam o texto em XML.

O Serializador gerado automaticamente transforma o XML de volta para texto. Ele não converte o texto em PDF. Para obter a saída em PDF, edite o Serializador e insira um processador XmlToDocument.

Em outro exemplo, suponha que um Analisador use um transformador AddString para adicionar um prefixo à saída de uma âncora de Conteúdo. O Serializador gerado automaticamente não remove o prefixo. Se você precisar removê-lo, poderá inserir um componente, como um transformador Substituir.

Criando um Serializador por meio da Edição do Script

1. No nível global do Script, clique duas vezes no sinal de reticências (. . .), digite um nome para o Serializador e pressione **ENTER**.
2. À direita do sinal de igual, clique duas vezes no sinal de reticências, selecione um **Serializador** e pressione **ENTER**.
3. Expanda a árvore sob o componente **Serializador**. Atribua suas propriedades conforme necessário.
4. Adicione um esquema que defina a sintaxe XML da entrada de Serializador.
5. Na linha **contains**, adicione uma sequência de ações e âncoras de serialização aninhadas.
6. Execute e teste o Serializador, e modifique o Script conforme necessário.

Amostra Online

Para ver um exemplo de um Serializador que criamos editando o Script, abra o projeto `samples\Projects\ManualSerializer\ManualSerializer.cmw`. Você pode executar o Serializador no arquivo de entrada `Example XML of Person.xml`.

Criando um Serializador dentro de uma Ação RunSerializer

Além de definir um Serializador no nível global, você pode definir um Serializador dentro de uma ação `RunSerializer`.

Âncoras de Serialização

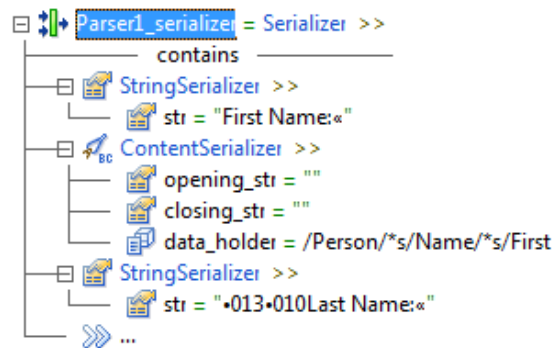
As âncoras de Serialização são os componentes principais usados em um Serializador. Elas são análogas às âncoras que são usadas em um Analisador, exceto pelo fato de trabalharem na direção oposta. As âncoras leem dados de localizações no documento de origem e gravam os dados no XML. As âncoras de serialização `anchors` leem os dados XML e gravam os dados nas localizações do documento de saída.

As âncoras de serialização mais importantes são **ContentSerializer** e **StringSerializer**.

- Um **ContentSerializer** grava o conteúdo de um determinado retentor de dados no documento de saída. Ele é o inverso de uma âncora de **Conteúdo**, que lê o conteúdo de um documento de origem.
- Um **StringSerializer** grava uma string predefinida na saída. Ele é o inverso de uma âncora de **Marcador**, que localiza uma string predefinida em um documento de origem.

Exemplo de Âncoras de Serialização

O seguinte exemplo ilustra três âncoras de serialização:



O primeiro `StringSerializer` instrui o Serializador a gravar o seguinte texto no documento de saída:

```
First Name:<tab>
```

O `ContentSerializer` grava o valor do elemento `Person/Name/First` na saída.

O segundo `StringSerializer` grava a string:

```
<newline>Last Name:<tab>
```

Nota: O editor do IntelliScript representa a nova linha e guia com os códigos ASCII e « respectivamente.

Agora suponha que você execute o Serializador no seguinte XML:

```
<Person gender="M">
  <Name>
    <First>Ron</First>
    <Last>Lehrer</Last>
  </Name>
  <Id>547329876</Id>
  <Age>27</Age>
</Person>
```

A partir das âncoras de serialização ilustradas, a saída é:

```
First Name<tab>Ron<newline>Last Name<tab>
```

A exibição desse texto é:

```
First Name:      Ron
Last Name:
```

O Serializador contém âncoras de serialização adicionais que não são mostradas na ilustração acima. A saída completa do Serializador é:

```
First Name:      Ron
Last Name:      Lehrer
Id:             547329876
Age:            27
Gender:         M
```

Sequência de Âncoras de Serialização

Um Serializador executa as âncoras de serialização na sequência de suas definições.

As âncoras de serialização gravam os dados em sequência, sempre os anexando ao final do documento de saída. Você pode mudar a ordem alterando a sequência na configuração do Serializador.

Você pode intercalar ações com as âncoras de serialização. As ações são executadas como parte da sequência.

Propriedades do Serializador Padrão

A seguinte tabela descreve as propriedades padrão no componente **Serializador** e em muitas âncoras de serialização:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componente Serializador

Um **Serializador** converte documentos XML em documentos de saída de qualquer formato. Ele usa âncoras de serialização para identificar e manipular os dados na origem.

Serializador

Um **Serializador** converte documentos XML em documentos de saída.

A seguinte tabela descreve as propriedades do componente **Serializador**:

Propriedade	Descrição
default_transformers	Define uma lista de transformadores que o Serializador aplica a todos os dados serializados.
example_source	Define um documento de origem XML de amostra. Quando você executa o Serializador na ferramenta Developer, ele opera no documento de exemplo. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Vazio. Será solicitado que você especifique um documento de origem quando o Serializador for executado. Padrão. - InputPort. Define uma porta de entrada. - LocalFile. Define um arquivo no sistema de arquivos local. - Text. Define uma string. - URL. Define uma URL.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
output_file_extension	Define a extensão do arquivo de saída gerado. O padrão é ".txt".
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
root_tag	Define o nome de um elemento XML raiz que não está no esquema de entrada do Serializador. Nota: Se a entrada do Serializador em XML de outro componente no Script e as configurações de transformação do Processador de Dados adicionarem um elemento raiz de wrapper ao redor da saída, você deverá definir essa propriedade como o nome do elemento raiz de wrapper.
origem	Define um recipiente de dados que contém a origem para a serialização. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .
destino	Define um recipiente de dados que contém o resultado da serialização. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .
validate_source_document	Determina o nível de validação do XML de origem que o Serializador realiza. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Parcial. Permite alguns desvios do esquema. Padrão. - Estrito. Força o esquema estritamente.

Referência de Componente de Âncora de Serialização

As âncoras de serialização em um **Serializador** identificam e manipulam os dados no documento de origem.

AlternativeSerializers

A âncora de serialização **AlternativeSerializers** define um conjunto de âncoras de serialização alternativas que são aninhadas abaixo do Serializador pai. Defina um critério para a alternativa que o Serializador deve aceitar. Somente a alternativa aceita afeta a saída do Serializador. As outras âncoras de serialização não têm efeito sobre a saída.

A seguinte tabela descreve as propriedades da âncora de serialização **AlternativeSerializers**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
selector	Define o critério para selecionar uma das âncoras de serialização alternativas. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- ScriptOrder. O Serializador testa as âncoras de serialização aninhadas na ordem em que estão definidas no Script. Ele aceita a primeira âncora de serialização bem-sucedida. Se todas as âncoras de serialização aninhadas falharem, o componente AlternativeSerializers falhará.- NameSwitch. O Serializador procura a âncora de serialização aninhada cuja propriedade nome está especificada em um recipiente de dados. Ele ignora outras âncoras de serialização aninhadas. Se a âncora de serialização nomeada falhar, o componente AlternativeSerializers falhará. O padrão é ScriptOrder.

Exemplo

A entrada XML pode conter um elemento de `Produto` ou um elemento de `Serviço`, mas não ambos. Você deseja serializar o elemento que está na entrada.

Defina uma âncora de serialização **AlternativeSerializers** e defina a sua propriedade de **seletor** como `ScriptOrder`.

Sob o componente **AlternativeSerializers**, aninhe duas âncoras de serialização **ContentSerializer**. Configure uma delas para processar o elemento de `Produto` e outra para processar o elemento de `Serviço`.

ContentSerializer

A âncora de serialização **ContentSerializer** grava o valor dos dados serializados no documento de saída.

A tabela a seguir descreve as propriedades da âncora de serialização **ContentSerializer**:

Propriedade	Descrição
<code>allow_empty_values</code>	Determina se data_holder pode estar vazio. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. O data_holder pode estar vazio.- Desmarcado. O data_holder não pode estar vazio, caso contrário, o ContentSerializer falhará. O padrão é desmarcado.
<code>closing_str</code>	Define a string que a âncora grava após o data_holder .
<code>data_holder</code>	Define o recipiente de dados que contém os dados serializados.
<code>desativado</code>	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
<code>ignore_default_transformers</code>	Determina se os transformadores padrão do Serializador são aplicados aos dados serializados. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Os transformadores padrão do Serializador não são aplicados.- Desmarcado. Os transformadores padrão do Serializador são aplicados. O padrão é desmarcado.
<code>nome</code>	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
<code>on_fail</code>	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
<code>opening_str</code>	Define a string que a âncora grava antes do conteúdo do data_holder .

Propriedade	Descrição
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que são aplicados aos dados serializados.

DelimitedSectionsSerializer

A âncora de serialização **DelimitedSectionsSerializer** processa seções de dados. As seções da saída são separadas por uma determinada string do separador.

Sob o **DelimitedSectionsSerializer**, aninhe outras âncoras de serialização. Cada âncora de serialização aninhada gera uma seção única.

A tabela a seguir descreve as propriedades da âncora de serialização **DelimitedSectionsSerializer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
separador	Define um Serializador que define a string do separador. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - AlternativeSerializers - ContentSerializer - EmbeddedSerializer - GroupSerializer - RepeatingGroupSerializer - StringSerializer - Serializador Definido pelo Usuário O padrão é StringSerializer.
separator_position	Define a posição do separador relativo às seções. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - after. Grava um separador depois de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - around. Grava separadores antes e depois de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - before. Grava um separador antes de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - between. Grava um separador entre as seções sucessivas, mas não antes da primeira, nem depois da última. Por exemplo, 1 2 3 4 O padrão é before.
using_placeholders	Determina se o DelimitedSectionsSerializer grava o separador de uma seção opcional que está ausente na entrada do XML. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - sempre. Sempre grava o separador de uma seção ausente. Por exemplo, 1 3 - nunca. Nunca grava o separador de uma seção ausente. Por exemplo, 1 3 - quando necessário. Sempre grava o separador de uma seção interna ausente. Nunca grava o separador de uma seção terminal ausente. Por exemplo, 1 3 O padrão é quando necessário.

Exemplo

A entrada XML contém um resumo do funcionário. Você deseja gravar os dados em um documento de texto de saída no seguinte formato:

```

-----
Jane Palmer
Employee ID 123456
-----
Professional Experience
...
-----
Education
...

```

Defina um **DelimitedSectionsSerializer** com a linha de hífen como `separador`. Como você deseja uma linha de hífen antes de cada seção, defina `separator_position` como `antes`.

Dentro do **DelimitedSectionsSerializer**, aninhe três componentes de **GroupSerializer**. O primeiro **GroupSerializer** grava a seção `Jane Palmer`, o segundo grava a seção `Professional Experience` e assim por diante.

Seções Opcionais

Neste exemplo, suponha que a segunda seção, `Professional Experience`, está faltando em alguns documentos XML de entrada, mas você deseja gravar o seu separador na saída da seguinte forma:

```
-----  
Jane Palmer  
Employee ID 123456  
-----  
-----  
Education  
...  
-----
```

Para dar suporte a essa situação, configure o **DelimitedSectionsSerializer** da seguinte maneira:

- No segundo **GroupSerializer**, selecione a propriedade **opcional**. Isso significa que se o **GroupSerializer** falhar, ele não deve fazer com que o **DelimitedSectionsSerializer** falhe.
- Em **DelimitedSectionsSerializer**, defina **using_placeholders** como `sempre`. Isso significa gravar o separador de uma seção opcional, mesmo se a própria seção estiver ausente.

Como alternativa, suponha que se a seção `Professional Experience` estiver ausente, você não vai querer gravar o separador:

```
-----  
Jane Palmer  
Employee ID 123456  
-----  
Education  
...  
-----
```

Nesse caso, configure o **DelimitedSectionsSerializer** da seguinte forma:

- No segundo **GroupSerializer**, selecione a propriedade **opcional**.
- Em **DelimitedSectionsSerializer**, defina **using_placeholders** como `nunca`. Isso significa não gravar o separador de uma seção ausente.

EmbeddedSerializer

A âncora de serialização **EmbeddedSerializer** ativa um **Serializador** secundário, que grava sua saída no mesmo documento de saída.

A seguinte tabela descreve as propriedades da âncora de serialização **EmbeddedSerializer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
schema_connections	<p>Conecta os recipientes de dados que são mencionados no Serializador secundário aos recipientes de dados que são mencionados no Serializador pai. A propriedade contém uma lista de subcomponentes de Conexão que definem as correspondências. Para obter mais informações, consulte "Conectar" na página 252.</p> <p>Se todos os recipientes de dados nos Serializadores principal e secundário forem idênticos, você poderá omitir esta propriedade. Se houver diferenças entre os recipientes de dados, será necessário conectá-los explicitamente, mesmo aqueles que são idênticos.</p>
Serializador	Define o nome de um Serializador secundário que está definido no nível global do Script.

Exemplo

A entrada XML é uma árvore de família. A entrada contém elementos `Person` (Pessoa), que são recursivamente aninhados, conforme mostrado:

```
<Person>      <!-- Parent -->
...
  <Person>    <!-- Child -->
    ...
      <Person> <!-- Grandchild -->
        ...
      </Person>
    </Person>
  </Person>
```

Em um **Serializador**, um componente **EmbeddedSerializer** pode chamar a si próprio recursivamente até que todos os níveis de aninhamento estejam esgotados.

Neste exemplo, a propriedade **schema_connections** conecta `Person` a `Person/Person`. Isso instrui a instância secundária do serializador a processar um nível aninhado da entrada. Quando os dois elementos `Pessoa` tiverem o mesmo tipo de dados, bastará conectar apenas o elemento pai (`Pessoa`) e não os elementos aninhados (`Person/*s/Name`, `Person/*s/BirthDate`, etc.)

GroupSerializer

A âncora de serialização **GroupSerializer** une suas âncoras de serialização aninhadas. Você pode definir propriedades de **GroupSerializer** que afetam os membros do grupo.

A seguinte tabela descreve as propriedades da âncora de serialização **GroupSerializer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
origem	Define um recipiente de dados que contém a origem para a serialização. Para obter mais informações, consulte "Visão Geral de Validadores, Notificações e Tratamento de Falhas" na página 398 .
destino	Define um recipiente de dados que contém o resultado da serialização. Para obter mais informações, consulte "Visão Geral de Validadores, Notificações e Tratamento de Falhas" na página 398 .

RepeatingGroupSerializer

A âncora de serialização **RepeatingGroupSerializer** grava uma estrutura repetitiva no documento de saída.

Use um **RepeatingGroupSerializer** quando os dados XML contiverem um recipiente de dados de ocorrência múltipla. Ele efetua iterações sobre as ocorrências do recipiente de dados e processa a saída dos dados. Para obter mais informações, consulte ["Recipientes de Dados de Ocorrência Múltipla" na página 201](#).

Em **RepeatingGroupSerializer**, aninhe âncoras de serialização que processam e geram a saída de cada ocorrência do recipiente de dados. É possível definir um separador que o **RepeatingGroupSerializer** grava na saída entre as iterações.

A seguinte tabela descreve as propriedades da âncora de serialização **RepeatingGroupSerializer**:

Propriedade	Descrição
count	Define o número de iterações a serem executadas. Se essa propriedade ficar em branco, as iterações continuarão até que a entrada seja esgotada.
current_iteration	Define um data retentor onde RepeatingGroupSerializer gera o número de iteração atual. Você pode usar um ContentSerializer para gravar o número na saída.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
on_iteration_fail	Determina a ação quando uma iteração falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Limpo. Nenhuma ação. - CustomLog. Grava no log do usuário. - LogError. Grava uma mensagem de erro no log do Mecanismo. - LogInfo. Grava uma mensagem de informações no log do Mecanismo. - LogWarning. Grava uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Dispara uma notificação. Use on_iteration_fail para gravar uma entrada quando uma única iteração falhar. Use a propriedade on_fail para gravar uma entrada quando o RepeatingGroupSerializer inteiro falhar. Para obter mais informações, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Propriedade	Descrição
separador	Define uma âncora de serialização que determina string de separador. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - AlternativeSerializers - ContentSerializer - EmbeddedSerializer - GroupSerializer - RepeatingGroupSerializer - StringSerializer - Serializador Definido pelo Usuário O padrão é em branco.
separator_position	Define a posição do separador relativo às seções. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - after. Grava um separador depois de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - around. Grava separadores antes e depois de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - before. Grava um separador antes de cada seção, incluindo as primeiras seções. Por exemplo, 1 2 3 4 - between. Grava um separador entre as seções sucessivas, mas não antes da primeira, nem depois da última. Por exemplo, 1 2 3 4 O padrão é before.
skip_failed_iterations	Determina se iterações com falha são ignoradas. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. RepeatingGroup ignora uma iteração com falha e continua com a próxima iteração. Se uma iteração for bem-sucedida, RepeatingGroup terá êxito. - Limpo. RepeatingGroup falhará se qualquer iteração falhar. A propriedade skip_failed_iterations somente terá efeito se a propriedade separador for definida. O padrão é selecionado.
origem	Define um recipiente de dados que contém a origem para a serialização. Para obter mais informações, consulte "Visão Geral de Validadores, Notificações e Tratamento de Falhas" na página 398 .
destino	Define um recipiente de dados que contém o resultado da serialização. Para obter mais informações, consulte "Visão Geral de Validadores, Notificações e Tratamento de Falhas" na página 398 .

Exemplo

A entrada XML contém a seguinte estrutura:

```
<Persons>
  <Person>
    <Name>John</Name>
    <Age>35</Age>
  </Person>
  <Person>
    <Name>Larissa</Name>
    <Age>42</Age>
  </Person>
  ...
</Persons>
```

Um **RepeatingGroupSerializer**, usando um caractere de nova linha como um separador, pode gerar esses dados para:

```
John      35
Larissa   42
```

Você pode iterar por vários recipientes de dados de ocorrência múltipla em paralelo. Por exemplo, você pode iterar entre uma lista de homens e uma lista de mulheres e gerar uma lista de casais casados. Para fazer isso, insira um **ContentSerializer** dentro do grupo de repetição para cada recipiente de dados.

StringSerializer

A âncora de serialização **StringSerializer** grava uma string predefinida no documento de saída.

A seguinte tabela descreve as propriedades da âncora de serialização **StringSerializer**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
str	Define a string que o Serializador grava para a saída.

CAPÍTULO 20

Mapeadores

Este capítulo inclui os seguintes tópicos:

- [Criando um Mapeador, 350](#)
- [Componentes Aninhados Dentro de um Mapeador, 350](#)
- [Exemplo de Mapeador, 351](#)
- [Propriedades do Mapeador Padrão, 352](#)
- [Referência de Componente de Mapeador, 353](#)
- [Referência do Componente de Âncora Mapeadora, 355](#)

Criando um Mapeador

1. Adicione esquemas de entrada e saída ao objeto de esquema e depois faça referência a esses esquemas na transformação de Processador de Dados.
2. No nível global do Script, adicione um componente **Mapeador**.
3. Atribua as propriedades **source** e **target** do **Mapeador** aos elementos de entrada e saída do **Mapeador**, respectivamente.
4. Atribua a propriedade **example_source** a um documento de entrada XML de amostra.
À medida que você adiciona componentes ao Mapeador, a ferramenta Developer codifica com cores as localizações correspondentes no exemplo de origem. As cores podem ajudar você a confirmar se os componentes estão definidos corretamente.
5. Edite as outras propriedades do **Mapeador** conforme necessário.
6. Dentro do **Mapeador**, aninhe uma sequência de ações **Mapa**, âncoras Mapeadoras e qualquer outro componente necessário.
7. Teste o Mapeador e modifique o Script.

Componentes Aninhados Dentro de um Mapeador

Dentro de um **Mapeador**, é possível aninhar os seguintes componentes:

- Qualquer número de ações de **Mapa**. As ações recuperam um recipiente de dados da saída e gravam o conteúdo na saída.

- Opcionalmente, qualquer número de âncoras Mapeadoras. Para obter mais informações, consulte [“Referência do Componente de Âncora Mapeadora” na página 355](#).
- Opcionalmente, qualquer número de ações adicionais.

As ações de `Mapa` e as âncoras Mapeadoras podem estar em qualquer sequência. Você também pode inserir outras ações na sequência.

Observe que um Mapeador usa ações de `Mapa` em vez de âncoras Mapeadoras para gravar a saída XML. Isso pode parecer um pouco diferente de Analisadores e Serializadores, em que a saída é criada por âncoras e âncoras de serialização, respectivamente. Na realidade, esse é apenas um problema de terminologia. A ação de `Mapa` pode ter sido definida como uma âncora Mapeadora. Ela é definida como uma ação porque é útil em outras circunstâncias não relacionadas aos Mapeadores.

Exemplo de Mapeador

Para ilustrar a configuração do Mapeador, apresentamos um exemplo simples.

XML de Origem

A entrada do Mapeador é um documento XML que contém uma lista de nomes pessoais e seus números de ID associados.

```
<Persons>
  <Person ID="10">Bob</Person>
  <Person ID="17">Larissa</Person>
  <Person ID="13">Marie</Person>
</Persons>
```

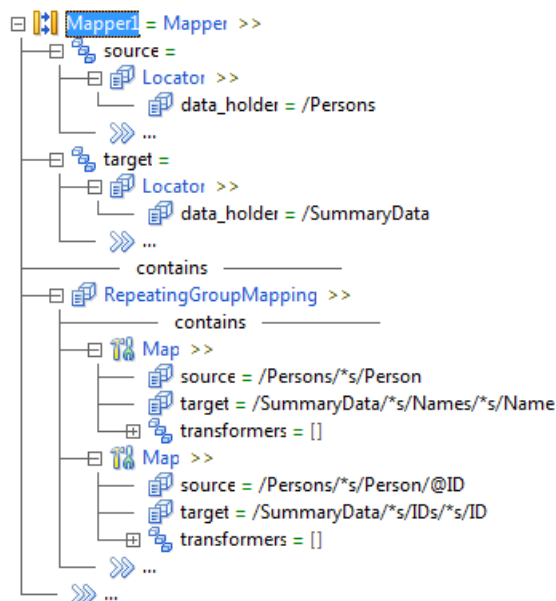
XML de Saída

A saída desejada do Mapeador é uma lista XML de nomes e números de ID sem associação entre eles.

```
<SummaryData>
  <Names>
    <Name>Bob</Name>
    <Name>Larissa</Name>
    <Name>Marie</Name>
  </Names>
  <IDs>
    <ID>10</ID>
    <ID>17</ID>
    <ID>13</ID>
  </IDs>
</SummaryData>
```

Configuração do Mapeador

A seguinte configuração do Mapeador executa a transformação desejada:



O `RepeatingGroupMapping` itera sobre os elementos `Pessoa` da entrada. Ele usa as ações de `Mapa` para gravar os dados nos elementos `Nome` e `ID` da saída.

Propriedades do Mapeador Padrão

A seguinte tabela descreve as propriedades padrão no componente de **Mapeador** e em muitas âncoras Mapeadoras:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
opcional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componente de Mapeador

Um Mapeador lê um documento de origem XML e o converte em outro documento XML.

Mapeador

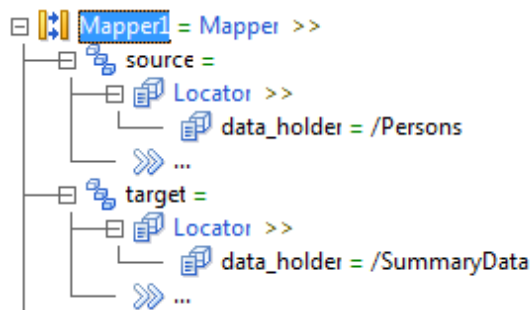
Um componente **Mapeador** executa transformações de XML para XML. Ele converte um documento XML de origem em um documento de saída que possui uma estrutura XML diferente.

A seguinte tabela descreve as propriedades do componente **Mapeador**:

Propriedade	Descrição
example_source	<p>Define um documento de origem XML de amostra. Quando você executa o Mapeador na ferramenta Developer, ele opera no documento de exemplo. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Vazio. É necessário especificar um documento de origem quando o Script é executado. - InputPort. Define uma porta de entrada. - LocalFile. Define um arquivo no sistema de arquivos local. - Text. Define uma string. - URL. Define uma URL. <p>O padrão é vazio.</p>
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte “Notificações” na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
root_tag	Define o nome de um elemento XML raiz que não está definido no esquema. Por exemplo, se o elemento de nível superior do esquema for <code>Person</code> , mas a entrada XML aninhar <code>Person</code> em um elemento denominado <code>InputWrapper</code> , defina root_tag como <code>InputWrapper</code> .
origem	Define um componente de localizador que define um recipiente de dados XML. O recipiente de dados contém a raiz da origem XML para o mapeamento. Para obter mais informações, consulte “Visão Geral de Localizadores, Chaves e Indexação” na página 360 .
destino	Define um componente de localizador que define um recipiente de dados XML. O recipiente de dados contém a raiz do XML de saída para o mapeamento. Para obter mais informações, consulte “Visão Geral de Localizadores, Chaves e Indexação” na página 360 .
validate_source_document	Determina o nível de validação do XML de origem que o serializador realiza. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Parcial. Permite alguns desvios do esquema. - Estrito. Força o esquema estritamente. O padrão é Parcial.

Você deve usar as propriedades **source** e **target** para identificar os elementos raiz dos documentos XML. Por exemplo, se o elemento de documento da origem for `Persons` e o elemento de documento da saída for `SummaryData`, defina a **origem** e o **destino** da seguinte maneira:



Referência do Componente de Âncora Mapeadora

As âncoras Mapeadoras em um Mapeador identificam e manipulam dados em um documento XML.

AlternativeMappings

A âncora Mapeadora **AlternativeMappings** define um conjunto de âncoras Mapeadoras alternativas. Defina um critério para a seleção de uma alternativa. Somente a alternativa aceita afeta a saída.

A seguinte tabela descreve as propriedades da âncora Mapeadora **AlternativeMappings**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
selector	Determina o critério para a seleção de um Mapeador alternativo. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- ScriptOrder. O Script testa as âncoras Mapeadoras aninhadas na sequência em que elas são definidas no Script. Ele aceita a primeira âncora de serialização bem-sucedida. Se todas as âncoras Mapeadoras aninhadas falharem, o componente AlternativeMappings também falhará.- NameSwitch. O Script procura a âncora Mapeadora aninhada cuja propriedade nome esteja especificada em um recipiente de dados. Ele ignora as outras âncoras Mapeadoras aninhadas. Se a âncora Mapeadora nomeada falhar, o componente AlternativeMappings também falhará. O padrão é ScriptOrder.

Exemplo

A entrada XML pode conter um elemento de `Produto` ou um elemento de `Serviço`, mas não ambos. Você deseja processar o elemento que está na entrada.

Defina uma âncora Mapeadora **AlternativeMappings** e defina a sua propriedade **seletor** como `ScriptOrder`.

Dentro de **AlternativeMappings**, aninhe duas ações de Mapa. Configure uma delas para processar o elemento de Produto e outra para processar o elemento de Serviço.

EmbeddedMapper

A âncora Mapeadora **EmbeddedMapper** ativa um **Mapeador** secundário que armazena sua saída no mesmo documento de saída.

A seguinte tabela descreve as propriedades da âncora Mapeadora **EmbeddedMapper**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
Mapeador	Define o nome do Mapeador secundário.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
schema_connections	Conecta os recipientes de dados que são mencionados no Mapeador secundário aos recipientes de dados que são mencionados no Mapeador pai. A propriedade contém uma lista de subcomponentes de Conexão que definem as correspondências. Para obter mais informações, consulte "Conectar" na página 252 . Se todos os recipientes de dados nos Mapeadores principal e secundário forem idênticos, você poderá omitir esta propriedade. Se houver diferenças entre os recipientes de dados, você deverá conectá-los explicitamente.

Exemplo

A entrada XML é uma árvore de família. A entrada contém elementos `Person` (`Pessoa`), que são recursivamente aninhados, conforme mostrado:

```
<Person>          <!-- Parent -->
...
  <Person>        <!-- Child -->
...
    <Person>      <!-- Grandchild -->
...
  </Person>
</Person>
</Person>
```

Um **Mapeador**, pode usar um componente **EmbeddedMapper** para chamar a si próprio recursivamente até que todos os níveis de aninhamento estejam esgotados.

Neste exemplo, `Pessoa` está conectada a `Pessoa/Pessoa`. Isso instrui a instância secundária do Mapeador a processar um nível aninhado da entrada. Quando os dois elementos `Pessoa` tiverem o mesmo tipo de dados, bastará conectar apenas o elemento pai (`Pessoa`) e não os elementos aninhados (`Person/*s/Name`, `Person/*s/BirthDate`, etc.)

GroupMapping

A âncora Mapeadora **GroupMapping** vincula suas ações e âncoras Mapeadoras aninhadas. Você pode definir propriedades de **GroupMapping** que afetam os membros do grupo.

A seguinte tabela descreve as propriedades da âncora Mapeadora **GroupMapping**:

Propriedade	Descrição
absent	Determina o comportamento de GroupMapping quando uma de suas ações ou âncoras Mapeadoras aninhadas e obrigatórias falha. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. GroupMapping é bem-sucedido.- Limpo. GroupMapping falha. Use este recurso para testar a ausência de âncoras Mapeadoras aninhadas. O padrão é limpo.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .
on_fail	A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Cleared. Não executar nenhuma ação.- CustomLog. Gravar no log do usuário.- LogError. Gravar uma mensagem de erro no log do mecanismo.- LogInfo. Gravar uma mensagem de informações no log do mecanismo.- LogWarning. Gravar uma mensagem de aviso no log do Mecanismo.- NotifyFailure. Enviar uma notificação. O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399 .

Propriedade	Descrição
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
origem	Define um componente de localizador que define um recipiente de dados XML. O recipiente de dados contém a raiz da origem XML para o mapeamento. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .
destino	Define um componente de localizador que define um recipiente de dados XML. O recipiente de dados contém a raiz do XML de saída para o mapeamento. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .

RepeatingGroupMapping

A âncora Mapeadora **RepeatingGroupMapping** processa uma estrutura repetitiva na entrada ou na saída.

Use um **RepeatingGroupMapping** quando a entrada ou a saída XML contiver um recipiente de dados de ocorrência múltipla. Ele efetua iterações sobre ocorrências dos recipientes de dados. Para obter mais informações, consulte ["Recipientes de Dados de Ocorrência Múltipla" na página 201](#).

Em **RepeatingGroupMapping**, aninhe âncoras Mapeadoras e ações que processam cada ocorrência do recipiente de dados.

A seguinte tabela descreve as propriedades da âncora Mapeadora **RepeatingGroupMapping**:

Propriedade	Descrição
count	Define o número de iterações a serem executadas. Se essa propriedade ficar em branco, as iterações continuarão até que a entrada seja esgotada.
current_iteration	Define um recipiente de dados no qual o RepeatingGroupMapping gera o número da iteração atual.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificações	Uma lista de componentes NotificationHandler que lidam com notificações a partir de componentes aninhados. Para obter mais informações, consulte "Notificações" na página 418 .

Propriedade	Descrição
on_fail	<p>A ação a ser executada se o componente falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Cleared. Não executar nenhuma ação. - CustomLog. Gravar no log do usuário. - LogError. Gravar uma mensagem de erro no log do mecanismo. - LogInfo. Gravar uma mensagem de informações no log do mecanismo. - LogWarning. Gravar uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Enviar uma notificação. <p>O padrão é Desmarcado. Para obter mais informações sobre como lidar com falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
on_iteration_fail	<p>Determina a ação quando uma iteração falhar. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Limpo. Nenhuma ação. - CustomLog. Grava no log do usuário. - LogError. Grava uma mensagem de erro no log do Mecanismo. - LogInfo. Grava uma mensagem de informações no log do Mecanismo. - LogWarning. Grava uma mensagem de aviso no log do Mecanismo. - NotifyFailure. Dispara uma notificação. <p>Use on_iteration_fail para gravar uma entrada quando uma única iteração falhar. Use a propriedade on_fail para gravar uma entrada quando o RepeatingGroupMapping inteiro falhar. Para obter mais informações, consulte "Tratamento de Falhas" na página 399.</p>
optional	<p>Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. <p>O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399.</p>
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
skip_failed_iterations	<p>Determina se iterações com falha são ignoradas. É possível escolher uma das seguintes opções:</p> <ul style="list-style-type: none"> - Selecionado. RepeatingGroup ignora uma iteração com falha e continua com a próxima iteração. Se uma iteração for bem-sucedida, RepeatingGroup terá êxito. - Limpo. RepeatingGroup falhará se qualquer iteração falhar. <p>A propriedade skip_failed_iterations somente terá efeito se a propriedade separator for definida.</p> <p>O padrão é selecionado.</p>
origem	Define um recipiente de dados que contém a origem para o mapeamento. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .
destino	Define um recipiente de dados que contém o resultado do mapeamento. Para obter mais informações, consulte "Visão Geral de Localizadores, Chaves e Indexação" na página 360 .

Exemplo

Para obter mais informações, incluindo um exemplo de um **RepeatingGroupMapping**, consulte o ["Exemplo de Mapeador" na página 351](#).

CAPÍTULO 21

Localizadores, Chaves e Indexação

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Localizadores, Chaves e Indexação, 360](#)
- [Exemplo de Localizadores, 361](#)
- [Exemplo de Indexação por Chave, 363](#)
- [Propriedades de Origem e Destino, 365](#)
- [Localizador Padrão e Propriedades de Chaves, 371](#)
- [Referência de Componentes de Localizador e Chave, 371](#)

Visão Geral de Localizadores, Chaves e Indexação

Ao criar uma transformação, um problema frequente é como localizar os recipientes de dados que você deseja processar. Se os mesmos recipientes de dados puderem ocorrer várias vezes em uma estrutura XML, poderá haver ambiguidades em identificar as ocorrências. Este capítulo explica como usar os componentes do `Localizador` e `Chave` para resolver as ambiguidades.

Os componentes descritos neste capítulo permitem que você identifique ocorrências múltiplas de recipientes de dados de três maneiras:

- Sequencialmente. Cada iteração de um componente processa a próxima ocorrência do recipiente de dados.
- Por número de ocorrência. Por exemplo, um componente pode selecionar a terceira ocorrência de um recipiente de dados.
- Por uma chave, como um atributo ou um elemento aninhado. A chave identifica exclusivamente a ocorrência do recipiente de dados.

A abordagem sequencial é o padrão. Ela está sujeita a algumas complexidades que você pode controlar usando o componente do `Localizador`.

As abordagens de número de ocorrência e de chave são conhecidas coletivamente como indexação. O termo é análogo ao índice de um livro, em que você usa um número de página ou uma chave de assunto para identificar a localização das informações. Você pode implementar a indexação usando componentes chamados `LocatorByOccurrence`, `LocatorByKey` e `Chave`.

Você pode usar o localizador e os componentes de chave em analisadores, serializadores ou mapeadores. Você pode usar os componentes para identificar as ocorrências de recipientes de dados na entrada, saída ou ambos.

Os componentes do localizador estão aninhados nas propriedades de `origem` e `destino` de vários componentes de transformação. O significado e o uso das propriedades de `origem` e de `destino` é explicado abaixo.

Exemplo de Localizadores

Para entender os problemas envolvidos na identificação de recipientes de dados, considere o exemplo a seguir. O exemplo ilustra o uso de:

- A propriedade de `destino`
- O componente do `Localizador`

Explicaremos a ampla estrutura de tópicos em detalhes aqui. Nas seções do capítulo a seguir, voltaremos para explicar em detalhes como o `destino` e o `Localizador` funcionam.

Entrada e Saída

Suponha que o esquema de saída de um Analisador ofereça suporte à seguinte estrutura:

```
<Report>
  <Company>
    <Employee>John</Employee>
    <Employee>Leslie</Employee>
    <Employee>Pedro</Employee>
  </Company>
  <Company>
    <Employee>Marie</Employee>
    <Employee>Larry</Employee>
    <Employee>Frances</Employee>
  </Company>
</Report>
```

O documento de origem que o Analisador processa é uma lista contendo um único funcionário por empresa:

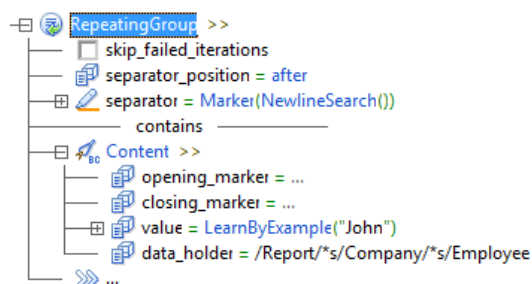
```
John
Marie
```

A saída do Analisador deve ser:

```
<Report>
  <Company>
    <Employee>John</Employee>
  </Company>
  <Company>
    <Employee>Marie</Employee>
  </Company>
</Report>
```

Solução Incorreta

Suponha que você use o seguinte **RepeatingGroup** para analisar o documento de origem:



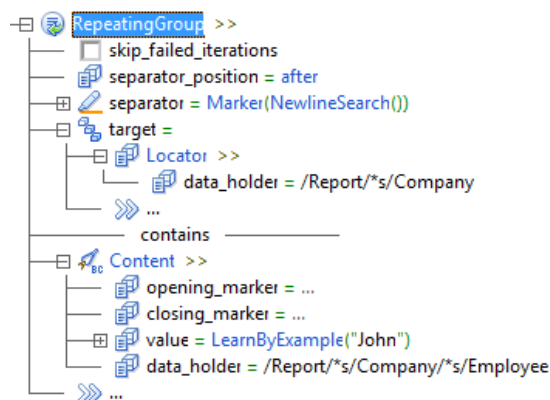
A saída é incorreta:

```
<Report>
  <Company>
    <Employee>John</Employee>
    <Employee>Marie</Employee>
  </Company>
</Report>
```

O problema é que a `Empresa` e o `Funcionário` são elementos de ocorrência múltipla. O `RepeatingGroup` cria vários elementos de `Funcionário` corretamente, mas não sabe que cada elemento `Funcionário` deve ser aninhado em um elemento `Empresa` separado.

Solução Correta

Para resolver a ambiguidade, você pode atribuir a propriedade de destino do `RepeatingGroup`.



O destino identifica o recipiente de dados que o `RepeatingGroup` deve criar. O destino contém um componente de Localizador apontando para o elemento `Empresa`. Isso significa que cada iteração do `RepeatingGroup` deve criar uma nova ocorrência do elemento `Empresa`.

Se você configurar o `RepeatingGroup` dessa maneira, a saída estará correta.

Exemplo de Indexação por Chave

Para explicar outros problemas de identificação do recipiente de dados, apresentamos um exemplo de indexação por chave.

O exemplo é um mapeador que usa a indexação para identificar as ocorrências dos recipientes de dados na entrada e na saída. No lado da entrada, a indexação corresponde os dados correspondentes de diferentes partes de uma estrutura XML. No lado da saída, a indexação encontra o local correto de um elemento em uma estrutura XML.

O exemplo ilustra o uso de:

- As propriedades de origem e destino
- Os componentes de Localizador, Chave e LocatorByKey

Nas seções do capítulo a seguir, explicaremos a operação detalhada dessas propriedades e componentes.

Entrada

A entrada XML é um relatório listando os nomes de pais e seus filhos.

- Para cada pai, o XML lista um nome, um sobrenome e uma ID.
- Para cada filho, o XML lista um nome, um passatempo e uma ID do pai.

```
<Report>
  <Parents>
    <Parent id="1" firstName="John" lastName="Smith"/>
    <Parent id="2" firstName="Jane" lastName="Doe"/>
  </Parents>
  <Children>
    <Child name="Eric" hobby="Swimming" parentID="1"/>
    <Child name="Elizabeth" hobby="Biking" parentID="2"/>
    <Child name="Mary" hobby="Painting" parentID="1"/>
    <Child name="Edward" hobby="Swimming" parentID="2"/>
  </Children>
</Report>
```

Saída

A saída desejada é uma lista de passatempos e os filhos que participam de cada hobby.

```
<Hobbies>
  <Hobby name="Swimming">
    <Person firstName="Eric" lastName="Smith"/>
    <Person firstName="Edward" lastName="Doe"/>
  </Hobby>
  <Hobby name="Biking">
    <Person firstName="Elizabeth" lastName="Doe"/>
  </Hobby>
  <Hobby name="Painting">
    <Person firstName="Mary" lastName="Smith"/>
  </Hobby>
</Hobbies>
```

Estrutura de Tópicos da Abordagem da Transformação

A transformação usa a seguinte abordagem:

1. Na entrada XML, a transformação identifica os elementos `Filho` e `Pai` correspondentes da seguinte maneira:

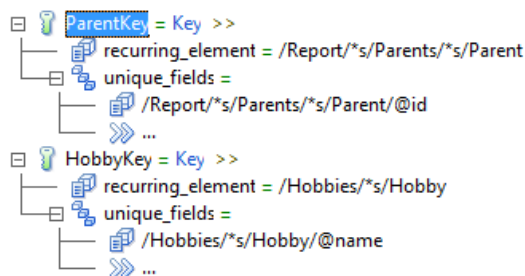
`id atributo de Pai = parentID atributo de Filho`

2. A transformação cria elementos de `Passatempo` e `Pessoa`. Ele identifica o elemento de `Passatempo` onde deve aninhar cada elemento de `Pessoa` da seguinte maneira:
`nome atributo de Passatempo = passatempo atributo de Filho`
3. A transformação grava o nome do filho no elemento `Pessoa`.
4. A transformação grava o sobrenome do pai no elemento `Pessoa`.

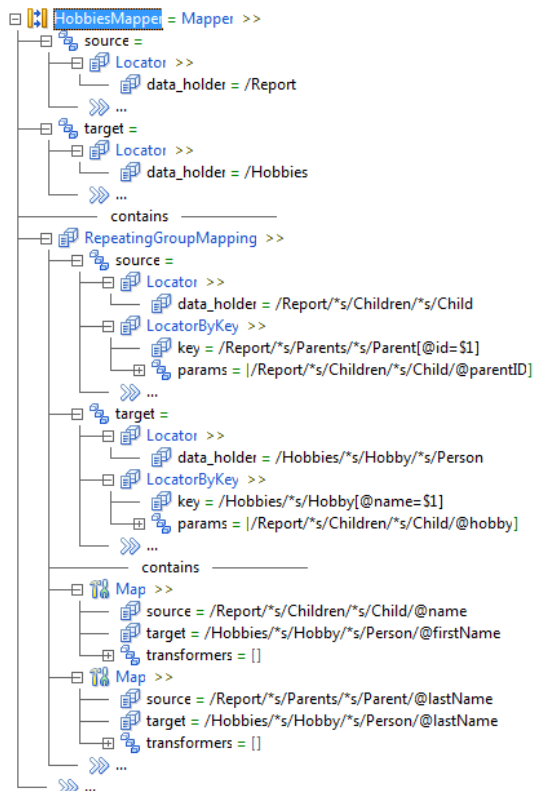
Configuração do Mapeador

Componentes `Key` definem identificadores para os recipientes de dados.

- O primeiro `Key` especifica que o atributo `id` é um identificador exclusivo de um elemento `Parent`.
- O segundo `Key` especifica que o atributo `name` é um identificador exclusivo de um elemento `Hobby`.



O Script então define um `Mapeador` com a seguinte configuração:



Os componentes do `Mapeador` realizam as seguintes funções:

- A propriedade `source` de `RepeatingGroupMapping` especifica que cada iteração obtém sua entrada dos seguintes recipientes de dados:
 - De uma ocorrência do elemento `Child`
 - Da ocorrência correspondente do elemento `Parent`
- A propriedade `target` de `RepeatingGroupMapping` especifica que cada iteração armazena sua saída nos seguintes recipientes de dados:
 - Em uma ocorrência do elemento `Person`
 - Na ocorrência correspondente do elemento `Hobby`
- A primeira ação `Map` copia o atributo `name` de `Child` para o atributo `firstName` de `Person`.
- A segunda ação `Map` copia o atributo `lastName` de `Parent` para o atributo `lastName` de `Person`.

Uso de Indexação

O exemplo usa a indexação por chave para identificar as ocorrências dos retentores de dados `Pai` e `Hobby`.

- Na propriedade `origem` do `RepeatingGroupMapping`, a indexação identifica a ocorrência do `Pai` que corresponde a um `Filho`.
- Na propriedade `destino`, a indexação identifica a ocorrência de `Hobby` onde uma `Pessoa` deve ser aninhada.

Propriedades de Origem e Destino

As propriedades `origem` e `destino` existem em componentes, como os seguintes:

- Em analisadores:
 - `Parser`
 - `Group`
 - `RepeatingGroup`
 - `EnclosedGroup`
 - `FindReplaceAnchor`
- Em serializadores:
 - `Serializer`
 - `GroupSerializer`
 - `RepeatingGroupSerializer`
- Em mapeadores:
 - `Mapper`
 - `GroupMapping`
 - `RepeatingGroupMapping`

Em todas estas categorias, o significado e o uso das propriedades é idêntico:

- A propriedade `origem` identifica os retentores de dados existentes que uma transformação deve usar.
- A propriedade `destino` identifica os retentores de dados que podem ou não já existir. Se eles existirem, a transformação os usará. Se não existirem, a transformação os criará.

Depois de definir a `origem` e/ou o `destino`, os componentes subsequentes usarão os retentores de dados identificados. Por exemplo, se você definir o `destino` de um `Grupo`, as âncoras aninhadas no `Grupo` usarão os retentores de dados que o `destino` identifica.

Nota: Não há propriedades denominadas `origem` e `destino` em alguns outros componentes, como `Mapear`. Essas propriedades têm um significado e um uso diferentes do acima. Para obter uma explicação, consulte os componentes nos quais as propriedades são usadas.

Propriedade de Origem

A propriedade **origem** identifica as ocorrências existentes de retentores de dados. O valor da propriedade **origem** é uma lista que contém um ou mais dos seguintes componentes:

Origem	Descrição
Localizador	Identifica um retentor de dados de ocorrência única ou múltiplas ocorrências. No último caso, cada iteração acessa a próxima ocorrência, em sequência.
LocatorByKey	Identifica uma ocorrência de um retentor de dados de múltiplas ocorrências usando uma chave.
LocatorByOccurrence	Identifica uma ocorrência de um retentor de dados de múltiplas ocorrências por número.

Comportamento Padrão

Se você não atribuir a propriedade de `origem` de um componente, ele identificará os recipientes de dados da seguinte maneira:

- Se houver apenas uma ocorrência de recipiente de dados, o componente usará a ocorrência existente.
- Se houver várias ocorrências do recipiente de dados, o comportamento será da seguinte maneira:
 - Em um contexto iterativo, como dentro de um `RepeatingGroupSerializer`, cada iteração acessará a próxima ocorrência do recipiente de dados na sequência.
 - Em um contexto não iterativo, como um `GroupSerializer` que não está aninhado dentro de um componente iterativo, o componente acessará a primeira ocorrência do recipiente de dados.

Ambiguidades no Comportamento Padrão

O comportamento padrão poderá apresentar ambiguidades. As ambiguidades poderão surgir nas circunstâncias a seguir.

- Em casos onde um elemento de várias ocorrências está aninhado com outro elemento de várias ocorrências. Para obter mais informações, consulte [“Exemplo 1: Recipientes de Dados com Várias Ocorrências Aninhadas” na página 367](#).
- Em casos onde o esquema permite recipientes de dados alternativos, definidos com `xs:choice`.
- Em casos onde o esquema permite que falte um recipiente de dados, definido com `minOccurs = 0`.

Nesses casos, é prudente atribuir a propriedade de `origem` explicitamente.

O Recipiente de Dados Deve Existir

A propriedade de `origem` identifica um recipiente de dados que já existe no escopo da transformação. Se o recipiente de dados não existir, o componente que contém a propriedade de `origem` falhará.

Por exemplo, suponha que a propriedade de origem de um `Grupo` contenha um `LocatorByOccurrence` não opcional que aponta para a terceira ocorrência de um recipiente de dados. Se somente duas ocorrências existirem, o `Grupo` falhará.

Usando a Propriedade de Origem para Entrada ou Saída

Normalmente, um componente usa a propriedade `origem` para identificar onde ele deve obter a entrada. Por exemplo, um `GroupSerializer` pode usar a propriedade para identificar uma ocorrência que deve ser serializada.

Também é possível usar a propriedade para identificar onde o componente deve armazenar a saída. Por exemplo, suponha que um Analisador já tenha criado 10 ocorrências de um elemento XML. Depois que as ocorrências foram criadas, uma âncora `Grupo` atribui um atributo em uma ocorrência do elemento. O `Grupo` pode usar a propriedade `origem` para identificar a ocorrência.

Exemplo 1: Recipientes de Dados com Várias Ocorrências Aninhadas

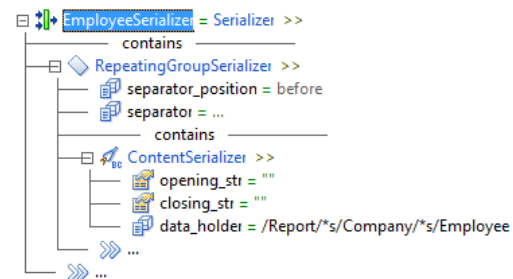
Suponha que o esquema de entrada de um serializador exige a seguinte estrutura:

```
<Report>
  <Company>
    <Employee>John</Employee>
    <Employee>Leslie</Employee>
    <Employee>Pedro</Employee>
  </Company>
  <Company>
    <Employee>Marie</Employee>
    <Employee>Larry</Employee>
    <Employee>Frances</Employee>
  </Company>
</Report>
```

Você deseja iterar por meio de todos os elementos de `Funcionário` e produzir a saída seguinte:

```
John
Leslie
Pedro
Marie
Larry
Frances
```

Você pode criar um `RepeatingGroupSerializer` e configurá-lo para a saída do recipiente de dados `Funcionário`.

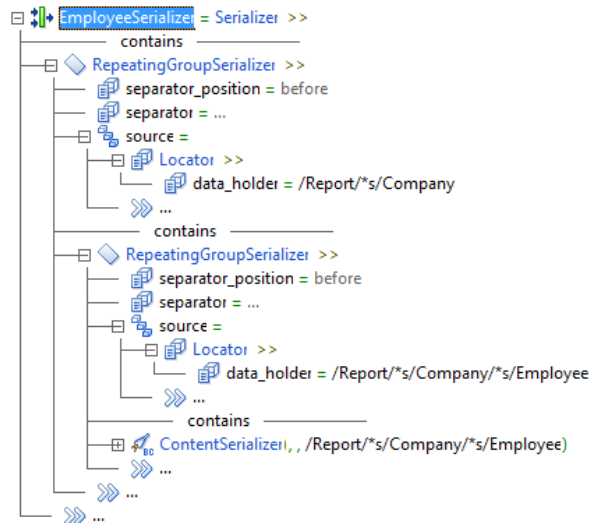


Isso não funciona corretamente. Por padrão, cada iteração seleciona uma nova instância de `Funcionário` dentro da mesma `Empresa`. O resultado é a seguinte saída:

```
John
Leslie
Pedro
```

Em outras palavras, o `RepeatingGroupSerializer` acessa somente a primeira `Empresa`.

Você pode resolver o problema ao aninhar o `RepeatingGroupSerializer` dentro de outro `RepeatingGroupSerializer`. Para resolver qualquer potencial ambiguidade, você pode configurar as propriedades de origem explicitamente.

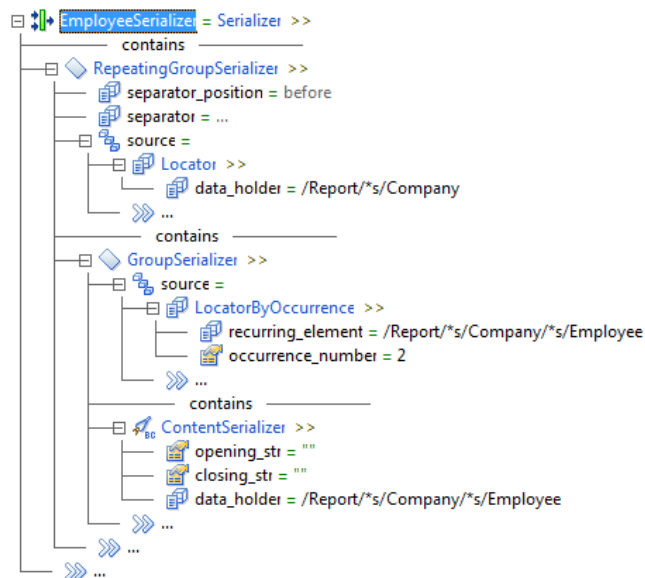


Cada iteração externa de `RepeatingGroupSerializer` processa uma ocorrência de diferente de `Empresa`. Cada iteração aninhada de `RepeatingGroupSerializer` processa uma ocorrência de diferente de `Funcionário`. O resultado é a saída desejada.

Como alternativa, suponha que você deseja iterar somente o segundo elemento `Funcionário` de cada `Empresa`. A saída desejada é:

```
Leslie
Larry
```

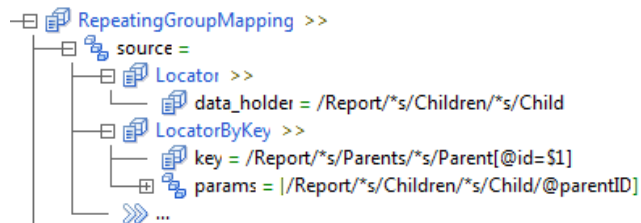
Você pode fazer isso configurando um único `RepeatingGroupSerializer`, cuja origem é `Empresa`. Isso faz com que cada iteração acesse a próxima instância de `Empresa`. Dentro do iteração, você pode configurar um `GroupSerializer`, cuja propriedade de origem usa um `LocatorByOccurrence` para selecionar o segundo `Funcionário`. Isso gera a saída desejada.



Exemplo 2: Indexação

No exemplo de indexação por chave no início deste capítulo, usamos um `RepeatingGroupMapping` configurado conforme indicado a seguir. Neste exemplo, a propriedade `source` identifica dois recipientes de dados:

- Ela usa um componente `Localizador` para identificar uma ocorrência de `Child`. Cada iteração processa a próxima ocorrência de `Child` em sequência.
- Ela usa um componente `LocatorByKey` para identificar uma ocorrência de `Parent`. Isso faz com que cada iteração processe a ocorrência de `Parent` que corresponde à ocorrência de `Child`.



Para obter mais informações, consulte [“Exemplo de Indexação por Chave” na página 363](#).

Propriedade de Destino

A propriedade **destino** identifica uma ocorrência de um retentor de dados que pode ou não já existir. Se a ocorrência existir, o componente a usará. Se a ocorrência não existir, o componente a criará.

O valor da propriedade **destino** é uma lista que contém um ou mais dos seguintes componentes:

Destino	Descrição
Localizador	Identifica um retentor de dados de ocorrência única ou múltiplas ocorrências. No último caso, cada iteração cria uma nova ocorrência.
LocatorByKey	Identifica uma ocorrência de um retentor de dados de múltiplas ocorrências por uma chave de indexação. Se a ocorrência ainda não existir, ela será criada.
LocatorByOccurrence	Identifica uma ocorrência de um retentor de dados de múltiplas ocorrências por número. Se a ocorrência ainda não existir, ela será criada juntamente com qualquer ocorrência de interposição necessária. Por exemplo, se houver quatro ocorrências e LocatorByOccurrence especificar a décima ocorrência, as ocorrências 5 a 9 também serão criadas, mas estarão vazias.

Comportamento Padrão

Se você não atribuir a propriedade **target** de um componente, esse componente identificará recipientes de dados da seguinte maneira:

- Se o esquema permitir apenas uma ocorrência do recipiente de dados, o Script acessará ou criará essa ocorrência.
- Se o recipiente de dados puder ter ocorrências múltiplas, o comportamento será o seguinte:
 - Em um contexto iterativo, por exemplo, dentro de **RepeatingGroup**, cada iteração cria uma nova ocorrência do recipiente de dados.
 - Em um contexto não iterativo, por exemplo, um **Grupo** não aninhado dentro de um componente iterativo, o componente cria uma nova ocorrência do recipiente de dados.

Ambiguidades no Comportamento Padrão

O comportamento padrão poderá apresentar ambiguidades. As ambiguidades poderão surgir nas circunstâncias a seguir.

- Em casos onde um elemento de várias ocorrências está aninhado com outro elemento de várias ocorrências. Para obter mais informações, consulte [“Exemplo 1: recipientes de dados aninhados de ocorrência múltipla” na página 370](#).
- Em casos onde o esquema permite recipientes de dados alternativos, definidos com `xs:choice`.
- Em casos onde o esquema permite que falte um recipiente de dados, definido com `minOccurs = 0`.

Nesses casos, é prudente atribuir a propriedade de `destino` explicitamente.

Recipiente de dados Pode Ser Criado

A propriedade de `destino` identifica um recipiente de dados que pode ou não já existir no escopo da transformação. Se o recipiente de dados não existir, ele será criado.

Por exemplo, suponha que a propriedade de `destino` de um `Grupo` contém um `LocatorByKey` que aponta para uma determinada ocorrência de um recipiente de dados. Se a ocorrência já existir, o `Grupo` a usará. Se a ocorrência não existir, o `Grupo` a criará.

Usando a Propriedade de Destino para Entrada ou Saída

Normalmente, um componente usa a propriedade `destino` para identificar onde ele deve armazenar a saída. Por exemplo, um `Grupo` pode usar a propriedade para identificar uma ocorrência que indica onde os dados são armazenados.

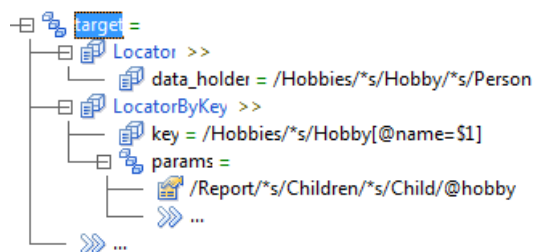
Também é possível usar a propriedade para identificar onde um componente deve obter a entrada. Por exemplo, suponha que um `GroupSerializer` contém uma ação que calcula os dados e os armazena em uma variável. O `GroupSerializer` ativa um `ContentSerializer` que grava a variável na saída. Você pode usar a propriedade `destino` para criar a ocorrência da variável que o `GroupSerializer` usa. A variável atua como a entrada do `ContentSerializer`.

Exemplo 1: recipientes de dados aninhados de ocorrência múltipla

O exemplo de localizadores no início deste capítulo ilustra como usar a propriedade `target` para diferenciar entre recipientes de dados pai e filho de ocorrência múltipla. Para obter mais informações, consulte [“Exemplo de Localizadores” na página 361](#).

Exemplo 2: Indexação

O exemplo de indexação por chave no início deste capítulo ilustra como usar a propriedade `target` com indexação. A seguinte figura ilustra como configurar a propriedade `target` de `RepeatingGroupMapping`:



A propriedade `target` identifica os seguintes recipientes de dados:

- Um componente `Localizador` identifica uma ocorrência de `Person`. Cada iteração cria uma nova ocorrência de `Person`.
- Um componente `LocatorByKey` identifica a ocorrência do elemento `Hobby`, no qual a ocorrência de `Person` deve estar aninhada. Se o elemento `Hobby` já existir, a transformação o usará. Se o elemento `Hobby` ainda não existir, a transformação o criará.

Para obter mais informações, consulte [“Exemplo de Indexação por Chave” na página 363](#).

Localizador Padrão e Propriedades de Chaves

A seguinte tabela descreve as propriedades padrão que são usadas nos componentes localizador e chave:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é <code>Desmarcado</code> . Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componentes de Localizador e Chave

Componentes de localizador e chave identificam elementos no Script ou em recipientes de dados.

Chave

O componente **Key** define atributos ou elementos que servem como um identificador exclusivo de seu elemento pai.

É possível definir um componente **Key** somente no nível global do Script e fazer referência a ele em qualquer parte do Script. O nome de um componente **Key** faz distinção entre maiúsculas e minúsculas.

A seguinte tabela descreve as propriedades do componente **Key**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
recurring_element	Define um elemento de ocorrência múltipla cujas ocorrências são identificadas pela chave.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
unique_fields	Define a chave.

Exemplo

O Exemplo de Indexação por Chave define uma chave para o elemento de Passatempo na estrutura a seguir:

```
<Hobbies>
  <Hobby name="Swimming">
    <Person firstName="Eric" lastName="Smith"/>
    <Person firstName="Edward" lastName="Doe"/>
  </Hobby>
  <Hobby name="Biking">
    <Person firstName="Elizabeth" lastName="Doe"/>
  </Hobby>
  <Hobby name="Painting">
    <Person firstName="Mary" lastName="Smith"/>
  </Hobby>
</Hobbies>
```

A chave é o atributo `nome`, que identifica exclusivamente cada Passatempo.

```
HobbyKey = Key >>
├── recurring_element = /Hobbies/*s/Hobby
├── unique_fields =
└── /Hobbies/*s/Hobby/@name
    >> ...
```

Chaves de Composição

Opcionalmente, você pode definir uma lista de recipientes de dados como uma chave de composição. Para fazer isso, aninhe vários recipientes de dados sob a propriedade `unique_fields`.

Considere o seguinte exemplo:

```
<Persons>
  <Person ID="17" SubID="A">Bob</Person>
  <Person ID="17" SubID="B">Jane</Person>
  <Person ID="35" SubID="A">Larry</Person>
</Persons>
```

Nem o atributo `ID` nem o atributo `SubID` identificam um elemento `Pessoa` de forma exclusiva. A combinação de `ID` e `SubID`, no entanto, é um identificador exclusivo. Você pode definir `ID` e `SubID` como uma chave de composição.

Restrições de Chave

O `unique_fields` deve estar aninhado dentro do `recurring_element`. Eles podem ser atributos do elemento, eles podem ser elementos aninhados em um nível de aninhamento ou podem ser atributos dos elementos aninhados.

Por exemplo, isso significa que `Persons/Person/SocialSecurity/@Number` pode ser uma chave válida para `Persons/Person`, porque `@Number` está aninhado dentro de `Persons/Person`. Por outro lado, `Pessoas/Filho` não é uma chave válida para `Pessoas/Pessoa` porque ele não está aninhado corretamente.

O `unique_fields` deve identificar o ancestral mais próximo que pode ter ocorrências múltiplas. Por exemplo, se `Pai` e `Filho` forem elementos de ocorrência múltipla, então `Parent/Child/@name` pode ser uma chave válida para `Pai/Filho`, mas não para `Pai`.

O `unique_fields` deve ter tipos de dados simples. Eles não podem ser estruturas.

Ocorrências de Irmão e não Irmão

Uma chave identifica exclusivamente as ocorrências irmãs de um elemento. É permitido que as ocorrências não irmãs tenham a mesma chave.

Considere a seguinte estrutura XML:

```
<Report>
  <Company>
    <Employee ID="1">John</Employee>
    <Employee ID="2">Leslie</Employee>
  </Company>
  <Company>
    <Employee ID="1">Marie</Employee>
    <Employee ID="2">Larry</Employee>
  </Company>
</Report>
```

O atributo `ID` pode ser uma chave válida para `Funcionário` porque ele identifica exclusivamente um `Funcionário` em uma única `Empresa`. A duplicação de valores de `ID` em diferentes elementos `Empresa` não invalida a chave.

Chaves de Elementos Reutilizáveis

Você pode definir uma chave em um elemento reutilizável que é definido no esquema.

Por exemplo, suponha que `Pessoas/Pessoa` pode ocorrer em vários contextos diferentes dentro do XML. Se você definir `ID` como uma chave para `Pessoas/Pessoa`, a chave será válida em qualquer contexto onde `Pessoas/Pessoa` for usado.

Exclusividade Forçada de uma Chave

O Script impõe a exclusividade de uma **Chave**. Isso tem as seguintes consequências:

- Se duas ou mais ocorrências de mesmo nível de um elemento de entrada tiverem os mesmos valores de chave, o Script considerará que cada ocorrência substitui as ocorrências anteriores. Ele utilizará apenas a última ocorrência que encontrar.
- Se uma ocorrência de um elemento de entrada não tiver um valor de chave, essa ocorrência será ignorada.
- Se o Script gerar um elemento em chave como saída, e já existir um elemento de mesmo nível com o mesmo valor de chave, a ocorrência existente será substituída.

Nesses casos, o Script grava um aviso no log de eventos.

Localizador

O **Localizador** identifica um recipiente de dados de ocorrência única ou de ocorrência múltipla nas propriedades de **origem** e **destino**. Em recipientes de dados de ocorrência múltipla, cada iteração de um componente que usa o **Localizador** processa a próxima ocorrência do recipiente de dados.

A tabela a seguir descreve as propriedades do componente **Localizador**:

Propriedade	Descrição
data_holder	Define o recipiente de dados que o componente Localizador identifica.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

LocatorByKey

O componente **LocatorByKey** identifica uma ocorrência de um recipiente de dados de ocorrência múltipla nas propriedades **source** e **target**.

Antes de usar **LocatorByKey**, você deve definir uma **Chave** no nível global do Script. A **Chave** especifica os recipientes de dados que identificam exclusivamente a ocorrência.

A seguinte tabela descreve as propriedades do componente **LocatorByKey**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
key	Define a representação de predicado XPath da chave. Por exemplo, se você tiver definido <code>Hobbies/Hobby/@name</code> como uma Chave , selecione <code>Hobbies/Hobby[@name=\$1]</code> . Essa propriedade é necessária.

Propriedade	Descrição
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
params	Define os valores dos parâmetros no predicado XPath. Cada valor tem um sinal de cifrão (\$) e um número inteiro que representa a posição do parâmetro na lista de parâmetros. Essa propriedade é necessária.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Conflitos Entre Localizadores

Em caso de conflitos, um **LocatorByKey** aninhado substituirá um localizador pai.

Por exemplo, suponha que a propriedade de **destino** de um **Grupo** contém um **LocatorByKey** que aponta para a terceira ocorrência de um elemento. Um **Grupo** aninhado contém um **LocatorByKey** apontando para a quinta ocorrência. O **Grupo** aninhado usará a quinta ocorrência.

LocatorByOccurrence

O componente **LocatorByOccurrence** é usado na propriedade **source** para identificar uma ocorrência de um recipiente de dados de ocorrência múltipla.

Por exemplo:

- Um elemento que pode ocorrer várias vezes em um documento XML
- Uma variável que pode ocorrer várias vezes

O componente identifica a ocorrência pelo número. Por exemplo, se houver dez ocorrências de um recipiente de dados, você poderá usar **LocatorByOccurrence** para processar a terceira ocorrência.

LocatorByOccurrence pode ser usado para iteração ao longo das ocorrências em uma estrutura que se repete, como uma âncora **RepeatingGroup**.

O componente **LocatorByKey** identifica uma ocorrência de um recipiente de dados de ocorrência múltipla nas propriedades **source** e **target**.

Antes de usar **LocatorByKey**, você deve definir uma **Chave** no nível global do Script. A **Chave** especifica os recipientes de dados que identificam exclusivamente a ocorrência.

A seguinte tabela descreve as propriedades do componente **LocatorByOccurrence**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
occurrence_number	Define o número da ocorrência.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
recurring_element	Define o recipiente de dados que o componente identifica.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Conflitos Entre Localizadores

Em caso de conflitos, um **LocatorByOccurrence** aninhado substituirá um localizador pai.

Por exemplo, suponha que a propriedade de **destino** de um **Grupo** contém um **LocatorByOccurrence** que aponta para a terceira ocorrência de um elemento. Um **Grupo** aninhado contém um **LocatorByOccurrence** apontando para a quinta ocorrência. O **Grupo** aninhado usará a quinta ocorrência.

CAPÍTULO 22

Streamers

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Streamers, 377](#)
- [Streamers de Texto, 378](#)
- [XML Streamers, 382](#)
- [Propriedades Padrão do Streamer, 384](#)
- [Referência de Componente de Streamer, 385](#)
- [Referência de Subcomponente de Streamer, 394](#)

Visão Geral de Streamers

Um Streamer divide um grande documento de origem em partes menores que uma transformação pode processar separadamente. Streamers são úteis em transformações que processam entradas muito grandes, como fluxos de dados multigigabytes. Um Streamer pode ter uma entrada de buffer ou um arquivo de entrada.

Um Streamer oferece as seguintes vantagens:

- A transformação analisa cada segmento de origem quando estiver disponível, em vez de aguardar até que ele receba toda a origem.
- A transformação tem requisitos de memória reduzidos.

Por exemplo, um fluxo de entrada de Streamer pode conter dados de transações do mercado de ações. O fluxo transmite para um servidor continuamente no decorrer do dia de negócios. Um Script com um Streamer processa cada transação quando ela é recebida, em vez de aguardar até o final do dia.

Em outro exemplo, você recebe um grande arquivo de origem por meio de uma conexão FTP. Usando um Streamer, o Script pode processar o arquivo antes que ele seja recebido completamente.

A transformação do Processador de Dados fornece os seguintes tipos de Streamers:

- Streamer. Processa entradas de texto extensas.
- XmlStreamer. Processa entradas XML extensas.

Streamers são componentes executáveis. Defina o componente de **Streamer** ou de **XmlStreamer** no nível global do Script e defina-o como o componente de inicialização da transformação. O Streamer funciona dividindo a entrada em segmentos e transmitindo-os a outros componentes executáveis, que podem ser Analisadores, Mapeadores ou Serializadores.

Streamers de Texto

Um componente `Streamer` divide um grande documento de texto em partes menores. O `Streamer` divide o texto de origem em cabeçalho, rodapé e segmentos repetidos. Conforme exigido pela estrutura de origem, o `Streamer` pode subdividir os segmentos repetidos em cabeçalhos, rodapés aninhados e segmentos repetidos. O `Streamer` pode passar cada segmento para uma transformação apropriada.

Segmentos

Um `Streamer` identifica os segmentos da sua entrada. Ele transmite os segmentos individualmente para as transformações, como os Analisadores, os Mapeadores ou os Serializadores, que processam os dados do segmento.

Um `Streamer` pressupõe que a origem é composta de:

- Um segmento de cabeçalho
- Qualquer número de segmentos repetidos
- Um segmento de rodapé

Para cada tipo de segmento, o `Streamer` define uma transformação que processa o segmento.

Os segmentos repetidos podem ser simples ou complexos. Um segmento simples é uma única unidade de dados. Um segmento complexo tem seus próprios cabeçalho, segmentos repetidos e rodapé.

Cabeçalhos e rodapés são sempre segmentos simples.

Segmentos Simples

Um segmento simples tem um marcador de abertura que identifica onde ele começa e um marcador de fechamento que identifica onde ele termina. Portanto, um segmento simples tem a seguinte estrutura:

```
Opening marker
Data
Closing marker
```

O `Streamer` transmite o segmento ao componente de transformação especificado, como um Analisador.

Você pode omitir alguns marcadores da definição do `Streamer`. Por exemplo:

- Se você omitir o marcador de abertura do cabeçalho de origem, o cabeçalho será considerado para começar no início da origem.
- Se você omitir o marcador de fechamento, o segmento terminará no marcador de abertura do próximo segmento.

Segmentos Complexos

Um segmento complexo tem um cabeçalho e um rodapé. Entre o cabeçalho e o rodapé, ele pode conter qualquer número de segmentos simples aninhados, por exemplo:

```
Header
Simple segment
Simple segment
Simple segment
Footer
```

Um segmento complexo pode conter segmentos complexos aninhados, por exemplo:

```
Header
Complex segment
```

```
Complex segment
Complex segment
Footer
```

Você também pode definir um segmento complexo sem cabeçalho e rodapé, por exemplo:

```
Simple segment
Simple segment
Simple segment
```

Os segmentos simples aninhados devem ser do mesmo tipo. Ou seja, eles devem ser identificados pelo mesmos marcadores de abertura e fechamento.

Exemplo

Um fluxo de dados contém dados de transação de ações. O fluxo tem a seguinte estrutura:

- O cabeçalho começa com a string `yy-MM-dd/`, que é uma data, seguido por uma barra.
- O cabeçalho contém vários dados, seguidos pela string `ENDHEAD/`.
- Os segmentos de repetição começam com a string `TRANS HH:mm nnn/`, onde `HH:mm` é o tempo em um relógio de 24 horas e `nnn` é um número de série de qualquer tamanho.
- O fluxo de dados é encerrado com a string `END/`.

O exemplo a seguir é um fluxo de dados em conformidade com essa especificação, onde ... representa dados arbitrários que devem ser analisados:

```
06-12-13/...ENDHEAD/TRANS 09:30 1...TRANS 09:30 2...TRANS 09:31 03...TRANS 09:32
14...END/
```

Você pode analisar esse fluxo usando um Streamer com a estrutura esquemática a seguir. Observe que os marcadores de abertura e fechamento estão localizados procurando um determinado padrão ou string.

Segmento	Tipo	Marcador de Abertura	Marcador de Fechamento
Cabeçalho	Simple	[0-9][0-9]-[0-9][0-9]-[0-9][0-9]/	ENDHEAD/
Repetição	Simple	TRANS [0-9][0-9]:[0-9][0-9] [0-9]+/	none
Rodapé	Simple	END/	none

Concatenação de Cabeçalho

Opcionalmente, você pode configurar um Streamer para concatenar o segmento de cabeçalho com cada segmento repetido. O Streamer transmite o resultado concatenado para uma transformação.

Por exemplo, suponha que um Streamer transmita o segmento repetido para um Analisador. A origem tem a seguinte estrutura, onde `Segment1` e assim por diante são instâncias do segmento de repetição:

```
Header
Segment1
Segment2
Segment3
```

Se você selecionar a opção de concatenação, o Streamer enviará os seguintes dados para o Analisador:

```
HeaderSegment1
HeaderSegment2
HeaderSegment3
```

Saída de um Streamer

Um Streamer gera um documento de saída independente para cada um dos segmentos de origem.

Saída no Ambiente de Design

Se você executar um Streamer no ambiente de design, ele combinará os segmentos de saída individuais em uma única saída.

Por exemplo, suponha que o Streamer transmita cada segmento para um Analisador. A saída de cada Analisador é um documento XML. A saída combinada é uma sequência de documentos XML, por exemplo:

```
<?xml version="1.0" encoding="windows-1252"?>
<header>...</header>

<?xml version="1.0" encoding="windows-1252"?>
<repeating_segment>...</repeating_segment>

<?xml version="1.0" encoding="windows-1252"?>
<repeating_segment>...</repeating_segment>

<?xml version="1.0" encoding="windows-1252"?>
<footer>...</footer>
```

Essa saída não é um XML bem formado porque ele contém vários elementos raiz.

Saída de Disposição em uma Marca Raiz

Você pode encapsular a saída combinada de um Streamer em uma marca raiz para converter a saída em XML bem formado.

Na guia Geração de XML das configurações de transformação do Processador de Dados, selecione **Adicionar elemento raiz XML** e insira o nome do elemento raiz do wrapper.

Por exemplo, se você especificar um elemento raiz de wrapper chamado `MyRoot`, a saída se tornará:

```
<MyRoot>
  <?xml version="1.0" encoding="windows-1252"?>
  <header>...</header>

  <?xml version="1.0" encoding="windows-1252"?>
  <repeating_segment>...</repeating_segment>

  <?xml version="1.0" encoding="windows-1252"?>
  <repeating_segment>...</repeating_segment>

  <?xml version="1.0" encoding="windows-1252"?>
  <footer>...</footer>
</MyRoot>
```

Usando Marcadores e Variáveis em Streamers

Em um componente `Streamer`, você não pode usar os componentes `Marcador` e `Variável` regulares que são usados em outros tipos de transformações. Em vez disso, você deve usar o componente `MarkerStreamer` para definir os marcadores de abertura e de fechamento dos segmentos simples. Você pode usar o componente `StreamerVariable` para armazenar os dados temporários que são compartilhados por todos os segmentos.

Criando um Streamer

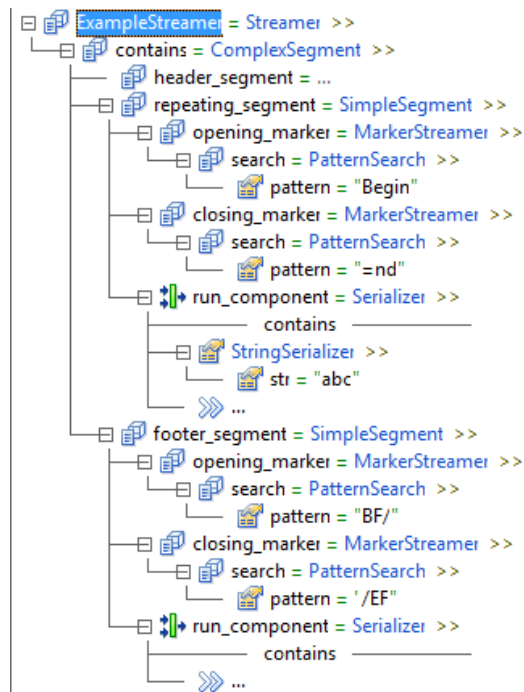
1. Analise a estrutura de origem e identifique os tipos de segmentos.

2. Crie ou abra um Script.
3. No Script, configure uma transformação, como um Analisador, um Mapeador ou um Serializador, que possa processar cada tipo de segmento simples.
4. No mesmo Script, configure um componente **Streamer**.
5. Dentro do **Streamer**, aninhe segmentos **ComplexSegment** e **SimpleSegment** correspondentes à estrutura de origem.
6. Para cada **SimpleSegment**, defina o marcador de abertura e o marcador de fechamento, se necessário. Defina a transformação que processa o segmento.
7. Defina o **Streamer** como o componente de inicialização.

Exemplo 1 de Configuração de Streamer

O seguinte Streamer contém segmentos simples. Cada segmento tem um marcador de abertura e de fechamento predefinido.

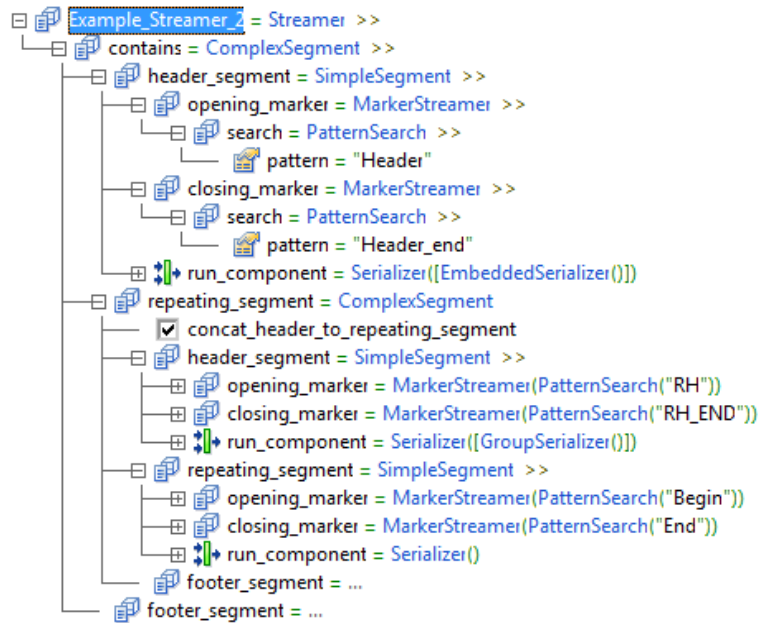
O Streamer transmite o cabeçalho e os segmentos repetidos para um Analisador denominado `body_p`. Ele transmite o rodapé a um Analisador denominado `foot_p`.



Exemplo 2 de Configuração de Streamer

O Streamer a seguir contém um `ComplexSegment` aninhado e repetido. O segmento aninhado `ComplexSegment` tem seu próprio cabeçalho e `SimpleSegment` aninhado e repetido. O `ComplexSegment` não tem um rodapé.

Observe que a propriedade `concat_header_to_repeating_segment` foi selecionada. O efeito dessa propriedade é concatenar o cabeçalho para cada instância do segmento repetido. O `Streamer` transmite os segmentos concatenados ao Analisador `body_p`.



XML Streamers

Um componente `XmlStreamer` divide um documento XML grande em partes menores. O `XmlStreamer` divide a origem XML em segmentos de cabeçalho, corpo e rodapé. Os segmentos de corpo podem conter elementos repetidos ou não repetidos. O `XmlStreamer` pode transmitir cada segmento de XML para uma transformação apropriada, normalmente um Mapeador ou um Serializador.

Um `XmlStreamer` funciona de maneira bastante semelhante a um `Streamer`, com algumas diferenças devido à entrada XML estruturada. Veja a seguir os principais recursos:

- Os segmentos de corpo são definidos como elementos XML. É possível configurar o corpo com vários elementos do mesmo tipo ou de tipos diferentes, em qualquer sequência.
- O cabeçalho é definido como a parte inteira do XML antes do primeiro segmento de corpo. Na configuração de `XmlStreamer`, não é necessário definir os elementos que formam o cabeçalho.
- O rodapé é definido como toda a parte do XML após o último segmento de corpo. Na configuração de `XmlStreamer`, não é necessário definir os elementos que formam o rodapé.
- Em muitos casos, os segmentos de cabeçalho e rodapé não são um XML bem formado. Para ativar a transmissão dos segmentos para um Mapeador ou um Serializador, você pode configurar componentes modificadores que convertem esses segmentos em XML bem formado.

Para ajudar a entender esses recursos, considere a seguinte estrutura XML de origem:

```
<stream>
  <headerline1>MainHeader</headerline1>
  <substreams>
    <substream>
      <subheaderline1>SubHeader</subheaderline1>
    <segments>
```

```

        <segment1>Segment1A</segment1>
        <segment1>Segment1B</segment1>
        <segment2>Segment2A</segment2>
        <segment1>Segment1C</segment1>
        <segment2>Segment2B</segment2>
    </segments>
    <subfooterline1>SubFooter</subfooterline1>
</substream>
<substream>...</substream>
<substream>...</substream>
</substreams>
<footerline1>MainFooter</footerline1>
</stream>

```

Neste exemplo, você pode definir os segmentos de corpo como os elementos `substream`. O cabeçalho é tudo o que precede o primeiro `substream`:

```

<stream>
  <headerline1>MainHeader</headerline1>
  <substreams>

```

O rodapé é tudo o que segue o último `substream`:

```

  </substreams>
  <footerline1>MainFooter</footerline1>
</stream>

```

Os segmentos de cabeçalho e rodapé não são um XML bem formado. É possível aplicar modificadores que adicionam marcas de fechamento ou abertura para torná-los bem formados. Por exemplo, um modificador pode converter o cabeçalho em:

```

<stream>
  <headerline1>MainHeader</headerline1>
  <substreams>
  </substreams>
</stream>

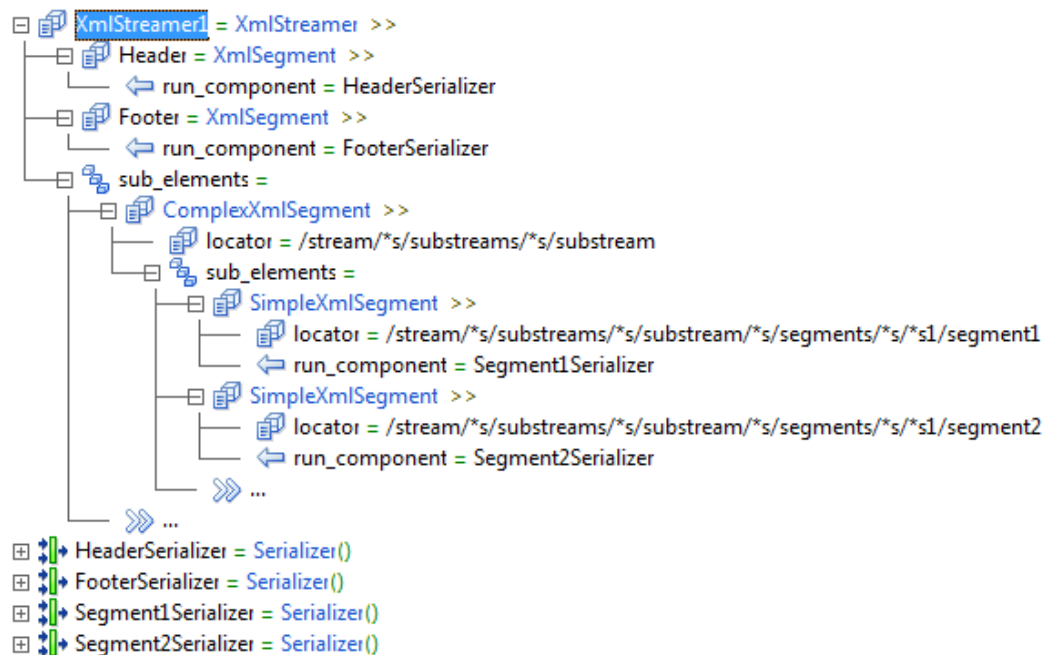
```

Você pode configurar o `XmlStreamer` de forma a transmitir o segmento de cabeçalho, o segmento de rodapé e cada instância do segmento `substream` para uma transformação apropriada, como um Mapeador ou um Serializador.

Nota: Os elementos de segmento de cabeçalho estão disponíveis durante o processamento dos elementos de corpo. No entanto, os elementos de rodapé ainda não estão disponíveis durante o processamento de elementos de corpo. Os elementos de rodapé são processados somente depois que a transformação conclui a leitura dos elementos de corpo.

Como alternativa, você pode subdividir os elementos `substream` em segmentos `segment1` e `segment2` e enviar cada um deles para seu próprio Mapeador ou Serializador. Observe que `segment1` e `segment2` seguem uns aos outros em uma sequência aleatória. O `XmlStreamer` ignora a sequência e processa `segment1` e `segment2` na ordem em que eles ocorrerem.

A seguinte figura ilustra a configuração para essa finalidade. O Script define Serializadores independentes para os segmentos de cabeçalho, de rodapé, `segment1` e `segment2`.



Nota: Mesmo que a execução do componente de rodapé seja exibida antes dos elementos de corpo neste exemplo, os elementos de rodapé serão processados somente depois que a transformação concluir a leitura de elementos de corpo.

Como uma distinção adicional, você pode definir transformações para os cabeçalhos e rodapés aninhados dentro de cada elemento substream.

Propriedades Padrão do Streamer

A seguinte tabela descreve as propriedades comuns dos componentes de Streamer:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Referência de Componente de Streamer

Os streamers dividem um documento de origem grande em partes menores que uma transformação pode processar separadamente.

ComplexSegment

Um componente **ComplexSegment** define uma estrutura de origem que contém um cabeçalho, uma parte repetida e um rodapé.

A seguinte tabela descreve as propriedades do componente **ComplexSegment**:

Propriedade	Descrição
concat_header_to_repeating_segment	Determina se o sistema inclui header_segment com cada instância de repeating_segment . Ele transmite o resultado para run_component de repeating_segment . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selected. Cada segmento repetido tem uma cópia do cabeçalho.- Cleared. Cada segmento repetido aparece sem o cabeçalho. Para obter mais informações, consulte "Concatenação de Cabeçalho" na página 379 . O padrão é Desmarcado.
footer_segment	Define a parte de rodapé da origem. Nessa propriedade, é possível aninhar um SimpleSegment que define o rodapé. Se essa propriedade for definida, o Script processará a origem como se ela não tivesse rodapé.
header_segment	Define a parte de cabeçalho da origem. Nessa propriedade, é possível aninhar um SimpleSegment que define o cabeçalho. Se essa propriedade for definida, o Script processará a origem como se ela não tivesse cabeçalho.
repeating_segment	Define a parte repetida da origem. Nessa propriedade, é possível aninhar um SimpleSegment que define os dados repetidos. Você também pode aninhar um ComplexSegment que tenha sua própria estrutura de cabeçalho/repetição/rodapé.

ComplexXmlSegment

Um componente **ComplexXmlSegment** define uma estrutura aninhada dentro da parte de corpo de uma entrada **XmlStreamer**. A estrutura aninhada pode ter seu próprio cabeçalho, corpo e rodapé.

Em um **ComplexXmlSegment**, é possível aninhar componentes **XmlSegment**, **ComplexXmlSegment** e **SimpleXmlSegment**.

A seguinte tabela descreve as propriedades do componente **ComplexXmlSegment**:

Propriedade	Descrição
allow_unmarked_text	Determina se os segmentos na lista sub_elements podem ser separados por interposição de texto ou por outros elementos. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Os segmentos podem ser separados por interposição de texto ou por outros elementos. O conteúdo de interposição é ignorado. - Limpo. Os segmentos podem ser separados somente por um espaço em branco. O padrão é limpo.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
Rodapé	Define como processar o rodapé de ComplexXmlSegment . Para obter mais informações, consulte "XmlSegment" na página 392 . O padrão é XmlSegment.
Cabeçalho	Define como processar o cabeçalho de ComplexXmlSegment . Para obter mais informações, consulte "XmlSegment" na página 392 . O padrão é XmlSegment.
locator	Define um recipiente de dados.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sub_elements	Define uma lista de componentes ComplexXmlSegment ou SimpleXmlSegment que definem como processar o corpo de ComplexXmlSegment .

JsonStreamer

O **JsonStreamer** aceita uma entrada de arquivo JSON muito grande e divide a entrada em segmentos. Ele transmite cada tipo de segmento a uma transformação predefinida, como um Analisador, Mapeador ou Serializador.

O **JsonStreamer** deve ser definido no nível global do Script e ser o componente de inicialização da transformação.

Nota: O número de segmentos é determinado automaticamente de acordo com a precisão da porta de entrada.

A seguinte tabela descreve as propriedades do componente **JsonStreamer**:

Propriedade	Descrição
run_component	Define uma transformação que processa o segmento. Você pode escolher uma das seguintes opções: <ul style="list-style-type: none">- O nome de um Analisador, Serializador ou Mapeador que está configurado no nível global do Script.- Um componente Mapeador ou Serializador. Configure o Mapeador ou o Serializador no segmento.- O componente WriteSegment que copia o segmento para a saída. Para obter mais informações, consulte "WriteSegment" na página 396.

MarkerStreamer

Um componente **MarkerStreamer** define os marcadores de abertura e de fechamento de segmentos simples. Ele é semelhante a uma âncora de **Marcador** regular, mas é usada somente em Streamers.

A tabela a seguir descreve as propriedades do componente **MarkerStreamer**:

Propriedade	Descrição
adjacente	Determina se o MarkerStreamer deve ser adjacente ao fim do segmento anterior. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Requer que o MarkerStreamer esteja adjacente ao fim do segmento anterior.- Desmarcado. O MarkerStreamer pode ser separado do fim do segmento anterior. O padrão é limpo. Use essa propriedade para garantir que os segmentos não estejam separados por qualquer outro texto ou um espaço em branco.
count	Determina qual ocorrência de marcador para iniciar o processamento. Por exemplo, defina a contagem como 3 para ignorar a primeira e segunda ocorrências de marcador. Essa propriedade está sendo desativada. Ele está disponível para compatibilidade com projetos existentes. Não use-o em novos projetos.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
marcação	Determina se o marcador é usado como um ponto de referência para identificar o próximo segmento ou marcador. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- posição inicial. Somente antes.- posição final. Somente depois.- completo. Coloca um ponto de referência antes e depois do marcador atual. O padrão é completo.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
pesquisar	Define como o MarkerStreamer localiza o texto. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- NewlineSearch. Pesquisa um caractere de nova linha.- OffsetSearch. Ignora um número predefinido de caracteres após o ponto de referência anterior.- PatternSearch. Pesquisas uma expressão regular.- TextSearch. Pesquisa uma string explícita.

Usando a Propriedade de Marcação para Definir os Limites de Segmento

Você pode usar a propriedade de marcação para controlar se os dados no marcador de abertura e fechamento estão incluídos no segmento e são passados para uma transformação.

A regra é que o Streamer transmite os dados entre os pontos de referência mais internos ao redor do segmento. Por exemplo:

- Se o marcador de abertura tiver `marcador = posição inicial`, o ponto de referência mais interno será no início. Todo o marcador está incluído no segmento.
- Se o marcador de abertura tiver `marcador = posição final` ou `,`, o ponto de referência mais interno será no final. O marcador é excluído do segmento.

Os relacionamentos inversos se aplicam ao marcador de fechamento.

Para ilustrar isso, considere um segmento simples com a seguinte estrutura:

```
BEGIN...data...END
```

Um `MarkerStreamer` identifica o marcador de abertura procurando o texto `BEGIN`. Outro `MarkerStreamer` identifica o marcador de fechamento procurando `END`.

A tabela a seguir ilustra como a propriedade de marcação afeta os limites de segmento.

Marcação de um Marcador de Abertura	Marcação de um Marcador de Fechamento	Segmento Transmitido para a Transformação
completo	completo	...data...
completo	iniciar	...data...
completo	finalizar	...data...END
iniciar	completo	BEGIN...data...
iniciar	iniciar	BEGIN...data...
iniciar	finalizar	BEGIN...data...END
finalizar	completo	...data...
finalizar	iniciar	...data...
finalizar	finalizar	...data...END

SimpleSegment

Um componente **SimpleSegment** define uma unidade de dados que contém um marcador de abertura e um marcador de fechamento. Ele também define a transformação que processa a unidade de dados.

Os marcadores de abertura e de fechamento são definidos com expressões regulares. Para obter mais informações sobre a sintaxe de expressões regulares, consulte [“RegularExpression” na página 281](#).

A seguinte tabela descreve as propriedades do componente **SimpleSegment**:

Propriedade	Descrição
closing_marker	Define uma expressão regular que identifica o final do segmento. Se essa propriedade for indefinida, o final do segmento será o final da origem ou o início do próximo segmento. O padrão é <code>MarkerStreamer</code> .
count	Define o número máximo de segmentos a serem transmitidos para a transformação. Por exemplo, se a contagem for 3, o streamer procurará três instâncias consecutivas do segmento. Os três segmentos serão transmitidos em conjunto para a transformação. Se ele localizar apenas um ou dois segmentos, esses segmentos serão transmitidos. Se os segmentos forem pequenos, a transmissão de vários segmentos a uma transformação poderá melhorar o desempenho, pois reduz a sobrecarga do Streamer. Dentro da transformação, use um componente, como RepeatingGroup , para processar os segmentos individuais. O padrão é 1.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é <code>Cleared</code> .
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
opening_marker	Define uma expressão regular que identifica o início do segmento. Se essa propriedade for indefinida, o início do segmento será o início da origem ou o final do segmento anterior. O padrão é <code>MarkerStreamer</code> .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
run_component	Define uma transformação que processa o segmento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- O nome de um Analisador, Serializador ou Mapeador que está configurado no nível global do Script.- Um componente Mapeador ou Serializador. Configure o Mapeador ou o Serializador no segmento.- O componente WriteSegment que copia o segmento para a saída. Para obter mais informações, consulte "WriteSegment" na página 396.

SimpleXmlSegment

Um componente **SimpleXmlSegment** define um segmento de corpo de uma entrada **XmlStreamer**. Ele define o elemento contendo o segmento e a transformação que deve processar esse segmento.

Como um **SimpleXmlSegment** é um elemento XML, o segmento é sempre bem formado. É possível aplicar um modificador que altera o segmento antes de transmitir esse segmento para uma transformação.

A seguinte tabela descreve as propriedades do componente **SimpleXmlSegment**:

Propriedade	Descrição
count	Define o número máximo de segmentos a serem transmitidos para a transformação. Por exemplo, se a contagem for 3, o streamer procurará três instâncias consecutivas do segmento. Os três segmentos serão transmitidos em conjunto para a transformação. Se ele localizar apenas um ou dois segmentos, esses segmentos serão transmitidos. Se os segmentos forem pequenos, a transmissão de vários segmentos a uma transformação poderá melhorar o desempenho, pois reduz a sobrecarga do Streamer. Dentro da transformação, use um componente, como RepeatingGroup , para processar os segmentos individuais. O padrão é 1.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
localizador	Define um recipiente de dados.
modificador	Define como o segmento é modificado antes de ser transmitido para uma transformação. É possível selecionar os seguintes componentes modificadores: - AddHeaderModifier. Transmite o segmento junto com o cabeçalho da seção XML no qual esse segmento está localizado. - AddStringModifier. Concatena o segmento com strings de prefixo ou sufixo. - DoNothingModifier. Não modifica o segmento. - WellFormedModifier. Adiciona marcas de fechamento e/ou um elemento raiz para garantir que o segmento seja um XML bem formado. Para obter mais informações sobre modificadores, consulte a "Referência de Subcomponente de Streamer" na página 394 . O padrão é DoNothingModifier.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
run_component	Define uma transformação que processa o segmento. É possível escolher uma das seguintes opções: - O nome de um Analisador, Serializador ou Mapeador que está configurado no nível global do Script. - Um componente Mapeador ou Serializador . Configure o Mapeador ou o Serializador no segmento. - O componente WriteSegment que copia o segmento para a saída. Para obter mais informações, consulte "WriteSegment" na página 396 .

Streamer

O componente **Streamer** divide a entrada de texto em segmentos. Ele transmite cada tipo de segmento a uma transformação predefinida, como um Analisador, um Mapeador ou um Serializador.

O **Streamer** deve ser definido no nível global do Script e deve ser o componente de inicialização da transformação.

Em um **Streamer**, é necessário aninhar um **ComplexSegment**. O **ComplexSegment** pode conter componentes **SimpleSegment** ou **ComplexSegment** aninhados.

A seguinte tabela descreve as propriedades do componente **Streamer**:

Propriedade	Descrição
contains	Define a estrutura geral da origem. O padrão é ComplexSegment.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
max_lookup_size	Define a quantidade máxima de novos dados, em quilobytes, que o componente Streamer pesquisa para cada novo segmento. Para obter um desempenho ideal, defina essa propriedade como duas vezes o tamanho máximo possível do segmento. Quando um aplicativo ativa um serviço Streamer implantado por meio de uma API, ele deve definir o parâmetro de tamanho do bloco como um valor menor que max_lookup_size . O padrão é 10000.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
on_end_of_input	Define uma transformação que é executada no final do fluxo de entrada. Por exemplo, a transformação pode gerar a saída de uma mensagem de resumo. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- O nome de um Analisador, Serializador ou Mapeador que está configurado no nível global do Script.- Um componente Mapeador ou Serializador. Configure o mapeador ou o serializador no Streamer.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
root_tag	Define uma marca XML na qual o Streamer encapsula a saída combinada de todos os segmentos. Para obter mais informações, consulte “Saída de um Streamer” na página 380 .

StreamerVariable

Um componente **StreamerVariable** é uma variável definida pelo usuário cujo escopo inclui todos os segmentos de um **Streamer** ou **XmlStreamer**.

Por exemplo, se um Streamer contiver três Analisadores, o valor de um **StreamerVariable** estará disponível para todos os três Analisadores. Um Analisador que processa um segmento de cabeçalho pode recuperar dados desse cabeçalho e armazená-los no **StreamerVariable**. Os outros Analisadores, que processam o segmento repetido e o segmento de rodapé, podem acessar o valor do **StreamerVariable**. Não é possível usar uma **Variável** comum para essa finalidade, pois o valor da variável não é compartilhado entre os segmentos.

Em outros sentidos, o componente **StreamerVariable** é semelhante a uma **Variável** comum. No entanto, um **StreamerVariable** deve ter um tipo de dados simples de ocorrência única. Para obter mais informações, consulte [“Variáveis” na página 197](#).

É possível definir um **StreamerVariable** somente no nível global do Script.

A seguinte tabela descreve as propriedades do componente **StreamerVariable**:

Propriedade	Descrição
initialization	Um valor inicial para StreamerVariable , atribuído quando a transformação é iniciada. Selecione InitialValue e insira o valor.
val_type	Define o tipo de dados que a variável pode armazenar. Atribua um tipo simples, como <code>xs:string</code> ou <code>xs:integer</code> . Variáveis de streamer não podem ter tipos complexos ou de ocorrência múltipla. O padrão é <code>xs:string</code> .

XmlSegment

Um componente **XmlSegment** define um segmento de cabeçalho ou rodapé de uma entrada **XmlStreamer**. Ele também define a transformação que processa o cabeçalho ou rodapé.

Um cabeçalho ou rodapé não modificado não é necessariamente um XML bem formado. Ao atribuir um componente modificador, você pode configurar o **XmlSegment** de forma a sempre retornar um XML bem formado.

A seguinte tabela descreve as propriedades do componente **XmlSegment**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
modificador	Define como o segmento é modificado antes de ser transmitido para uma transformação. É possível selecionar os seguintes componentes modificadores: <ul style="list-style-type: none"> - AddHeaderModifier. Transmite o segmento junto com o cabeçalho da seção XML no qual esse segmento está localizado. - AddStringModifier. Concatena o segmento com strings de prefixo ou sufixo. - DoNothingModifier. Não modifica o segmento. Esse é o padrão. - WellFormedModifier. Adiciona marcas de fechamento e/ou um elemento raiz para garantir que o segmento seja um XML bem formado. Para obter mais informações sobre modificadores, consulte a "Referência de Subcomponente de Streamer" na página 394 . O padrão é WellFormedModifier.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
run_component	Define uma transformação que processa o segmento. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - O nome de um Analisador, Serializador ou Mapeador que está configurado no nível global do Script. - Um componente Mapeador ou Serializador. Configure o Mapeador ou o Serializador no segmento. - O componente WriteSegment que copia o segmento para a saída. Para obter mais informações, consulte "WriteSegment" na página 396.

XmlStreamer

O componente **XmlStreamer** divide uma entrada XML em segmentos de cabeçalho, corpo e rodapé. Ele transmite cada tipo de segmento para uma transformação predefinida, como um Mapeador ou um Serializador.

O **XmlStreamer** deve ser definido no nível global do Script e deve ser o componente de inicialização da transformação.

Em um **XmlStreamer**, é possível aninhar componentes **XmlSegment**, **ComplexXmlSegment** e **SimpleXmlSegment**.

A seguinte tabela descreve as propriedades do componente **XmlStreamer**:

Propriedade	Descrição
allow_unmarked_text	Determina se os segmentos na lista sub_elements podem ser separados por interposição de texto ou por outros elementos. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Os segmentos podem ser separados por interposição de texto ou por outros elementos. O conteúdo de interposição é ignorado.- Limpo. Os segmentos podem ser separados somente por um espaço em branco. O padrão é limpo.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
Rodapé	Define como processar o rodapé de ComplexXmlSegment . Para obter mais informações, consulte "XmlSegment" na página 392 . O padrão é XmlSegment.
Cabeçalho	Define como processar o cabeçalho de ComplexXmlSegment . Para obter mais informações, consulte "XmlSegment" na página 392 . O padrão é XmlSegment.
max_lookup_size	Define a quantidade máxima de novos dados, em quilobytes, que o componente XmlStreamer pesquisa para cada novo segmento. Para obter um desempenho ideal, defina essa propriedade como duas vezes o tamanho máximo possível do segmento. Quando um aplicativo ativa um serviço XmlStreamer implantado por meio de uma API, ele deve definir o parâmetro de tamanho do bloco como um valor menor que max_lookup_size . O padrão é 10000.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
sub_elements	Define uma lista de componentes ComplexXmlSegment ou SimpleXmlSegment que definem como processar o corpo da entrada XML.

Referência de Subcomponente de Streamer

Os subcomponentes do streamer modificam os segmentos de um **Streamer** ou um **XmlStreamer**.

AddHeaderModifier

Em um **XmlStreamer**, o componente **AddHeaderModifier** adiciona o cabeçalho do segmento atual ao segmento. O componente adiciona marcas de fechamento conforme exigido para garantir que o resultado seja um XML bem formado.

Você pode usar **AddHeaderModifier** para passar um segmento para uma transformação no contexto do seu cabeçalho.

A tabela a seguir descreve as propriedades do componente **AddHeaderModifier**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

No exemplo a seguir, `<segment1>` é um segmento de repetição, precedido de um cabeçalho e seguido de um rodapé.

```
<stream>
  <headerline1>...</headerline1>
  <segments>
    <segment1>...</segment1>
    <segment1>...</segment1>
    <segment1>...</segment1>
  </segments>
  <footerline1>...</footerline1>
</stream>
```

Você pode configurar um **XmlStreamer** que retorna o seguinte cabeçalho, que não é um XML bem formado:

```
<stream>
  <headerline1>...</headerline1>
  <segments>
```

Se você aplicar **AddHeaderModifier** ao `<segment1>`, o modificador coloca um prefixo com o cabeçalho em cada instância de `<segment1>`. Ele adiciona marcas de fechamento para garantir que o XML fique bem formado. O resultado é o seguinte segmento:

```
<stream>
  <headerline1>...</headerline1>
  <segments>
    <segment1>...</segment1>
  </segments>
</stream>
```

Se você aplicar **AddHeaderModifier** a um segmento de cabeçalho, o modificador adicionará o cabeçalho do elemento pai ao segmento. Não aplique **AddHeaderModifier** ao cabeçalho inicial da entrada do **XmlStreamer** porque o cabeçalho inicial não tem um elemento pai.

AddStringModifier

Em um **XmlStreamer**, o componente **AddStringModifier** adiciona strings antes e depois de um segmento.

A tabela a seguir descreve as propriedades do componente **AddStringModifier**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
pré	Define a string antes do segmento.
pós	Define a string depois do segmento.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

DoNothingModifier

Em um **XmlStreamer**, esse componente é um espaço reservado. Ele não modifica o segmento ao qual é aplicado.

WellFormedModifier

Em um **XmlStreamer**, o componente **WellFormedModifier** garante que um segmento seja o XML bem formado. Ele pode adicionar marcas de abertura, fechamento ou raiz, conforme exigido para essa finalidade.

A seguinte tabela descreve as propriedades do componente **WellFormedModifier**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.

Propriedade	Descrição
new_root_element	Define o elemento raiz. O elemento raiz deve ser um ancestral do segmento. Se você não atribuir esta propriedade, o modificador não adicionará um elemento raiz.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.

Considere a seguinte entrada XML:

```
<stream>
  <headerline1>...</headerline1>
  <substreams>
    <substream>
      <subheaderline1>...</subheaderline1>
      <segments>
        <segment1>...</segment1>
        <segment1>...</segment1>
        <segment1>...</segment1>
      </segments>
      <subfooterline1>...</subfooterline1>
    </substream>
    <substream>...</substream>
    <substream>...</substream>
  </substreams>
  <footerline1>...</footerline1>
</stream>
```

Suponha que você configure um **XmlStreamer** que retorna o seguinte cabeçalho, que não é o XML bem formado:

```
<substream>
  <subheaderline1>...</subheaderline1>
  <segments>
```

Se você aplicar o **WellFormedModifier** a este cabeçalho, o modificador adicionará as marcas de fechamento. O resultado é o seguinte segmento bem formado:

```
<substream>
  <subheaderline1>...</subheaderline1>
  <segments>
  </segments>
</substream>
```

Agora suponha que você configure o **WellFormedModifier** para adicionar `<stream>` como um elemento raiz. O resultado é:

```
<stream>
  <substreams>
    <substream>
      <subheaderline1>...</subheaderline1>
      <segments>
      </segments>
    </substream>
    <substreams>
  </substreams>
</stream>
```

Observe que o modificador adicionou os elementos `<stream>` e `<substreams>`, preservando a estrutura de esqueleto XML original.

WriteSegment

O componente **WriteSegment** copia um segmento para uma localização de saída especificada. O componente não altera o segmento copiado. O componente **WriteSegment** é uma opção da propriedade **run_component** do componente **XmlSegment**.

A seguinte tabela descreve as propriedades do componente **WriteSegment**:

Propriedade	Descrição
saída	<p>Define a localização de saída. A propriedade saída tem as seguintes opções:</p> <ul style="list-style-type: none">- OutputDataHolder. Grava em um recipiente de dados.- OutputFile. Grava em um arquivo.- OutputPort. Define o nome de um AdditionalOutputPort onde os dados são gravados.- ResultFile. Grava o arquivo de resultados padrão da transformação.- StandardErrorLog. Grava no log do usuário. Para obter mais informações, consulte "Tratamento de Falhas" na página 399. <p>Para obter mais informações sobre essas opções, consulte "Referência do Subcomponente da Ação" na página 331. O padrão é ResultFile.</p>

CAPÍTULO 23

Validadores, Notificações e Tratamento de Falhas

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Validadores, Notificações e Tratamento de Falhas, 398](#)
- [Tratamento de Falhas, 399](#)
- [Validadores, 402](#)
- [Propriedades do Validador Padrão, 403](#)
- [Referência de Componente do Validador, 403](#)
- [Notificações, 418](#)
- [Referência de Componente de Notificação, 419](#)

Visão Geral de Validadores, Notificações e Tratamento de Falhas

Ao criar uma transformação, você deve considerar as seguintes questões:

- O que acontece se os dados de entrada forem inválidos? Por exemplo, uma data pode ter o formato incorreto, uma string pode ser muito longa ou os registros poderão estar fora de sequência.
- O que acontece se os dados estiverem ausentes da entrada? Por exemplo, um endereço pode omitir número da casa.
- O que acontece se a entrada tiver uma estrutura anormal? Por exemplo, os registros poderão estar fora de sequência.

Qualquer um desses problemas pode ocorrer devido a um erro de entrada. Se isso acontecer, poderá ocorrer erros de transformações e falhas.

Os problemas também podem ocorrer em circunstâncias normais. Por exemplo, um protocolo de entrada pode permitir que determinados campos estejam ausentes.

Você pode incorporar recursos de transformação que detectem esses problemas e realizar as ações apropriadas. As seguintes abordagens estão entre as ações possíveis:

- Faça a transformação falhar e não gere nenhuma saída.
- Faça parte da transformação falhar e reverta sua saída, mas permita que a transformação gere saída para outras partes dos dados.

- Continue toda a transformação, mas grave uma mensagem em um log de usuário.
- Continue toda a transformação, mas grave uma mensagem no arquivo de resultados da transformação.

Este capítulo explica o que acontece se uma transformação falhar e como você pode tratar problemas de falha. Em seguida, ele explica como você pode detectar erros de validação de dados que podem causar falhas e como você pode gravar notificações sobre esses problemas na saída.

Tratamento de Falhas

Uma falha é um evento que impede que um componente processe dados da maneira esperada. Uma âncora poderá falhar se ela pesquisar texto que não existe no documento de origem. Um transformador ou ação poderá falhar se a entrada estiver vazia ou tiver um tipo de dados inadequado.

Uma falha pode ser uma ocorrência perfeitamente normal. Por exemplo, um documento de origem pode conter uma data opcional. Um Analisador contém uma âncora de **Conteúdo** que processará a data se ela existir. Se a data não existir em um documento de origem específico, a âncora de **Conteúdo** falhará.

Configurando a transformação de forma correta, você pode controlar o resultado de uma falha. No exemplo acima, você pode configurar o Analisador para ignorar os dados não encontrados e continuar o processamento.

O log de eventos exibirá avisos sobre falhas. Além disso, você pode configurar uma transformação para gravar uma mensagem de falha em um log de usuário.

Usando a Propriedade Opcional para o Tratamento de Falhas

Você pode usar a propriedade `opcional` para controlar o comportamento de uma transformação quando ocorre uma falha.

Falha Causa Falha no Pai

Se a propriedade **opcional** de um componente não for selecionada, uma falha do componente fará com que seu pai falhe. Se o pai também for não opcional, seu próprio pai falhará, e assim por diante.

Por exemplo, suponha que um **Analisador** contenha um **Grupo** e o **Grupo** contenha um **Marcador**. Todos os componentes são não opcionais. Se o **Marcador** não existir no documento de origem, o **Marcador** falhará. Isso faz com que o **Grupo** falhe, o que por sua vez fará com que o **Analisador** falhe.

Graficamente, é possível representar esses relacionamentos da seguinte maneira:

```
Parser      //Falhou
Group       //Falhou
Marker      //Falhos
```

Falha Opcional Não Causa Falha no Pai

Se a propriedade **opcional** de um componente for selecionada, uma falha do componente não será transmitida para o pai.

No exemplo acima, suponha que o **Grupo** é opcional. O **Marcador** com falha faz com que o **Grupo** falhe, mas o **Analisador** não falhará.

```
Parser      //Obteve êxito
Group       //Falhou
Marker      //Falhou
```

Reversão

Se um componente falhar, seus efeitos serão revertidos.

Por exemplo, suponha que um **Grupo** contenha três âncoras de **Conteúdo** não opcionais que armazenam valores em recipientes de dados. Se a terceira âncora de **Conteúdo** falhar, o **Grupo** também falhará. O Script reverte os efeitos das duas primeiras âncoras de **Conteúdo**. Os dados já armazenados em recipientes de dados pelas duas primeiras âncoras de **Conteúdo** são removidos.

A reversão somente se aplica aos efeitos principais de uma transformação, como um Analisador que armazena valores em recipientes de dados ou um Serializador que grava em sua saída. A reversão não se aplica a efeitos colaterais. No exemplo acima, se o **Grupo** contiver uma ação **WriteValue** que grava uma linha em um arquivo de saída de texto, essa linha não será excluída.

Definindo a Propriedade Optional

É possível definir a propriedade **optional** de um componente das seguintes maneiras:

- Edite as propriedades avançadas de um componente no Script.
- Clique com o botão direito no componente e depois clique em **Tornar Opcional** ou **Tornar Obrigatório**.

Componentes com uma Propriedade Opcional Faltando

Alguns componentes não têm a propriedade **optional** porque eles nunca falham, independentemente da entrada.

Um exemplo é a ação **Classificar**. Se a ação **Classificar** não encontrar dados para classificar, ela simplesmente não fará nada. Ela não informará uma falha.

Gravando uma Mensagem de Falha no Log do Usuário

É possível configurar um componente para gerar a saída de eventos de falha em um log definido pelo usuário. Por exemplo, se uma âncora não conseguir localizar texto no documento de origem, ela poderá gravar uma mensagem no log do usuário. Isso pode ocorrer mesmo que a âncora esteja definida como opcional, de forma que a falha não finalize o processamento da transformação.

O log do usuário pode conter informações como:

- Nível da falha: Informações, Aviso ou Erro
- Nome do componente que falhou
- Descrição da falha
- Localização do componente com falha no Script
- Informações adicionais sobre o status da transformação, como os valores de recipientes de dados.

Configurando a Saída do Log do Usuário

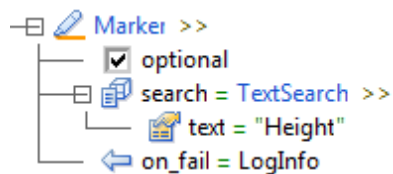
Para definir a saída de log do usuário, atribua a propriedade `on_fail` dos componentes de transformação apropriados. Os componentes a seguir têm uma propriedade `on_fail`:

- Analisadores e âncoras
- Serializadores e âncoras de serialização
- Mapeadores e âncoras mapeadoras

A propriedade `on_fail` pode ter os seguintes valores:

- `LogError`. Grava uma mensagem de erro que contém a variável de sistema `VarLastFailure` do log do usuário.
- `LogWarning`. Igual a `LogError`, mas exibe a mensagem como um aviso, em vez de um erro.
- `LogInfo`. Igual a `LogError`, mas exibe a mensagem como uma informação, em vez de um erro.
- `CustomLog`. Executa um serializador que grava uma mensagem personalizada no log do usuário ou em outra localização. Para obter mais informações, consulte [“CustomLog” na página 307](#).
- `NotifyFailure`. Ativa uma notificação.

O exemplo a seguir ilustra uma âncora de `Marcador` com uma configuração `LogInfo`:



Se o `Marcador` não existir no documento de origem, o sistema gravará a seguinte entrada no log do usuário:

```
*** INFO *** : Marker, [MyParser[11].Marker], Can't find Marker<optional>('Height').
```

Exibindo o Log do Usuário

O log do usuário é um arquivo de texto ASCII. Nas plataformas Windows, a localização padrão do log do usuário é:

```
c:\Informatica\DataTransformation\UserLogs
```

Nas plataformas UNIX, a localização padrão é:

```
<INSTALL_DIR>/UserLogs
```

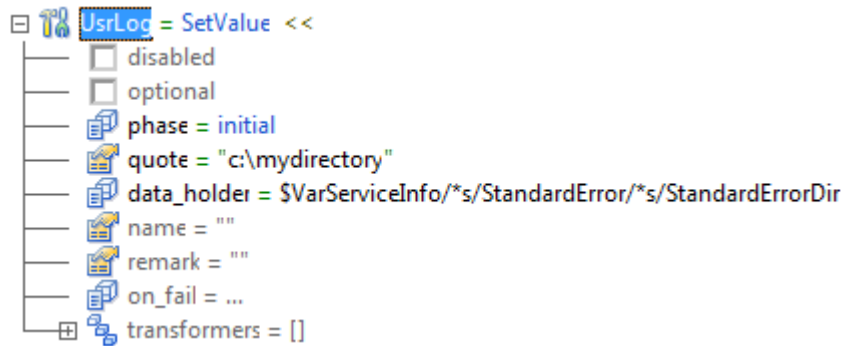
Por padrão, cada execução de uma transformação gera um log de usuário com um nome exclusivo:

```
<service_name>+<unique_string>.log
```

Uma transformação pode definir a localização do user-log no tempo de execução usando as ações **SetValue** para atribuir as variáveis do sistema a seguir. Defina a propriedade de fase de **SetValue** para . Certifique-se de que **SetValue** seja executado antes que qualquer componente possa gravar no log do usuário.

Variável	Descrição
VarServiceInfo > StandardError > <i>StandardErrorDir</i>	Caminho de diretório do log do usuário.
VarServiceInfo > StandardError > <i>StandardErrorName</i>	O nome do arquivo de log do usuário.

No exemplo a seguir, uma ação **SetValue** define o diretório do user-log para `c:\meudiretório`.



Validadores

Um componente validador confirma se a sua entrada está em conformidade com uma condição. Você pode usar validadores para verificar a entrada dos tamanhos máximo ou mínimo de string ou dos valores numéricos, a conformidade com expressões ou várias outras condições. Você pode aplicar vários validadores à mesma entrada.

Se a entrada não estiver em conformidade com a condição, o validador disparará uma notificação. Um componente **NotificationHandler** pode processar a notificação. Por exemplo, se você usar validadores em um Analisador, um **NotificationHandler** poderá inserir uma mensagem de aviso na saída do Analisador. Para obter mais informações, consulte [“Notificações” na página 418](#).

Você pode inserir validadores em localizações, como a propriedade **validadores** de uma âncora **Conteúdo** ou ação **Mapear**. O validadores permitem que você avise se a entrada é inválida, sem necessariamente falhar a **Conteúdo** ou a **Mapear**.

Além disso, para os validadores descritos neste capítulo, você pode validar dados em um conjunto de regras definidas pelo usuário e gerar um relatório de validação XML. Para obter mais informações, consulte [“ValidateValue” na página 327](#).

Propriedades do Validador Padrão

A seguinte tabela descreve as propriedades padrão de validadores:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

Referência de Componente do Validador

Os componentes do validador testam os dados de entrada para conformidade com as regras definidas.

AlternativeValidators

O validador **AlternativeValidators** contém um conjunto de validadores aninhados que se aplicam à entrada. Use um **AlternativeValidators** para aplicar a lógica OR a um conjunto de condições de validação. Os dados serão válidos se atenderem a qualquer uma das condições.

A seguinte tabela descreve as propriedades do validador **AlternativeValidators**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negation	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notify	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
selector	Determina os critérios para a seleção de um validador entre os validadores aninhados abaixo do componente AlternativeValidators . É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- ScriptOrder. O Analisador testa os validadores aninhados na sequência definida no Script. Ele aceita o primeiro validador bem-sucedido. Se todos os validadores falharem, a entrada será inválida.- NameSwitch. O Analisador busca o validador aninhado cuja propriedade nome está especificada no recipiente de dados definido em option_name. Ele ignora os outros validadores. Se o validador nomeado falhar, a entrada será inválida. O padrão é ScriptOrder.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

EDIFACTValidation

O validador **EDIFACTValidation** testa se uma string de origem é uma mensagem EDIFACT válida.

A seguinte tabela descreve as propriedades do validador **EDIFACTValidation**:

Propriedade	Descrição
disabled	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
enabled	Determina a configuração para param1 .
name	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
optional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
param1	Determina se a entrada é opcional. param1 se chama is_optional e tem apenas uma propriedade, enabled . enabled tem as seguintes opções: <ul style="list-style-type: none">- Selected. Os dados de entrada são opcionais.- Cleared. Os dados de entrada são obrigatórios.
param2	Define um tipo de dados EDI. param2 se chama input_type e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados.
param3	Define um intervalo de inteiros. param3 se chama minmax_limits e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados que especifica dois inteiros separados por um hífen.
param4	Define uma lista de valores. param4 se chama enumerations e tem apenas uma propriedade, value . value é uma string de chave fixa ou um recipiente de dados que especifica uma lista de strings ou inteiros separados por vírgula.
remark	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
value	Define um valor para param1 , param2 ou param3

Nota: Esse componente foi preterido. O editor do IntelliScript o exibe apenas para Scripts herdados. Não o use em novos Scripts. Em vez disso, use outros componentes de validador.

Enumeração

O validador **Enumeração** testa se um valor é um membro de um conjunto de valores.

A tabela a seguir descreve as propriedades do validador **Enumeração**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
enumerações	Define uma lista de valores.
ignore_case	Determina se a comparação faz distinção entre maiúsculas e minúsculas. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A comparação não faz distinção entre maiúsculas e minúsculas.- Desmarcado. A comparação faz distinção entre maiúsculas e minúsculas. O padrão é desmarcado.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

LengthEquals

O validador **LengthEquals** testa se o tamanho de uma string igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **LengthEquals**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. A entrada vazia é válida. - Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
tamanho	Define o tamanho da string.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida. - Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

MaxLength

O validador **MaxLength** testa se o tamanho de uma string é menor ou igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **MaxLength**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
tamanho	Define o tamanho máximo da string.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

MaxNumber

O validador **MaxNumber** testa se um número é menor ou igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **MaxNumber**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.
valor	Define o valor máximo do número.

MinLength

O validador **MinLength** testa se o tamanho de uma string é maior ou igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **MinLength**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
tamanho	Define o tamanho mínimo da string.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

MinNumber

O validador **MinNumber** testa se um número é maior ou igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **MinNumber**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.
valor	Define o valor mínimo do número.

NumberEquals

O validador **NumberEquals** testa se um número é igual a um valor especificado.

A tabela a seguir descreve as propriedades do validador **NumberEquals**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.
valor	Define o valor do número.

ValidateByExpression

O validador **ValidateByExpression** avalia uma expressão de JavaScript. Se a expressão for falsa, o validador considerará que a entrada é inválida.

A seguinte tabela descreve as propriedades do validador **ValidateByExpression**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. A entrada vazia é válida. - Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. O Script ignora o componente. - Limpo. O Script aplica o componente. O padrão é Cleared.
expressão	Define uma expressão de JavaScript. Use \$0 para a entrada do validador. Use um sinal de dólar (\$) mais um inteiro para retentores de dados adicionais definidos em parâmetros , começando com \$1. Por exemplo, a seguinte expressão verifica se a entrada tem o valor <code>Ron Lehrer</code> : <pre>\$0 == "Ron Lehrer"</pre>
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida. - Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none"> - Selecionado. Uma falha de componente não causa uma falha no componente pai. - Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
params	Define uma lista dos retentores de dados que contém parâmetros para uso na expressão.
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

ValidateByPattern

O validador **ValidateByPattern** testa se uma string corresponde a uma expressão regular. Para obter mais informações, consulte [“RegularExpression” na página 281](#).

A seguinte tabela descreve as propriedades do validador **ValidateByPattern**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
expressão	Define uma expressão regular.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

ValidateByTransformer

O validador **ValidateByTransformer** aplica uma lista de um ou mais transformadores à entrada. Se a lista de transformadores falhar, o validador considerará a entrada como inválida.

A seguinte tabela descreve as propriedades do validador **ValidateByTransformer**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
run_transformers	Define uma lista de transformadores.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

ValidateByType

O validador **ValidateByType** testa se sua entrada está em conformidade com um tipo de dados especificado.

A seguinte tabela descreve as propriedades do validador **ValidateByType**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.
val_type	Define um tipo de dados. Selecione um tipo de padrão ou um tipo definido nos esquemas do projeto.

ValidateDate

O validador **ValidateDate** testa se uma data está em conformidade com um formato de data ICU especificado, por exemplo, `aaaa-MM-dd`. Para obter mais informações, consulte [“DateFormatICU” na página 266](#).

A seguinte tabela descreve as propriedades do validador **ValidateDate**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
format_string	Define um formato de data ICU.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte “Notificações” na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte “Tratamento de Falhas” na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

ValidatorPipeline

O validador **ValidatorPipeline** aplica uma lista de validadores aos dados. Se qualquer um dos validadores relatar invalidez ou se um validador estiver marcado como **opcional** e falhar, o **ValidatorPipeline** disparará uma notificação.

Use um **ValidatorPipeline** para aplicar a lógica AND a um conjunto de condições de validação. Os dados são válidos quando satisfazem todas as condições.

A seguinte tabela descreve as propriedades do validador **ValidatorPipeline**:

Propriedade	Descrição
allow_empty_value	Determina se uma entrada vazia é aceita como válida. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. A entrada vazia é válida.- Desmarcado. A entrada vazia não é válida. O padrão é desmarcado.
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
negação	Determina se a condição de validação é negada. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Marcado. Se a condição for verdadeira, a entrada não será válida e, se a condição for falsa, a entrada será válida.- Desmarcado. Se a condição for verdadeira, a entrada será válida e, se a condição for falsa, a entrada não será válida. O padrão é desmarcado.
notificar	Define o nome de uma notificação. Se a entrada não estiver em conformidade com a condição de validação, o validador disparará a notificação. Para obter mais informações, consulte "Notificações" na página 418 . O padrão é desmarcado.
opcional	Determina se uma falha de componente causa uma falha no componente pai. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. Uma falha de componente não causa uma falha no componente pai.- Limpo. Uma falha de componente causa uma falha no componente pai. O padrão é Desmarcado. Para obter mais informações sobre falhas de componentes, consulte "Tratamento de Falhas" na página 399 .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
transformadores	Define uma lista de transformadores que se aplicam à entrada. A condição de validação é aplicada ao resultado dos transformadores. Os transformadores têm somente um efeito temporário sobre os dados para fins de validação. A entrada não é alterada permanentemente.

Notificações

Uma notificação é um sinal que ocorreu um problema na transformação. Quando o problema ocorrer, uma transformação ativará a notificação. Você pode configurar manipuladores para processar as notificações.

Os exemplos a seguir ilustram algumas maneiras de usar notificações:

- Um validador pode ativar uma notificação. Um componente **NotificationHandler** pode gravar uma mensagem de aviso de validação no arquivo de resultados da transformação ou em um log.
- Uma âncora **StructureDefinition** pode definir um conjunto de componentes de **NotificationHandler** para processar divergências entre os registros de entrada e a estrutura de entrada exigida. Se uma

incompatibilidade ocorrer, o **NotificationHandler** apropriado gravará uma mensagem no arquivo de resultado ou em um log.

- Um ação **Notificar** para acionar uma notificação em qualquer localização de uma transformação. Um **NotificationHandler** pode gravar uma mensagem no arquivo de resultado ou em um log.

A tabela a seguir descreve os tipos de notificações:

Notificação	Descrição
MandatoryStructureMissing	Um registro obrigatório não é exibido na entrada.
MismatchIDs	Os IDs do registro e do subelemento correspondem parcialmente. Por exemplo, há dois identificadores de registro, e apenas um deles corresponde.
StructureBelowMinOccurs	Há menos registros correspondentes do subelemento do que o definido em minOccurs .
StructureExceedsMaxOccurs	Há mais registros correspondentes do subelemento do que o definido em maxOccurs .
StructureOutOfSequence	Os registros correspondem aos subelementos, mas não na sequência necessária. Por exemplo, os subelementos definem uma sequência ABC, mas a entrada contém ACB.
UnexpectedRecord	Os registros correspondem aos subelementos, mas não na hierarquia necessária. Por exemplo, o subelemento define uma sequência ABC, e D está definido em outra localização. A entrada contém ABD.
UnrecognizedRecord	Nenhum subelemento corresponde a um dos identificadores de registro.
XsdValidationError	A entrada não corresponde aos requisitos do esquema.

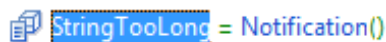
Referência de Componente de Notificação

Os componentes de notificação executam ações quando um componente falha.

Notificação

O componente define o nome de uma notificação. Configure o componente no nível global do Script.

A seguinte figura mostra uma notificação denominada `StringTooLong`:

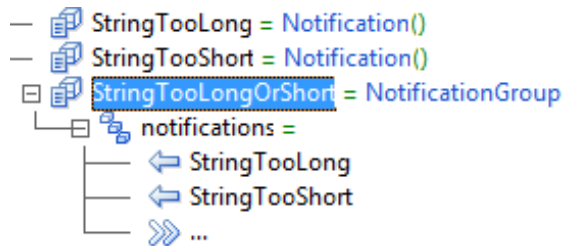


```
StringTooLong = Notification()
```

NotificationGroup

O componente define um único nome que faz referência a um conjunto de nomes de notificação. Configure o componente no nível global do Script.

O seguinte exemplo mostra um grupo denominado `StringTooLongOrShort`:



É possível configurar um **NotificationHandler** para processar `StringTooLongOrShort`. Se uma transformação disparar uma notificação `StringTooLong` ou `StringTooShort`, o manipulador processará essa notificação.

A seguinte tabela descreve as propriedades do componente **NotificationGroup**:

Propriedade	Descrição
notifications	Define uma lista de notificações.

NotificationHandler

O componente **NotificationHandler** define uma lista de ações a serem executadas por uma determinada notificação.

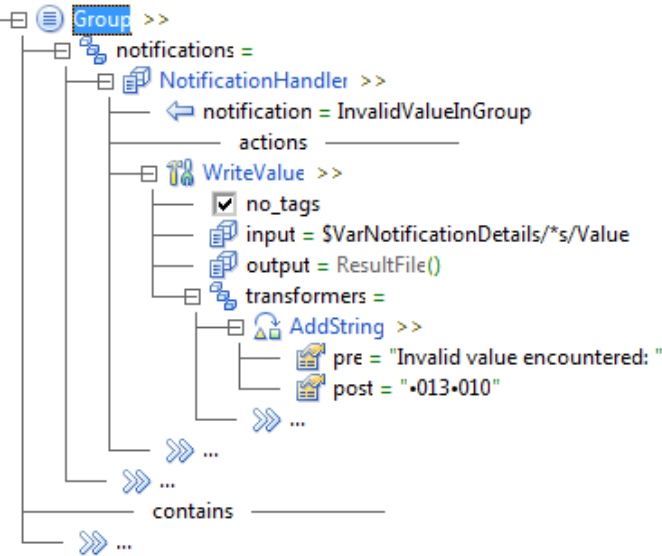
Insira o componente **NotificationHandler** em localizações como a propriedade **notificações** de **Grupo** ou **RepeatingGroup**. Dentro do componente, você pode inserir uma ação **WriteValue** que armazena uma mensagem em um recipiente de dados.

A variável `VarNotificationDetails` armazena informações sobre a notificação acionada mais recentemente. Um **NotificationHandler** grava as informações armazenadas em `VarNotificationDetails` na saída. Para obter mais informações sobre `VarNotificationDetails`, consulte ["Variáveis de Sistema" na página 197](#).

A tabela a seguir descreve as propriedades do componente **NotificationHandler**:

Propriedade	Descrição
desativado	Determina se o Script ignora o componente e todos os componentes filho. Use essa propriedade para testar, depurar e modificar um Script. É possível escolher uma das seguintes opções: <ul style="list-style-type: none">- Selecionado. O Script ignora o componente.- Limpo. O Script aplica o componente. O padrão é Cleared.
nome	Um rótulo descritivo para o componente. Esse rótulo aparece no arquivo de log e na exibição Eventos . Use a propriedade name para identificar o componente que causou o evento.
notificação	Define o nome de uma notificação para o NotificationHandler processar. Selecione uma notificação predefinida ou um nome de notificação definido em um componente de Notificação ou NotificationGroup . Para configurar um manipulador que processe qualquer notificação, selecione anyNotification .
comentário	Um comentário definido pelo usuário que descreve a finalidade ou a ação do componente.
origem	Define um recipiente de dados que você pode usar para a entrada do NotificationHandler .
destino	Define um recipiente de dados para a saída do NotificationHandler .

A figura a seguir mostra uma âncora de **Grupo** que está configurada com um **NotificationHandler**:



Se um componente no **Grupo** acionar uma notificação **InvalidValueInGroup**, o manipulador a processará. O manipulador grava a variável *VarNotificationDetails/Value* junto com uma string de texto no arquivo de resultado da transformação.

NotifyFailure

NotifyFailure é um valor possível da propriedade **on_fail** de âncora e outros componentes. Se o componente falhar, **NotifyFailure** acionará uma notificação.

A tabela a seguir descreve as propriedades do componente **NotifyFailure**:

Propriedade	Descrição
notificar	Define a notificação a ser acionada. Selecione uma notificação predefinida ou um nome de notificação definido em um componente de Notificação ou NotificationGroup .
valor	Define o valor da variável <i>VarNotificationDetails/Value</i> variável. Um NotificationHandler pode incluir o valor em sua saída.

CAPÍTULO 24

Regras de Validação

Este capítulo inclui os seguintes tópicos:

- [Visão Geral das Regras de Validação, 422](#)
- [Referência do Elemento Regras de Validação, 423](#)
- [Editar as Regras de Validação em um Editor Externo, 429](#)
- [Criar um Objeto Regras de Validação, 429](#)
- [Importar um Serviço do Data Transformation com Regras de Validação, 430](#)

Visão Geral das Regras de Validação

Uma transformação do Processador de Dados usa as Regras de Validação para verificar os dados de entrada ou de saída da transformação. Você pode usar o objeto Regras de Validação para validar dados XML de acordo com um conjunto de regras definidas pelo usuário. Se os dados violarem as regras, a ação gerará um relatório de validação de XML. Ao criar um objeto Regras de Validação, você pode fornecer um arquivo de amostra para testá-lo.

As Regras de Validação verificam os elementos de entrada ou de saída de uma transformação do Processador de Dados. Use o editor de Regras de Validação para criar, definir, editar e gerenciar regras.

Depois de criar um objeto Regras de Validação, adicione os elementos Regras de Validação. Defina cada elemento no editor. O editor adiciona os elementos à hierarquia de Regras de Validação.

O nível raiz da hierarquia contém uma descrição do objeto Regras de Validação e uma referência ao arquivo XML de amostra que você pode usar para depurar o objeto Regras de Validação. A hierarquia de Regras de Validação contém as pastas Pesquisas e Regras. Um elemento Pesquisa define uma tabela de pesquisa que contém códigos e conversões. Você pode adicionar um ou mais elementos Pesquisa à pasta Pesquisa.

O elemento Regra define uma regra de validação. Se a Regra for avaliada como falsa, o objeto Regras de Validação informará um erro. Você pode adicionar um ou mais elementos Regra à pasta Regras.

Você pode aninhar os seguintes elementos em um elemento Regra:

Elemento Variável

Um elemento Variável define uma variável com um tipo de dados simples. O elemento contém uma expressão XPath avaliada como o valor da Variável.

Elemento Declaração

O elemento Declaração define a lógica de uma regra. O elemento contém uma expressão XPath. Se a expressão for falsa, a Regra informará um erro de validação.

Elemento Lista

O elemento Lista define uma variável complexa que contém uma lista de valores.

Elemento Rastreamento

O elemento Rastreamento adiciona o valor de uma expressão XPath ao arquivo de eventos.

Depois de criar um elemento, defina os atributos do elemento. Os atributos definem a lógica do elemento.

Você pode copiar e colar os elementos no editor de Regras de Validação. Você também pode editar o objeto Regras de Validação em um editor externo e adicionar, editar ou excluir elementos no XML.

Você pode importar um serviço do Data Transformation com qualquer número de arquivos VRL. Você pode copiar todo ou parte de um arquivo VRL no editor de Regras de Validação. Você pode abrir o arquivo VRL em um editor externo, copiar os elementos e colá-los na hierarquia do editor de Regras de Validação.

Você pode chamar o objeto Regras de Validação de um componente Script da transformação do Processador de Dados com a ação **ValidateValue**.

Referência do Elemento Regras de Validação

O nível superior de uma hierarquia de regra de validação contém um elemento Regra ou Pesquisa. Em um elemento Regra, você pode aninhar os elementos Declaração, Lista, Rastreamento e Variável.

Atributos do Elemento Declaração

O elemento Declaração define a lógica de uma regra. O elemento contém uma expressão XPath. Se a expressão for falsa, a regra informará um erro de validação.

Uma Regra deve conter pelo menos um elemento Declaração.

A seguinte tabela descreve as propriedades do elemento Declaração:

Propriedade	Descrição
Dados Adicionais	Opcional. Uma expressão XPath que pode ser avaliada e inserida no relatório de erro.
Código	Opcional. Um atributo de cadeia que identifica o elemento Declaração. Se nenhum Código for especificado, o Código será obtido da Regra pai.
Descrição	Opcional. Uma descrição do elemento Declaração. Se nenhuma descrição for especificada, a descrição será obtida da Regra pai.
Localização	Opcional. Um valor de cadeia que pode ser anexado à expressão XPath equivalente ao nó selecionado pela Regra pai. A expressão é inserida no relatório de erros.
Descrição da Regra	Uma descrição de somente leitura da Regra na qual o elemento está aninhado.
XPath	Obrigatório. Uma expressão Booleano do XPath que expressa a lógica da regra.

Atributos do Elemento Lista

O elemento Lista define uma variável complexa que contém uma lista de valores.

A seguinte tabela descreve as propriedades do elemento Lista:

Propriedade	Descrição
Anexar	Um atributo Boolean que determina o que acontecerá se a lista tiver o mesmo nome de uma lista existente. Se ativado, os novos valores serão adicionados à lista existente. Se desativado, a lista existente será excluída e uma nova lista será criada. O valor padrão é ativado.
Função	Opcional. Uma expressão XPath que é avaliada em relação a cada um dos nós selecionados. O valor da expressão é adicionado à lista.
Nome	Obrigatório. O nome da lista. Use esse nome para fazer referência à Lista em expressões XPath de elementos aninhados na Regra pai.
Descrição da Regra	Uma descrição de somente leitura da Regra na qual o elemento está aninhado.
Selecionar	Obrigatório. Uma expressão XPath que seleciona nós para calcular os itens na lista.

Atributos do Elemento Pesquisa

O elemento Pesquisa define uma tabela de pesquisa que contém códigos e conversões.

A seguinte tabela descreve as propriedades do elemento Pesquisa:

Propriedade	Descrição
Arquivo	Obrigatório. O arquivo que contém a tabela de pesquisa.
Nome	Obrigatório. O nome da tabela de pesquisa.

Arquivo de Pesquisa

O seguinte exemplo mostra um arquivo de Pesquisa de amostra:

```
<LookupTable xmlns="http://www.Itemfield.com/Engine/V4/lookupTable" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Entry key="a" value="1"/>
  <Entry key="b" value="2"/>
  <Entry key="c" value="3"/>
</LookupTable>
```

O formato da tabela de pesquisa é idêntico a um `XMLLookupTable` usado no componente `LookupTransformer`. Para obter mais informações, consulte [“LookupTransformer” na página 278](#).

Você pode usar a função `dt:lookup()` em uma expressão XPath para procurar um valor na tabela de pesquisa.

Atributos do Elemento Regra

O elemento Regra define uma regra de validação. Se a Regra for avaliada como falsa, a regra de validação relatará um erro.

A seguinte tabela descreve as propriedades do elemento Regra:

Propriedade	Descrição
Código	Obrigatório. Um identificador da Regra. Deve ser uma única palavra.
Descrição	Obrigatório. Uma descrição da Regra em texto livre, de qualquer tamanho.
Ativado	Determina se a Regra está ativada ou desativada. O valor padrão é ativado.
Executar Todas as Declarações	Determina se as Declarações que estão aninhadas na Regra serão executadas. Por padrão a propriedade está ativada, portanto, a Regra processa todas as Declarações. Se estiver desativada, a regra interromperá o processamento das Declarações depois que uma das Declarações falhar.
Selecionar	Obrigatório. Uma expressão XPath que seleciona o nó ou o conjunto de nós ao qual a Regra se aplica.

Atributos do Elemento Rastreamento

O elemento Rastreamento adiciona o valor de uma expressão XPath ao relatório de erros. O elemento contém a expressão XPath.

A seguinte tabela descreve as propriedades do elemento Rastreamento:

Propriedade	Descrição
Descrição	Obrigatório. Uma descrição do elemento Rastreamento.
Ativado	Determina se o Rastreamento está ativado ou desativado. O valor padrão é ativado.
Descrição da Regra	Uma descrição de somente leitura da Regra na qual o elemento está aninhado.
XPath	Obrigatório. Uma expressão XPath que pode ser avaliada e inserida no relatório de erro.

Atributos do Elemento Variável

Um elemento Variável define uma variável com um tipo de dados simples. O elemento contém uma expressão XPath. O valor da Variável é o valor da expressão.

A seguinte tabela descreve as propriedades do elemento Variável:

Propriedade	Descrição
Nome	Obrigatório. O nome da Variável. Use esse nome para fazer referência à Variável nas expressões XPath de elementos aninhados na Regra pai.
Descrição da Regra	Uma descrição de somente leitura da Regra na qual o elemento está aninhado.
XPath	Obrigatório. Uma expressão XPath que calcula ou se refere aos nós aos quais a Variável se aplica.

Editor de XPath

Alguns elementos contêm expressões XPath. Para criar expressões, digite uma expressão XPath no editor de XPath.

Extensões de XPath

Os elementos Regra, Declaração, Lista, Variável e Rastreamento podem conter expressões XPath. As funções a seguir podem fazer parte da expressão XPath.

O objeto Regras de Validação define as seguintes funções como extensões do XPath 1.0. As extensões pertencem ao espaço de nome **dt**, definido como <http://validations.informatica.com>.

Função	Descrição
<code>dt:all-equal(param1[, param2, ...])</code>	Retornará verdadeiro se todos os itens na lista forem iguais. Os parâmetros podem ser listas ou valores simples.
<code>dt:check-uniqueness(value1[, value2, ...])</code>	Retornará verdadeiro se todos os valores na lista de parâmetros forem exclusivos. Cada valor pode ser: <ul style="list-style-type: none">- Um valor simples- Uma variável que contém uma lista de cadeias- Uma variável que contém uma lista de nós onde cada nó é um valor simples.
<code>dt:date-add(startDate, format, numberOfDays)</code>	Retorna a data. Se numberOfDays for do tipo flutuante , a função arredondará para baixo o valor do próximo número inteiro mais baixo.
<code>dt:date-diff(startDate, format, endDate, format)</code>	Retorna o número de dias entre duas datas.

Função	Descrição
<code>dt:date-format (date, inputFormat, outputFormat)</code>	Converte o formato de date de inputFormat em outputFormat .
<code>dt:date-valid (string-to-match, format)</code>	Retornará verdadeiro se a cadeia corresponder ao formato de data especificado.
<code>dt:deep-equal (element1, element2)</code>	Retornará verdadeiro se ambas as seguintes opções forem verdadeiras: <ul style="list-style-type: none"> - element1 e element2 têm os mesmos atributos, com os mesmos valores. - element1 e element2 têm os mesmos elementos filho, na mesma ordem, e os elementos filho são totalmente iguais.
<code>dt:empty (xpath)</code>	Retornará verdadeiro se um XPath não contiver elementos filho.
<code>dt:exist (xpath)</code>	Retornará verdadeiro se um XPath existir. Equivalente a count(xpath) > 0 .
<code>dt:is-sorted-lex (ascending, param1[, param2, ...])</code>	Retornará verdadeiro se os itens da lista forem classificados de maneira lexicográfica. O primeiro parâmetro é um Boolean que define a direção de classificação: verdadeira para crescente ou falsa para decrescente.
<code>dt:last-day-of-month (date, format)</code>	Retorna o último dia do mês selecionado.
<code>dt:list-items (list, separator)</code>	Retorna uma cadeia de todos os itens na lista, separados por separador . O separador padrão é uma vírgula.
<code>dt:lookup (string, lookupName, default)</code>	Pesquisa uma cadeia em uma tabela de pesquisa. Especifique a cadeia a ser pesquisada, o nome da tabela de pesquisa e um valor padrão a ser retornado se a cadeia não existir na tabela.
<code>dt:min-lex (value1[, value2, ...])</code>	Classifica a lista de parâmetros de entrada de maneira lexicográfica e retorna o primeiro item na lista classificada. Cada valor pode ser: <ul style="list-style-type: none"> - Um valor simples - Uma variável que contém uma lista de cadeias - Uma variável que contém uma lista de nós onde cada nó é um valor simples.
<code>dt:next-sequence ()</code>	Retorna um conjunto de nós que contém o elemento atual e todos os seguintes irmãos, até outro elemento do mesmo nome.
<code>dt:regex-match (string-to-match, regex)</code>	Retornará verdadeiro se uma cadeia corresponder a uma expressão regular.
<code>dt:regex-replace (inputString, patternRegex, replacementString)</code>	Retorna inputString com todas as instâncias de patternRegex substituídas por replacementString . Se nada corresponder a patternRegex , o valor inputString será retornado sem alterações.

Função	Descrição
<code>dt:string-replace (string1, string2, string3)</code>	Substitui todas as instâncias da string2 na string1 pela string3 .
<code>dt:unadjusted-calculation-period-dates (startDate, endDate, timeUnit, timeUnitMultiplier, lookupDate, dateFormat)</code>	Retornará verdadeiro se lookupDate estiver no primeiro período de tempo no intervalo especificado, onde o período de tempo é timeUnitMultiplier vezes timeUnit e o intervalo é entre startDate e endDate . <ul style="list-style-type: none"> - timeUnitMultiplier é um número inteiro. - timeUnit é uma das seguintes cadeias: <ul style="list-style-type: none"> - Dia ou D - Semana ou S - Mês ou M - Ano ou A - lookupDate é uma data no intervalo entre startDate e endDate. - dateFormat define os formatos de lookupDate, startDate e endDate

Para obter mais informações sobre o formato da data, consulte [“DateFormatICU” na página 266](#).

Exemplo: usando dt:next-sequence()

Você pode usar a função `dt:next-sequence()` para acessar dados lógicos hierárquicos que não estão aninhados no elemento atual.

Considere a seguinte entrada:

```
<Root>
  <A/>
  <B/>
  <C/>
</Root>
```

O XPath `/Root/A/dt:next-sequence()` retorna o seguinte conjunto de nós:

```
<A/>
<B/>
<C/>
```

No seguinte exemplo, cada elemento `id` está associado a um conjunto de elementos irmãos denominados `name`, `quantity` e `price`:

```
<items>
  <id>100</id>
  <name>Plate</name>
  <quantity>4</quantity>
  <price>10</price>
  <id>101</id>
  <name>Toaster</name>
  <quantity>6</quantity>
  <price>10</price>
  <id>102</id>
  <name>Knife</name>
  <quantity>10</quantity>
  <price>5</price>
</items>
```

Os elementos `name`, `quantity` e `price` estão logicamente aninhados em `id`, embora não estejam aninhados fisicamente.

A seguinte regra exige que `price*quantity` de cada `id` seja menor ou igual a 50:

```
<rule code="sum1" select="/items/id" description="For each id, check that the total
price (price*quantity) does not exceed 50">
  <variable name="current">dt:next-sequence()</variable>
  <variable name="total">${current[3]}*${current[4]}</variable>
  <assert additionalData="${total}"><![CDATA[ ${total} <= 50 ]]></assert>
</rule>
```

Para o primeiro elemento `id`, a variável `$current` é um conjunto de nós com o seguinte valor:

```
<id>100</id>
<name>Plate</name>
<quantity>4</quantity>
<price>10</price>
```

A expressão `$current[3]*$current[4]` avalia como $4 \times 10 = 40$. A regra confirma que $40 < 50$.

Para os elementos `id` subsequentes, a regra avalia a expressão de uma forma semelhante.

Editar as Regras de Validação em um Editor Externo

Você pode selecionar para editar as Regras de Validação em um editor externo. O editor exibe a hierarquia de Regras de Validação no XML com todos os elementos aninhados criados por você. Você pode copiar, editar ou excluir elementos no XML.

Nota: Se você selecionar para salvar as alterações ao editar um objeto Regras de Validação em um editor externo, toda a transformação será salva e não somente as alterações nas Regras de Validação.

Criar um Objeto Regras de Validação

Você pode usar o editor de Regras de Validação para criar, exibir e editar um objeto Regras de Validação.

1. Na exibição **Objetos** da transformação do Processador de Dados, crie Regras de Validação.
2. Para abrir as Regras de Validação, clique no objeto Regras de Validação.
3. Para adicionar um elemento, no painel esquerdo do editor, clique com o botão direito do mouse na hierarquia de regra e selecione Novo > <element>, onde <element> representa o tipo de elemento que você deseja adicionar.

Um elemento em branco é exibido.

4. No painel direito, defina os atributos do elemento.
5. Adicione elementos e atribua os atributos a cada elemento, conforme necessário.

Em cada fase, você pode clicar com o botão direito do mouse e realizar as seguintes operações:

- Anexar um elemento irmão
- Adicionar um elemento filho
- Excluir um elemento
- Ativar ou desativar uma regra
- Desfazer ou refazer operações

6. Salve o objeto Regras de Validação.

Importar um Serviço do Data Transformation com Regras de Validação

Você pode importar um serviço do Data Transformation que contém Regras de Validação do repositório do sistema de arquivos da máquina na qual o serviço foi salvo. Importe um arquivo .cmw do serviço de Data Transformation para o repositório do Modelo para criar uma transformação do Processador de Dados. A ferramenta Developer importa a transformação e as regras de validação com o arquivo .cmw.

1. Clique em **Arquivo > Importar**,
A caixa de diálogo **Importar** é exibida.
2. Selecione **InformaticaImportar Serviço do Data Transformation** e clique em **Avançar**.
A página **Importar Serviço de Data Transformation** será exibida.
3. Navegue até o arquivo .cmw de serviço que você deseja importar.
A ferramenta Developer nomeia a transformação de acordo com o nome de arquivo do serviço. É possível alterar o nome.
4. Procure uma localização no Repositório onde você deseja salvar a transformação e clique em **Concluir**.
A ferramenta Developer importa a transformação e as regras de validação com o arquivo .cmw.
5. Para editar o objeto Regras de Validação, clique duas vezes no objeto Regras de Validação na exibição **Object Explorer**.
O editor de Regras de Validação é exibido e mostra a hierarquia do objeto Regras de Validação.

CAPÍTULO 25

Componentes de Script Personalizados

Este capítulo inclui os seguintes tópicos:

- [Visão Geral de Componentes de Script Personalizados, 431](#)
- [Exemplo de Componente Personalizado, 431](#)
- [Propriedades de Componentes Personalizados, 432](#)
- [Desenvolvendo um Componente Personalizado, 432](#)
- [Configurando um Componente Personalizado, 434](#)

Visão Geral de Componentes de Script Personalizados

Ao criar e configurar um Script de transformação de Processador de Dados, você pode usar vários componentes internos. Também é possível programar componentes personalizados, como processadores de documentos ou transformadores, e inseri-los em um Script. Quando você exporta a transformação de Processador de Dados como um serviço de Transformação de Dados, esse serviço executa os componentes personalizados.

Você implementa os componentes personalizados em Java. Para obter mais informações sobre as interfaces que você deve implementar, consulte a *Referência da Interface Java de Componente Externo*.

Exemplo de Componente Personalizado

Suponha que você precise analisar um formato de dados binários patenteado. Em vez de analisar os dados binários diretamente, você prefere convertê-los em uma representação de texto que é mais fácil de analisar.

Para fazer isso, é possível programar um processador de documentos personalizado, que pode se chamar `MyBinaryToText`. O processador pode ter propriedades como as seguintes:

Propriedade	Descrição
KeepLineBreaks	Uma propriedade Booleana. Quando ela é true, o processador preserva os caracteres de quebra de linha nos dados binários.
MaxLineLength	Uma propriedade de inteiro. Especifica o comprimento máximo das linhas de texto para saída.
Ignore	Uma propriedade de string. Instrui o processador a ignorar campos de dados que começam com a string especificada.

Depois de desenvolver o processador, você pode instalá-lo e usá-lo em Scripts.

Propriedades de Componentes Personalizados

As propriedades de um componente personalizado podem ter tipos de dados de inteiro, booleano, string ou lista de string. É possível atribuir um valor de propriedade constante ou o nome de um recipiente de dados que contém o valor.

É possível ocultar algumas das propriedades no editor do IntelliScript. Por exemplo, um componente personalizado pode oferecer suporte a quatro propriedades. Em seu arquivo de configuração TGP, você pode configurá-lo para exibir apenas as duas primeiras propriedades. O Script transmite somente as propriedades exibidas para o componente. O componente pode atribuir seus próprios valores padrão às propriedades ocultas.

O número máximo de propriedades que um componente personalizado pode ter depende do tipo de componente. Para um componente de processador de documentos, o número máximo de propriedades é 4. Para um componente de transformador, o número máximo de propriedades é 10.

Desenvolvendo um Componente Personalizado

1. Crie uma classe que implemente uma ou mais das seguintes interfaces:

Tipo de Componente	Tipo de Entrada	Interface
Processador de documentos	Arquivo	CMXFileProcessor
Processador de documentos	Buffer	CMXByteArrayProcessor
Transformador	String	CMXStringTransformer
Transformador	Buffer	CMXByteArrayTransform

Para obter mais informações sobre essas interfaces, consulte a *Referência da Interface de Componentes Externos*.

2. Compile o projeto em um arquivo JAR.
3. Armazene o JAR no subdiretório `externLibs\user` do diretório de instalação de cada computador no qual você planeja usar o componente.
4. Crie um arquivo de Script que define o nome de exibição do componente e suas propriedades. Armazene o arquivo no subdiretório `autoInclude\user` do diretório de instalação.

Para obter mais informações sobre essa etapa, consulte [“Configurando um Componente Personalizado” na página 434](#).

Dessa forma, você pode usar o componente personalizado em transformações.

Exemplo de Interface Java

Como exemplo, considere um processador de documentos que aceita uma entrada de arquivo. O processador deve implementar a classe `CMXFileProcessor`, que tem o seguinte método:

```
public String process(  
    CMXContext context,  
    String in,  
    String additionalFilesDir,  
    CMXEventReporter reporter)  
    throws Exception
```

O significado dos parâmetros é o seguinte:

Parâmetro	Descrição
context	Parâmetro de entrada. Um objeto que contém as propriedades que o Script transmite para o componente. O método parameters do objeto retorna um vetor que contém os valores de propriedades.
in	Parâmetro de entrada. O caminho completo do arquivo no qual o componente opera.
additionalFilesDir	Parâmetro de saída opcional. O caminho de um diretório temporário no qual o componente grava arquivos. No final do processamento, o Script exclui todo o conteúdo do diretório.
reporter	Parâmetro de entrada. Um objeto que fornece o método <code>report</code> , que o componente pode usar para gravar eventos no log de eventos.

Exemplo de Componentes Java Personalizados

Para obter amostras da implementação dos componentes personalizados, consulte o seguinte subdiretório do diretório de instalação:

```
samples\SDK\ExternalParameters\Java_SDK\Java
```

O diretório contém os seguintes exemplos:

Exemplo	Descrição
FilePP.java	Um processador de documento que aceita a entrada do arquivo.
ByteArrayPP.java	Um processador de documento que aceita a entrada do buffer.

Exemplo	Descrição
StringTT.java	Um transformador que aceita a entrada da string.
ByteArrayTT.java	Um transformador que aceita a entrada do buffer.

Configurando um Componente Personalizado

Depois de desenvolver um componente personalizado, você deve preparar um arquivo de Script que define esse componente. Não é possível preparar o arquivo TGP no editor do IntelliScript. Em vez disso, você deve prepará-lo em um editor de texto.

Depois de instalar o componente e o arquivo TGP, você pode configurar o componente personalizado no editor do IntelliScript.

1. Crie um arquivo de texto e salve-o com uma extensão *.tgp.

Nota: Você pode definir mais de um componente externo em um único arquivo TGP.

2. Para cada propriedade com suporte pelo seu componente externo, adicione linhas como as seguintes ao arquivo TGP:

```
profile <CustomPropertyName> ofPT <DataType>
{
    paramName = "<CustomPropertyName>" ;
}
```

<CustomPropertyName> é o nome de uma propriedade que você deseja exibir no editor do IntelliScript. <DataType> é o tipo de dados da propriedade. Os tipos de dados com suporte são `NamedParamIntT` para uma propriedade de inteiro, `NamedParamBoolT` para uma propriedade booleana, `NamedParamStringT` para uma propriedade de string ou `NamedParamListT` para uma propriedade que consiste em uma lista de strings.

3. Para cada componente externo que você deseja definir, adicione linhas como as seguintes ao arquivo TGP:

```
profile <ExternalComponentName> ofPT <ComponentType>
{
    jclass = "<ClassName>" ;
    param1 = <CustomPropertyName>() ;
    param2 = <CustomPropertyName2>() ;
}
```

<ExternalComponentName> é o nome do componente externo que você deseja exibir no editor do IntelliScript. <ComponentType> é um dos seguintes valores:

Para	ComponentType
Um processador de documentos Java com 0 a 4 propriedades	ExternalJavaProcessorNoParamsT ExternalJavaProcessor1ParamsT ExternalJavaProcessor2ParamsT ...
Um transformador Java com 0 a 10 propriedades	ExternalJavaTransformerNoParamsT ExternalJavaTransformer1ParamsT ExternalJavaTransformer2ParamsT ...

<ClassName> é o nome totalmente qualificado da classe Java. No Windows, <DllName> é o nome da DLL, sem a extensão *.dll. No Linux ou UNIX, ele é o nome do objeto compartilhado, sem o prefixo lib ou a extensão *.so.

<CustomPropertyName1> e <CustomPropertyName2> são os nomes das propriedades que você configurou na etapa 2.

- Salve o arquivo *.tgp.
- Armazene o arquivo no subdiretório DataTransformation\autoInclude\user do diretório de instalação de cada computador no qual você deseja usar o componente.
- Se a ferramenta Developer estiver aberta, feche-a e reabra-a.
- Se um erro autoInclude for exibido, revise o arquivo TGP em busca de erros de sintaxe ou inconsistências de nomenclatura e abra novamente a ferramenta Developer.
- Abra um projeto e insira o componente personalizado no Script. O nome do componente personalizado, que você atribuiu na etapa 3 acima, aparece na lista suspensa do IntelliScript. O editor do IntelliScript exibe suas propriedades.

Amostras de Script que Contêm Componentes Personalizados

Você pode encontrar amostras de arquivos de Script que contêm componentes personalizados nos seguintes subdiretórios do diretório de instalação:

```
samples\SDK\ExternalParameters\Java_SDK\autoInclude
samples\SDK\ExternalParameters\Cpp_SDK\autoInclude
```

ÍNDICE

A

- AbsURL
 - componente [260](#)
- ações
 - comparado com transformadores [296](#)
 - definindo [296](#)
 - efeitos colaterais [295](#)
 - entrada e saída [295](#)
 - propriedades de [296](#), [403](#)
- AcroForms
 - processando formulários PDF [167](#)
- AddEmptyTagsTransformer
 - componente [260](#)
- AddEventAction
 - componente [297](#)
- AddHeaderModifier
 - componente [394](#)
- AdditionalInputPort
 - componente [154](#)
- AdditionalOutputPort
 - componente [156](#)
- AddString
 - componente [261](#)
- AddStringModifier
 - componente [395](#)
- AggregateValues
 - componente [298](#)
- AllStructure
 - componente [249](#)
- AllStructureLocal
 - componente [250](#)
- Alternativas
 - componente [216](#)
- AlternativeMappings
 - componente [355](#)
- AlternativeSerializers
 - componente [340](#)
- AlternativeValidators
 - componente [404](#)
- analisador
 - componentes de Script [150](#)
- Analisador
 - componente [151](#)
 - descrição [24](#)
 - para dados COBOL [56](#)
- analisador secundário
 - âncora EmbeddedParser [344](#)
- Analisador secundário
 - âncora EmbeddedParser [224](#)
- analisadores
 - chamando secundários [344](#)
- Analisadores
 - chamando secundário [224](#)
 - executando secundário [320](#)

- analisadores alternativos
 - selecionando [217](#)
- análise estruturada
 - StructureDefinition [241](#)
- AppendListItems
 - componente [300](#)
- AppendValues
 - componente [301](#)
- aritméticos
 - computações [303](#)
- arquivo de amostra
 - Parquet [62](#)
- arquivo de entrada de exemplo
 - transformação do Processador de Dados [26](#)
- arquivo de exemplo
 - Avro [51](#)
 - JSON [59](#)
 - XML [64](#)
- arquivo de origem de amostra
 - definindo na transformação do Processador de Dados [42](#), [77](#)
- arquivo TGP
 - para componentes personalizados [434](#)
- arquivos de esquema
 - adicionando a objetos de esquema [123](#)
 - definindo um editor padrão [133](#)
 - editar [134](#)
 - removendo de objetos de esquema [123](#)
- arquivos PDF
 - usando o processador PdfToTxt_4 [173](#)
- Arquivos XSD
 - esquemas [191](#)
- Assistente de Transformação de Processo de Dados
 - descrição [50](#)
- atribuindo
 - valor de saída [325](#)
- atributos
 - recipientes de dados [191](#)
- atributos XML
 - recipientes de dados [191](#)
- attributeFormDefault
 - objeto de esquema [130](#)
- AttributeSearch
 - componente [245](#)
- autoInclude
 - componentes personalizados [434](#)

B

- Base64Decoder
 - componente [262](#)
- Base64Encoder
 - componente [262](#)
- Biblioteca
 - criando na transformação do Processador de Dados [43](#)
 - propriedades do elemento [119](#)

- Biblioteca (
 - Visão geral [118](#)
- BidiConvert
 - componente [263](#)
- BinaryFormat
 - componente [179](#)
- BIRT
 - gerador de relatórios XmlToDocument [169](#), [170](#)
 - gerador de relatórios XmlToDocument_372 [169](#)
 - gerador de relatórios XmlToDocument_45 [170](#)

C

- CalculateValue
 - componente [303](#)
- caminho
 - resolvendo relativo [260](#)
- caracteres
 - especiais no IntelliScript [81](#)
- caracteres especiais
 - inserindo no IntelliScript [81](#)
- caracteres não teclado
 - inserção [143](#)
- CDATADecode
 - componente [263](#)
- CDATAEncode
 - componente [264](#)
- ChangeCase
 - componente [265](#)
- chave
 - propriedades de [371](#)
- Chave
 - componente [371](#)
- ChoiceStructure
 - componente [251](#)
- ChoiceStructureLocal
 - componente [251](#)
- classificando
 - recipientes de dados de ocorrência múltipla [326](#)
- Classificar
 - componente [326](#)
- COBOL
 - importando definição de dados [56](#)
 - recursos com Suporte [56](#)
 - testando o Analisador [57](#)
 - testando o Serializador [58](#)
- codificação
 - transformador de página de código [270](#)
- Codificador
 - componente [270](#)
- codificando
 - esquema XSD [192](#)
- combinações
 - de listas [305](#)
- CombineValues
 - componente [305](#)
- CommaDelimited
 - componente [185](#)
- ComplexSegment
 - componente [385](#)
- ComplexXmlSegment
 - componente [385](#)
- componente de inicialização
 - transformação do Processador de Dados [28](#)
- componente de inicialização, Script
 - descrição [144](#)
- componente de Notificação [419](#)
- componente de Script
 - descrição [140](#)
 - Java personalizado, desenvolvendo [432](#)
- componente global
 - definindo [141](#)
- componente local
 - definindo [141](#)
- componente NotificationHandler [420](#)
- componente NotifyFailure [421](#)
- componente personalizado
 - exemplo [431](#)
 - Java, desenvolvendo [432](#)
- componente ValidateDate [416](#)
- componente, global
 - definindo [141](#)
- componente, inicialização em Script
 - descrição [144](#)
- componente, local
 - definindo [141](#)
- componente, Script
 - descrição [140](#)
 - Java personalizado, desenvolvendo [432](#)
- componentes
 - no IntelliScript [78](#)
- componentes de Script
 - descrição [140](#)
 - nomes [141](#)
 - propriedades, descrição [142](#)
- componentes de Script personalizados
 - descrição [431](#)
 - propriedades [432](#)
- componentes de Script, personalizados
 - descrição [431](#)
 - propriedades [432](#)
- componentes globais
 - definindo [81](#)
- componentes, Script
 - descrição [140](#)
 - nomes [141](#)
 - propriedades, descrição [142](#)
- componentes, Script personalizado
 - descrição [431](#)
 - propriedades [432](#)
- comprimento mínimo
 - objeto de esquema [126](#)
- concatenação
 - strings [300](#)
- condição
 - garantindo no documento de origem [311](#)
- Conectar
 - componente [252](#)
- configurações de codificação
 - transformação do Processador de Dados [30](#)
- configurações de controle de saída
 - transformação do Processador de Dados [32](#)
- conjunto de quadros
 - analisando HTML [165](#)
- ContentSerializer
 - componente [336](#), [341](#)
- Conteúdo
 - componente [218](#)
- conteúdo misto
 - mapeando para [206](#)
 - no esquema [192](#)
 - recipientes de dados em [194](#)
- Conversão de PDF
 - configurando [174](#)

- cores
 - realce no exemplo de documento de origem [145](#)
- CreateGuid
 - componente [265](#)
- CreateList
 - componente [306](#)
- CreateUUID
 - componente [265](#)
- critérios de pesquisa
 - para âncoras [210](#)
- CustomFormat
 - componente [180](#)
- CustomLog
 - componente [307](#)

D

- da direita para a esquerda
 - invertindo [263](#), [275](#)
- dados
 - validando [402](#)
- dados inválidos
 - detectando [398](#)
- dados não encontrados
 - tratamento de falhas [399](#)
- datas
 - formato de [266](#)
- DateAddICU
 - componente [307](#)
- DateDiff
 - componente [308](#)
- DateDiffICU
 - componente [308](#)
- DateFormat
 - componente [267](#)
- DateFormatICU
 - componente [266](#)
- decimais com sinais
 - números [274](#)
- decimais compactados
 - números [274](#)
- Declaração
 - elemento Regra de Validação [423](#)
- Delimitador
 - componente [186](#)
- delimitadores
 - hierarquia personalizada [180](#)
 - relação com âncoras [205](#)
- DelimitedSections
 - componente [221](#)
- DelimitedSectionsSerializer
 - componente [342](#)
- DelimiterHierarchy
 - componente [186](#)
- deslocamento
 - definido dinamicamente [246](#)
- destino
 - propriedade [365](#), [369](#)
- dinâmico
 - deslocamento [246](#)
 - pesquisar [248](#)
- diretrizes de codificação
 - transformação do Processador de Dados [32](#)
- dividindo
 - arquivos [329](#)
- dividindo entradas extensas
 - Streamer [377](#)

- DocList
 - componente [159](#)
- documento de origem, realce
 - descrição [145](#)
- documento, origem de exemplo
 - descrição [145](#)
- DoNothingModifier
 - componente [395](#)
- Dos96HebToAscii
 - componente [268](#)
- DownloadFileToDataHolder
 - componente [309](#)
- dp:as_XML
 - função do Processador de Dados [109](#)
- dp:get_id
 - função do Processador de Dados [109](#)
- dp:lookup
 - função do Processador de Dados [109](#)
- dp:output
 - função do Processador de Dados [109](#)
- DumpValues
 - componente [310](#)
- DynamicTable
 - componente [269](#)

E

- EbcdicToAscii
 - componente [269](#)
- EDI
 - delimitadores para análise [186](#)
- editor de configuração de tabela
 - PdfToTxt_4 [173](#)
- Editor de Expressão de Entrada
 - transformação do Processador de Dados [108](#)
- Editor de Expressão do XPath
 - transformação do Processador de Dados [108](#)
- editor de XMap
 - grade [87](#)
- editor de XPath
 - elementos Regra de Validação [426](#)
- editor do IntelliScript
 - componentes e propriedades [78](#)
 - descrição [146](#)
- Editor do IntelliScript [77](#)
- editor, IntelliScript
 - descrição [146](#)
- editores
 - IntelliScript [77](#)
- elementFormDefault
 - objeto de esquema [130](#)
- elementos
 - recipientes de dados [191](#)
- elementos de XML
 - recipientes de dados [191](#)
- elementos Regra de Validação
 - editor de XPath [426](#)
- EmbeddedMapper
 - componente [356](#)
- EmbeddedParser
 - componente [224](#)
- EmbeddedSerializer
 - componente [344](#)
- EmbeddedStructure
 - componente [253](#)
- enclosed
 - group [226](#)

- EnclosedGroup
 - componente [226](#)
- EnclosingDelimiters
 - componente [187](#)
- EncodeAsUrl
 - componente [270](#)
- EnsureCondition
 - componente [311](#)
- entrada relacional desnormalizada
 - Transformação de Processador de Dados [71](#)
- entrada relacional dinamizada
 - Transformação de Processador de Dados [71](#)
- entrada relacional normalizada
 - Transformação de Processador de Dados [70](#)
- Enumeração
 - componente [405](#)
- erros
 - tratamento de falhas [399](#)
 - tratamento de falhas de validação [147](#)
- escolher hierarquia
 - Transformação de Processador de Dados [67, 73](#)
- escopo de pesquisa
 - ajustando [212](#)
 - para âncoras [210](#)
- espaço de nome
 - alterando o prefixo gerado [125](#)
- espaços de nome
 - objeto de esquema [126](#)
- esquema
 - Avro
 - esquema [51](#)
 - codificando [192](#)
 - JSON
 - esquema [59](#)
 - para dados COBOL [56](#)
 - Parquet
 - definição [62](#)
 - esquema [62](#)
 - XML
 - esquema [64](#)
- esquemas
 - Arquivos XSD [191](#)
 - esquemas incluídos [192](#)
 - recursos sem suporte [192](#)
 - representação do IntelliScript [194](#)
 - transformação do Processador de Dados [29](#)
 - validação [195](#)
- evento de aviso
 - transformação do Processador de Dados [37](#)
- evento de erro fatal
 - transformação do Processador de Dados [37](#)
- evento de falha
 - transformação do Processador de Dados [37](#)
- evento de falha opcional
 - transformação do Processador de Dados [37](#)
- evento de notificação
 - transformação do Processador de Dados [37](#)
- eventos
 - manipulando [398](#)
 - transformação do Processador de Dados [37](#)
- eventos de notificação
 - StructureDefinition [244](#)
- Excel
 - analisando como XML [164](#)
 - gerando do XML [171](#)
- ExcelToDataXml
 - componente [164](#)

- ExcelToXml
 - componente [164](#)
- ExcludelItems
 - componente [314](#)
- exemplo de documento de origem, realce
 - descrição [145](#)
- exemplo de origem
 - painel Esquema do XMap [86](#)
- Exemplo de Origem
 - criando na transformação do Processador de Dados [44](#)
- Exibição Ajuda de Scripts
 - transformação do Processador de Dados [25](#)
- exibição Ajuda do IntelliScript
 - descrição [146](#)
- exibição Componente
 - transformação do Processador de Dados [25](#)
- Exibição de Configurações
 - transformação do Processador de Dados [30](#)
- Exibição de Eventos do Processador de Dados
 - exibindo log de eventos [40](#)
 - transformação do Processador de Dados [38](#)
- Exibição de eventos, Processador de Dados
 - exibindo log de eventos [40](#)
- exibição de Portas
 - Transformação de Processador de Dados [67, 73](#)
- Exibição de Referência
 - transformação do Processador de Dados [29](#)
- Exibição de Tipos Simples
 - objeto de esquema [128](#)
- exibição Esquema
 - Objeto de esquema [125](#)
 - propriedades avançadas de tipos simples [128](#)
- exibição Eventos
 - transformação do Processador de Dados [38](#)
- exibição Origem Hexadecimal
 - descrição [146](#)
- Exibição Origem Hexadecimal do Processador de Dados
 - descrição [25](#)
- exibição Origem Hexadecimal, Processador de Dados
 - descrição [25](#)
- Exibição Repositório
 - transformação do Processador de Dados [25](#)
- exibição, Ajuda do IntelliScript
 - descrição [146](#)
- exibição, Origem Hexadecimal
 - descrição [146](#)
- ExpandFrameSet
 - componente [165](#)
- expressão de teste
 - Editor de Expressão do XPath de Entrada [108](#)
- expressões regulares
 - sintaxe [282](#)
- expressões XPath
 - XMap [104](#)
- ExternalJavaPreProcessor
 - componente [166](#)
- ExtractRecord
 - componente [228](#)
- extraindo conteúdo
 - Âncora de conteúdo [218](#)

F

- falha
 - em vigor no pai [399](#)
- falha opcional
 - em vigor no pai [399](#)

- falhas
 - manipulação [399](#)
 - manipulando [398](#)
- falhas de validação
 - manipulação [147](#)
- fases
 - aninhado [210](#)
- FileSearch
 - componente [159](#)
- FindReplaceAnchor
 - componente [228](#)
- FormatNumber
 - componente [271](#)
- formato
 - pré-processadores [189](#)
- formulários
 - processamento de PDF [167](#)
- FromFloat
 - componente [272](#)
- FromInteger
 - componente [273](#)
- FromPackDecimal
 - componente [274](#)
- FromSignedDecimal
 - componente [274](#)
- funções do Processador de Dados
 - descrição [109](#)

G

- gerador de relatórios
 - BIRT [169](#), [170](#)
- grade
 - XMap [87](#)
- group
 - realizando ações em [231](#)
 - repetição [236](#)
- GroupMapping
 - componente [357](#)
- GroupSerializer
 - componente [346](#)
- Grupo
 - componente [231](#)
- grupo de repetição
 - realce no exemplo de documento de origem [145](#)
- grupo de substituição
 - objeto de esquema [127](#)

H

- Hebraico
 - conversão de página de código [275](#)
- hebrewBidi
 - componente [275](#)
- HebrewDosToWindows
 - componente [275](#)
- HebrewEBCDICOldCodeToWindows
 - componente [275](#)
- hebUniToAscii
 - componente [275](#)
- hebUtf8ToAscii
 - componente [275](#)
- HIPAA
 - validação [166](#)
- HIPAAValidator
 - componente [166](#)

- HL7
 - componente [187](#)
- hora
 - sistema [198](#)
- hora do sistema
 - variável [198](#)
- horas
 - formato de [266](#)
- HTML
 - removendo marcas [283](#)
 - transformando entidades [275](#)
- HtmlEntitiesToASCII
 - componente [275](#)
- HtmlFormat
 - componente [181](#)
- HtmlProcessor
 - componente [190](#), [276](#)

I

- identificadores
 - IntelliScript [81](#)
- Idioma da Regra de Validação
 - VRL [327](#)
- independência de plataforma
 - Analísadores [150](#)
- indexação
 - exemplo [363](#)
- Indexação
 - recipientes de dados de ocorrência múltipla [202](#)
- initialization
 - variáveis [200](#)
- InjectFP
 - componente [276](#)
- InjectString
 - componente [277](#)
- InlineTable
 - componente [277](#)
- InputPort
 - componente [160](#)
- inserindo caracteres
 - não teclado [143](#)
- instrução de Grupo de Repetição
 - editor de XMap [87](#)
 - instrução Executar XMap
 - editor de XMap [87](#)
 - instrução RunMapplet
 - editor de XMap [87](#)
- instrução de Opção
 - editor de XMap [87](#)
- instrução de predicado
 - expressões XPath [105](#)
- instrução de Roteador
 - editor de XMap [87](#)
- instrução Executar XMap
 - instrução de mapeamento [98](#)
- Instrução padrão
 - editor de XMap [87](#)
- instrução RunMapplet
 - instrução de mapeamento [99](#)
- Instruções de grupo
 - XMap [90](#)
- instruções de Grupo de Repetição
 - descrição [92](#)
- instruções de mapeamento
 - editor de XMap [87](#)

- Instruções de Mapeamento
 - Recortar e Colar [104](#)
- instruções de Opção
 - editor de XMap [94](#)
- instruções de Roteador
 - editor de XMap [94](#)
- IntelliScript
 - definindo âncoras em [208](#)
 - editando [79](#)
 - ícones usados no [82](#)
 - restrições de nomenclatura [81](#)
- interface da linha de comando
 - comando CM_console [136](#)
- iterações
 - âncora RepeatingGroup [236](#)

J

- Java
 - componente personalizado, desenvolvendo [432](#)
- JavaScript
 - extensões [313](#)
 - referência de sintaxe [312](#)
- JavaTransformer
 - componente [278](#)

L

- LearnByExample
 - componente [246](#)
- LengthEquals
 - componente [406](#)
- Lista
 - elemento Regra de Validação [424](#)
- listas
 - classificando [326](#)
 - combinando [305](#)
 - criando [306](#)
 - de variáveis [201](#)
 - recipientes de dados de ocorrência múltipla [201](#)
- LocalFile
 - componente [160](#)
- localizações
 - marcando no documento de origem [234](#)
- Localizador
 - componente [374](#)
- localizadores
 - propriedades de [371](#)
- LocatorByKey
 - componente [374](#)
- LocatorByOccurrence
 - componente [375](#)
- log
 - definição [38](#)
- log de eventos
 - eventos de personalizados [297](#)
 - exibindo [40](#)
- log de eventos de tempo de design
 - descrição e localização [39](#)
- log de eventos de tempo de execução
 - transformação do Processador de Dados [39](#)
- log de eventos, tempo de design
 - descrição e localização [39](#)
- log do usuário
 - localização que define a variável [198](#)
 - transformação do Processador de Dados [40](#)

- log, eventos de tempo de design
 - descrição e localização [39](#)
- logs
 - gravando em [398](#), [418](#)
- LookupTransformer
 - componente [278](#)
- loop
 - âncora RepeatingGroup [236](#)

M

- manipulação de falhas
 - variáveis para [198](#)
- manipulando
 - falhas [399](#)
 - falhas de validação [147](#)
- Mapa
 - componente [89](#), [315](#)
- mapeador
 - validação de entrada [196](#)
- Mapeador
 - chamando secundário [356](#)
 - componente [353](#)
 - criando [350](#)
 - descrição [24](#)
- Mapeador secundário
 - âncora EmbeddedMapper [356](#)
- mapeadores
 - usando a indexação [363](#)
- Mapeadores
 - executando no Analisador [318](#)
 - propriedades de [352](#)
- Mapplet
 - execução secundária [320](#), [322](#)
 - referência [29](#)
- Marcador
 - componente [234](#)
- marcadores
 - em Streamers [380](#)
- MarkerStreamer
 - componente [387](#)
- máximo de ocorrências
 - objeto de esquema [126](#)
- MaxLength
 - componente [408](#)
- MaxNumber
 - componente [408](#)
- mensagens
 - notificações de aviso [418](#)
- mínimo de ocorrências
 - objeto de esquema [126](#)
- MinLength
 - componente [409](#)
- MinNumber
 - componente [410](#)
- modo de intelli
 - descrição [146](#)
- modo de Script
 - descrição [146](#)

N

- NewlineSearch
 - componente [246](#)
- nome do serviço
 - armazenamento de variáveis [198](#)

- nomes
 - componentes de Script [141](#)
 - IntelliScript [81](#)
- NormalizeClosingTags
 - componente [280](#)
- notificações
 - disparando [317](#)
 - gerando [398](#)
 - gravando mensagens [418](#)
- Notificar
 - componentes [317](#)
- NotificationGroup
 - componente [419](#)
- NumberEquals
 - componente [411](#)
- números
 - formatando [271](#)

O

- objeto de esquema
 - propriedades avançadas de elementos [127](#)
 - arquivos de esquema [123](#)
 - attributeFormDefault [130](#)
 - definindo um editor padrão [133](#)
 - editando um arquivo de esquema [132](#)
 - elementFormDefault [130](#)
 - elementos complexos [129](#)
 - espaços de nome [126](#)
 - exibição Visão Geral [124](#)
 - grupo de substituição [127](#)
 - importando [131](#)
 - localização do arquivo [130](#)
 - propriedade abstrata [126](#)
 - propriedade de bloqueio [127](#)
 - propriedade herdar de [129](#)
 - propriedade herdar por [129](#)
 - propriedades avançadas de elementos complexos [129](#)
 - propriedades do atributo [130](#)
 - propriedades do elemento [126](#)
 - sincronização [132](#)
 - tipo simples [128](#)
 - visão geral [123](#)
- Objeto de esquema
 - exibição Esquema [125](#)
- objeto Regra de Validação
 - criando na transformação do Processador de Dados [44](#)
- objeto XMap
 - criando na transformação do Processador de Dados [43](#)
- ocorrência múltipla
 - destruindo ocorrências [202](#)
 - recipientes de dados [201](#)
 - variáveis [201](#)
- ocorrência única
 - recipientes de dados [201](#)
- OffsetSearch
 - componente [246](#)
- Opção
 - componente [95](#)
- origem
 - propriedade [365](#), [366](#)
- origem de exemplo
 - configurando [145](#)
 - descrição [145](#)
 - exibindo [146](#)
- origem de exemplo de LocalFile
 - descrição [145](#)
- origem de exemplo de Text
 - descrição [145](#)
- origem de exemplo de URL
 - descrição [145](#)
- OutputDataHolder
 - componente [331](#)
- OutputFile
 - componente [331](#)
- OutputPort
 - componente [160](#)

P

- Padrão
 - componente [97](#)
- padrão de correspondência
 - expressões regulares [282](#)
- padrões
 - abertura e fechamento de segmento [378](#)
- páginas de código
 - esquema XSD [192](#)
 - transformando [270](#)
- parâmetros
 - transmitindo para a transformação [200](#)
- parâmetros de serviço
 - transmitindo para a transformação [200](#)
- PatternSearch
 - componente [247](#)
- PDF
 - processando formulários PDF [167](#)
- PdfFormToXml_1_00
 - componente [167](#)
- PdfToTxt_3_02
 - componente [167](#)
- PdfToTxt_4
 - componente [168](#)
 - usando [173](#)
- personalizar exibição
 - painel Esquema do XMap [86](#)
- Pesquisa
 - elemento Regra de Validação [424](#)
- pesquisador
 - componentes [213](#), [245](#)
- pesquisar
 - string de pesquisa definida dinamicamente [248](#)
- phase
 - de pesquisa de âncoras [209](#)
- ponto de entrada, Script
 - descrição [144](#)
- pontos de referência
 - âncora de Marcador [234](#)
 - âncoras ao redor [208](#)
 - ao redor de âncoras [387](#)
 - do escopo de pesquisa [212](#)
- porta de entrada de arquivo
 - transformação do Processador de Dados [26](#)
- porta de entrada de buffer
 - transformação do Processador de Dados [26](#)
- porta de parâmetro de serviço
 - transformação do Processador de Dados [26](#)
- porta de saída de arquivo
 - transformação do Processador de Dados [28](#)
- porta de saída de buffer
 - transformação do Processador de Dados [28](#)
- portas de parâmetro de serviço
 - transformação do Processador de Dados [27](#)

- pós-processador
 - XmlToDocument [169](#), [170](#)
 - XmlToDocument_372 [169](#)
 - XmlToDocument_45 [170](#)
- Posicional
 - componente [187](#)
- PostScript
 - componente [188](#)
- PowerpointToTextML
 - componente [168](#)
- pré-processadores
 - definindo [162](#)
 - documento [162](#)
 - formato [189](#)
- prefixo gerado
 - alterando para o espaço de nome [125](#)
- problemas de desempenho
 - variável VarLastFailure [198](#)
- processadores
 - documento [162](#)
 - Java personalizado [166](#)
 - referência [163](#)
 - usando transformadores como [258](#)
- processadores de documentos
 - definindo [162](#)
 - executando vários [169](#)
 - Java personalizado [166](#)
 - referência [163](#)
- ProcessByTransformers
 - componente [168](#)
- ProcessorPipeline
 - componente [169](#)
- propriedade abstrata
 - objeto de esquema [126](#)
- propriedade anulável
 - objeto de esquema [126](#)
- propriedade base
 - objeto de esquema [128](#)
- propriedade com valor fixo
 - objeto de esquema [126](#)
- propriedade de bloqueio
 - objeto de esquema [127](#)
- propriedade de enumeração
 - objeto de esquema [126](#)
- propriedade de marcação
 - de âncoras [387](#)
- propriedade de padrão
 - objeto de esquema [126](#)
- propriedade de variedade
 - objeto de esquema [128](#)
- propriedade direction
 - de âncoras [208](#)
- propriedade disabled
 - seleccionando no menu [83](#)
- propriedade example_source
 - Mapeador [353](#)
 - Serializador [338](#)
- propriedade herdar de
 - objeto de esquema [129](#)
- propriedade herdar por
 - objeto de esquema [129](#)
- propriedade marking
 - de âncoras [208](#)
- propriedade opcional
 - seleccionando no menu [83](#)
 - tratamento de falhas [399](#)
- propriedade optional
 - configurando [400](#)

- propriedade phase
 - de âncoras [208](#)
- propriedade reduzir espaço em branco
 - objeto de esquema [128](#)
- propriedades
 - componentes de Script personalizados [432](#)
 - componentes de Script, descrição [142](#)
 - de ações [296](#), [403](#)
 - de âncoras [208](#)
 - de Mapeadores [352](#)
 - de Serializadores [338](#)
 - de Streamers [384](#)
 - no IntelliScript [78](#)
 - Transformadores [259](#)
- propriedades avançadas
 - descrição [142](#)
- propriedades do atributo
 - objeto de esquema [130](#)
- propriedades do componente
 - valores, descrição [143](#)
- propriedades ocultas
 - exibindo [142](#)
- propriedades simples
 - descrição [142](#)
- propriedades, avançadas
 - descrição [142](#)
- propriedades, componente
 - valores, descrição [143](#)
- propriedades, ocultas
 - exibindo [142](#)
- propriedades, simples
 - descrição [142](#)

R

- Rastreamento
 - elemento Regra de Validação [425](#)
- realce no exemplo de documento de origem
 - descrição [145](#)
- recipientes de dados
 - conteúdo misto [194](#)
 - destruindo ocorrências [202](#)
 - identificando origem e destino [365](#)
 - indexação de ocorrência múltipla [360](#)
 - ocorrência única ou múltipla [201](#)
- recipientes de dados de ocorrência múltipla
 - classificando [326](#)
 - combinando [305](#)
 - criando lista em [306](#)
 - indexação [360](#)
 - mapeando âncoras para [206](#)
- RecordStructure
 - componente [253](#)
- RecordStructureLocal
 - componente [254](#)
- recortando e colando
 - componentes de Script [146](#)
- recuperando conteúdo
 - Âncora de conteúdo [218](#)
- Redimensionar
 - componente [285](#)
- referência
 - analizadores [151](#)
 - âncoras [216](#)
 - âncoras de serialização [340](#)
 - âncoras Mapeadoras [355](#)
 - delimitadores [184](#)

- referência ()
 - formatos [178](#)
 - indexação [371](#)
 - Mapeadores [353](#)
 - pré-processadores de formato [189](#)
 - processadores de documentos [163](#)
- regex
 - expressões regulares [282](#)
- registros
 - extraíndo e validando [241](#)
 - extraíndo no StructureDefinition [228](#)
- Regra
 - elemento Regra de Validação [425](#)
- Regra de Validação
 - editor de XPath [426](#)
 - elemento [423-426](#)
 - elemento Declaração [423](#)
 - elemento Lista [424](#)
 - elemento Pesquisa [424](#)
 - elemento Rastreamento [425](#)
 - elemento Regra [425](#)
 - elemento Variável [426](#)
- Regras de Validação
 - definição [422](#)
 - editar no editor externo [429](#)
 - editor [422, 429](#)
 - importando o serviço do Data Transformation [430](#)
- RegularExpression
 - componente [281](#)
- RemoveMarginSpace
 - componente [283](#)
- RemoveRtfFormatting
 - componente [283](#)
- RemoveTags
 - componente [283](#)
- RepeatingGroup
 - componente [236](#)
- RepeatingGroupMapping
 - componente [358](#)
- RepeatingGroupSerializer
 - componente [346](#)
- ResetVisitedPages
 - componente [317](#)
- ResultFile
 - componente [332](#)
- reversão
 - depois de uma falha [400](#)
- ReverseTransformer
 - componente [285](#)
- RTF
 - componente [188](#)
- RtfFormat
 - componente [182](#)
- RtfProcessor
 - componente [190, 286](#)
- RtfToASCII
 - componente [286](#)
- RtfToTextML
 - componente [169](#)
- RunMapper
 - componente [318](#)
- RunMapplet
 - componente [320](#)
- RunParser
 - componente [320](#)
- RunPCWebService
 - componente [322](#)

- RunSerializer
 - componente [323](#)
- RunXMap
 - componente [324](#)

S

- saída em PDF
 - pós-processador XmlToDocument [169, 170](#)
 - pós-processador XmlToDocument_45 [170](#)
 - XmlToDocument postprocessor_372 [169](#)
- saída relacional desordenada
 - Transformação de Processador de Dados [76](#)
- saída relacional dinâmica
 - Transformação de Processador de Dados [75](#)
- saída relacional normalizada
 - Transformação de Processador de Dados [75](#)
- Script
 - amostras [147](#)
 - criando na transformação do Processador de Dados [42, 77](#)
 - descrição [139](#)
 - editor, IntelliScript [146](#)
 - estrutura [140](#)
 - importando amostra [148](#)
- Script de amostra
 - importando [148](#)
- search
 - direção da âncora [208](#)
- segmento de cabeçalho
 - Streamer [378](#)
- segmento de rodapé
 - Streamer [378](#)
- segmento repetido
 - Streamer [378](#)
- segmentos
 - processando no Streamer [378](#)
- segmentos complexos
 - streamer [378](#)
- SegmentSearch
 - componente [247](#)
- selecionar e clicar
 - definindo âncoras [207](#)
- SequenceStructure
 - componente [255](#)
- SequenceStructureLocal
 - componente [255](#)
- serialização
 - modo [334](#)
 - usando transformadores em [259](#)
- Serializador
 - componente [338](#)
 - controlando geração automática [333](#)
 - para dados COBOL [56](#)
- Serializadores
 - executando no Analisador [323](#)
 - propriedades de [338](#)
 - solução de problemas gerados automaticamente [335](#)
- serializer
 - validação de entrada [196](#)
- ServiceLocation
 - variável [197](#)
- serviço de Data Transformation
 - importando para o repositório do Modelo [46](#)
- serviço de Transformação de Dados
 - executando a partir da linha de comando [136](#)
- Serviço do Data Transformation
 - importando para o repositório do Modelo [46](#)

- Serviço do Data Transformation ()
 - importar vários serviços [46](#)
- serviço, Transformação de Dados
 - executando a partir da linha de comando [136](#)
- SetValue
 - componente [325](#)
- SGML
 - componente [188](#)
- SimpleSegment
 - componente [388](#)
- SimpleXmlSegment
 - componente [389](#)
- sincronizar com o editor
 - Transformação do Processador de Dados [45](#)
- sistema
 - variáveis [197](#)
- SpaceDelimited
 - componente [188](#)
- streamer
 - segmentos complexos [378](#)
- Streamer
 - componente [390](#)
 - concatenação de cabeçalho [379](#)
 - criando [380](#)
 - descrição [24](#)
 - dividindo entradas extensas [377](#)
 - JsonStreamer [386](#)
 - padrões de abertura e fechamento de segmento [378](#)
 - saída [380](#)
 - segmento de cabeçalho [378](#)
 - segmento de rodapé [378](#)
 - segmento repetido [378](#)
- Streamer de texto
 - princípio de operação [378](#)
- Streamers
 - propriedades de [384](#)
- StreamerVariable
 - componente [391](#)
- strings
 - concatenando [300](#), [301](#)
- StringSerializer
 - componente [336](#)
- StructureDefinition
 - componente [241](#)
 - registros de extração [228](#)
- substituindo texto
 - no documento de origem [228](#)
- Substituir
 - componente [284](#)
- SubString
 - componente [286](#)
- suporte a PDF
 - convertendo arquivos PDF [168](#)
- suporte para PDF
 - convertendo arquivos PDF [167](#)

T

- TabDelimited
 - componente [189](#)
- tabelas
 - processando PDF [173](#)
- tamanho máximo
 - objeto de esquema [126](#)
- testar uma biblioteca
 - transformação de Processador de Dados [121](#)

- TextFormat
 - componente [182](#)
- TextML
 - esquema XML [172](#)
- Texto
 - componente [160](#)
- texto ausente
 - pesquisando por Grupo opcional [233](#)
- TextSearch
 - componente [248](#)
- tipo complexo
 - propriedades avançadas [129](#)
- tipo simples
 - objeto de esquema [128](#)
- tipos
 - XSI [195](#)
- tipos de dados
 - procurando [214](#)
- tipos de dados de lista
 - atributos [202](#)
- tipos de dados derivados
 - Tipo XSI [195](#)
- tipos de eventos
 - transformação do Processador de Dados [37](#)
- tipos de lista
 - mapeando para [214](#)
- tipos de membros
 - objeto de esquema [128](#)
- tipos XSI
 - mapeando recipientes de dados [195](#)
- ToFloat
 - componente [287](#)
- ToInteger
 - componente [288](#)
- ToPackDecimal
 - componente [288](#)
- transformação de Processador de Dados
 - testando uma biblioteca [121](#)
- Transformação de Processador de dados
 - ambiente não nativo [48](#)
- Transformação de Processador de Dados
 - entrada dinamicada [71](#)
 - entrada relacional
 - Transformação de Processador de Dados [68](#)
 - entrada relacional desnormalizada [71](#)
 - entrada relacional normalizada [70](#)
 - mapeando XML para portas [73](#)
 - mapeando XML para portas relacionais [67](#)
 - saída dinâmica [75](#)
 - saída relacional
 - Transformação de Processador de Dados [74](#)
 - saída relacional normalizada [75](#)
- transformação de Processamento de Dados
 - componente de inicialização [28](#)
- transformação do Processador de Dados
 - configurações de codificação [30](#)
 - configurações [30](#)
 - configurações de controle de saída [32](#)
 - configurações de processamento [34](#)
 - configurações de XML [35](#)
 - descrição [24](#), [25](#)
 - exibições [25](#)
 - exportando como um serviço [46](#)
 - logs de usuário [40](#)
 - portas de entrada [26](#)
 - portas de parâmetro de serviço [27](#)
 - portas de saída [28](#)
 - testando no Visualizador de Dados [45](#)

Transformação do Processador de Dados

- Configurações de XMap [35](#)

- criando [41](#)

- portas [26](#)

- saída relacional desordenada [76](#)

Transformador

- descrição [24](#)

transformadores

- como pré-processadores de documento [258](#)

- comparado com ações [296](#)

- Java personalizado [278](#)

- na serialização [259](#)

- padrão [258](#)

- sequências de [258](#)

- usando como processadores de documentos [168](#)

Transformadores

- propriedades de [259](#)

transformadores padrão

- em formato [258](#)

TransformationStartTime

- componente [289](#)

TransformByParser

- componente [290](#)

TransformByProcessor

- componente [291](#)

TransformByService

- componente [291](#)

TransformerPipeline

- componente [292](#)

transformers

- definindo [257](#)

- usando em âncoras [257](#)

TypeSearch

- componente [249](#)

U

UNIX

- criando Analisadores para [150](#)

URL

- relativo ao absoluto [260](#)

V

validação

- entrada XML [196](#)

- saída do Analisador de XML [195](#)

- XML [195](#)

validadores

- dados de entrada [402](#)

ValidateByExpression

- componente [412](#)

ValidateByPattern

- componente [413](#)

ValidateByTransformer

- componente [414](#)

ValidateByType

- componente [415](#)

ValidateValue

- componente [327](#)

ValidatorPipeline

- componente [417](#)

valores, propriedades de componente

- descrição [143](#)

VarCurrentPost

- variável [198](#)

VarCurrentURL

- variável [198](#)

variáveis

- definidas pelo usuário [197](#)

- Editor XMap [111](#)

- em Streamers [380](#)

- initialization [200](#)

- listas [201](#)

- mapeando âncoras para [200](#), [206](#)

- recipientes de dados [191](#)

- sistema [197](#)

- usando em ações [200](#)

Variável

- componente [201](#)

- elemento Regra de Validação [426](#)

VarLastFailure

- variável [198](#)

VarLinkURL

- variável [197](#)

VarRequestedURL

- variável [198](#)

VarServiceInfo

- variável [198](#)

VarSystem

- hora do sistema [198](#)

VRL

- Idioma da Regra de Validação [327](#)

W

WellFormedModifier

- componente [395](#)

Word

- analisando como XML [169](#)

WordToXml

- componente [169](#)

WriteSegment

- componente [396](#)

WriteValue

- componente [328](#)

X

Xerces

- validação de XML [196](#)

XMap

- descrição [85](#)

- executando no analisador [324](#)

- executando no mapeador [324](#)

- executando no serializador [324](#)

- Instruções de grupo [90](#)

- instruções de Roteador [94](#)

- painel Esquema [86](#)

- tipos de instrução [87](#)

- variáveis [111](#)

XML

- adicionando marcas vazias [260](#)

- esquemas [191](#)

- mapeando âncoras para [205](#)

- transformação XSLT [293](#)

- validação [196](#)

XML Streamer

- princípio de operação [382](#)

XmlFormat

- componente [183](#)

- XMLLookupTable
 - componente [293](#)
- XmlSegment
 - componente [392](#)
- XmlStreamer
 - componente [393](#)
- XmlToDocument
 - componente [169](#), [170](#)
- XmlToDocument_372
 - componente [169](#)
- XmlToDocument_45
 - componente [170](#)
- XmlToExcel
 - componente [171](#)

- XmlToXlsx
 - componente [171](#)
- XPath
 - notação modificada [194](#)
- XSD
 - codificação de esquema [192](#)
 - editores [191](#)
 - recursos de esquema sem suporte [192](#)
- XSLT
 - executando transformações [330](#)
- XSLTMap
 - componente [330](#)
- XSLTTransformer
 - componente [293](#)