
Building Packages for Percussion CM System

Version 6.7

Copyright and Licensing Statement

All intellectual property rights in the SOFTWARE and associated user documentation, implementation documentation, and reference documentation are owned by Percussion Software or its suppliers and are protected by United States and Canadian copyright laws, other applicable copyright laws, and international treaty provisions. Percussion Software retains all rights, title, and interest not expressly granted. You may either (a) make one (1) copy of the SOFTWARE solely for backup or archival purposes or (b) transfer the SOFTWARE to a single hard disk provided you keep the original solely for backup or archival purposes. You must reproduce and include the copyright notice on any copy made. You may not copy the user documentation accompanying the SOFTWARE.

The information in Rhythmyx documentation is subject to change without notice and does not represent a commitment on the part of Percussion Software, Inc. This document describes proprietary trade secrets of Percussion Software, Inc. Licensees of this document must acknowledge the proprietary claims of Percussion Software, Inc., in advance of receiving this document or any software to which it refers, and must agree to hold the trade secrets in confidence for the sole use of Percussion Software, Inc.

The software contains proprietary information of Percussion Software; it is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited.

Due to continued product development this information may change without notice. The information and intellectual property contained herein is confidential between Percussion Software and the client and remains the exclusive property of Percussion Software. If you find any problems in the documentation, please report them to us in writing. Percussion Software does not warrant that this document is error-free.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior written permission of Percussion Software.

Copyright © 1999-2009 Percussion Software.
All rights reserved

Licenses and Source Code

Rhythmyx uses Mozilla's JavaScript C API. See <http://www.mozilla.org/source.html> for the source code. In addition, see the Mozilla Public License <http://www.mozilla.org/source.html>.

Netscape Public License

Apache Software License

IBM Public License

Lesser GNU Public License

Other Copyrights

The Rhythmyx installation application was developed using InstallShield, which is a licensed and copyrighted by InstallShield Software Corporation.

The Sprinta JDBC driver is licensed and copyrighted by I-NET Software Corporation.

The Sentry Spellingchecker Engine Software Development Kit is licensed and copyrighted by Wintertree Software.

The Java™ 2 Runtime Environment is licensed and copyrighted by Sun Microsystems, Inc.

The Oracle JDBC driver is licensed and copyrighted by Oracle Corporation.

The Sybase JDBC driver is licensed and copyrighted by Sybase, Inc.

The AS/400 driver is licensed and copyrighted by International Business Machines Corporation.

The Ephox EditLive! for Java DHTML editor is licensed and copyrighted by Ephox, Inc.

This product includes software developed by CDS Networks, Inc.

The software contains proprietary information of Percussion Software; it is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited.

Due to continued product development this information may change without notice. The information and intellectual property contained herein is confidential between Percussion Software and the client and remains the exclusive property of Percussion Software. If you find any problems in the documentation, please report them to us in writing. Percussion Software does not warrant that this document is error-free.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior written permission of Percussion Software.

AuthorIT™ is a trademark of Optical Systems Corporation Ltd.

Microsoft Word, Microsoft Office, Windows®, Window 95™, Window 98™, Windows NT® and MS-DOS™ are trademarks of the Microsoft Corporation.

This document was created using AuthorIT™, Total Document Creation (see <http://www.author-it.com>).

Schema documentation was created using XMLSpy™.

Percussion Software

600 Unicorn Park Drive

Woburn, MA 01801 U.S.A.

781.438.9900

Internet E-Mail: technical_support@percussion.com

Website: <http://www.percussion.com>

Contents

| | |
|--|-----------|
| Building a Simple Package | 7 |
| Preparing to Build the Package..... | 8 |
| Copying the per.example.HelloWorld Content Type | 8 |
| Copying and Modifying the perc.example Configuration Files | 9 |
| Building a Package | 10 |
| Creating the einv.example Descriptor | 11 |
| Building the einv.example Package..... | 14 |
| Configuration Files | 17 |
| Default Configuration Files | 18 |
| Configuration Definition Files..... | 20 |
| Building Packages | 23 |
| Planning a Package | 24 |
| Creating a Descriptor | 26 |
| Building Packages | 28 |
| Testing a Package | 29 |
| Documenting a Package | 31 |
| Design Object Configuration Reference | 33 |
| Action Menu Configuration Reference..... | 34 |
| Action Menu Configuration Reference..... | 39 |
| Action Menu Configuration Reference..... | 41 |
| Action Menu Configuration Reference..... | 45 |
| Action Menu Configuration Reference..... | 46 |
| Action Menu Configuration Reference..... | 49 |
| Action Menu Configuration Reference..... | 51 |
| Action Menu Configuration Reference..... | 55 |
| Action Menu Configuration Reference..... | 62 |
| Action Menu Configuration Reference..... | 72 |
| Action Menu Configuration Reference..... | 76 |
| Action Menu Configuration Reference..... | 78 |

CHAPTER 1

Building a Simple Package

To demonstrate the process of building a solution package, we will build a simple package based on the perc.example package used to demonstrate the process of installing and configuring a solution package.

NOTE: Before proceeding with this exercise, you should already be familiar with the process of installing and configuring a package. If you are not already familiar with this process, you should complete the exercises described in *Installing and Maintaining Packages for Percussion CM System*.

Preparing to Build the Package

A design object in the Percussion CM System can only be included in one package, so we will need to copy the `perc.example.HelloWorld` Content Type used in the `perc.example` package and rename the copy to `einv.example.HelloWorld`. The name of each design object should consist of three parts: the publisher prefix, the solution name, and the design object name.

- The publisher prefix is a three-to-five character string used to identify the publisher of the package and to help ensure uniqueness among the design objects and properties installed on the system. Solutions from different vendors may use the same names for design objects or their properties. Including a vendor prefix in the name of the design object or property allows the system to process all of these objects and properties correctly. Percussion Software uses *perc* as its publisher prefix.
- The solution name is the name of the solution in the package. The solution name should be a short but meaningful name for the solution included in the package, as in the solution names *example* and *exshared* ("example shared field") in the example packages provided by Percussion Software.
- The design object name is the name for the specific design object; this is the name you would give the object without the publisher prefix and solution name. In the example package, the Content Type design object is *HelloWorld*.

In the renamed design object:

- We will use the new publisher prefix *einv* (for Enterprise Investments).
- We will use the same solution name: *example*.
- We will use the same design object name: *HelloWorld*.

We will name the package *einv.example*.

We will use copies of the configuration files installed with the `perc.example` package, renamed for the `einv.example` package. We will modify the configuration files to refer to the new `einv.example.HelloWorld` Content Type.

Copying the `perc.example.HelloWorld` Content Type

In this procedure we will copy and rename the `perc.example.HelloWorld` Content Type using the vendor prefix `einv`, for the vendor of the `EnterpriseInvestments` package. (NOTE: You can substitute your own vendor prefix in this procedure.)

To copy the `perc.exmaple.HelloWorld` Content Type:

- 1 Start and log in to the Percussion CM System Workbench.
- 2 Click on the Content Design tab, expand the Content Types node. Select the `perc.example.HelloWorld` Content Type, right click, and from the popup menu, choose *Copy*.
- 3 Select the Content Types node, right-click, and from the popup menu, choose *Paste*.

The Percussion Workbench displays the Content Type Copy dialog.

- 4 In the Copy Name column of the row of the perc.example.HelloWorld Content Type, enter *env.example.HelloWorld*.
- 5 Click the [OK] button.

The env.example.HelloWorld Content Type is added in the Percussion Workbench.

Copying and Modifying the perc.example Configuration Files.

We will need new configuration files to refer to the new Template we have created.

Use the standard process in your operating system to copy the perc.example_configDef.xml and perc.example_defaultConfig.xml files and rename them to *env.example.configDef.xml* and *env.example_defaultConfig.xml*.

Open the files you copied. Search and replace *perc* with *env*.

Building a Package

Once we have created our objects, we are ready to build the package. Use the Percussion Package Builder to build your packages.

The Percussion CM System Packager tool is part of the Percussion Developer Tools suite, and was installed with your Workbench, Server Administrator, and other developer tools. To start the Percussion CM System Packager tool, click on the Windows [Start] button, then choose *Programs > Percussion CM System > Packager*.

When you start the Packager tool, the main dialog is the Package List.

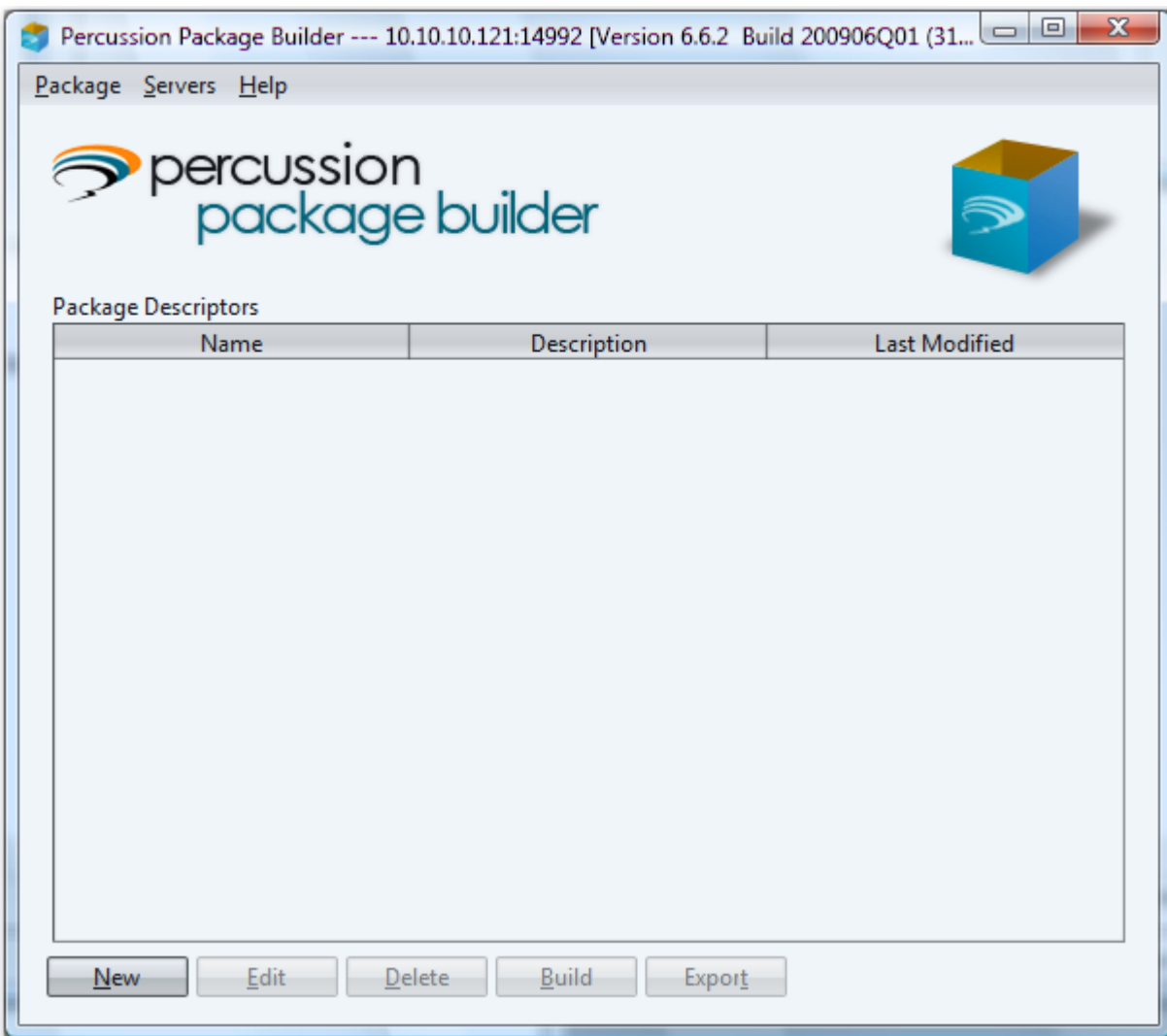


Figure 1: Package List Dialog

The Package List dialog lists the package descriptors that are defined on the server to which you are connected. If you do not see any packages listed, you are not connected to a server.

We will assume the following data for the connection procedure. Substitute your own connection data.

- Host: 10.10.10.121
- Port: 9992
- Username: admin1
- Password: demo

To connect to your server:

- 1 In the Menu bar, choose *Servers > Connection Manager*.
Percussion Package Builder displays the Connection Manager dialog.
- 2 Click the **[New]** button.
Package Manager clears the Connection Manager dialog.
- 3 In the **Host** field, enter *10.10.10.121*.
- 4 In the **Port** field, enter *9992*.
- 5 In the **Username** field, enter *admin1*.
- 6 In the **Password** field, enter *demo*.
- 7 Click **[Save and Connect]**.

This action connects and logs you in to the Percussion CM Server.

The process of building a package involves two phases:

1. Creating a package descriptor.
2. Building the package itself.

Creating the *env.example* Descriptor

A descriptor defines the design elements that will be included in a package, and various supporting data for the package.

We will name our descriptor *env.example*; the package will inherit this name. The descriptor will include the *env.example.HelloWorld* Content Type and the *env.example_configDef.xml* and *env.example_defaultConfig.xml* configuration files. This will be version 1.0.0 of the package, and the minimum version of Percussion CM System it will support will be 6.7. The publisher will be Enterprise Investments, using the URL www.enterpriseinvestments.com. The package will not include any file resources. The *perc.exshared* package will be included as a required dependency for the package.

To create the *env.example* descriptor:

- 1 On the Package List dialog, click the **[New]** button.
The Packager displays the Descriptor dialog with the General tab selected.
- 2 In the **Package Name** field, enter *env.example*.
- 3 In the **Description** field, enter *A modified example using the env.example.HelloWorld Content Type*.
- 4 In the **Publisher Name** field, enter *EnterpriseInvestments*.

- 5 In the Publisher URL field, enter *www.enterpriseinvestments.com*.
- 6 In the Package Version field, enter *1.0.0*.
- 7 In the Minimum CMS Version field, enter *6.7*.

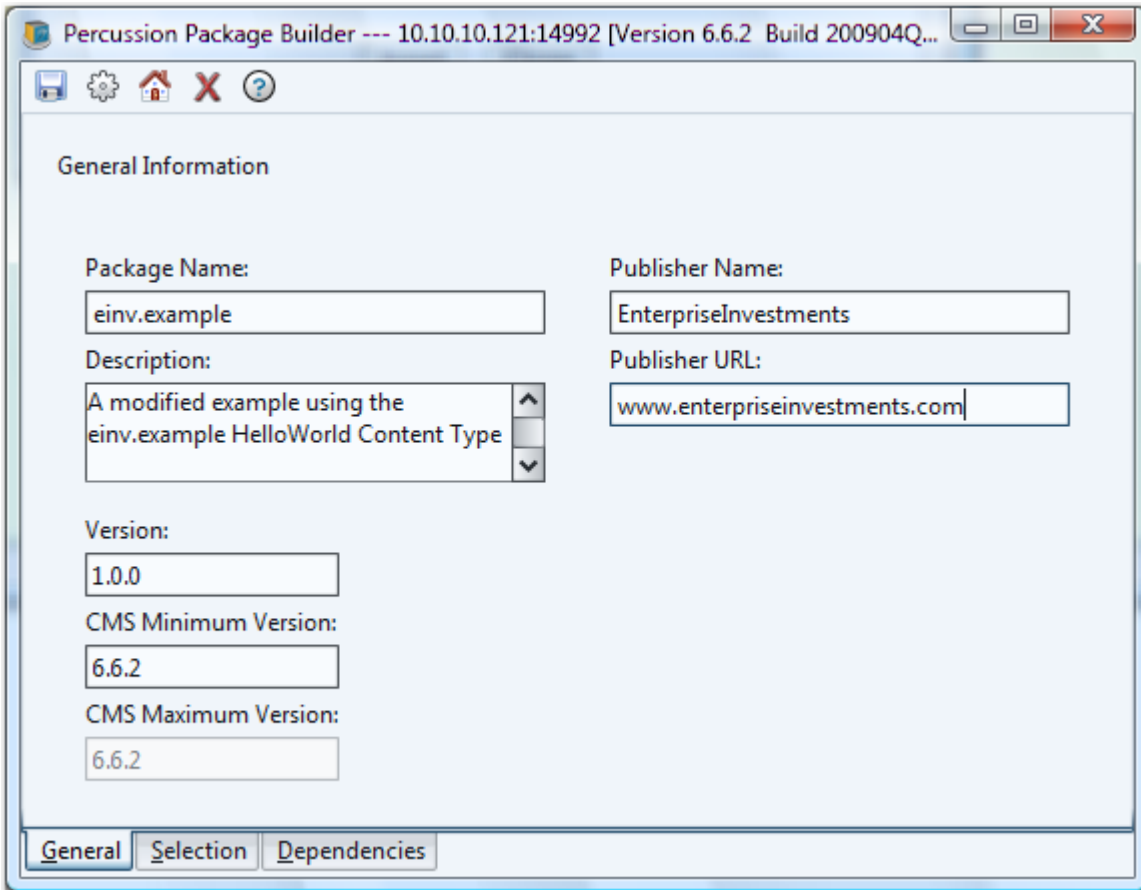


Figure 2: Package Builder General Tab with data for the *einv.example* descriptor

- 8 Click on the Selection tab.
- 9 In the Design Elements box, expand the Content Type node and check the box for the *einv.example.HelloWorld* Content Type.
- 10 Click the [...] button next to the Configuration Definition field. Browse to the file *einv.example.config_Def.xml*, select it, and click the [OK] button. The Packager adds this configuration definition file to the descriptor.

- 11 Click the [...] button next to the Default Configuration field. Browse to the file `env.example.default_Config.xml`, select it, and click the [OK] button. The Packager adds this configuration file to the descriptor.

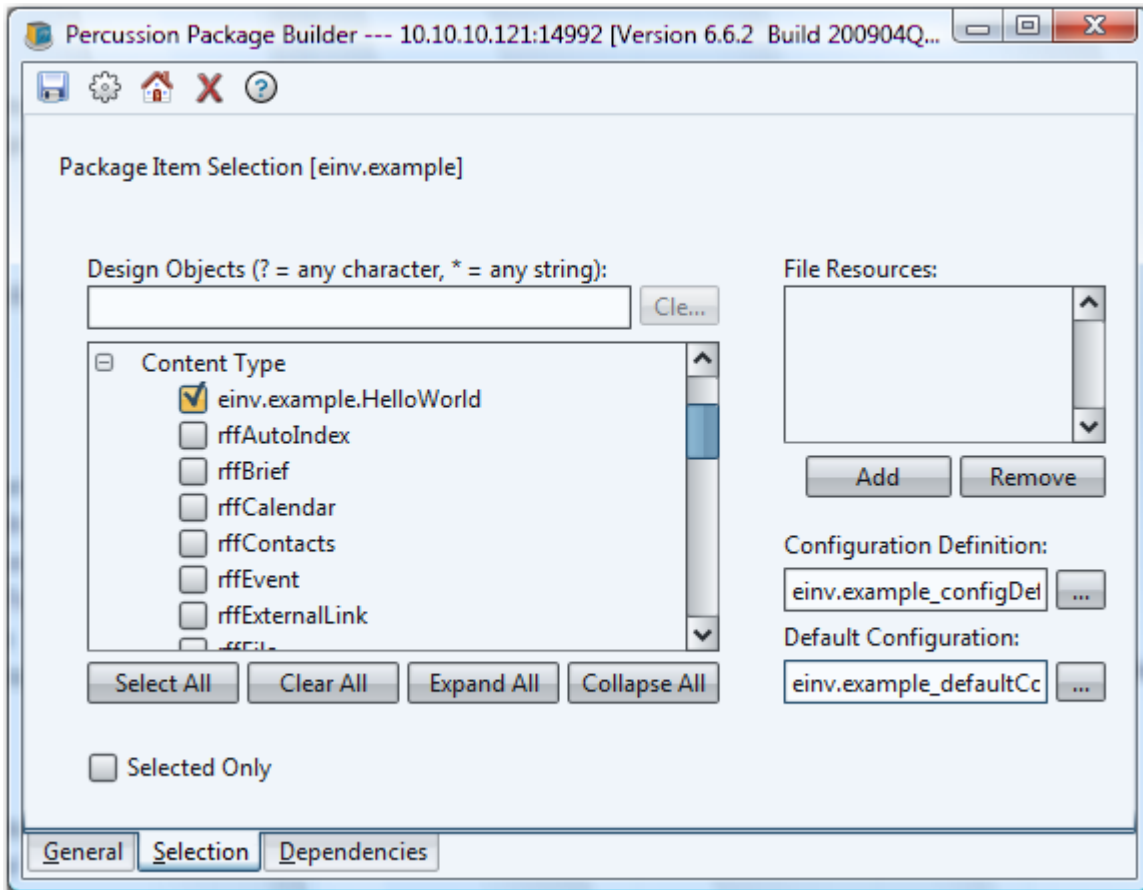


Figure 3: Package Builder Select tab with `env.example` Content Type selected

- 12 Click the Dependencies tab.

13 In the Additional Required Packages list, check the perc.exshared package.

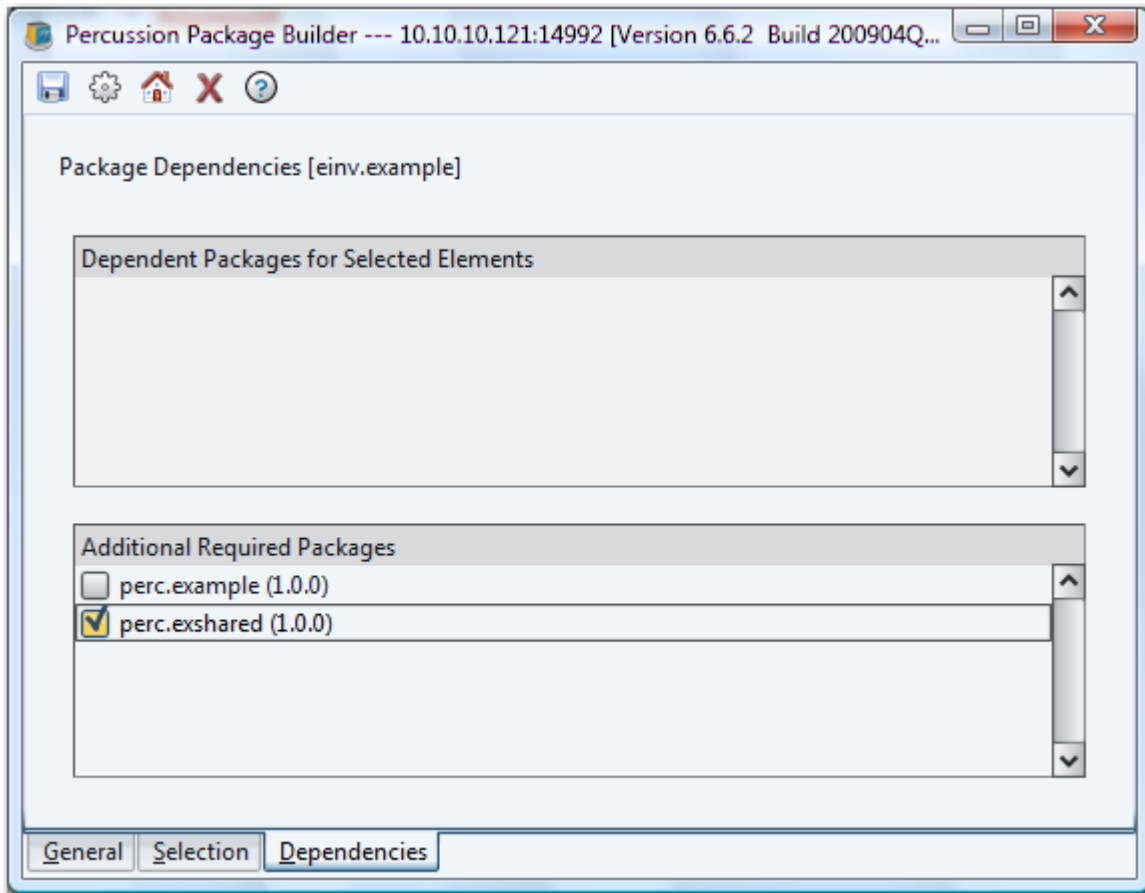


Figure 4: Package Builder Dependencies tab with perc.exshared package specified as a dependency

(Note: we are assuming that the perc.example package was in its original installation configuration when it was copied to make env.example. If you had not returned the perc.example.HelloWorld Content Type to its original configuration before copying it, the Content Type will include the New Field, and the perc.exshared package will be included as a required dependency.

14 Click the save button.

The Packager Builder saves the descriptor.

Building the env.example Package

Once you have a descriptor, you can build a package. To build the env.example package:

- 1 On the Package List dialog, select the env.example descriptor.
- 2 In the Menu bar, choose *Package > Build*.
- 3 The Packager displays the Select Target Directory dialog. Enter the path to the root of your Percussion CM Server installation. You can also browse to the location where you want to store the package.

- 4 Click the **[OK]** button.
- 5 The Packager build the package, which is stored as `env.example.ppkg` in the specified directory.

Now that you have built the package, you can test it. The most effective way to test the package is to deploy it to a Percussion CM Server installation that has no other implementation. Make the configuration changes described in the topic "Configuring the Hello World Package" in the document *Installing and Maintaining Packages for Percussion CM System* to confirm that your package is working correctly.

Configuration Files

In the Percussion CM System, implementers can define allowable configurations for design objects. When a design object is installed, the customer specifies the configurations for their system.

Available configurations are defined by three XML configuration files:

- The configuration definition XML file
This file defines the configurations available for the package. Configuration definitions contain Spring beans and must conform to the Spring bean schemas. Configuration definition files are stored in the directory `<ServerRoot>/rxconfig/Packages/ConfigDefs`. While configuration definition files can have any name during development, during installation the Percussion Package Installer renames the file according to the convention `packagename_configDef.xml` where `packagename` is the name defined for the package when you create it in the Percussion Package Builder tool.
- Two configuration specification XML files These files define the specific values for each configuration. Configuration specification files must conform to the schema `localConfig.xml`.
 - One of these files is the default specification file. It is stored in the directory `<ServerRoot>/rxconfig/Packages/DefaultConfigs`, and specifies the default values for all configurations in the package. When you are developing the package, this is the only specification file you need. At this time, the file can have any name. When the package is installed on the target server, the Percussion Package Installer saves it with the name `packagename_defaultConfig.xml` where `packagename` is the name defined for the package when you create it in the Percussion Package Builder tool.
 - The other file is the local specification file. It is stored in the directory `<ServerRoot>/rxconfig/Packages/LocalConfigs` and specifies the values to be used in the local environment for specific configurations (the values specified in this file override the values specified in the default configuration file). When you install a package, the Percussion Package Installer creates the local configuration file by saving a copy of the default configuration file in the `LocalConfigs` directory; the new file is named `packagename_localConfig.xml` where `packagename` is the name defined for the package when you create it in the Percussion Packager tool.

Note that these configurations are optional. You can create a package that does not permit configuration. If you do allow configuration, however, both the configuration definition and the default configuration must be included in the package.

Default Configuration Files

The root node of the default configuration file is the `<!SolutionConfigurations>` node. At a minimum, this node requires the following attributes:

- `publisherPrefix` The value of this node is your PublisherPrefix. The publisher prefix is a three- to five-letter abbreviation used to identify each publisher of packages. Packages, design objects, and properties from different publishers could have the same name. The publisher prefix should be included in the names to ensure uniqueness. For example, in the `perc.example` package, this attribute is `publisherPrefix="perc"`.
- `publisherName` The value of this node is the name of your organization. For example, in the `HelloWorld` package, this attribute is `publisherName="Percussion Software, Inc."`.

The `<SolutionConfigurations>` node for the `perc.example` package is

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software, Inc." type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
```

NOTE: The namespaces are not necessary, but if included, must be defined as in the example code above.

For each solution in the package, you should include one `<SolutionConfig>` node. Each `<SolutionConfig>` node includes the configurations for the set of design objects in a specific solution. Each `<SolutionConfig>` must include a `name` attribute. The value of this attribute should be the name of the specific solution. For example, in the `perc.example` default configuration file, the value of the attribute is `HelloWorld`.

Each `<SolutionConfig>` node contains a set of one or more `<property>` nodes, which specify the configurable properties for the design objects in the solution, and the value or values of each configurable property. All `<property>` nodes require the `name` attribute, which specifies the name of the property. The name is arbitrary, but should have meaning for the user that will be configuring your package.

For the simplest properties, the value can be set as part of the `<property>` node itself, using the `value` attribute. For example, in the `perc.example` package, the label of the `perc.example>HelloWorld` Content Type is set in this way:

```
<property name="label" value="HelloWorld"/>
```

A property may also contain a list of values. To specify a list of values, do not include the `value` attribute in the `<property>` node. Add a `<pvalues>` child node. The `<pvalues>` node then contains a set of `<pvalue>` child nodes. The content of a `<pvalue>` node is one of the values you want to specify. An example of this configuration is configuring Template visibility to a Site.

```
<property name="Visibility">
  <pvalues>
    <pvalue>rffPgCiGeneric</pvalue>
    <pvalue>rffPgCiEvent</pvalue>
    <pvalue>rffPgCiHome</pvalue>
    <pvalue>rffPgCiGenericWord</pvalue>
    <pvalue>rffPgCiPressRelease</pvalue>
  </pvalues>
</property>
```

Rather than a simple list, a configuration may consist of a set of pairs of values, such as a name/value pair. The `<pvalues>` node is also used when specifying a set of pairs, but the child nodes are `<pair>` nodes. Each pair node has two attributes, `value1` and `value2`. The `value1` attribute is the first value in the pair, while the `value2` attribute is the second value in the pair. A common example of this approach to configuration is association Content Type/Template pairs with a Slot:

```
<property name="slotAssociations">
  <pvalues>
    <pair value1="rffEvent" value2="rffSnTitleLink"/>
    <pair value1="rffGeneric" value2="rffSnTitleCalloutLink"/>
    <pair value1="rffGenericWord"
value2="rffSnTitleCalloutAndMoreLink"/>
  </pvalues>
</property>
```

In this example, `value1` specifies the Content Type for each association, while `value2` specifies the Template that can be used with that Content Type in the Slot.

You also use this method to specify sets of name/value pairs. When specifying a name/value pair, the `value1` attribute specifies the name in the pair and `value2` specifies the value in the pair. You can use this approach to configure a set of binding variables for a Template:

```
<property name="bindingset">
  <pvalues>
    <pair value1="$binding1" value2="binding1value" />
    <pair value1="$binding2" value2="binding2value" />
    <pair value1="$binding3" value2="binding3value" />
  </pvalues>
</property>
```

Configuration Definition Files

A configuration definition file consists of a set of Spring beans that define the design objects and properties available to be configured. The root node is a standard Spring `<beans>` node with all required Spring namespaces.

```
<!-- See Item 1 -->
<bean id="NewCopy"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <!-- See Item 2 -->
  <property name="name" value="NewCopy"/>
    <!-- See Item 3 -->
    <property name="type" value="RELATIONSHIP_CONFIGNAME"/>
      <!-- See Item 4 -->
      <property name="propertySetters">
        <list>
          <!-- See Item 5 -->
          <bean
class="com.percussion.rx.config.impl.PSRelationshipConfigSetter">
            <property name="properties">
              <!-- See Item 6 -->
              <map>
                <entry key="deepCloning"
value="\${perc.SystemObjects.NewCopy.deepCloning}"/>
                <entry key="shallowCloning"
value="\${perc.SystemObjects.NewCopy.shallowCloning}"/>
                <entry key="fieldOverrides"
value="\${perc.SystemObjects.NewCopy.fieldOverrides}"/>
              </map>
            </property>
          </bean>
          <bean
class="com.percussion.rx.config.impl.PSRelationshipConfigEffectSetter">
            <property name="properties">
              <map>
                <entry key="effects"
value="\${perc.SystemObjects.NewCopy.effects}"/>
              </map>
            </property>
          </bean>
        </list>
      </property>
    </bean>
  </bean>
</property>
</bean>
```

- 1 The individual configurations are Spring beans. The value of the `id` attribute of the `<bean>` node is arbitrary, but best practice is to use the name of the design object being configured. The value of the `class` attribute of the `<bean>` node specifies the configuration handler class. In most cases, the value should be `com.percussion.config.impl.PSObjectConfigHandler`, which has no properties. Site and Location Scheme configuration each use their own handler:
 - Site: `com.percussion.rx.config.impl.PSSiteConfigHandler`

- Location Scheme: `com.percussion.rx.config.impl.PSLocationSchemeConfigHandler`

See the configuration reference for details about these handlers.

- 2 A configuration bean includes either a name property node or a names property node. The name property is used for configuring a single design object. The value of the name property must be the name of the design object being configured. (Thus, it should match the value of the id attribute of the `<bean>` node.)

```
<property name="name" value="NewCopy" />
```

In some cases, a package may include multiple design objects that share the same name. In that situation, best practice is to append a suffix indicating the type of design object to avoid confusion. For example, if a package included both a Content Type and a Workflow named Generic, the name property of the configuration beans would be:

```
<property name="name" value="Generic_ct" />
<property name="name" value="Generic_wf" />
```

The names property is used for configuring multiple design objects. The value of the names property must resolve to one or more design object names. It may be a replacement variable (`${publisherPrefix.Solution.DesignObjectVariable}`) or a wildcard expression using the star character (*; for example, `**invest*`).

- 3 Each configuration bean must include a type property node.

```
<property name="type" value="RELATIONSHIP_CONFIGNAME" />
```

The value specifies the type of the design object. See the `DesignObjectConfigurationReference` for details.

- 4 Configurable properties are defined in a propertySetters node:

```
<property name="propertySetters">
```

- 5 The propertySetters node contains a bean that defines the property setter class:

```
<bean
class="com.percussion.rx.config.impl.PSRelationshipConfigSetter">
```

The property setter class specified must be the correct property setter for the type of design object. See the *Design Object Configuration Reference* (on page 33) for the correct property setter class for each type of design object.

- 6 Within the property setter class bean, the properties property node contains the actual configurations within a `<map>` node. The `<map>` node contains a set of `<entry>` nodes.

```
<property name="properties">
  <map>
    <entry key="deepCloning"
value="${perc.SystemObjects.NewCopy.deepCloning}" />
    <entry key="shallowCloning"
value="${perc.SystemObjects.NewCopy.shallowCloning}" />
    <entry key="fieldOverrides"
value="${perc.SystemObjects.NewCopy.fieldOverrides}" />
  </map>
</property>
```

Each <entry> node defines a configurable property. An entry node requires two attributes. The value of the key attribute specifies the design object property you want to expose for configuration. The value of the value attribute is the fully-qualified replacement variable for that property. The fully-qualified replacement variable includes the following data from the default and local configuration files:

- the publisher prefix, as specified in the publisherPrefix attribute of the <SolutionConfigurations> node;
- the solution name, as specified in the name attribute of the <SolutionConfig> node; and
- the configuration name, as specified in the name attribute of the <property> child of the <SolutionConfig> node; this value is arbitrary, but the value chosen should describe the property in the business language of the user.

The replacement variable can include any valid XML character, including spaces. Characters that must be escaped in XML must be escaped in the replacement variable.

Note that some configurable properties are more complex, and may include additional properties. In these cases, the <entry> node will contain an additional <Map> child with further <entry> nodes.

CHAPTER 3

Building Packages

The process of building a package includes the following phases:

- 1 Plan your package, determining the design objects that will be in the package, supporting files, and other package data. This phase is optional, but will save you time, especially when creating the package descriptor.
- 2 Create the package descriptor.
- 3 Build the package.
- 4 If your package allows configuration:
 - a) Export a configuration definition file. Remove any configurations you do not want to allow for your package.
 - b) Build a default configuration file. Typically, the two files are developed and finalized concurrently.
 - c) Update the descriptor with the configuration files and rebuild the package.

Once you have built the package, best practice is to test the package by deploying it to a new server.

NOTE: The Package Builder does not package SQL views. If your package includes an XML application that uses a database view, you must create a SQL script to create the view and deliver it with the package to your customers. The SQL view must be added to the target environment before installing the package.

Planning a Package

Planning your package is an optional step, but it is a recommended practice to save time and reduce errors when creating the descriptor.

To plan your package, determine the following package data:

- The name of the package.
- The version of the package.
- Which versions of Percussion CM Server are supported for the package. In other words, to which versions can you install the package? The maximum version is always the current version of your server. When creating updates to a package on a later version of the server, you must decide whether you want to continue to support earlier versions of the Percussion CM Server. Packages can be installed to Percussion CM Server Version 6.7 or later. Packages cannot be installed to Percussion CM Server Version 6.6.1 or any earlier version of Rhythmyx. Best practice is to test installation of the package on each version supported to ensure that it works correctly.
- Which design objects will be included in the package.
- Whether any of the design objects require supporting files. If so, which files, and where do they reside?
- Whether any of the design objects require design objects from another package to work correctly. Usually the Package Builder flags package dependencies automatically. If the Package Builder cannot detect a dependency, you must flag it manually.
- Whether the package supports configuration. If so, what properties of the design objects in the package are configurable. Note that configuration is optional; you do not have to allow configuration of the design objects in the package. If you do allow configuration, both the configuration definition file and the default configuration file are required. A package must either include both files or neither. If you include one but not the other, you will not be able to save your descriptor.

Ideally, planning your package begins even earlier, when you are architecting your solution. When defining the implementation plan for your solution, you should plan the name of your package, and the names of the design objects in your package. To avoid collisions with design objects in packages from other publishers, you should use a fully-qualified name for each design object consisting of the following:

- the publisher prefix
- the solution name
- the design object name

For example, the Content Type included in the example package provided by Percussion Software is `perc.example>HelloWorld`, where

- *perc* is the publisher prefix for Percussion Software;
- *example* is the name of the solution; and
- *HelloWorld* is the name for the Content Type itself.

(NOTE: If your package includes a Java extension you have created and you have implemented the `init` method of the extension, the `init` method is called when the package is installed. Any properties that the `init` method loads may not be available after the initial installation of the package. We recommend lazy loading of properties in extensions designed for packages.)

Creating a Descriptor

To create a descriptor:

- 1 On the Package Descriptor List dialog, in the Menu bar, choose *Package > New*.
The Package Builder displays the Descriptor Definition dialog.
- 2 The value in the **Package Name** field defaults to *New_Package_Descriptor*. Change this to the name you want to give the package.

Package names should take the format `PublisherPrefix.SolutionName` where `PublisherPrefix` is the three- to -five-letter prefix you have chosen as your publisher prefix, and `SolutionName` is the name of the solution you are packaging. For example, `perc.Example` is one of the example packages shipped by Percussion Software.
- 3 In the **Version** field, enter the version of the package.
- 4 In the **CMS Minimum Version** field, enter the minimum version of the Percussion CM Server that can successfully run the design elements in the package. (Note: The value in this field cannot be earlier than 6.7; packaging is only supported in Percussion CM System Verison 6.7 and later.) The **CMS Maximum Version** defaults to the version of the Percussion CM Server to which the Package Builder is connected and cannot be changed.
- 5 Click on the Selection tab.
- 6 In the **Publisher Name** field, enter the name of your organization. In the **Publisher URL** field, enter the URL of your organization's Web site.
- 7 Click the Selection tab.
- 8 To add a design element, in the Design Elements, box, expand the node of type of the design element, then check the box next to the design element you want to include in your package.

To filter the list of design objects, enter a filter string in the unnamed field over the list of design objects. You can use the "?" and "*" characters as wildcards in the filter string. The "?" character matches one character, while the "*" character matches zero or more characters.


To select all displayed design elements, click the [**Select All**] button.

To clear all selections, click the [**Clear All**] button.

To expand all design object types, click the [**Expand All**] button. To collapse all design object types, click the [**Collapse All**] button.


To see only design objects that are currently selected for the descriptor, check the **Selected Only** checkbox.
- 9 Some design elements may require supporting file resources. To add a supporting file resource to the package:
 - a) Under the File Resources box, click the [**Add**] button.

Package Builder displays the File Resources dialog. Note that you can only add files that reside under the installation root of the server to which you are connected.

- b) In the CM Server **File Structure** field, browse to the directory where the files you want to add reside. Select the files. Standard multi-select options are supported.
 - c) Click the [**Add**] button to move files to the **File Resources** field.
 - d) Repeat Steps b and c to select additional files to add to the descriptor.
 - e) Click the [**OK**] button to add the files to the descriptor.
- 10** Configuration files are optional, but if you include either configuration file in the descriptor, you must include both. To add a Configuration Definition file or a Default Configuration file:
- a) Click the browse button  next to the field for the type of configuration file you want to add to the descriptor.
The Package Builder displays a browse dialog.
 - b) Browse to the directory where you store the configuration file you want to add.
 - c) Select the file you want to add.
 - d) Click [**Select**].
The Package Builder adds the selected file to the field.
- 11** Click the Dependencies tab.
- If a design element in the package depends on another design element that is included in a different package, the other package will be listed in the **Dependent Packages for Selected Elements** table. The contents of this table cannot be changed. You must install these packages before you can install packages built from the descriptor you are creating.
- 12** All other packages are listed in the **Additional Required Packages** table. Check packages in this table to add them as dependencies that must be installed before you can install packages build from the descriptor you are creating.
- 13** To save the descriptor, click the save button. (screenshot).

Building Packages

To build a package

- 1 Start the process. Two options are available.
 - On the Package List dialog, in the Menu bar, choose *Package > Build*.
 - On the Descriptor dialog, click the  button.

Regardless of the way you start the process, the Package Builder displays the Select Target Directory dialog.

- 2 Enter the directory where you want to build the package. You can use the browse button [...] to open a browse dialog you can use to choose the directory.
- 3 Click the [OK] button.

The Package Builder builds the package. When the process is complete, the Package Builder displays a completion dialog.

- 4 Click the [OK]

Testing a Package

Once you have built your package, you should test it to ensure that it is working as intended.

Install the package on a Percussion CM Server instance where the design objects in the package do not currently exist. This server may be a fresh installation or it can be a server from which a previous installation of the package has been removed. If the package you want to test has dependencies on other packages, confirm that these packages are installed on the target server before testing your package.

Confirm that the package is installed correctly and that no errors occurred during installation. Confirm that all supporting files were installed to the correct location. Review all design objects and validate that they are working as intended without configuration. For example, if your package included Content Types, confirm that you can create a new Content Item and move it through its Workflow. If your package included Templates, preview one or more Content Items using the Template. Publish the Site and confirm that the Content Items are published correctly and formatted as expected.

Modify the local configuration of each design object and confirm that the modified configurations are working as intended. You may want to try several different configurations to confirm that the design objects can be configured as you intend.

CHAPTER 4

Documenting a Package

Documenting your packages helps the package provide the most value to your customer.

At a minimum, you should provide package installation and configuration documentation for your customer's Web master. This document informs the Web master about the following aspects of the package:

- any package dependencies for the design objects in your package, including how to obtain these dependent packages and any recommended configurations to support your package
- the design objects included in the package, including their intended use and function
- the configuration options available for the solution, including any specific configurations recommended to enhance performance or to achieve a specific outcome.

You may also want to provide a package manifest for customer reference. To create a manifest, in the Package Builder, select the package for which you want to create a manifest, and in the Menu bar, choose *Export > Config Summary*.

A number of delivery options are available for this documentation, including printable formats such as the .pdf format, or online delivery through HTML pages or wiki.

You can also provide documentation to the business user by extending the Help for the Content tab of Content Explorer. The Help for this tab is delivered in Sun Microsystem's JavaHelp? format. You can extend the Content tab Help by adding a JavaHelp? Helpset to your package. When the package is installed, a link to your helpset will be added to the Content tab Help, and your Helpset will be appended to the Content tab Help. To ensure maximum usability of your Helpset, be sure to include both an index and a full-text search database.

Each JavaHelp Helpset should be developed in a unique subdirectory of `<CMServerRoot>/Docs/Rhythmyx/Custom` to ensure that the Helpset files will be delivered to the correct location when installing your package. When building your package, include all the files from your JavaHelp Helpset as File Resources for your package.

For details about JavaHelp, see <http://java.sun.com/javase/technologies/desktop/javahelp/>; in particular, see the JavaHelp Users Guide (http://java.sun.com/javase/technologies/desktop/javahelp/download_binary.html#userguide)).

NOTE: The Help for the Active Assembly interface is not extendable.

CHAPTER 5

Design Object Configuration Reference

The following types of design objects can be configured:

- Action Menus
- Content Lists
- Content Types
- Contexts
- Editions
- Keywords
- Location Schemes
- Relationships
- Sites
- Slots
- Templates

All property names are case-sensitive. Values are also case-sensitive, unless the value accepts free-form text, which is case-insensitive.

Action Menu Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

ACTION

Property Setter Class

com.percussion.rx.config.impl.PSActionSetter

Configurable Properties

| Property | Description |
|----------|---|
| label | The Action Menu Entry label, as defined in the Menu Entry label field on the General tab of the Action Menu Editor in the Percussion Workbench. |

| Property | Description | | | | | | | | | | | | | | | | | | | | |
|------------------------|--|----------|-------------|------------------------|---|-----------------------|--|---------------------|--|-----------------------|---|----------------|---|--------------------|--|--------------------|--|--------------|---|-----------------|---|
| visibility | <p>Visibility of the Action Menu Entry as defined on the Visibility tab of the Action Menu Editor in the Rhythmyx Workbench. The following configuration properties must be used:</p> <table border="1" data-bbox="553 426 1369 1472"> <thead> <tr> <th data-bbox="553 426 854 489">Property</th> <th data-bbox="854 426 1369 489">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="553 489 854 600">hideForAssignmentTypes</td> <td data-bbox="854 489 1369 600">Valid values in the default and local configurations include <i>none</i>, <i>reader</i>, <i>assignee</i>, <i>admin</i></td> </tr> <tr> <td data-bbox="553 600 854 711">hideForCheckoutStatus</td> <td data-bbox="854 600 1369 711">Valid values in the default and local configurations include <i>checked in</i>, <i>checked out</i>, <i>checked out by me</i></td> </tr> <tr> <td data-bbox="553 711 854 848">hideForContentTypes</td> <td data-bbox="854 711 1369 848">Valid values in the default and local configurations include the fully-qualified names of all Content Types defined in the system.</td> </tr> <tr> <td data-bbox="553 848 854 926">hideForFolderSecurity</td> <td data-bbox="854 848 1369 926">Valid values in the default and local configurations include <i>read</i> and <i>write</i></td> </tr> <tr> <td data-bbox="553 926 854 1037">hideForLocales</td> <td data-bbox="854 926 1369 1037">Valid values in the default and local configurations include all Locales defined in the system.</td> </tr> <tr> <td data-bbox="553 1037 854 1115">hideForObjectTypes</td> <td data-bbox="854 1037 1369 1115">Valid values in the default and local configurations include <i>folder</i> and <i>item</i>.</td> </tr> <tr> <td data-bbox="553 1115 854 1251">hideForPublishable</td> <td data-bbox="854 1115 1369 1251">Valid values in the default and local configurations include <i>archive</i>, <i>ignore</i>, <i>publish</i>, and <i>unpublish</i> and any custom choices added to the Publishable keyword.</td> </tr> <tr> <td data-bbox="553 1251 854 1362">hideForRoles</td> <td data-bbox="854 1251 1369 1362">Valid values in the default and local configurations include all Roles defined in the system.</td> </tr> <tr> <td data-bbox="553 1362 854 1472">hideForWorkflow</td> <td data-bbox="854 1362 1369 1472">Valid values in the default and local configurations include all Workflows defined in the system.</td> </tr> </tbody> </table> <p>Each configuration property node contains a list of contexts for which the Action Menu will be hidden. Configuration only allows hiding. Menus are visible by default unless specifically hidden by a configuration.</p> | Property | Description | hideForAssignmentTypes | Valid values in the default and local configurations include <i>none</i> , <i>reader</i> , <i>assignee</i> , <i>admin</i> | hideForCheckoutStatus | Valid values in the default and local configurations include <i>checked in</i> , <i>checked out</i> , <i>checked out by me</i> | hideForContentTypes | Valid values in the default and local configurations include the fully-qualified names of all Content Types defined in the system. | hideForFolderSecurity | Valid values in the default and local configurations include <i>read</i> and <i>write</i> | hideForLocales | Valid values in the default and local configurations include all Locales defined in the system. | hideForObjectTypes | Valid values in the default and local configurations include <i>folder</i> and <i>item</i> . | hideForPublishable | Valid values in the default and local configurations include <i>archive</i> , <i>ignore</i> , <i>publish</i> , and <i>unpublish</i> and any custom choices added to the Publishable keyword. | hideForRoles | Valid values in the default and local configurations include all Roles defined in the system. | hideForWorkflow | Valid values in the default and local configurations include all Workflows defined in the system. |
| Property | Description | | | | | | | | | | | | | | | | | | | | |
| hideForAssignmentTypes | Valid values in the default and local configurations include <i>none</i> , <i>reader</i> , <i>assignee</i> , <i>admin</i> | | | | | | | | | | | | | | | | | | | | |
| hideForCheckoutStatus | Valid values in the default and local configurations include <i>checked in</i> , <i>checked out</i> , <i>checked out by me</i> | | | | | | | | | | | | | | | | | | | | |
| hideForContentTypes | Valid values in the default and local configurations include the fully-qualified names of all Content Types defined in the system. | | | | | | | | | | | | | | | | | | | | |
| hideForFolderSecurity | Valid values in the default and local configurations include <i>read</i> and <i>write</i> | | | | | | | | | | | | | | | | | | | | |
| hideForLocales | Valid values in the default and local configurations include all Locales defined in the system. | | | | | | | | | | | | | | | | | | | | |
| hideForObjectTypes | Valid values in the default and local configurations include <i>folder</i> and <i>item</i> . | | | | | | | | | | | | | | | | | | | | |
| hideForPublishable | Valid values in the default and local configurations include <i>archive</i> , <i>ignore</i> , <i>publish</i> , and <i>unpublish</i> and any custom choices added to the Publishable keyword. | | | | | | | | | | | | | | | | | | | | |
| hideForRoles | Valid values in the default and local configurations include all Roles defined in the system. | | | | | | | | | | | | | | | | | | | | |
| hideForWorkflow | Valid values in the default and local configurations include all Workflows defined in the system. | | | | | | | | | | | | | | | | | | | | |
| url | The Action Menu Entry URL, as defined in the URL field on the Command tab of the Action Menu Editor in the Percussion Workbench. | | | | | | | | | | | | | | | | | | | | |
| urlParams | Parameters of the Action Menu Entry URL as defined in the Parameters table on the Command tab of the Action Menu Editor in the Percussion Workbench. In the configuration definition file, the configurable properties are defined in a map of entries. | | | | | | | | | | | | | | | | | | | | |

Example Configuration Bean

```
<bean id="Quick_Edit"
class="com.percussion.rx.config.impl.PSubjectConfigHandler">
  <property name="name" value="Quick_Edit"/>
  <property name="type" value="ACTION"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PActionSetter">
      <property name="properties">
        <map>
          <entry key="label"
value="${perc.ActionMenuExample.ActionLabel}"/>
          <entry key="description"
value="${perc.ActionMenuExample.ActionDescription}"/>
          <entry key="URL"
value="${perc.ActionMenuExample.ActionUrl}"/>
          <entry key="urlParams">
            <!-- Like other collections, all params must be
supplied, they replace current settings -->
            <!-- Note in this example only the Workflow Action
parameter (WFAction) is specified as configurable. -->
            <map>
              <entry key="sys_contentid"
value="${sys_contentid}"/>
              <entry key="sys_revision"
value="${sys_revision}"/>
              <entry key="sys_view" value="sys_All"/>
              <entry key="WFAction"
value="${perc.ActionMenuExample.DynamicParam}"/>
            </map>
          </entry>
          <entry key="visibility">
            <!-- Each visibility context is optional. The value in
the default and local configuration files is a List of values
appropriate to the context as defined in the table above. -->
            <map>
              <entry key="Assignment Types"
value="${perc.ActionMenuExample.HideForAssignmentTypes}"/>
              <entry key="Checkout Status"
value="${perc.ActionMenuExample.HideForCheckoutStatus}"/>
              <entry key="Content Types"
value="${perc.ActionMenuExample.HideForContentTypes}"/>
              <entry key="Folder Security"
value="${perc.ActionMenuExample.HideForFolderSecurity}"/>
              <entry key="Locales"
value="${perc.ActionMenuExample.HideForLocales}"/>
              <entry key="Object Types"
value="${perc.ActionMenuExample.HideForObjectTypes}"/>
              <entry key="Publishable"
value="${perc.ActionMenuExample.HideForPublishable}"/>
              <entry key="Roles"
value="${perc.ActionMenuExample.HideForRoles}"/>
              <entry key="Workflow"
value="${perc.ActionMenuExample.HideForWorkflow}"/>
            </map>
          </entry>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

```

        </entry>
    </map>
</property>
</bean>
</property>
</bean>

```

Example Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="
Percussion Software" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="ActionMenuExample">
        <property name="ActionLabel" value="Quick Edit"/>
        <property name="ActionDescription" value="Transitions the
current item into a Quick Edit state, check it out and opens it in edit
mode."/>
        <property name="ActionUrl"
value="../sys_action/transitcheckoutedit.xml"/>
        <property name="DynamicParam" value="Quick Edit"/>
        <property name="HideForAssignmentTypes">
            <pvalues>
                <pvalue>none</pvalue>
                <pvalue>reader</pvalue>
            </pvalues>
        </property>
        <property name="HideForCheckoutStatus">
            <pvalues>
                <pvalue>checked out</pvalue>
            </pvalues>
        </property>
        <property name="HideForContentTypes">
            <pvalues>
            </pvalues>
        </property>
        <property name="HideForFolderSecurity">
            <pvalues>
            </pvalues>
        </property>
        <property name="HideForLocales">
            <pvalues>
            </pvalues>
        </property>
        <property name="HideForObjectTypes">
            <pvalues>
                <pvalue>folder</pvalue>
            </pvalues>
        </property>
        <property name="HideForPublishable">
            <pvalues>
                <pvalue>archive</pvalue>
                <pvalue>ignore</pvalue>
                <pvalue>unpublish</pvalue>
            </pvalues>
        </property>
    </SolutionConfig>
</SolutionConfigurations>

```

```
    </property>
    <property name="HideForRoles">
      <pvalues>
        </pvalues>
    </property>
    <property name="HideForWorkflow">
      <pvalues>
        </pvalues>
    </property>
  </SolutionConfig>
</SolutionConfigurations>
```

Content List Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

CONTENT_LIST

Property Setter Class

com.percussion.rx.config.impl.PSContentListSetter

Configurable Properties

| Property | Description |
|-----------------|---|
| deliveryType | Name of the Delivery Type as specified in the Delivery Type drop list on the Content List Editor in the Publishing Design tab of Content Explorer. |
| expander | Name of the Template Expander as specified in the Template Expander drop list on the Content List Editor in the Publishing Design tab of Content Explorer. |
| expanderParams | Map of parameters for the Template Expander specified in the expander property, as specified in the fields associated with the Template Expander on the Content List Editor in the Publishing Design tab of Content Explorer. |
| filter | Name of the Item Filter as specified in the Item Filter drop list on the Content List Editor in the Publishing Design tab of Content Explorer. |
| generator | Name of the Generator as specified in the Generator drop list on the Content List Editor in the Publishing Design tab of Content Explorer. |
| generatorParams | Map of parameters for the Generator specified in the generator property, as specified in the fields associated with the Generator on the Content List Editor in the Publishing Design tab of Content Explorer. |

Example Configuration Bean

```
<bean id="CI_FullBinary"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="rffCiFullBinary"/>
  <property name="type" value="CONTENT_LIST"/>
  <property name="propertySetters">
    <bean
class="com.percussion.rx.config.impl.PSContentListSetter">
      <property name="properties">
        <map>
          <entry key="deliveryType"
value="\${perc.ContentListExample.DeliveryType}"/>

```

```
        <entry key="filter"
value="{perc.ContentListExample.Filter}"/>
        <entry key="generator"
value="{perc.ContentListExample.Fenerator}"/>
        <entry key="generatorParams"
value="{perc.ContentListExample.FeneratorParams}"/>
        <entry key="expander"
value="{perc.ContentListExample.Expander}"/>
        <entry key="expanderParams"
value="{perc.ContentListExample.ExpanderParams}"/>
    </map>
</property>
</bean>
</property>
</bean>
```

Example Default Configuration

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="ContentListExample">
        <property name="DeliveryType" value="filesystem"/>
        <property name="Filter" value="public"/>
        <property name="Generator"
value="Java/global/percussion/system/sys_SearchGenerator"/>
        <propertySet name="GeneratorParams">
            <property name="query" value="select rx:sys_contentid,
rx:sys_folderid from rx:rfffile,rx:rffimage,rx:rffnavimage where
jcr:path like '//Sites/CorporateInvestments%'"/>
        </propertySet>
        <property name="Expander"
value="Java/global/percussion/system/sys_SiteTemplateExpander"/>
        <propertySet name="ExpanderParams">
            </propertySet>
        </SolutionConfig>
</SolutionConfigurations>
```

Content Type Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

NODEDEF

Property Setter Class

com.percussion.rx.config.impl.PSContentTypeSetter

Configurable Properties

| Property | Description |
|-----------------|--|
| label | .The Content Type label, as defined in the Label field on the Properties tab of the Content Type Editor in the Percussion Workbench. |
| defaultWorkflow | Name of the Default Workflow, as defined in the Default Workflow field on the Properties tab of the Content Type Editor in the Percussion Workbench. The Workflow specified must be one of those defined in the workflow property. |
| iconValue | Value defining the icon representing the Content Type, as defined in the Content Type Icon drop list on the Properties tab of the Content Type Editor in the Percussion Workbench.. Valid values are null, indicating that no icon represents the Content Type; a file name, specifying the icon graphic file to represent the Content Type (the path to the file should be relative to the Percussion CM Server installation root). |
| fields | A set of fields to configure. Existing fields can be configured, and system or shared fields can be added and configured. New local fields CANNOT be added to a Content Type as configurations. Each entry consists of a field and a list of configurations of configurable properties (see the table below). |
| template | List of Templates associated with the Content Type, as defined in the Allowed Templates node of the Content Type on the Content Tab in the Percussion Workbench.. |
| workflows | List of Workflows associated with the Content Type, as defined in the Allowed Workflows node of the Content Type on the Content Tab in the Percussion Workbench. |

Configurable Properties of Content Type Fields

Content Type Field Property Setter Class

com.percussion.rx.config.impl.PSContentTypeSetter

| Property | Description |
|-------------------|--|
| label | .Field label as defined in the Label field on the Field Properties dialog of the Percussion Workbench. |
| default | Default value of the field as specified in the Default Value field on the Field Properties dialog of the Percussion Workbench. Only literal values can be configured. |
| sequence | The position of the field in relation to ther other fields defined for the Content Type. |
| controlParameters | Parameters for the control used for the field as defined in the parameters table on the Control Properties dialog in the Percussion Workbench. |
| showInPreview | Value of the Show in Preview flag on the Field Properties dialog of the Percussion Workbench. Valid values are true, false, yes, and no. |
| userSearchable | Value of the Allow this field to be searched flag on the Field Properties dialog of the Percussion Workbench. Valid values are true, false, yes, and no. |
| required | Value of the Required flag on the Field Validation dialog of the Percussion Workbench. Valid values are true, false, yes, and no. Only valid on a local field, or a system or shared field that has no other validation defined. |

Example Configuration Bean

```
<bean id="rffBrief"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="rffBrief"/>
  <property name="type" value="NODEDEF"/>
  <property name="propertySetters">
    <list>
      <bean
class="com.percussion.rx.config.impl.PSContentTypeSetter">
        <property name="properties">
          <map>
            <entry key="label"
value="\${perc.ContentTypeExample.CT.Label}"/>
            <entry key="iconValue"
value="\${perc.ContentTypeExample.CT.IconValue}"/>
            <entry key="defaultWorkflow"
value="\${perc.ContentTypeExample.CT.DefaultWorkflow}"/>
            <entry key="templates"
value="\${perc.ContentTypeExample.CT.Templates}"/>
            <entry key="workflows"
value="\${perc.ContentTypeExample.CT.Workflows}"/>
            <entry key="fields"
value="\${perc.ContentTypeExample.CT.Fields}"/>
          </map>
        </property>
      </bean>
    </list>
  </property>
</bean>
```

```

        <bean
class="com.percussion.rx.config.impl.PSContentTypeFieldSetter">
    <property name="fieldName" value="placeholder"/>
    <property name="properties">
        <map>
            <entry key="default"
value="\${perc.contentTypeExample.FD.Default}"/>
            <entry key="required"
value="\${perc.contentTypeExample.FD.Required}"/>
            <entry key="sequence"
value="\${perc.contentTypeExample.FD.Sequence}"/>
            <entry key="showInPreview"
value="\${perc.contentTypeExample.FD.ShowInPreview}"/>
            <entry key="userSearchable"
value="\${perc.contentTypeExample.FD.UserSearchable}"/>
        </map>
    </property>
</bean>
        <bean
class="com.percussion.rx.config.impl.PSContentTypeFieldSetter">
    <property name="fieldName" value="callout"/>
    <property name="properties">
        <map>
            <entry key="label"
value="\${perc.contentTypeExample.Callout.Label}"/>
            <entry key="controlParameters"
value="\${perc.contentTypeExample.Callout.ControlParameters}"/>
        </map>
    </property>
</bean>
</list>
</property>
</bean>

```

Example Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="contentTypeExample">
        <propertySet name="CT">
            <property name="Label" value="Brief"/>
            <property name="IconValue" value="rffBrief.gif"/>
            <property name="DefaultWorkflow" value="Simple Workflow"/>
            <property name="Templates">
                <pvalues>
                    <pvalue>rffSnCallout</pvalue>
                </pvalues>
            </property>
            <property name="workflows">
                <pvalues>
                    <pvalue>Simple Workflow</pvalue>
                    <pvalue>Standard Workflow</pvalue>
                </pvalues>
            </property>

```

```
<property name="fields">
  <propertySet name="placeholder">
    <property name="default" value=""/>
    <property name="required" value="false"/>
    <property name="sequence" value="7"/>
    <property name="showInPreview" value="true"/>
    <property name="userSearchable" value="false"/>
  </propertySet>
</property>
</propertySet>
<propertySet name="FD">
  <property name="Default" value=""/>
  <property name="Required" value="false"/>
  <property name="Sequence" value="7"/>
  <property name="ShowInPreview" value="true"/>
  <property name="UserSearchable" value="false"/>
</propertySet>
<propertySet name="Callout">
  <property name="Label" value="Callout"/>
  <propertySet name="ControlParameters"/>
</propertySet>
</SolutionConfig>
</SolutionConfigurations>
```

Context Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

CONTEXT

Property Setter Class

com.percussion.rx.config.impl.PSContextSetter

Configurable Properties

| Property | Description |
|-----------------------|--|
| defaultLocationScheme | Name of the default Location Scheme for the Context, as specified in the Default Scheme drop list in the Context editor. |

Example Configuration Bean

```
<bean id="CI_Full"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="Site_Folder_Assembly"/>
  <property name="type" value="CONTEXT"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSContextSetter">
      <property name="properties">
        <map>
          <entry key="defaultLocationScheme"
value="{perc.ContextExample.DefaultLocationScheme}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

Example Default Configuration

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="ContextExample">
    <property name="DefaultLocationScheme" value="Generic"/>
  </SolutionConfig>
</SolutionConfigurations>
```

Edition Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

EDITION

Property Setter Class

com.percussion.rx.config.impl.PSEditionSetter

Configurable Properties

| Property | Description |
|-----------|--|
| priority | Priority of the Edition, as specified on the Priority drop list on the Edition editor. Valid values are LOW, LOWEST, MEDIUM, HIGH, or HIGHEST |
| preTasks | <p>List of tasks to execute before running the Edition, specified as a list of <propertySet> elements. Tasks are executed in the order specified. The configured list of tasks overrides the list of tasks defined in the Edition. (Thus, any tasks you want to preserve must be specified in the configuration.)</p> <p>To remove all pre-tasks from the Edition, define the list as a single empty <propertySet> node: <propertySet/>.</p> <p>See below for task configuration properties.</p> |
| postTasks | <p>List of tasks to execute after running the Edition, specified as a list of <propertySet> elements. Tasks are executed in the order specified. The configured list of tasks overrides the list of tasks defined in the Edition. (Thus, any tasks you want to preserve must be specified in the configuration.)</p> <p>To remove all post-tasks from the Edition, define the list as a single empty <propertySet> node: <propertySet/>.</p> <p>See below for task configuration properties.</p> |

Both pre-tasks and post-tasks use the following configuration properties:

| Property | Description |
|-------------------|---|
| extensionName | Fully-qualified name of the task extension. |
| extensionParams | List of task extension parameters as name/value pairs (parameter name/parameter value). |
| continueOnFailure | Continue on failure flag. Valid values are true, false, yes, and no. |

Example Configuration Bean

```

<bean id="CI_Full"
class="com.percussion.rx.config.impl.PEObjectConfigHandler">
  <property name="name" value="CI_Full"/>
  <property name="type" value="EDITION"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSEditionSetter">
      <property name="properties">
        <map>
          <entry key="priority"
value="\${perc.ExampleEdition.Priority}"/>
          <entry key="preTasks"
value="\${perc.ExampleEdition.PreTasks}"/>
          <entry key="postTasks"
value="\${perc.ExampleEdition.PostTasks}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>

```

Example Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="ExampleEdition">
    <property name="Priority" value="LOWEST"/>
    <property name="PreTasks">
      <propertySet>
        <property name="extensionName"
value="Java/global/percussion/task/sys_editionCommandTask" />
        <property name="extensionParams">
          <pvalues>
            <pair value1="command" value2="copy
c:\Rhythmyx\web_resources\enterprise_investments c:\Program
Files\Apache Group\Apache\htdocs\EI_Home" />
          </pvalues>
        </property>
        <property name="continueOnFailure" value="false" />
      </propertySet>
    </property>
    <property name="PostTasks">
      <propertySet>
        <property name="extensionName"
value="Java/global/percussion/task/sys_editionCommandTask" />
        <property name="extensionParams">
          <pvalues>
            <pair value1="command" value2="C:\Program
Files\Mozilla Firefox\Firefox.exe http://validator.w3.org/checklink?
uri=www.enterpriseinvestments.com&hide_type;=all&depth;=&check;=Check"
/>
          </pvalues>
        </property>
      </propertySet>
    </property>
  </SolutionConfig>

```

```
        <property name="continueOnFailure" value="false" />
    </propertySet>
</property>
</SolutionConfig>
</SolutionConfigurations>
```

Example Default Configuration clearing pre-tasks and post-tasks

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="exampleedition">
    <property name="Priority" value="LOWEST"/>
    <property name="PreTasks">
      <propertySet/>
    </property>
    <property name="PostTasks">
      <propertySet/
    </property>
  </SolutionConfig>
</SolutionConfigurations>
```


Keyword Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

KEYWORD_DEF

Property Setter Class

com.percussion.rx.config.impl.PSKeywordSetter

Configurable Properties

| Property | Description |
|-------------|--|
| choicePairs | The Keyword choices, defined as a set of name/value pairs. The configured set of values replaces the existing set of values. Therefore, any existing values that you want to preserve must be included in the configuration. |

Example Configuration Bean

```
<bean id="perc.BodyMarkup"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="Body_Markup"/>
  <property name="type" value="KEYWORD_DEF"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSKeywordSetter">
      <property name="properties">
        <map>
          <entry key="choicePairs"
value="${perc.KeywordExample.BodyMarkupChoices}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

Example Default Configuration

```
<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="KeywordExample.">
    <property name="BodyMarkupDescription" value="Body Markup
Lookups"/>
    <property name="BodyMarkupChoices">
```

<!--"value1" in the value pair is the Keyword Label displayed in the user interface; "value2" is the value stored in the Repository for the Keyword -->

```
<pvalues>
  <pair value1="Text" value2="1"/>
  <pair value1="HTML" value2="2"/>
</pvalues>
</property>
</SolutionConfig>
</SolutionConfigurations>
```

Location Scheme Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSLocationSchemeConfigHandler

Handler Properties

| Property | Description |
|-------------|--|
| contexts | List of existing Contexts to which the Location Scheme configurations will be applied. (Contexts cannot be added through configuration.) The value of this property can be a list of Context names (applies the configurations to the specified Contexts), a simple wildcard ("*"); applies the configurations to all Contexts), a string with one or more wild cards ("*Site*" will match both "Publish_Site" and "Site_Folder_Assembly"; applies the configurations to all Contexts whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}); applies the configurations to the Contexts specified in the replacement variable in the default or local configurations). Property is required. |
| name | Name of a Location Scheme. Can be either an existing Location Scheme or a new one. The value name can be hard-coded, which applies the configuration only the specified Location Scheme, or a replacement variable (\${perc.solution.replacementvariable}), which applies the configurations to the Location Scheme specified in the replacement variable in the default or local configurations. This property is required. The specified name must be unique within each Context. If a Context specified in the context property contains multiple instances of a name specified in this property, the server will return an error when attempting to apply the configuration. |
| contentType | The name of a Content Type, as displayed on the Content Type editor in the Percussion Workbench. Required if adding a new Location Scheme; otherwise optional. Any specified combination values of the context, contentType, and template properties must be unique in the system. The specified Content Type must already be associated with the Template specified in the template property. |
| template | The name of a Template, as displayed on the Template editor in the Percussion Workbench. Required if adding a new Location Scheme; otherwise optional. Any specified combination values of the context, contentType, and template properties must be unique in the system. The specified Template must already be associated with the Content Type specified in the contentType property. |

Type

LOCATION_SCHEME (NOTE: When using PSLocationSchemeConfigHandler, the type property is optional. If it is included, the value MUST be LOCATION_SCHEME.)

Property Setter Class

com.percussion.rx.config.impl.PSContextSetter

Configurable Properties

| Property | Description |
|-----------------|---|
| expression | <p>The JEXL expression of the Location Scheme. When this property is configured the system assumes that the sys_JexlAssemblyLocation Location Scheme Generator is being used.</p> <p>When creating a new Location Scheme through configuration, either this property or the generator property must be used.</p> <p>This property cannot be used if the generator property is being used.</p> |
| generator | <p>The name of the Location Scheme generator extension. The value must be the short name of the extension, not the fully-qualified extension class name.</p> <p>When creating a new Location Scheme through configuration, either this property or the expression property must be used.</p> <p>This property cannot be used if the expression property is being used.</p> |
| generatorParams | <p>Optional property used with the generator property to specify the parameters of the Location Scheme generator. Value is a list of name/value pairs, in the order that the parameters are defined for the generator extension. For extensions provided by Percussion Software, see the "Extension Reference" in the Percussion Technical Reference Manual. This information should also included in the package documentation provided to customers. Only String datatype parameters are supported. Backend-column parameters are not supported.</p> <p>If used when configuring an existing Location Scheme, the parameter values specified override the existing parameters of the Location Scheme generator.</p> <p>This property cannot be used if the expression property is being used.</p> |

Example Configuration

Modifying a specified existing Context, using XML extension

Configuration Bean

```
<bean id="LocationSchemeOneContextXSLGenerator"
class="com.percussion.rx.config.impl.PSLocationSchemeConfigHandler">
  <property name="name" value="CI_Generic"/>
  <property name="contexts" value="Publish"/>
  <property name="propertySetters">
    <bean
class="com.percussion.rx.config.impl.PSLocationSchemeSetter">
      <property name="properties">
        <map>
          <entry key="generator"
value="${perc.LocationSchemeExample.Generator}"/>
          <entry key="generatorParams"
value="${perc.LocationSchemeExample.GeneratorParams}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

```

        </property>
    </bean>
</property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="LocationSchemeExample">
        <property name="Generator"
value="sys_casDefaultAssemblyLocation"/>
        <property name="GeneratorParams">
            <pvalues>
                <pair value1="root" value2="/home/support"/>
                <pair value1="path" value2="publish"/>
                <pair value1="suffix" value2="html"/>
            </pvalues>
        </property>
    </SolutionConfig>
</SolutionConfigurations>

```

Example Configuration

Add new JEXL scheme to user-specified set of Contexts; this configuration is recommended for providing the most flexibility to the customer

Configuration Bean

```

<bean id="NewJEXLSchemeUserSpecifiedContexts"
class="com.percussion.rx.config.impl.PSLocationSchemeConfigHandler">
    <property name="name" value="BinaryLocations"/>
    <property name="contexts"
value="\${perc.LocationSchemeExample.ContextSet}"/>
    <property name="propertySetters">
        <bean
class="com.percussion.rx.config.impl.PSLocationSchemeSetter">
            <property name="properties">
                <map>
                    <entry key="expression"
value="\${perc.LocationSchemeExample.Expression}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="LocationSchemeExample">
        <property name="ContextSet">
            <pvalues>

```

```
        <pvalue>Publish</pvalue>
        <pvalue>Site_Folder_Assembly</pvalue>
        <pvalue>Staging</pvalue>
    </pvalues>
</property>
<property name="Expression">
    <pvalue>'$sys.variables.rxs_urlroot + $sys.pub_path +
$sys.template.prefix + 'item' +
$sys.item.getProperty('rx:sys_contentid').String +
$rx.location.getFirstDefined($sys.item, 'rx:sys_suffix',
'.gif')'</pvalue>
</property>
</SolutionConfig>
</SolutionConfigurations>
```

Example Configuration

Add new JEXL Location Scheme to wildcard-defined set of Contexts; the Location Scheme will be added to all Contexts that match the wildcard [in this case "pub", matches all Contexts that include the string "pub"]. This approach is best suited to an internal deployment package

Configuration Bean

```
<bean id="NewJEXLSchemeUserSpecifiedContexts"
class="com.percussion.rx.config.impl.PSLocationSchemeConfigHandler">
    <property name="name" value="BinaryLocations"/>
    <property name="contexts" value="*pub*"/>
    <property name="propertySetters">
        <bean
class="com.percussion.rx.config.impl.PSLocationSchemeSetter">
            <property name="properties">
                <map>
                    <entry key="expression"
value="$${perc.LocationSchemeExample.Expression}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>
```

Default Configuration

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="LocationSchemeExample">
        <property name="Expression">
            <pvalue>'$sys.variables.rxs_urlroot + $sys.pub_path +
$sys.template.prefix + 'item' +
$sys.item.getProperty('rx:sys_contentid').String +
$rx.location.getFirstDefined($sys.item, 'rx:sys_suffix',
'.gif')'</pvalue>
        </property>
    </SolutionConfig>
</SolutionConfigurations>
```

Relationship Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

RELATIONSHIP_CONFIGNAME

Property Setter Class

com.percussion.rx.config.impl.PSRelationshipSetter

Configurable Properties

| Property | Description |
|----------------|--|
| shallowCloning | <p>Configuration for shallow cloning as defined on the Cloning tab of the Relationship Editor in the Percussion Workbench.</p> <p>In the configuration files, this property consists of a set of two properties:</p> <ul style="list-style-type: none"> ▪ enabled: value values are TRUE or FALSE; if the value is FALSE, the conditionproperty is ignored ▪ condition, which defines the set of conditions in which the system will make a shallow clone of the object. Each condition is defined in a <propertySet>. The first property in the set specifies the type of condition. Two options are available for the type ▪ Conditional: the condition is processed by a conditional rule. See ""Specifying Conditions" below for details. ▪ Extension: the condition is processed by a Percussion CMS extension. See "Specifying Extensions" below for details about specifying an extension. ▪ If multiple conditions are specified, include a boolean property in each property. Valid values for the boolean property are AND and OR. Standard precedence rules apply. |
| deepCloning | <p>Configuration for deep cloning as defined on the Cloning tab of the Relationship Editor in the Percussion Workbench.</p> <p>Deep cloning is configured in the same manner as shallow cloning.</p> |

| Property | Description | | | | | | | | | | |
|---------------------|---|----------|-------------|-----------|---|-----------|--|---------------------|--|-----------|---|
| fieldOverrides | Configurations of field overrides when cloning a Content Item. The configuration for each field consists of a property set including the following properties <table border="1"> <thead> <tr> <th>Property</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>fieldName</td> <td>specifies the field whose value will be modified during cloning</td> </tr> <tr> <td>extension</td> <td>name of the extension that will process the override</td> </tr> <tr> <td>extensionProperties</td> <td>optional; set of properties of the extension</td> </tr> <tr> <td>condition</td> <td>optional; specifies one or more conditions for field override processing; see "Specifying Conditions" below for details about defining conditions</td> </tr> </tbody> </table> | Property | Description | fieldName | specifies the field whose value will be modified during cloning | extension | name of the extension that will process the override | extensionProperties | optional; set of properties of the extension | condition | optional; specifies one or more conditions for field override processing; see "Specifying Conditions" below for details about defining conditions |
| Property | Description | | | | | | | | | | |
| fieldName | specifies the field whose value will be modified during cloning | | | | | | | | | | |
| extension | name of the extension that will process the override | | | | | | | | | | |
| extensionProperties | optional; set of properties of the extension | | | | | | | | | | |
| condition | optional; specifies one or more conditions for field override processing; see "Specifying Conditions" below for details about defining conditions | | | | | | | | | | |
| effects | List of effect configurations for the Relationship as defined on the Effects tab of the Relationship Editor in the Percussion Workbench. This is a distributed property; multiple Effects can be configured within one package configuration. See below for details about effect configuration beans. | | | | | | | | | | |

Specifying Conditions

Each condition is defined by a set of rules defined under the <rules> property. The <rules> must include at least the following properties

- variable1 The value of the variable1 property is a replacement value.
- operator properties. The value of the operator property must be one of the following:
 - =
 - =
 - <
 - <=
 - >
 - >=
 - IS NULL
 - IS NOT NULL
 - BETWEEN
 - NOT BETWEEN
 - IN
 - NOT IN
 - LIKE
 - NOT LIKE

Depending on the operator, a variable2 property may be required; like variable1, the value is a replacement value.

If multiple rules are being defined, a boolean property is required; valid values for the boolean property are AND and OR.

Specifying Extensions

Each extension is defined in a `<propertySet>`. Use the name property to specify the name of the extension. Use the extensionParams property to specify the extension parameters. The extensionParams property contains a `<propertySet>`. Within the `<propertySet>`, the value of the name attribute of each property specifies the name of the extension parameter, while the value of the value attribute specifies the parameter value.

Effect Configuration

Property Setter Class: `com.percussion.rx.config.impl.PSRelationshipConfigEffectSetter`

| Property | Description |
|------------------|---|
| executionContext | Condition in which the Effect will be invoked. Valid values include <ul style="list-style-type: none"> ▪ Pre-Construction ▪ Pre-Checkin ▪ Pre-Clone ▪ Pre-Destruction ▪ Pre-Update ▪ Pre-Workflow ▪ Post-Checkout ▪ Post-Workflow |
| direction | When to trigger the Effect. Valid values include UP, DOWN, and EITHER. |
| extension | Name of the Effect extension |
| extensionParams | Effect extension parameters. See "Specifying Extensions" for details about defining an extension |
| condition | Condition that determines whether theEffect will be processed. |

Example Configuration Bean

```
<bean id="Translation-Mandatory"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="Translation-Mandatory" />
  <property name="type" value="RELATIONSHIP_CONFIGNAME" />
  <property name="propertySetters">
    <list>
      <bean
class="com.percussion.rx.config.impl.PSRelationshipConfigSetter">
        <property name="properties">
          <map>
```

```

        <entry key="deepCloning"
value="\${perc.SystemObjects.Translation-Mandatory.deepCloning}"/>
        <entry key="shallowCloning"
value="\${perc.SystemObjects.Translation-Mandatory.shallowCloning}"/>
        <entry key="fieldOverrides"
value="\${perc.SystemObjects.Translation-Mandatory.fieldOverrides}"/>
    </map>
</property>
</bean>
<bean
class="com.percussion.rx.config.impl.PSRelationshipConfigEffectSetter">
    <property name="properties">
        <map>
            <entry key="effects"
value="\${perc.SystemObjects.Translation-Mandatory.effects}"/>
        </map>
    </property>
</bean>
</list>
</property>
</bean>

```

Example Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software, Inc." type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="SystemObjects">
        <propertySet name="Translation-Mandatory.deepCloning">
            <property name="enabled" value="true"/>
            <property name="condition">
                <propertySet>
                    <property name="type" value="Conditional"/>
                    <property name="rules">
                        <propertySet>
                            <property name="variable1"
value="PSXOriginatingRelationshipProperty/category"/>
                            <property name="variable2"
value="rs_promotable"/>
                            <property name="operator" value="="/>
                            <property name="boolean" value="AND"/>
                        </propertySet>
                    </propertySet>
                </propertySet>
            </propertySet>
        </propertySet>
        <propertySet name="Translation-Mandatory.shallowCloning">
            <property name="enabled" value="false"/>
        </propertySet>
        <property name="Translation-Mandatory.fieldOverrides">
            <propertySet>
                <property name="fieldName" value="sys_title"/>
                <property name="extension"
value="Java/global/percussion/relationship/sys_CloneTitle"/>
                <property name="extensionParams">

```

```

        <pvalues>
            <pvalue>[{0}] Copy of {1}</pvalue>
            <pvalue>PSXSingleHTMLParameter/sys_lang</pvalue>
        </pvalues>
    <pvalue>PSXContentItemStatus/CONTENTSTATUS.TITLE</pvalue>
    </property>
</propertySet>
<propertySet>
    <property name="fieldName" value="sys_communityid"/>
    <property name="extension"
value="Java/global/percussion/cms/sys_cloneOverrideField"/>
    <property name="extensionParams">
        <pvalues>
            <pvalue>
../sys_trFieldOverride/TranslationFieldOverride.xml
            </pvalue>
            <pvalue>CommunityId</pvalue>
            <pvalue>sys_contentid</pvalue>
        </pvalues>
    </property>
</propertySet>
<propertySet>
    <property name="fieldName" value="sys_workflowid"/>
    <property name="extension"
value="Java/global/percussion/cms/sys_cloneOverrideField"/>
    <property name="extensionParams">
        <pvalues>
            <pvalue>
../sys_trFieldOverride/TranslationFieldOverride.xml
            </pvalue>
            <pvalue>WorkflowId</pvalue>
            <pvalue>sys_contentid</pvalue>
        </pvalues>
    </property>
</propertySet>
<propertySet>
    <property name="fieldName" value="sys_lang"/>
    <property name="extension"
value="Java/global/percussion/generic/sys_Literal"/>
    <property name="extensionParams">
        <pvalues>
            <pvalue>PSXSingleHTMLParameter/sys_lang</pvalue>
        </pvalues>
    </property>
</propertySet>
</property>

```



```
        <property name="extension"  
value="Java/global/percussion/relationship/effect/sys_AttachTranslatedFo  
lder"/>  
    </propertySet>  
  </property>  
</SolutionConfig>  
</SolutionConfigurations>
```

Site Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSSiteConfigHandler

Handler Properties

| Property | Description |
|----------|--|
| name | Used to specify setter properties for a single Site. Specifies the Site to which the specified setter properties will be applied. A Site configuration bean can use either the name property or the names property, but not both. They are mutually exclusive. |
| names | Used to specify setter properties for multiple Sites. Specifies the Sites to which the specified setter properties will be applied. The value of this property can be a list of Site names (applies the configurations to the specified Sites), a simple wildcard ("*"; applies the configurations to all Sites), a string with one or more wild cards ("*Invest*" will match both EnterpriseInvestments and CorporateInvestments; applies the configurations to all Sites whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}; applies the configurations to the Sites specified in the replacement variable in the default or local configurations). A Site configuration bean can use either the name property or the names property, but not both. They are mutually exclusive. |

Type

SITE (NOTE: When using PSSiteConfigHandler, the Type property is optional. If included, the value MUST be SITE.)

Property Setter Class

com.percussion.rx.config.impl.PSSimpleSitetSetter

Configurable Properties

| Property | Description |
|----------------|--|
| siteFolderPath | The path to the root Folder of the Site under the //Sites node in Content Explorer, as specified in the Rhythmyx Path field in the Site editor. |
| publishedPath | The path to the location where the published HTML files will be published, as defined in the Published Path field in the Site editor. |
| publishedUrl | The URL of the published Site, as defined in the Published URL field in the Site editor. |
| globalTemplate | Name of the default Global Template used for the Site, as specified in the Global Template drop list on the Site editor. |

| Property | Description |
|-------------------|---|
| unpublishFlags | Flag indicating Content Items on the Site should be unpublished, as specified in the Unpublish Flags field on the Site editor. |
| ftpAddress | IP address of the FTP server used to deliver published output to the Site, as specified in the FTP Server IP Address field on the Site editor. |
| ftpPort | Port of the FTP server used to deliver published output to the Site, as specified in the FTP Server Port field on the Site editor. |
| ftpUser | User name of the account used to log in to the FTP server, as specified in the Username field on the Site editor. |
| ftpPassword | Password of the account used to log in to the FTP server, as specified in the Password field on the Site editor. |
| allowedNamespaces | Comma-separated list of XML namespaces to preserve when cleaning up namespaces in rich-text fields when assembling content for the Site, as specified in the Allowed Namespaces field on the Site editor. Entries in this list should have a corresponding entry in the file <Rhythmyxroot> /rxconfig/server/stylesheetCleanupFilter.xml. If a namespace specified for the Site does not have a corresponding entry in the that file, it will be filtered out. |
| navTheme | Theme indicator for stylesheet coding in Managed Navigation, as specified in the Nav Theme field on the Site editor. |
| SiteVariables | Set of Context variables. Variables defined are merged with existing variables. Variables can be configured for a specific Site, a set of Sites, or all Sites |

Site Variables

| Property | Description |
|----------|--|
| name | Name of the Site Context Variable, as specified in the Context Variable Name field on the Add Context Variable page. |
| context | The Context to which the value of the variable will be applied, as specified in the Context drop list on the Add Context Variable page. |
| value | Value of the specified variable for the specified Context, as specified in the Context Variable Value field on the Add Context Variable page. |

Example Configuration

Configure variables for one specific Site

Configuration Bean

```
<bean id="ciSite"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
  <property name="name" value="Corporate_Investments"/>
  <property name="propertySetters">
    <list>
      <bean class="com.percussion.rx.config.impl.PSSiteSetter">
        <property name="properties">
          <map>
```

```

        <entry key="siteFolderPath"
value="${perc.SiteExample.CmsPath}"/>
        <entry key="publishedPath"
value="${perc.SiteExample.PublishedPath}"/>
        <entry key="publishedUrl"
value="${perc.SiteExample.PublishedUrl}"/>
        <entry key="globalTemplate"
value="${perc.SiteExample.GlobalTemplateName}"/>
        <entry key="unpublishFlags"
value="${perc.SiteExample.UnpublishFlags}"/>
        <entry key="allowedNamespaces"
value="${perc.SiteExample.AllowedNamespaces}"/>
        <entry key="navTheme"
value="${perc.SiteExample.NavTheme}"/>
        <entry key="ftpAddress"
value="${perc.SiteExample.ftp.IpAddress}"/>
        <entry key="ftpPort"
value="${perc.SiteExample.ftp.Port}"/>
        <entry key="ftpUser"
value="${perc.SiteExample.ftp.UserId}"/>
        <entry key="ftpPassword"
value="${perc.SiteExample.ftp.Password}"/>
    </map>
</property>
</bean>
<bean class="com.percussion.rx.config.impl.PSSiteSetter">
    <property name="properties">
        <map>
            <entry key="variables"
value="${perc.SiteExample.SiteVariables_CI}"/>
        </map>
    </property>
</bean>
</list>
</property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="SiteExample">
        <!-- Properties for the CI Site -->
        <property name="CmsPath" value="//Sites/CorporateInvestments"/>
        <property name="PublishedPath" value="../CI_Home.war"/>
        <property name="PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
        <property name="globalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
        <propertySet name="ftp">
            <property name="IpAddress" value="127.0.0.1"/>
            <property name="Port" value="21"/>
            <property name="UserId" value=""/>
            <property name="Password" value=""/>
        </propertySet>
    </SolutionConfig>
</SolutionConfigurations>

```



```

<property name="UnpublishFlags" value="u"/>
<property name="AllowedNamespaces" value=""/>
<property name="NavTheme" value=""/>
<!-- Site (Context) Variables for the CI Site -->
<property name="SiteVariables_CI">
  <propertySet>
    <property name="name" value="rxs_urlroot"/>
    <property name="context" value="Site_Folder_Assembly"/>
    <property name="value" value="/CI_Home"/>
  </propertySet>
  <propertySet>
    <property name="name" value="rxs_navbase"/>
    <property name="context" value="Preview"/>
    <property name="value"
value="../web_resources/corporate_investments"/>
  </propertySet>
  <propertySet>
    <property name="name" value="rxs_navbase"/>
    <property name="context" value="Site_Folder_Assembly"/>
    <property name="value" value="/CI_Home/resources"/>
  </propertySet>
</property>
</SolutionConfig>
</SolutionConfigurations>

```

Example Configuration

Configure variables for a specific set of Sites, user specifies the Sites; this configuration is recommended for providing the most flexibility to the customer

Configuration Bean

```

<bean id="Corporate_Investments"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
  <property name="names" value="{CommonConfigSiteNames}"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSSiteSetter">
      <property name="properties">
        <map>
          <entry key="siteFolderPath"
value="{perc.SiteExample.CI_CmsPath}"/>
          <entry key="publishedPath"
value="{perc.SiteExample.CI_PublishedPath}"/>
          <entry key="publishedUrl"
value="{perc.SiteExample.CI_PublishedUrl}"/>
          <entry key="globalTemplate"
value="{perc.SiteExample.CI_GlobalTemplateName}"/>
          <entry key="unpublishFlags"
value="{perc.SiteExample.CI_UnpublishFlags}"/>
          <entry key="allowedNamespaces"
value="{perc.SiteExample.CI_AllowedNamespaces}"/>
          <entry key="navTheme"
value="{perc.SiteExample.CI_NavTheme}"/>
          <entry key="ftpAddress"
value="{perc.SiteExample.CI_ftp.Address}"/>

```

```
        <entry key="ftpPort"
value="\${perc.SiteExample.CI_ftp.Port}"/>
        <entry key="ftpUser"
value="\${perc.SiteExample.CI_ftp.User}"/>
        <entry key="ftpPassword"
value="\${perc.SiteExample.CI_ftp.Password}"/>
    </map>
</property>
</bean>
</property>
<bean id="Enterprise_Investments"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="name" value="Enterprise_Investments"/>
    <property name="propertySetters">
        <bean class="com.percussion.rx.config.impl.PSSiteSetter">
            <property name="properties">
                <map>
                    <entry key="siteFolderPath"
value="\${perc.SiteExample.EI_CmsPath}"/>
                    <entry key="publishedPath"
value="\${perc.SiteExample.EI_PublishedPath}"/>
                    <entry key="publishedUrl"
value="\${perc.SiteExample.EI_PublishedUrl}"/>
                    <entry key="globalTemplate"
value="\${perc.SiteExample.EI_GlobalTemplateName}"/>
                    <entry key="unpublishFlags"
value="\${perc.SiteExample.EI_UnpublishFlags}"/>
                    <entry key="allowedNamespaces"
value="\${perc.SiteExample.EI_AllowedNamespaces}"/>
                    <entry key="navTheme"
value="\${perc.SiteExample.EI_NavTheme}"/>
                    <entry key="ftpAddress"
value="\${perc.SiteExample.EI_ftp.Address}"/>
                    <entry key="ftpPort"
value="\${perc.SiteExample.EI_ftp.Port}"/>
                    <entry key="ftpUser"
value="\${perc.SiteExample.EI_ftp.User}"/>
                    <entry key="ftpPassword"
value="\${perc.SiteExample.EI_ftp.Password}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>
</property>
</bean>
<bean id="CIEICommonConfigurations"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="names"
value="\${perc.siteexample.commonConfigSiteNames}"/>
    <property name="propertySetters">
        <bean class="com.percussion.rx.config.impl.PSSiteSetter">
            <property name="properties">
                <map>
                    <entry key="variables"
value="\${perc.SiteExample.CommonConfigSiteVariables}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>
```

```

        </bean>
    </property>
< /bean>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software, Inc." type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="SiteExample">
        <!-- Properties for the CI Site -->
        <property name="CI_CmsPath"
value="//Sites/CorporateInvestments"/>
        <property name="CI_PublishedPath" value="../CI_Home.war"/>
        <property name="CI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
        <property name="CI_GlobalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
        <propertySet name="CI_ftp">
            <property name="IpAddress" value="127.0.0.1"/>
            <property name="Port" value="21"/>
            <property name="UserId" value=""/>
            <property name="Password" value=""/>
        </propertySet>
        <property name="CI_UnpublishFlags" value="u"/>
        <property name="CI_AllowedNamespaces" value=""/>
        <property name="CI_NavTheme" value=""/>
        <!-- Properties for the EI Site -->
        <property name="EI_CmsPath"
value="//Sites/CorporateInvestments"/>
        <property name="EI_PublishedPath" value="../CI_Home.war"/>
        <property name="EI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
        <property name="EI_GlobalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
        <propertySet name="EI_ftp">
            <property name="IpAddress" value="127.0.0.1"/>
            <property name="Port" value="21"/>
            <property name="UserId" value=""/>
            <property name="Password" value=""/>
        </propertySet>
        <property name="EI_UnpublishFlags" value="u"/>
        <property name="EI_AllowedNamespaces" value=""/>
        <property name="EI_NavTheme" value=""/>
        <!-- Site (Context) Variables applied to both Sites -->
        <property name="CommonConfigSiteNames">
            <pvalues>
                <pvalue>Corporate_Investments</pvalue>
                <pvalue>Enterprise_Investments</pvalue>
            </pvalues>
        </property>
        <property name="CommonConfigSiteVariables">
            <propertySet>

```

```

        <property name="name" value="rxs_urlroot"/>
        <property name="context" value="Site_Folder_Assembly"/>
        <property name="value" value="/Investments_Home"/>
    </propertySet>
    <propertySet>
        <property name="name" value="rxs_navbase"/>
        <property name="context" value="Preview"/>
        <property name="value"
value=" ../web_resources/common_investments"/>
    </propertySet>
    <propertySet>
        <property name="name" value="rxs_navbase"/>
        <property name="context" value="Site_Folder_Assembly"/>
        <property name="value"
value="/Investments_Home/resources"/>
    </propertySet>
</property>
</SolutionConfig>
</SolutionConfigurations>

```

Example Configuration

Configure variables for a specific set of Sites specified by wildcards; all sites that match the wildcard [in this case *invest*; matches all Sites whose name includes the string "invest" will be updated with the configurations; this approach is best suited to in internal deployment package where sites share common data

Configuration Bean

```

<bean id="Corporate_Investments"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="name" value="Corporate_Investments"/>
    <property name="propertySetters">
        <bean class="com.percussion.rx.config.impl.PSSiteSetter">
            <property name="properties">
                <map>
                    <entry key="siteFolderPath"
value="\${perc.SiteExample.CI_CmsPath}"/>
                    <entry key="publishedPath"
value="\${perc.SiteExample.CI_PublishedPath}"/>
                    <entry key="publishedUrl"
value="\${perc.SiteExample.CI_PublishedUrl}"/>
                    <entry key="globalTemplate"
value="\${perc.SiteExample.CI_GlobalTemplateName}"/>
                    <entry key="unpublishFlags"
value="\${perc.SiteExample.CI_UnpublishFlags}"/>
                    <entry key="allowedNamespaces"
value="\${perc.SiteExample.CI_AllowedNamespaces}"/>
                    <entry key="navTheme"
value="\${perc.SiteExample.CI_NavTheme}"/>
                    <entry key="ftpAddress"
value="\${perc.SiteExample.CI_ftp.Address}"/>
                    <entry key="ftpPort"
value="\${perc.SiteExample.CI_ftp.Port}"/>
                    <entry key="ftpUser"
value="\${perc.SiteExample.CI_ftp.User}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>

```

```

        <entry key="ftpPassword"
value="\${perc.SiteExample.CI_ftp.Password}"/>
        </map>
    </property>
</bean>
</property>
</bean>
<bean id="Enterprise_Investments"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="name" value="Enterprise_Investments"/>
    <property name="propertySetters">
        <bean class="com.percussion.rx.config.impl.PSSiteSetter">
            <property name="properties">
                <map>
                    <entry key="siteFolderPath"
value="\${perc.SiteExample.EI_CmsPath}"/>
                    <entry key="publishedPath"
value="\${perc.SiteExample.EI_PublishedPath}"/>
                    <entry key="publishedUrl"
value="\${perc.SiteExample.EI_PublishedUrl}"/>
                    <entry key="globalTemplate"
value="\${perc.SiteExample.EI_GlobalTemplateName}"/>
                    <entry key="unpublishFlags"
value="\${perc.SiteExample.EI_UnpublishFlags}"/>
                    <entry key="allowedNamespaces"
value="\${perc.SiteExample.EI_AllowedNamespaces}"/>
                    <entry key="navTheme"
value="\${perc.SiteExample.EI_NavTheme}"/>
                    <entry key="ftpAddress"
value="\${perc.SiteExample.EI_ftp.Address}"/>
                    <entry key="ftpPort"
value="\${perc.SiteExample.EI_ftp.Port}"/>
                    <entry key="ftpUser"
value="\${perc.SiteExample.EI_ftp.User}"/>
                    <entry key="ftpPassword"
value="\${perc.SiteExample.EI_ftp.Password}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>
<bean id="InvestConfig"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="name" value="*invest*"/>
    <property name="propertySetters">
        <bean class="com.percussion.rx.config.impl.PSSiteSetter">
            <property name="properties">
                <map>
                    <entry key="variables"
value="\${perc.SiteExample.InvestSiteVariables}"/>
                </map>
            </property>
        </bean>
    </property>
</bean>
</property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software, Inc." type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="SiteExample">
    <!-- Properties for the CI Site -->
    <property name="CI_CmsPath"
value="//Sites/CorporateInvestments"/>
    <property name="CI_PublishedPath" value="../CI_Home.war"/>
    <property name="CI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
    <property name="CI_GlobalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
    <propertySet name="CI_ftp">
      <property name="IpAddress" value="127.0.0.1"/>
      <property name="Port" value="21"/>
      <property name="UserId" value=""/>
      <property name="Password" value=""/>
    </propertySet>
    <property name="CI_UnpublishFlags" value="u"/>
    <property name="CI_AllowedNamespaces" value=""/>
    <property name="CI_NavTheme" value=""/>
    <!-- Properties for the EI Site -->
    <property name="EI_CmsPath"
value="//Sites/CorporateInvestments"/>
    <property name="EI_PublishedPath" value="../CI_Home.war"/>
    <property name="EI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
    <property name="EI_GlobalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
    <propertySet name="EI_ftp">
      <property name="IpAddress" value="127.0.0.1"/>
      <property name="Port" value="21"/>
      <property name="UserId" value=""/>
      <property name="Password" value=""/>
    </propertySet>
    <property name="EI_UnpublishFlags" value="u"/>
    <property name="EI_AllowedNamespaces" value=""/>
    <property name="EI_NavTheme" value=""/>
    <!-- Properties for the CI Site -->
    <property name="CI_CmsPath"
value="//Sites/CorporateInvestments"/>
    <property name="CI_PublishedPath" value="../CI_Home.war"/>
    <property name="CI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
    <property name="CI_globalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
    <propertySet name="CI_ftp">
      <property name="IpAddress" value="127.0.0.1"/>
      <property name="Port" value="21"/>
      <property name="UserId" value=""/>
      <property name="Password" value=""/>
    </propertySet>
  </SolutionConfig>
</SolutionConfigurations>

```

```

    <property name="CI_UnpublishFlags" value="u"/>
    <property name="CI_AllowedNamespaces" value="" />
    <property name="CI_NavTheme" value="" />
    <!-- Properties for the EI Site -->
    <property name="EI_CmsPath"
value="//Sites/CorporateInvestments"/>
    <property name="EI_PublishedPath" value="../CI_Home.war"/>
    <property name="EI_PublishedUrl"
value="http://127.0.0.1:9992/CI_Home"/>
    <property name="EI_GlobalTemplateName"
value="rffGtCorporateInvestmentsCommon"/>
    <propertySet name="EI_ftp">
        <property name="IpAddress" value="127.0.0.1"/>
        <property name="Port" value="21"/>
        <property name="UserId" value="" />
        <property name="Password" value="" />
    </propertySet>
    <property name="EI_UnpublishFlags" value="u"/>
    <property name="EI_AllowedNamespaces" value="" />
    <property name="EI_NavTheme" value="" />
    <!-- Site (Context) Variables applied to all Sites that include
the string "invest" -->
    <property name="investSiteVariables">
        <propertySet>
            <property name="name" value="rxs_urlroot"/>
            <property name="context" value="Site_Folder_Assembly"/>
            <property name="value" value="/Investments_Home"/>
        </propertySet>
        <propertySet>
            <property name="name" value="rxs_navbase"/>
            <property name="context" value="Preview"/>
            <property name="value"
value="../web_resources/common_investments"/>
        </propertySet>
        <propertySet>
            <property name="name" value="rxs_navbase"/>
            <property name="context" value="Site_Folder_Assembly"/>
            <property name="value" value="/Investments_Home/resources"/>
        </propertySet>
    </property>
</SolutionConfig>
</SolutionConfigurations>

```

Site/Template Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSSiteConfigHandler

Handler Properties

| Property | Description |
|----------|--|
| name | Used to specify setter properties for a single Site. Specifies the Site to which the specified setter properties will be applied. A Site configuration bean can use either the name property or the names property, but not both. They are mutually exclusive. |
| names | Used to specify setter properties for multiple Sites. Specifies the Sites to which the specified setter properties will be applied. The value of this property can be a list of Site names (applies the configurations to the specified Sites), a simple wildcard ("*"; applies the configurations to all Sites), a string with one or more wild cards ("*Invest*" will match both EnterpriseInvestments and CorporateInvestments; applies the configurations to all Sites whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}; applies the configurations to the Sites specified in the replacement variable in the default or local configurations). A Site configuration bean can use either the name property or the names property, but not both. They are mutually exclusive. |

Type

SITE (NOTE: When using PSSiteConfigHandler, the Type property is optional. If included, the value MUST be SITE.)

Property Setter Class

com.percussion.rx.config.impl.PSSiteTemplateVisibilitySetter

Configurable Properties

| Property | Description | | | | | | |
|------------|--|----------|-------------|-------|--|-----------|---|
| visibility | <p>Defines the visibility of the specified Templates on the specified Sites. Must contain two properties:</p> <table border="1"> <thead> <tr> <th>Property</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>sites</td> <td>List of Site names, as specified in the Name field on the Site editor. The value of this property can be a list of Site names (makes the specified Templates visible to the listed Sites), a simple wildcard ("*"; makes the specified Templates visible to all Sites), a string with one or more wild cards ("*Invest*" will match both EnterpriseInvestments? and CorporateInvestments?; makes the specified Templates visible to all Sites whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}); makes the specified Templates visible to the Sites specified in the replacement variable in the default or local configurations)</td> </tr> <tr> <td>templates</td> <td>List of Template names, as shown on the Template editor. The listed Templates will be associated with the Sites specified in the sites property. The Templates will be removed from any Sites not listed in the sites property. Thus, if you want to maintain an existing association between a Site and a Template, you must include the Site and the Template in the configuration.</td> </tr> </tbody> </table> | Property | Description | sites | List of Site names, as specified in the Name field on the Site editor. The value of this property can be a list of Site names (makes the specified Templates visible to the listed Sites), a simple wildcard ("*"; makes the specified Templates visible to all Sites), a string with one or more wild cards ("*Invest*" will match both EnterpriseInvestments? and CorporateInvestments?; makes the specified Templates visible to all Sites whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}); makes the specified Templates visible to the Sites specified in the replacement variable in the default or local configurations) | templates | List of Template names, as shown on the Template editor. The listed Templates will be associated with the Sites specified in the sites property. The Templates will be removed from any Sites not listed in the sites property. Thus, if you want to maintain an existing association between a Site and a Template, you must include the Site and the Template in the configuration. |
| Property | Description | | | | | | |
| sites | List of Site names, as specified in the Name field on the Site editor. The value of this property can be a list of Site names (makes the specified Templates visible to the listed Sites), a simple wildcard ("*"; makes the specified Templates visible to all Sites), a string with one or more wild cards ("*Invest*" will match both EnterpriseInvestments? and CorporateInvestments?; makes the specified Templates visible to all Sites whose name matches the wildcards), or a replacement variable (\${perc.solution.replacementvariable}); makes the specified Templates visible to the Sites specified in the replacement variable in the default or local configurations) | | | | | | |
| templates | List of Template names, as shown on the Template editor. The listed Templates will be associated with the Sites specified in the sites property. The Templates will be removed from any Sites not listed in the sites property. Thus, if you want to maintain an existing association between a Site and a Template, you must include the Site and the Template in the configuration. | | | | | | |

Example Configuration

Make Templates visible to one specific Site

Configuration Bean

```
<bean id="siteVariables"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
  <property name="name" value="Corporate Investments"/>
  <property name="propertySetters">
    <bean
class="com.percussion.rx.config.impl.PSSiteTemplateVisibilitySetter">
      <property name="properties">
        <map>
          <entry key="visibility"
value="\${perc.SiteTemplateVisibility.Visibility}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

Default Configuration

```
<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="SiteTemplateVisibility">
    <property name="Visibility">
      <pvalues>
        <pvalue>rffPgCiGeneric</pvalue>
        <pvalue>rffPgCiEvent</pvalue>
        <pvalue>rffPgCiHome</pvalue>
        <pvalue>rffPgCiGenericWord</pvalue>
        <pvalue>rffPgCiPressRelease</pvalue>
      </pvalues>
    </property>
  </SolutionConfig>
</SolutionConfigurations>
```

Example Configuration

Make Templates visible to a specific set of Sites, user specifies the Sites; this configuration is recommended for providing the most flexibility to the customer

Configuration Bean

```
<bean id="siteVariables"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
  <property name="names"
value="\${perc.SiteTemplateVisibility.SiteList}"/>
  <property name="propertySetters">
    <bean
class="com.percussion.rx.config.impl.PSSiteTemplateVisibilitySetter">
      <property name="properties">
        <map>
          <entry key="visibility"
value="\${perc.SiteTemplateVisibility.Visibility}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

Default Configuration

```
<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="SiteTemplateVisibility">
    <property name="SiteList">
      <pvalues>
        <pvalue>Corporate Investments</pvalue>
        <pvalue>Enterprise Investments</pvalue>
      </pvalues>
    </property>
    <property name="Visibility">
      <pvalues>
        <pvalue>rffBnImage</pvalue>
        <pvalue>rffBnBinary</pvalue>
        <pvalue>rffSnCallout</pvalue>
      </pvalues>
    </property>
  </SolutionConfig>
</SolutionConfigurations>
```

```

        <pvalue>rffSnImageLink</pvalue>
        <pvalue>rffSnTitleLink</pvalue>
    </pvalues>
</property>
</SolutionConfig>
</SolutionConfigurations>

```

Example Configuration

Make Templates visible to a specific set of Sites specified by wildcards; all sites that match the wildcard [in this case *invest*; matches all Sites whose name includes the string "invest" will be updated with the configurations; this approach is best suited to in internal deployment package where sites share common data

Configuration Bean

```

<bean id="siteVariables"
class="com.percussion.rx.config.impl.PSSiteConfigHandler">
    <property name="names" value="*invest*" />
    <property name="propertySetters">
        <bean
class="com.percussion.rx.config.impl.PSSiteTemplateVisibilitySetter">
            <property name="properties">
                <map>
                    <entry key="visibility"
value="${perc.SiteTemplateVisibility.Visibility}" />
                </map>
            </property>
        </bean>
    </property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="SiteTemplateVisibility">
        <property name="Visibility">
            <pvalues>
                <pvalue>rffBnImage</pvalue>
                <pvalue>rffBnBinary</pvalue>
                <pvalue>rffSnCallout</pvalue>
                <pvalue>rffSnImageLink</pvalue>
                <pvalue>rffSnTitleLink</pvalue>
            </pvalues>
        </property>
    </SolutionConfig>
</SolutionConfigurations>

```

Slot Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

SLOT

Property Setter Class

com.percussion.rx.config.impl.PSSlotSetter

Configurable Properties

| Property | Description |
|--------------------------|---|
| finderParams | <p>Set of Content Finder parameters. The set specified in the configuration overrides the set defined in the Slot.</p> <p>The set should be specified in a <propertySet> node, with the properties specified in individual <property> nodes.</p> |
| contentTypeTemplatePairs | <p>Set of Content Type/Template pairs associated with the Slot. The set specified in the configuration overrides the set defined in the Slot.</p> <p>The set should be defined as a list of pairs of values.</p> <p>If the package is uninstalled, merged associations will be removed.</p> <p>A Content Type/Template pair cannot be specified in more than one package. If a later package attempts to add an existing Content Type/Template pair, it will fail validation and the configuration will fail.</p> |

Example Configuration Bean

```
<bean id="slotExample"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="rffEvents"/>
  <property name="type" value="SLOT"/>
  <property name="propertySetters">
    <bean
class="com.percussion.rx.config.impl.PSTemplateSlotSetter">
      <property name="properties">
        <map>
          <entry key="finderParams"
value="\${perc.SlotExample.FinderParameters}"/>
          <entry key="contentTypeTemplatePairs"
value="\${perc.SlotExample.SlotAssociations}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>
```

```
    </property>
</bean>
```

Example Default Configuration

```
<SolutionConfigurations publisherPrefix="perc" publisherName="Percussion
Software" type="config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="slotexample">
    <property name="SlotExample">
      <pvalues>
        <pair value1="rffEvent" value2="rffSnTitleLink"/>
      </pvalues>
    </property>
    <propertySet name="FinderParameters">
      <property name="query" value="select rx:sys_contentid,
rx:sys_contentstartdate from rx:rffcalendar where jcr:path like
:sitepath order by rx:sys_contentstartdate asc"/>
      <property name="type" value="sql"/>
      <property name="template" value="rffSnTitleCalloutLink"/>
      <property name="max_results" value="4"/>
      <property name="sys_lang" value=""/>
    </propertySet>
  </SolutionConfig>
</SolutionConfigurations>
```

Template Configuration Reference

Configuration Handler Class

com.percussion.rx.config.impl.PSObjectConfigHandler

Type

TEMPLATE

Property Setter Class

com.percussion.rx.config.impl.PSTemplateSetter

Configurable Properties

| Property | Description |
|---------------------|---|
| label | Template label, as defined in the Template label field of the General tab of the Template Editor |
| globalTemplateUsage | Only has meaning for Page Templates. Specifies the way the Global Template for the Page is determined. Valid options are <i>Default</i> (uses the default Global Template for the Site or Folder), <i>Defined</i> (uses the specified Global Template; requires the globalTemplate Property), or <i>None</i> (no Global Template is applied to the Page). |
| globalTemplate | Only has meaning if the value of the globalTemplateUsage property is <i>Defined</i> . Specifies the Global Template to use for the Page Template. |
| mimetype | Specifies the MIME Type of the output of the Template, as defined in the Mime Type field on the Template Editor. Valid values are the same as the valid values for that field. |
| charSet | Specifies the character set used for the output of the Template, as defined in the Character Set field of the Template Editor. Valid values are the same as the valid values for that field. |
| locationPrefix | Optional prefix for the filename of the published Template, as defined in the Prefix field on the Template Editor. |
| locationSuffix | Optional suffix for the filename of the published Template, as defined in the Suffix field on the Template Editor. |

| Property | Description | | | | | | | | |
|-------------|--|-------|-------------|--------|-------------------------------|---------|--|-------|-----------------------------|
| publishWhen | <p>Specifies the circumstances when the Template should be published when publishing to a Site Folder as defined by the Publish Option on the Template Editor. Valid values include:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Always</td> <td>Always publish this Template.</td> </tr> <tr> <td>Default</td> <td>Publish this Template if it is the Default Template for the Content Type</td> </tr> <tr> <td>Never</td> <td>Never publish this Template</td> </tr> </tbody> </table> | Value | Description | Always | Always publish this Template. | Default | Publish this Template if it is the Default Template for the Content Type | Never | Never publish this Template |
| Value | Description | | | | | | | | |
| Always | Always publish this Template. | | | | | | | | |
| Default | Publish this Template if it is the Default Template for the Content Type | | | | | | | | |
| Never | Never publish this Template | | | | | | | | |
| bindingSet | List of bindings to replace the existing set of bindings on the Template | | | | | | | | |
| bindings | List of bindings to merge with the existing set of bindings on the Template. If a binding in the list already exists, the value from the list overrides the existing value. If a binding in the list does not exist, it is appended to the list of existing bindings. The default or local configuration can include the optional entry binding_sequence to re-order the sequence of bindings | | | | | | | | |

Example Configuration

No Global Template override, merge bindings

Configuration Bean

```

<bean id="rffPgCiGeneric"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="rffPgCiGeneric"/>
  <property name="type" value="TEMPLATE"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSTemplateSetter">
      <property name="properties">
        <map>
          <entry key="label"
value="\${perc.TemplateExample.Label}"/>
          <entry key="publishWhen"
value="\${perc.TemplateExample.PublishWhen}"/>
          <entry key="mimeType"
value="\${perc.TemplateExample.MimeType}"/>
          <entry key="charset"
value="\${perc.TemplateExample.Charset}"/>
          <entry key="locationPrefix"
value="\${perc.TemplateExample.LocationPrefix}"/>
          <entry key="locationSuffix"
value="\${perc.TemplateExample.LocationSuffix}"/>
          <entry key="bindings"
value="\${perc.TemplateExample.Bindings}"/>
        </map>
      </property>
    </bean>
  </property>
</bean>

```

```
    </property>
</bean>
```

Default Configuration

```
<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
  <SolutionConfig name="TemplateExample">
    <property name="Label" value="P - CI Generic"/>
    <property name="PublishWhen" value="Default"/>
    <property name="MimeType" value="text/html"/>
    <property name="Charset" value="UTF-8"/>
    <property name="CocationPrefix" value=""/>
    <property name="LocationSuffix" value=""/>
    <property name="Bindings">
      <pvalues>
        <pair value1="$rxs_navbase_2"
value2="$sys.variables.rxs_navbase_2"/>
        <pair value1="$rxs_navbase"
value2="$sys.variables.rxs_navbase"/>
        <pair value1="$rxs_navbase_1"
value2="$sys.variables.rxs_navbase_1"/>
      </pvalues>
    </property>
  </SolutionConfig>
</SolutionConfigurations>
```

Example Configuration

Global Template Override and binding set override

Configuration Bean

```
<bean id="rffPgCiGeneric"
class="com.percussion.rx.config.impl.PSObjectConfigHandler">
  <property name="name" value="rffPgCiGeneric"/>
  <property name="type" value="TEMPLATE"/>
  <property name="propertySetters">
    <bean class="com.percussion.rx.config.impl.PSTemplateSetter">
      <property name="properties">
        <map>
          <entry key="label"
value="$${perc.TemplateExample.Label}"/>
          <entry key="globalTemplate"
value="$${perc.TemplateExample.GlobalTemplate}"/>
          <entry key="globalTemplateUsage"
value="$${perc.TemplateExample.GlobalTemplateUsage}"/>
          <entry key="publishWhen"
value="$${perc.TemplateExample.PublishWhen}"/>
          <entry key="mimeType"
value="$${perc.TemplateExample.MimeType}"/>
          <entry key="charset"
value="$${perc.TemplateExample.Charset}"/>
          <entry key="locationPrefix"
value="$${perc.TemplateExample.LocationPrefix}"/>
        </map>
      </property>
    </bean>
  </property>
</property>
```



```

        <entry key="locationSuffix"
value=" ${perc.TemplateExample.LocationSuffix} " />
        <entry key="bindingSet"
value=" ${perc.TemplateExample.BindingSet} " />
    </map>
</property>
</bean>
</property>
</bean>

```

Default Configuration

```

<SolutionConfigurations publisherPrefix="perc"
publisherName="Percussion" type="config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localConfig.xsd">
    <SolutionConfig name="TemplateExample">
        <property name="label" value="P - CI Generic" />
        <property name="GlobalTemplate"
value="rffGtCorporateInvestmentsCommon" />
        <property name="GlobalTemplateUsage" value="Default" />
        <property name="PublishWhen" value="Default" />
        <property name="MimeType" value="text/html" />
        <property name="Charset" value="UTF-8" />
        <property name="LocationPrefix" value="" />
        <property name="LocationSuffix" value="" />
        <property name="BindingSet">
            <pvalues>
                <pair value1="$rxs_navbase"
value2="$sys.variables.rxs_navbase" />
                <pair value1="$rxs_navbase_1"
value2="$sys.variables.rxs_navbase_1" />
                <pair value1="$rxs_navbase_2"
value2="$sys.variables.rxs_navbase_2" />
            </pvalues>
        </property>
    </SolutionConfig>
</SolutionConfigurations>

```