# **BPS 1131 - BIRT General Designer Usability** Improvements Specification

**Feature Specification** 

#### **Abstract**

This is the feature specification for BIRT general designer usability improvements.

# **Revisions**

Version	Author	Date	Description of Changes
Draft 1	Zhiqiang Qian	May 5 2009	Initial Draft

Abstract	
Revisions	
New Widget for Dimension Properties	
Introduction	
Goals	
Features	
Custom Report Template Extension	5
Introduction	5
Use Cases	
Feature Overview	7
Extensions	
Interfaces	
Resource Folder Setting Enhancement	
Introduction	
Use Cases	
Features	
Relative Path to Current Project	
Workspace Path	
Variable Path	

# **New Widget for Dimension Properties**

## Introduction

Currently in the Property Editor, the dimension property editing UI consists of two widgets: a text control plus a combo box control. They are used to edit the dimension measure and unit value respectively. As the two widgets are managed separately, when user change the focus from the text control to the combo box control or vice versa, it will always trigger the property change event, while at this moment user may not have completed the editing yet.

For example, user wants to change the dimension value from "1 in" to "100 pixels", currently he must be careful to change the unit first, otherwise, when he changed the measure value to "100" and attempt to change the unit, the dimension value will temporarily be changed to "100 in", which will then cause an unnecessary re-layout. For certain report item like Image, this may also trigger a creation of a very large image instance, which may cause an OutOfMemory error.

🛱 Property Editor - Picture 🗙 🔝 Problems				
Properties Binding Highli	Properties Binding Highlights			
General	General			
Reference	Name: Element ID: 146			
Hyperlink				
Alt Text	Width: 99 pixels  Height:			
Border	in			
Page Break	Style: None picas			
Visibility	points			
Table of Contents	pixels			
Bookmark				

### Goals

Our goal is to merge the two widgets into one and change the event handling, so it can avoid the unnecessary event.

Basically, when the focus is changed inside the container widget, it will not file the property change event, only when the focus is set to outside the container widget, the event is filed.



# **Features**

The new custom widget has the same functions as the existing one.

# **Custom Report Template Extension**

# Introduction

Currently BIRT designer provides a template folder setting to allow user to register custom report templates. When user creates a report, the new report wizard will list all the registered report templates reside in this folder.

🚝 Preferences		<u>_ 0 ×</u>
type filter text	Template	← + ⇒ + ▼
<ul> <li>General</li> <li>Ant</li> <li>Data Management</li> <li>Help</li> <li>Install/Update</li> <li>Java</li> <li>JavaScript</li> <li>JPA</li> <li>Plug-in Development</li> <li>Remote Systems</li> <li>Report Design</li> <li>Bidirectional Properties</li> <li>Chart</li> <li>Comment Template</li> <li>Crosstab</li> <li>Data Set Editor</li> <li>Element Names</li> <li>Preview</li> <li>Resource</li> <li>Template</li> <li>Server</li> <li>Server</li> <li>Service Policies</li> <li>Tasks</li> <li>Team</li> <li>Usage Data Collector</li> </ul>	Please select the template folder. Template folder Select	Configure Project Specific Settings
?		OK Cancel

🖨 New Report	
Report Choose the type of report you want to creat	te.
Report templates:         Blank Report         My First Report         Simple Listing         Grouped Listing         Dual Column Listing         Chart & Listing         Dual Column Chart & Listing         Side by Side Chart & Listing         Side by Side Chart & Listing         Custom Template         Image: Show Report Creation Cheat Sheet         Report Orientation:       Left To Right	Preview: <pre></pre>
Pack	Next > Finish Cancel

However, there's some limitation of current solution as described below.

#### **Use Cases**

- User want to have a structured view for all the report template so they can be better organized and understood. But currently the template list only supports one level.
- User want to put report templates in multiple folders in the file system but currently there's only one template folder setting.
- User want to access report templates that are not stored as local disk files, e.g. from server using custom communication protocols.

### **Feature Overview**

A new wizard template extension will be introduced to resolve these issues. The default template wizard UI now will look like this:

New Report	<u>_                                    </u>
Report Choose the type of report you want to create.	
Report templates:	Image: State 1         Image:
Show Report Creation Cheat Sheet	
Report Orientation: Left To Right	<b>_</b>
?	Next > Finish Cancel

The original built-in templates and legacy custom templates will be grouped under a "Standard" node.

If user has registered some template extensions, the wizard UI may look like this:

New Report	
Report Choose the type of report you want to create.	
Report templates: Standard Blank Report My First Report Simple Listing Chart & Listing Lual Column Listing Chart & Listing Dual Column Chart & Listing Side by Side Chart & Listing Side by Side Chart & Listing E Custom Template Banking Disconing Table Personal My Check 1 My Check 3	Preview:         Preview:
Show Report Creation Cheat Sheet Report Orientation: Left To Right	
? < Back	Next > Finish Cancel

#### **Extensions**

The report template provider extension is achieved through the existing designer Element Adpater extension point.

A typical extension registration is like this:

```
<extension point="org.eclipse.birt.report.designer.ui.elementAdapters">
<adaptable
class="org.eclipse.birt.report.designer.internal.ui.wizards.ExtensionTemplateListProvi
der">
<adapter class="test.TemplateTest1"
id="TestPlug.adapter1"
```

```
priority="10"
type="org.eclipse.birt.report.designer.ui.templates.ITemplateProvider">
</adapter>
</adaptable>
</extension>
```

User must implement a provider class which implements org.eclipse.birt.report.designer.ui.templates.ITemplateProvider interface. The Element Adapter extension point further provides some optional control like priority, enablement, etc. User can always refine these setting according to custom requirements.

The *ITemplateProvider* interface do not restrict the user to read report templates from disk file, so it can easily adapt the custom report template storage cases.

Note this template provider extension only extends the capability of reading report templates. For custom writing(or publishing) capability, user should implement their own management logic and register standard Eclipse UI extensions if needed.

#### Interfaces

Following are the related java interfaces that for user to implement a template extension provider:

```
public interface ITemplateProvider
{
  /**Gets the provider ID.
   * @return
   */
  String getParentBaseName();
  /**Gets the entries.
   * @return
   */
  ITemplateEntry[] getTemplates();
  /**
   * Release the resources allocated if applicable.
   */
  void release();
}
public interface ITemplateEntry
{
  /**Gets the name to display.
   * @return
   */
  String getName();
  /**Gets the image to show in the tree.
```

```
* @return
*/
Image getImage();
}
```

```
public interface ITemplateFolder extends ITemplateEntry
{
```

```
/**Gets the ID to decide the location on the tree.
  * @return
   */
  String getBaseName();
  /**Gets the children.
  * @return
  */
  ITemplateEntry[] getChildren();
}
public interface ITemplateFile extends ITemplateEntry
{
  /**Gets the Template
  * @return
   */
  ReportDesignHandle getReportHandle();
}
```

# **Resource Folder Setting Enhancement**

## Introduction

In current BIRT designer, the Resource Folder setting only supports two kinds of notations: "<Current Project Folder>" or an absolute local path. The "<Current Project Folder>" refers to the project root path for current opened report design if applicable; the absolute local path points to a fixed local file system directory path. Although we already support project-wise setting for Resource Folder, in some cases, it is still not very convenient for user to specify a preferable setting.

🖶 Preferences	
type filter text	Resource $\leftarrow \bullet $
<ul> <li>JavaScript</li> <li>JPA</li> <li>Plug-in Development</li> <li>Remote Systems</li> <li>Report Design</li> <li>Bidirectional Properties</li> <li>Chart</li> <li>Comment Template</li> <li>Crosstab</li> <li>Data Set Editor</li> <li>Element Names</li> <li>HTML Button</li> <li>Preview</li> <li>Resource</li> <li>Template</li> <li>Server</li> <li>Service Policies</li> <li>Tasks</li> <li>Team</li> <li>Web</li> </ul>	Configure Project Specific Settings Type or select the default resource folder to use for the library explorer, images, property files, and so on.  Resource folder <  Select
Web Services     ✓	Restore Defaults Apply
0	OK Cancel

Browse For Folder	? ×
Please select the default resource folder:	
💼 🚝 Mu Desussente	_
	-
Surban (S.)	
E System (C:)	
🖽 🖘 Data (D:)	
🕀 🐨 Work (E:)	
🕀 🖳 🖳 DVD Drive (F:)	
😟 🕕 Live Mesh Folders	
🕀 🛅 Shared Documents	
🗄 🛅 zgian's Documents	
🗄 🙀 My Sharing Folders	
🕀 📢 My Network Places	-
	_
Folder: My Documents	
Make New Folder OK Can	cel

### **Use Cases**

- User want to specify the resource folder path to "<Current Project Folder>/resources". The path points to a fixed "resources" sub folder under the project which contains current report design. This is convenient when multiple report projects all contain a "resources" sub folder and user will frequently move the report design between projects.
- User want to specify the resource folder path to the root of a specific resource project in the workspace like "resource-project". All the report design in current workspace will reference the resources from this resource project. So when user switch to another workspace, it will automatically refer to the resources under the same resource project path, or user can simply import or copy the old resource project to the new workspace.
- User want to specify the resource folder relative to some special path like "eclipse home", "system path", etc. User doesn't want to hard code the path so when the environment changed, he need not change the setting again.

#### Features

This enhancement aims at providing a more flexible way for Resource folder setting. In addition to the original support, we will add following enhancements:

#### **Relative Path to Current Project**

The "<Current Project Folder>" notation support is extended, so it can support syntax like "<Current Project Folder>/subfolder".

Preferences	
type filter text	Resource $\leftarrow \star \Rightarrow \star \star$
<ul> <li>⊕. General</li> <li>⊕. Ant</li> <li>⊕. Data Management</li> <li>⊕. Help</li> <li>Install/Update</li> <li>⊕. Java</li> <li>⊕. Bluggin Development</li> </ul>	Configure Project Specific Settings Type or select the default resource folder to use for the library explorer, images, property files, and so on. Resource folder
Pilg-II Development     Pilg-II Development     Bidirectional Properties     Chart     Comment Template     Crosstab     Data Set Editor     Element Names     HTML Button     New Preview Prototype Pi     Preview     Resource     Template     Team     XML	Workspace File System Variables
•	Restore Defaults Apply
?	OK Cancel

# Workspace Path

It can support the path points to a project root or sub folder within current workspace.

Folder Selection		
Choose a location relative to the workspace:		
crosstab/SWF		
<pre></pre>		
CustomerListAfter_Grouping     CustomerListAfter_Mapping		
Resource	= • ⇒ • <b>•</b> .	
<u>Configure Project Specific Settings</u> Type or select the default resource folder to use for the library explorer, images, property files, and so on.		
Resource folder \${workspace_loc:crosstab/SWF}		
Workspace File System	Variables	

#### Variable Path

It can support parsing path from the built-in eclipse variable system.

Select Variable			
Choose a variable (? = any character, * = any string):			
build_files         build_project         build_type         container_loc         container_name         container_path         eclipse_home         env_var         file_prompt         folder_prompt         java_extensions_regex         java_type_name			
password_prompt project_loc project_name project_path Variable Description:			
Returns the set of absolute file system paths whose modification caused the current build. A list of the characters, 'a' (added), 'c' (changed), 'r' (removed), 'f'			
Resource	$\Leftarrow \star \Rightarrow \star \bigstar$		
<u>Configure Project Specific Settings</u> Type or select the default resource folder to use for the library explorer, images, property files, and so on.			
Resource folder \${eclipse_home}			
Workspace File System.	Variables		