|  |  |
| --- | --- |
|  |  |
| Exadel JSF4BIRT |
| Copyright  Exadel Inc., 2010 |
|  |  |
| USER GUIDE |  |

**Document Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 1.0 | 02/04/10 | Andrey Markavtsov | Initial draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Table of Content

# Overview

JSF4Birt components library provides possibility to render BIRT-based reports on the page within applications using JSF framework. Report viewers designed and implemented as JSF components and allow easy and clear compatibility with JSF through defined list of tags and attributes. Library provides common used best practice for assigning report data (report parameters, UI options) to BIRT wrapper components.

## Birt wrapper component

Birt wrapper designed to display BIRT reports created in BIRT eclipse designer. This component uses BIRT runtime API. Birt wrapper component calls appropriate API from BIRT runtime and gets generated HTML code and puts it on the page. Everything is done on server side during usual JSF lifecycle processing.

## Actuate wrapper component

Actuate wrapper designed for loading reports from Actuate server installed. Component uses Actuate JS API to load and run reports on the page from client side.

# Requirements

**JDK 1.5** or higher is required.

Supported version of **JSF is 1.2**.\*. But we have assumption that component works fine also within newest *JSF version 2*.*\** (Not tested yet).

Supported version of **BIRT runtime is 2.3 or 2.5**.

Supported version of **Actuate JS API is 2.0**.

# Component library setup

Components library (jsf4birt) outgoes as one JAR file containing JSF components itself, tag lib definition, faces configuration file and JavaScript. It should be added to classpath of application server to allow JSF framework to recognize components in runtime. IDE should automatically define scope of available tags and list of attributes to be used for application page development.

## Installation instruction for Birt wrapper usage

As was mentioned above Birt wrapper uses BIRT runtime API which is required to be installed under J2EE (or Tomcat) server before component usage. See the following instruction how to install BIRT runtime for server:

1. Download birt runtime library here: <http://www.eclipse.org/downloads/download.php?file=/birt/downloads/drops/R-R1-2_5_1-200909220630/birt-runtime-2_5_1.zip> to use Birt Engine v2.5 or <http://www.eclipse.org/downloads/download.php?file=/birt/downloads/drops/R-R1-2_3_2_2-200906011507/birt-runtime-2_3_2_2.zip> to use v2.3

2. Create a */WEB-INF/lib* directory underneath the Tomcat *webapps* directory.

3. Copy all the jars in the *birt-runtime-[version]/ReportEngine/lib* directory from the Report Engine download into your *WebReport/WEB-INF/lib* directory.

4. Create a directory named *platform* in your *WEB-INF* folder.

5. Copy the *birt-runtime-[version]/Report Engine/plugins* and *birt-runtime-[version]/ReportEngine/configuration* directories to the *platform* directory you just created.

6. Add ‘jsf4birt’ library to classpath.

## Installation instruction for Actuate wrapper usage

1. Add ‘jsf4birt’ library to classpath
2. Install & configure Actuate server
3. Configure actuate server parameters in web.xml: URL to actuate server, login/password.

(Optional – these parameters can defined directly on the page through component attributes)

<context-param>

<param-name>actuate.serverUrl</param-name>

<param-value>http://localhost:8900/iportal</param-value>

</context-param>

<context-param>

<param-name>actuate.serverLogin</param-name>

<param-value>administrator</param-value>

</context-param>

<context-param>

<param-name>actuate.serverPassword</param-name>

<param-value></param-value>

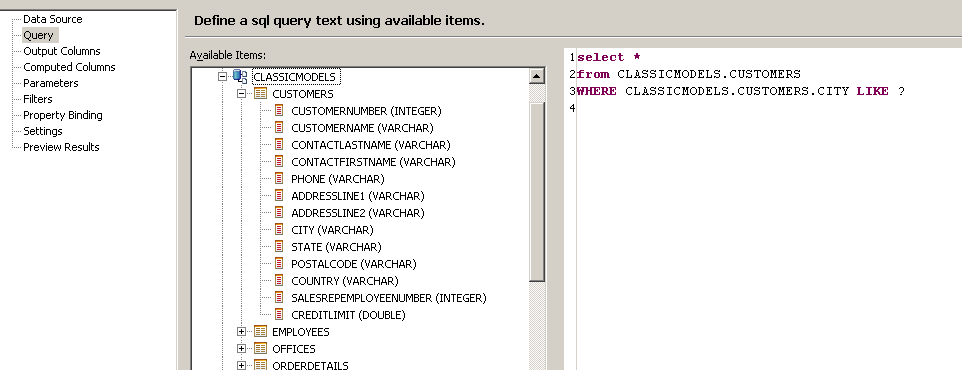
</context-param>

# Usage examples

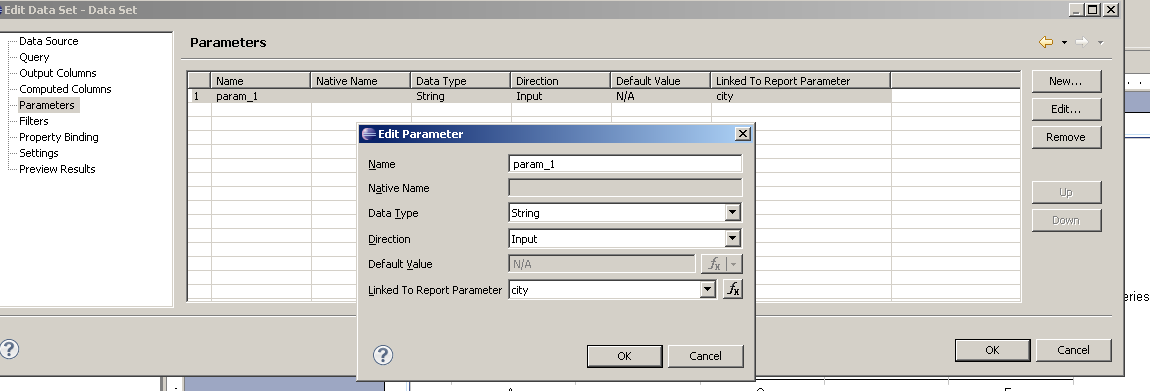
## Birt wrapper component

BIRT runtime contains DB sample which contains some classic schemas. Let’s take a look at CUSTOMER table. It includes customer name (first/last), city, credit limit & etc. Imagine that we would like to create BAR chart displaying credit limit for all customers from city defined from application user interface. So, firstly we should create report in eclipse designer:

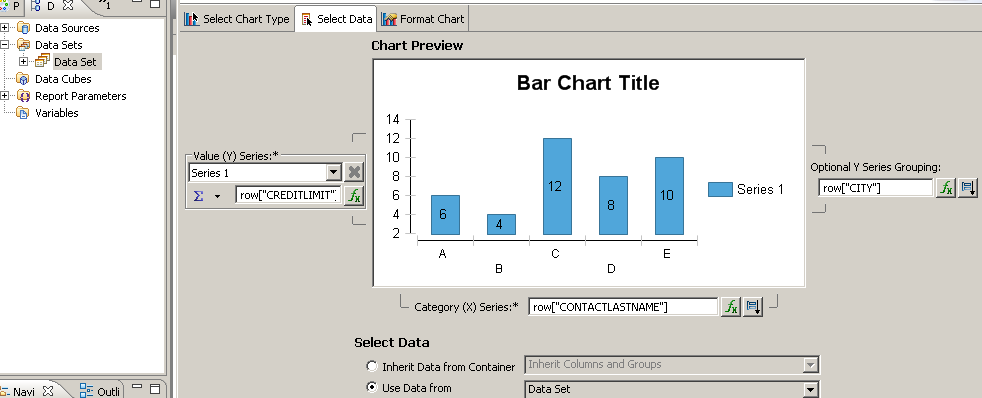
1. Define data set



DB query contains one ‘city’ parameter. We should bind it to report parameter



1. Add BAR chart to report add bind data



**Ready report design file here: **

After the report was placed under server we can use jsf4Birt tag library to display report in our page.

<h:form>

<h:inputText value=*"#{bean.city}"* /><br/>

<h:commandLink value=*"Search"* />

<birt:birtWrapper reportDesign=*"/Reports/new\_report\_8.rptdesign"*>

<f:param name=*"city"* value=*"#{bean.cityParam}"*/>

</birt:birtWrapper>

</h:form>

<birt:birtWrapper> tag defines placeholder on the page where generated report will be rendered.

‘reportDesign’ attribute specifies relative path to report design file.

‘city’ parameter should be passed to report through <f:param> tag.

Bean class code:

public class Bean {

String city;

**public** String getCity() {

**return** city;

}

// we should return parameter with ‘%’ suffix for SQL LIKE operator //syntax

**public** String getCityParam () {

**return** city != **null** ? city + "%" : **null**;

}

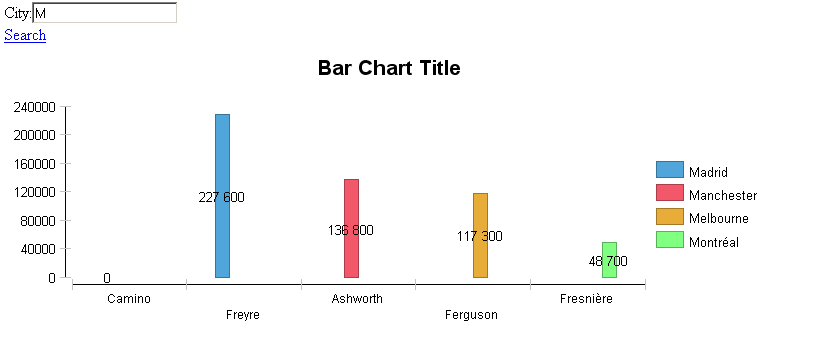
**public** **void** setCity(String city) {

**this**.city = city;

}

}

After application run we can enter first initial for city and press ‘Search’ button. Report will be updated and will display chart according to parameter entered:



We can specify CSS styles, width, height & etc for birt wrapper tag. See tlddoc for details.

## Actuate wrapper component

Page code sample:

<birt:actuateWrapper id=*"id"*

width=*"900px"*

reportName=*"#{bean.reportName}"*

height=*"700px"*

style=*"float: left; position: relative"*>

<f:facet name=*"emptyReport"*>

<h:panelGroup layout=*"block"* style=*"background-color: green"*>

<table style=*"width: 100%; height: 100%"*>

<tr>

<td>Select a report</td>

</tr>

</table>

</h:panelGroup>

</f:facet>

<birt:actuateUIOption name=”*enabledMainMenu”* value=*”false”* />

<f:facet name=*"header"*>

<h:panelGroup layout=*"block"* >

<h:outputText value=*”Component Header”* />

</h:panelGroup>

</f:facet>

</birt:actuateWrapper>

‘reportName’ attribute defines path to report file on actuate server

‘emptyReport’ facet defines area will be rendered if ‘reportName’ is null. In current example we display ‘Select a report’ message in the center of the report container.

‘header’ facet defines component header will be rendered above the report container.

<birt:actuateUOption> in our case switches off the main menu for the report. This tag can specify any UI option available via actuate JS API. See here: <http://www.birt-exchange.org/documentation/JavaComponents10/JSAPI-JSdoc/symbols/actuate.viewer.UIOptions.html> for details.

Component will load actuate JS API from actuate server, initialize it and run report specified:

