

CONTENTS INCLUDE:

- What is BIRT
- Design and Runtime Components
- Getting BIRT
- BIRT Report Designers
- BIRT File Types
- Hot Tips and more...

Getting Started with **BIRT**

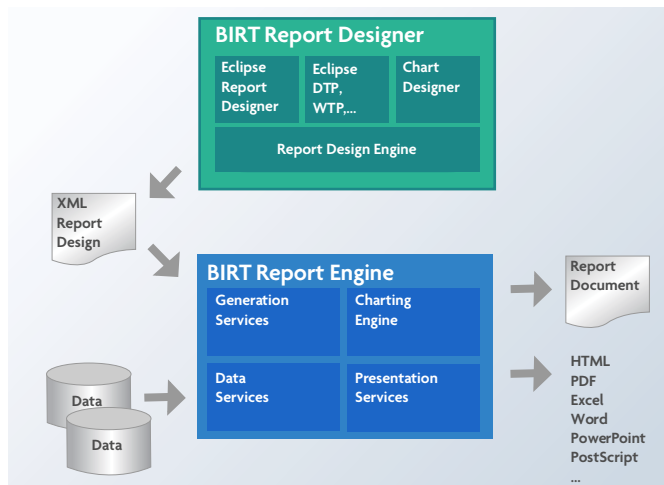
By *Virgil Dodson*

WHAT IS BIRT?

Eclipse Business Intelligence and Reporting Tools (BIRT) is an open source, Eclipse-based reporting system that integrates with your Java/J2EE application to produce compelling reports. BIRT is the only top-level Eclipse project focused on business intelligence. BIRT provides core reporting features such as report layout, data access and scripting. This Refcard provides an overview of the BIRT components focusing on a few key capabilities of the BIRT Designer, BIRT Runtime APIs, and BIRT Web Viewer. This Refcard should be interesting to report designers as well as developers or architects involved in integrating BIRT reports into applications.

DESIGN AND RUNTIME COMPONENTS

BIRT has two main components: a report designer based on Eclipse, and a runtime component that you can add to your application. The charting engine within BIRT can also be used by itself, allowing you to add charts to your application.



GETTING BIRT

Open Source BIRT can be downloaded from <http://download.eclipse.org/birt/downloads/> or <http://www.birt-exchange.com>. There are several different packages containing BIRT depending on your needs.

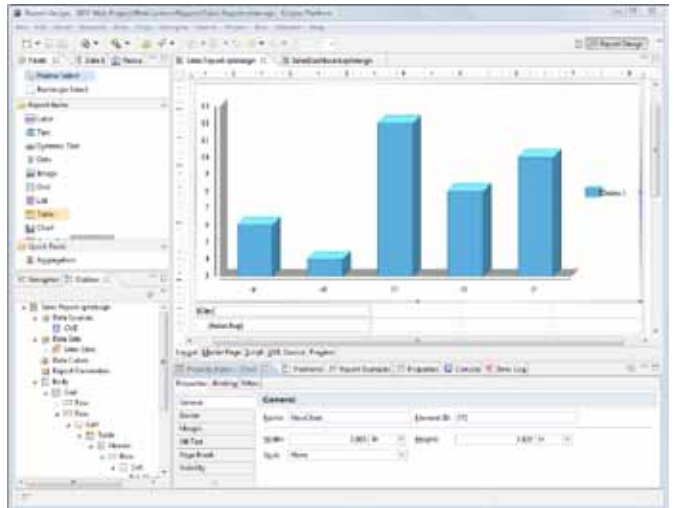
BIRT All-In-One Download	The fastest way to get started designing BIRT reports on Windows. Includes everything you need to start designing BIRT Reports, including the full Eclipse SDK.
BIRT Framework	This download allows you to add the BIRT plug-in to your existing Eclipse environment. (Make sure you check the dependencies and update those too.)

RCP Designer	Simple to use rich client version of the BIRT Report Designer dedicated to creating reports without the rest of the Eclipse development environment.
BIRT Runtime	Deployment components of the BIRT project including a command line example, API examples, and example web viewer.
BIRT Web Tools Integration	Contains the plug-ins required to use the BIRT Web Project Wizard and the BIRT Viewer JSP tag library.



You can also get BIRT into your existing Eclipse environment through the Eclipse Update Manager. Be sure to also select the Data Tools Project when using this approach.

BIRT REPORT DESIGNERS



Get the Most Out of BIRT

- Flash Charts and Graphs
- End User Report Designer
- Report Servers

Try the BIRT-based Product Line at <http://www.birt-exchange.com>

BIRT Exchange
sponsored by Actuate

The BIRT report designers are easy-to-use, visual report development tools that meet a comprehensive range of reporting requirements. The report designers include task-specific editors, builders, and wizards that make it easy to create reports that can be integrated into web applications. All BIRT report designers support:

- Component-based model for reuse
- Ease of use features
- Support for a wide range of reports, layouts and formatting
- Programmatic control
- Data access across multiple data sources

BIRT FILE TYPES

Design File (*.rptdesign)	An XML file that contains the data connection information, report layout and instructions. Created when making a report in the BIRT Designer.
Template File (*.rpttemplate)	Ensures all reports you create start with some common elements such as a company header or predefined styles. The starting point for a BIRT report.
Library File (*.rptlibrary)	Stores commonly used report elements, such as a company logo, so they are managed in one place for all reports.
Report Document (*.rptdocument)	The completed report including layout instructions and data. Can be transformed into final report output, such as HTML, PDF, and XLS.

BIRT DATA SOURCES

BIRT supports a variety of data sources and can be extended to support any data to which you have access. In addition to the list below, BIRT also ships with a connection to the ClassicModels sample database and can be easily extended to connect to your custom data source. BIRT also includes a Joint Data Set which allows you to join data across data sources.

Flat File Data Source	Supports tab, comma, semicolon, and pipe delimited data
JDBC Data Source	Supports connections to relational databases
Scripted Data Source	Allows you to communicate with Java objects or to any data you can get from you application.
Web Services Data Source	Supports connections to a web service. A wizard helps you point at a service through a WSDL and select the data
XML Data Source	Supports data from XML

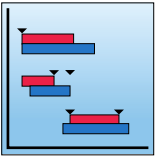
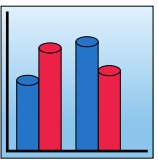
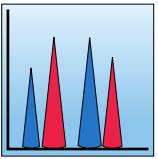
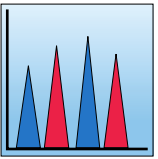
PALETTE OF REPORT ITEMS

Label	Use to include static (or localized) text within a report. Typically for report titles, column headers or any other report text.
Text	Use to include richly formatted text to your report, including the ability to integrate HTML formatting with your dynamic data.
Dynamic Text	Use to integrate your static text with dynamic or conditional data.
Data	Use to include data from your connection in the report.
Image	Use to include images from various embedded sources or dynamic locations.
Grid	Use to define the layout of a report. Can be nested within other grids to support complex layouts.
List	Use to display Data elements from your data source that repeat and creates a new report row for each data set row. Can contain multiple levels of grouping.
Table	Use to display repeating data elements within your report and has support for multiple columns and multiple levels of grouping.
Chart	Use to add rich, interactive charting to your BIRT report.

Cross Tab	Use to display grouped and dynamic data by both the row and column level.
Aggregation	Use to build totals for tables and groups. Includes over 30 built-in functions like COUNT, SUM, MAX, MIN, AVE, RUNNINGSUM, COUNTDISTINCT, and RANK.

CHART TYPES

Bar		Side-by-Side Bar Charts show bars from each series one beside the other. These bars are arranged so that they each have the same width. The width of the bars depends on the number of series being plotted. Stacked Bar Charts show bars stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Bar Charts show bars stacked one over the other in such a way that the total height of the stacked bar (from its lowest to its highest) is 100%
Line		Overlay Line Charts show lines from each series independent of the others. The lines are shown joining the values for the series. Stacked Line Charts show lines stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Line Charts show lines stacked one over the other in such a way that the total height of the stacked lines (from the lowest point to the highest in each unit) is 100%
Area		Overlay Area Charts show areas from each series independent of the others. The areas are shown joining the values for the series. Stacked Area Charts show areas stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Area Charts show areas stacked one over the other in such a way that the total height of the stacked areas (from the lowest point to the highest in each unit) is 100%
Pie		Pie Charts show values as slices of a pie. The size of each slice is proportional to the value it represents. Pie charts for multiple series are plotted as multiple pies, one for each series.
Meter		Standard Meter Charts contain a single dial with region(s). The background of the dial can be divided into regions with different colors. Superimposed Meter Charts contain multiple dials with identical regions. The dials overlap together so that it can represent multiple needles within a single region.
Scatter		Scatter Charts show the values arranged on the plot using the category and value data as coordinates. Each data value is indicated by a marker.
Stock		A Candlestick Stock Chart contains a box with lines extending up and down from the ends. The upper and lower edges of the box are the stock open and close values. The upper and lower points of the line are the high and low values. A Bar-Stick Stock Chart contains a vertical line with two horizontal lines sticking to it. The upper and lower points of the vertical line are the stock open and close values. The two horizontal lines are the high and low values.
Bubble		Bubble Charts show the values arranged on the plot using the category and value data as coordinates. Each data value is indicated by a marker.
Difference		Difference Charts use two fills to represent the positive and negative areas.

Gantt		Standard Gantt Charts contain a connection line with start and end markers.
Tube		Side-by-Side Tube Charts show tubes from each series one beside the other. These tubes are arranged so that they each have the same width. The width of the tubes depends on the number of series being plotted. Stacked Tube Charts show tubes stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Tube Charts show tubes stacked one over the other in such a way that the total height of the stacked tube (from its lowest point to its highest) is 100%
Cone		Side-by-Side Cone Charts show cones from each series one beside the other. These cones are arranged so that they each have the same width. The width of the cones depends on the number of series being plotted. Stacked Cone Charts show cones stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Cone Charts show cones stacked one over the other in such a way that the total height of the stacked cone (from its lowest point to its highest) is 100%
Pyramid		Side-by-Side Pyramid Charts show pyramids from each series one beside the other. These pyramids are arranged so that they each have the same width. The width of the pyramids depends on the number of series being plotted. Stacked Pyramid Charts show pyramids stacked one above the other. The positive and negative values are stacked separately above and below the origin. Percent Stacked Pyramid Charts show pyramids stacked one over the other in such a way that the total height of the stacked pyramid (from its lowest point to its highest) is 100%

Creating your first report:



- Create a new Report Project from the category of Business Intelligence of Reporting Tools. Change to the Report Design perspective.
- File -> New -> Report. Select the template called "My First Report" which launches a cheat sheet containing a step-by-step tutorial assisting you with connecting to data sources, creating data sets, and laying out your report.

LOCALIZATION

BIRT supports internationalization of report data including support for bidirectional text. BIRT also supports the localization of static report elements within a report allowing you to replace report labels, table headers, and chart titles with localized text. BIRT uses resources files with name/value pairs and a *.properties file extension. For example, a file called MyLocalizedText_de.properties can include a line that says "welcomeMessage=Willkommen". To use these files within a BIRT report:

Assign Resource File to entire report	Report -> Properties -> Resources -> Resource File
Assign individual keys to a label	Label -> Properties -> Localization -> Text key

STYLES

Reports designed with the BIRT report designer can be richly formatted with styles that match your existing web application

Built-in Styles	Built-in styles can be shared in a report library for managing style across multiple reports.
CSS Style Sheet	BIRT can import CSS files at design time or reference existing CSS files at run time.

REPORT PARAMETERS

A BIRT report can contain parameters that effect the report. Parameters can be supplied by the user or passed in programmatically from the application. Parameters can be bound to a data set query effectively filtering the report data. Parameters can also be used in expressions and scripting. For example, a parameter can be used with a visibility expression to hide a column or entire table. Available report parameter values can be supplied from a static list, dynamically created from a data set, or even cascading dynamic parameters. For example, selecting a Country presents the available States, and selecting a State presents the available Cities. Related parameters can be grouped for easier user navigation.

Parameter collection from the user can be in several forms:

Text Box	Empty text area where the user can type the values desired
Combo/List Box	A list of values is presented to the user. This list can be provided as a static list or dynamically generated based on a data set query. multiple values can be accepted.
Radio Button	Provides boolean Yes/No, True/False, On/Off of parameters
Custom	Parameters can be passed in programmatically so you can create your own web front end to collect the parameters from the user.

CUSTOMIZATION WITH EXPRESSIONS, SCRIPTING AND EVENTS

BIRT includes out-of-the-box functionality that is available through drag-and-drop or by setting some properties, but also supports more advanced customizations through expressions, scripting, and events. The expression builder in BIRT allows you to do conditional report processing just about anywhere you need to instead of hard coding values. For example, the expression below will display the shipped date for orders that have already shipped, otherwise, it will display the order date.

```
if (dataSetRow["STATUS"] == "Shipped") {
    dataSetRow["SHIPPEDDATE"];
} else {
    dataSetRow["ORDERDATE"];
}
```

Scripting of a BIRT report can be done in either JavaScript or Java depending on your skill set and needs. Scripting allows you to circumvent the traditional event processing of the BIRT report. You can add scripting to report object, data source, and data element event types. Each of these event types has several events that you can overwrite.

For example, you can use scripting to navigate your Java objects and add them to a BIRT Data Set.

```
favoritesClass = new Packages.SimpleClass();
favorites = favoritesClass.readData();
...
var favrow = favorites.get(currentrow);
var Customer = favrow[0];
var Favorite = favrow[1];
var Color = favrow[2];
row["Customer"]=Customer;
row["Favorite"]=Favorite;
row["Color"]=Color;
```

Use scripting to change bar colors on a chart based on plotted data.

```
if (dph.getOrthogonalValue() < 1000) {
    fill.set(255,0,0); //red
} else if (dph.getOrthogonalValue() < 5000) {
    fill.set(255,255,0); //yellow
} else {
    fill.set(0,255,0); //green
}
```

Use scripting to add or drop a report table based on a user parameter.

```
if (params["showOrders"] == false){
    reportContext.getReportRunnable().designHandle.getDesignHandle()
        .findElement("table1").drop();
}
```

Or use scripting to include dynamic images that are based on the report data.

```
if (row["CREDITLIMIT"] <= 0) {
    "down.jpg"
} else {
    "up.jpg"
}
```

REPORT DEPLOYMENT OPTIONS

Once you create your report designs, there are several different ways to generate the report output. Obviously, you can run these reports directly from the BIRT Designer, but you can also run BIRT reports from the command line, generate BIRT reports from your Java application using the BIRT APIs, integrate and customize the example web viewer, or deploy your reports with third-party components and report servers.

APIs

BIRT supplies several APIs and an example J2EE application for generating and viewing reports. The major APIs are the Design Engine API (DE API), Report Engine API (RE API) and the Chart Engine API (CE API). In addition to the APIs, BIRT supports scripting using either Java or JavaScript within report designs.

Design Engine API (DE API)	Use the Design Engine API (DE API) to create a custom report designer tool, or to explore or modify BIRT report designs. The BIRT Designer uses this API. You can call this API within a BIRT script to modify the currently running report design.
Report Engine API (RE API)	Use the Report Engine API to run BIRT reports directly from Java code or to create a custom web application front end for BIRT.
Chart Engine API (CE API)	Use the Chart Engine API to create and render charts apart from BIRT.

BIRT REPORT ENGINE TASKS

There are several tasks supplied by the Report Engine API that can be used to generate report output. A few key tasks are listed below.

IRunAndRenderTask	Use this task to run a report and create the output directly to one of the supported output formats. This task does not create a report document.
IRunTask	Use this task to run a report and generate a report document, which is saved to disk.
IGetParameterDefinitionTask	Use this task to obtain information about parameters and their default values.
IDataExtractionTask	Use this task to extract data from a report document. The BIRT viewer uses this class to extract report data into CSV format.

WORLD'S SIMPLEST BIRT ENGINE EXAMPLE

```
static void executeReport() throws EngineException
{
    IReportEngine engine=null;
    EngineConfig config = null;

    try{
        // start up Platform
        config = new EngineConfig( );
        config.setBIRTHome("C:\\BIRT_231\\birt-runtime-2_3_1\\
            ReportEngine");
        config.setLogConfig("C:\\BIRT_231\\logs", java.util.logging.Level.
            FINEST);
        Platform.startup( config );

        // create new Report Engine
        IReportEngineFactory factory = (IReportEngineFactory) Platform
            .createFactoryObject( IReportEngineFactory.EXTENSION_REPORT_
                ENGINE_FACTORY );
        engine = factory.createReportEngine( config );

        // open the report design
        IReportRunnable design = null;
        design = engine.openReportDesign("C:\\BIRT_231\\designs\\param
            .rptdesign");

        // create RunandRender Task
        IRunAndRenderTask task = engine.createRunAndRenderTask(design);
        // pass necessary parameters
        task.setParameterValue("ordParam", (new Integer(10101)));
        task.validateParameters();

        // set render options including output type
        PDFRenderOption options = new PDFRenderOption();
        options.setOutputFileName("my_report.pdf");
        options.setOutputFormat("pdf");

        task.setRenderOption(options);

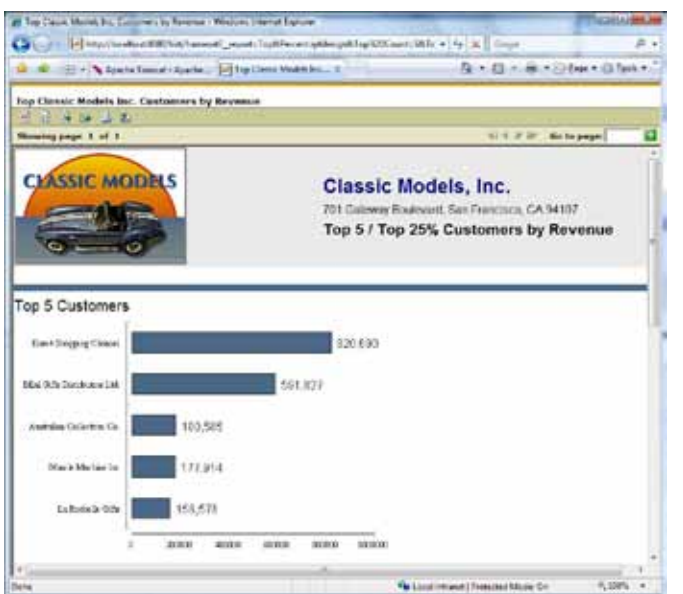
        // run task
        task.run();
        task.close();
        engine.destroy();

    }catch( Exception ex){
        ex.printStackTrace();
    }
    finally
    {
        Platform.shutdown( );
    }
}
```

Hot
Tip

If you are deploying the BIRT Engine within an RCP application you should NOT set the BIRT Home variable or execute the Platform.startup() method.

WEB VIEWER



The BIRT WebViewer is an example application that illustrates generating and rendering BIRT report output in a web application. This viewer demonstrates report pagination, an integrated table of contents, report export to several formats, and printing to local and server side printers.

The BIRT Web Viewer can be used in a variety of ways:

Stand-alone	Use as a pre-built web application for running and viewing reports.
Modify Viewer Source	Use as a starter web application that you can customize to your needs.
RCP Application	Use as a plug-in for your existing RCP application.
Integrated with existing web application	The viewer can be integrated with URLs or BIRT JSP tag library.

The BIRT Web Viewer consists of two main Servlets, the ViewerServlet and the BirtEngineServlet. These Servlets handle three mappings: (/frameset, /run, and /preview).

/frameset	Renders the report in the full AJAX viewer, complete with toolbar, navigation bar and table of contents features. This mapping also generates an intermediate report document from the report design file to support the AJAX based features. For example. http://localhost:8080/viewer/frameset?_report=myreport.rptdesign&parm1=value
/run	Runs and renders the report but does not create a report document. This mapping does not supply HTML pagination, TOC or toolbar features, but does use the AJAX framework to collect parameters, support report cancelling and retrieve the report output in HTML format. For example. http://localhost:8080/viewer/run?_report=myreport.rptdesign&parm1=value
/preview	Runs and renders the report but does not generate a report document, although an existing report document can be used; in this case, just the render operation occurs. The output from the run and render operation is sent directly to the browser. For example http://localhost:8080/viewer/preview?_report=myreport.rptdesign&parm1=value

VIEWER URL PARAMETERS

Below are a few of the key URL parameters available for the viewer. These parameters can be used along with the Servlet mappings, such as, run, frameset, and preview, listed in the Web Viewer section.

Attribute	Description
__id	Unique identifier for the viewer.
__title	Sets the report file.
__showtitle	Determines if the report title is shown in the frameset viewer. Defaults to true. Valid values are true and false.
__toolbar	Determines if the report toolbar is shown in the frameset viewer. Defaults to true. Valid values are true and false.
__navigationbar	Determines if the navigation bar is shown in the frameset viewer. Defaults to true. Valid values are true and false.
__parameterpage	Determines if the parameter page is displayed. By default, the frameset, run, and preview mappings automatically determine if the parameter page is required. This setting overrides this behavior. Valid values are true and false.
__report	Sets the name of the report design to process. This setting can be absolute path or relative to the working folder.
__document	Sets the name for the rptdocument. The document is created when the report engine separates run and render tasks, and is used to support features like table of contents and pagination. This setting can be an absolute path or relative to the working folder.
__format	Specifies the desired output format, such as pdf, html, doc, ppt, or xls.
__Locale	Specifies the locale for the specific operation. Note that this setting overrides the default locale.
__page	Specifies page to render.
__pagerange	Specifies page range to render such as, 1-4, 7.
__bookmark	Specifies a bookmark in the report to load. The viewer automatically loads the appropriate page.

VIEWER WEB.XML SETTINGS

The BIRT Web Viewer has several configuration options. These settings can be configured by modifying the web.xml file located in the WebViewerExample/WEB-INF folder. Below are a few of the key settings available for the viewer.

Attribute	Description
BIRT_VIEWER_LOCALE	This setting sets the default locale for the Web Viewer.
BIRT_VIEWER_WORKING_FOLDER	This is the default location for report designs. If the report design specified in a URL parameter is relative, this path is pre-pended to the report name.
BIRT_VIEWER_DOCUMENT_FOLDER	If the __document parameter is not used, a report document is generated in this location. If this setting is left blank, the default value, webapp/documents, is used. If the __document URL parameter is used and the value is relative, the report document is created in the working folder.
BIRT_VIEWER_IMAGE_DIR	Specifies the default location to store temporary images generated by the report engine. If this setting is left blank, the default location of webapp/report/images is used.
BIRT_VIEWER_LOG_DIR	Specifies the default location to store report engine log files. If this setting is left blank, the default location of webapp/logs is used.
BIRT_VIEWER_LOG_LEVEL	Sets the report engine log level. Valid values are: OFF, SEVERE, WARNING, INFO, CONFIG, FINE, FINER, and FINEST.
BIRT_VIEWER_SCRIPTLIB_DIR	Specifies the default location to place JAR files used by the script engine or JARs containing Java event handlers. These JARs are appended to the classpath. If this setting is left blank the default value of webapp/scriptlib will be used.
BIRT_RESOURCE_PATH	This setting specifies the resource path used by report engine. The resource path is used to search for libraries, images, and properties files used by a report. If this setting is left blank, resources are searched for in the same directory as the report.
BIRT_VIEWER_MAX_ROWS	Specifies the maximum number of rows to retrieve from a dataset.
BIRT_VIEWER_PRINT_SERVERSIDE	This setting specifies whether server side printing is supported. If set to OFF the toolbar icon used for server side printing is removed automatically. Valid values are ON and OFF.

VIEWER JSP TAG LIBRARY

The BIRT Web Viewer includes a set of tags to make it easy to integrate BIRT reports into browser pages. These tags are available from the BIRT Web Tools Integration download. Below are a few the key JSP tags and a description of their usage.

Tag	Description
viewer	Displays the complete Viewer inside an IFRAME. This tag allows you to use frameset and run Servlet mappings. The AJAX Framework is used.
report	Displays the report inside an IFRAME or DIV tag without the Viewer. This tag allows you to use preview mapping and does not create an rptdocument. The AJAX Framework is not used.
param	Used to set parameter values when using the viewer or report tags. This tag must be nested within the viewer or report tag.
value	Used to specify multiple values for a given param tag.
parameterPage	Used to launch the BIRT parameter dialog or to create a customized parameter entry page. This tag can be used with the frameset, run, or preview mappings to launch the viewer after the parameters are entered.
paramDef	Used within a parameterPage tag to retrieve pre-generated HTML for specific parameter control types such as radio, checkbox, dynamic or cascaded parameters.

SIMPLE VIEWER JSP TAG EXAMPLE

```
<%@ taglib uri="/birt.tld" prefix="birt" %>
<birt:viewer
id="birtViewer" pattern="preview"
reportDesign="TopNPercent.rptdesign"
height="600" width="800"
format="html"
title="My Viewer Tag"
isHostPage="false"
showTitle="true" showToolBar="true"
showNavigationBar="true"
showParameterPage="true">
</birt:viewer>
```

BIRT REPORT OUTPUT FORMATS

In addition to delivering paginated report content to a web browser, BIRT also supports several other output formats. These formats listed below are support by both the Report Engine API as well as the BIRT Web Viewer.

Paginated web output	An example web viewer is included with BIRT allowing for on demand paginated web output.
DOC	Microsoft Word Document.
HTML	Suitable for creating HTML pages of report data deployable to any server.
PDF	Adobe PDF output suitable for emailing or printing.
Postscript	Output can be directed to a printer that supports postscript.
PPT	Powerpoint output.
XLS	Excel file output.

BIRT EXTENSION POINTS

The APIs in BIRT define extension points that let the developer add custom functionality to the BIRT framework. These extensions can be in the form of custom data sources, report

items, chart types, output formats, and functions. Once implemented, these custom extensions will show along with the built-in types. For example, you can create a custom report item, like a rotated text label, that will show up in the BIRT Palette along with the existing items.

Data Sources	BIRT supports the Open Data Access (ODA) architecture, which means it can be extended to support custom data sources.
Functions	BIRT allows you to create custom functions that extend those available in BIRT Expressions.
Report Items	Report Items can be extended, allowing you to create your own custom report item.
Chart Types	Additional chart types can be added to BIRT as plug-ins.
Output Emitters	BIRT can be extended to include your own custom output type. For example, a simple CSV emitter exists and can be added to BIRT.

ADDITIONAL BIRT RESOURCES

Eclipse BIRT Project Site	http://www.eclipse.org/birt
BIRT Exchange Community Site	http://www.birt-exchange.com
Submitting/Searching BIRT Bugs	http://bugs.eclipse.org/bugs/enter_bug.cgi?product=BIRT
Online BIRT Documentation	http://www.birt-exchange.com/modules/documentation/

ABOUT THE AUTHOR



Virgil Dodson is a Developer Evangelist at Actuate Corporation and blogger/forum moderator at the BIRT Exchange community site. Virgil has over 13 years experience as a software developer. For the past 6 years he has helped Java developers get started with Actuate's embedded reporting products. He holds a Bachelor of Science degree in Computer Information Systems from DeVry.

RECOMMENDED BOOK



Topics Discussed Include:

- Installing and deploying BIRT
- Deploying a BIRT report to an application server
- Understanding BIRT architecture
- Scripting in a BIRT report design
- Integrating BIRT functionality in applications
- Working with the BIRT extension framework

BUY NOW

books.dzone.com/books/integrating-extending-birt

Professional Cheat Sheets You Can Trust

"Exactly what busy developers need: simple, short, and to the point."

James Ward, Adobe Systems

Upcoming Titles

- Scrum
- BPM & BPMN
- IntelliJ IDEA Update
- db4o
- Agile Adoption
- JavaFX
- JSF 2.0

Most Popular

- Spring Configuration
- jQuery Selectors
- Windows Powershell
- Dependency Injection with EJB 3
- Netbeans IDE JavaEditor
- Getting Started with Eclipse
- Very First Steps in Flex

Download Now
Refcardz.com



DZone communities deliver over 4 million pages each month to more than 1.7 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheatsheets, blogs, feature articles, source code and more. "DZone is a developer's dream," says PC Magazine.

DZone, Inc.
1251 NW Maynard
Cary, NC 27513
888.678.0399
919.678.0300
Refcardz Feedback Welcome
refcardz@dzone.com
Sponsorship Opportunities
sales@dzone.com



\$7.95