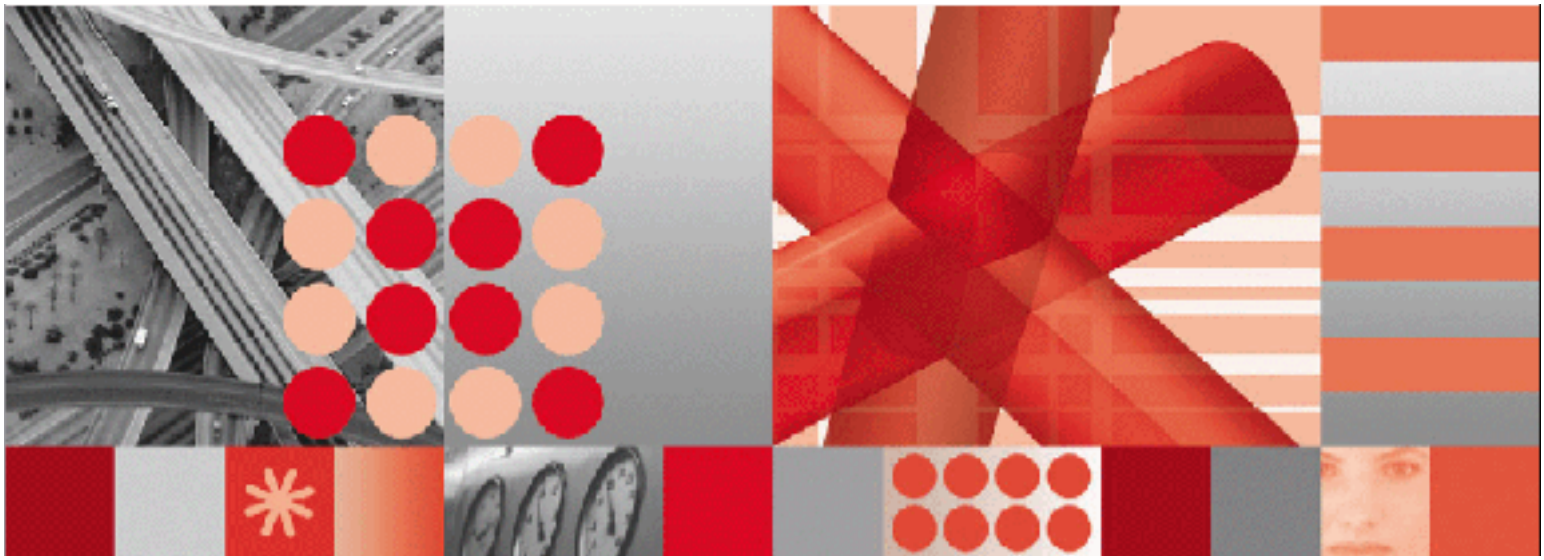
Tivoli.

IBM

IBM Tivoli Change and Configuration Management Database 7.1.1
IBM Maximo Asset Management 7.1
IBM Tivoli Service Request Manager 7.1
IBM Tivoli Asset Management for IT 7.1



Report Developer Guide

Note
Before using this information and the product it supports, read the information in "Notices" on page 41.

This edition applies to version 7, release 1, modification 1 of IBM Tivoli Change and Configuration Management Database, version 7, release 1, modification 0 of IBM Maximo Asset Managment, IBM Service Request Manager, and IBM Tivoli Asset Managment for IT and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About This Publication

This guide describes report set up, creation, and design tasks for the Report Developer. It also describes advanced reporting features.

Intended Audience

This guide is for Report Developers who will be performing the following tasks:

- creating a BIRT Report
- importing and exporting report design files
- loading and configuring BIRT Report Designer
- setting up localized reports
- using advanced BIRT reporting features
- working with report parameters

Loading and Configuring BIRT Report Designer

1

This guide is written for Java programmers with practical knowledge of BIRT (Business Intelligence and Reporting Tools) Report Designer and the Eclipse environment. Eclipse is an open source community whose projects are focused on building an open development platform.

This chapter contains the following sections:

- BIRT Reporting Tool
- BIRT Report Designer Files
- BIRT Report Designer Source Files
- BIRT Report Designer File Structure
- System Setup for Design-Time Files
- BIRT Report Designer Configuration

BIRT Reporting Tool

IBM uses BIRT as its reporting tool. BIRT is an Eclipse-based open source reporting system for Web applications.

This section defines BIRT Components.

Defining BIRT Components

IBM Corporation integrates the following BIRT components:

- BIRT Report Engine
- BIRT Report Designer

BIRT Report Engine

The BIRT Report Engine is an Eclipse-based tool for viewing reports.

The BIRT Report Engine produces a report in HTM, CSV, or PDF format so you can view that report in your browser.

Installing BIRT Report Engine

The BIRT Report Engine is embedded on your system. When you installed your application software, you also installed the BIRT Report Engine.

BIRT Report Designer

Use BIRT Report Designer for report development. The designer uses an Eclipse-based interface and creates reports that integrate into Web applications.

Installing BIRT Report Designer

For information on installing the BIRT Report Designer, access IBM developerWorks.

NOTE If you do not have JDK 1.5 (5.0), you must install it.

BIRT Report Designer Files

BIRT Report Design files are xml files, noted by the extension .rptdesign. BIRT Reports can contain single or multiple files. The files are categorized as either library files or resource files.

BIRT library files are also xml files and have the extension .rptlibrary. BIRT library files contain code that is used multiple times for items such as font type, size, page numbers, and time stamp. The individual files can be reused multiple times in BIRT report designs and templates.

Resource Files contain items such as images or external files. Resource files can be used by either report design files or library files. Property files are also resource files. Many different report designs use the same property file.

The xml of the BIRT Report details which library files and resource files the report requires. Without these files, the BIRT report does not execute.

BIRT Report Designer Source Files

IBM delivers multiple reports to you so you can analyze specific problems, view data in a manner that is not available in a specific application, or print out record results for action or record keeping.

NOTE Refer to the Information Center for a complete list of out-of-the-box reports IBM ships to you with this product.

In addition, you can customize the reports IBM delivers to you to either add or delete a field, add your corporate logo, or change the sorting. You can also create your own customized reports to meet your specific business requirements.

Therefore, IBM includes the report design files so you can customize them or review the content as examples for creating your own reports.

BIRT Report Designer File Structure

The BIRT Report Designer file structure contains the following subfolders:

- libraries
- reports
- scriptlibrary
- templates
- tools

The following sections describes these subfolders in detail.

Libraries Folder

The libraries folder contains all application properties files. The properties file contains the report title and label values. This file is required to import a report and for localization.

The libraries folder also contains the following system library files:

- libraries.xml
- MaximoSystemLibrary.rptlibrary

NOTE Do not modify the two system library files in the libraries folder. If you modify these files, you may corrupt your reports.

Reports Folder

Each application in your system corresponds to a folder in your reports subfolder. Your license controls which applications you can see.

The report design file structure for BIRT contains the following path:

`<Product_root>\reports\birt\reports\`***<application folder name>***

Each of the application folders corresponds to an application. The application folder contains report design files. For example, the People folder represents the People application. The People folder contains the person_details.rptdesign file that represents the Person Details report.

The People folder also contains a report.xml file. This file is common to all reports in this subfolder. For example, all the reports in the People folder use the reports.xml file, also in the People folder. The exact content of the file varies by application.

The report.xml file contains information necessary for you to import a report, such as the file name, description, and parameter information.

Scriptlibrary Folder

The scriptlibrary folder contains scripting code. The system requires this code for you to enable IBM reports.

Templates Folder

The templates folder contains IBM report templates. These are the only templates you should use when you create a new report. For more information on templates, see "Understanding Report Templates," on page 9.

Tools Folder

The tools folder contains tools for you to import and export reports from your client machine to the system database. For more information on importing and exporting, see Chapter 6.

System Setup for Design-Time Files

This section contains actions you must complete to setup and install Design-Time files on your system. After completing this section, you can begin customizing or creating your reports.

Prerequisites to Installation

To create or customize BIRT reports, you must have both Eclipse and BIRT Report Designer downloaded on your client machine. This release uses and supports the following versions:

- BIRT Report Designer version 2.1.2.

  For information on installing the BIRT Report Designer, access IBM developerWorks.

- Eclipse version 3.2.2

  If you do not already have Eclipse loaded on your machine, contact your System Administrator for downloading information.

Installing Design Time Files

The following instructions describe how to install Design-Time files.

In the file paths below, v20060926-0959 is a version number and may be different depending on the your version.

1. Locate the compiled classes for report scripting from the <Product_root> location.

   ```
   <Product_root>\reports\birt\scriptlibrary\classes
   ```

2. Copy the entire \com folder to the following location:

   ```
   eclipse\plugins\org.eclipse.birt.report.viewer_2.1.2.v20060926-
   0959\birt\WEB-INF\classes
   ```

If necessary, create the `\classes` folder.

3 Copy mxreportdatasources.properties using the same source and destination folders from Step 1.

4 Edit the file as follows:

   a Set the URL, driver, username, password, and schemaowner properties, following the sample format provided.

   b Change #<DataSourceName> to maximoDataSource.

Additional Information

As the Report Developer, you will need the mxreportdatasources.properties file for connection information.

```
#<DataSourceName>.<propertyName>=value

# driver for ORACLE
# oracle.jdbc.driver.OracleDriver
# sample url for ORACLE
# jdbc:oracle:thin:@<HOST>:<PORT>:<SID>
# sample schemaowner for ORACLE
# maximo

# driver for SQLServer
# com.inet.tds.TdsDriver
# sample url for SQLServer
# jdbc:inetdae7a:hostname:port?database=dbname&language=us_
# english&nowarnings=true
# sample schemaowner for SQLServer
# dbo

# driver for DB2
# com.ibm.db2.jcc.DB2Driver
# sample url for DB2
# jdbc:db2://localhost:50000/dbalias
# sample schemaowner for DB2
# maximo

#<DataSourceName>.url=value
#<DataSourceName>.driver=value
#<DataSourceName>.username=value
#<DataSourceName>.password=value
#<DataSourceName>.schemaowner=value
```

The values ***localhost*** and ***dbalias*** (the name of your database) are variables. You can check your database information by selecting **Start Menu> Programs>IBM DB2>Set-up Tools>Configuration Assistant**.

maximoDataSource is the current default set for all reports.

5 Copy the following JDBC driver(s):

   `<Product_root>\applications\maximo\lib` or from the database-specific zip file

   to:

   `eclipse\plugins\org.eclipse.birt.report.viewer_2.1.1.v20060926-0959\birt\WEB-INF\lib`

These drivers are only used by BIRT Report Designer.

NOTE For Oracle, use oraclethin.zip.
For SQL Server, use opta.jar.
For DB2, use db2jcc.jar and db2jcc_license_cu.jar

You have completed installing your Design-Time files.

BIRT Report Designer Configuration

Complete the following steps to configure the BIRT Report Designer.

NOTE Create a shortcut to eclipse .exe. Be sure to include the path pointing to the JDK 1.5 install:

C:\eclipse_download\BIRT\eclipse\eclipse.exe ***-vm "C:\program files\IBM\Java50\bin\javaw.exe" vmargs -Xmx512m***

1. Open up the BIRT Report Designer in Eclipse.

   NOTE Use forward slashes or the Select button when specifying the folder paths in Eclipse.

2. Specify the resource folder location.

   a. Select **Window>Preferences**

   b. Expand Report Design and select Resource.

   c. Specify the report library location:

   ```
   <Product_root>\reports\birt\libraries
   ```

3. Specify the templates folder location.

   a. Select **Window>Preferences**

   b. Expand Report Design and select Templates.

   c. Specify the templates location:

   ```
   <Product_root>\reports\birt\templates
   ```

4. Disable the Comment template by following these steps:

   a. Open the Preferences window by selecting **Window>Preferences**.

   b. Expand the Report Design folder and select the Comment template.

   c. Clear the flag from the **Generate comment when creating a report design** field.

5. Import the report project.

a. Select **File>Import**

b. Expand the General folder and select Existing Projects into Workspace.

c. Click **Next.**

d. Browse to the report files location:

```
<Product_root>\reports\birt\reports
```

e. Select the project ***YourReports***. Click **Finish**.

6. Import the library project. This step is optional and applies only if you are changing the libraries.

a. Select **File>Import**

b. Expand the General folder and select Existing Projects into Workspace. Click **Next**.

c. Browse to the report library location from Step 2.

d. Select the project ***YourReportLibraries.*** Click **Finish**.

7. Import the report script project. This step is optional and only applies if you are building the report script classes.

a. Check out the all source files in the following location:

```
<Product_root>\applications\maximo\maximouiweb\webmodule\WEB
-INF\birt
```

b. Select **File>Import.**

c. Expand the General folder and select Existing Projects into Workspace, then **Next**.

d. Browse to the following location:

```
<Product_root>\applications\maximo\maximouiweb\webmodule\WEB
-INF\birt\script
```

e. Select the project ***YourReportScriptLibrary.*** Click **Finish**.

Creating a BIRT Report 2

This chapter describes how to write your first BIRT report using BIRT Report Designer. This chapter contains the following sections:

- Preparing to Write your First Report
- Understanding Report Templates
- Creating a Grouped BIRT Report

Preparing to Write your First Report

Prior to creating a new report in BIRT Report Designer, IBM suggests you develop and test all required queries in your database query tool. BIRT does not validate SQL and a query tool may provide clearer error messages.

Additionally, you will want to know the datatypes of all the fields in your report. Determine data types (maxtypes) of the fields used in your queries through either of the following methods:

- query maxattribute directly as shown in the following SQL statement:

```
select attributename, maxtype from maxattribute where objectname
= 'WORKORDER' order by attributename
```

- open the Database Configuration application and use the **Type** field on the Attributes tab for your selected object.

Understanding Report Templates

IBM has developed a number of report templates for you to use when you create new reports. To see these templates, from the BIRT Report Designer select **File>New>Report** or choose New Report from the drop-down list. Although many templates will appear, select only those IBM specifically creates and enables for you. These templates are identified by the words, “Tivoli Maximo”. These templates contain the necessary code for integration and provide a consistent look and feel between reports.

The following table describes these templates:

Name	Description
Tivoli Maximo Subreport Template	Template creates a report with one or more subreports. Each subreport may have different data fields.
Tivoli Maximo Grouped Report Template	Template creates a report with one or more groups. Each group has identical data fields.
Tivoli Maximo List Report Template	Template creates a report with a single header. The report typically has multiple rows, depending on the amount of data available.
The following templates include graphical elements, in addition to the standard reporting formats. Each template includes three chart types (a pie chart, a bar chart, and a line chart). Select one chart type for your report and delete the other two.	
Tivoli Maximo Subreport Chart Template	Template creates a report with one or more subreports. Each subreport can have different data fields.
Tivoli Maximo Grouped Chart Report Template	Template creates a report with one or more groups. Each group has identical data fields.
Tivoli Maximo List Chart Report Template	Template creates a report with a single header. The header has multiple rows of data.

Creating a Grouped BIRT Report

This section describes how to create a grouped BIRT report. The report this example uses is the Database Configuration report. This filename for this report is listtabl.rptdesign. You can find this file in the following location:

```
<Product_root>\reports\birt\reports\configur
```

1 Develop and test all database SQL queries in a query tool.

Create a Report

2 Select **File>New>Other**. Expand the Business Intelligence and Reporting Tools folder.

3 Click **Report,** then **Next**. Assign the report to a project and name it.

4 Click **Next**. Select the Tivoli Maximo Grouped Report Template.

5 Click **Finish**. The report design appears.

Specify the Query

6 In Data Explorer view, select the data set. Click the Script tab. Select the Open method from the drop-down list.

7 Copy your query from the query tool and paste it into the method body under the existing sample query. Format your query to match the sample.

Create the Output Columns

8 Double-click the data set to open the Properties dialog box. In the Output Columns editor, enter a column for each field in your query and any tabulated columns. If you leave the Open Method visible as you do this, you can use it

for reference on the columns. You can give the output columns different names than the database fields or keep the names the same.

See the data type for each output column based on the maxtype of the field, as specified in the BIRT Data Type Mapping chart. For more information, see "BIRT Data Type Mapping", on page 16.

NOTE If you need more than one data set, you can copy the existing data set before continuing with this procedure. Additional data sets are often necessary when you are creating subreports.

Fetch the Data Rows

9 On the Script tab, choose the Fetch method from the drop-down list. Following the example provided in the template, add a line for each column that retrieves the value of the data set and updates the output column with that value.

Use the appropriate method based on the data type of the field, following the BIRT Data Type mapping.

Turn on debugging

10 Click Outline view, select the report, then the Script tab. Be sure you select the initialize method from the drop-down list. Add the debug lines.

Add Report Parameters

11 Go to Data Set view, right-click on Report Parameters, and select **New**. Fill in the data. Click **OK**.

Incorporate the report parameter into your SQL clause in the Data Set Open method

12 Add an optional unbound parameter.

Use the createParamWhereClause() method to create the appropriate SQL. The following is an example of text you can add to the open method:

```
// Build the where clause with the unbound parameter DBTable
var where = params["where"];
if (params ["DBTable"].value)
where = where + " and " +
MXReportSqlFormat.createParamWhereClause ("mo.objectname",
params["DBTable"]);
```

Create the Report Layout

13 Select the Layout tab and add the report title.

14 If the report does not require the display of the input parameters in the header, delete the grid containing the parameters.

Add Detail Data

15 From the Data view, expand the data set. Drag and drop output columns to the layout detail row. If you need to add more columns to the table, select the Table tab, then the Column tab. To avoid disruption to the layout, always insert columns to the left.

16 Check the appearance of your layout.

Add column headings

17 Go to the Palette view. Drag and drop an existing label to other column headings. For example, if the first column heading is labeled Status, you can add that label name to the other columns.

Add Group Details

18 Open the Outline view and click **Body>Table> Groups**. Double-click on the Table Group entry to bring up the group editor. Add the column to group on. Click **OK** to add the group details.

Preview the Report

19 Click the Preview tab. The report runs with the default parameters.

20 To specify parameters other than the default, click Show Report Parameters and enter valid values.

21 Add group labels.

Associate the report with a label localization properties file that stores the key/value pairs for the labels

22 To complete this task, complete the following steps:

a Select the Outline view.

b In the Properties tab, select Resources.

c If a properties file exists for your application, select it. If not, enter the name of the new properties file. Be sure the name you enter matches the name of your application folder in reports. The system creates the property file in the following location:

```
...\reports\BIRT\libraries
```

The properties file name for the Database Configuration Report is configur.properties.

d Click **OK**.

Select a label to localize in the layout editor

23 Complete the following steps:

a On the Properties tab, select the Localization parameter.

b Click **Detail** ("…") beside the **Text key** field.

c Enter either a new key value pair or select an existing one.

d Click **OK**.

Repeat this step for all labels in your layout. For more information on enabling localized reports, see Chapter 4, "Setting up Localized Reports."

Set column lengths to ensure proper display in PDF files

24 Select a tab and the corresponding column of the table. You can resize the column to set the width to a specific value. Do this for all table and grid columns.

View the report in PDF format

25 To view the report in PDF format, select the view report icon. Click **View report as PDF**.

26 The report appears. Go back to the layout editor and adjust columns as needed.

27 Create the report import file, reports.xml. There is one import file in each application report directory.

For additional information on the reports.xml file, see Chapter 6.

The following text represents the reports.xml file for the Database Configuration report:

```
<!--
This file is used to define what reports need to be imported for the
CONFIGUR application
-->

<reports>
<report name="listtabl_1.rptdesign">
<attribute name="filename">listtabl.rptdesign</attribute>
<attribute name="description">Database Configuration</attribute>
<attribute name="toolbarlocation">NONE</attribute>
<attribute name="toolbaricon">NONE</attribute>
<attribute name="toolbarsequence"></attribute>
<attribute name="attacheddoc">0</attribute>
<attribute name="norequestpage">0</attribute>
<attribute name="detail">0</attribute>
<attribute name="reportfolder">CONFIGUR</attribute>

<parameters>
<parameter name="DBTable">
<attribute name="attributename"></attribute>
<attribute name="lookupname"></attribute>
<attribute name="sequence"></attribute>
<attribute name="labeloverride"> Database Table</attribute>
<attribute name="defaultvalue"></attribute>
<attribute name="required">0</attribute>
<attribute name="hidden">0</attribute>
<attribute name="operator"></attribute>
<attribute name="multilookup">0</attribute>
</parameter>
</parameters>
<resources>
<resource>
<reference>configur.properties</reference>
<filename>${libraryfolder}/configur.properties<
/filename>
</resource>
</resources>
</report>

</reports>
```

For more information on creating the import file, see “Registering a Report with Application Toolbar Access,” on page 22.

Import the report

28 Open a command prompt and change the directory to the following location:

```
<Product_root>\reports\birt\tools
```

29 To import all reports in the Database Configuration application, run the following command:

```
importreports configur
```

For more information on importing reports, see Chapter 6.

Log into the system to verify that you loaded your report

30 Sign in to your system as the Report Administrator and open the Report Administration application. In the **Report File Name** field, type Database to filter to the Database Configuration report.

31 Select your report and the report details tab appears. If required, apply security privileges.

NOTE You set Report Level security on the Security tab. You set Application Level security though the Set Application Security action.

32 Prior to running the report the first time, click **Generate Request Page**.

33 To preview your report, click **Preview**. Enter a value for the Parameter database table or click **Submit** to run the report against all records.

The grouped BIRT report appears in the report browser.

Advanced BIRT Reporting Features

3

This chapter contains information on advanced reporting features to assist you in using BIRT Report Designer. This chapter covers the following topics:

- Reviewing Database Types, SQL Information, and Update Functionality
- Executing Additional Queries
- Formatting the Report
- Creating a Hyperlink from one Report to Another
- Debugging
- Registering a Report with Application Toolbar Access
- Registering a Report for Multiple Applications

Reviewing Database Types, SQL Information, and Update Functionality

This section contains the following subsections to assist you in creating a BIRT report:

- BIRT Data Type Mapping
- SQL Design Notes
- Date Formatting
- Testing for Null
- Scalar Functions
- Conditional SQL
- Adding Database Update Functionality

BIRT Data Type Mapping

The following chart shows the database type, the corresponding BIRT Data Type, and the method used within the BIRT Designer to retrieve its value. This chart defines the data fields used during the Fetch method and how the Data Set Output columns are populated.

Database Type	BIRT Data Type	Data Set Method used to Retrieve
ALN, CLOB, GL, LONGALN, LOWER, UPPER	String	getString(String attributeName)
YORN (See following section on the YORN database type)	String	getBooleanString(String attributeName)
DATE, DATETIME, TIME	DateTime	getTimestamp(String attributeName)
AMOUNT, DECIMAL, DURATION (See the following section on the DURATION database type.)	Decimal	getDouble(String attributeName)
FLOAT	Float	getFloat(String attributeName)
DURATION (See the following section on the DURATION database type.)	String	getDuration(String attributeName)
INTEGER, SMALLINT	Integer	getInteger(String attributeName)

NOTE The following Database Types are not supported in reports: BLOB, CRYPTO and CRYPTOX.

YORN Database Type

YORN fields are stored in the database as numbers (0 and 1) but are presented in the system as localized text. The `getBooleanString(String attributeName)` method performs both tasks: retrieves the numeric value and translates it to the appropriate text. You also can obtain the translated value from the integer using `getBooleanString(int intValue)`.

Duration Database Type

The database stores DURATION as a number (fractional hours) but in the system presents DURATION as a string in the format HH:MM. The `getDuration` method returns the formatted string. If you require the numeric value instead, use the `getDecimal` method.

An additional utility method, `MXReportUtil.getDuration(String attributeName)` allows you to convert the string from double to single.

SQL Design Notes

Be sure to use database-independent SQL whenever possible. Use ANSI SQL join syntax (left outer, right outer) . Also, use ANSI functions such as CASE and COALESCE instead of proprietary functions such as DECODE and ISNULL. Do not use owner qualification (MAXIMO.workorder) and reference all database objects in lower-case.

Date Formatting

The system provides static MXReportSqlFormat methods to support date formatting. All return strings of JDBC-formatted date functions can be used in report SQL statements for all supported databases. The following statements provide an example:

```
"where actualdate <=" + MXReportSqlFormat.getCurrentDateFunction()
evaluates to:
where actualdate <= { ts '2007-04-01 00:00:00' }
```

`getCurrentDateFunction()` – current date
`getCurrentTimestampFunction()` – current date and time
`getDateFunction(Date d)` – date based on date input
`getTimeFunction(Date d)` – time based on date input
`getTimestampFunction(Date d)` – date and time based on date input
`getStartDayTimestampFunction(Date d)` – date based on date input, with time component set at start of day (for start date parameters)
`getEndDayTimestampFunction(Date d)` – date based on date input, with time component set at end of day (for end date parameters)

Testing for Null

The COALESCE function is supported on all database types and can be used directly in the query. If you must use a proprietary null conversion function, IBM provides the following data set method:

`maximoDataSet.getNullValueFunction``(String param, String nullVal)` – This method returns NVL, ISNULL, or COALESCE depending on the database type.

For example:

```
"select " + maximoDataSet.getNullValueFunction("parent", "wonum")
```

evaluates to:

Oracle:`select nvl(parent, wonum)`
DB2:`select coalesce(parent, wonum)`
SQL Server:`select isnull(parent, wonum`

If `nullVal` is a string literal, place it in single quotes:

```
"select" + maximoDataSet.getNullValueFunction("parent",
"'NONE'")
```

Be careful using string literals this way since the system does not localize them.

Scalar Functions

The method `MXReportSqlFormat.getScalarFunction(functionName, variable parameters)` returns a JDBC scalar function based on the function name and a variable list of parameters. You can use this method to access database functions in a database independent manner, as suggested in the JDBC specification for commonly used functions.

Conditional SQL

When you must use database-specific SQL, conditionally create the query based on the database type. To do so, use the following methods on the data set:

`Boolean isOracle(), Boolean isSQLServer(), Boolean isDB2()`
`String getDatabaseProductName()` – Returns the database name from the connection

If only a few lines are different, you can isolate those lines and set them as appropriate, as shown in the following example:

```
var dbText = "";
if (maximoDataSet.isOracle())
{
dbText = <Oracle-specific syntax>;
}
else
{
dbText = <DB2 & SQL Server syntax>;
}
sqlText = "select wonum, description, " + dbText + " from
workorder";
```

Adding Database Update Functionality

This section describes how to add database update functionality to reports. With this functionality, the reports should be able to execute Database SQL UPDATE/ INSERT/DELETE statements against a specific data source:

NOTE The following examples illustrate the SQL UPDATE statement, but you can also use SQL INSERT and DELETE.

1. Execute the update within a DataSet (any of the open/describe/fetch/close/ beforeOpen/beforeClose/onFetch/afterOpen/afterClose methods).

```
myTxn =
MXReportTxnProvider.create(this.getDataSource().getName());
myStmt = myTxn.createStatement();
myStmt.setQuery("update ... set .... = ....");
myTxn.save();
```

2. Execute the update outside of a DataSet.

```
myTxn = MXReportTxnProvider.create("MAXIMODATASOURCE");
myStmt = myTxn.createStatement();
myStmt.setQuery("update ... set .... = ....");
myTxn.save();
```

3. Execute multiple updates.

```
myTxn =
MXReportTxnProvider.create(this.getDataSource().getName());
myStmt1 = myTxn.createStatement();
myStmt1.setQuery("update ... set .... = ....");
myStmt2 = myTxn.createStatement();
myStmt2.setQuery("update ... set .... = ....");
myTxn.save();
```

4 Execute with parameters.

```
myTxn =
MXReportTxnProvider.create(this.getDataSource().getName());
myStmt = myTxn.createStatement();
myStmt.setQuery("update ... set .... = ?, ... = ?");
myStmt.setQueryParameterValue(1, new Integer(0));// using
Integer object as an example. Also note the parameter index
starts form 1
myStmt.setQueryParameterValue(2, "MyValue");// using String
object as an example
myTxn.save();
```

Executing Additional Queries

You can run additional queries in both the Open and Fetch methods. Each method can have one or more additional queries returning one or more fields. The following sections describe these types of queries.

Queries in the Fetch Method

A single SQL Statement does not always provide all the data fields for a report. Use the main query to populate most of the output columns and additional queries to retrieve the remaining data, as shown in the following example:

```
if (!maximoDataSet.fetch())
return (false);

// Set output columns from main query
row["assetnum"] = maximoDataSet.getString("assetnum");

// Execute secondary query
classStrucDataSet =
MXReportDataSetProvider.create(this.getDataSource().getName(),
"class");
classStrucDataSet.open();

sqlText = "select description from classstructure where
classstructureid=? ";
classStrucDataSet.setQuery(sqlText);

// Use value from main query as foreign key in secondary query
classStrucDataSet.setQueryParameterValue(1,
maximoDataSet.getString("classstructureid"));

if (classStrucDataSet.fetch())
{
// Set output columns from secondary query
row["description"] = classStrucDataSet.getString
("description");
}
// Always close the data set
classStrucDataSet.close();

return(true);
```

Queries in the Open Method

These types of queries are useful for performing lookups of static values that the report needs. You can insert the values into global variables and use them throughout the report.

For example, there are three cost fields in invcost, and the correct one to use for a given item depends on the site for the item. Sites are assigned a default cost type in maxvars. Since the report has a single-value parameter on Site, you can look up the site-specific cost once and use it throughout the report.

The process is the same as described for the Fetch method, except that it uses variables instead of output columns to store the results.

Linking Result Sets

When you run additional queries in the Fetch method, you usually must link them to the current data row. You do this either by directly including the value or by using data set parameters.

The following text is from the previous example on running queries in the Fetch method:

```
sqlText = "select description from classstructure where
classstructureid=?";
classStrucDataSet.setQuery(sqlText);

// Use value from main query as foreign key in secondary query
classStrucDataSet.setQueryParameterValue(1,
maximoDataSet.getString("classstructureid"));
```

In this example, you set the parameter to the value of a field in a data set. The field is a string so you use the data set `getString` method. You can also use the `getTimestamp` method, but not the fetch methods that return primitive data types. Instead of the fetch method, use the following method:

```
getDoubleObject(String attributeName)
getFloatObject(String attributeName)
getIntegerObject(String attributeName)
```

The other common situation when you must link result sets is when linking subreports. Subreport queries are similar to Open method queries in that they both execute each time you fetch a record in the main query. The difference is that subreport queries have their own data sets. The contents of the subreport can be contained in an independent child table, which is bound to the secondary data set and nested in a cell in the parent table.

To link a subreport query to a main query, include the linking fields (foreign keys) in the main query. In the subreport query, reference the linking fields using the “rows” variable as shown in the following example:

```
sqlText = "select laborcode, craft from labtrans where refwo = '"
+ rows[0][ "wonum"] + "' and siteid = '" + rows[0]["siteid"] + "'";
```

Formatting the Report

Drag the fields from the data set to the report. Set a fixed width for each column, otherwise the report will not format correctly when you print it in PDF format from the report browser.

For example, if you use 8.5" x 11" paper and plan a margin of .5" on either side, you would space your columns based on 10" (since IBM delivers report templates in landscape format). For a five column report, you can plan 2" for each column.

Set any parameter display fields. Do not drag parameters from the Data Explorer into the report. Instead, drag a Data element and set the Value Expression to the parameter.

If there are groups, set the keys.

Creating a Hyperlink from One Report to Another

Many reports contain hyperlinks so you can drill down to Detail reports. In the following example, you drill down from the Job Plan List report (jobplan.rptdesign) to the Job Plan Details Report (jobplan_print.rptdesign). You can use this as an example when you create your own hyperlinks.

When you specify a report to link to, BIRT validates that the report exists and reads its parameter information. Therefore, before you can set the hyperlink properties for a field, you must create a placeholder .rptdesign for the target report. The placeholder must be in the correct application folder and have the correct file name. The report design does not need to be complete.

Once the target report is in place, use the following steps to create the link:

1. Select the Data element in the source report and choose Hyperlink in the Properties dialog box. Select the ellipse to open the Hyperlink Options dialog. Set the Hyperlink Type to Drill-through.

2. Under Select a Target Report, enter the relative path to the linked report.

   If the report is in the same folder, enter only the report name.

   If the report is in a different folder, use the relative path. For example, to link from Job Plan List (in the JOBPLAN folder) to Job Plan Details (also in the JOBPLAN folder), enter the following text:

   ```
   ..\JOBPLAN\jobplan_print.rptdesign
   ```

3. In the Report Parameters area, select the where parameter. In the **Values** field, enter a where clause that specifies the relationship between the current row and the linked report. For example, to link from Job Plan List to Job Plan Details, enter the following text:

   ```
   "jobplan.jpnum='" + row["jpnum"] + "' and jobplan.siteid='" +
   row["siteid"] + "'"
   ```

Debugging

To log predefined information about the report, add the following lines to the report initialize method (replace the file path with a valid path):

```
mxReportScriptContext.setDefaultLogLevel("DEBUG");
mxReportScriptContext.setDefaultLogFile("c:/temp/myreport.log");
```

These lines are not used at runtime, only when you preview the report from the designer. The log levels IBM supports are DEBUG, INFO, WARN, ERROR, FATAL.

To log custom information, you can use the `mxReportScriptContext` variable throughout your report to get the script logger.

```
scriptLogger = mxReportScriptContext.getReportScriptLogger();
if (scriptLogger.isDebugEnabled())
{
scriptLogger.debug("***My Debug Message ****");
}
```

Unlike the default logging, the system writes these messages to the log files when you run the report within the system. ReportLogger has the following methods that you can use to log information:

```
boolean isDebugEnabled();
boolean isErrorEnabled();
boolean isFatalEnabled();
boolean isInfoEnabled();
boolean isWarnEnabled();
void debug(Object message);
void info(Object message);
void warn(Object message);
void error(Object message);
void fatal(Object message);
```

Registering a Report with Application Toolbar Access

Reports that do not contain user-inputted parameters can be enabled for Application Toolbar Access.

- **BV (Browser View)** – A browser view report is a report accessible from the Application Toolbar that opens immediately in the report browser. You click once on the Report icon from the Application toolbar and the BIRT report appears in a new report browser.
- **DP (Direct Print)** – A direct print report is a report that immediately opens up Adobe Acrobat and prints on the user's default printer. The end user clicks once on the Report icon from the Application toolbar and the report prints to the default printer of the user.

- **DPA (Direct Print with Attach Documents)** – A direct print with attachments is similar to a direct print report. If attached documents are associated with a record, the end user can print these documents with the report. If the user clicks **Yes**, the attached documents print to the default printer along with the report. If the user clicks **No**, only the report prints automatically to the default printer.

The table below describes the types of application toolbar access:

	Description	Database Field	Toolbar Location	Sequence
BV	Browser View	REPORT.QL	REPORT.QLLOC	REPORT.TOOLBARSEQUENCE
DP	Direct Print	REPORT.DP	REPORT.DPLOC	REPORT.TOOLBARSEQUENCE
DPA	Direct Print with Attach Docs	REPORT.PAD	REPORT.PADLOC	REPORT.TOOLBARSEQUENCE

To enable Application toolbar access, these fields must be set in the reports.xml when the report is imported. Additionally, the limit records (detail) field must be enabled and the recordlimit field set for DP and DPA functionality.

The Work Order List in the WOTRACK folder is an example of a report that IBM enables for Browser View. The following text shows the xml of the report:

```
<reports>
<report name="wotrack.rptdesign">
<attribute name="filename">../WOTRACK/wotrack.rptdesign</
attribute>
<attribute name="description">Activity List</attribute>
<attribute name="qlloc">ALL</attribute>
<attribute name="ql">1</attribute>
<attribute name="toolbarsequence">1</attribute>
<attribute name="attacheddoc">0</attribute>
<attribute name="norequestpage">0</attribute>
<attribute name="detail">0</attribute>
<attribute name="reportfolder">WOTRACK</attribute>

<resources>
<resource>

<reference>wotrack.properties</reference>

<filename>${libraryfolder}/wotrack.properties</filename>
</resource>
</resources>
</report>
```

Registering a Report for Multiple Applications

Reports can be accessed from multiple applications. Instead of copying the design files, store the report in the primary application report folder. Register the report to the other applications by including it in the reports.xml for each application.

As an example, the following steps show you how IBM registered the Work Order List Report (wotrack.rptdesign) in multiple applications.

1. Review the import entry in the reports.xml for the home application (WOTRACK).
2. View the entry to the reports.xml file for the other application folders that use the report (QUICKREP, CHANGE, RELEASE, ACTIVITY).
3. Note the change to the ***<filename>*** entry to reflect the relative path to the actual report location. For example, the following lines appear under the Activity folder:

```
<attribute name="filename">../WOTRACK/wotrack.rptdesign</
attribute>
<attribute name="reportfolder">../WOTRACK</attribute>
```

4. Other report administration values are appropriately changed.

NOTE When IBM registers a report to multiple applications, the company is cautious when entering information in the following Report Administration fields:

Report File Name – IBM entered the file name of the report exactly as it was created in BIRT Report Designer. The file name must have the .rptdesign suffix.

Application – IBM entered the application that end users will use to open this report.

Multiple application names and reports can correspond to the same report design file name (.rptdesign).

If bound parameters exist, they may need to modified since bindings that work in one application may not work in another.

Setting up Localized Reports

4

This chapter contains information for setting up localized reports. If you need to enable reports for localization, you must complete the following topics:

- Localizing Report Labels
- Localizing Data
- Formatting

Localizing Report Labels

You enable the localization of report titles and labels through property files. The system stores property files in the following location:

```
<Product_root>\reports\birt\libraries
```

A separate property file is available for each application that contains reports. For example, all Work Order reports in the Work Order application use the following properties file:

```
<Product_root>\reports\birt\libraries\wotrack.properties
```

By using a single property for each application, you minimize the number of times you need to define and localize commonly repeated label values such as description or status.

Enabling Report Labels

Complete the instructions in this section to enable labels for localization.

1. Select a report and open it in BIRT Report Designer.
2. Click the Layout tab and open the Property Editor.
3. Select the Localization property
4. If no values appear, click **Browse** to locate the Resource (properties) file from which you can access the report.

   NOTE One properties file is available for each system application.

5 Select the correct text value from the list of existing values. If the required text is not available, you need to enter a new value and save the value to the Resource file.

You save resource files in the following location:

```
<Product_root>\reports\birt\libraries
```

Importing Report Labels

After you localize a report, import the reports to load the values from the Resource file into the REPORTLABEL database table. Once the values are in the REPORTLABEL table, the system enables them for localization.

Localizing Data

You can enable Runtime Data Translation based on the user's language by calling a method for every data set column you want to translate.

The report query must include the Unique ID (UID) column(s) from the table(s) containing the translated field(s). You do not have to create output columns for these fields.

Call the following method at the end of the data set Open method. The arguments are not case-sensitive.

```
registerDataTranslation(queryColumn, queryUIDColumn, mboName,
mboAttributeName)
```

`queryColumn` – the field in the query to be translated

`queryUIDColumn` – the unique ID field in the query from the table containing the translated field

`mboName` — the maxattribute objectname for the translated attribute

`mboAttributeName`- the maxattribute attributename for the translated attribute

```
maximoDataSet =
MXReportDataSetProvider.create(this.getDataSource().getName(),
this.getName());
maximoDataSet.open();
var sqlText = new String();
sqlText = "select itemnum, description, itemid from item"
maximoDataSet.setQuery(sqlText);
maximoDataSet.registerDataTranslation("description", "itemid",
"ITEM", "DESCRIPTION");
```

Note the following items:

- Use the `getBooleanString` method to fetch and translate YORN fields.
- Do not apply currency symbols. Instead, use the system currency code field.
- Use only specified date formats.

Examples of Report Data

Since you have your own unique localization requirements, each data control IBM delivers to you in the "out-of-the-box" reports will NOT be enabled for localization.

IBM enables the following report and data fields. You can use these as examples.

- Report Location – `<Product_root>\reports\birt\reports\JOBPLAN`
- Report Name – Job Plan List
- Report File Name – jobplan.rptdesign
- Report Field – Description

Formatting

Review the following formatting issues as they relate to localizing your reports:

1. All Table elements should have widths set to 100%.
2. IBM added a new style called Titlesub. You can use this style for text that appears directly underneath the title. Many detail reports use this style for the key (such as wonum) and description.

   Work Order Details is an example of a report that uses this type of subtitle.
3. IBM made changes to the subreport template to address different issues. Evaluate your reports to see if you should apply the following changes:

   - When the subreport contained no records, the space for the row was still occupied in the .PDF version. To resolve this, all subreports now exist in single cells, stacked on top of each other.
   - There were styles applied to the rows containing the subreports. You should remove all styles on rows or cells that contain tables.
   - To resolve issues in enabling page breaks after the last subreport, IBM added a group. The group key is set to the unique key for the report. For example, in Person Details (person_details.rptdesign) the key is set to Personid. The page break after property on the group is set to "Always excluding last."

     As a result, there will be a page break after each person record (including the related subreports) but not after the last person, which would cause a blank page at the end.

     IBM deleted the report footer rows on the last records as this would have also caused a trailing blank page. In addition, IBM also removed the table footer rows in the subreport tables to tighten up the report in PDF format.
4. For Date and/or Date Time fields, always use the following formats:

   - Date - Short Date
   - Time - Medium Time

Working with Report Parameters

5

This chapter contains the following sections:

- Report Types
- Bound and Unbound Parameters in SQL Statements
- Using Boolean Values as Parameters

Report Types

The system has three types of reports, based on the existence of parameters. Parameters filter the information that appears on reports.

- Current/Selected/All
- Parameter-Based
- Current/Selected/All and Parameter-Based

The following sections describe these three types of reports.

Current/Selected/All Parameters

Current/Selected/All reports are reports without parameters that an end user runs by selecting either the current record, selected records, or all records within an application. The Job Plan Details report (jobplan_print.rptdesign) is an example of a Current/Selected/All report.

Parameter-Based Reports

Parameter-Based reports are reports that contain parameters that the end user runs against parameter values defined by the report administrator or report developer. There are two types of parameters-based reports: Bound and Unbound.

To determine if a parameter is bound, select the report in the Report Administration application and check the **Attribute Name** field. If the system populates the field, the parameter is bound. If the field does not populate the field, the parameter is unbound.

The Summary of Asset Failures by Location is an example of a Parameter-Based report.

The following sections describe both types of Parameter-Based reports in greater detail:

- Bound Parameters
- Unbound Parameters

Bound Parameters

Bound parameter exist in the main table of the application where the report is registered, or exists, via a maxrelationship that has been set up for the application. Bound parameters are appended to the Maximo where clause.

The personid parameter in the Person Details report (person_details.rptdesign) is an example of a bound parameter.

Unbound Parameters

Unbound parameters do not exist in the main table of the application, or are not available through any maxrelationship for the main table. Unbound parameters are not included in the Maximo where clause.

The dbtable parameter in the Electronic Audit Transactions reports (eAudit_trans.rptdesign) is an example of an unbound parameter.

Current/Selected/All and Parameter-Based

Current/Selected/All and Parameter-Based reports are reports with parameters that an end user runs against current reports, selected reports, or all reports within an application.

Bound and Unbound Parameters in SQL Statements

This section describes using bound and unbound parameters in SQL Statements, as well as other SQL-related information (using Boolean values as parameters).

Bound Parameters in SQL Statements

Bound parameters are included in the where parameter and do not need to be explicitly included in the report SQL.

You must include the where parameter in the SQL, as shown in the following example:

```
sqlText="select wonum, description from workorder where " +
params["where"];
```

Unbound Parameters in SQL Statements

Unbound parameters are passed to the report in a comma-delimited string and may contain operators, so the values must be parsed before being included in the

report SQL. The following method parses these values so you can include them in the report SQL:

```
MXReportSqlFormat.createParamWhereClause(String columnName, String
paramValue)
```

This method creates a SQL Where clause based on a comma separated list of values contained in paramValue. The parameter value can be specified with a prefix operator. The operator can be any one of these symbols: <=, <, >=, >, !=, = . If you do not specify an operator, the system assumes that the search is based on operator SQL LIKE as in the following example:

```
createParamWhereClause("siteid", "=BEDFORD,=MCLEAN")
```
evaluates to:
```
((siteid = 'BEDFORD') or (siteid = 'MCLEAN'))
createParamWhereClause("siteid", "!=BEDFORD,!=MCLEAN,TEXAS")
```
evaluates to:
```
((siteid != 'BEDFORD') and (siteid != 'MCLEAN')) or ((siteid like
'%TEXAS%'))
```

Adding Unbound Parameters to Report SQL

Unbound parameters must be manually included in the report SQL. The method varies depending on whether you defined the parameter as multiselect in the Report Administration application.

Multiselect parameters are passed as a comma-delimited string and must be converted to the correct syntax using the `createParamWhereClause` as shown in the following example:

```
"select wonum, description from workorder where " + params["where"]
+ " and " + createParamWhereClause("workorder.status",
params["status"]);
```

Single-select parameters, such as the where parameter, you can include directly. String parameters must be enclosed in single quotes, and date parameters must be converted to a JDBC function. Numbers do not require any special formatting, as shown in the following example:

```
sqlText = "select jobplan, description from jobplan where " +
params["where"]
+ " and jobplan.siteid = '" + params["siteid"] + "'" //Quoted
string
+ " and jobplan.wopriority = " + params["wopriority"]//Integer
+ " and jobplan.installdate >= "
+ MXReportSqlFormat.getStartDayTimestampFunction(params
["startDate"]);
```

Single-select parameters can also be used in conjunction with data set parameters. You create parameter markers by inserting a question mark in the SQL where each parameter value should go. Then use the data set `setQueryParameterValue(int index, Object value)` method to resolve them. If there are multiple parameters in one query, the method indexes them in the order they appear in the SQL, starting with 1.

The query above could be rewritten in the following format:

```
sqlText = "select jobplan, description from jobplan where " +
params["where"]
```

```
+ " and jobplan.siteid = ? and jobplan.wopriority = ? "
+ " and jobplan.duration >= "
+ MXReportSqlFormat.getStartDayTimestampFunction
(params["startDate"]);

maximoDataSet.setQuery(sqlText);
maximoDataSet.setQueryParameterValue(1, params["siteid"]);
maximoDataSet.setQueryParameterValue (2, params[prioritiy”]);
```

Notice that the where and date parameters still use the original format. The where parameter must always be directly included. Unformatted date parameters may be handled as query parameters but currently the JDBC format methods such as `getStartDayTimestampFunction` cannot be used in conjunction with query parameters.

Optional parameters are best handled by direct inclusion. In the following example, site and start date are optional. If values are specified, they are appended to the where parameter (to preserve the existing where parameter content). The work order priority is still mandatory.

```
var where = params["where"];
if (params["siteid"].value)
where = where + " and " + jobplan.siteid = '" + params["siteid"] +
"'";
if (params["startdate"].value)
where = where + " and matusetrans.actualdate >= " +
MXReportSqlFormat.getStartDayTimestampFunction(params["startdate"]
);
sqlText = "select job plan, description from job plan where " +
params["where"]
+ " and job plan.wopriority = " + params["wopriority"];
```

Using Boolean Values as parameters

Reports that take boolean values as parameters must follow these guidelines:

1. You must define the parameter in the report design as a string type.
2. If the parameter value needs to be passed to a SQL statement, then the parameter value must be converted to an integer value (1 or 0) as the database has either 1 or 0.

   IBM has added a new API call to the dataset code (**getBooleanInteger(string)**) that can be used for this purpose. Here is an example:

   ```
   var isActiveFlag = params["isactive"];
   mySQL = "select isactive from collection where isactive=?";
   myDataSet.setQuery(mySQL);
   myDataSet.setQueryParameterValue(1,
   myDataSet.getBoolenanInteger(isActiveFlag));
   ```

   or

   ```
   mySQL = "select isactive from collection where isactive=?";
   myDataSet.setQuery(mySQL);
   myDataSet.setQueryParameterValue(1,
   myDataSet.getBooleanInteger(params["isactive"]));
   ```

3. Do not use default value for YORN parameters. If default values are required, you can use “true” or “false”.

Importing and Exporting Design Files

6

This chapter contains the following sections:

- Creating the Import File
- Importing Design Files
- Exporting Design Files

Creating the Import File

There are two methods of importing BIRT reports into the system database:

- import reports through the Report Administration application
- use the import utility. If you use the import utility, you must create a corresponding xml file.

Each application folder under `<Product_root>\reports\birt\reports` has an import file named reports.xml. For example, the Database Configuration reports folder (CONFIGUR) contains a reports.xml file.

If you defined the report parameters in the xml file with appropriate attributes, then the import tool inserts or updates the report parameter table (reportlookup) with the information. The attribute names of the parameters are the columns of the reportlookup table.

If the parameter name you defined in the reports.xml for a given report does not exist in the report, the system ignores it. Some of the attribute values default to what is in the report if they are not specified in the import file.

When specifying a greater than or less than symbol for a parameter operator, you must escape the symbols as follows:

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

The Security Group report (security_group.rptdesign) in the SECURGROUP folder is an example of a report containing parameters.

The Job Plan List report (jobplan.rptdesign) in the JOBPLAN folder is an example of a report without any parameters.

Importing Design Files

The purpose of importing a report is to load report.xml and design files into the database. Both of there files types are necessary to import a report:

- reports.xml file – This file varies by application and contains all the information necessary to import a report such as file name, description, and parameter information.
- design file – This file can be either a library file (code used multiple times such as font type, size, page numbers, and time stamp) or a resource file (images or external files).

  To import one file, use the Report Administration application. See the online Help for additional information.

  To import multiple files, use the import reports command (importreports.cmd). The following section describes the types of import commands.

Importing Multiple Design Files

Complete the following section to import multiple design files:

1. Browse to the following location:

   `<Product_root>\reports\birt\tools` and open the reporttools.properties file.

2. Update or verify values for the following properties:
   - maximo.report.birt.hostname
   - maximo.report.birt.mxename
   - maximo.report.birt.registryport
   - maximo.report.birt.username
   - maximo.report.birt.password
   - maximo.report.birt.outputfolder
3. On an application server, use the command prompt to go to the following folder:

   `<Product_root>\reports\birt\tools`

4 Run the importreports.cmd command to import reports, libraries, and resource files. The following import commands are available to you:

Command	Description
importreports.cmd	imports all reports, libraries, and resource files
importreports help	provides details on how to run the command and various parameters
importreports libraries	imports all libraries
importsreports	imports all reports
importreports ***<application name>***	imports all reports. For example, if you run: importreports CONFIGUR The command imports all reports in the Database Configuration application.

Exporting Design Files

Use the export (exportreport.cmd) command to export design files. The following commands are available for exporting files:

Command	Description
exportreports	exports all libraries, reports, and various report subfolders
exportreports report	exports all reports and various report subfolders
exportreports library	exports all libraries
exportreports ***<application name>***	exports all reports for the specified application. For example, if you run: exportreports CHANGE The command exports all reports in the WOTRACK report folder.

The report exports to the folder you defined in the following locations:

- the maximo.report.birt.outputfolder property in the reporttools.properties file
- the **Report Folder** field in the Report Administration application

System Properties File Descriptions

A

The following table identifies and provides a brief description of report-specific file descriptions in the System Properties file:

Property File	Description
mxe.report.birt.maxconcurrentrun	Maximum number of reports you can run concurrently.
mxe.report.birt.queueideltimeseconds	Number of seconds the Report Queue Manager is idle after you run a report.
mxe.activex	The ActiveX Controls you enable so you can print a report with attached documents.
mxe.report.reportsInAPage	Number of reports to appear in an online reports page. The default value is 5.

Cron Task File Descriptions

B

This following table identifies and provides a brief description of report-specific file descriptions in the Cron Task file:

Cron Task	Description
REPORTLOCKRELEASE	Report Queue Lock Release Task
REPORTSCHEDULE	Report Schedule Cron Task
REPORTUSAGECLEANUP	Report Usage Log Cleanup Task

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM
AIX
DB2
developerWorks
Everyplace
ibm.com
Lotus
Maximo
Notes
QuickPlace
Tivoli
WebSphere

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

A

B

C

D

E

F

H

I

L

M

N

O

P

Q

R

S

T

U

Y