



IBM Maximo Asset Management V7.5 Report Development Guide

The collage displays several key reports from the IBM Maximo Asset Management V7.5 suite:

- SLA Details:** A table-based report showing SLA parameters like name, status, and description.
- Login History:** A line graph showing 'Maximum Concurrent Logins' over time.
- Details of Asset Failures by Location:** A pie chart showing the distribution of failures by location, categorized by severity (Error, Low, Critical).
- Asset Availability:** A bar chart showing availability percentages for different asset categories.
- Software Usage Summary:** A pie chart showing usage distribution by vendor.
- Service Level Exception:** A detailed report on SLA breaches, including application, status, and description.
- Work Order Details:** A comprehensive report on work orders, including status, description, and assigned resources.
- OEE% by Equipment:** A bar chart comparing Overall Equipment Effectiveness (OEE) across different equipment units.
- Item Availability:** A detailed report on the availability of various assets, including location, status, and last update.

V7.5 Reporting.....	5
Report Types.....	6
Installation and Configuration.....	7
1. Report Designer Download.....	8
2. Setting up the Report Designer to work within V7.5	9
Prerequisites.....	9
3. Configure Properties file.....	11
Properties File Values	12
Database username and Password.....	13
4. Accessing the Report Design Tool.....	15
5. Importing V75 Project	17
Common Install and Configuration Issues.....	22
Upgrading a 7.1 BIRT report designer instance to 7.5	24
Report Developer Database Access	25
Report Design Files.....	29
Report File Structure.....	31
Delivered Report File Structure	31
birtplatform:.....	31
eri:.....	31
Libraries:.....	31
Reports	32
Scriptlibrary	33
Report Templates.....	34
Tools	34
Your Custom Reports and the Report File Structure.....	35
Developing a report.....	39
Specifying the Query	40
Creating the Output Columns	42
V7.5-BIRT Data Mapping.....	43
Notes:.....	43
Updating the Fetch	44
Formatting the Report	45
Formatting Notes.....	46
Defining the Property File	47
Defining the Property File - Specific Steps.....	48
Report Development Considerations.....	50
Date Methods	50
Hyperlinking.....	54
Populating the Data Set.....	58
Closing the Data Set	58
Executing Additional Queries.....	59

Queries in the Fetch Method	59
Testing for Null	60
Scalar Functions	60
Enabling Rich Text Formatting	61
Bound Parameters	65
Unbound parameters	66
Specifying Bound parameters in the report design	68
Specifying Unbound parameters in the report design	68
Multi-select or single-select unbound parameters	68
Parsing Unbound Parameters	70
Creating Custom Report Parameter Lookups	71
A. Using valuelists for parameter lookups with fields that have domains	72
B. Using existing lookups	76
C. Modifying existing lookups	77
Parameter Notes	82
Number of Parameter Values	82
Utilizing Parameter Values on a Report's Request Page	82
Boolean Parameter Values	82
Optional Parameters	83
YORN Lookup	83
Viewing Parameters	84
Extending Ad Hoc Reports in BIRT Designer	85
Use Ad Hoc Reporting as a base for Custom Report Development	85
Report Designer best practices for debugging	92
Miscellaneous Features	93
Database Update Functionality	93
Registering a Report with Quick Toolbar Access	95
Importing Report Designs into the V75 Database	96
Set Up: reporttools.properties	97
Import Command Utility	99
Export Command Utility	101
Export Example	102
Additional Export Details	103
Understanding the reports.xml import file	104
Preparing the reports.xml	105
Miscellaneous Utilities	109
Update Reports Utility	109
Customizing Reports Reference Materials	110
Changing Report Logos	110
Understanding Report Paper Sizes	112
Modifying OOB Reports	113
Utilizing the V7.5 Report Booklet	114

Additional References	116
Revision History	117
Trademarks	119

V7.5 Reporting

To respond to today's dynamic Business Environment critical business information needs to be immediately available. This business information can come in a variety of formats, and is often required as a report - either a formatted business report, known as an Enterprise report, or an Ad Hoc report which is created on the fly by users.

IBM Maximo ® includes an Open Reporting Architecture, which enables you a number of different reporting options to choose from. The report options have been significantly enhanced in Maximo 7.5 and include a wide range of reporting tools.

The embedded reporting tool in the Maximo 7.5 Releases is BIRT, Business Intelligence and Reporting. As the embedded reporting tool, it enables the deepest levels of integrations throughout the various Maximo applications.

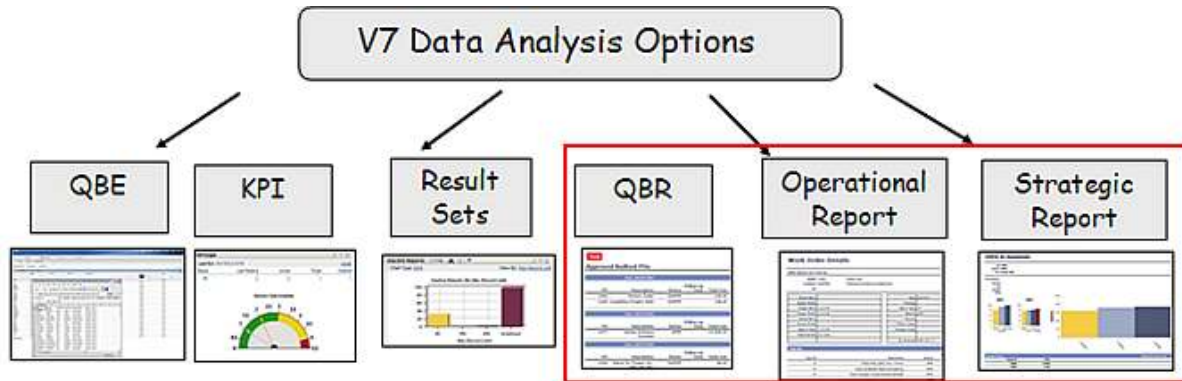
This guide details the processes in developing BIRT Reports using the BIRT Report Designer. This includes how report designs are utilized in V7.5, including their file structure of design, library and property files. Additionally, information on customizing, importing, and exporting are discussed.

Also, references to other support documentation detailing common report development customizations, including report logging, implementing barcodes and changing report logos are referenced.

*Note: This document applies only to the embedded report tool in the Maximo® Base Services 7.5 Releases.

Report Types

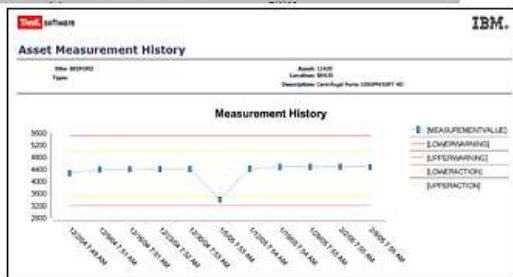
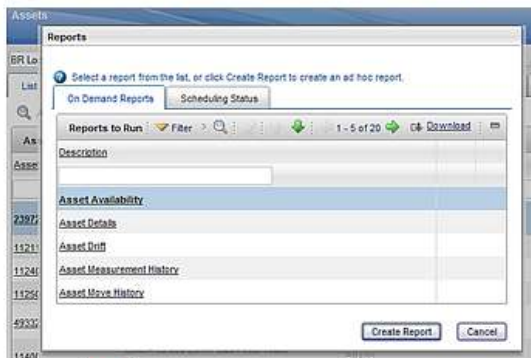
Within the Version 7.5 release, there are a number of different data analysis options. Of these options, three are specifically focused on reporting which are highlighted in red below.



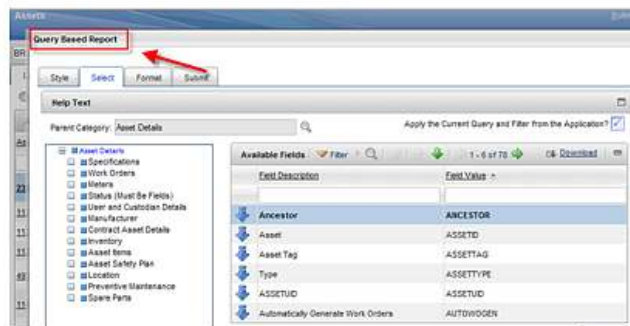
Operational reports are often referred to as transactional reporting, and are the day to day reports users require to complete their business tasks. Strategic reports enable viewing of data in varying perspectives thru the use of complex graphs, in depth calculations or scenarios. Both of these are created by a Developer in the report design tool. Ad hoc, or QBR, reports are created by users within the applications, and are not required to be created within the report design tool.

This guide focuses on Operational and Strategic reports. Information on Ad Hoc (QBR or Query Based Reports) reports is contained in a separate document titled 'V75 QBR Ad Hoc Reporting'. Links to all reference materials are noted on the last pages of this document.

Operational and Strategic Reports



QBR - Ad Hoc - Reports



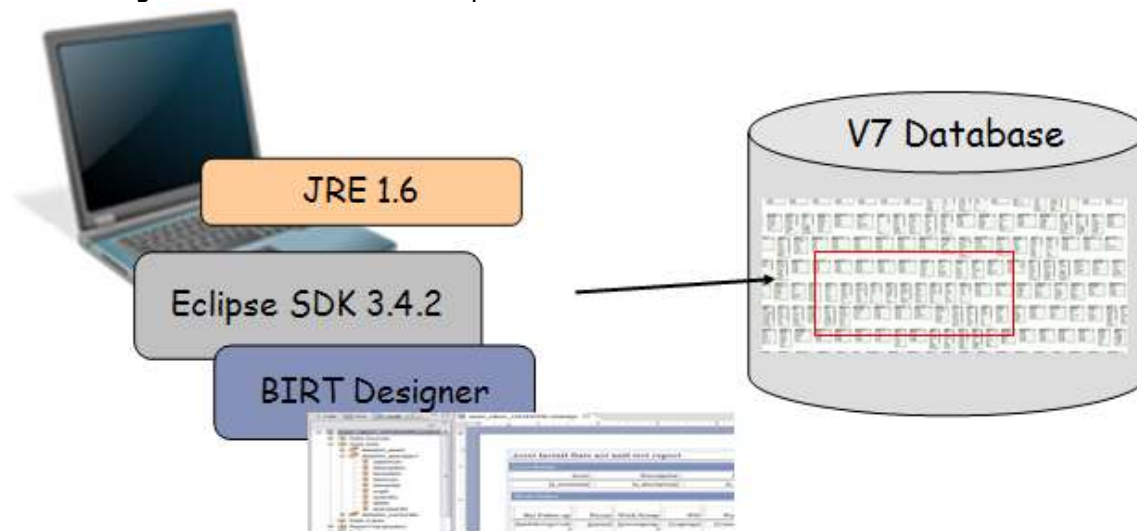
This screenshot shows a detailed data table report. The table has multiple columns including Asset, Location, and various dates. The data is organized into a grid with alternating row colors.

Installation and Configuration

The BIRT Designer is an Eclipse Based Tool that Java Developers use to create and customize V7.5 Enterprise reports. In V7.5, BIRT Designer 2.3.2 is used, which is based on Eclipse 3.4.2.

To enable the report integration, custom library, style sheet, templates and data sources have been created. These files insure a consistent, look and feel for all reports, plus most importantly, insure that reports will execute correctly from the various applications. These files must be used on all custom reports to insure the report integration executes properly.

The BIRT Designer is installed on the client machine of the Java Developer(s) who will be creating or customizing reports. It is not required to be on each user's machine - only those users who will be creating or customizing reports. Since the BIRT Designer executes off the Eclipse Framework, Eclipse must first be installed on the Java Developers Workstation, and then the BIRT Designer is installed within Eclipse. The client must also have a JRE version installed.



This first section will detail the installation and configuration of the BIRT designer for use in Version 7.5, including:

1. Report Designer Download
2. Setting up the Report Designer to work with V7.5
3. Configure Properties File
4. Accessing the BIRT Designer
5. Importing V75 Project
6. Common Configuration and Install Issues

1. Report Designer Download

Before beginning the installation, make sure a copy of IBM JDK 1.6 (6.0) has been installed locally.

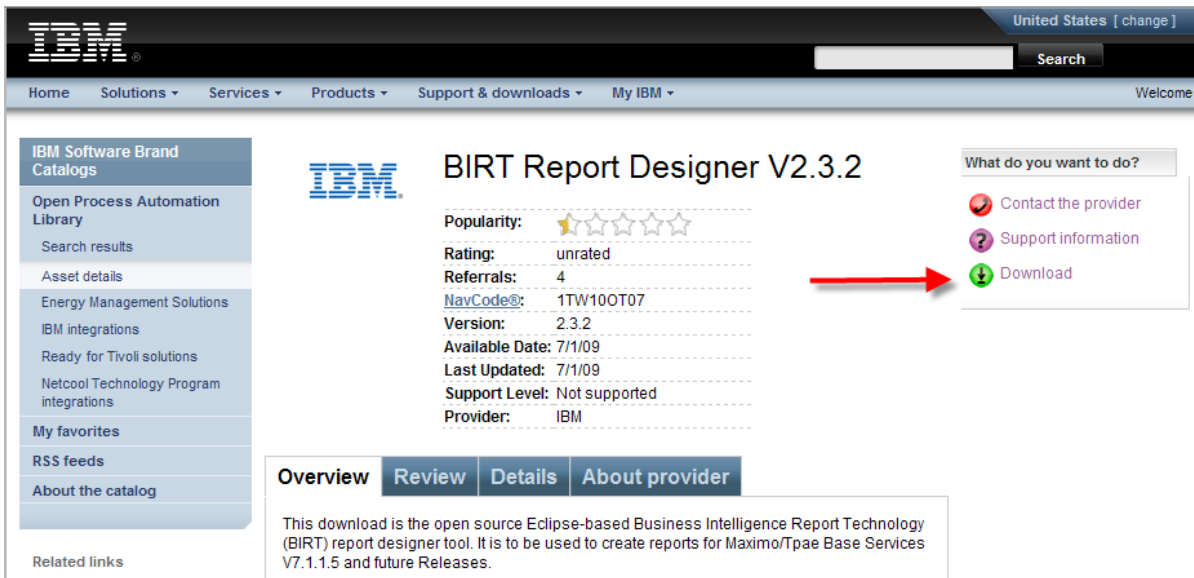
To utilize the BIRT Report Designer, it must first be downloaded to your local client machine.

The download for BIRT 2.3.2 with Eclipse 3.4.2 (V7.5 and later releases) can be found here:

<http://www-304.ibm.com/software/brandcatalog/ismlibrary/details?catalog.label=1TW100T07>

Or its shortened url of

<http://ibm.co/JRb1N1>



The screenshot shows the IBM Software Brand Catalog page for BIRT Report Designer V2.3.2. The page features a navigation menu on the left, a search bar at the top, and a main content area with product details. A red arrow points to the 'Download' button in the 'What do you want to do?' section.

Property	Value
Popularity:	★☆☆☆☆
Rating:	unrated
Referrals:	4
NavCode®:	1TW100T07
Version:	2.3.2
Available Date:	7/1/09
Last Updated:	7/1/09
Support Level:	Not supported
Provider:	IBM

What do you want to do?

- Contact the provider
- Support information
- Download**

Overview | Review | Details | About provider

This download is the open source Eclipse-based Business Intelligence Report Technology (BIRT) report designer tool. It is to be used to create reports for Maximo/Type Base Services V7.1.1.5 and future Releases.

Extract the zip file to a local directory which does not include any spaces (for example:

C:\birt_232)

2. Setting up the Report Designer to work within V7.5

After downloading and extracting the report designer, in this section, you will copy over files from Version 7.5 to the report designer.

Prerequisites

To perform the next steps, you will need a local copy of V7.5 or higher, with the report source.

2A. Locate the compiled classes used for the V75 Report Scripting from your copy of V7.5 located in:

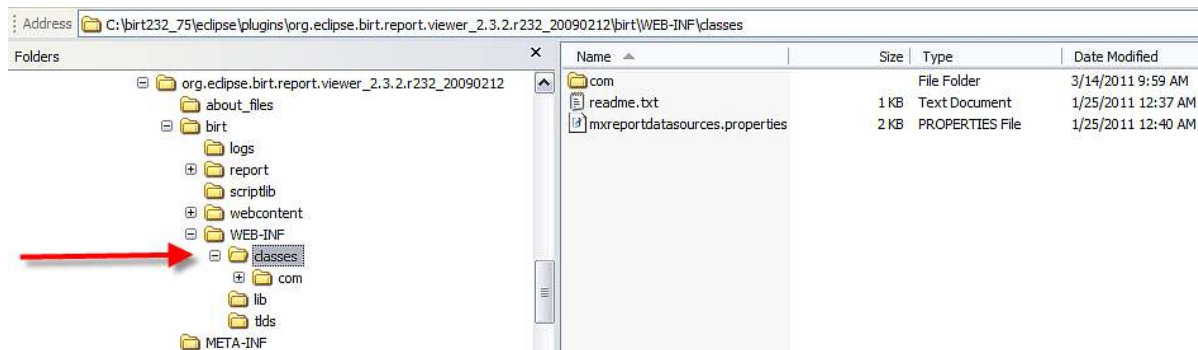
<V75>\reports\birt\scriptlibrary\classes

2B. Navigate to the Eclipse BIRT location below

<birt_232>\eclipse\plugins\org.eclipse.birt.report.viewer_2.3.2.r232_20090212\birt\WEB-INF

Copy the entire V75 \classes folder from step 2A to this Eclipse directory. This will create a new directory as shown below

<birt_232>\eclipse\plugins\org.eclipse.birt.report.viewer_2.3.2.r232_20090212 \birt\WEB-INF\classes



2C. Copy the specific JDBC driver for your database type from

<V75>\applications\maximo\lib

to:

eclipse\plugins\org.eclipse.birt.report.viewer_2.3.2.r232_20090212\birt\WEB-INF\lib

For Oracle, copy/paste oraclethin.jar

For SQL Server, copy/paste sqljdbc.jar

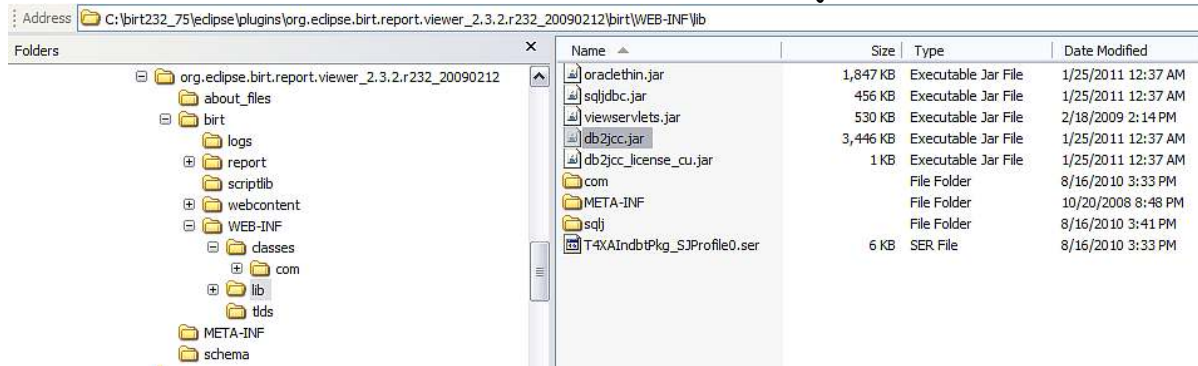
For DB2, copy/paste db2jcc.jar and db2jcc_license_cu.jar

These database drivers are only used by the report designer.

2D. Open the Jar files from Step 2C above for your specific database type and *extract the contents of the jar file to*

<eclipse>\plugins\org.eclipse.birt.report.viewer_2.3.2.r232_20090212\birt\WEB-INF\classes

The screen shot below is for a DB2 environment, where the DB2 jar files have been extracted.



Note: If you see an exception error like 'ClassNotFound' in BIRT Designer after following these steps, double check that you have extracted the jar files properly in step 2D. Not extracting the jar files will cause a 'ClassNotFound' Error.

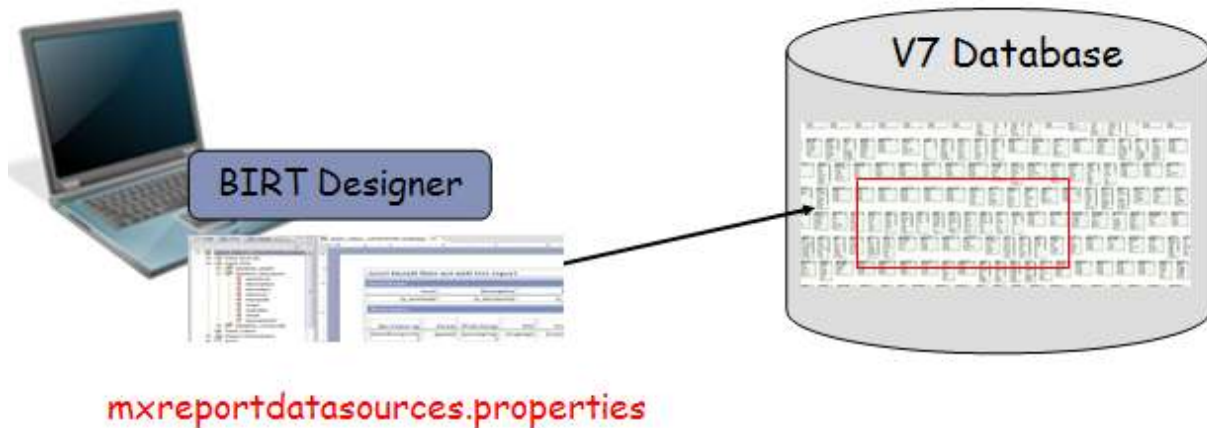
3. Configure Properties file

The `mxreportdatasources.properties` file is used to define the connection to the V75 database from the BIRT Designer client tool. It is only used for database access within the report design tool. When V75 Reports are executed from within the V75 Applications, their connection information will be passed dynamically from V75.

Typically, this file is enabled for developers in two different ways -

1. It is configured by an administrator and then distributed to each developer for his use.
2. It is configured by the developer himself.

The database that is typically used in this configuration is a test or development database. It is not the production, transactional database.



This section will review the various ways this property file can be configured.

Properties File Values

The mxreportdatasources.properties file contains information for the report designer to connect to your V75 database. You copied this file in step 2B to your local Eclipse BIRT Directory. You now need to edit this property file for your unique configuration. The values you need to define include

- Database URL
- Database Driver
- Database Username and Password
- Schema Owner

Additionally, you will need to change #<DataSourceName> to maximoDataSource.

Portions of the property file are shown below. An example of how you will configure it is shown below with sample values for a DB2 database in the outlined area.

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

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

```
# driver for SQLServer
# com.microsoft.sqlserver.jdbc.SQLServerDriver
# sample url for SQLServer
# jdbc:sqlserver://hostname:port;databaseName=dbname;integratedSecurity=false;
# 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
```

```
maximoDataSource.url= jdbc:db2://IBM-A5:50000/DB2A
maximoDataSource.driver=com.ibm.db2.jcc.DB2Driver
maximoDataSource.username=wilson
maximoDataSource.password=wilson
maximoDataSource.schemaowner=maximo
```

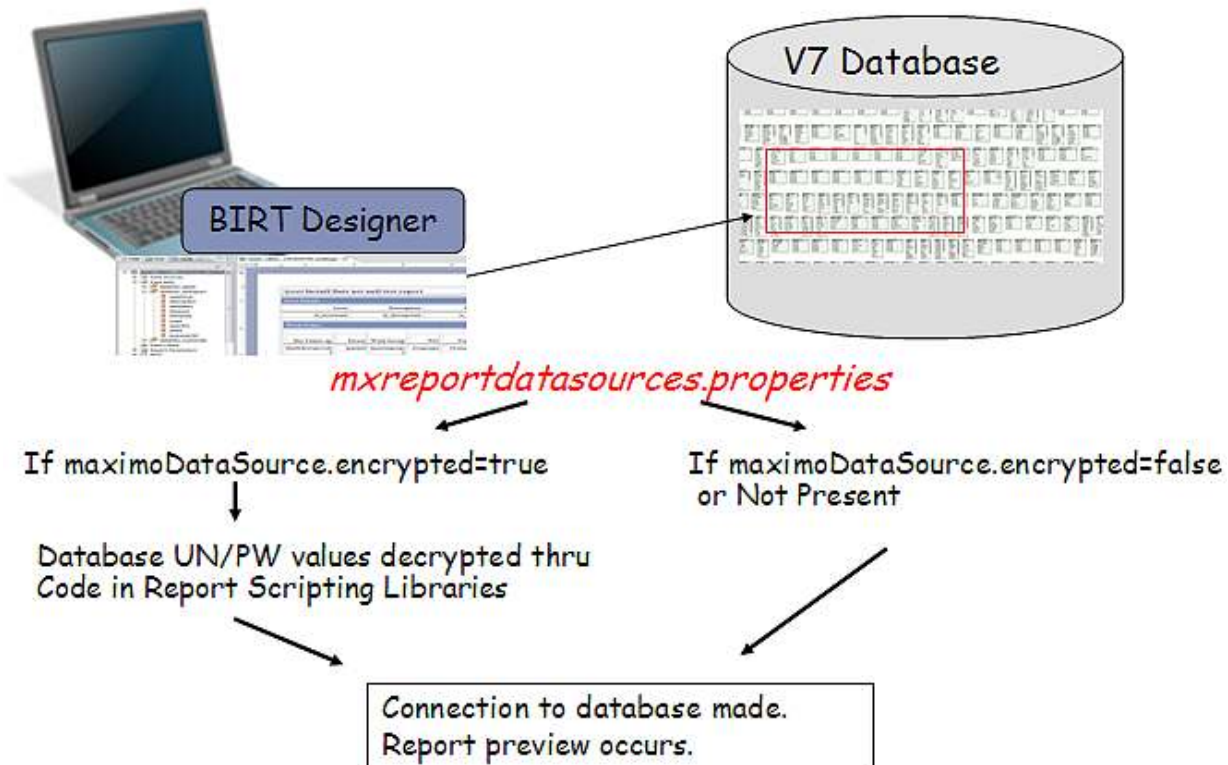
The values in green are the default for all reports, and you must use this text exactly.

The values in red represent a sample DB2 database.

Database username and Password

Within the `mxreportdatasources.properties` file, you define the database username and password for the report developer. This enables the developer to preview the report results within the report design tool to confirm the report is functioning correctly.

You can choose to either input these values directly into the properties file, or you can choose to encrypt the values.



Property File Encryption

To encrypt the username and password values within the `mxreportdatasources.properties` file, follow the steps below:

1. Define the username and password values in `mxreportdatasource.properties`.
2. Open up a command prompt. Navigate to the location `<V75>\reports\birt\tools` and execute `encryptproperties.bat`. This utility will
 - A. Encrypt the username and password values
 - B. Add a value to the property file called: `maximoDataSource.encrypted=true`
3. Once the username and password is encrypted, you can distribute it to your developer(s) for use in their local environment.

After the encryption process, the property file will be updated to values similar to what is shown below.

```
maximoDataSource.encrypted=true
maximoDataSource.schemaowner=maximo
maximoDataSource.username=YaNJYGUPFrc\=
maximoDataSource.url=jdbc\:db2\://IBM-A5\:50000/DB2A
maximoDataSource.password=mEWNcVcBRfuBL54acL+JSg\=\=
maximoDataSource.driver=com.ibm.db2.jcc.DB2Driver
```

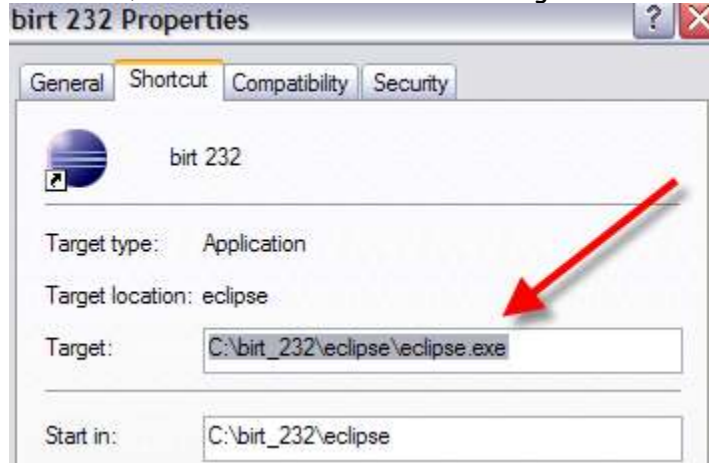
Notes on properties file

1. The encryption process adds escape characters to the URL, which do not affect its value
2. The encryption process only encrypts values identified by maximoDataSource values
3. You may not want to grant each report developer full database access by using the system maximo database user privileges as the developer creates and test report designs. Instead, you may want the developer to have restricted database access. This restriction usually requires that the report developer be granted 'read only' access to a limited number of database objects. To do this, a unique database user is required. Details in how to do this can be found in later in this guide in the section titled 'Report Developer Database Access'.

4. Accessing the Report Design Tool

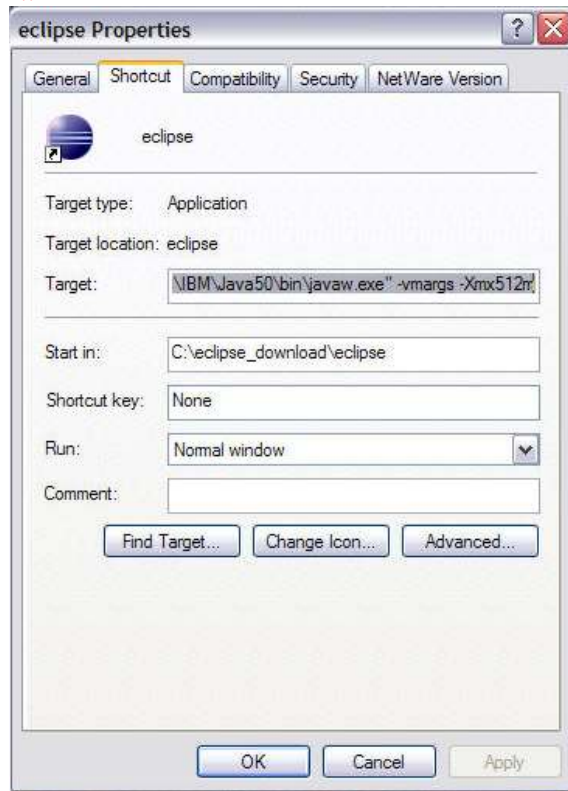
These next steps detail how you will access the Report Designer.

4A. First, create a shortcut to BIRT Designer 232 from eclipse.exe.

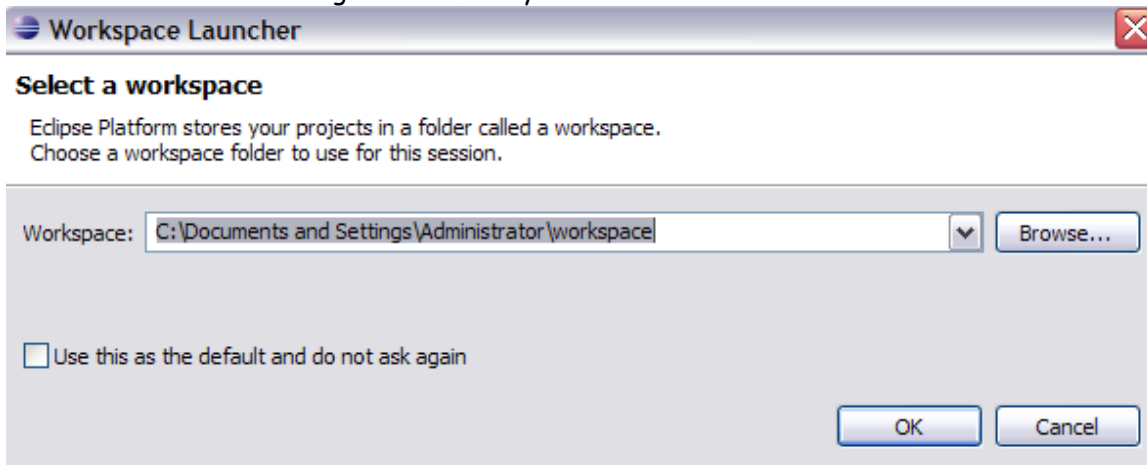


4B. Update the shortcut's target to include your JDK 1.6 install by modifying the path highlighted in red below.

`C:\birt_232\eclipse\eclipse.exe -vm "C:\Program Files\IBM\Java60\jre\bin\java.exe" -vmargs -Xmx512m`

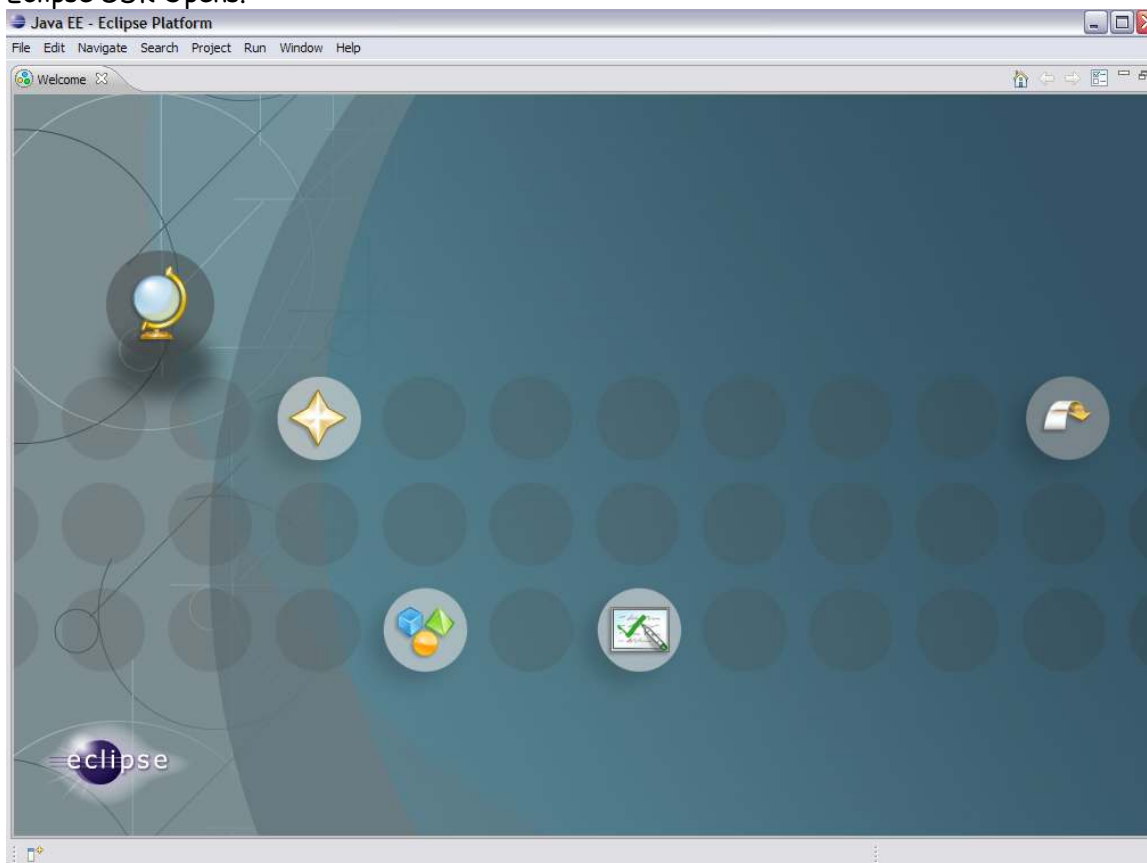


4C. Launch the BIRT Designer 232 from your new shortcut.



4D. Select an applicable workspace location for your environment. Check 'Use this as default' field and OK.

Eclipse SDK Opens.



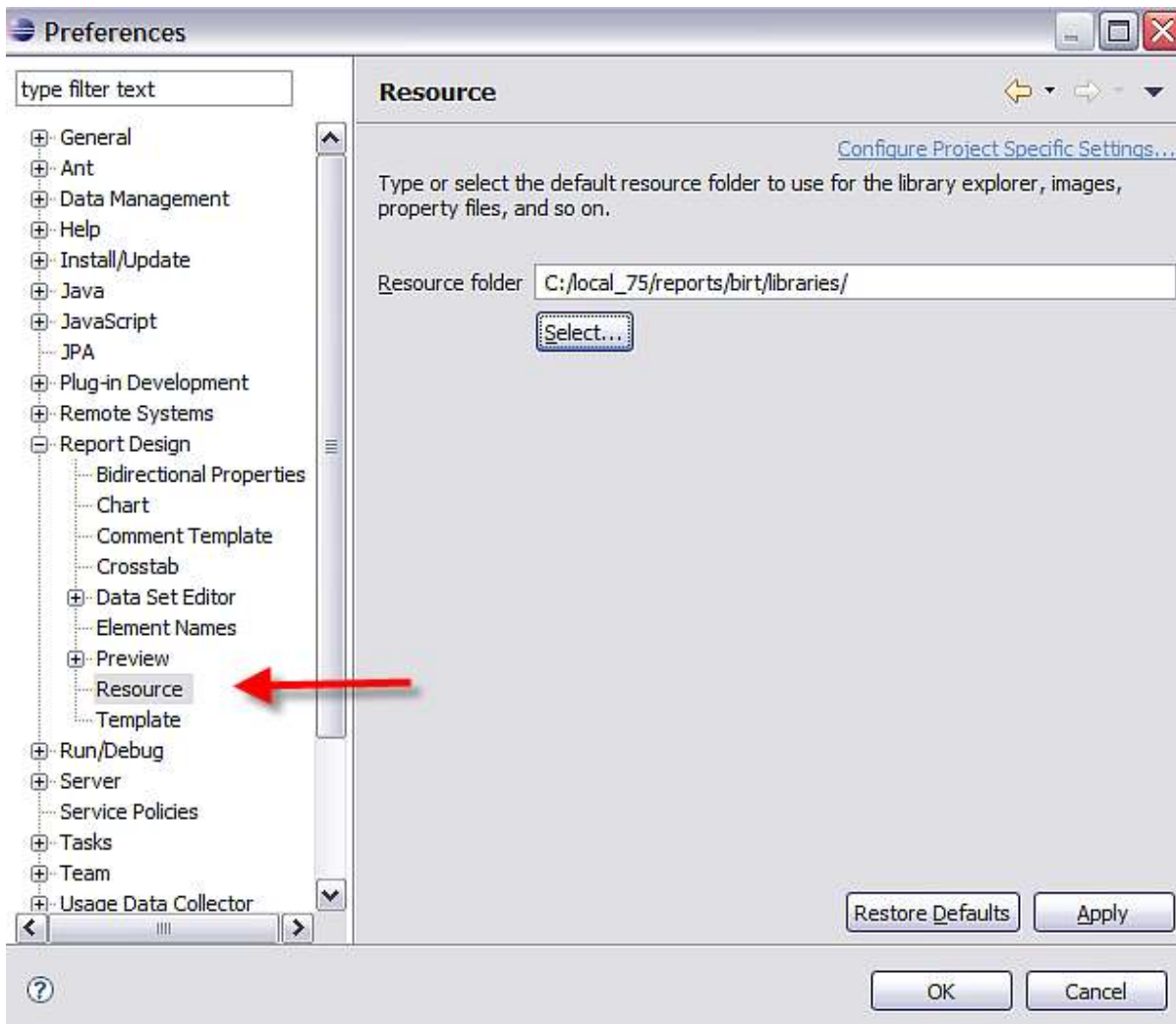
5. Importing V75 Project

In this last section, you will import the V7.5 Project into your report development environment.

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

5A. First, specify the resource folder location to import the V75 Libraries.

- A. From the Menu, select Window - Preferences.
- B. Expand Report Design and select Resource.
- C. Browse to your local report library location and select Apply.
 <V75>/reports/birt/libraries



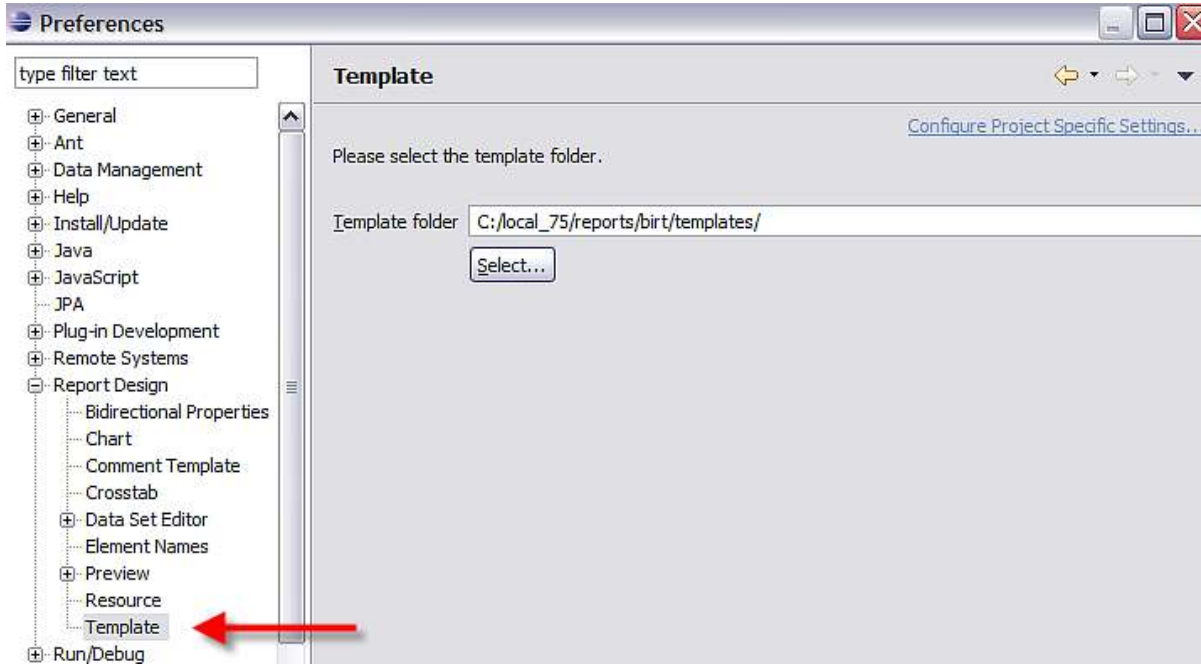
5B. Specify the templates folder location to import the V75 Templates.

A. If not already open, from the menu, Select Window - Preferences.

B. Expand Report Design and select Templates.

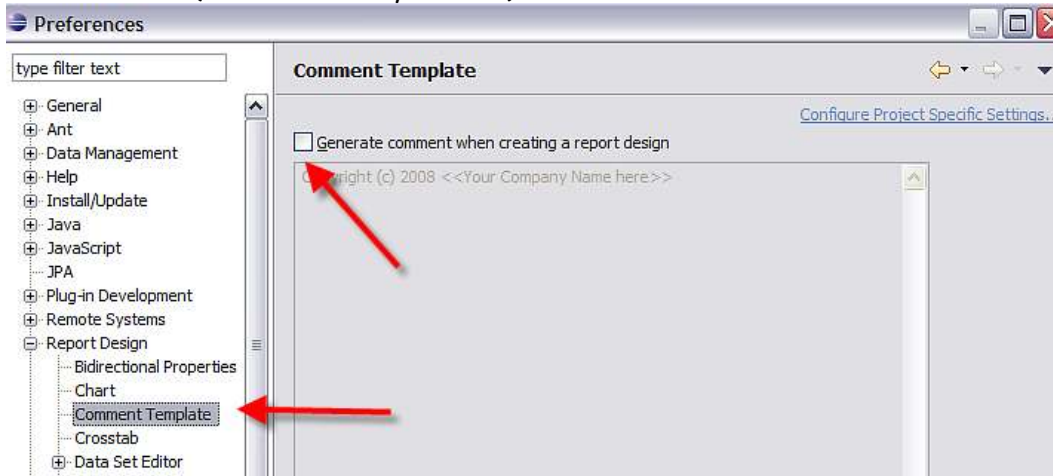
C. Browse to your local report template location and select Apply.

<V75>/reports/birt/templates



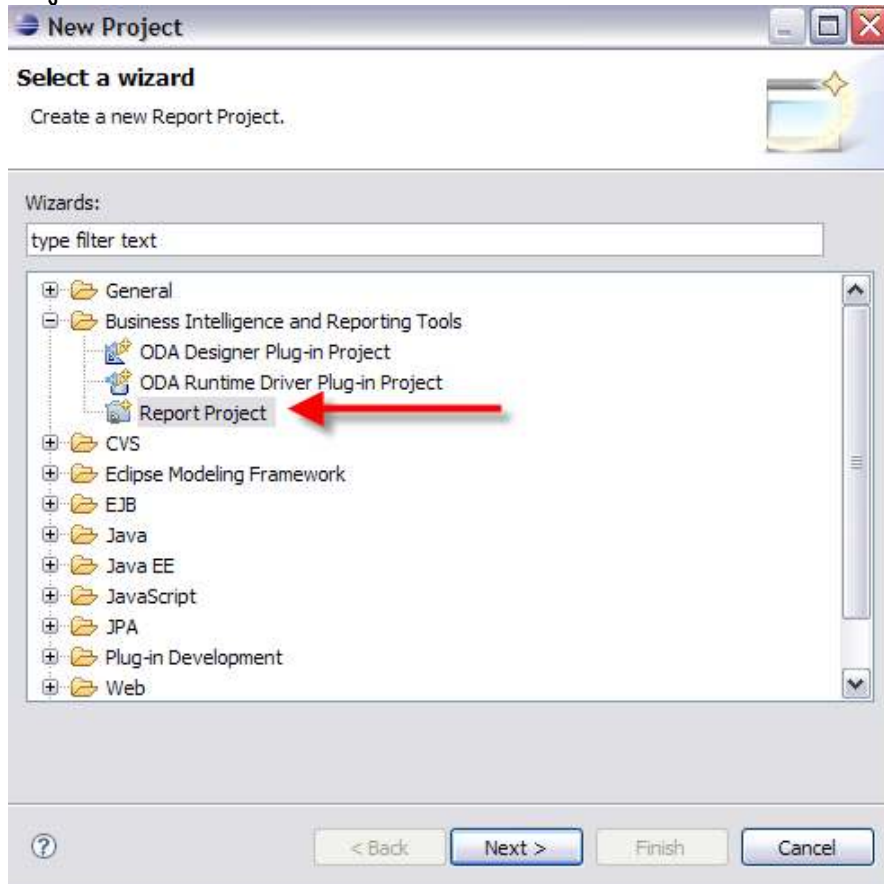
5C. Disable the Comment Template.

- A. If not already open, from the menu, Select Window - Preferences.
- B. Remove the flag from the 'Generate comment when creating a report design' field if it is set. (It is disabled by default)



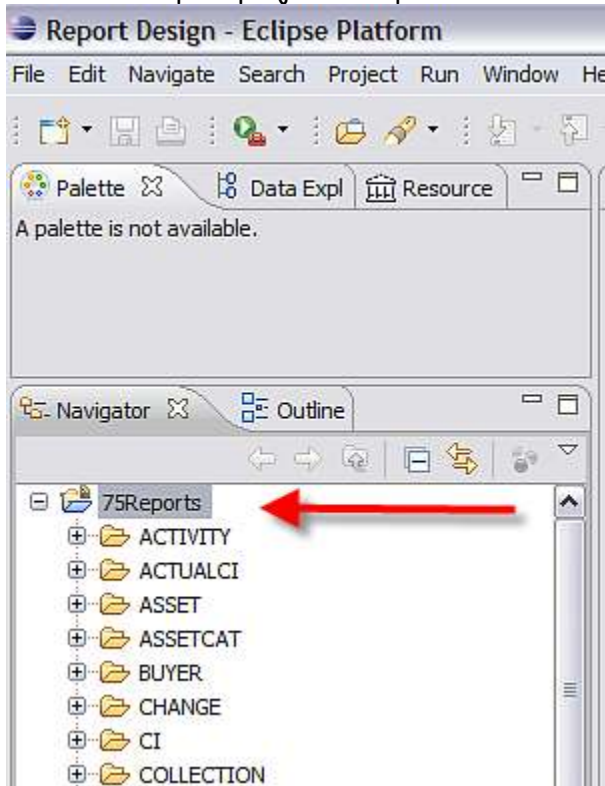
5D. Next, import the report project. This will bring the V75 Reports into your project workspace.

- A. Click File, New Project. Under Business Intelligence and Reporting Tools, select Report Project. Click Next.

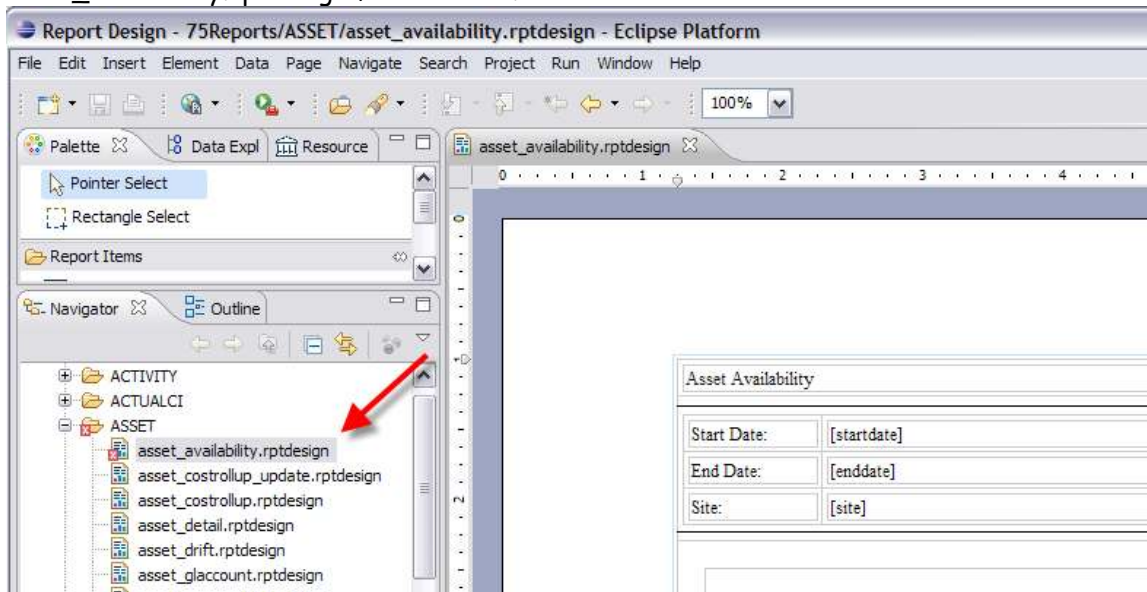


- B. Enter a Project Name.
 - a. Remove the flag in the 'Use Default Location' field.
 - b. For the Location Field Value, browse to the location of your V7 Report Source. <V75>\reports\birt\reports. Click Finish.

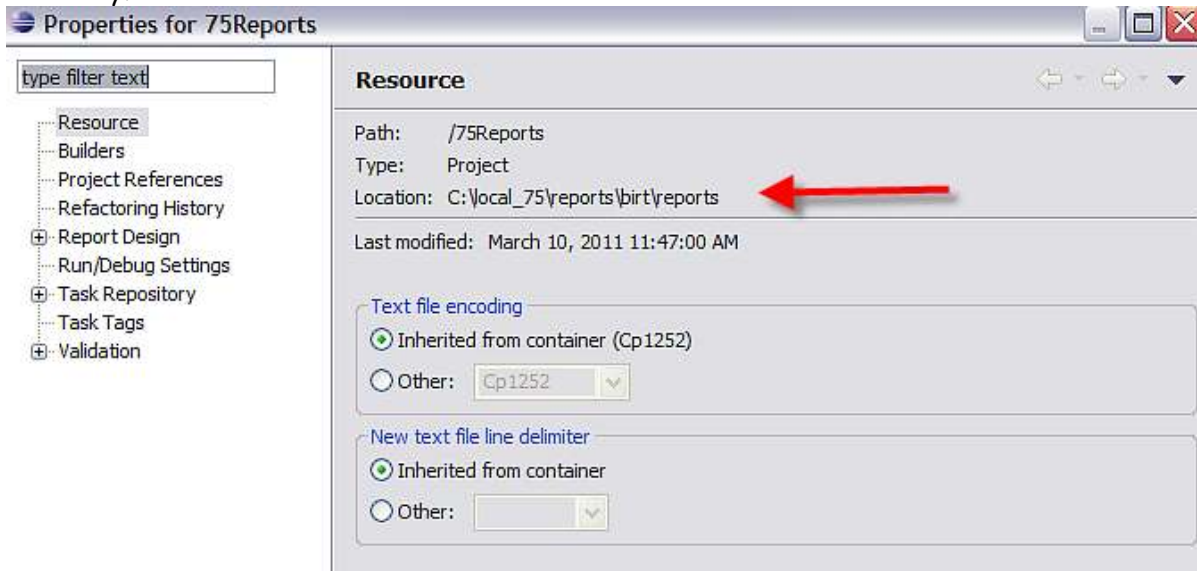
C. The report project is imported.



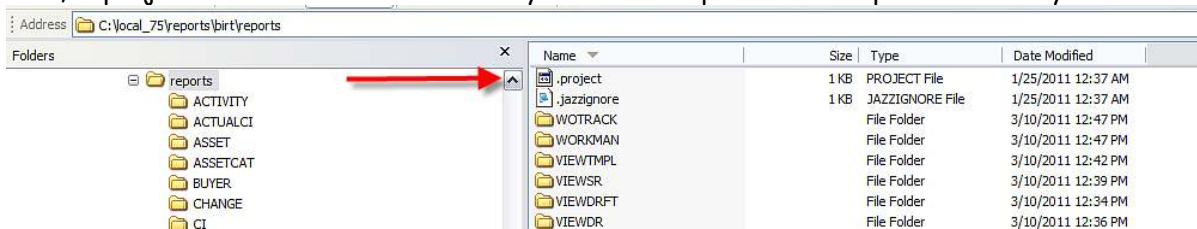
D. To confirm the reports imported properly, expand a folder and then double click on an rptdesign file to open it. In the screenshot below, the ASSET folder is expanded, and the asset_availability.rptdesign file selected.



E. You can confirm that the correct project is imported. To do this, right click on the Project.
*NOTE: The location should be your V75 Report Source. It should not be under an Eclipse directory.



Also, a .project file will be created under your <V75>\reports\birt\reports directory.



Common Install and Configuration Issues

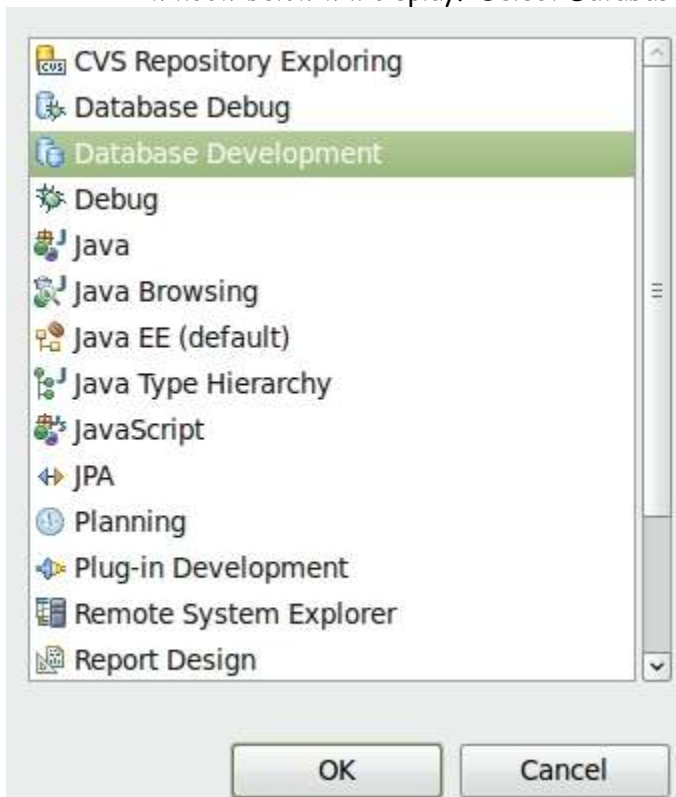
If you have issues configuring the report designer, verify the items below have been properly configured or installed.

1. JDK 1.6 is installed.
2. The BIRT Designer 'All In One' Package was used
3. The supported versions of Eclipse 3.4.2 and BIRT 2.3.2 are being used.



Provider	Feature Name	Version	Feature Id
Eclipse.org	Apache Commons Codec Plug-i	1.3.0.v20070920-	org.apache.commons.codec
Eclipse.org	Apache Derby Core Plug-in for I	10.3.1.4	org.apache.derby.core
Eclipse.org	BIRT Advanced XML Editor Plug	2.3.2.r232_v20090	org.eclipse.birt.report.designer.edi
Eclipse.org	BIRT Chart Context-sensitive H	2.3.2.r232_v2008:	org.eclipse.birt.chart.cshelp
Eclipse.org	BIRT Chart Framework	2.3.2.r232_v2008:	org.eclipse.birt.chart
Eclipse.org	BIRT Chart Runtime	2.3.2.r232_v2008:	org.eclipse.birt.chart.runtime
Eclipse.org	BIRT Context-sensitive Help	2.3.2.r232_v2008:	org.eclipse.birt.cshelp
Eclipse.org	BIRT Documentation	2.3.2.r232_v2008:	org.eclipse.birt.doc
Eclipse.org	BIRT Example	2.3.2.r232_v2008:	org.eclipse.birt.example
Eclipse.org	BIRT Report Designer	2.3.2.r232_v2008:	org.eclipse.birt.report.designer
Eclipse.org	BIRT Report Runtime	2.3.2.r232_v2008:	org.eclipse.birt.report.runtime
Eclipse.org	Business Intelligence and Repo	2.3.2.r232_v2008:	org.eclipse.birt
Eclipse.org	Data Tools Platform Connectivi	1.6.2.v200810071	org.eclipse.datatools.connectivity.
Eclipse.org	Data Tools Platform Home Descri	1.6.2.v200810071	org.eclipse.datatools.descrip

4. Both steps 2C and 2D were performed.
 - a. For 2D, make sure that you extracted the Database Jar Files to the directory. If the jar files are copied only, errors will result.
5. Confirm that the mxreportdatasources.properties file was configured for your unique environment.
 - a. To help confirm this, check if you can connect to the configured database using Eclipse's Data Tools Connections.
 - b. To do this, from the Menu, select Window - Open Perspective - Other and the window below will display. Select Database Development.



- c. Then go thru the steps of adding a new connection profile and testing Database Connectivity from the BIRT Designer.

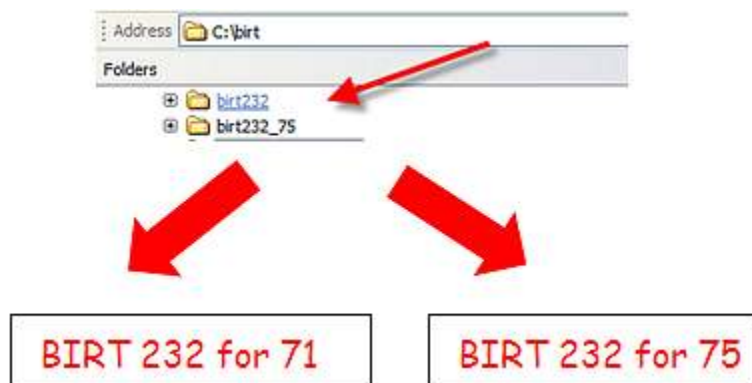
Upgrading a 7.1 BIRT report designer instance to 7.5

If you are upgrading for a 7.1x environment to 7.5, there have been a number of platform and report updates which can impact your report development environment. These updates include platform support of Sql Server 2005, JRE 1.6, along with updated report scripting libraries and mxreportdatasource.properties file.

Because of these changes, it is highly recommended that you utilize two BIRT instances during the upgrade process. This means that you would utilize two versions of BIRT Report Designer 2.3.2 - one for your 7.1x instance, and one for your 7.5 instance. If you utilize one BIRT Report Design instance to point to the very different scripting classes in 7.1x and 7.5, you will experience problems.

Therefore, in the 7.1x instance, you would continue to use the existing platforms, property file and scripting classes.

In the 7.5 instance, you would use the updated platform - including JRE 1.6, along with the 7.5 scripting classes and property file.



Report Developer Database Access

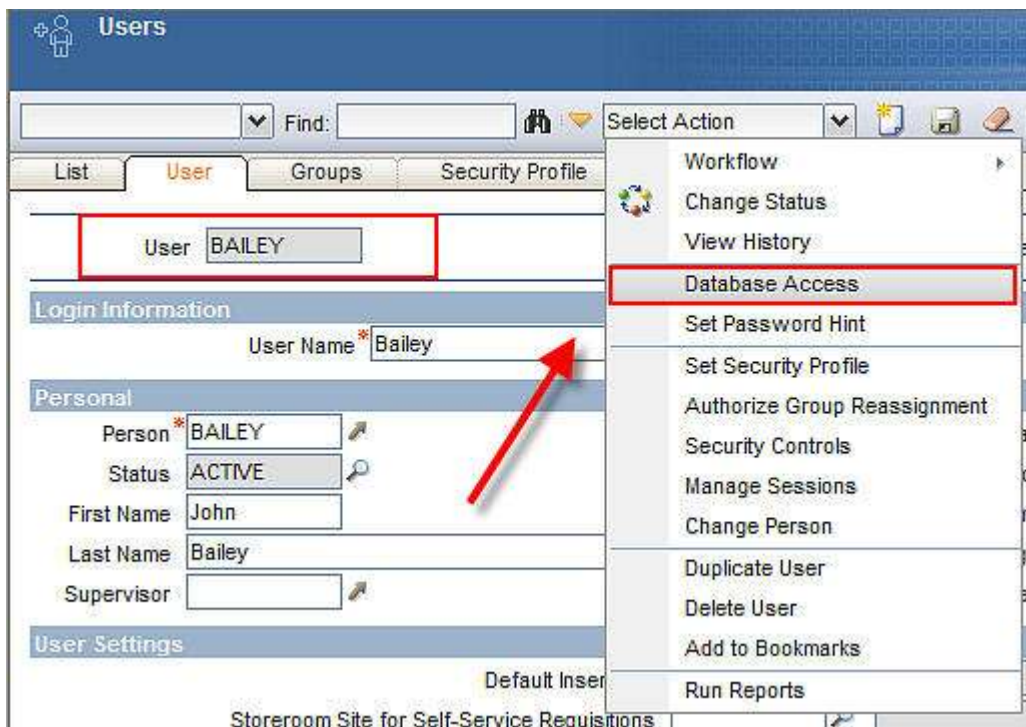
Report developers require database access to create and customize reports. This enables them to test their report design to insure the applicable content is being retrieved.

Clients may not want to grant each report developer full database access by using the system maximo database user privileges ad the developer creates and test report designs. Instead, they want the developer to have restricted database access. This restriction usually requires that the report developer be granted 'read only' access to a limited number of database objects. To do this, a unique database user is required.

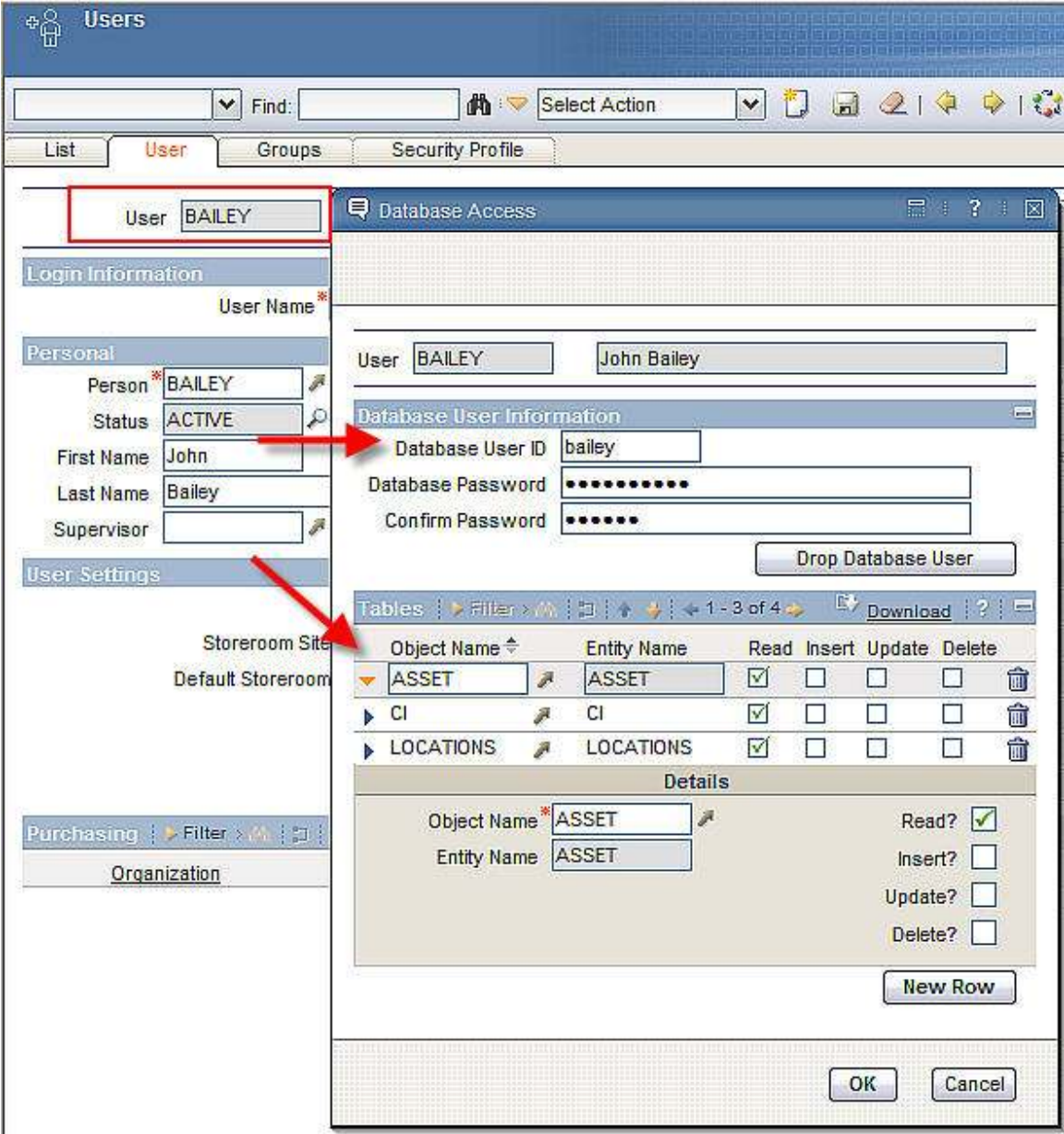
The steps below detail a few different ways on how this restricted access can be granted thru the use of a unique database user. After this unique database user is created, the steps for incorporating the new database user for the report developer are described.

Method 1 - Creating Database User and Access within User Application

If you are using Oracle or SQL Server, you can directly create a new database user through the User Application in V7. To do this, first create a new user for your report developer. Then, from the Action Menu select 'Database Access'.



Enter a unique Database User ID, along with the database password. Then, using the table section in the bottom portion of the dialog, specify the database objects that the report developer should have access to. Grant database 'read only' access to these specific database objects the report developer will be creating reports against. In this example, the developer, Bailey, is given read only database access to the Asset, CI and Locations objects.

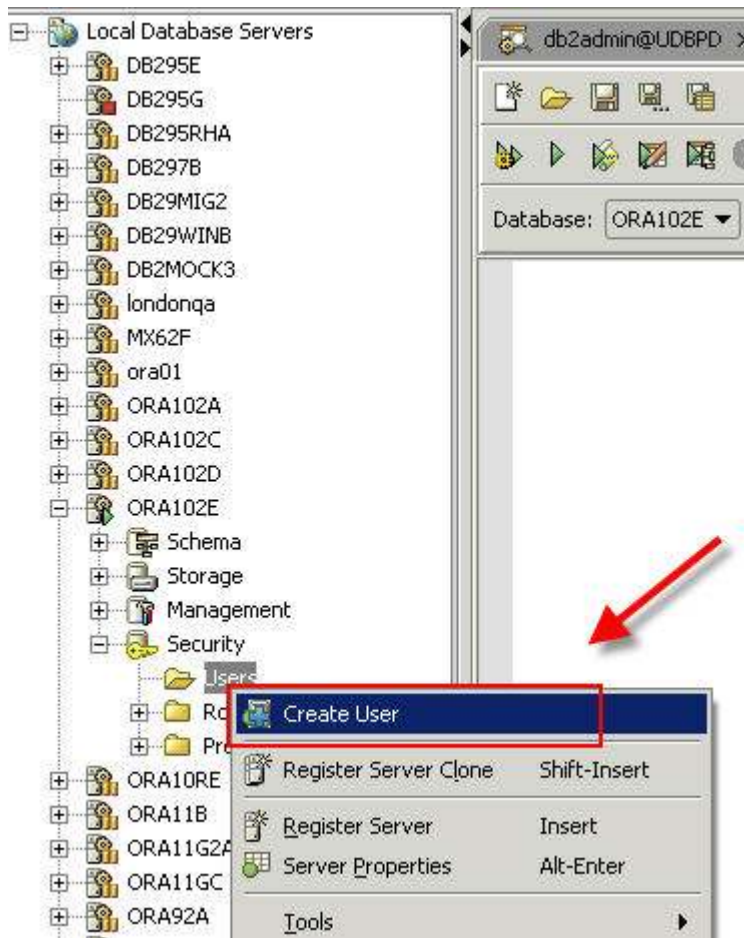


Note: If you are using DB2, the new database user must also be an Operating System (OS) User. Therefore, the DB2 user must first be added as an OS user before performing the action above.

Method 2 - Creating Database User and Access within Database Configuration Tools

You can also create a new database user and specify access through a Database Configuration Tool.

To do this, access the database querying tool, and locate the database instance you are working with. From the tool, add a new database user. (*Note: The method in which you access this functionality will vary by database tool and type.)



Once the database user is created, then grant 'Read only' database privileges via scripts to the specific database tables he will have access to. Example scripts are shown below, where the report developer, Bailey, is granted 'read only' access to the ASSET, CI and LOCATIONS objects.

```
grant select on MAXIMO.ASSET to bailey
```

```
grant select on MAXIMO.CI to bailey
```

```
grant select on MAXIMO.LOCATIONS to bailey
```

Configuring the BIRT Report Designer to use the new report developer database user

After the new database user has been created, this database user will then be used by the report developer in his unique instance. To enable this, the mxreportdatasources.properties file will be updated.

This mxreportdatasources.properties file specifies the database url and driver, along with the database user and password that should be used. Using our example above, values for this property file could be:

```
maximoDataSource.url=jdbc:db2://V7116:50000/HARRIER  
maximoDataSource.driver=com.ibm.db2.jcc.DB2Driver  
maximoDataSource.username=bailey  
maximoDataSource.password=bailey1abc
```

Report Design Files

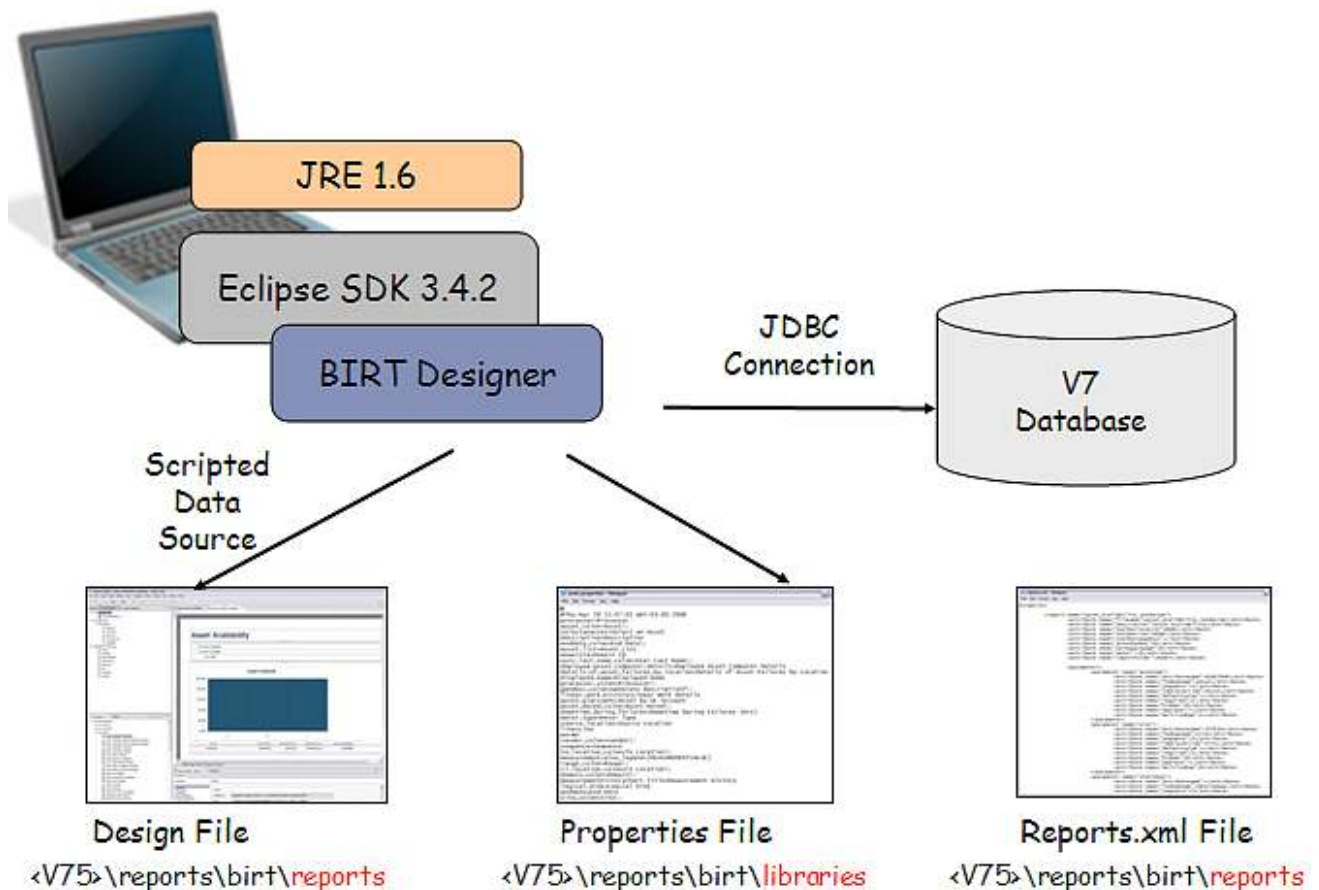
Before creating reports, a review of the report design process will be detailed. There are three files created for report designs.

1. Design File. This contains the details on the report - its sql, grouping, sorting, hyperlinking, etc. An example of this is the asset_availability.rptdesign file.

The Design File uses a custom scripted data source. This is done to fully utilize the specific functionality for Runtime Data Translation and Time Zone Conversions.

The scripted data source calls the JDBC Connection to execute the report against the V7.5 Database.

2. Properties File. This contains the text values and keys of each column label and report title. There is one properties file for each application that has reports. This enables the same label values (ex. Description) to be used only once. This property file is one of the major components used in localization. An example of this is the asset.properties file.
3. Reports.xml File. This file defines the report information (its design file name, its parameters, its application etc.) and is used to import the report files into the database. There is one reports.xml file for each application.



The chart below shows how the report files interact with each other. At the top level is the design file, which always has the file extension of .rptdesign.

Each of the reports has a dependency on the Maximo® System Library File. A BIRT Design file can only have a dependency on either another design file (.rptdesign) or another library file (.rptlibrary)

The Maximo System Library file has its own import file, called libraries.xml. If a change is made to the Maximo System Library, the libraries.xml file is used to import that library change into the database.

The Maximo System Library file contains references to the resources, or image files. These typically have a .gif or .jpg extension. When a resource file is imported into the database, the files are converted to .zip format. (These files are stored as BLOB data types in the database.)

The properties files are also resource files. Properties files are referenced in the reports.xml which is used to import the reports into the database.

File Name	Dependency	Resource	Description	Location**
Asset.rptdesign			Asset List Design File	reports
	maximoSystemLibrary.rptlibrary		Maximo System Library	libraries
		.gif/.jpg files	Resources or Image Files	libraries
		asset.properties	Asset Property file	libraries
Reports.xml			Information on report and its parameters. Used for importing	reports

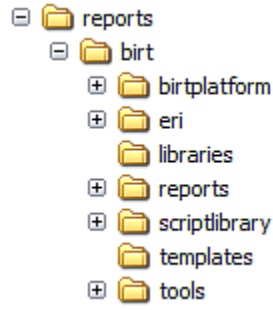
**Location in the chart has been condensed. Its full path is <v75>\reports\birt\....

More details on these files are contained in the report file structure below.

Report File Structure

Delivered Report File Structure

The reporting infrastructure contains not only the files required for the report engine, but also the design files, libraries, templates and various tools used during the reporting processes. The 75 file structure is shown and detailed below.



birtplatform:

Contain files required for the BIRT engine. These files should not be modified.

eri:

Files for configuring V7RI (Integration used when a Version 6 environment uses V7 Reporting)

Libraries:

Library, Resource and Property files required to support the report design files.

A. Library. Libraries store re-usable components, functionality and images. Reports that use libraries are automatically updated with the latest library information when they are executed.

One system library, called `MaximoSystemLibrary.rptlibrary`, is used. It contains two core items:

1. Master Pages. This defines items like the margins for printing, and the controls used for page formatting (ex page n of m). This is contained in the library because it is used on all reports, and rarely changes.
2. Themes. This contains the style sheet, which defines the font type, font size and other text characteristics to be used in the reports. The theme in the library is referred to as the style in the report design. The `maximoTheme` contains the specific colors and formatting for the reports.

The `libraries.xml` file is used for importing the `MaximoSystemLibrary` file.

B. Resource. Resource files are `.gif` or `.jpg` images used in report designs. Two resource files are used. These are `IBM_logo_black.gif` and `tivoli.gif`, which are the two logos displayed at the top right and left hand corner of each V7 Out of the Box report. Resource files are imported into the database as zipped files.

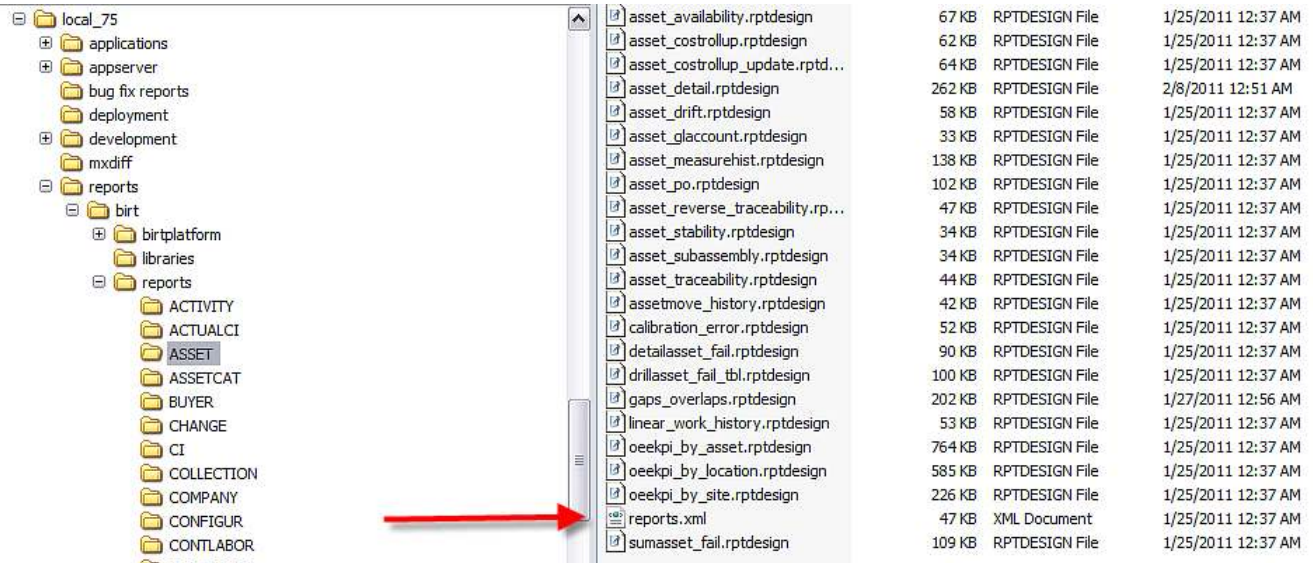
Clients may want to customize the reports to use their own corporate logos. Information on how to do this is in 'Changing Logos in BIRT Reports' referenced on the last page of this document.

C. Properties File. Each of the application's properties file is contained within this subdirectory. Property files contain the text values of the report titles, and column/Subheader labels.

Property files are created at the application level, and not at the report level, because reports within an application frequently share the same text label values. (Example: Asset Reports often use the same labels of Asset, Location, Site, multiple times.)

Reports

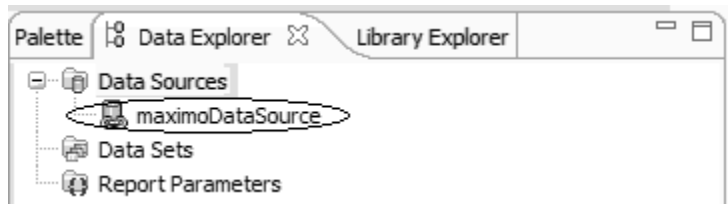
Contains Report Design Files stored within their corresponding application subfolder. Also contains the reports.xml file with information on each report used for importing.



Scriptlibrary

The Script library contains script library classes and the mxreportdatasources.properties file used by BIRT Designer tool to connect to databases.

For the integration of BIRT, when a report developer creates a report, a Custom Scripted Data Source is used. This Scripted Data Source is called 'maximoDataSource'.



A scripted data source is used to fully utilize the specific functionality for Runtime Data Translation and Time Zone Conversions. An example of this functionality is the localized values of Description. If a client is running both English and Spanish environments, and the English values of descriptions been localized into Spanish, the scripted data source is required to insure the localized Spanish descriptions display in reports. The classes for the scripting are contained within this subfolder.

Notes on Script Library:

1. Whenever you update your system to a new patch release or version of Maximo Base Services, the script library may have been updated in the new release. To insure that you use the most recent script libraries in your environment, copy the latest script library from

<V75.xDirectory>\reports\birt\scriptlibrary\classes

To

<birt>\eclipse\plugins\<birt report viewer directory>\birt\WEB-INF\classes

For example, when you upgrade from Maximo Base Services 7.5 to a future fixpack of Maximo Base Services 7.5.x, copy the 7.5.x classes directory to your existing BIRT instance.

For information on upgrading from Maximo Base Services 7.1 to 7.5, reference the section in this guide titled 'Upgrading a 7.1 BIRT report designer instance to 7.5'.

2. For details on the script library, including the methods available, reference the V7 Java Docs available on IBM's Integrated Service Management Library website.

<http://bit.ly/pPtBKn>

For version 7.5, you can find the report scripting methods in
<Javadocs7500>\com\ibm\tivoli\maximo\report\script

Report Templates

Six Template files are used as starting point in creating report design files.

File Name	Template Name	Description
maximoListReport	Tivoli Maximo List Report Template	For simple listing report - traditional row, column format
maximoGroupReport	Tivoli Maximo Grouped Report Template	Same as listing report - but contains sections for grouping results - ex. group by site or status
maximoSubreport	Tivoli Maximo Subreport Template	Used for complex reports, including detail reports
maximoChartListReport.	Tivoli Maximo Chart List Report Template	Simple listing report, which includes a graphic for either bar, line or pie chart before the report's results.
maximoChartGroupReport	Tivoli Maximo Chart Grouped Report Template	Grouped report with graphic for either bar, line or pie chart before the report's results.
maximoChartSubreport	Tivoli Maximo Chart Subreport Template	Complex report with graphic for either bar, line or pie chart before the report's results.

***NOTES:**

1. When creating any report to be used within Maximo, you must start with one of the Tivoli Maximo templates as they contain the required scripted data source and library files needed for the integration.

If you do not use a template or an out of the box report as your starting point, your reports will eventually fail when executed from the V7.5 environment. The templates contain critical scripting classes which are used by the report engine to determine when a report has started and finished. Without these scripting classes, the report queue will build up and you will soon receive out of memory errors.

2. You can download additional portrait report templates from IBM's ISM Library at <http://bit.ly/iwBvoc>

Tools

Files used to importing and exporting report design files from database. More information on these tools is contained in the Import and Exporting sections.

Additional Notes on Report Source:

1. There are no separate library or design files for the three database types that are supported. Within the report source, the sql is being written in ANSI Standards, so it will be applicable to any of the 3 database types.

- There may be a few out of the box reports where the database specific sql is required. In these cases, the sql will be written with conditional statements (ex. If database type = IBM DB2®, do this. If not, do this + that...etc)

Your Custom Reports and the Report File Structure

The section above reviewed the delivered report source and file structure. However, you may need to create or modify reports to meet your individual business needs. In this case, you will have new or modified report design files, reports.xml and properties file.

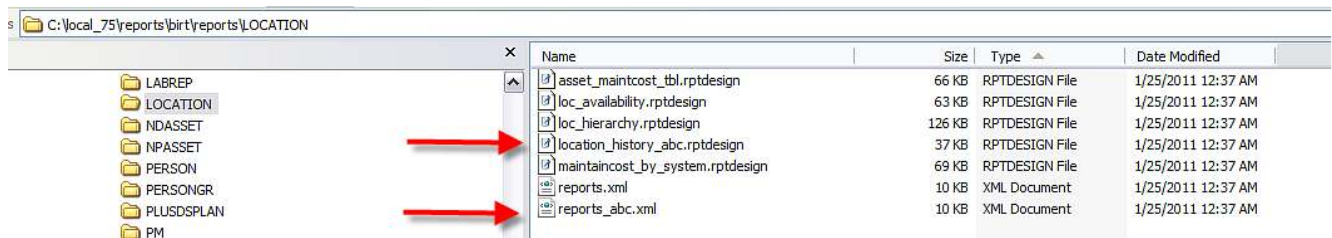
To streamline the administration and maintenance of your custom report design files, and also to insure that they are properly updated in future hot fix and fix pack releases, it is highly recommended that you implement a file structure similar to what is shown below.

For New Custom Reports - Report Design and XML file

For any new custom reports you create, it is highly recommended that you assign them unique report file names, and also create new reports.xml files for these. You may want to make them unique by utilizing your company name, or another identifier in their file name and reports.xml.

To illustrate this, let's imagine you created a new report for the Location application, which is titled Location History Report. To highlight this as your report, you may want to call it location_history_abc.rptdesign, where abc is the name of your company.

Additionally, when you create its reports.xml, instead of modifying the existing location's reports.xml for this new report, create your own unique reports.xml titled reports_abc.xml, which is located under the directory: <V75>\reports\birt\reports\LOCATION



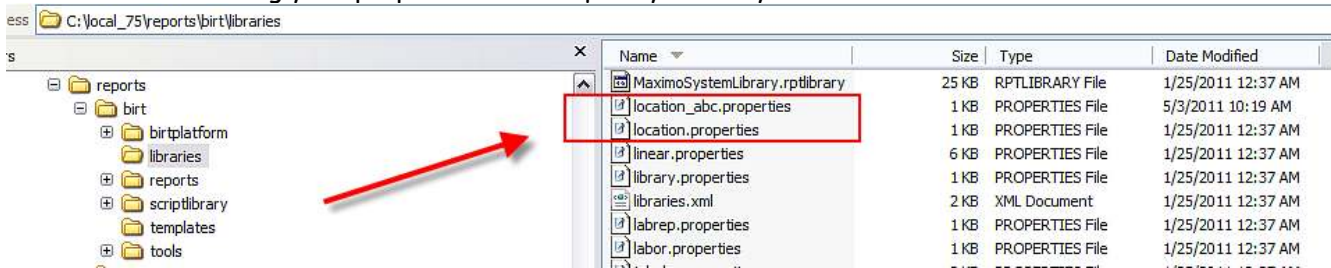
For New Custom or Modified Reports - Properties file

As stated above, the properties file contains the text for the title and labels within your report. This is used to insure the values can be properly localized.

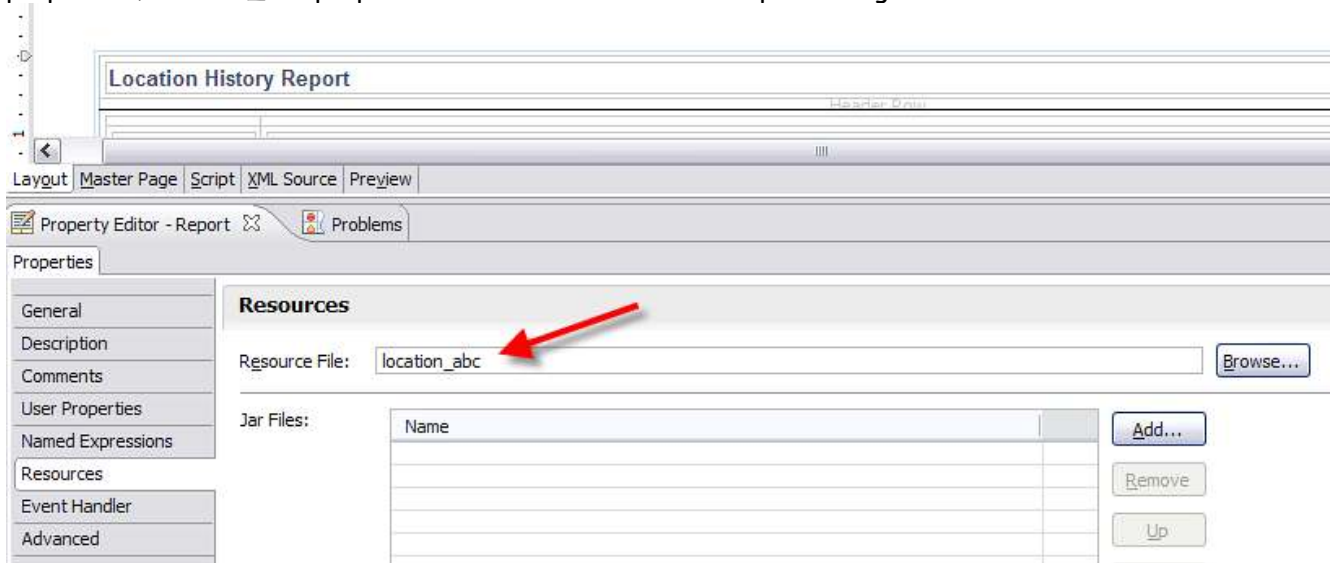
When you create custom reports, you can choose to either modify the existing properties file for the application, or create your own new properties file. To determine the solution that is best for your environment, you may want to take the following into consideration

1. A single report design can only reference a single properties file.
2. Applications can utilize multiple properties file. During the command import process, all properties file for the application will be imported.
3. Report titles, labels may be modified during release, fix pack or hot fix updates. Therefore, if you modify the delivered properties file with your customizations, your updates may be overridden during an update.

Based on this, you may want to create your own custom properties file, by copying the delivered file and then renaming your properties file to quickly identify it.



Then, when your developer adds new labels for your new reports, he will add them to the custom properties, `location_abc.properties` as shown below in the Report Designer.



By creating unique file names and unique reports.xml files for your custom reports, they will always be imported during the import process. The import process imports any xml file it sees - not just the delivered reports.xml file. Additionally, when modifications are made to the out of the box reports, you will not have to merge your changes - they will be kept separate.

For Modifications to Delivered Reports - Report Design and XML file

You may decide that you simply need to add or remove fields to a delivered report to meet your data analysis requirements.

In this case, it is recommended that you follow the same process as above, in making a copy of the original report design file, renaming it to a unique file name, and then making the customizations to the new report design file. This same process would apply to the reports.xml file. Make a copy of it, rename it, and make the change to the new reports.xml file.

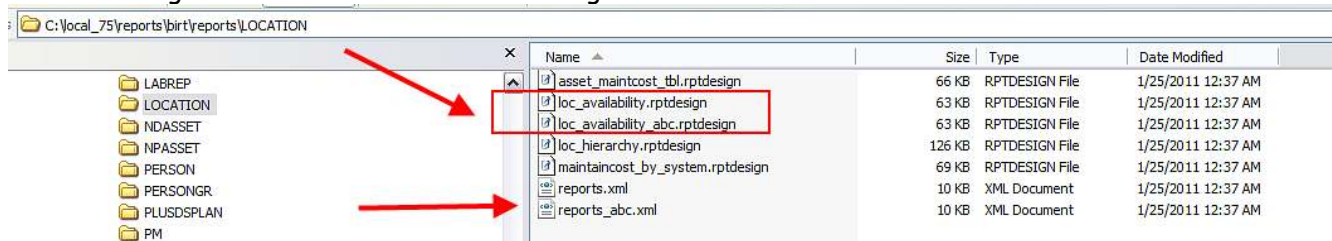
Following a similar example as above, you need to modify the Location Availability Report. To do this, you

1. Copy the loc_availability.rptdesign file
2. Rename the copied version to loc_availability_abc.rptdesign
3. Make the changes to the report in the Report Design tool and save.

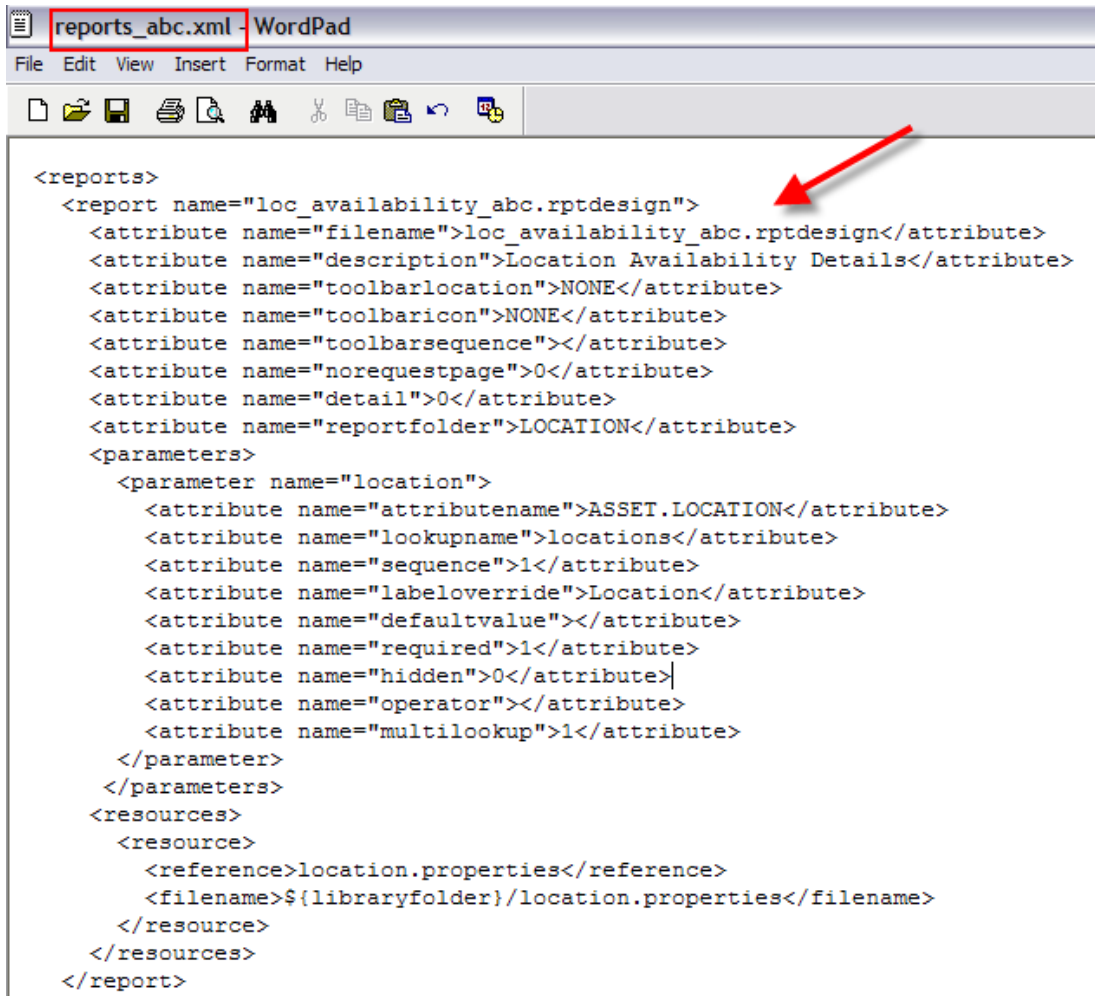
For the reports.xml file, you

1. Copy the reports.xml file
2. Rename the copied version to reports_abc.xml.
3. Find the location availability entry and modify it to use the new file name, loc_availability_abc.rptdesign and any other changes to the report and parameters. Delete all other references to design files that you have not modified.

Your resulting file structure will look something like this



An example of the copied and modified reports_abc.xml is below. Note again, that this version of the reports.xml should only include the entries for the files you have updated. Do not leave in entries of the report design files you have not modified.



```
<reports>
  <report name="loc_availability_abc.rptdesign">
    <attribute name="filename">loc_availability_abc.rptdesign</attribute>
    <attribute name="description">Location Availability Details</attribute>
    <attribute name="toolbarlocation">NONE</attribute>
    <attribute name="toolbaricon">NONE</attribute>
    <attribute name="toolbarsequence"></attribute>
    <attribute name="norequestpage">0</attribute>
    <attribute name="detail">0</attribute>
    <attribute name="reportfolder">LOCATION</attribute>
    <parameters>
      <parameter name="location">
        <attribute name="attributename">ASSET.LOCATION</attribute>
        <attribute name="lookupname">locations</attribute>
        <attribute name="sequence">1</attribute>
        <attribute name="labeloverride">Location</attribute>
        <attribute name="defaultvalue"></attribute>
        <attribute name="required">1</attribute>
        <attribute name="hidden">0</attribute>
        <attribute name="operator"></attribute>
        <attribute name="multilookup">1</attribute>
      </parameter>
    </parameters>
    <resources>
      <resource>
        <reference>location.properties</reference>
        <filename>${libraryfolder}/location.properties</filename>
      </resource>
    </resources>
  </report>
</reports>
```

Notes:

1. With this approach on duplicating and modifying the report source and files, you will end up with two entries of the location availability report in your database and also in the Report Administration application. These are the original report, and the report you have customized.
 - A. To only make the customized version (loc_availability_abc.rptdesign) available to your users, only enable report file security to this file in the report administration application.
 - B. Or, you could remove the original file (loc_availability.rptdesign) from the original reports.xml file. However, you would need to repeat this process for each future fix pack or release upgrade you receive.
2. For recommendations on the properties file for modifications to delivered reports, please see the section above titled 'For New Custom or Modified Reports - Properties file'

Developing a report

This section details how to create a report design within the Report Designer for V7.5.

Note: Before beginning this process, review the section titled 'Extending Ad Hoc Reports in the BIRT Designer'. This section details how you can streamline the report development process by minimizing the steps below thru the use of exported ad hoc reports. It is highly recommended that you utilize this process to save both development time and resources.

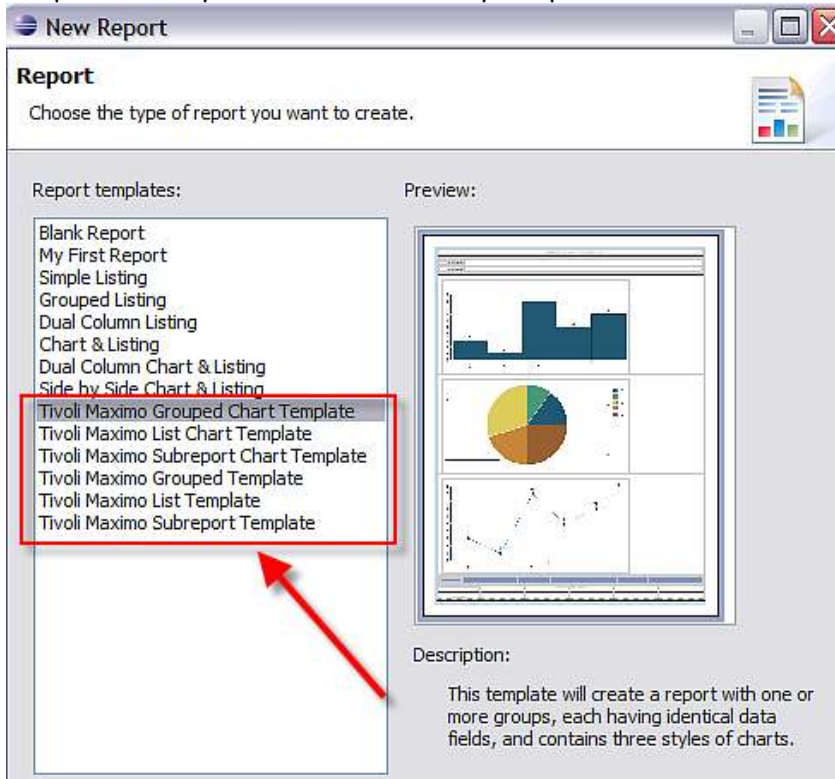
If you want to create a report design file without using the ad hoc exporting process, follow the process below.

The steps required for report development are:

1. Specifying the query
2. Creating the output columns
3. Updating the Fetch to map the query columns to the output columns
4. Formatting the report
5. Defining the property file

To begin, access the Report Designer. Then, select File - New - Report or choose New Report from the dropdown list. A number of sample reports and templates are displayed.

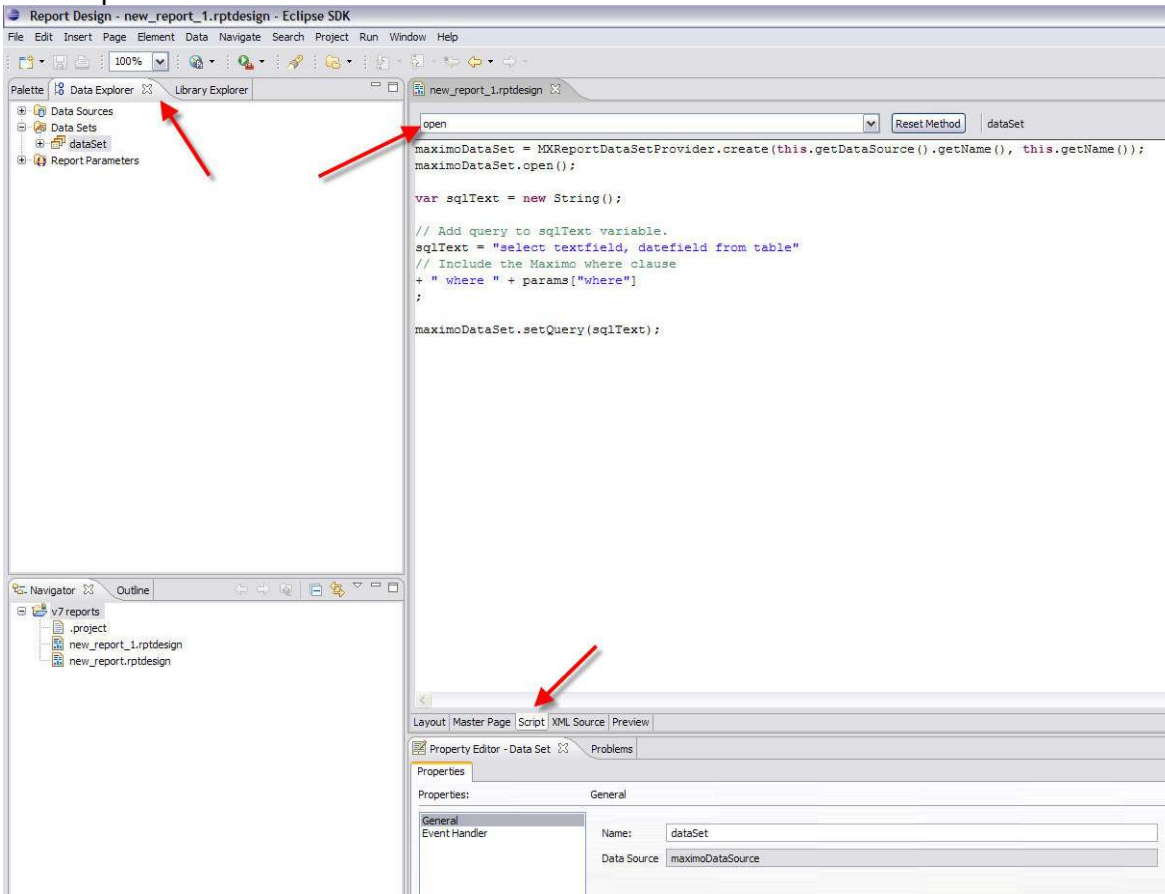
Select the desired Tivoli Maximo template from the list. You must select a Tivoli Maximo template as they contain the necessary scripted data source and library for the integration.



Specifying the Query

The first step in creating a BIRT report is to input the sql statement. When doing this, it is highly recommended that you first develop and test all required queries in your separate database query tool. BIRT does not validate SQL and a query tool will provide clearer error messages.

To input the sql, select the data set in the Data Explorer and choose the Script tab. Select the Open method from the dropdown list. 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 provided in the template.



Notes on the sql:

1. It is recommended that ANSI SQL join syntax (left outer, right outer) should be used. ANSI functions such as *CASE* and *COALESCE* should be used instead of proprietary functions such as *DECODE* and *ISNULL*.
2. Owner qualification (*MAXIMO.workorder*) should NOT be used
3. Reference all database objects in lower-case.
4. Each report must contain the base table name of the application it will be accessed from in its sql statement.

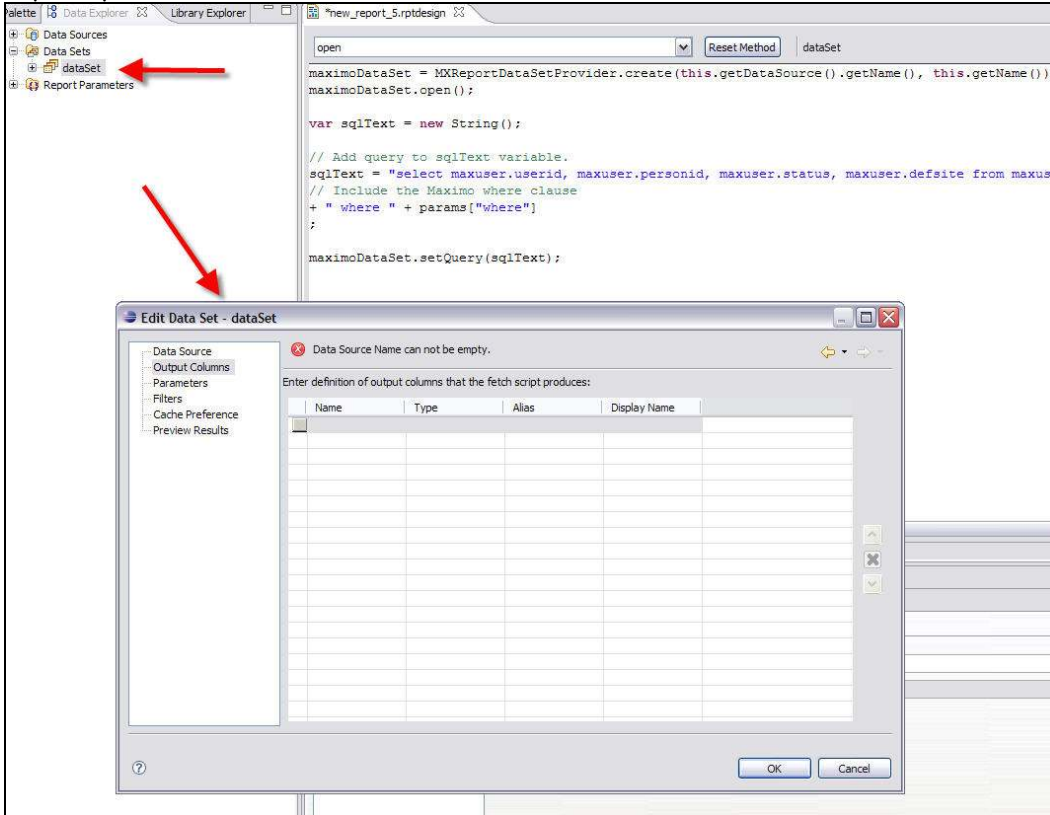
This can be explained using the example of a report that is being created for the Location Application. When the sql for the report is being prepared, the base table name of the application must be included in the sql. You can find the base table name for an application by executing a query similar to what is shown below:

```
select maintbname from maxapps where app = 'LOCATION'
```

Then, once you obtain the base table name, confirm that it is included in your sql. Even if you do not include any fields from the base table, it still must be included in the report's sql.

Creating the Output Columns

Next, the report output columns will be defined. Double-click the data set to open the properties dialog. In the Output Columns editor, enter a column for each field in your query, as well as for any computed columns.



Set the data type for each output column based on the maxtype of the field. The chart below shows the V7.5 Database Type, the corresponding BIRT Data Type, and the method used within the report designer to retrieve its value.

V7.5-BIRT Data Mapping

V7.5 Database Type	BIRT Data Type	Data Set Method used to Retrieve
ALN, CLOB, GL, LONGALN, LOWER, UPPER	String	getString(String attributeName)
YORN*	String	getBooleanString(String attributeName)
DATETIME, TIME	DateTime	getTimestamp(String attributeName)
DATE	Date	getDate (String attributeName)
AMOUNT, DECIMAL, DURATION**	Decimal	getDouble(String attributeName)
FLOAT	Float	getFloat(String attributeName)
DURATION**	String	getDuration(String attributeName)
INTEGER, SMALLINT	Integer	getInteger(String attributeName)

To determine the V7.5 data types (maxtypes) of the fields used in your queries, you can query the maxattribute object directly in the database. An example of this is:

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

Or, you can use Database Configuration application in V7.5 to look up the maxtypes, using the Type field on the Attributes tab.

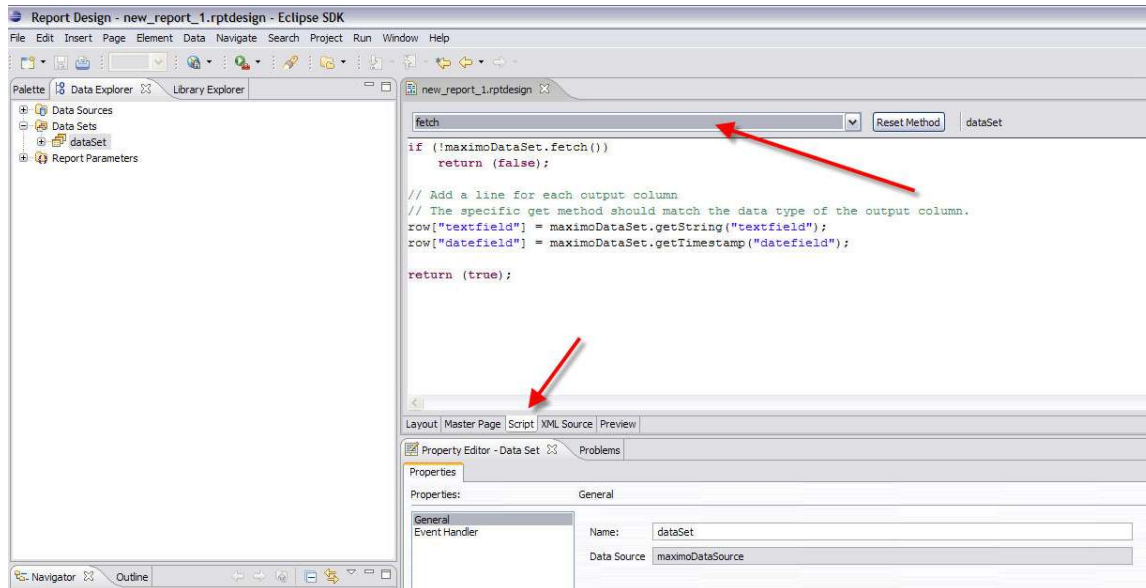
Notes:

1. You do not have to give the output columns the same names as the database fields, although it is usually easier to do so.
2. The following Database Types are not supported in reports: BLOB, CLOB, CRYPTO and CRYPTOX.
3. YORN fields are stored in the database as numbers (0 and 1) but are presented in V7 as localized text. The `getBooleanString(String attributeName)` method will perform both tasks: retrieve the numeric value and translate it to the appropriate text. You also can obtain the translated value from the integer using `getBooleanString(int intValue)`.
4. DURATION is stored in the database as a number (decimal hours) but in V7 it is presented as a string in the format HH:MM. The `getDuration` method will return the formatted string. If you require the numeric value instead, you can use `getDecimal`. An additional utility method, `MXReportUtil.getDuration(String attributeName)`, is provided to perform the conversion from double to string.
5. If you leave the open method visible as you do this, you can use it for reference on the columns.

Updating the Fetch

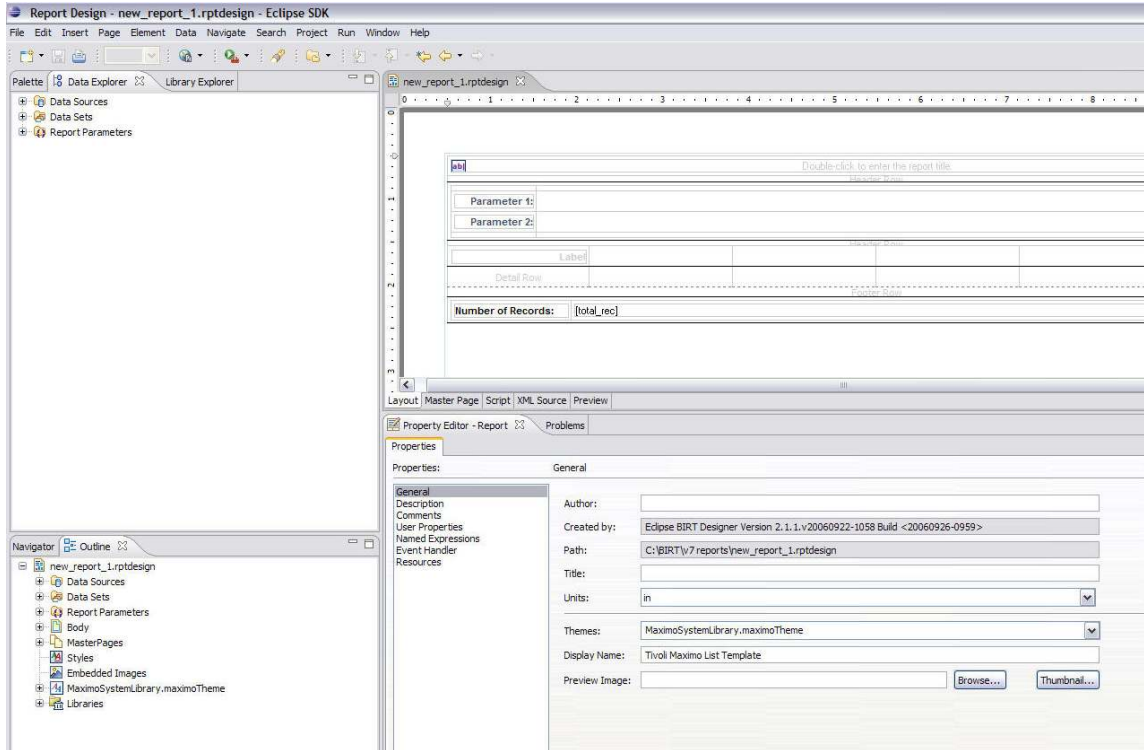
In the third step, you need to map the query columns to the output columns. This is done by updating the fetch method.

On the Script tab, choose the Fetch method from the dropdown. Add a line for each column that retrieves the value of the field from 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 Maximo-BIRT Data mapping chart shown in the section above.



Formatting the Report

Begin formatting your report by dragging the fields from the data set to the report. Set a fixed width for each column, otherwise the report will not format correctly in PDF.



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.

Formatting Notes

1. All Table elements should have widths set to 100%. Some templates included fixed table widths (in inches) and this is incorrect. You can also remove the height if it is set.
2. The style "titlesub" can be used for text that appears directly under the title. Examples of out of the box reports that use this style are detail reports, like Work Order Details, for the detail report key and description.
3. All subreports exist in a single cell, stacked on top of each other.
4. To receive a page break after the last subreport, a group was added. The group key is set to the unique key for the report - for example in Person Details (person_details.rptdesign) it is set to Personid. The page break after property on the group is set to "Always excluding last".

Now 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. The report footer rows have been deleted, again because this would cause a trailing blank page.

5. If you try to view your report within the designer as 'View as PDF', it will not work unless you install the iText jar. You will receive this error:

org.eclipse.birt.report.service.api.ReportServiceException: Report engine fails to create extension to handle this request.

6. If you want to change the font of the reports to a Unicode or other format, you should change the Style Sheet used in the Report Library.

The default fonts used in the out of the box reports are Verdana, Arial, Sans-serif.

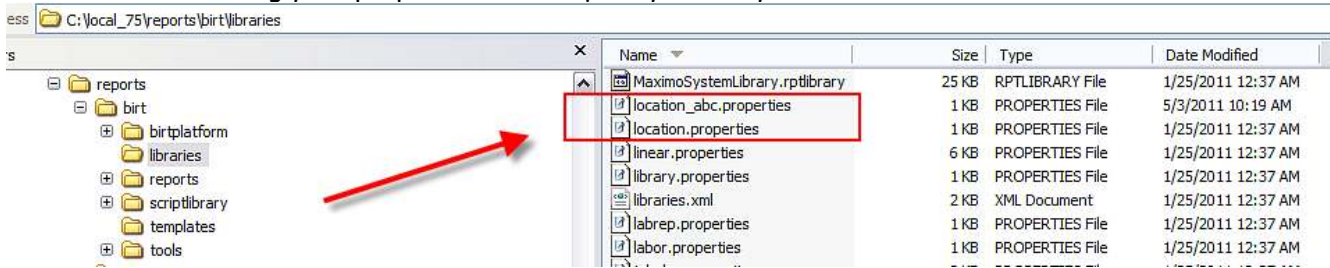
The font used will depend on what is available from the user's browser. It will start with Verdana, and if that is not available, it will use Arial and then Sans-Serif.

Defining the Property File

As noted in the Report Design File Structure Section, properties files hold the text values for the labels and titles used within a report. Additionally, in the sub-section titled 'For New Custom or Modified Reports - Properties file', it detailed how you can choose to either modify the existing properties file for the application, or create your own new properties file. The main items to take into consideration are recapped here below -

1. A single report design can only reference a single properties file.
2. Applications can utilize multiple properties file. During the command import process, all properties file for the application will be imported.
3. Report titles, labels may be modified during release, fix pack or hot fix updates. Therefore, if you modify the delivered properties file with your customizations, your updates may be overridden during an update.

Based on this, you may want to create your own custom properties file, by copying the delivered file and then renaming your properties file to quickly identify it.



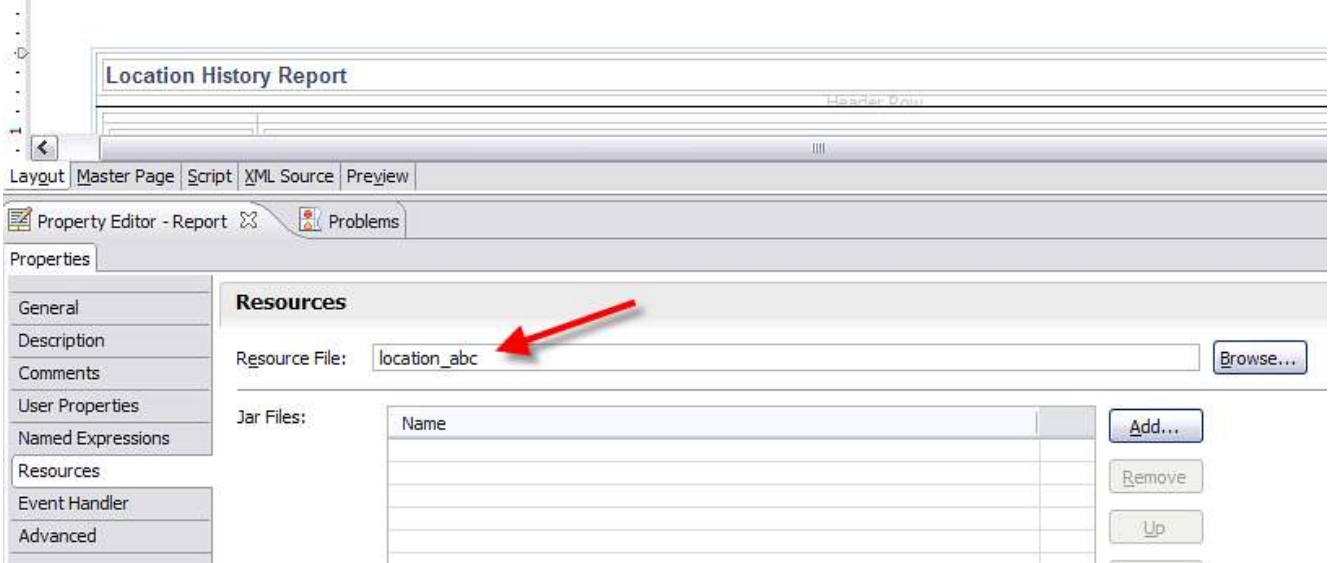
Then, when your developer adds new labels for your new reports, he will add them to the custom properties, location_abc.properties as shown in the steps below in the Report Designer.

Defining the Property File - Specific Steps

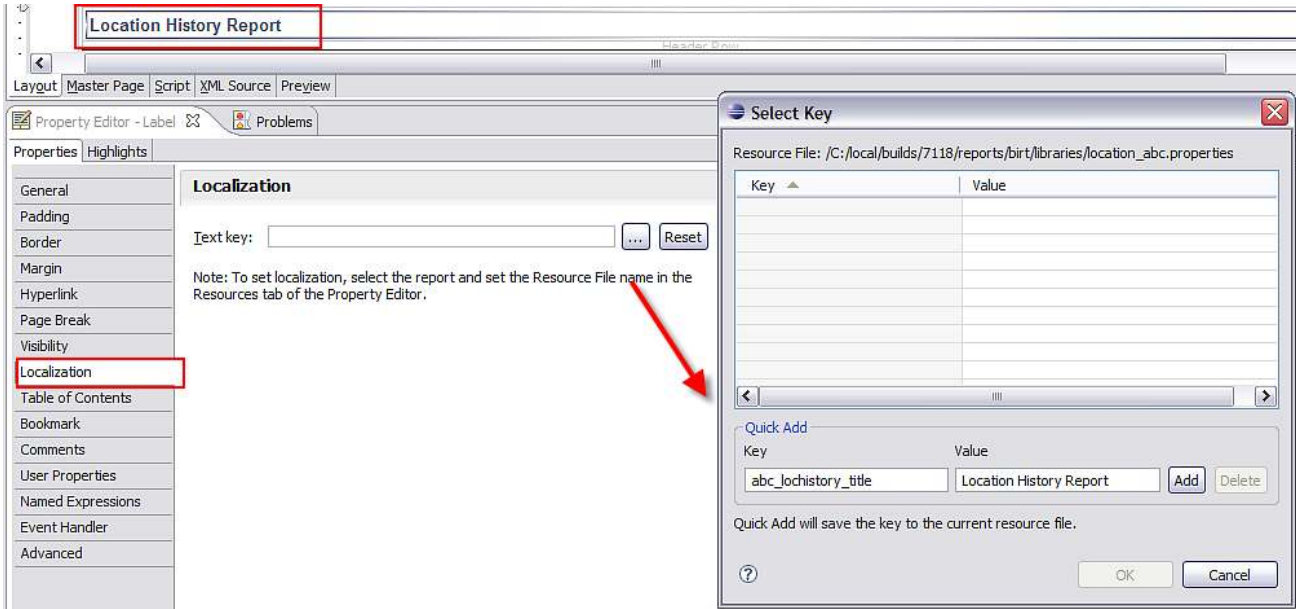
The specific steps to associate the report with a label properties file are detailed below.

1. Within the report designer, in the Outline tab, highlight the report name. In Property Editor - Properties, select Resources. Click on the 'Add' Button in the Resource File Field.

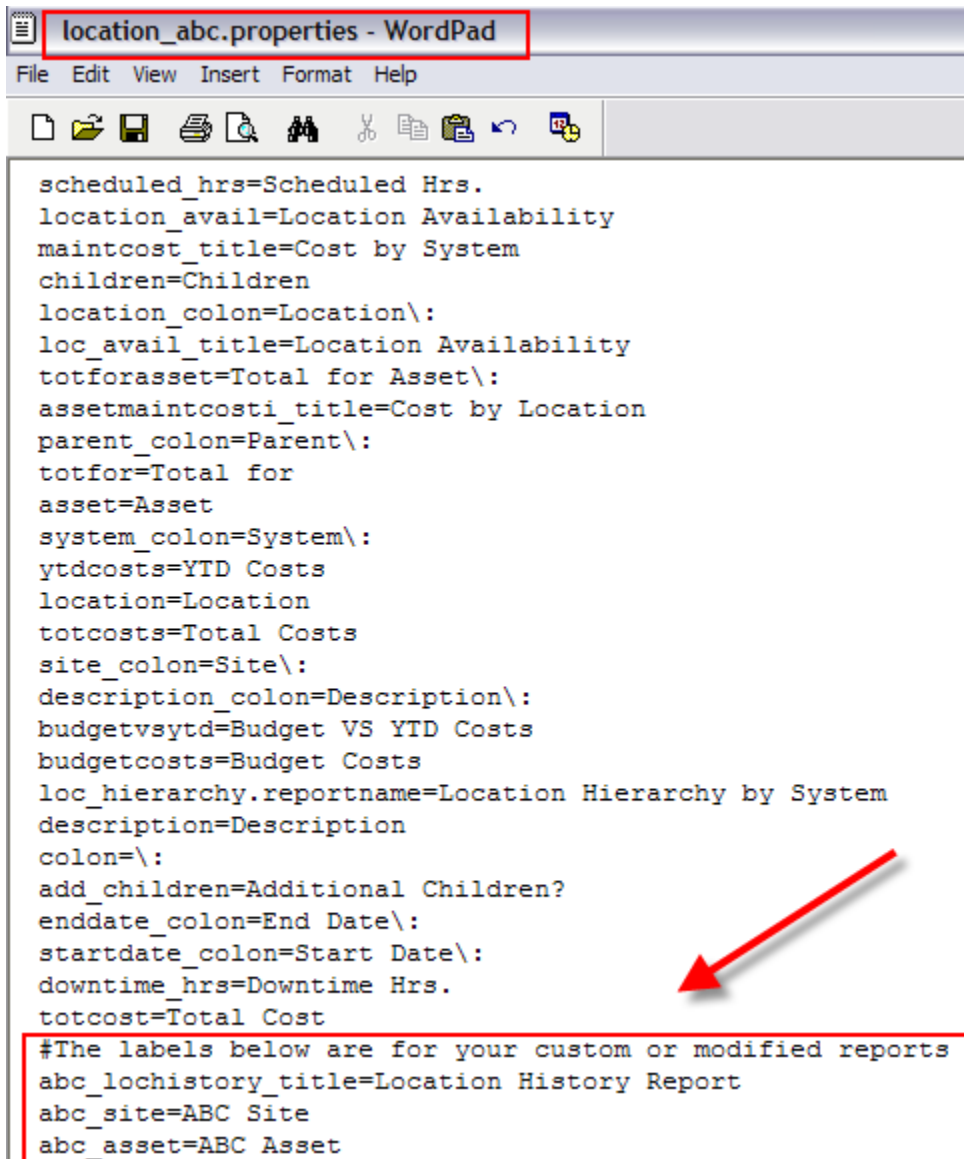
Browse to the location of your properties file. In this case, the copied location_abc properties file is chosen.



2. Now, you can build the properties file by highlighting either a title or a label. Then, select the Localization property on the left, and click the browse icon to either select an existing label or add a new label.



3. After your developer has completed creating the properties file, you may want to add an identifier to the properties file to highlight where your custom labels start as shown below. This will enable you to quickly identify them for future fix pack, version upgrades.



```
location_abc.properties - WordPad
File Edit View Insert Format Help

scheduled_hrs=Scheduled Hrs.
location_avail=Location Availability
maintcost_title=Cost by System
children=Children
location_colon=Location\:
loc_avail_title=Location Availability
totforasset=Total for Asset\:
assetmaintcosti_title=Cost by Location
parent_colon=Parent\:
totfor=Total for
asset=Asset
system_colon=System\:
ytdcosts=YTD Costs
location=Location
totcosts=Total Costs
site_colon=Site\:
description_colon=Description\:
budgetvsytd=Budget VS YTD Costs
budgetcosts=Budget Costs
loc_hierarchy.reportname=Location Hierarchy by System
description=Description
colon=\:
add_children=Additional Children?
enddate_colon=End Date\:
startdate_colon=Start Date\:
downtime_hrs=Downtime Hrs.
totcost=Total Cost
#The labels below are for your custom or modified reports
abc_lochistory_title=Location History Report
abc_site=ABC Site
abc_asset=ABC Asset
```

Report Development Considerations

Date Methods

The MXReportSqlFormat static methods are provided to support date formatting. Most of them return JDBC date/time/timestamp literals that can be used in report SQL statements for all supported databases. For example:

```
"where actualdate <=" + MXReportSqlFormat.JdbcDateFormat.DATE.format(MXReportSqlFormat  
.getCurrentDateFunction())
```

evaluates to:

```
where actualdate <= { ts '05-01-2011 13:22:45' }
```

Below are the recommended date methods

getCurrentDateFunction() - current date as java.util.Date

JdbcDateFormat.DATE.format(Date d)* - JDBC DATE literal based on date input

JdbcDateFormat.TIME.format(Date d)* - JDBC TIME literal based on date input

JdbcDateFormat.TIMESTAMP.format(Date d)* - JDBC TIMESTAMP literal based on date input

restrictBetweenDays(String field, Date start, Date end) * - creates a JDBC TIMESTAMP literal restriction on field as: ([field] >= [start, zeroed time] and [field] < [end, added 1 day, zeroed time])

Example: Restricting asset on asset.statusdate within 1 date interval where startdate is 2010/05/01 12:32:55 and enddate is 2011/05/01 8:03:02

```
var where = MXReportSqlFormat.restrictBeweenDays("asset.statusdate", params["startdate"],  
params["enddate"])
```

The variable where will contain:

```
( asset.statusdate >= { ts 2010/05/01 0:0:0 } and asset.statusdate < { ts 2011/05/02 0:0:0 } )
```

restrictBetweenDateLiterals(String field, String start, String end) * - useful when you already have the JDBC literal TIME, DATE or TIMESTAMP output is: ([field] >= [start] and [field] < [end])

ModifyDate.NEXT_DAY.skip(Date d) * - returns new date based on [d] one day ahead, time is unmodified

ModifyDate.PREVIOUS_DAY.skip(Date d) * - returns new date based on [d] one day backwards, time is unmodified

ModifyTime.STARTDAY.set(Date d) * - returns new date based on [d] with time portion 0:0:0.000

ModifyTime.ENDDAY.set(Date d) * - returns new date based on [d] with time portion 23:58:59.998

Note: Methods noted with an asterisk were introduced in the 7.5 release

Note: You may find the functions below in some of the delivered reports. These functions can be used, however, the ones noted above are the preferred functions moving forward.

`getCurrentTimestampFunction()` - JDBC TIMESTAMP literal based current date & time

use `JdbcDateFormat.DATE.format(getCurrentDateFunction())`

`getDateFunction(Date d)` - JDBC DATE literal based on Date d input

use `JdbcDateFormat.DATE.format(Date d)`

`getTimeFunction(Date d)` - JDBC TIME literal based on Date d input

use `JdbcDateFormat.TIME.format(Date d)`

`getTimestampFunction(Date d)` - JDBC TIMESTAMP literal based on Date d input

use `getEndDayTimestampFunction(Date d)`

`getStartDayTimestampFunction(Date d)` - JDBC TIMESTAMP literal based on date input, with time component set at start of day (for start date parameters)

use `ModifyTime.STARTDAY.set(Date d)`

`getEndDayTimestampFunction(Date d)` - JDBC TIMESTAMP literal based on date input, with time component set at end of day (for end date parameters)

use `ModifyTime.ENDDAY.set(Date d)`

Date Formats

BIRT offers custom date formatting. However, due to localization issues, you are strongly encouraged to use only Date/Time controls using Short, Medium or Long Date/Time formatting will be used.

All 'Out of the Box' Reports will use the date formatting below:

For Dates: Short Date

3/29/11

For Date/Time: General Date

March 29, 2011 4:03:00 PM EDT

When both the date and time need to be displayed in a condensed format - for example, target start, actual start, target finish etc - two controls will be used. These are:

Short Date + Medium Time.

So, within a Work order report where Actual Start within a column needs to display, the field would show as 4/26/11 4:12:34 PM and would be created by using 2 controls: Short Date + Medium Time.

Linking Result Sets

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

From the Fetch query example above:

```
sqlText = "select description from classtructure where classtructureid=?";  
classStrucDataSet.setQuery(sqlText);
```

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

In this example, the parameter is set to the value of a field in a data set. The field is a string so the data set `getString` method is used. The `getTimestamp` method may also be used but the fetch methods that return primitive data types cannot; instead use the following:

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

The other common situation where you must link result sets is when linking subreports. Subreport queries are similar to Open method queries in that they are both executed each time a record in the main query is fetched. The main difference is that subreport queries should 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:

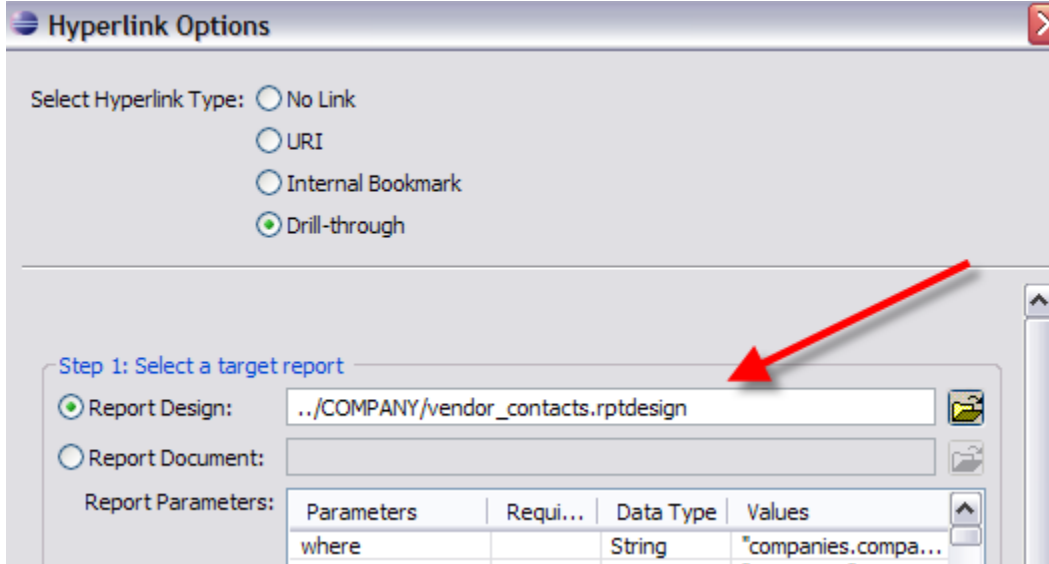
```
sqlText = "select laborcode, craft from labtrans where refwo = "  
+ rows[0][ "wonum" ] + " and siteid = " + rows[0][ "siteid" ] + """;
```

Hyperlinking

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 at least create a placeholder .rptdesign for the target report, in the correct application folder and with the correct file name. When initially specifying this, the target 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 window. 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, just enter the report name. If the report is in a different folder, use the relative path.

An example of this is the delivered PO Details report. Its report design file is located in the PO folder, and it has a hyperlink on the field Vendor to the Vendor Details report located in the COMPANY folder. Therefore, the Report Design of its target report would be:
..\COMPANY\vendor_contacts.rptdesign



3. In the Report Parameters area, add the following parameters:
 - a. Select the where parameter. In the Values field, enter a where clause that specifies the relationship between the current row and the linked report.
 - b. Select the appname parameter. In the Values field, enter params["appname"] if the linked report is registered to the same application as the calling report. If it is registered to another application, enter the correct application name, for example "PO".

For example, to link from PO List to PO Details, enter poprint.rptdesign for the target report, and then create the following parameters. Include the quotes as shown:

where	"poline.ponum='" + row["ponum"] + "' and poline.siteid='" + row["siteid"] + "'"
appname	params["appname"]

4. Under "Show target report in", select "Same Frame".

Notes on Hyperlinks:

1. If you are hyperlinking to a report, and a data restriction is in place, make sure to qualify the table (object) name. If it is not qualified, the hyperlinked report may display blank data.

For example, if the report is registered in the SR application

The query should not be: pmcomtype is null and status not in ('DRAFT')

Instead, the query should be qualified as: ... sr.pmcomtype is null and sr.status not in ('DRAFT')

2. If you design a report to have hyperlinks targeted to the same report, the report output may not change after drilling though more than once from the initial link.

This occurs as a `__requestId` internal parameter is used to distinguish each report executed by a user from the browser. This `__requestId` parameter value is unique within the user's current session for the report that is executed.

When hyperlinks are involved, the Report URL for the hyperlink is generated by the report server and does not contain this internal `__requestId` parameter. Therefore, this parameter will have a value of null for all hyperlinks. Typically, if a hyperlink is for a different report, the null value and the combination of the hyperlinked report name act as a unique key to distinguish the report execution. But, if the hyperlink is for the same report, then any two such links to the same report will be treated as equal, as the key becomes the same.

In the V7 report integration, this unique key is used to get rid of the temporary files created when a report is run again. (For example, if the same report is run again, then the previous report information is discarded using the previous key stored in the HTTP session.) This minimizes the generation of temporary files for repeated execution of the same report. When this logic is combined with the hyperlinks to the same report, the temporary files are never deleted, as multiple executions are treated the same, because the keys are identical. Because of this, the report output does not appear to change.

To resolve the problem, the hyperlink creation has to be forced to generate the `__requestId` parameter. This can be accomplished by adding a parameter to the hyperlink.

```
<structure>  
  <property name="paramName">__requestid</property>  
  <expression name="expression">java.lang.System.currentTimeMillis() + (hyperlinkCounter++)</expression>  
</structure>
```

Note that the expression has to have a unique key that is unique to the current user and the current report. Since a report can have multiple hyperlinks, be sure to generate links that are unique within the report for that user's execution. Additionally, the `hyperlinkCounter` has to be declared in the initialization of the report script code.

3. If a user hyperlinks from one report to another, no additional code is required for localization. The language code is passed through internal report context and is not passed as part of the hyperlink.

- Whether the report is a regular report or a hyperlink report, the report has to go through a single servlet that knows about the already logged in user and the user's locale/languagecode/timezone information. This information is automatically passed to the report engine or to the scripting code through a framework provided report context.

*Note: For more details on localization and how to enable for reporting, reference the V7 Report Localization Guide.

4. Many out-of-the box reports contain hyperlinks. You may want to review their specific source code for more examples of how hyperlinks are set. To locate which reports contain hyperlinks, access the V75 Report Booklet referenced at the end of this guide, and at the url below. Search the 'V75 Reports' tab for hyperlink to find delivered reports with this functionality.

http://www-01.ibm.com/support/docview.wss?rs=3214&context=SSLKT6&q1=BIRT&q2=BOOKLET&uid=swg21305005&loc=en_US&cs=utf-8&lang=en

Populating the Data Set

If you need more than one data set (usually only required when creating subreports), you may wish to make a copy of the existing data set before starting.

Closing the Data Set

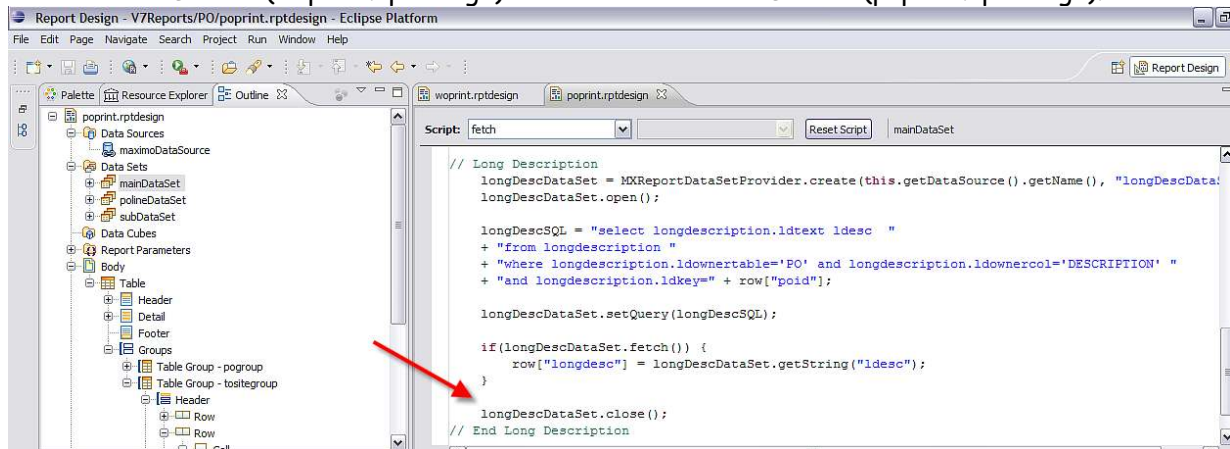
Any data set that is not fully fetched in a loop must be explicitly closed so cursors do not remain open after the report is executed. If the data set is not closed, and the same report is continually executed from the V7 instance over many days, the report may cause a failure of all reports to not show data.

An example of this is shown below using the Long Description data set as an example. Because only one row is fetched from this set, this type of fetch is not closed automatically, and therefore, must change to

```
if(longDescDataSet.fetch()){  
    row["longdesc"] = longDescDataSet.getString("ldtext");  
}
```

`longDescDataSet.close();`

Out of the box reports that contain examples of closing the long description data set include **Work Order Details (woprint.rptdesign)** and **Purchase Order Details (poprint.rptdesign)**.



Note: This is an example of running queries in the `fetch()` method in the 'Executing Additional Queries' section below that demonstrate that the data set should be closed.

Executing Additional Queries

Additional queries may be run in both the Open and Fetch methods. Each method can have one or more additional queries returning one or more fields.

Queries in the Fetch Method

It is sometimes difficult to provide all data fields for a report with a single SQL statement. You can populate most of the output columns with the main query, and run additional queries to retrieve the remaining fields, for 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 classtructure where classtructureid=? ";
classStrucDataSet.setQuery(sqlText);

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

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

return(true);
```

Note: queries that are executed multiple time should use setQueryParameterValue() for caching improvements.

Dynamically Filtering Data

There are several situations in which you will need to apply a dynamic filter to a report SQL statement. You may filter report results using Report Parameters, which receive values passed from Maximo. You may also use dynamic filters to link multiple queries.

Testing for Null

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

`maximoDataSet.getNullValueFunction(String param, String nullVal)` - Returns *NVL*, *ISNULL*, or *COALESCE* depending on the database type. For example:

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

evaluates to:

```
select nvl(parent, wonum) - for Oracle
```

```
select coalesce(parent, wonum) - for DB2
```

```
select isnull(parent, wonum) - for SQL Server:
```

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

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

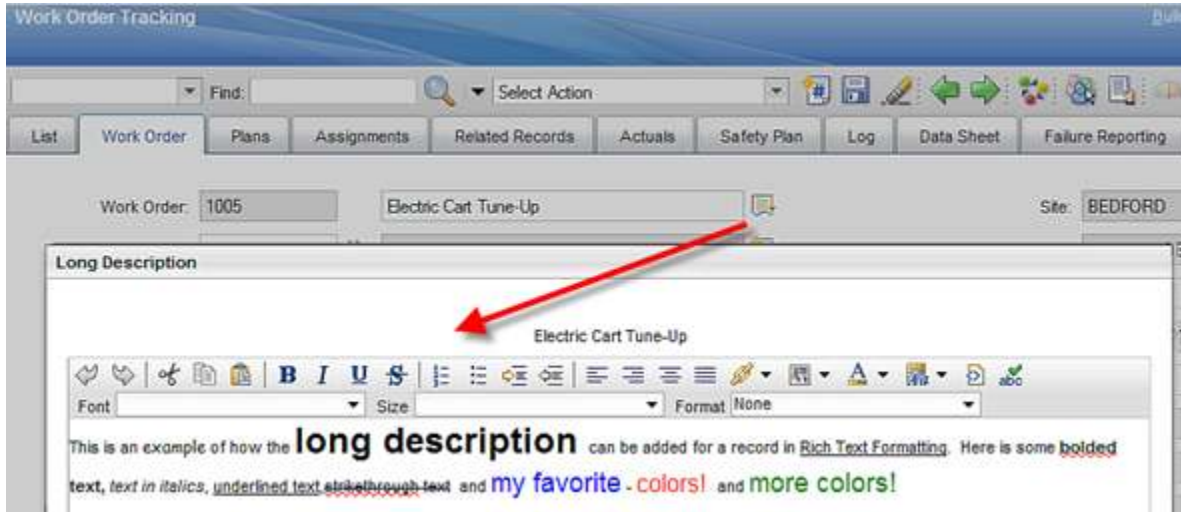
However, be careful with using string literals this way, since they will not be localized.

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. This can be used to access database functions in a database independent manner as suggested in the JDBC specification for commonly used functions.

Enabling Rich Text Formatting

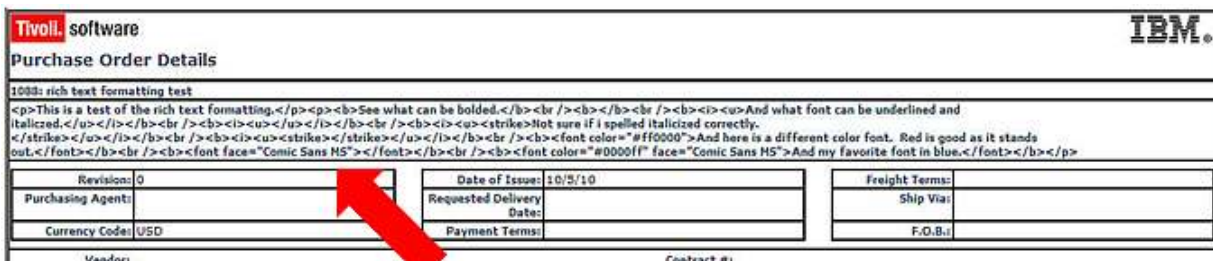
Beginning in the Version 7.5 release, you can input rich text in the long description fields of applications. This enables you to highlight certain areas of critical information for your users. An example of all the various types of font features that can be enabled is shown below in the work order tracking application.



This rich text font can be displayed in reports. For all the out of the box reports using long description fields, the rich text font has been enabled. For a full listing of these reports, reference the V75 report booklet at the end of this document.

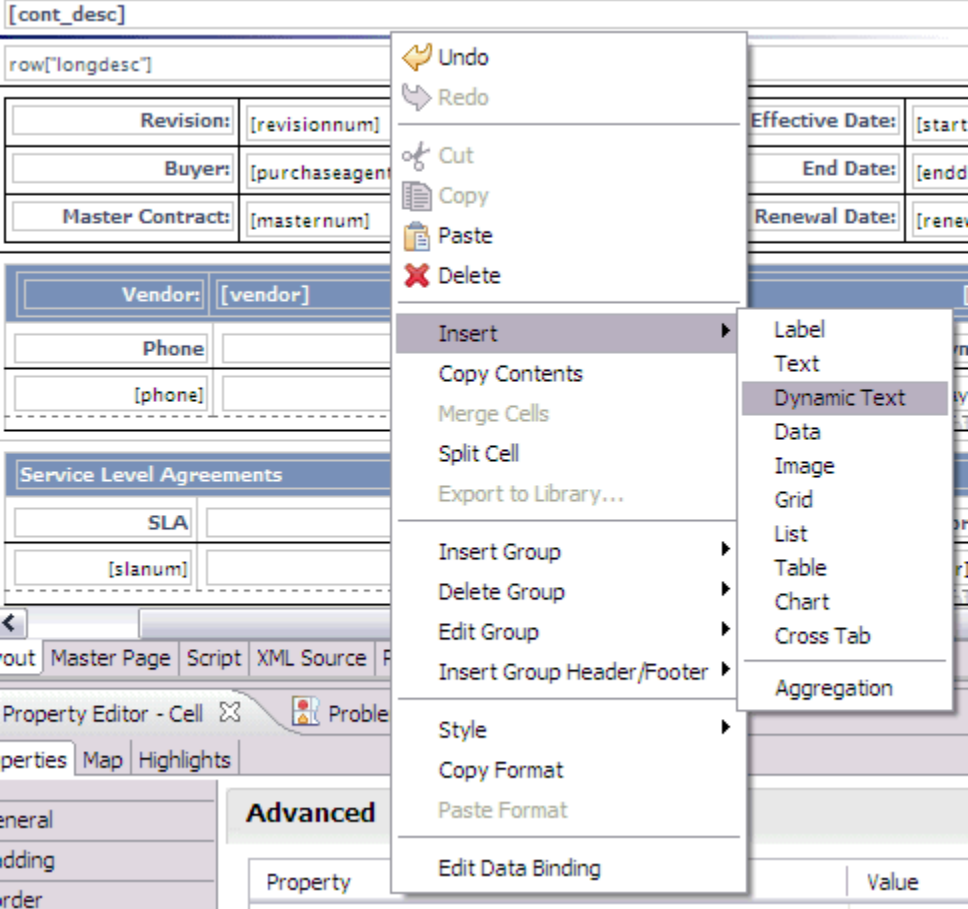
You may have created custom reports which include the long description field. The steps to update these custom reports to use rich text formatting are shown below.

Note: If you do not update your custom reports using long description fields - and your users input rich text and then print out the custom report - they will receive illegible text as highlighted by the red arrow below.



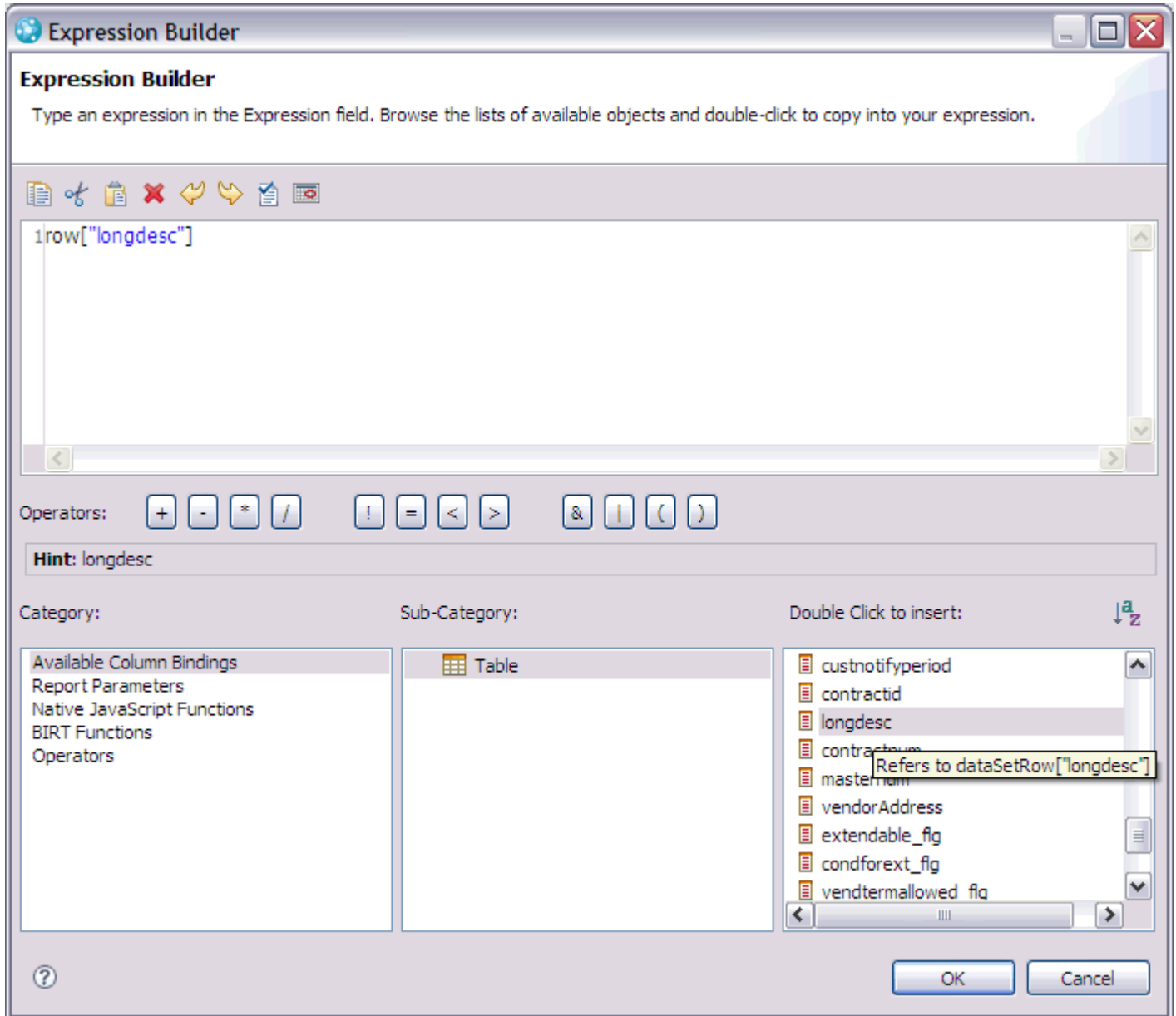
To enable your custom report to display rich text formatting in the long description field, follow these steps in the report designer.

- A. If the long description field exists in the report, delete it.
- B. Next, insert a Dynamic Text field.



C. The Dynamic Text property has the HTML property set. Add the attribute of the field, longdesc.

Note: The long description attribute will vary depending on how it is defined in the query.



After saving and importing the report into the V7.5 instance, the long description field in the report will display the rich text formatting as it was input in the application as shown below.

Tivoli. software
IBM®

Work Order Details

1005: Electric Cart Tune-Up

This is an example of how the **long description** can be added for a record in Rich Text Formatting. Here is some **bolded text**, text in *italics*, underlined text, ~~strikethrough text~~ and **my favorite** - colors! and **more colors!**

Asset: 12300 Electric Cart
 Location: SHIPPING Shipping and Receiving Department
 Ct:

Sched Start:		Site: BEDFORD	Job Plan: JF12300
Sched Finish:		Priority: 7	Supervisor: MILLER
Target Start: 12/31/98		Work Type: CM	Lead:
Target Finish: 12/31/98		Status: APPR	Vendor:

Parameters

Report parameters are used to filter the report data to meet the user's individual business needs or request. V75 reports can execute against a variety of parameter types depending on how they are configured. The three options are:

1. Parameterized Reports
2. Application Reports
3. Both Parameterized and Application Reports

Reference: For more details on the functionality of each of these parameters, reference the v7 Report Design Guide or V75 Report Feature Guide noted at the end of this document.

This section will focus on Parameterized Reports, and their two corresponding types: Bound and Unbound.

Bound Parameters

Bound parameters either

- exist in the main table of the application the report is registered to or
- exist via a maxrelationship that has been set up for the application.

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

An example of a bound parameter is the Security Group parameter in the Security Group report. Its corresponding entry is shown below from the Report Administration application. Notice its attribute value. Bound parameters will ALWAYS have the Attribute Name Field Populated - whereas Unbound Parameters will NEVER have the Attribute Name field Populated.

The screenshot shows the 'Report Administration' interface. At the top, there are tabs for 'List', 'Report', 'Security', and 'Performance'. The 'Report' tab is active. Below the tabs, there are fields for 'Report File Name' (security_group.rptdesign), 'Report Type' (BIRT), 'Imported by' (MAXADMIN), 'Application' (SECURGROUP), 'Report Folder' (SECURGROUP), and 'Last Import Date' (12/22/08 8:45 AM). A red arrow points to the 'Report File Name' field.

Below these fields is a 'Settings' section with a 'Parameters' table. The table has columns for 'Parameter Name', 'Attribute Name', 'Sequence', and 'Display Name'. The 'securitygroup' parameter is selected, and its details are shown in a 'Details' section below the table. A red arrow points to the 'Attribute Name' field in the details section, which is populated with 'GROUPNAME'.

Parameter Name	Attribute Name	Sequence	Display Name
securitygroup	GROUPNAME	1	Security Group
independent	INDEPENDENT	2	Independent
pwduration	PASSWORDDURATION	3	Password Lasts this Nu
groupuser	GROUPUSER.USERID	4	User Members

Details

Parameter Name	securitygroup	Display Sequence	1
Attribute Name	GROUPNAME	Required?	<input type="checkbox"/>
Lookup Name		Multi-Lookup Enabled?	<input checked="" type="checkbox"/>
Display Name	Security Group	Default Value	
		Operator	

Unbound parameters

- do not exist in the main table of the application and
- are not available through any relationship (defined in maxrelationship) for the main table.
Unbound parameters are not included in the where clause.

An example of an unbound parameter is the User parameter in the Electronic Signature Transaction report. This parameter is unbound because it does not exist in the main table of the application (CONFIGUR) and does not exist in one of the maxrelationship to this application. This is shown below. Notice that its Attribute Name field is blank.

The screenshot shows the 'Report Administration' interface. At the top, there's a search bar and a 'Select Action' dropdown. Below that are tabs for 'List', 'Report', 'Security', and 'Performance'. The 'Report' tab is active, showing details for the report 'esig_trans.rptdesign' (Electronic Signature Transaction) in the 'CONFIGUR' application. The report type is 'BIRT' and it was imported by 'MAXADMIN' on 12/22/08 at 8:44 AM.

Under the 'Settings' section, there's a 'Parameters' table with 4 columns: Parameter Name, Attribute Name, Sequence, and Display Name. The 'user' parameter is selected, and its details are shown in the 'Details' section below. The 'Attribute Name' field for the 'user' parameter is blank, which is highlighted by a red arrow. Other fields in the details section include 'Display Sequence' (1), 'Required?' (unchecked), 'Multi-Lookup Enabled?' (checked), 'Default Value' (empty), and 'Operator' (empty).

Parameter Name	Attribute Name	Sequence	Display Name
user		1	User(s)
application		2	Application(s)
startdate		3	Start Date
enddate		4	End Date

Details	
Parameter Name	user
Attribute Name	
Lookup Name	
Display Name	User(s)
Display Sequence	1
Required?	<input type="checkbox"/>
Multi-Lookup Enabled?	<input checked="" type="checkbox"/>
Default Value	
Operator	

The Chart below details each of the fields available for parameters in Report Administration (and its reports.xml file) , and whether or not they should be populated for bound versus unbound parameters.

	Bound	Unbound
Advantage	Can have lookups, and do not need to be defined in report's design.	Flexibility.
Parameter Name	Do not need to be defined in Report's design file	Must be defined in Report's design file
Attribute Name	ALWAYS Populated	NEVER Populated
Lookup Name	Can either be populated or not	Can only be used for Unbound Dates (*DateLookup Only)
Operator (>, >=, <, <=)	Optional	NEVER Populated
Multi-Lookup Enabled?	Yes or No	Yes or No
Display Sequence	Numeric Value	Numeric Value
Override Label	Any Text	Any Text
Default Value	Can either be populated or not. *NOTE: Default Values are not enabled for localization	Can either be populated or not *NOTE: Default Values are not enabled for localization
Required?	Yes or No	Yes or No
Examples	security_group.rptdesign	eSig_trans.rptdesign

Specifying Bound parameters in the report design

Bound parameters will be added automatically by V75 to the where parameter, and will be included in the SQL as follows:

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

Specifying Unbound parameters in the report design

Unbound parameters must be manually included in the report SQL. The method that you use to do this will vary. You should choose the method depending on where the query will be executed and if it will run multiple times.

Multi-select or single-select unbound parameters

Multi-select or single-select parameters enable users to enter different numbers of values for parameters. For example, you can enter values like asset1, asset2, asset3 in a multi-select asset parameter.

Multi-select parameters will be passed as a comma-delimited string, may or may not contain = or != symbols, and must be converted to the correct syntax using the `MXReportSqlFormat.createParamWhereClause()` method described previously.

For example, consider status as multi-select and worktype and owner as single select:

```
var params["where"] = "1=1";
var params["status"] = "=WAPPR, =APPR";
var params["worktype"] = "MINOR";
var params["owner"] = "O'NEAL";
"select wonum, description from workorder where "
params["where"]
+ " and "
MXReportSqlFormat.createParamWhereClause("workorder.status", params["status"])
" and " + MXReportSqlFormat.createParamWhereClause("workorder.worktype", params["worktype"])
" and " + MXReportSqlFormat.createParamWhereClause("workorder.owner", "=" + params["owner"]);
```

This will result on something similar to:

```
select wonum, description
from workorder
where 1=1
and ((workorder.status = 'WAPPR') and (workorder.status = 'APPR'))
and (workorder.worktype like '%MINOR%')
and (workorder.owner = 'O'NEAL')
```

With the `createParamWhereClause()` you can escape characters as needed, build where clause from V7.5's formatted value list and understand the operators (=, !=) provided with the values.

Also, you can use query substitution variables for subdataset queries or nested datasets. Please note however, that this option is not optimized for parameters as it is unable to deal with operators.

```
sqlText = "select asset, description from asset where " + params["where"]  
+ " and asset.siteid = ? and asset.priority = ? " + " and asset.installdate >= ? ";
```

```
maximoDataSet.setQuery(sqlText);  
maximoDataSet.setQueryParameterValue(1, params["siteid"]);  
maximoDataSet.setQueryParameterValue(2, params["priority"]);  
maximoDataSet.setQueryParameterValue(3, new java.sql.Date(params["startDate"]));
```

Finally for unbound date value parameters, you may want to use the `MXReportSqlFormat` methods that return JDBC String literals (which are database agnostic) and can be concatenated on the query directly:

```
+ " and asset.installdate >= +  
MXReportSqlFormat.JdbcDateFormat.TIMESTAMP.format(ModifyTime.STARTDAY.set(params["  
startDate"]))
```

Parsing Unbound Parameters

Unbound parameters are passed to the report in a comma-delimited string and may contain operators, so the values must be parsed before including in the report SQL. The following method is used for this:

`MXReportSqlFormat.createParamWhereClause(String columnName, String paramValue)` - 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 where the operator can be any one of `<=`, `<`, `>=`, `>`, `!=`, `=`. If no operator is specified, then it assumes that the search is based on operator SQL LIKE. For 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%'))
```

If you have unbound parameters that need to be manually included in the SQL (are not included in the where clause), *do not* directly include them as follows:

```
sqlText = "select asset, description from asset where asset.siteid = '" + params["siteid"] + "'"
```

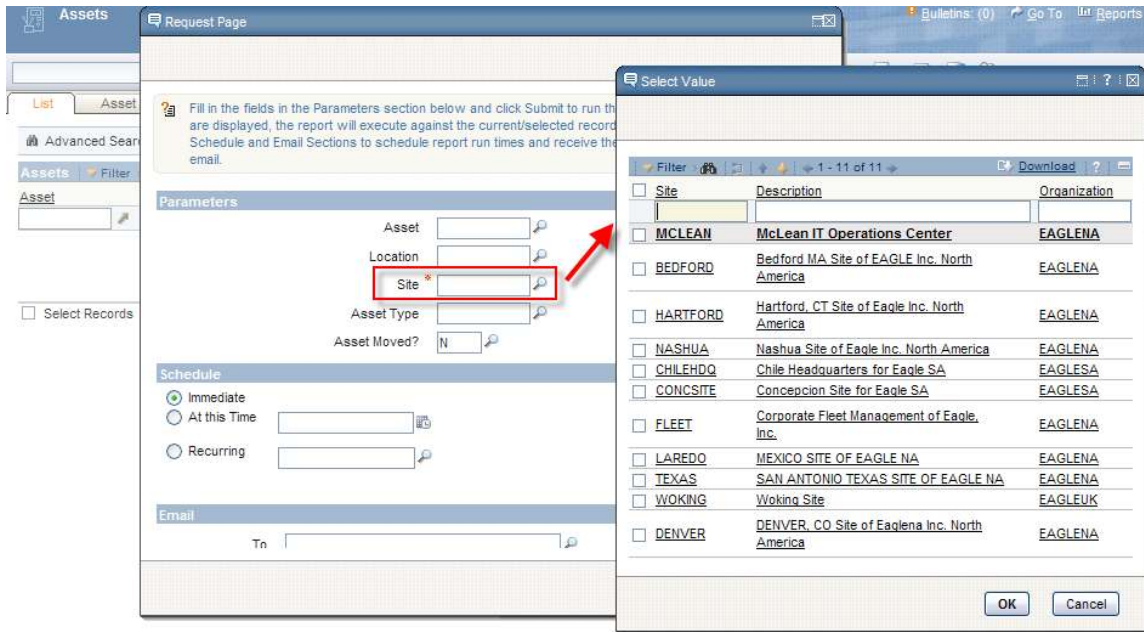
Instead, pass them through the `MXReportSqlFormat.createParamWhereClause` method:

```
sqlText = "select asset, description from asset where asset.siteid = " +  
MXReportSqlFormat.createParamWhereClause("asset.siteid", params["siteid"]);
```

It is advised to use this method on all parameters - not just multi select ones. Use `MXReportSqlFormat.createParamWhereClause("<table>.<column>", "=<value>")` when the value is known to be exact. The "=" before the value ensures output as an exact search clause while without it the clause may be generated using like `'<value>%'`.

Creating Custom Report Parameter Lookups

You may need to create parameters with lookups for your custom reports. Lookups are accessed from parameters on a report's request page. In the screenshot below, the report's request page with five different parameters is shown below. The lookup for the Site parameter is highlighted.



This section presents a variety of options for you to consider when you need to create custom lookups, including

- A. Using valuelists for parameter lookups with fields that have domains
- B. Using existing lookups
- C. Modifying existing lookups

A. Using valuelists for parameter lookups with fields that have domains

In this method, parameter lookups will be enabled by using valuelists for fields that have domains. Domains have a special status because field validation classes are not required if the field has a domain and the 'valuelist' lookup is used. Lookups for fields with domains can nearly always be used for report parameters.

To illustrate this, a lookup will be created for the Work Order Class parameter on the Version 7.5 Estimated versus Actual Work Order Cost Report.

1. Sign into Maximo as a user with access to the following Maximo Applications: Report Administration, Domains, Database Configuration and Application Designer.

2. Access the Report Administration application.

3. Search for the Estimated vs Actual Work Order Cost Report, and open up the Work Order Class parameter. The attribute name for its parameter is populated - so it is a bound parameter. However, notice its Lookup Name field is blank.

The screenshot shows the 'Report Administration' interface. At the top, the 'Report File Name' is 'wotrack_costanalysis.rptdesign' and the report is titled 'Estimated vs Actual Work Order Costs'. The 'Parameters' section shows a table with the following data:

Parameter Name	Attribute Name	Sequence	Display Name
class			
woclass	WOCLASS	3	Work Order Class

Below the table, the 'Details' for the 'woclass' parameter are shown:

Parameter Name	woclass	Display Sequence	3
Attribute Name	WOCLASS	Required?	<input checked="" type="checkbox"/>
Lookup Name		Multi-Lookup Enabled?	<input checked="" type="checkbox"/>
Display Name	Work Order Class	Default Value	
		Operator	

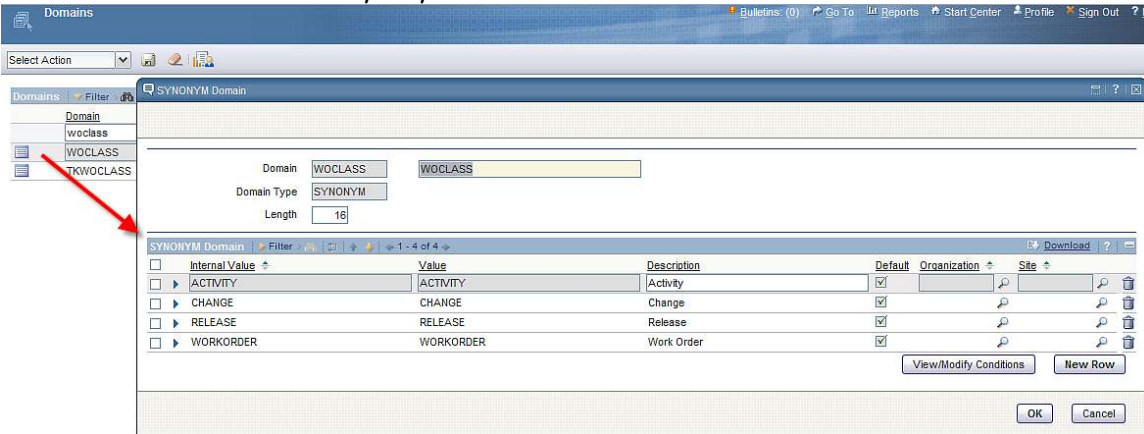
A red arrow points to the empty 'Lookup Name' field.

4. Next, verify that a domain exists for Work Order Class. Go to System Configuration - Platform Configuration - Domains, and search for WOCLASS under Domains.

The screenshot shows the 'Domains' application. The table below lists the domains:

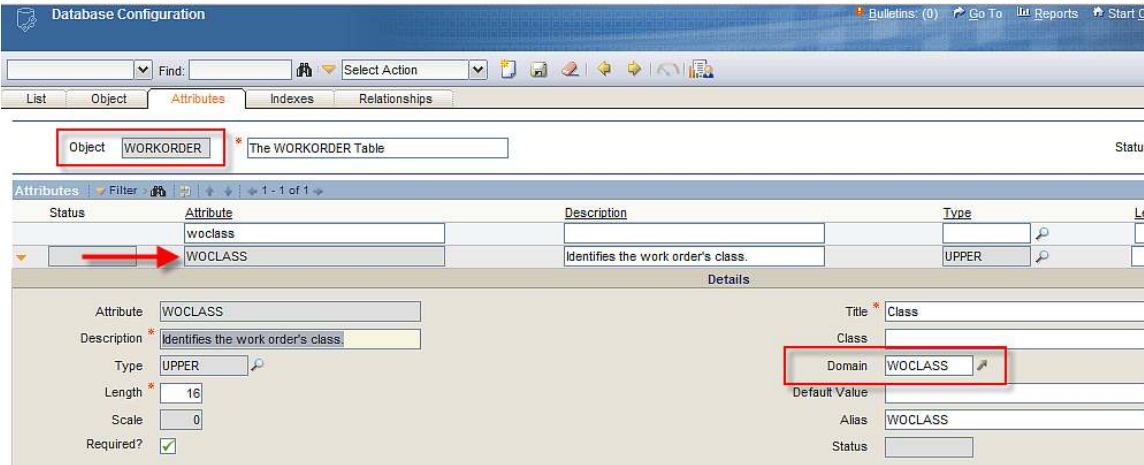
Domain	Description	Domain Type	Data Type
woclass			
WOCLASS	WOCLASS	SYNONYM	UPPER
TKWOCLASS	Combination of TKCLASS and WOCLASS dom	TABLE	UPPER

Click on its detail to see its synonym domain values as shown below.



5. Now, verify that the WOCLASS domain is associated with the WORKORDER.WOCLASS attribute. To do this, access System Configuration - Platform Configuration - Database Configuration.

- Search for the Workorder Object.
- Then, search for its attribute WOCLASS. Notice it has a Domain value of WOCLASS.



Notes on Domains:

- A. If either the domain, or the attribute's relationship to the domain did not exist, they would have to be created. Details on how to do this are described in the 'Application Developer Guide'
- B. For more information on domains, access the 'System Administration Guide'.

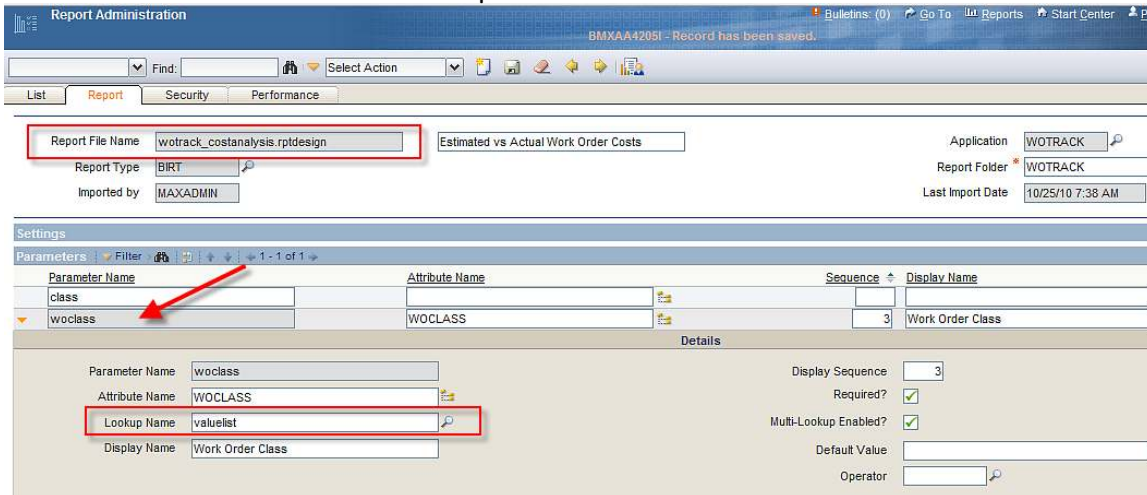
Both of these guides are available at the Information Center below:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.itmaxam.doc/welcome.htm>

6. In this step, the attribute's domain will be added to the report, so lookup values can be enabled from the parameter.

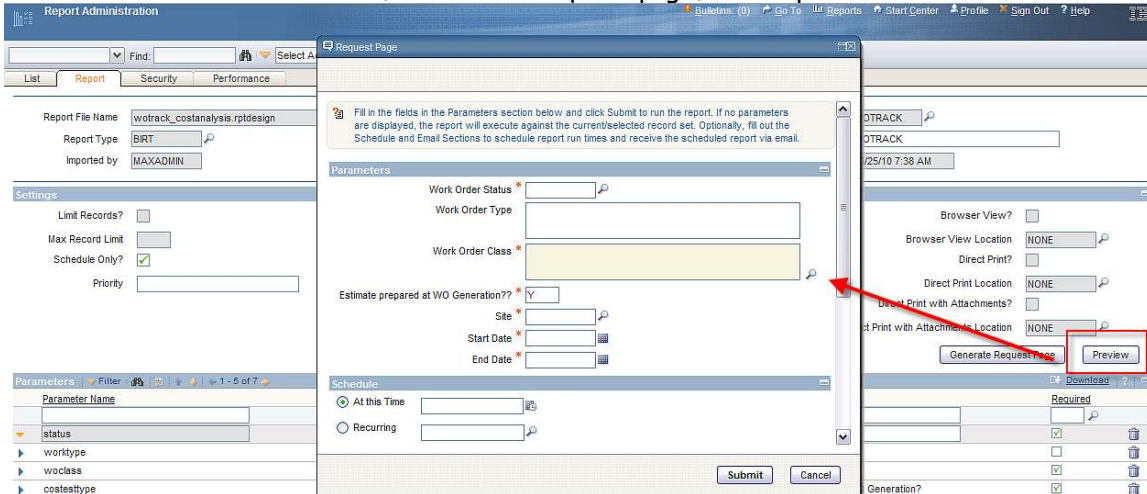
To do this, go back to the Report Administration application.

- Locate the report, and open up the work order class parameter.
- Enter valuelist in its Lookup Name field



7. Save the change, and recreate the report xml by clicking on the button 'Generate Request Page'.

8. Click on the Preview Button, and on the request page, a lookup now exists for Status.



9. Click on the Lookup next to the Work Order Class parameter, and its lookup values display.

The screenshot shows a 'Request Page' window with a 'Parameters' section. The 'Work Order Class' field has a lookup icon. A 'Select Value' dialog box is open, displaying a list of values with checkboxes. A red arrow points to the dialog box.

Parameters

Work Order Status *

Work Order Type

Work Order Class *

Estimate prepared at WO Generation?? *

Site *

Start Date *

End Date *

Schedule

At this Time

Recurring

Select Value

Filter 1 - 4 of 4

<input type="checkbox"/> Value	Description
<input type="checkbox"/> ACTIVITY	Activity
<input type="checkbox"/> CHANGE	Change
<input type="checkbox"/> RELEASE	Release
<input type="checkbox"/> WORKORDER	Work Order

B. Using existing lookups

You may be able to use existing V75 lookups with custom bound report parameters. You can see which lookups are available by searching thru the lookup on the Lookup Name field in the Report Administration applicatio.

You may find this to be a trial-and-error process since V75 lookup behavior is controlled by field classes, which are classes that are assigned to the attribute definition in Database Configuration. Many of the default lookups will not work correctly when applied to report parameters, either because there is no field class for the bound attribute, or because there is logic in the field class that inappropriately limits the results of the lookup. In these cases the lookup may return no results, a subset of the expected results, or may contain Invalid Bindings.

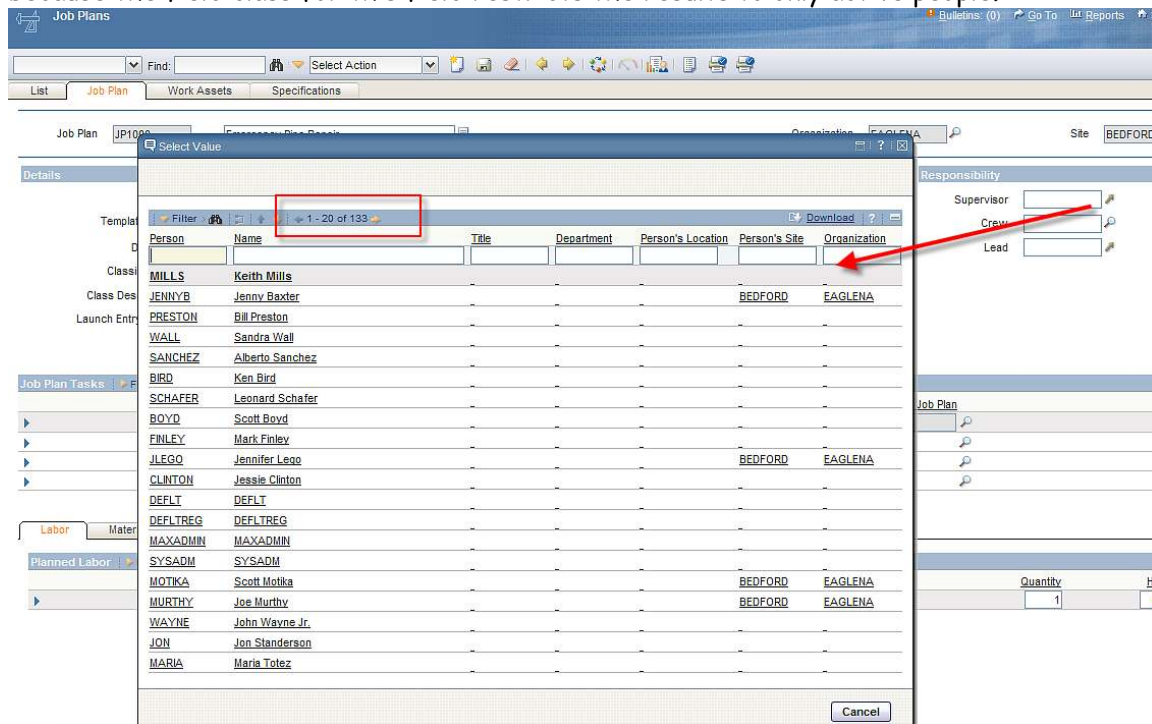
With this method, you simply try out the lookup(s) you identify as possible candidates and evaluate whether they return the desired results. You can use SQL logging to examine the query used to populate the lookup to ensure there are no inappropriate filters applied.

C. Modifying existing lookups

If the lookup attribute does not have the required field class, or the field class is not configured to provide the expected values, you can produce the desired results by creating a copy of the lookup and specifying a value for the mboname attribute. This method also has the advantage that you can modify the fields included in the lookup.

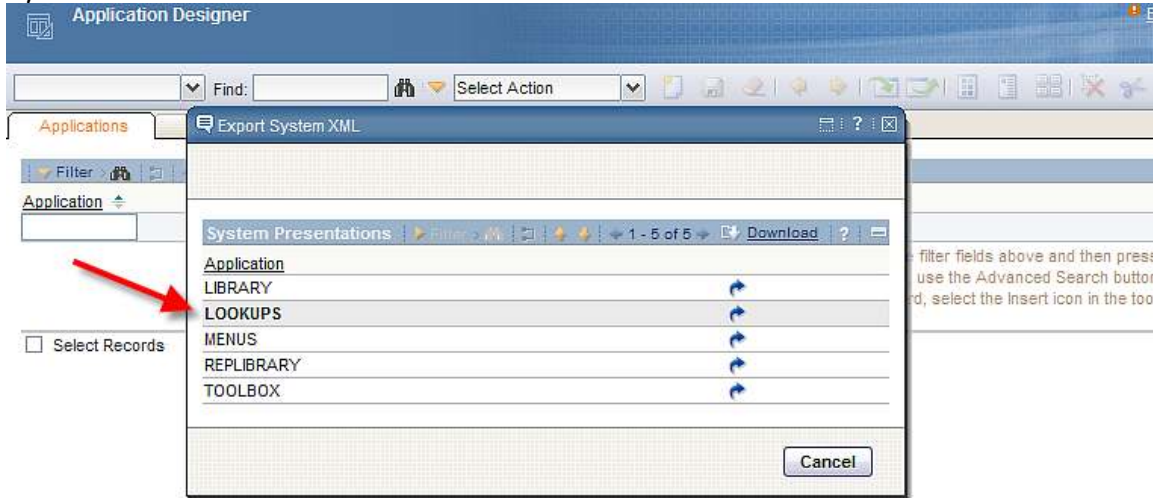
As an example, the person lookup will be modified to use with a parameter bound to the supervisor field in the Job Plan application.

This existing lookup is shown below within the Job Plan application. Notice that the lookup includes 133 people instead of the full 134 in the person table for the maxdemo database. This is because the field class for this field restricts the results to only active people.



To modify the existing lookup for reporting, follow the steps below.

1. Go to System Configuration - Platform Configuration - Application Designer and select Export System XML from the Select Action menu.



2. Open the file in a text editor. Locate the person lookup by searching for id=person. The first line of this is shown below.

```
<table id="person" inputmode="readonly" selectmode="single">
```

3. Copy the person lookup and scroll to the bottom of the file. Insert some lines before the </systemlib> .

4 Paste the copied person lookup, and then modify the first line to include the mboname attribute, for example:

```
<table id="person" inputmode="readonly" selectmode="single" >
```

Should be updated to

```
<table id="person_rpt" inputmode="readonly" selectmode="single" mboname="person" >
```

5. Replace all remaining occurrences of 'id="person"' with 'id="person_rpt"', for example:

```
<tbody id="person_lookup_tablebody" filterexpanded="true" filterable="true" displayrowsperpage="20" >
```

would become

```
<tbody id="person_rpt_lookup_tablebody" filterexpanded="true" filterable="true" displayrowsperpage="20" >
```

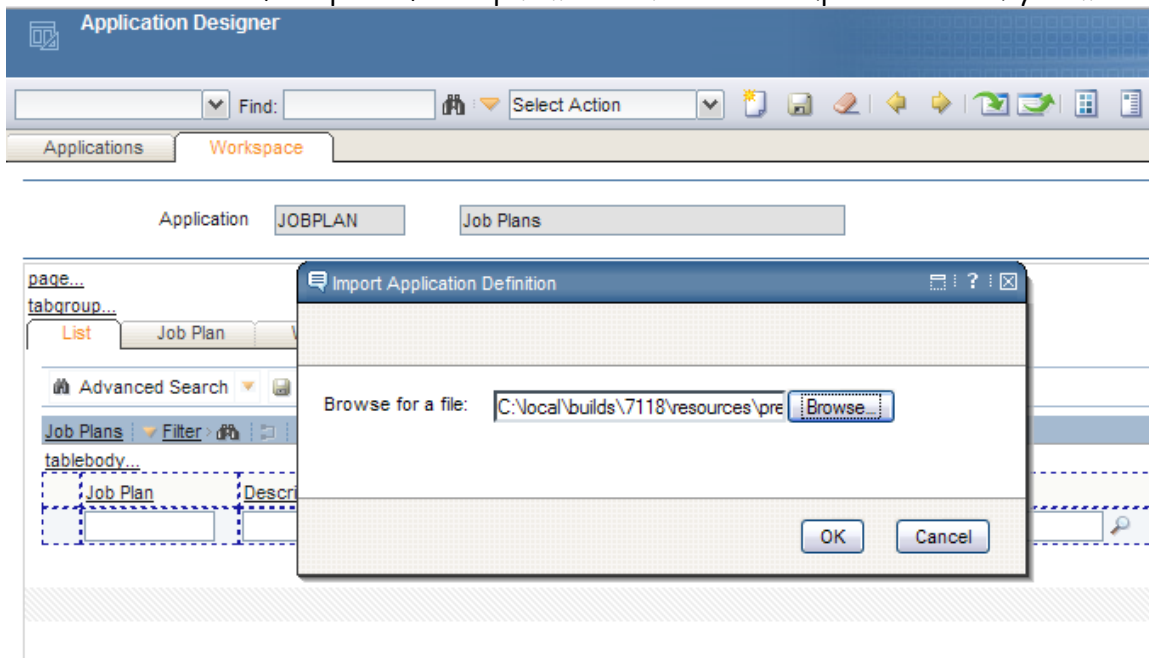
The new entire entry is shown below.

```
</tbody>
</table>
<table id="person_rpt" inputmode="readonly" selectmode="single" mboname="person" >
  <tbody displayrowsperpage="20" filterable="true" filterexpanded="true" id="person_rpt_lookup_tablebody">
    <tablecol dataattribute="personid" id="person_rpt_lookup_tablebody_col_2" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="displayname" id="person_rpt_lookup_tablebody_col_5" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="title" id="person_rpt_lookup_tablebody_col_6" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="department" id="person_rpt_lookup_tablebody_col_7" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="location" id="person_rpt_lookup_tablebody_col_8" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="locationsite" id="person_rpt_lookup_tablebody_col_9" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
    <tablecol dataattribute="locationorg" id="person_rpt_lookup_tablebody_col_10" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
  </tbody>
</table>
```

5. The changes made to the xml file now have to be imported into Version 7.5. To do this, go back to the Application Designer.

6. Click on the 'Import Application Definition' Icon in the toolbar. Browse to the location of the lookups.xml file that you modified.

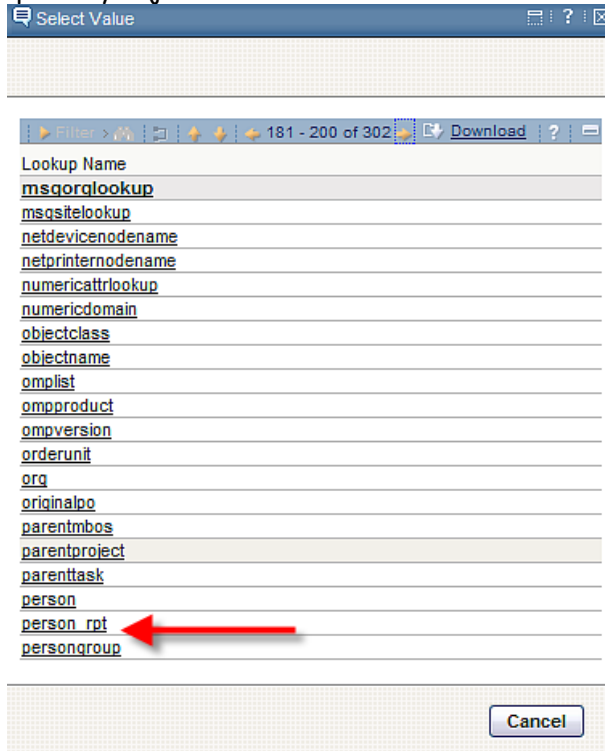
Note: the default path of lookups.xml is <V7.5>resources\presentations\system



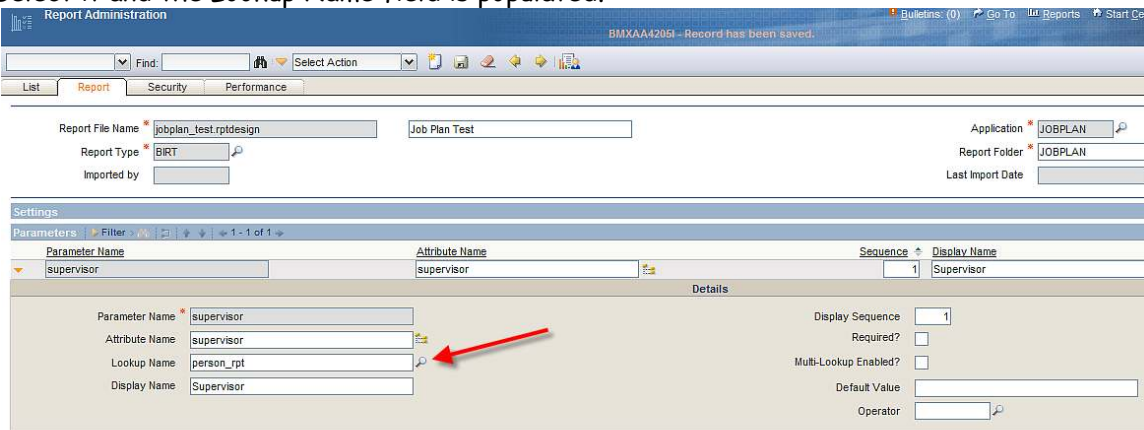
7. Click OK. When it is completed, a message will display in the toolbar.

8. Next, access the Report Administration application. Add a new sample report called Job Plan Test, with a supervisor parameter.

In the Parameter's Lookup Name field, click on the lookup. Scroll to find the new person_rpt parameter lookup that you just added.



8. Select it and the Lookup Name field is populated.



9. Save the record. Generate the report xml for the test report you have just registered.

After the XML has been created, click on Preview. Next to the Supervisor parameter, a lookup now exists. Click on the Lookup, and the values for Supervisor appear. Notice that the person records are no longer filtered, and all records appear.

Request Page

Fill in the fields in the Parameters section below and click Submit. When parameters are displayed, the report will execute against the selected person. Optionally, fill out the Schedule and Email Sections to schedule a scheduled report via email.

Parameters

Supervisor

Schedule

Immediate
 At this Time
 Recurring

Email

To
Subject
Comments

File Type
 PDF
 XLS

Select Value

Filter 1 - 20 of 134

<input type="checkbox"/>	Person	Name	Title
<input type="checkbox"/>	JENNYB	Jenny Baxter	
<input type="checkbox"/>	PRESTON	Bill Preston	
<input type="checkbox"/>	WALL	Sandra Wall	
<input type="checkbox"/>	SANCHEZ	Alberto Sanchez	
<input type="checkbox"/>	BIRD	Ken Bird	
<input type="checkbox"/>	SCHAFFER	Leonard Schaffer	
<input type="checkbox"/>	BOYD	Scott Boyd	
<input type="checkbox"/>	FINLEY	Mark Finley	
<input type="checkbox"/>	JLEGO	Jennifer Lego	
<input type="checkbox"/>	CLINTON	Jessie Clinton	
<input type="checkbox"/>	DEFLT	DEFLT	
<input type="checkbox"/>	DEFLTREG	DEFLTREG	
<input type="checkbox"/>	MAXADMIN	MAXADMIN	
<input type="checkbox"/>	SYSADM	SYSADM	
<input type="checkbox"/>	MOTIKA	Scott Motika	
<input type="checkbox"/>	MURTHY	Joe Murthy	
<input type="checkbox"/>	WAYNE	John Wavne Jr.	
<input type="checkbox"/>	JON	Jon Standerson	
<input type="checkbox"/>	MARIA	Maria Totez	
<input type="checkbox"/>	JOSEFINA	Josefina Lezt	

Parameter Notes

Number of Parameter Values

1. The maximum number of User Inputted Report parameters that are enabled for reports is 23. These include 15 Non-Date Time Parameters, and 8 Date-Time parameters. If more than the 15 Non-Date and 8 Date-Time Parameters are entered, invalid bindings will display on the report's request page.

Utilizing Parameter Values on a Report's Request Page

2. Bound and Unbound parameters behave the same way when a user enters values on the Request Page. This means that if there is a parameter for Status, the following will occur:

User entered parameter value	Report Results
=APPR	Records where status = APPR
APPR	Records where status = WAPPR, APPR
%APPR	Records where status = WAPPR, APPR

Boolean Parameter Values

3. Reports that use boolean values as parameters must follow the guidelines below:

A. The parameter in the report design must be defined as a string type
This is required for localization purposes.

B. If the parameter value is required to be passed to a SQL statement, then the parameter value must be converted to integer value (1 or 0) as the database has 1 or 0
An API call has been added 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.getBooleanInteger(isActiveFlag));
```

or

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

Optional Parameters

4. 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). Priority is still mandatory:

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

YORN Lookup

A YORN lookup is available for Yes or No values. This lookup can be used in reports to eliminate the question of 'Do I enter Yes or Y or 1?' in a parameters value. For the Out of the Box Reports, the Security Group Access report (security_group.rptdesign) has been updated to include the new YORN lookup. This can be used as an example of how you can apply this lookup.

A condensed version of the reports.xml for this report is below to show how it's the YORN parameter is set. To find the complete version, access the file under

<V75>\reports\birt\reports\USER

```
<report name="security_group.rptdesign">
  <parameters>
    <parameter name="independent">
      <attribute name="attributename">INDEPENDENT</attribute>
      <attribute name="lookupname">yornlookuplist</attribute>
      <attribute name="sequence">2</attribute>
      <attribute name="labeloverride">Independent</attribute>
      <attribute name="defaultvalue">>false</attribute>
    </parameter>
  </parameters>
</report>
```

Viewing Parameters

6. If you drag parameters directly on to the report, you will receive the following errors in the Web Viewer, although report content will not be affected:

A report document error occurred when loading: Subquery

A report document error occurred when loading: Result Class

This happens because the bindings are created only at the cell level, not at the table level. To ensure the correct binding, insert Data elements and using the Expression Builder, set the values to the parameters (choose "Report Parameters" from the Category window).

Requirements for using lookups with Parameters

7. To enable a parameter lookup, the parameter must have an equivalent attribute. This makes the parameter bound as noted in the beginning portion of the parameter section.

Additionally, as noted above, unbound parameter values which have no attributes, cannot have a lookup. Domain lookups can only be used when bound to a field that has the domain assigned to it. The only two exceptions to this are the datelookup and yornlookuplist which can be associated to unbound parameter values.

Extending Ad Hoc Reports in BIRT Designer

To reduce report development time, you can utilize the Ad Hoc reporting functionality as an excellent starting point for your custom report development. The section below details how this can be enabled.

Use Ad Hoc Reporting as a base for Custom Report Development

When Ad Hoc Report is created and saved, its resulting design file (.rptdesign) that was created on the fly is saved to the database. This enables the report to be accessed in the future for immediate run access or scheduling and emailing.

Additionally, once the report is saved in the database, it can be extended within the Report Designer tool. By simply exporting the report and opening it in the design tool, you can quickly build upon the report design by adding calculations, graphs or additional features.

This can become a huge time saving feature for your custom report development because ad hoc reports can be created with complex sql from multiple tables, filters and application queries. Instead of having the developers perform the tasks of creating and testing the sql, laying out and defining the fields - you can let the V7.5 framework perform this work. The developer can then extend the ad hoc report as required in the designer.

To show how this can be accomplished, an example below is given. First, the Ad Hoc report is created and saved. In this case, the report is created in the Asset Application, and called 'Asset Specifications and Work Order Details.'

Tivoli software										IBM
Asset Specification and Work Order Details										
Asset Details										
Asset	Description	Location	Parent	Rotating Item	Site	Asset Tag	Type	Calendar		
11430	Centrifugal Pump 100GPM/60FT HD	BR430	11400	PUMP100	BEDFORD	6491				COMPANY1
Specifications										
Asset	ASSETSPECID	Unit of Base Measure	End Base Measure	End Measure	Start Base Measure	Start Measure				
11430	230									
11430	231									
11430	227									
11430	228									
11430	229									
11430	232									
Work Orders										
Work Order	Work Type	Status	Status Date	Scheduled Start	Scheduled Finish	Target Start	Target Finish			
7721	CP	CLOSE	7/31/96 10:06:00 PM							
1695	EM	CLOSE	8/5/98 1:43:24 AM	7/26/98 7:00:12 AM	7/29/98 1:29:00 AM	7/26/98 7:00:12 AM	7/29/98 1:29:00 AM			
3838	EM	CLOSE	1/17/01 2:20:24 AM	1/7/01 7:37:12 AM	1/10/01 2:06:00 AM	1/7/01 7:37:12 AM	1/10/01 2:06:00 AM			
6727	EM	CLOSE	3/22/99 2:51:24 AM	3/12/99 8:08:12 AM	3/15/99 2:37:00 AM	3/12/99 8:08:12 AM	3/14/99 2:37:00 AM			
1538	EM	CLOSE	4/24/01 1:40:24 AM	4/14/01 6:57:12 AM	4/17/01 1:26:00 AM	4/14/01 6:57:12 AM	4/17/01 1:26:00 AM			
1498	EM	CLOSE	10/8/01 1:37:24 AM	9/28/01 6:54:12 AM	10/1/01 1:25:00 AM	9/28/01 6:54:12 AM	10/1/01 1:25:00 AM			
1277	EM	CLOSE	3/5/01 1:08:24 AM	2/23/01 6:25:12 AM	2/26/01 12:54:00 AM	2/23/01 6:25:12 AM	2/26/01 12:54:00 AM			

Then, the report is exported for its repository in the database to a local file system. This is done via a command utility. The command utility uses a property file to enable this. Therefore, you must configure the reporttools.properties file. It is located at: <V7>\reports\birt\tools

```
# HostName or IP address of the machine that has MAXIMO application running in an App Server
maximo.report.birt.hostname=localhost
```

```
# HTTP port of the application server (the port used to access maximo from browser)
maximo.report.birt.port=9080
```

```
# Indicates whether the SSL communication is enabled or not
maximo.report.birt.ssl=false

# User that has access to perform the operation
maximo.report.birt.username=wilson

# Password of the user that has access to perform the operation
maximo.report.birt.password=wilson

# Output folder used for the export operation
maximo.report.birt.outputfolder=c:/7116/reports/birt/export/asset
```

The output folder highlighted in red details the location of where the exported Ad Hoc Report will be placed.

To export an Ad Hoc Report, open a command window. Navigate down to the exportcommands path.... <V7>reports\birt\tools

```
C:\
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>cd c:\

C:\>cd 7116\reports\birt\tools

C:\7116\reports\birt\tools>exportreports app asset
```

Then, export all of the reports from the asset application by:
exportreports app asset

```
C:\
[exportreports] Exporting report: [ASSET] asset_glaccount.rptdesign
[exportreports] Exporting report: [ASSET] asset_measurehist.rptdesign
[exportreports] Exporting report: [ASSET] asset_po.rptdesign
[exportreports] Exporting report: [ASSET] asset_subassembly.rptdesign
[exportreports] Exporting report: [ASSET] assetmove_history.rptdesign
[exportreports] Exporting report: [ASSET] detailasset_fail.rptdesign
[exportreports] Exporting report: [ASSET] drillasset_fail_thl.rptdesign
[exportreports] Exporting report: [ASSET] gaps_overlaps.rptdesign
[exportreports] Exporting report: [ASSET] linear_work_history.rptdesign
[exportreports] Exporting report: [ASSET] mgmt_sw.rptdesign
[exportreports] Exporting report: [ASSET] oee_kpi_by_asset.rptdesign
[exportreports] Exporting report: [ASSET] oee_kpi_by_location.rptdesign
[exportreports] Exporting report: [ASSET] oee_kpi_by_site.rptdesign
[exportreports] Exporting report: [ASSET] sumasset_fail.rptdesign
[exportreports] Exporting report: [ASSET] asset_wilson1264444754.rptdesign
[exportreports] Exporting report: [ASSET] asset_wilson1264447774.rptdesign

BUILD SUCCESSFUL
Total time: 30 seconds

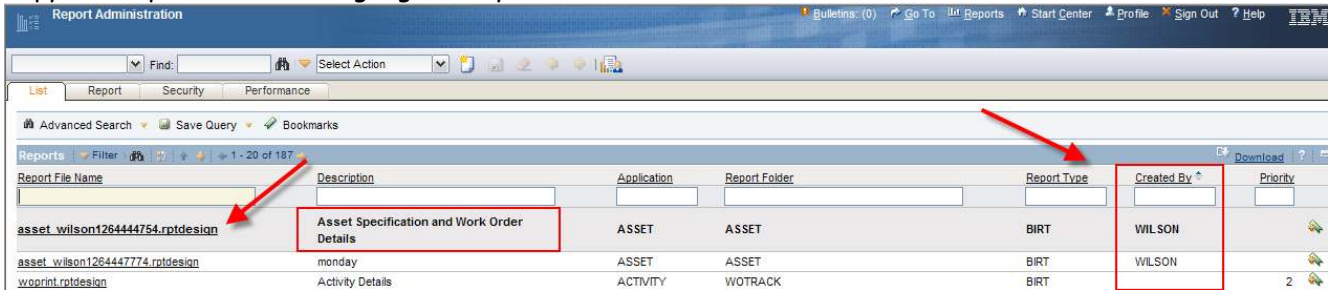
C:\7116\reports\birt\tools>
```

*Note: You can also export only a single report by using the command:

exportreport report <application> <reportfilename>

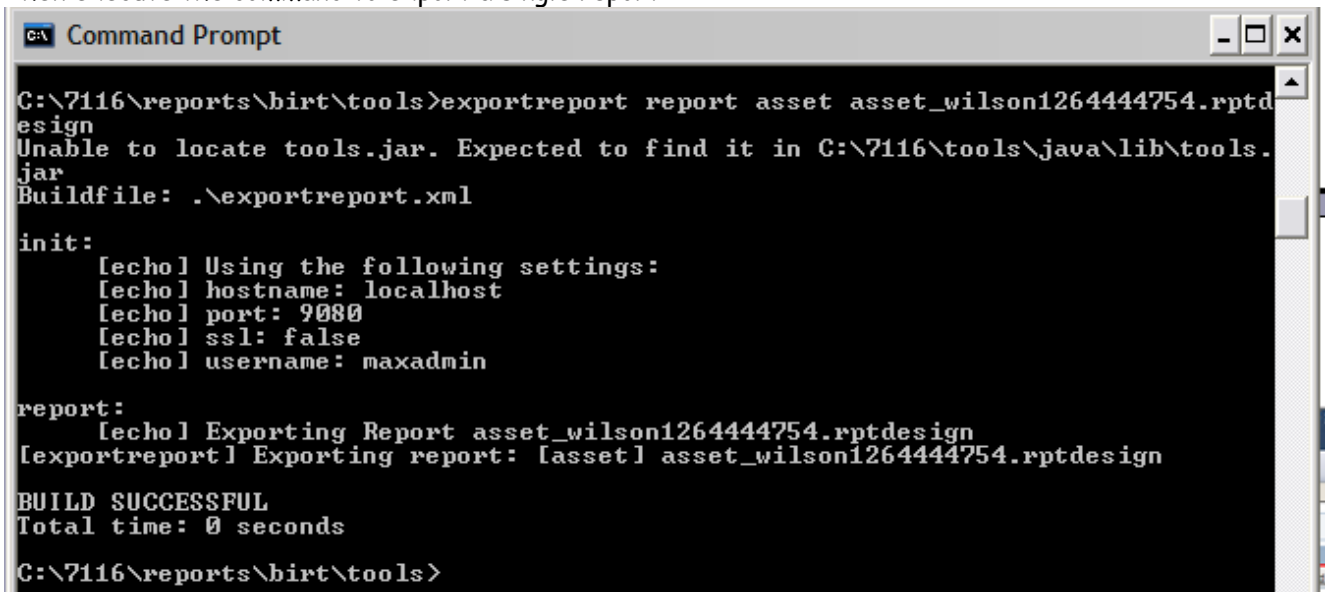
However, all command files work with file names - not the report descriptions that users assign during Ad Hoc Report Creation. So to use this command, you first must know the exact name of the QBR report. To find this name, access Report Admin, and filter on Created By. Ad Hoc reports are identified where the Created By field is not null.

Copy the report file name highlighted by the arrow on the left



Report File Name	Description	Application	Report Folder	Report Type	Created By	Priority
asset_wilson1264444754.rptdesign	Asset Specification and Work Order Details	ASSET	ASSET	BIRT	WILSON	
asset_wilson1264447774.rptdesign	monday	ASSET	ASSET	BIRT	WILSON	
woprint.rptdesign	Activity Details	ACTIVITY	WOTRACK	BIRT		2

Then execute the command to export a single report



```
C:\7116\reports\birt\tools>exportreport report asset asset_wilson1264444754.rptdesign
Unable to locate tools.jar. Expected to find it in C:\7116\tools\java\lib\tools.jar
Buildfile: .\exportreport.xml

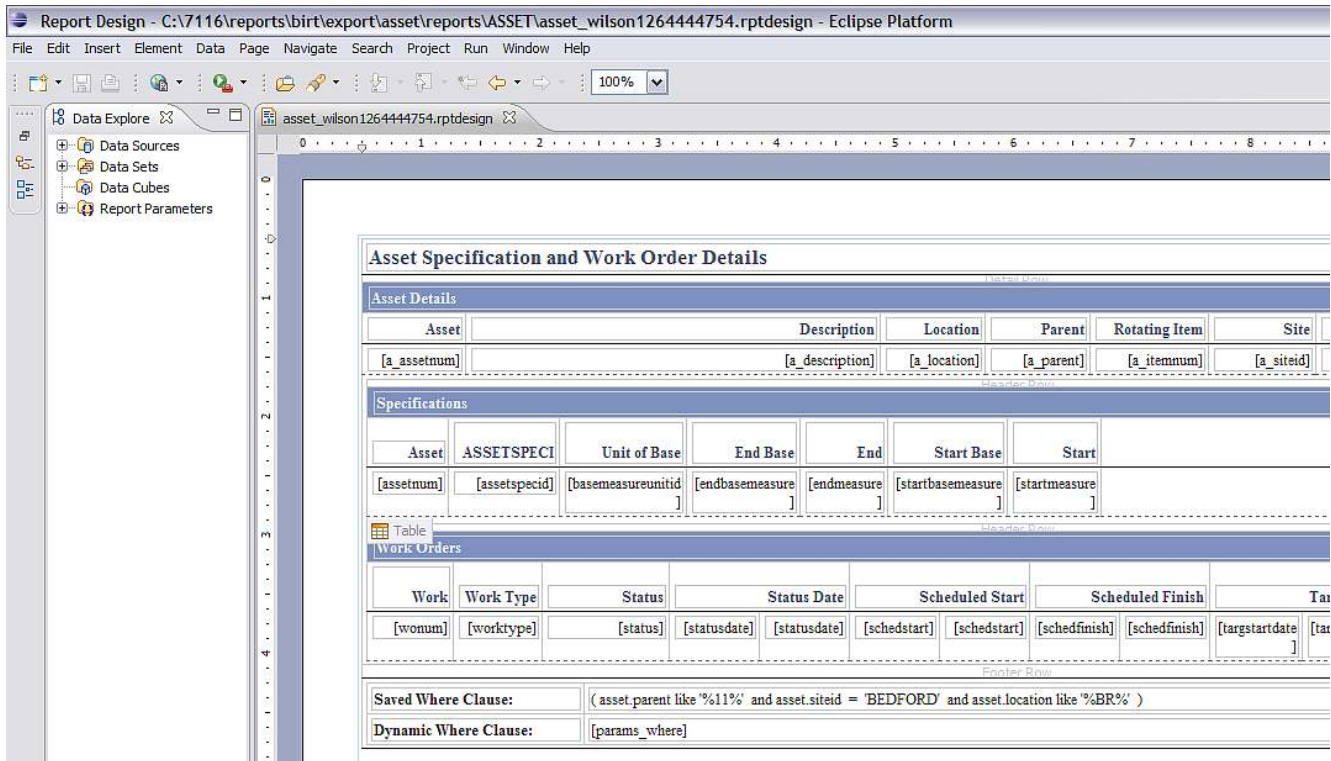
init:
[echo] Using the following settings:
[echo] hostname: localhost
[echo] port: 9080
[echo] ssl: false
[echo] username: maxadmin

report:
[echo] Exporting Report asset_wilson1264444754.rptdesign
[exportreport] Exporting report: [asset] asset_wilson1264444754.rptdesign

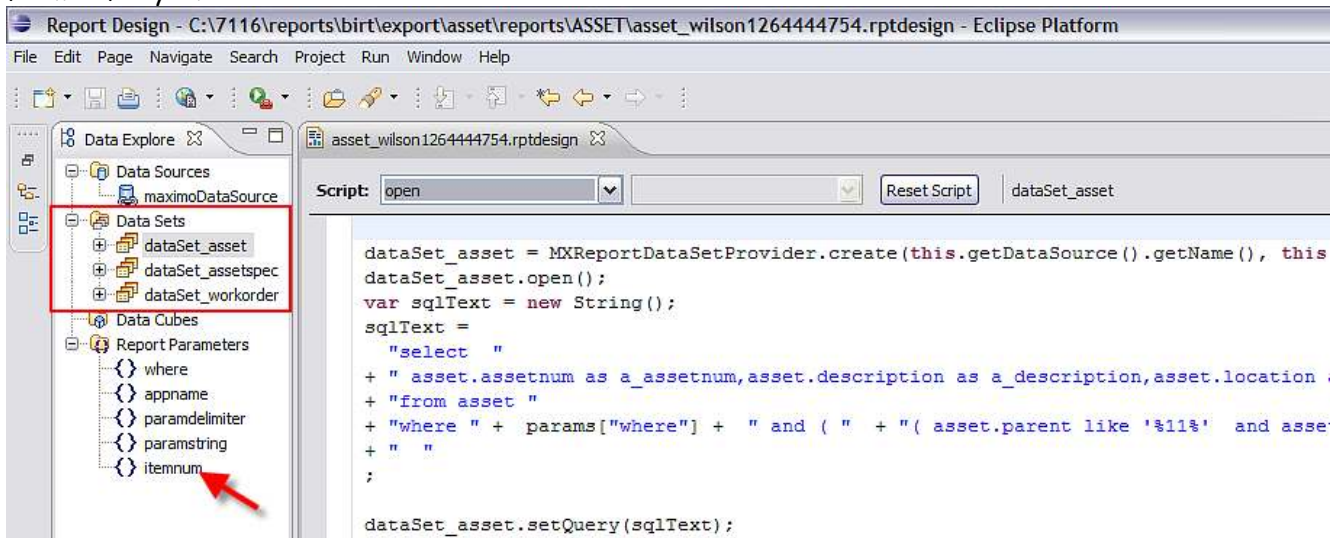
BUILD SUCCESSFUL
Total time: 0 seconds

C:\7116\reports\birt\tools>
```

Once you have exported the ad hoc report, open up the Report Designer tool. From the menu, click File - Open File and navigate to the directory where you exported your reports. Select the Ad Hoc Report's .rptdesign file, and it displays in the designer.



You can immediately see that you have an excellent beginning to extend this report further for any other customizations you may need. Multiple data sets (subreports) can be already populated, parameters included and complex sql statements including application queries can already be formed for you.



This can become an excellent starting point for your report developer.

Note: If the developer chooses to modify the design file, it is recommended that the report file name be modified to identify it from the original file. If the developer plans on utilizing this report as an Enterprise Report, he would need to create and/or append the reports.xml and properties file for the new enterprise report. Additionally, because it is now an enterprise report, it would need to be imported through the reports import command, or the UI utility in the Report Administration application.

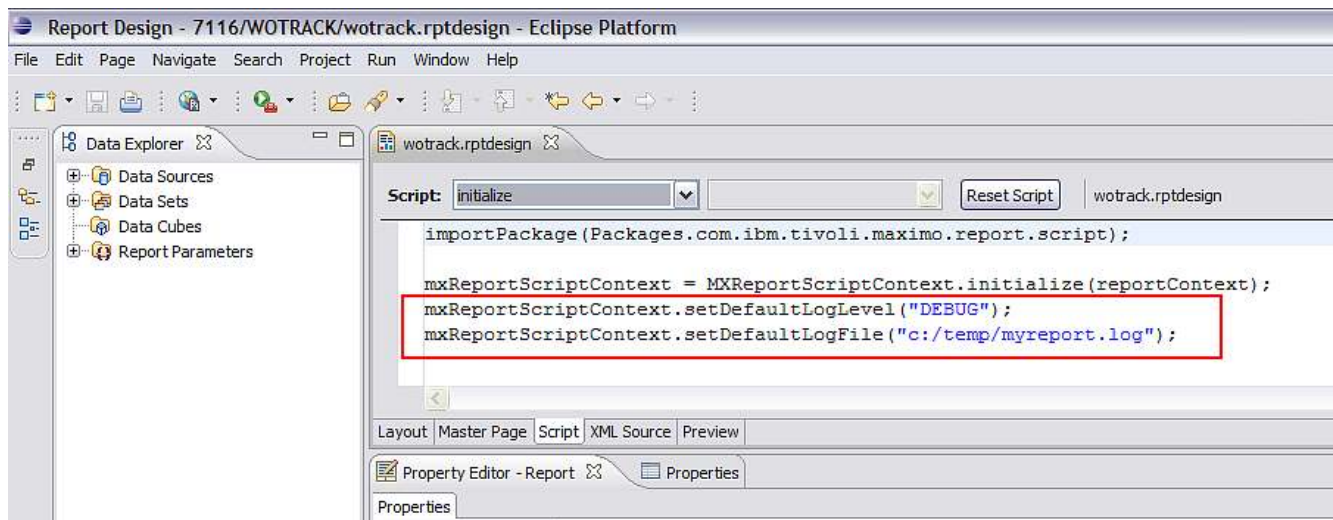
Debugging within the BIRT Report Design tool

You can log information about the report that you are developing within the BIRT Report Design Tool. This logging is used only when you preview a report within the Report Design Tool.

Reference: For more details on report logging, including how to enable it within the V7 application, reference the V7 Report Logging Guide noted at the end of this document.

To do this, access the Report initialize method, and add the following two lines of code:

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



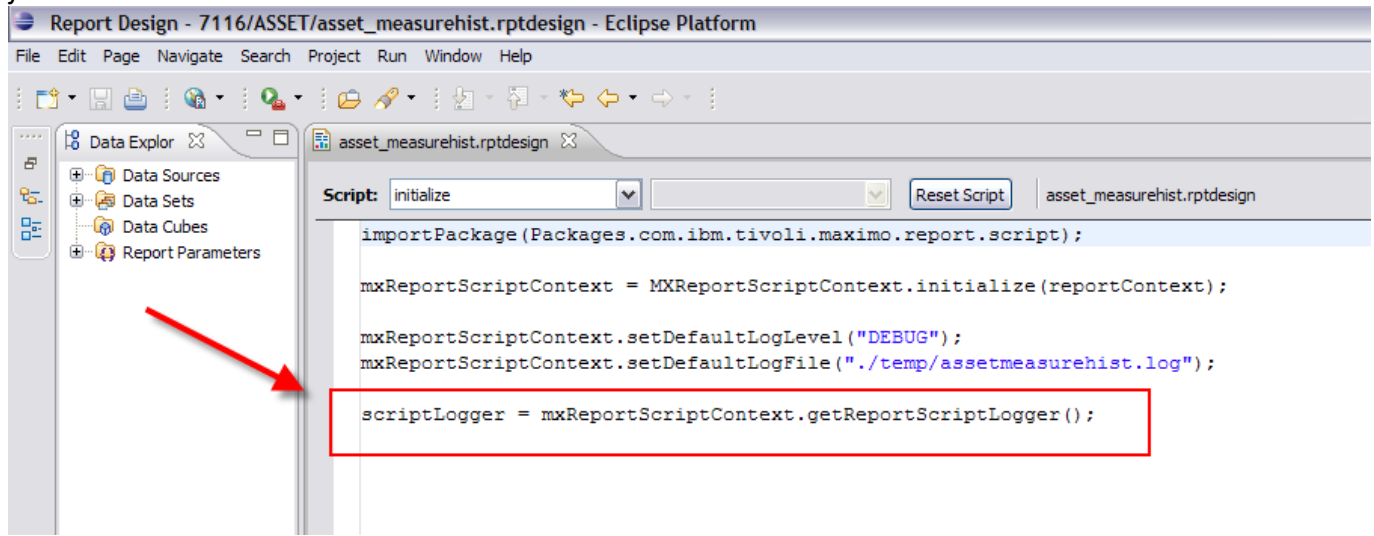
Five different log levels are supported, which are DEBUG, INFO, WARN, ERROR, FATAL. These levels are described in more detail below. Since this logging is primarily used for debugging report design issues, it is recommended that you use the DEBUG level.

Replace the file path location shown here as "c:/temp/myreport.log" with the file path for your individual environment.

Note: This logging is not used when executing a report from the Maximo applications. Once your report runs, you do not need to remove this logging.

Additionally, to log custom information, you can use the `mxReportScriptContext` variable to get a script logger, which can then be used throughout your report. You can add this to the report initialize method also as shown here.

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



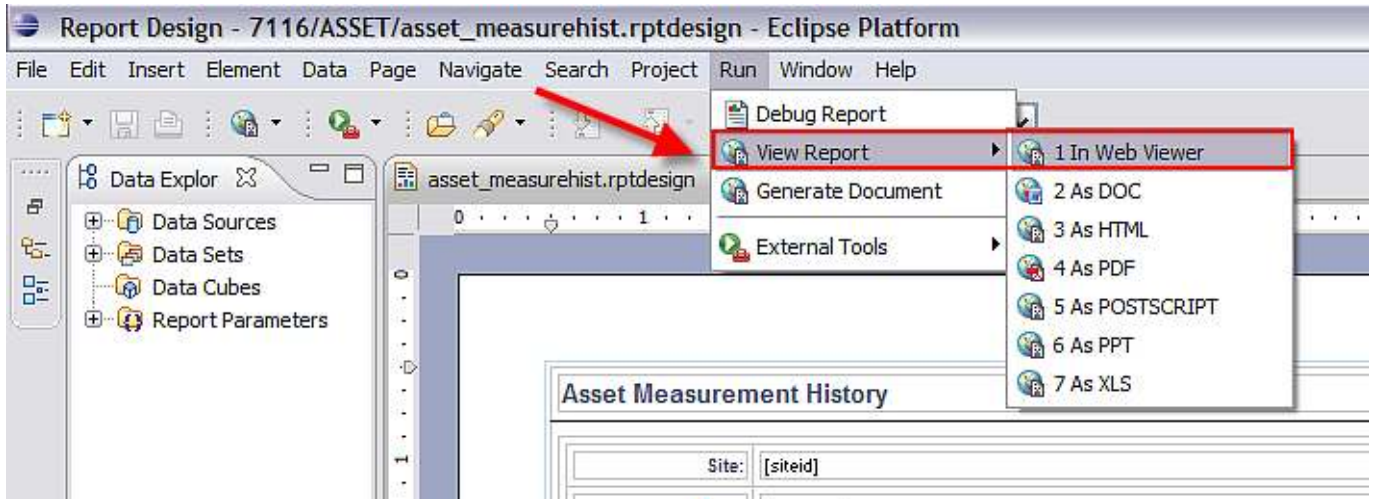
Unlike the default logging, these logging messages are written to the Maximo log files when the report is run from within Maximo. In this case, the default log level specified in the report is ignored. Instead, the `maximo.report.birt` log level from Maximo is used.

You can use any of the following methods below from `ReportLogger` to log information.

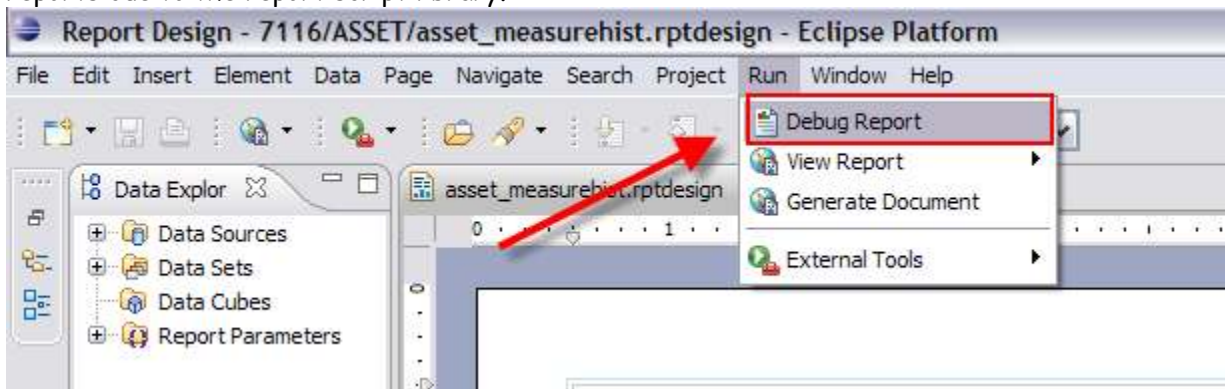
```
boolean isEnabled();  
boolean isErrorEnabled();  
boolean isFatalEnabled();  
boolean isInfoEnabled();  
boolean isWarnEnabled();  
  
void debug(Object message);  
void info(Object message);  
void warn(Object message);  
void error(Object message);  
void fatal(Object message);
```

Report Designer best practices for debugging

1. Preview reports by using the Web Viewer - View Report Section within the BIRT report designer. This displays the closest representation to report execution from within the various Maximo applications.



2. Within the BIRT 2.3.2 Report Designer, a 'Debug Report' Options is available. Do not use this functionality because it does not properly display information with the Maximo implementation of reports due to the report script library.



Note: More information on debugging within the V7 applications is noted in the Report Logging document referenced at the end of this document.

Miscellaneous Features

Database Update Functionality

You can 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. Examples of out of the box reports that use this functionality include:

Asset Cost Rollup (asset_costrollup_update.rptdesign, located under ASSET subfolder)

Inventory ABC (inventory_abc.rptdesign, located under INVENTOR subfolder)

Here is an example of how this can be used. (Note: all examples are illustrated with SQL UPDATE statement, but SQL INSERT and DELETE can be used similarly)

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

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

2. executing the update outside of a DataSet

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

3. executing 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. executing 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();
```

Registering a Report to Multiple Applications

Some reports can be accessed from multiple applications. To implement this in your custom environment, do not make copies of the design files in each application. Instead store the report in the primary application report folder, and register it to the other applications by including it in the reports.xml for each applicaiton. The detailed steps for this process are noted here.

1. Create the normal import entry in the home application's reports.xml.
2. Copy the entry to the reports.xml file for any other applicaiton that uses the report.
3. Change the <filename> entry to reflect the relative path to the actual report location, for example:
`<attribute name="filename">../PO/po_act.rptdesign</attribute>`
4. Change the other report administration values as appropriate.
5. If there are bound parameters, you may need to modify them since bindings that work in one application may not work in another.

Registering a Report with Quick Toolbar Access

Application reports can be enabled for Quick Toolbar Access. These are reports where no parameters are specified, and they execute against the application filter and/or query. The types of quick toolbar access are described and shown in the chart below.

BV: (Browser View) Enables the user to click on the BV icon in the application's toolbar, and the report opens immediately in the Report Browser.

DP: (Direct Print) Enables the user to click on the DP icon in the application's toolbar, and the report prints on the user's default printer. The report does not display in the report browser session.

DPA: (Direct Print with Attach Documents) Enables the user to click on the DPA icon in the application's toolbar, and the report and any of its printable attachments, print on the user's default printer. The report does not display in the report browser session.

	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 Documents	REPORT.PAD	REPORT.PADLOC	REPORT.TOOLBARSEQUENCE

An example of a report that has these settings defined can be found in the WOTRACK folder
<report name="wotrack.rptdesign">

```
<attribute name="filename">wotrack.rptdesign</attribute>
<attribute name="description">Work Order 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>

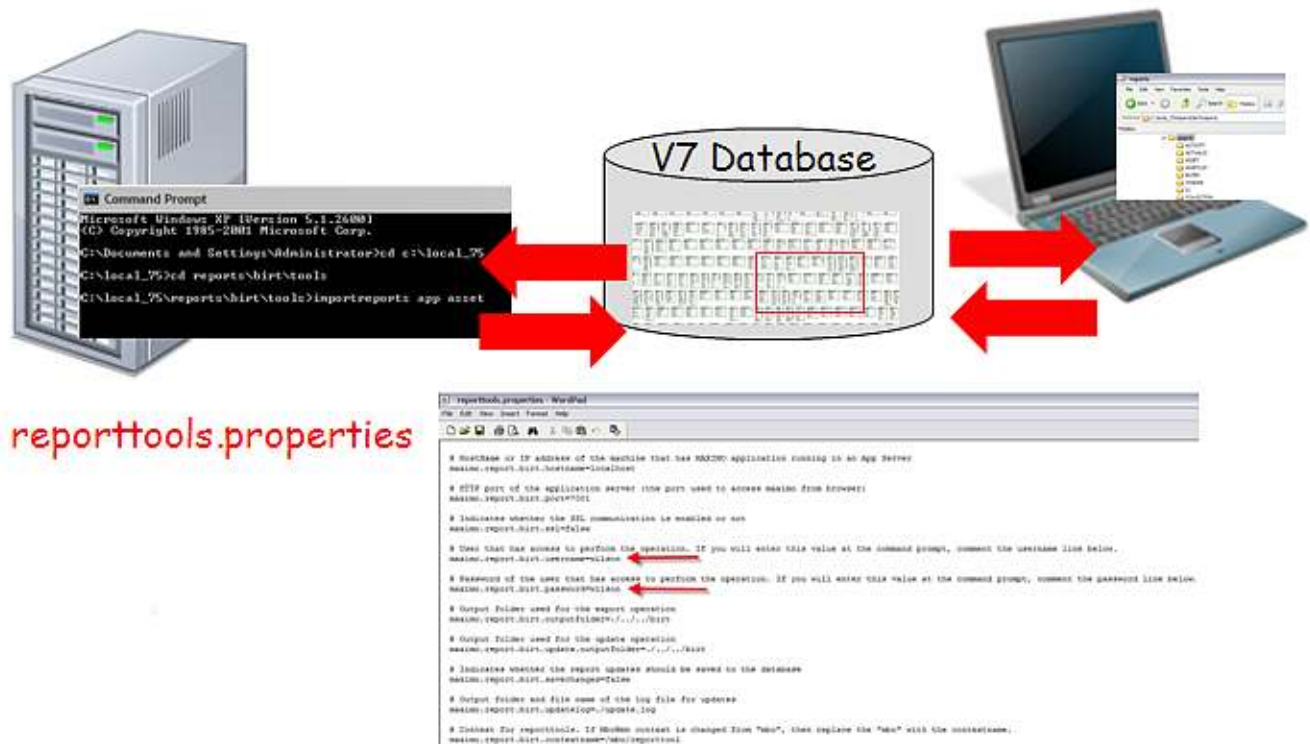
<resources>
```

Importing Report Designs into the V75 Database

As noted earlier, the repository for the report design files is the V75 database. These files are extracted at run time to meet the individual's report request.

Administrators and developers may need to import or export report design files from the V75 database repository. For example, they may need to export existing or ad hoc report design files from the database so the design files can be modified in the report design tool. Then, once the updates are made, the administrator would import the updated design file into the database.

Importing and exporting of the report design files can be done via command utilities. A key component of these utilities is a properties file called `reporttools.properties`. This property file contains information on the application server, the file directory where the reports designs are either coming from or going to, and username and password information on the user performing the operation. This process is shown in the diagram below.



Set Up: reporttools.properties

Before the importing or exporting processes can occur, the reporttools.properties file must be configured by following the steps below

Browse to the tool location ...<V75> \reports\birt\tools. Locate and open the reporttools.properties file shown below. Enter the standard values required for this file.

```
reporttools.properties - WordPad
File Edit View Insert Format Help
# HostName or IP address of the machine that has MAXIMO application running in an App Server
maximo.report.birt.hostname=localhost
# HTTP port of the application server (the port used to access maximo from browser)
maximo.report.birt.port=7001
# Indicates whether the SSL communication is enabled or not
maximo.report.birt.ssl=false
# User that has access to perform the operation. If you will enter this value at the command prompt, comment the username line below.
#maximo.report.birt.username=wilson
# Password of the user that has access to perform the operation. If you will enter this value at the command prompt, comment the password line below.
#maximo.report.birt.password=wilson
# Output folder used for the export operation
maximo.report.birt.outputfolder=../../birt
# Output folder used for the update operation
maximo.report.birt.update.outputfolder=../../birt
# Indicates whether the report updates should be saved to the database
maximo.report.birt.savechanges=false
# Output folder and file name of the log file for updates
maximo.report.birt.update.log=./update.log
# Context for reporttools. If MboWeb context is changed from "mbo", then replace the "mbo" with the contextname.
maximo.report.birt.contextname=/mbo/reporttool
```

Within this file, there are entries for the user who has privileges to import and export reports. This user is defined in the Security Group Application in Maximo per the privileges highlighted below.

The screenshot shows the Maximo Security Groups application interface. The 'Applications' tab is selected, and the 'Report Administration' table is displayed. The table has two columns: 'Description' and 'Main Object/Table'. The 'Report Administration' table is highlighted in blue. Below it, the 'Options for Report Administration' table is shown, with a red arrow pointing to the 'Import, Export' option. The 'Options for Report Administration' table has two columns: 'Description' and 'Grant Access'. The 'Options for Report Administration' table is highlighted in blue.

Description	Main Object/Table
report admin	
Report Administration	Base table for report app

Description	Grant Access
Import, Export	
Export Library	<input checked="" type="checkbox"/>
Export Report	<input checked="" type="checkbox"/>
Import Library File	<input checked="" type="checkbox"/>
Import Report	<input checked="" type="checkbox"/>

You have the option to both input the username and password into the property file, or not.

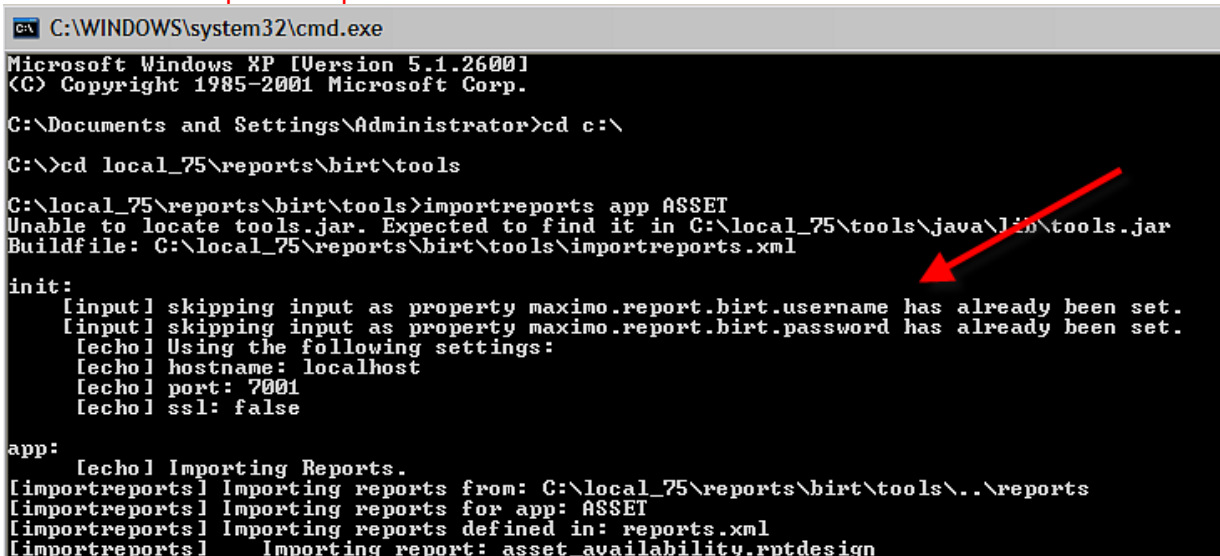
If you enter the username and password values in the reporttools.properties file, they will be used when the import or export utility is used. This is shown in the example below.

```
# User that has access to perform the operation
```

```
maximo.report.birt.username=wilson
```

```
# Password of the user that has access to perform the operation
```

```
maximo.report.birt.password=wilson
```



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>cd c:\

C:\>cd local_75\reports\birt\tools

C:\local_75\reports\birt\tools>importreports app ASSET
Unable to locate tools.jar. Expected to find it in C:\local_75\tools\java\lib\tools.jar
Buildfile: C:\local_75\reports\birt\tools\importreports.xml

init:
[input] skipping input as property maximo.report.birt.username has already been set.
[input] skipping input as property maximo.report.birt.password has already been set.
[echo] Using the following settings:
[echo] hostname: localhost
[echo] port: 7001
[echo] ssl: false

app:
[echo] Importing Reports.
[importreports] Importing reports from: C:\local_75\reports\birt\tools\..\reports
[importreports] Importing reports for app: ASSET
[importreports] Importing reports defined in: reports.xml
[importreports] Importing report: asset_availability.rptdesign
```

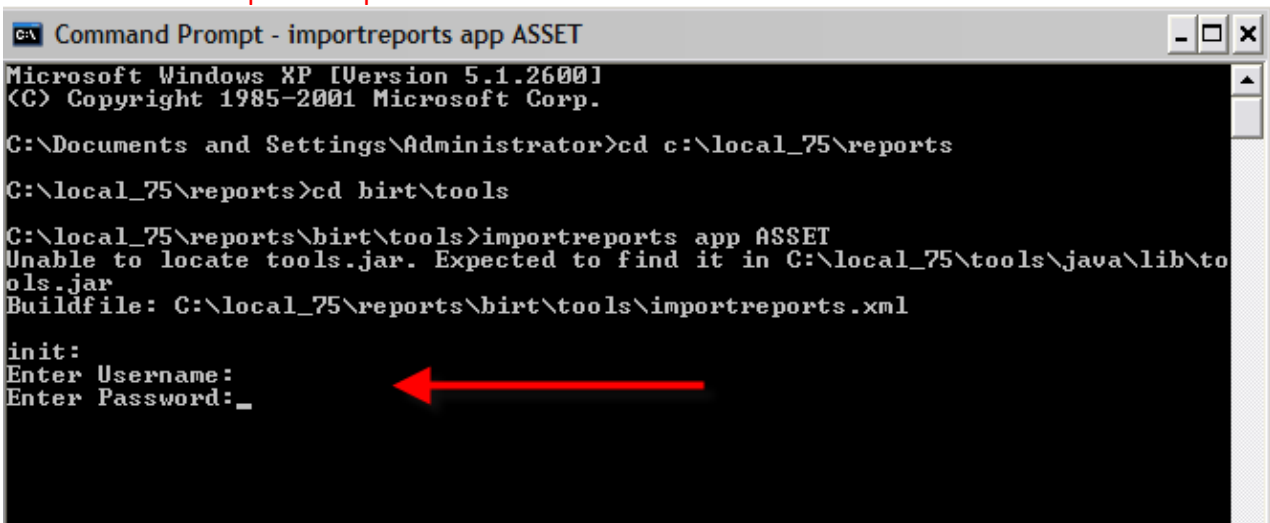
If you *do not* enter the username and password values in the reporttools.properties file, you will be prompted to enter them when you execute the import or export utilities per the example below.

```
# User that has access to perform the operation
```

```
#maximo.report.birt.username=abcabc
```

```
# Password of the user that has access to perform the operation
```

```
#maximo.report.birt.password=abcabc
```



```
Command Prompt - importreports app ASSET
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>cd c:\local_75\reports

C:\local_75\reports>cd birt\tools

C:\local_75\reports\birt\tools>importreports app ASSET
Unable to locate tools.jar. Expected to find it in C:\local_75\tools\java\lib\tools.jar
Buildfile: C:\local_75\reports\birt\tools\importreports.xml

init:
Enter Username:
Enter Password: _
```

Import Command Utility

Importing brings reports into the database. If the report design is new, a new record will be created in the database. If the report design exists, the import process will over-write the existing file. After the import is complete, the updated or new files will be located in the REPORTDESIGN table, which holds the design files, resource files and library files.

If you have a large number of reports to import, you may want to use the import command utility. The import process uses the reports.xml file to import the report design files in the database. A reports.xml file is available for each application that has reports, and is located in the directory <V75>reports\birt\reports.

The reports.xml file references each report design for the individual application. If a report design is not referenced in the reports.xml file, it will not be imported during the command utility process. Details on updating the reports.xml file for any new report files you may create can be found in the 'V75 Report Developer's Guide' referenced at the end of this document.

To use the import utility, follow the steps below.

1. On the V75 server, open a command prompt window and change to the folder <V75> \reports\birt\tools.

Then, run any of the following variations of the import utility

A. importreports

Imports all reports, libraries and resource files in the single import action.

B. importreports help

Displays details on the various import commands

C. importreports libraries

Imports all the libraries

D. importreports reports

Imports all the reports

E. importreports app [appname]

Imports all reports for a specified application.

This command is useful if you only want to import reports for a single application. For example: importreports app CONFIGUR will import all the reports in the CONFIGURATION application.

These are the reports found in the directory below:

```
<V75> \reports\birt\reports\CONFIGUR
```

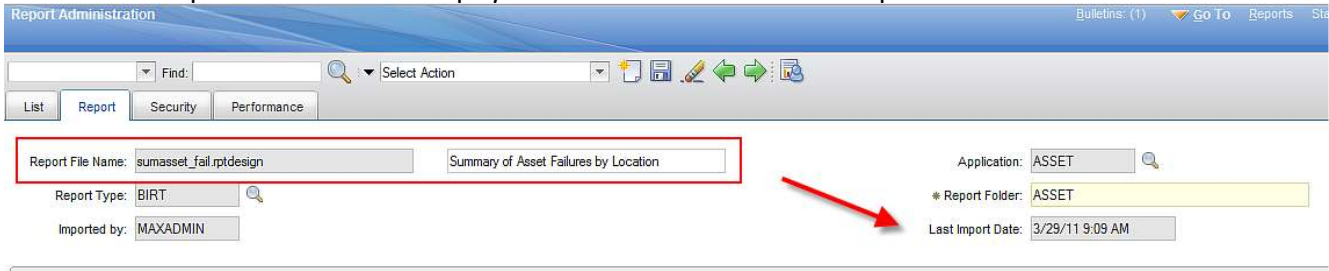
```

c:\ Command Prompt - imporeports
[importreports] Importing report: woprnt.rptdesign
[importreports] Importing reports for app: ACTUALCI
[importreports] Importing reports defined in: reports.xml
[importreports] Importing report: actualci_detail.rptdesign
[importreports] Importing report: actualci_history.rptdesign
[importreports] Importing reports for app: ASSET
[importreports] Importing reports defined in: reports.xml
[importreports] Importing report: asset_availability.rptdesign
[importreports] Importing report: detailasset_fail.rptdesign
[importreports] Importing report: drillasset_fail_tbl.rptdesign
[importreports] Importing report: sumasset_fail.rptdesign
[importreports] Importing report: asset_costrollup.rptdesign
[importreports] Importing report: asset_costrollup_update.rptdesign
[importreports] Importing report: assetmove_history.rptdesign
[importreports] Importing report: asset_glaccount.rptdesign
[importreports] Importing report: asset_po.rptdesign
[importreports] Importing report: mgmt_sw.rptdesign
[importreports] Importing report: asset_detail.rptdesign
[importreports] Importing report: asset_subassembly.rptdesign
[importreports] Importing report: asset.rptdesign
[importreports] Importing report: asset_measurehist.rptdesign
[importreports] Importing report: oekpi_by_site.rptdesign
[importreports] Importing report: oekpi_by_location.rptdesign
[importreports] Importing report: oekpi_by_asset.rptdesign

```

4. After the import is complete, sign into your V7.5 instance as the administrator. Go to the Report Administration application, and generate the XML for the reports. This completes the process for importing new or updated design files.

Additionally, you can view that the import occurred by looking at the 'Last Import Date' field for an individual report record. This displays the date/time of the last import.



Note: Details on importing a report thru the Report Administration application are provided in the V75 Report Feature guide referenced at the end of this document.

Export Command Utility

You may want to export report design files for a report developer to modify, or you may to extend an Ad Hoc or Query Based (QBR) report. For example, a user asks that an Ad Hoc report he created be extended to include a bar chart to the report. To save the developer time, the Ad Hoc report can be exported, and opened in the BIRT Designer. The developer can add the bar chart, and then import the design file back into the database as an enterprise report.

Exporting of reports is enabled as a utility only.

To enable exporting, go to the server, open a command prompt window and change to the folder <V75> \reports\birt\tools. Then, any of the commands below are available:

- A. `exportreports`
Exports all libraries and reports.
- B. `exportreports report`
Exports all reports.
- C. `exportreports library`
Exports all libraries.
- D. `exportreports app [appname]`
Exports all reports for the specified application.
- E. `exportreport report [appname] [reportfilename]`
Exports single, specified report for the specified application.

The report will be exported to the location defined by

- (1) the `reporttools.properties` file and
- (2) its report folder that it is registered to in the Report Administration application.

Export Example

The example below will use wotrack.rptdesign to show the exporting functionality. The reporttools.properties file has been set to use the output location shown below in red:

maximo.report.birt.outputfolder= **c:/V75/reports/birt/reports**

Additionally, the wotrack.rptdesign is located in the following applications and report folders:

Report Name	App Name	Report Folder	Description
woprint.rptdesign	QUICKREP	WOTRACK	Quick Reporting
woprint.rptdesign	CHANGE	WOTRACK	Change
woprint.rptdesign	RELEASE	WOTRACK	Release
woprint.rptdesign	ACTIVITY	WOTRACK	Activity
woprint.rptdesign	WOTRACK	WOTRACK	Work Order

When the various exports commands are executed, the following will occur:

1. exportreports

Exports all reports to c:/V75/reports/birt/reports and their various subfolders

AND Exports all libraries to c:/V75/reports/birt/libraries

2. exportreports report

Exports all reports to c:/V75/reports/birt/reports and their various subfolders

3. exportreports library

Export all libraries to c:/V75/reports/birt/libraries

4. exportreports app WOTRACK

Exports all reports registered to WOTRACK to c:/V75/reports/birt/reports/WOTRACK

5. exportreport report WOTRACK woprint.rptdesign

Exports woprint.rptdesign to c:/V75/reports/birt/reports/WOTRACK

Additional Export Details

- A. If a report structure is not available in the location where the export is to occur, a file structure will be created.

- B. If a reports.xml is not available in the location where the export is to occur, the reports.xml will be created.
 - This may occur if you create a new custom report design file, and register and import the report thru the Report Administration application.
 - If you do this and you make subsequent changes to the parameters or settings of the report in the Report Administration application, make sure to export the report design file so any changes you are made are captured in the new reports.xml file.

- C. If a reports.xml file does exist - the export will not overwrite the existing file.
 - In this case, a new one will be created using a -filename. Ex: In WOTRACK folder, if reports.xml exists and a new export occurs, a new reports-wotrack.xml file will be created.
This new reports-wotrack.xml will take precedence over the reports.xml file during any future importing actions.

- D. Both the import and export command utility tools use HTTP, not RMI, to support application server security. Only BASIC authentication is supported.

To enable the utilities for use with application server security, you must modify <V75>\applications\maximo\mboweb\webmodule\WEB-INF\web.xml. Open the file with a text editor and search for "AppServer security". Then, follow the instructions under the NOTE.

Understanding the reports.xml import file

One of the report developer's responsibilities is to create the reports.xml file. This file is required so the report design can be properly imported into the V7 database repository.

The reports.xml includes important information on the report, including its report design file, application, unique settings for the report (including direct print, sequencing or priority values) and parameter information.

If the reports.xml contains parameter values 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 defined in the reports.xml for a given report does not exist in the report, it is ignored. Some of the attribute values are defaulted 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:

	Symbol	Description
<	<	Less than
>	>	Greater than
&	&	Ampersand
'	'	Apostrophe
"	"	Quotation mark

As noted in the report design file structure section at the beginning of this guide, each application folder under reports\birt\reports will have an import file named reports.xml for the delivered, out of the box reports.

If you are modifying or creating your own custom reports, it is highly recommended that you create your own reports.xml file. This will insure that any updates you make are not over-written in future fix pack or release updates, and also enable you to quickly identify your custom reports. More details on this are in the section above titled 'Your Custom Reports and the Report File Structure'.

Preparing the reports.xml

If you are creating a reports.xml file for your modified or new custom report, you can use any text editor like word pad or notepad. The work to create the reports.xml file is not done in the report design tool.

You can create a reports.xml file by following the steps below.

1. Copy an existing reports.xml file either from the application your report will reside in - or one that is very similar to your new report requirements.
2. Rename the copied version to a unique identifier, which could include attaching your company name at the end of the file, like reports_abc.xml.
3. If you are modifying an existing report, modify the values for the new report file name and any other attribute changes.
4. If you are adding a new report, either enter new values or modify existing values for the new report content.
5. Delete all other references to design files that you have not modified. Save.

Notes:

- A. For an example of a report using a parameter, reference the Security Group Report (security_group.rptdesign) located in <V7.5> \reports\birt\reports\SECURITY
- B. For an example of reports using the application query and having various toolbar settings enabled, reference the Job Plan List and Detail reports located in <V75> \reports\birt\reports\JOBPLAN
- C. If you are unsure what Out of the box have which parameter values, access the V75 Report Booklet referenced at the end of this guide. This contains details on each report's parameters, along with various toolbar settings that are enabled for each report.
- D. Below please find more details each of the fields that can be used in defining a report in the reports.xml file. Each setting is not required, and those settings not required are noted in black text.

RED = Required Fields that must be used in defining any report
BLACK = Optional Fields.
BLUE = Text defining field value

<reports>

<report name="jobplan_test.rptdesign">

#Complete File Name of Report Design, including .rptdesign extension

<attribute name="filename">jobplan_test.rptdesign</attribute>

#File Name of Report Design, including .rptdesign extension

<attribute name="description">Job Plan Test</attribute>

#Description of Report Design which appears in 'Run Reports' Window

<attribute name="ql">0</attribute>

#Is Browser View enabled for Report? Can only be enabled if report does not have parameters. 0=No/1=Yes. Default is 0

<attribute name="dp">0</attribute>

#Is Direct Print enabled for Report? Can only be enabled if report does not have parameters. 0=No/1=Yes. Default is 0

<attribute name=" pad">0</attribute>

#Is Direct Print with Attachments enabled for Report? Can only be enabled if report does not have parameters. 0=No/1=Yes. Default is 0

<attribute name="toolbarsequence">1</attribute>

#Order of the report in relation to other reports enabled for toolbar access within the application. Value must be unique within a given application.

<attribute name="qlloc">NONE</attribute>

#Determines what tabs will display BV icon. Options are:

#ALL: Displays Report Icon on all toolbars in the app

#LIST: Only Displays Report Icon on List Tab of app

#MAIN: Displays Report Icon on all toolbars in app, except List tab

#NONE: Default Value. Does not display Report Icon in app.

<attribute name="dploc">NONE</attribute>

#Determines what tabs will display DP icon. Options are:

#ALL: Displays Report Icon on all toolbars in the app

#LIST: Only Displays Report Icon on List Tab of app

#MAIN: Displays Report Icon on all toolbars in app, except List tab

#NONE: Default Value. Does not display Report Icon in app.

<attribute name="padloc">NONE</attribute>

#Determines what tabs will display DPA icon. Options are #ALL: Displays Report Icon on all toolbars in the app

#LIST: Only Displays Report Icon on List Tab of app

#MAIN: Displays Report Icon on all toolbars in app, except List tab

#NONE: Default Value. Does not display Report Icon in app.

<attribute name="norequestpage">0</attribute>

#Does the report not require a request page? 0=No/1=Yes. Default is 0 - Report does require a request page. Used only for reports which update database or are only available via hyperlinks.

<attribute name="detail">0</attribute>

#Are limit records enabled for this report? Can only be enabled if report does not have parameters. 0=No/1=Yes. Default is 0.

<attribute name="recordlimit">50</attribute>

#If limit records are enabled (detail = 1), this field must be defined. It is the maximum # of records the report can execute against. Value must be > 0.

<attribute name="priority">2</attribute>

#Priority value of report used for Report Queuing. Priority is based on ascending order - the lower the #, the higher the priority.

<attribute name="usewherewithparam">0</attribute>

#Will the report execute against both current/selected records and user inputted parameters? 0=No/1=Yes. Default is 0. Can only be enabled if report has parameters.

<attribute name="scheduleonly">0</attribute>

#Can the report only be executed via a schedule job request? This means it cannot be executed immediately. 0=No/1=Yes. Default is 0.

<attribute name="displayorder">1</attribute>

#Order that the report should display in relation to the other reports registered to the application.

<attribute name="reportfolder">JOBPLAN</attribute>

#Location of report source file subfolder in <Version7>\reports\birt\reports

<parameters>

<parameter name="jpnun">

#Name of parameter. If the parameter is unbound, this text must exactly match the unbound parameter defined in the BIRT Designer (.rptdesign file)

<attribute name="attributename">JPNUM</attribute>

Either the attribute name from the main table of the app, or the attribute from one of the app's Maxrelationships. If this field is populated, the parameter is bound. If the field is not populated, the parameter is unbound.

<attribute name="lookupname"></attribute>

#Name of lookup. Depending on availability, a bound parameter may or may not have a lookup. Unbound parameters can only have lookups for date fields.

<attribute name="sequence">1</attribute>

#Order the parameter is displayed on the request page.

<attribute name="labeloverride">Job Plan</attribute>

#Parameter label text that displays on Request Page.

```
<attribute name="defaultvalue"></attribute>
```

#Default value displayed in parameter field on request pages. Default values are not localized.

```
<attribute name="required">0</attribute>
```

#Is the parameter required? 0=No/1=Yes. Default is 0.

```
<attribute name="operator"></attribute>
```

#Optional operators that can be applied to bound parameters. Values available are >, >=, <, <=. These can not be applied to unbound parameters.

```
<attribute name="multilookup">0</attribute>
```

#Can multiple values be input for a parameter? 0=No/1=Yes. Default is 0.

```
</parameter>
```

```
</parameters>
```

```
<resources>
```

```
<resource>
```

```
<reference>joplan_abc.properties</reference>
```

The property file used by this report

```
<filename>${libraryfolder}/jobplan-abc.properties</filename>
```

#The location of the property file. \${libraryfolder} refers to

```
<Version75>\reports\birt\libraries
```

```
</resource>
```

```
</resources>
```

```
</report>
```

```
</reports>
```

Miscellaneous Utilities

Update Reports Utility

Additional update utilities can be used to automate the process of applying updates to report designs, rather than manually editing each report. These are known as update utilities, and supplement the existing utilities of importing and exporting report designs. The update utilities are available for both Enterprise Reports, and Ad Hoc or QBR Reports.

For enterprise reports, the four update utilities available are:

1. `updatereports`

Updates all reports.

2. `updatereports savechanges`

Updates reports and saves the modified reports to the database.

3. `updatereports app [appname]`

Updates all reports for the specified app.

4. `updatereports app [appname] savechanges`

Updates all reports for the specified app and saves the modified reports to the database

For Ad Hoc reports, the update utility available is:

1. `updateqbrs`

Updates all QBR report designs

Specific details on how you can use these utilities can be found in the Update Reports Utility document located on IBM's support website. Information on locating this can be found in the Reference Materials section at the end of this document.

Customizing Reports Reference Materials

A number of different reporting reference materials are available to you. This section will highlight a few of those reference materials which specifically focus on customizing reports, and details how you might best be able to use them.

Changing Report Logos

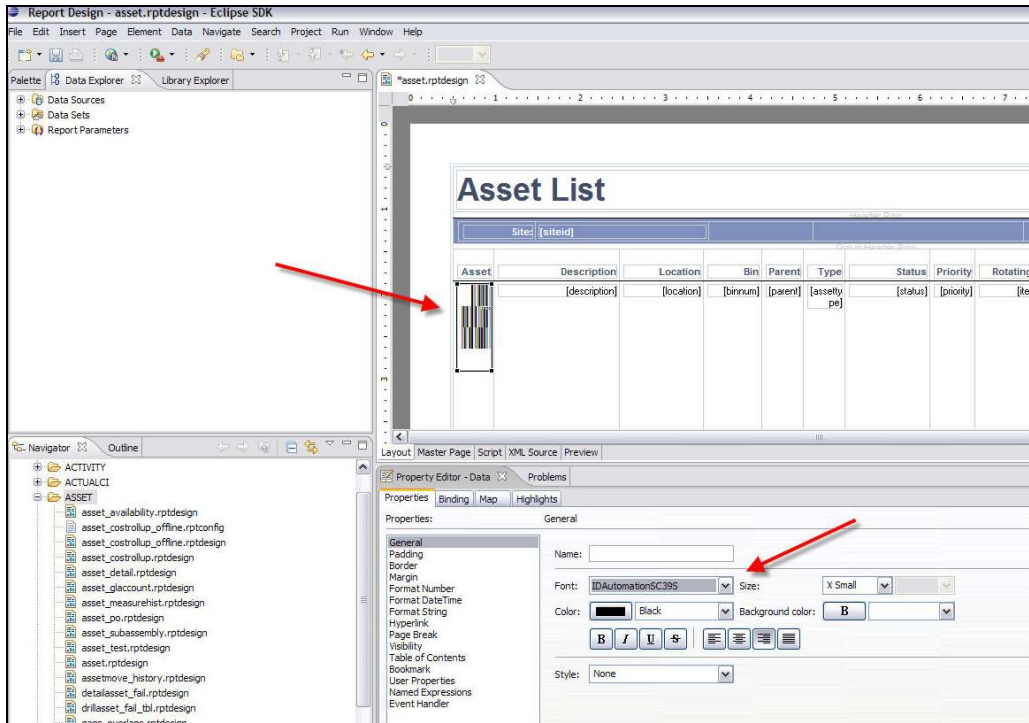
V7.5 reports contain two logos. A Tivoli logo is located on the upper left hand side, and an IBM Logo on the upper right hand side. You may want to update the V7.5 Reports to use their own corporate logos.

Asset	Description	Location	Bin	Parent	Type	Status	Priority	Rotating Item	Linear Asset?	Install Date	Mfr
1944	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1945	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1946	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1947	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1948	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1949	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON

Implementing Bar Code Fonts

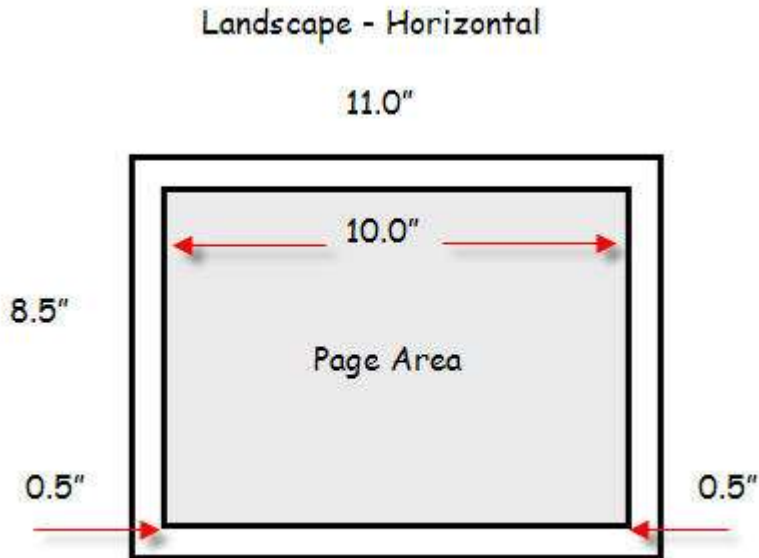
You may need to implement bar code fonts in your V7.5 reports. This document details the steps required to do this, including:

1. Enabling Bar Code fonts on the client machine where the report designer is installed
2. Enabling Bar Code fonts on the V.5 Server where the report will be executed.



Understanding Report Paper Sizes

This document reviews the components impacting report page sizes and orientation used in the V7.5 reports. It also details how you can customize them to meet your individual business needs.



Modifying OOB Reports

The out-of-the box reports that are supplied to you may not meet your individual business needs. You may need to add or remove fields to these reports to reflect your unique environment.

This document details how you can modify the out of the box. It uses the Work Order Details Report as an example. Three examples of modifications are detailed, including

- A. Deleting Fields from the Planned Labor Section
- B. Deleting Fields from the Actual Labor Section
- C. Adding Fields to the Actual Material Section



```
1 plannedLaborDataSet = MXReportDataSetProvider.create(this.getDataSource().getName(), this.getName());
2 plannedLaborDataSet.open();
3
4 var sqlText = new String();
5
6 // Add query to sqlText variable.
7 sqlText = "select wplabor.laborcode,wplabor.craft, wplabor.skilllevel, wplabor.vendor, wplabor.contractnum, "
8 + "wplabor.quantity, wplabor.orgid, wplabor.laborhrs, wplabor.rate, "
9 + "(wplabor.quantity * wplabor.laborhrs * wplabor.rate) as linecost, "
10 + "workorder.istask, workorder.taskid, wplabor.ratehaschanged "
11 + "from wplabor, workorder "
12 + "where workorder.wonum = wplabor.wonum "
13 + "and ((workorder.parent = " + rows[0]["wonum"].replace('/','g','') + " and workorder.taskid is not null) "
14 + "or (workorder.wonum = " + rows[0]["wonum"].replace('/','g','') + " and workorder.taskid is null)) "
15 + "and wplabor.siteid = workorder.siteid "
16 + "and wplabor.siteid = " + rows[0]["siteid"] + ""
17 + "and wplabor.orgid = " + rows[0]["orgid"] + ""
18 + "order by workorder.taskid "
19 ;
20
21 plannedLaborDataSet.setQuery(sqlText);
```

Utilizing the V7.5 Report Booklet

The V7.5 Report booklet details all the reports delivered in the base Maximo Services release. Within the booklet, is an .xls file which lists each report, and other important information on the application it is registered to, if it has any parameters, along with information on its graphs, sorting, grouping and what templates it is used.

If you are required to create a custom report, you may want to review the V7.5 Report Booklet to quickly find out-of-the-box (OOB) reports which have similar functionality. For example, if you wanted to review the code of a report that contains a pie chart, search within the booklet to quickly find a listing of reports with pie charts.

	Name	Description	LD?	Report File Name	App(s)	Graph	Enabled?	Parameters
4					Maximo			Configurable BV, DP, DPA, Limits
12	8	Asset Purchase Order Details			Asset			DP, Seq=2, Record Limit = 50
13	9	Assets by Subassembly Item			Asset		No	Item*
14	10	Summary of Asset Failures by Location	Yes	sumasset_fail.rptdesign	Asset		No	Location*, Start Date*, Date*
15	11	Details of Asset Failures by Location		detailasset_fail.rptdesign	Asset		No	Site*, Location*, Start Date*, End Date*, Asset Type, Status, Has moved?
		Presents historical view of downtime hours for selected						Problem Code, Location*, A Site, Start D

Additionally, the booklet contains additional information on some of the OOB reports which could include unique code in how it was prepared, or additional details on the data it analyzes. These are identified by having a 'Yes' in the column 'Additional Report Description Details' of the booklet.

Name	Description	Additional Report Description Details?	Report File Name	Maximo App(s)
5 Asset Measurement History	Using Yellow and Red Control Limit Values, displays Meter Readings by Date in Line Chart Format. A page break separates the details, which are displayed in ascending order of Measurement Date. This enables review of the most recent measurements first. Can be executed against single or multiple assets.	Yes	asset_measurehist.rptdesign	Asset
6 Asset Move History	For a single asset, displays all its asset move transactions. The transactions are displayed in descending order of Date Moved to enable the user to focus on the most recent moves first. This report does not display any non-required parameter values when they are not entered by the user.	Yes	assetmove_history.rptdesign	Asset

You can view this information by accessing the 'Additional V75 Report Desc' worksheet, where the long description details of the report can be viewed.

V75 Reports / **Additional V75 Report Desc** / Base 75 IS Reports / Report 75 Updates

3 <i>Asset Move History Report</i>	The Asset Move History Report is an example of report that does not display non-required parameter values when users do not enter them. This is done because (1) if the user did not enter a parameter value, it is not important to them - so they do not want to see a blank space on the report's header section for a non-filled in value (2) displaying empty parameter values in the report's header section takes up valuable real estate. If you only show what the user entered, they can immediately focus their eyes on these important attributes, and not be distracted by empty parameter values.
------------------------------------	---

Additional References

For a complete listing of the latest report documentation for the V7.1 and V7.5 releases, access the url below.

<http://www.ibm.com/developerworks/wikis/display/maximo/Report+Reference+Materials>

or it's shortened url of

<http://ibm.co/pxUyp6>

This page categorizes the documentation into detail, planning, development, customization and integration guides. Additionally, an overview of the guide is provided, along with its current revision level. All documentation referenced is available on IBM's support site via the url referenced within the page.

Report Reference Materials

[View](#) [Info](#) [Browse Space](#)

Added by [PamDenny](#), last edited by [PamDenny](#) on Sep 19, 2011 ([view change](#))
Labels: (None)

IBM Maximo Asset Management

[Home](#) > [Reports](#) > Report User Documentation

Report User Documentation

The tables below detail the report documentation currently available. This documentation is categorized into detail, planning, development, customization and integration guides. Additionally, an overview of the guide is provided, along with its current revision level. All documentation referenced is available on IBM's support site at the url below by searching on the document reference number or title.

http://www-947.ibm.com/support/entry/portal/Overview/Software/Tivoli/Maximo_Asset_Management

- 1 [Report Detail Guides](#)
- 2 [Report Planning and Upgrade Guides](#)
- 3 [Report Development Guides](#)
- 4 [Report Customization Guides](#)
- 5 [Report Integration Guides](#)

Report Detail Guides

Document Name	Description	Version	Reference Number	Revision #	Last Posting Date
V75 Report Booklet	Contains listings, file names, descriptions, details on parameters, formatting (grouping, sorting) and a pdf copy of each of the OOB (Out of the Box) Delivered Reports.	7.5	1497942		4/28/2011
V71 Report Booklet	Contains listings, file names, descriptions, details on parameters, formatting (grouping, sorting) and a pdf copy of each of the OOB (Out of the Box) Delivered Reports.	7.1	1305005	4	12/28/2009

Additional information on reporting, including Bilogs (Business Intelligence Blogs), User Forums and links to other wiki pages can be found on IBM's Service Management Connect at this url:

<https://www.ibm.com/developerworks/servicemanagement/am/index.html>

or it's shortened url of

<http://ibm.co/krPiz9>

Revision History

June 2012 - Revision 4

(1) Updated sample values for Oracle database in mxreportdatasources.properties file on page 12

May 2012 - Revision 3

(1) Added more screenshots in hyperlink area (2) Updated hyperlink for Report Designer Download on page 8 (3) Updated report reference material section

January 2012 - Revision 2

(1) Updated BIRT Designer shortcut path on page 15, step 4B for consistency (2) Updated report.xml example (3) Added get(date) data set mapping on page 43

October 2011 - Revision 1

(1) Updated reports xml for schedule only and display order (2) Updated url and folder location for 7.5 Java docs on page 33 (3) Added url to download portrait report templates on page 34

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.*

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.

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.

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.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “[Copyright and trademark information](#)” at www.ibm.com/legal/copytrade.shtml.

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