Crystal Reports XI

Java User-Defined Function Libraries (UFL)

Overview

This White Paper will explain how to create Java User-Defined Function Libraries (UFLs) with Crystal Reports XI. Custom Java UFLs can be constructed and used in Crystal Reports XI and reports deployed with the Java Reporting Component XI.

Contents

Introduction

One of the new features of Crystal Reports (CR) XI is the ability to create custom Java User Function Libraries (UFLs) that can be used in reports. UFLs provide the user with the ability to extend the CR function libraries by creating new functions that are not currently part of CR. These new functions can be used to provide more power and flexibility during report design.

In CR 10 and earlier, it was possible to create custom UFLs in COM-based languages such as Visual Basic or C++. In CR XI, it is now possible to create cross platform Java-based UFLs that can be used in CR and the Java Reporting Component (JRC).

Java UFLs can only be deployed with the JRC XI. Java UFLs cannot be processed in either Crystal Reports Server XI or BusinessObjects Enterprise XI. To create custom UFLs for these platforms, refer to the whitepaper on creating COM UFLs.

NOTE	When using Crystal Reports to design reports that contain user-defined function libraries (UFL), you can use only Java UFLs or only COM UFLs, but not both.

Creating Java User-Defined Function Libraries

In this section, we will go over the required steps to create a Java UFL. UFLs consist of two parts, the function and the library that contains the function. In Java, this translates into a class for each function that implements the FormulaFunction interface, and a class for the library that implements the FormulaFunctionLibrary interface. The FormulaFunctionLibrary interface contains or encapsulates all of the functions together.

When creating Java UFLs, the UFLs must follow a specific structure as outlined below. The JRC SDK contains the classes and interfaces that are used to create Java user-defined functions and libraries.

Sample User-Defined Function

The first step when creating a custom Java UFL, is to create the custom functions that will be contained in the UFL. The following code will be written outside of Crystal Reports in a development environment. It will not work in Crystal Reports until Crystal Reports has been configured to use Java UFLs.

To configure Crystal Reports to use Java UFLs refer to this section.

```
//File: MyFunction1.java
import com.crystaldecisions.reports.formulas.*;
import com.crystaldecisions.reports.common.value.*;
```

```
/**
 * @author Business Objects Developer Support
 * This sample demonstrates how to create a simple Java User-
 * Defined Function Library for Crystal Reports.
 */
public class MyFunction1 implements FormulaFunction {

        FormulaFunctionArgumentDefinition[] myFunc1Arguments =
{SimpleFormulaFunctionArgumentDefinition.string};

        public String getIdentifier() {

                return "myfunction1";

        }

        public FormulaValueType getReturnType() {

                return FormulaValueType.string;
        }

        public FormulaFunctionArgumentDefinition[] getArguments() {

                return myFunc1Arguments;

        }

        public void validateArgumentValues(FormulaValueReference[]
arguments) throws FormulaFunctionCallException { }

        public FormulaValue evaluate (FormulaValueReference[]
arguments) throws FormulaFunctionCallException {

                String strArg0 =
((StringValue)arguments[0].getFormulaValue()).getString();
                String string1 = "Input entered: " + strArg0;
                FormulaValue formulaVal =
StringValue.fromString(string1);

                return formulaVal;

        }

}
```

Sample User-Defined Function Library

Once the functions that will be contained in the UFL have been created, the next step is to create the library itself and define the functions that it contains. The sample library below contains only MyFunction1, created previously; however, UFLs can contain multiple custom functions.

```
//File: MyLibrary.java
import com.crystaldecisions.reports.formulas.*;

/**
 * @author Business Objects Developer Support
 * This sample demonstrates how to create the library that will
 * contain the set of custom functions.
 */
public class MyLibrary implements FormulaFunctionLibrary {
        private FormulaFunction[] functionArray = {new
MyFunction1()}; //Defines the list of functions that are contained
in this library.

        public FormulaFunction getFunction(int functionNumber) {
                return functionArray[functionNumber];
        }

        public int size() {
                return functionArray.length;
        }

}
```

Compiling the UFL

With the function and library classes built, the UFL should now be compiled for use with CR and the JRC. When compiling the UFL classes, ensure that the **CrystalFormulas.jar** and **CrystalReportingCommon.jar** have been added to the **CLASSPATH**. These JAR files can be found in the following folder:

C:\Program Files\Common Files\Business Objects\3.0\java\lib folder

Use a Java UFL in Crystal Reports

To use a Java UFL in an application you need to configure Crystal Reports to use Java UFLs, configure your computer to use Java user-defined function libraries with Crystal Reports then use the UFL in a report.

Enable Java user-defined function libraries in Crystal Reports

1. In Crystal Reports, on the **File** menu click **Options**.
2. Click the **Formula Editor** tab then, in the **UFL Support** drop-down box, click **Java UFLs Only**.

Configure your Computer to use Java User-Defined Function Libraries with Crystal Reports

1. With C R not running, browse to the following folder:

   C:\Program Files\Common Files\Business Objects\3.0\java\lib
2. Copy the following JAR files from this folder to your **CLASSPATH**:
   - CrystalFormulas.jar
   - CrystalReportingCommon.jar
   - u211java.jar
3. Browse to the following folder:

C:\Program Files\Common Files\Business Objects\3.0\java\lib\external

4. Copy the following JAR files from this folder to your **CLASSPATH:**

   - log4j.jar
   - icu4j.jar

5. Add the custom Java UFL classes to the **CLASSPATH** system variable.

CAUTION	Do not embed existing environment variables in the CLASSPATH as this environment variable is read directly without substituting the values of the embedded environment variables.

6. Set your JAVA_HOME system or user variable to the location of your J2SDK. For example:

C:\Program Files\Business Objects\JavaSDK\

7. On the Start menu, click Run, type “regedit”, then click ‘OK’.

8. Browse to the following registry sub key:

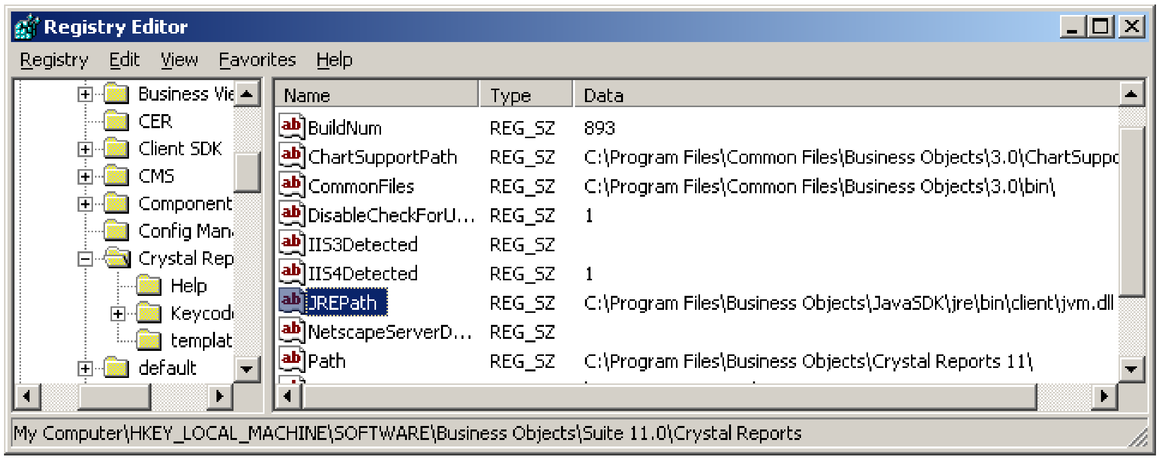
```
HKEY_LOCAL_MACHINE\SOFTWARE\Business Objects\Suite
11.0\Crystal Reports\
```

9. Right-click the Crystal Reports sub key, click New > String Value and name it JREPath.

10. Right-click the JREPath string value and click Modify. In the Value Data text box type the location of the jvm.dll. For example,

```
C:\<<javainstallroot>>\jre\bin\client\jvm.dll
```

This would appear in the registry similar to Figure 1 (below).



Name	Type	Data
BuildNum	REG_SZ	893
ChartSupportPath	REG_SZ	C:\Program Files\Common Files\Business Objects\3.0\ChartSuppo
CommonFiles	REG_SZ	C:\Program Files\Common Files\Business Objects\3.0\bin\
DisableCheckForU...	REG_SZ	1
IIS3Detected	REG_SZ	
IIS4Detected	REG_SZ	1
JREPath	REG_SZ	C:\Program Files\Business Objects\JavaSDK\jre\bin\client\jvm.dll
NetscapeServerD...	REG_SZ	
Path	REG_SZ	C:\Program Files\Business Objects\Crystal Reports 11\

My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Business Objects\Suite 11.0\Crystal Reports

Figure 1

11. Browse to the following file:

\Program Files\Common Files\Business Objects\3.0\java\CRConfig.xml

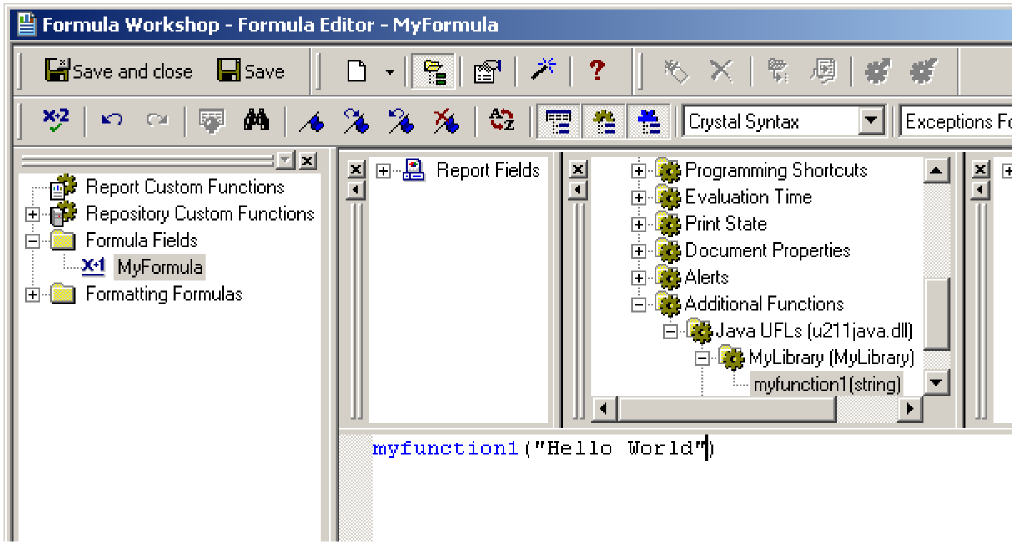
12. Add a reference to the library to this file.

For example, for the sample library "MyLibrary", add the following:

```
<ExternalFunctionLibraryClassNames>
   <classname>MyLibrary</classname>
</ExternalFunctionLibraryClassNames>
```

Use the UFL in a Formula

1. In a Crystal report, on the **Report** menu, click **Formula Workshop**.
2. Right-click **Formula Fields** then click **New**.
3. Name the formula *MyFormula* and click **OK**.
4. Under the **Functions** tree, expand the **Additional Functions** node then expand the **Java UFLs** node then expand the **MyLibrary** node (*See Figure* 2)



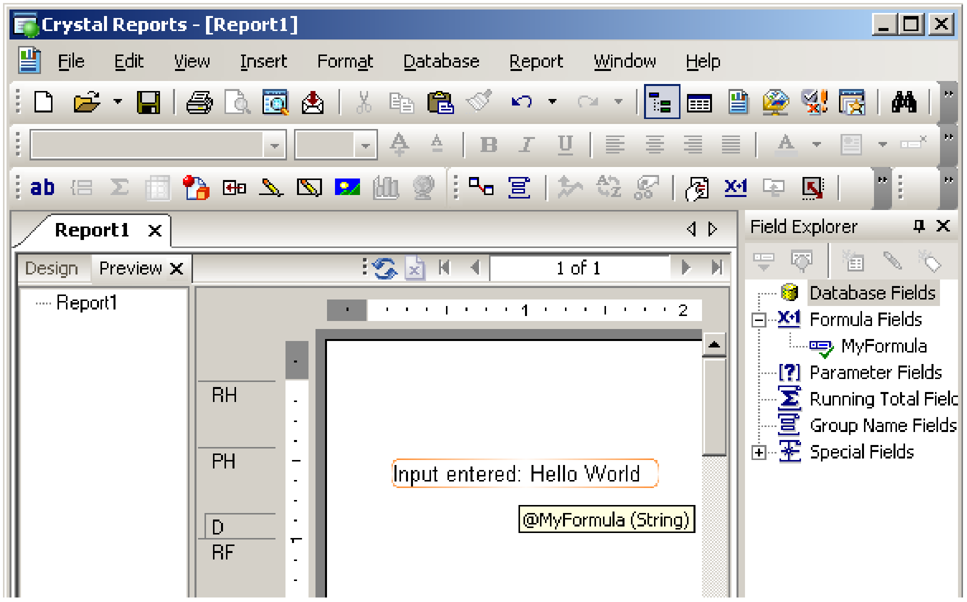
```
myfunction1("Hello World")
```

Figure 2 – Using the Java UFL in the Formula Workshop

5. Double-click the *myfuntion1* function and it will appear in formula window. Add text as this particular function requires text.

6. Click the **Save and Close** button to close the **Formula Workshop**.

7. Insert the formula onto the report. When the report is refreshed, *MyFormula* displays the following (*See Figure 3*):



Input entered: Hello World

Figure 3 – Previewing Report with 'MyFormula'

Configure your Computer to Deploy Reports with the JRC

To deploy reports that use Java user-defined function libraries with the JRC, copy the following JAR files to the WEB-INF\lib directory of your web application:

- CrystalFormulas.jar
- CrystalReportingCommon.jar
- log4j.jar
- icu4j.jar
- All of the JAR files required by the JRC (More Information)

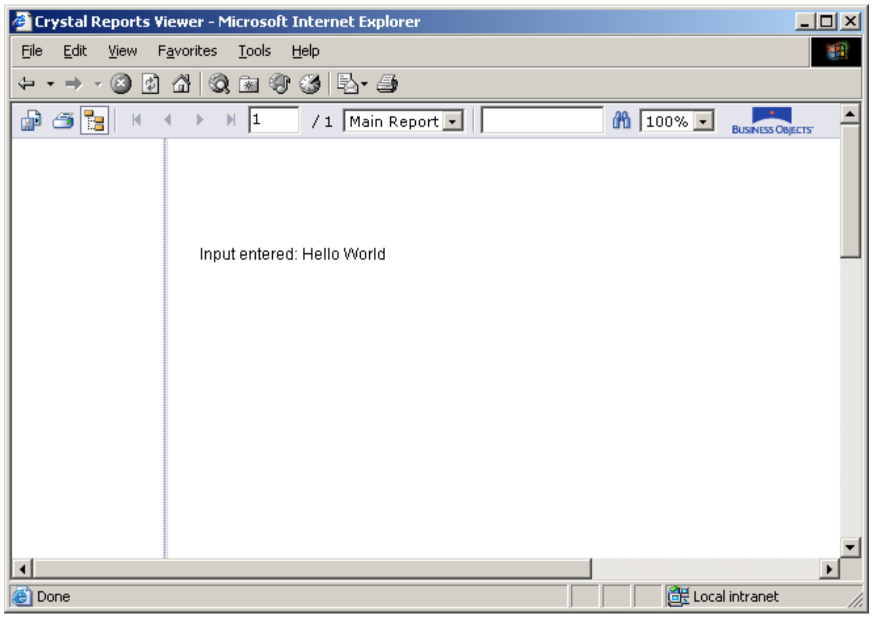
The compiled custom Java UFL .class files can be added to the WEB-INF/classes folder, or stored in a JAR file and added to the WEB-INF\lib folder.

Additionally, a reference to the library must be added to the CRConfig.xml file:

```
<ExternalFunctionLibraryClassNames>
    <classname>MyLibrary</classname>
</ExternalFunctionLibraryClassNames>
```

Run the Application

After configuring and deploying the JRC application to a Java Application Server, the UFL based report can now be viewed over the Web (*See Figure 4*):


Crystal Reports Viewer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
1 / 1 Main Report 100%
Done
Local intranet

Input entered: Hello World

Figure 4

More Information

For more information on using the Java Reporting Component XI, please see Crystal Reports Java Reporting Component Reference

For information on deployment for the JRC X search for article c2016910 at:

http://support.businessobjects.com/search/