



How to integrate Oracle BI Publisher via Web Services in Oracle Forms 11g

Version 4.0

White Paper, May 2013

Authors:

Axel Harsch, PITSS Jürgen Menge, Oracle Florin Serban, PITSS Rainer Willems, Oracle

Contributors:

Mireille Duroussaud, Oracle



Introduction
Creating a client for the Web Services with Oracle
Creating the Oracle Forms application
Importing the Java client in Oracle Forms Builder 20
Building the Forms application21
Parameters 24
Synchronous vs. asynchronous call 24
Configuring the Oracle Forms Runtime
Debugging26
Outlook 28



Introduction

This paper describes in a step-by-step way the generation of a client for the Web Services provided by Oracle BI Publisher and how to import and use this client in Oracle Forms 11g. This document is providing an update to the version 2.2, "How to integrate Oracle BI Publisher via Web Services in Oracle Forms"¹, including supplemental information regarding the integration between Oracle Forms 11g and BI Publisher 11g.

Used versions:

- Oracle Forms 11.1.2.0
- Oracle JDeveloper 11.1.2.3
 <u>http://www.oracle.com/technetwork/developer-tools/jdev/downloads/index.html</u>
- BI Publisher Enterprise 11.1.1.6 on WebLogic Server 10.3.5

The example included in this document should also work with Forms 11g, Rel. 2 (11.1.2.x) and JDeveloper 11g, Rel. 2 (11.1.2.x)

Oracle BI Publisher Web Service API

With Oracle BI Publisher 10.1.3.3.1 a Public Web Service API was firstly introduced into the product. After this initial implementation several versions of web services were integrated into the product. With the current release it is recommended a set of v2 web services which are documented here http://docs.oracle.com/cd/E23943_01/bi.1111/e22259/toc.htm

If Oracle BI Publisher is installed on a server the WSDL for the reporting web service can be found at: <u>http://<host>:<port>/xmlpserver/services/v2/ReportService?wsdl</u>

¹

http://www.oracle.com/technetwork/middleware/bi-publisher/overview/forms-bip-v22-129995.pdf



Creating a client for the Web Services with Oracle JDeveloper

We need to create a new application with Oracle JDeveloper, by selecting **Java Desktop Application** in the **New Gallery** dialog and entering the desired application name:

New Gallery		
Available Items Q Search		
<u>C</u> ategories:	Items:	Show All Descriptions
, General	🔁 Java Desktop Applica	tion
Connections Projects	Creates an application co application will include a JavaBeans features.	onfigured for building a generic Java application. The new project that is preconfigured to use Java, Swing, and
Web Services		
Database Tier		
Help		OK Cancel

Screate Java Desktop A Name your applicatio	Application - Step 1 of 3	×
Application Name Project Name Project Java Settings	Application Name: FormsBIPApp Directory: C:\JDeveloper\mywork\FormsBIPApp Brows Application Package Prefix:	se
Help	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cano	cel



📥 Create Java Desktop A	pplication - Ste	ep 2 of 3
Name your project		
Application Name	Project Name:	BIP_WebService
Review Name	Dir <u>e</u> ctory:	C:\JDeveloper\mywork\FormsBIPApp\BIP_WebService Browse
Project Java Settings	Project Fea <u>t</u> u	res:
	The Java pro challenges of distributed en	gramming language is a simple object-oriented language designed to meet the application development in the context of heterogeneous, network-wide wironments.
Help		< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cancel

We then name the project as *BIP_WebService* and press the **Finish** button.

Note: The code examples in this document are referring this project name, *BIP_WebService*. In case another name is needed, the code will have to be adjusted accordingly.

Note: In case a HTTP-Proxy is necessary, please define this in the **Tools** menu under **Preferences** at **Web Browser and Proxy**.



After the project is saved, right-click the project, choose **New** and in the **All Features** tab page select the Item **Web Services Client and Proxy** in the Category **Business Tier – Web Services**.

Q Search				
Categories:	Items:	Show All Descriptions		
General General	🧙 Java Web Service			
ADF Business Components	SJava Web Service from WSDL			
Business Intelligence	S PL/SQL Web Service	S PL/SQL Web Service		
EJB	S TopLink DB Web Service Provider	S TopLink DB Web Service Provider		
Web Services	Web Service Llient and Proxy Launches the Create Web Service Clien generate Java classes that can be used service. To enable this option, you must in the Application Navigator.	t and Proxy wizard, in which you d to make a call to a remote web select a project or a file within a project		
All Items	SAX-RPC Mapping File			
	👪 UDDI Registry Connection			
	🔯 URL Service Data Control			
	🛃 Web Service Data Control			
	WSDL Document			

Press **OK** and the Web Service wizard is opened. Accept the first two screens.

🖕 Create Web Service Client and Proxy - Step 1 of 7		
Create Web Service C	Client and Proxy	
🔍 Create Web Service	Welcome to the Create Web Service Client and Proxy Wizard.	
Select Client Style	This wizard helps you to create Java classes that can act as a proxy to a remote web service and a client class which uses the proxy to make a request to and retrieve a response from the web service.	
Specify Default Mapping	Click Next to continue.	
O Defined Handlers		
	Einish Cancel	



💩 Create Web Service Clie	nt and Proxy - Step 2 of 7
Select Client Style	01
🚊 Create Web Service Clie	Select the client that matches your requirements.
Select Client Style	<u>Client Style</u>
Select Web Service Desc	JAX-RPC Web Logic Style A client using the JAX-RPC mapping style for serialization.
Custom Mapping	JAX-WS Style A client in using the JAXB mapping style for serialization.
U Defined Handlers	
Help	< Back Next > Einish Cancel

Choose the appropriate URL of the web service endpoint of your BI Publisher Server in the **WSDL Document URL**:

http://<machine>:<port>/<yourappname>/services/v2/ReportService?wsdl

(Default for <yourappname> is *xmlpserver*)



े Create Web Service Clie	nt and Proxy - Step 3 of 7	X
Select Web Service De	escription	
O Create Web Service Clie	Select a WSDL document containing service descriptions.	
Select Client Style	WSDL Document URL:	
Select Web Service [http://vmpitssbi:9704/xmlpserver/services/v2/ReportService?wsdl	•
Specify Default Mapping		Browse
Custom Mappings	Copy WSDL Into Project	
O Defined Handlers		Binding Files
O Finish		
$\langle $		
<u>H</u> elp	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish	Cancel

Note: If for any reason a connection to this endpoint URL from JDeveloper is not possible, you can download the WSDL file in a web browser and copy it into a local directory of your application. Afterwards point to this local copy and proceed as described here.

Set the desired **Package Name** (here *bip_webservice.proxy*).

🖕 Create Web Service Client and Proxy - Step 4 of 7			
Specify Default Mapp	ing Options	01010101010101010101010101010101	9 05
Create Web Service Clie	Provide default mappings from WSDL parameters and headers are mapped	namespaces to Java packages. You car	also control how
Select Client Style	Package Name:	bip_webservice.proxy	Br <u>o</u> wse
Select Web Service Desc Specify Default Map	Root Package for <u>G</u> enerated Types:	bip_webservice.proxy.types	Brows <u>e</u>
Custom Mappings	The service appears to contain async	hronous features	
Defined Handlers	Generate As Async		
Ö Finish	<u>C</u> allback Package Name:		Browse
$\langle \bigcirc \rangle$			
Help	< <u>B</u> ack	Next > Einish	Cancel



Accept the next screen.

💩 Create Web Service Clie	vice Client and Proxy - Step 5 of 8		
Port Endpoints		0101010101010101040494949494949494949494	
Create Web Service Clie	If you wish, you o the service. This is or a different exte	an change the endpoint URLs that the proxy will use to access the ports of s useful if you want your proxy to run against a service on Integrated WLS, ernal server.	
Select Web Service Desc	Port Name	Endpoint URL	•
🗼 Specify Default Mapping	ReportService	http://vmpitssbi:9704/xmlpserver/services/v2/ReportService	
Port Endpoints			
Qustom Mappings			
O Defined Handlers			
O Finish			
()			
Help		< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cancel	

Select Don't generate any asynchronous methods in the next screen.

📤 Create Web Service Clie	nt and Proxy - Step 6 of 8
Asynchronous Metho	ods
🕂 Create Web Service Clie	You can choose to add methods to your proxy that allow clients to access the service in an asynchronous manner. This step allows you to choose the asynchronous methods to be added.
Select Client Style	Don't generate any asynchronous methods
Specify Default Mapping	 Generate asynchronous methods where specified by the JAX-WS binding Generate asynchronous methods for all operations
Asynchronous Metho	
Defined Handlers	
\leftarrow \rightarrow	
Help	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cancel



Accept the following two screens.

💩 Create Web Service Clie	nt and Proxy - Step 7 of 9	×
Policy		01010101010101010101010101010
🔎 Create Web Service Clie	Configure OWSM policies for	the web service client, if required.
Select Client Style	Policy Store (def location):	ber\system11.1.2.3.39.62.76.1\DefaultDomain\oracle\store\gmds
Select Web Service Desc	Show only the <u>c</u> ompatible	client policies for selection.
Specify Default Mapping	Ports:	ReportService
Port Endpoints	<u>M</u> TOM	
Asynchronous Methods	R <u>e</u> liability	
Policy	<u>A</u> ddressing	
Defined Handlers	Security Management	
O Finish	Policies: Q Search	Show Selected Policies Show Descriptions
	🗌 🔲 oracle/no_authenti	cation_client_policy
	This policy facilitates th This will include disablin assertions in addition to	e disabling of a globally attached authentication policy. <u>View</u> ng that whole global policy containing any other o the authentication assertion.
	oracle/no_message	protection_client_policy
	This policy facilitates th	e disabling of a globally attached message protection
(\longrightarrow)	Override Properties	
Help	<	Back Next > Finish Cancel

💩 Create Web Service Clie	ent and Proxy - Step 8 of 9	×
Defined Handlers	01010101010101010404040404	
Create Web Service Clie	Specify any handler classes you have which will deal with the web service message. The defined handlers may have associated initialization parameters, SOAP roles or SOAP head	lers.
Select Client Style	Ports: Ports	-
Create Web Service Client a	Defined Handlers:	
Specify Default Mapping	Add.	
Port Endpoints	Remo	ove
Asynchronous Methods		
Policy		
Defined Handlers		
🧉 Finish	Details Init Params Roles	
	Handler Name:	
	Handler Class: Browse	



Check that the *runReport()* method is available in the **Finish** screen.



JDeveloper creates a proxy stub (proxy client) to call the selected web service methods.



Now the web service needs to be tested. For this, the *main*() method needs to be edited in the *ReportServiceClient* class (please find the line "// Add your code here"), as in the following code fragment. This will call the BI report which was created with the help of the following document: Getting Started with Oracle Business Intelligence Publisher http://docs.oracle.com/html/E28374_01/toc.htm

Note: Adjust username, passwords, directory names accordingly to your environment.



```
public static void main(String [] args) throws AccessDeniedException,
InvalidParametersException, OperationFailedException, IOException
  {
    trv
    {
      ReportService Service reportServiceService = new
ReportService Service();
      ReportService
                            reportService
reportServiceService.getReportService();
      // Add your code to call the desired methods.
      final String username = "weblogic";
      final String password = "Welcome1";
      final String reportAbsolutePath = "~weblogic/My Data Models/My
Reports/Employee Report.xdo";
      // Testing runReport
      System.out.println("Testing runReport Service");
      ReportRequest repRequest = new ReportRequest();
      //Set general Report Parameters
      repRequest.setReportAbsolutePath(reportAbsolutePath);
      repRequest.setAttributeTemplate("Chart and Table Layout");
      repRequest.setAttributeFormat("pdf");
      repRequest.setAttributeLocale("en-US");
      repRequest.setSizeOfDataChunkDownload(-1);
      //Set User Parameter P DEPT
      ArrayOfParamNameValue arrayOfParamNameValue = new
ArrayOfParamNameValue();
      ParamNameValues paramNameValues = new ParamNameValues();
      ParamNameValue paramNameValue = new ParamNameValue();
      ArrayOfString arrayOfString = new ArrayOfString();
      paramNameValue.setName("P DEPT");
      arrayOfString.getItem().add("RESEARCH");
      paramNameValue.setValues(arrayOfString);
      arrayOfParamNameValue.getItem().add(paramNameValue);
      paramNameValues.setListOfParamNameValues
(arrayOfParamNameValue);
      repRequest.setParameterNameValues(paramNameValues);
      ReportResponse repResponse = new ReportResponse();
      repResponse = reportService.runReport(repRequest, username,
password);
      String contentType = repResponse.getReportContentType();
      System.out.println(contentType);
      byte[] baReport = repResponse.getReportBytes();
      FileOutputStream fio = new FileOutputStream("C:\\test.pdf");
        fio.write(baReport);
        fio.close();
    } catch (Exception ex) {
      ex.printStackTrace();
    }
  }
```

After following all the JDeveloper' hints to import necessary classes, the code should look as in the following screenshot:





Starting with BI Publisher 10.1.3.4 there is a new and important parameter in the complex type *ReportRequest - SizeOfDataChunkDownload*. According to the documentation this parameter should be set to **-1** if you don't want to split the resulting data into chunks. Otherwise BI Publisher will produce an empty file.



The Web Service Client may be tested now by clicking **Run** in the context menu of *ReportServiceClient.java* or in the toolbar. During the test the method *main()* is used.

To call the *runReport()* method from our Forms application we need some additional code because we should not:

- use the method main() from outside
- modify the generated method runReport().

There are two conceptual ways to do that:

- 1. to write a wrapper class around ReportServiceClient.java
- 2. to write an additional, customized method in the *ReportServiceClient.java* which will be called from outside

For reasons of simplicity we will select here the second solution and add a method *callRunReport()* in the *ReportServiceClient.java*.

First we define the parameters in the class PublicReportServiceClient.

```
public class ReportServiceClient {
  private String username;
  private String password;
  private String reportPath;
  private String format;
  private String template;
  private String locale;
  private int sizeOfDataChunkDownload;
  private String paramName;
  private String outFile;
```

and initialize them

```
public ReportServiceClient() throws Exception
{
    username = null;
    password = null;
    reportPath = null;
    format = null;
    template = null;
    locale = "en-US";
    sizeOfDataChunkDownload = -1;
    paramName = null;
    outFile = null;
}
```





Now we will add a public method *callRunReport()* that will call the generated method *runReport()* in the web service client. The key argument of this method is called *arrayOfParamNameValue* and is used to transport the information regarding the user parameters (pairs of name and values).

```
reportPath
public void callRunReport(String
                         ArrayOfParamNameValue
                                                  arrayOfParamNameValue,
                         String
                                                  username
                                                                       ,
                                                  password
                         String
                                                                       ,
                         String
                                                  format
                                                                       ,
                         String
                                                  template
                         String
                                                  outFile
                                                                       ) {
 try {
   ReportService_Service reportServiceService = new ReportService Service();
   ReportService reportService =
reportServiceService.getReportService();
   ParamNameValues paramNameValues
                                            = new ParamNameValues();
   // Calling runReport
   ReportRequest reportRequest = new ReportRequest();
   reportRequest.setReportAbsolutePath(reportPath);
   reportRequest.setAttributeTemplate(template);
   reportRequest.setAttributeFormat(format);
   reportRequest.setAttributeLocale(locale);
   reportRequest.setSizeOfDataChunkDownload(sizeOfDataChunkDownload);
   //set Parameters
   paramNameValues.setListOfParamNameValues(arrayOfParamNameValue);
   reportRequest.setParameterNameValues(paramNameValues);
   ReportResponse repResponse = new ReportResponse();
   repResponse = reportService.runReport(reportRequest, username, password);
   byte[] baReport = repResponse.getReportBytes();
   FileOutputStream fio = new FileOutputStream(outFile);
   fio.write(baReport);
   fio.close();
  } catch (Exception ex) {
    ex.printStackTrace();
```





Hint:

The address of the webservice endpoint (i.e. your host, port and application) is defined in the file *ReportService_Service.java*.





Now we have built and tested the Web Service Client from within JDeveloper. In order to call the Web Service from Forms, the Web Service Client must be deployed to the file system as a JAR file. (As an alternative to a JAR file which is a type of archive it is possible to use the generated files directly).

By right-clicking the project and selecting **New** from the context menu, the **New Gallery** will open. We need to select the **Category: General – Deployment Profiles** and then the **JAR File** item.

Q Search		
⊆ategories:	Items:	Show All Descriptions
Deployment Profiles	ADF Library JAR File	
Diagrams	Business Components Archive	
Java Maven	Business Components Service Inte	erface
Projects		
UML		
XML	EAR File	
Business Tier ADE Business Components	EJB JAR File	
Business Intelligence		
Data Controls		
EJB	🗋 JAR File	
Security	Create a simple JAR archive from	a Project.
TopLink/JPA Wob Somicos		
⊡ Database Tier	OSGi Bundle	
	🕅 RAR File	
All Items		



We can indicate the deployment profile name (here *mystub*).

🖕 Create Deployment Profile JAR File	×
Click OK to create your new deployment profile and immediately open it to see its configuration.	
Profile <u>Type</u> :	
JAR File	
Deployment Profile <u>N</u> ame:	
mystub	
Description:	
Create a simple JAR archive from a Project.	
Help OK Cancel	

Then we will accept the JAR Deployment Profile properties.

🖕 Edit JAR Deployment Pro	le Properties		×
Edit JAR Deployment Pro	Image: Properties JAR Options JAR File: Veloper\mywork\FormsBi Compress Archive Compress Archive Compression Level ✓ Include Manifest File Main Class: Additional Manifest File	IPApp\BIP_WebService\deploy\mystub.jar Default (META-INF/MANIFEST.MF) jiles to Merge into MANIFEST.MF	Erowse Browse Add Delete
New Delete		OK	Cancel



By selecting the	Filters option v	e can verify that a	all the necessary files a	re selected.
------------------	------------------	---------------------	---------------------------	--------------

ar Edit JAR Deployment Profile Prop	erties X
Q Search	Filters Files Patterns This file group includes the project output directory as a contributor. You may need to compile the project to see all files coming from the output directory. Image: Contents of This File Group's Contributors Image: Contents of This File Group's Contents of This
New Delete	Expand All Nodes Collapse All Nodes
Help	OK Cancel

Right-click the archive file under the **Resources** node and select **Deploy to JAR file**. This will deploy the JAR file to the file system.

🥧 Oracle JDevelope	r 11g Release 2 - FormsBIPApp.jws : BIP_WebService.jpr
<u>File E</u> dit <u>V</u> iew A	pplication Refa <u>c</u> tor <u>S</u> earch <u>N</u> avigate <u>B</u> uild <u>R</u> un Versi <u>o</u> ning <u>T</u> ools <u>W</u> indow <u>H</u> elp
🔮 🗁 📰 🎒 i 🖷	- 🌞 - 🤜 I - 😹 👪 🚵 I - 🔘 - 🔍 -
📊 🖉 Application N	avigator 🗴 📃 🕐 Start Page 🗴 🗟 ReportServiceClient.java 🗴 🛃 ReportService
FormsBIPApp	🗕 🚽 💽 🖌 🖉 🖉 🖉 🖉 🖉
Projects	Image: Compare With Replace
	Restore from Local History REPORTSERVICE_MSDL_LOCATION = url. Q Project Properties }
	public ReportService_Service(URL wsdl: super(wsdlLocation, serviceName);



Creating the Oracle Forms application

Importing the Java client in Oracle Forms Builder

Oracle Forms needs to be able to see the relevant Java files in the Forms Builder during design time. This includes at least the generated jar-File (*mystub.jar*) with the Web Service Client. For this, all the Java classes that need to be imported in Forms must be visible in the Forms Builder Classpath. There are two possible ways to expose the necessary classes to Forms Builder:

- 1. By entering all the necessary jar-Files including their absolute path in the registry key FORMS_BUILDER_CLASSPATH for the Oracle Home of your Developer Suite installation.
- 2. By setting the environment variable FORMS_BUILDER_CLASSPATH in a script and calling the Forms Builder from this script.

Here is an example:

Г



Building the Forms application

In the Forms Builder we will create a new Forms module and use the Java Importer from the menu **Program - Import Java Classes** to import the following classes:

- bip_webservice.proxy.ReportServiceClient
- bip_webservice.proxy.types.ArrayOfParamNameValue
- *bip_webservice.proxy.types.ArrayOfString*
- *bip_webservice.proxy.types.ParamNameValue*
- java.lang.String
- java.util.List

Import Java Classes	×
Select Java Classes:	
term org	
terrice	
±com	
AccessDeniedException	
InvalidParametersException	
🔤 📑 OperationFailedException	
ReportService	
BeportService Service	
tunes	
	-
4	Þ.
Import Classes: bip_webservice.proxy.ReportServiceClient	
Managan	
	<u> </u>
Import Liose Uptions	

If the classes are not accessible we have to modify the classpath and restart the Forms Builder. If the classes are accessible, in case we get error messages during the import, there may be some referenced classes that are missing in the classpath. In this case, we need to add them and restart the Forms Builder.



If the import succeeds we will get 6 new packages:

- ArrayOfParamNameValue
- ArrayOfString
- List
- ParamNameValue
- ReportServiceClient
- String

We need to rename the package List to Java_List and the package String to Java_String.

Now we can define the application logic in Forms and call the BI Publisher web service by using the *ReportServiceClient.callRunReport()* PL/SQL procedure.

Here is an example how the code may look like. The code assumes that there is a block named *CONTROL* with the items *REPORT_PATH, TEMPLATE, FORMAT, OUT_FILE_NAME, PARAM_VALUE*, a *GENERATE_REPORT* button and an optional item *EXCEP* in case we need to implement exception handling.

🚳 Oracle Forms Builder - E:\BI_Publis	sher_Research\TEST_BIP_11G.fmb - [TEST_BIP_11G: CNV_MAIN (CONTROL)]	
File Edit View Layout Program	Debug Tools Window Help	
D 🚅 🖬 🕹 XX 🖻 🛍 🎭	🔜 🕑 🕼 ▶ →E ÇE ^t E 💷 🔀 💞 ?	
Canvas: CNV_MAIN	▼ Block: CONTROL ▼ G 単 世	
Tr Arial (Westlich)	╸│᠀╺╸│вェ⊻│�ੑੑੵੑ╞₽日णөш│ҫ┖。	
0 16 32 48 64 80 96	11212814416017619228822424025627228830432033635236838440041643244846448049651252	28 54
	II Oracle BI Publisher Web Service	
		1
	F	-
A 📑 📴 Report Path :	REPORT_PATH EXCEP	
Template :	TEMPLATE	
• E 100		
E 6 144 Format :	FORMAT	
Filename :	OUT_FILE_NAME	
Parameter Value :	PARAM_VALUE	
208		
224	Generate Report	
240		
256		

Here is the program code of the WHEN-BUTTON-PRESSED trigger:

Note: Adjust username, passwords, directory names accordingly to your environment.

Declare		
lBIUsername	Varchar2(200);	
lBIPassword	Varchar2(200);	
lOutputFile	Varchar2(200);	
lOutputFormat	Varchar2(10);	
lFileExtension	Varchar2(10);	



```
Varchar2(200);
lParameterName
lParameterValue
                       Varchar2(200);
lReportPath
                       Varchar2(200);
lReportTemplate
                      Varchar2(200);
lObj
                      Ora Java.JObject;
lArrayOfParamNameValue Ora Java.JObject;
lParamNameValue Ora Java.JObject;
lArrayOfString
                      Ora Java.JObject;
lDone
                       Boolean;
Begin
  lObj := ReportServiceClient.New();
  -- Login information for BI Publisher server
  lBIUsername := 'weblogic';
  lBIPassword := 'Welcome1';
  -- Set the report path and report template
  lReportPath := :control.report path;
  lReportTemplate := :control.template;
  -- Set the format and the extension for the output file
  If :format ='excel' Then
   lFileExtension := 'xls';
  Else
    lFileExtension := :control.format;
  End If;
  lOutputFormat := :control.format;
  -- Set the download directory
  lOutputFile := 'E:\Oracle\Middleware\as1\forms\java\' ||
                 :control.out file name || '.' || lFileExtension;
  -- Pass the user parameter
  lParameterName := 'P DEPT';
  lParameterValue := :control.param value;
  lArrayOfParamNameValue := ArrayOfParamNameValue.New();
  lParamNameValue := ParamNameValue.New();
  ParamNameValue.SetName(lParamNameValue, lParameterName);
  lArrayOfString := ArrayOfString.New();
  lDone := Java List.Add(ArrayOfString.GetItem(lArrayOfString),
                         Java_String.New(lParameterValue));
  ParamNameValue.SetValues(lParamNameValue, lArrayOfString);
  lDone :=
Java List.Add(arrayOfParamNameValue.GetItem(lArrayOfParamNameValue),
              lParamNameValue);
  -- Call callRunReport
  ReportServiceClient.CallRunReport(10bj
                                    lReportPath
                                    lArrayOfParamNameValue,
                                    lBIUsername
                                    lBIPassword
                                                          ,
                                    lOutputFormat
                                                          1
                                    lReportTemplate
                                    lOutputFile
                                                          );
  --Show Report in browser--
  Web.Show Document( 'http://vmpitssbi:8090/forms/java/' ||
               :control.out file name || '.' || lFileExtension, ' blank');
End;
```



The report will be created as a file on the application server. Afterwards this file can be displayed with the built-in *web.show_document()*. For a multi-user environment we need to generate unique filenames.

The file in this simple example is generated into \$ORACLE_HOME\forms\java folder (*E*:*Oracle\Middleware\as1\forms\java*) which is mapped to the *"/forms/java/*" virtual directory.

The sample code has no error handling (which would be a good idea to have).

Parameters

ReportPath:	the location of the report definition file relative to the BI Publisher repository
	(example. /nk managel/Employee Salary ReportEmployee Salary Report.xuo).
ParameterName:	The name of the parameter passed to the BI Publisher web service.
ParameterValue:	The value of the parameter passed to the BI Publisher web service.
BIUsername /	
BIPassword:	Username and Password for authentication against the BI Publisher server
	Note: Instead of passing username and password to the BI Publisher server it is
	recommended to use the method login() or impersonate() to get a valid session
	cookie from BI Publisher server. With a valid session cookie the method
	runReportInSession() has to be used instead.
OutputFormat:	The desired format of the document. Valid values are pdf, rtf, excel, xml and html.
	If in the BI Publisher environment a specific format is not allowed, it cannot be
	generated via web services, too.
ReportTemplate:	The name of the template
	(Example: Table and Chart Layout)
	(we need to use the logical name of the template, not the physical one).
OutputFile:	Directory path and filename that needs to be generated.

Synchronous vs. asynchronous call

The example uses a synchronous call of the BI Publisher Web Service, in order to call the document via *web.show_document()* directly after the call of the method *callRunReport()*. For long running reports it would be better to call the reports asynchronous, which could be done via a multi-threated web service client.



Configuring the Oracle Forms Runtime

In the Forms runtime environment we need to specify all the used and referenced classes via the CLASSPATH variable of the *default.env* environment file. It should contain a list of all required jar-files, including their absolute path on the application server machine. Here is an example of the CLASSPATH variable in the *default.env* file:

```
CLASSPATH = E:\Oracle\Middleware\asl\forms\j2ee\frmsrv.jar;
E:\Oracle\Middleware\asl\jlib\ldapjclnt11.jar;
E:\Oracle\Middleware\asl\jlib\debugger.jar;
E:\Oracle\Middleware\asl\jlib\ewt3.jar;
E:\Oracle\Middleware\asl\jlib\share.jar;
E:\Oracle\Middleware\asl\jlib\utj.jar;
E:\Oracle\Middleware\asl\jlib\zrclient.jar;
E:\Oracle\Middleware\asl\reports\jlib\rwrun.jar;
E:\Oracle\Middleware\asl\forms\java\frmwebutil.jar;
E:\Oracle\Middleware\asl\jlib/start_dejvm.jar;
E:\Oracle\Middleware\asl\opmn\lib\optic.jar;
```

It is recommended to have a separate configuration in the *formsweb.cfg* and to define here a specific environment file for this application.

```
[bip_webservice]
envFile=bip_webservice.env
...
```

In the newly created *bip_webservice.env* file we need to append to the CLASSPATH variable the path to the *mystub.jar* as in the following example:

```
CLASSPATH = E:\Oracle\Middleware\as1\forms\j2ee\frmsrv.jar;
E:\Oracle\Middleware\as1\jlib\ldapjclntl1.jar;
E:\Oracle\Middleware\as1\jlib\debugger.jar;
E:\Oracle\Middleware\as1\jlib\ewt3.jar;
E:\Oracle\Middleware\as1\jlib\share.jar;
E:\Oracle\Middleware\as1\jlib\utj.jar;
E:\Oracle\Middleware\as1\jlib\zrclient.jar;
E:\Oracle\Middleware\as1\reports\jlib\rwrun.jar;
E:\Oracle\Middleware\as1\forms\java\frmwebutil.jar;
E:\Oracle\Middleware\as1\jlib/start_dejvm.jar;
E:\Oracle\Middleware\as1\opmn\lib\optic.jar;
E:\Oracle\Middleware\as1\forms\java\mystub.jar
```

In our case *E*:\Oracle\Middleware\as1 represents the ORACLE_HOME environment variable.

Now we may test the form by running the named configuration and calling the BI Publisher report from the application:

http://<host>:<port>/forms/frmservlet?config=bip_webservice& ...



Debugging

When running the Form for the first time we may get an unhandled exception ORA-105100 or ORA-105101. This indicates that Java classes are still missing in the runtime environment or that a Java error occurred during the execution of the code.

It would be helpful to have some more information about what happened by using some kind of exception handling. There is a way to display the Java error stack in the Form which could be of great help finding the cause of the error.

For this, we need to import the *java.lang.Exception* and *java.lang.StackTraceElement* classes and to activate the following options in the Forms-Java Importer:

- Include inherited methods/fields
- Include get/set for public Fields

Import Java Classes	
Select Java Classes:	<u>.</u>
±… netscape	Import Java Class Options 🛛 🗙
± mbip_webservice	- Import Options
	✓ Include inherited methods/fields
	Include get (set for public fields)
	<u>L</u> enerate persistent names
	Ask before overwriting existing packages
	OK
Import Classes: iava.lang.Exception	
Messages:	
Import Close Options	

In the code example below a nested exception catches the errors during the execution of the *Exception_toString()*.

We can add this code to our WHEN-BUTTON-PRESSED Trigger after importing the necessary classes.



```
Declare
. . . . . . . . . .
raisedException Ora Java.JObject;
stack trace Ora Java.JObject;
stackTrcElement Ora Java.JObject;
Begin
. . . . . . . . . .
-- Exception handling for Java-errors
Exception
  -- Check for ORA-105100
  When Ora_Java.Java_Error Then
   Message('Unable to call out to Java, ' || Ora_Java.Last_Error);
   Return:
  -- Check for ORA-105101
  When Ora_Java.Exception_Thrown Then
    raisedException := Exception .New(Ora Java.Last Exception);
    Begin
      :control.excep := 'Exception: ' ||
                         Exception .ToString(raisedException);
    Exception
      When Ora Java.Java Error Then
        Message ('Unable to call out to Java, ' || Ora Java.Last Error);
        Return;
    End;
    -- Get an array of StackTraceElement from the Exception
    stack trace := Exception .GetStackTrace(raisedException);
    -- Loop over all the Elements
    For i In 0 .. Ora_Java.Get_Array_Length(stack_trace) Loop
      -- Get each Element
      stackTrcElement := Ora_Java.Get_Object_Array_Element(stack_trace,
i);
      --Make a string out of it and add it to the error field
      :control.excep := :control.excep || Chr(10) ||
                         stackTraceElement.ToString(stackTrcElement);
    End Loop;
    Ora Java.Clear Exception;
    Return;
  When Others Then
   Message('Problem!');
   Return;
End;
```



Outlook

The example could be extended in many ways:

- It is possible to pass also complex parameter structures with more than one parameter and more than one value per parameter (multiple selection)
- Another direction could be to separate the processing of the report from the Forms application by using an asynchronous call of BI Publisher Web Service.
- It is also possible to write the generated report as a byte stream into a CLOB column in the database which would give us the opportunity to use features from the database (security, stored procedures, AQ etc.).

In case of questions please feel free to contact us Find our contacts in the lower right We like to support you

Please also have a look at PITSS' services page

About PITSS

PITSS is the leading provider of software & services for modernizing and effectively managing Oracle applications. The PITSS Group was established in 1999 and has gained international recognition with over 1,000 customers and a multitude of successful Oracle projects. PITSS is an Oracle Gold partner and, as a member of the Oracle Modernization Alliance (OMA), is the only Oracle Forms Migration partner for automated migrations. With sites in Stuttgart, Munich (Germany) and Troy (USA) as well as certified international partners, the company successfully provides support for IT projects of medium sized companies, large enterprises and public contractors across the globe.



How to integrate Oracle BI Publisher via Web Services in Oracle Forms 11g

May 2013

Authors: Axel Harsch, PITSS Jürgen Menge, Oracle Florin Serban, PITSS Rainer Willems, Oracle

Contributors: Mireille Duroussaud, Oracle

PITSS in Europe

Germany +49-711-728.752.00 info@PITSS.com www.PITSS.com

PITSS in Americas USA 248.740.0935 info@PITSSamerica.com www.PITSSamerica.com

Copyright 2013, PITSS GmbH All rights reserved