

Integrating Oracle BI Publisher Reports into Oracle ADF Applications

ORACLE WHITE PAPER | MARCH 2015



Table of Contents

Purpose of this Paper	1
Introduction to Oracle ADF and Oracle BI Publisher	1
Getting Started	3
Oracle BI Publisher	3
Oracle ADF	3
Implementation	4
Base Application	4
What was already done for you	4
What still has to be done	8
Consuming Application	10
What was already done for you	10
What still has to be done	10
Run and test the ADF Application	16
Summary	20

Purpose of this Paper

This paper describes the integration of enterprise reporting using Oracle BI Publisher into custom applications build with Oracle's Application Development Framework (ADF). The target audiences are mainly developers with basic knowledge of Oracle BI Publisher, Oracle ADF and Java. The example consists of two ADF applications which can be downloaded. The base application needs only small modifications to fit into the reader's technical environment. The consuming application only contains an empty page and has to be completed by the reader. The necessary steps are described in the following paper.

Introduction to Oracle ADF and Oracle BI Publisher

Built on top of Java Enterprise Edition (Java EE), Oracle Application Development Framework (ADF) provides a complete and integrated framework for enterprise web application and mobile browser application development. Within Oracle, ADF is used as the foundation of Oracle Fusion Applications, Oracle Cloud and On-premise Fusion Middleware products. Outside of Oracle, a constantly growing number of customers use Oracle ADF for building custom software solutions, or to extend the functionality of Fusion Applications.

For more information about Oracle ADF see:

http://www.oracle.com/technetwork/developer-tools/adf/overview.

With Oracle ADF Faces, the JavaServer Faces based view layer framework in ADF, developers build web pages declaratively and component based, including pages that are for information purposes like reporting.

Often the requirement in enterprise reporting is to produce high-quality reports that use pixel-perfect layouts and that are printed in document formats like PDF, MS Word, MS Excel or MS PowerPoint. This is where another Oracle product, Oracle BI Publisher, can be used to complement Oracle ADF online reporting capabilities.

Oracle BI Publisher is a modern, standard-based solution to develop, generate and distribute various types of business reports and documents. It is available to customers along with BI Suite of products and can also be installed as a standalone product. BI Publisher is also used by over 80 Oracle Products such as Enterprise Manager, Siebel CRM, and Oracle Financial Services etc. For more information about Oracle BI Publisher see:

http://www.oracle.com/technetwork/middleware/bi-publisher/overview.

Oracle BI Publisher provides three options for integrating with Oracle ADF:

- » Executing a report by calling a URL
- » Using the BI Publisher Java API
- » Using the BI Publisher web service API

The BI Publisher web service API is powerful and comprehensive. Oracle recommends this integration approach and therefore this whitepaper will focus on this approach.

For more information about Oracle BI Publisher web service API see: http://docs.oracle.com/cd/E28280_01/bi.1111/e22259/toc.htm.

Getting Started

You can download the sample code for this whitepaper. To work through the steps and complete the example you need the following software installed on your computer.

Oracle BI Publisher

In this example version 11.1.1.7.1 of BI Publisher is used. But it should also work with previous versions of the 11g release without modifications. For the ADF integration, this whitepaper however relies on the BI Publisher V2 web services introduced in release 11g.

BI Publisher 11*g* can be used as part of the Oracle BI EE installation or from the Trial Edition because the required web services are available in both installations.

In BI Publisher's server configuration the security model is set to "BI Publisher Security" but it should work also with other security models.

ORACLE BI Publisher Enterprise	Search <mark>All</mark>	¥	Administration Help	→] Sign Out $igodol Q$
Administration	Home	e Catalog 📑 New 🗸	📙 🔁 Open 🗸 🛛 Signed In As	administrator ~
Administration > Security Configuration				3
Security Center				
Security Configuration Users Roles and Permissions Digital Signature				
✓TIP Any changes will only take effect after the application is restarted.			Арр	ly Cancel
Authorization				
Security Model BI Publisher Security				

Figure 1 - Configuration of security model in Oracle BI Publisher

The example uses a set of sample reports provided by Oracle. So there is no need to create your own reports. Only the data sources of the sample reports have to be configured in a way that the required data can be found.

Oracle ADF

The ADF applications of this example are developed with Oracle JDeveloper 12c (12.1.3.) but also previous versions of JDeveloper or Oracle Enterprise Pack for Eclipse (OEPE) could be used for this purpose. Because the provided sample application was produced with JDeveloper 12.1.3 it is not possible to import it in previous versions of JDeveloper or in OEPE. In this case the sample applications could be used as a blueprint including custom Java code which is reusable.

Implementation

The example consists of two JDeveloper applications

» Base application BipClientApplication

contains the required functionality to generate reports using the web service API of BI Publisher. This application will be provided with the downloaded sample code and needs only some small modifications (host, port, user, password, etc. of BI Publisher server).

» Consuming application *BIPublisherApp* where the functionality from the base application will be integrated. This application has to be developed from scratch based on the description in this paper.

Base Application

What was already done for you

The base application *BipClientApplication* includes two projects *BipWebService* and *BipInterface* and is provided as a sample.

The project *BipWebService* consists of the following parts:

- » web service client communicating with BI Publisher server
- » wrapper code to prepare the required data structures and methods
- » data control to expose the required data structures and methods

The project BipInterface consists of the following parts:

- » bounded task flow to control the process of selecting report parameters and execute the report
- » page fragment with the user interface to select the report including runtime parameters
- » test page to display the bounded task flow as a static region in the UI

The base application was developed in the following steps. The description can only give an overview and assumes some experience with Oracle JDeveloper and ADF.

(1) First a JAX-WS client has to be gnerated for the BI Publisher web services

In this step a running instance of BI Publisher server 11.1.1.7.x is required (see Getting Started => Oracle BI Publisher). The client was generated using the wizard *Create Web Service Client and Proxy* into package *oracle.bip.webservice*.

From the four existing web service endpoints of BI Publisher 11.1.1.7.x we used the following three in our example:

http://<host>:<port>/xmlpserver/services/v2/ReportService?wsdl http://<host>:<port>/xmlpserver/services/v2/SecurityService?wsdl http://<host>:<port>/xmlpserver/services/v2/CatalogService?wsdl In order to generate the data controls based on Java classes (POJO) we need some wrapper code that exposes the required data structures and methods to the view layer.
 Here is a picture of the class structure the wrapper contains.





These classes are grouped in package oracle.bip.client.ui.

The method *main()* in *BipWrapper* is used only for testing purposes in JDeveloper. If such a test is required the parameters (admin user, admin password, report user, repository folder) should be set to valid values in the code (see below).

Requesting the execution of a report by BI Publisher server normally requires an authentication against the security provider (in our case BI Publisher server itself). The example uses two features of BI Publisher for authentication:

- » login at BI Publisher server returns a **token** (_sessionId) which is subsequently used in all web service calls (*inSession* methods)
- » to allow the login of arbitrary users the method *impersonate()* is used for the login. In this case the authentication is done always for the admin user on behalf of the user who wants to execute a report. The method logs in using admin account privileges, and then switch the owner of the BI Publisher server session to the passed-in username.

(3) When the wrapper code delivers the expected results the POJO data control can be generated for the class *BipWrapper*. This is done simply by running the option *Create Data Control* from the context menu for the class *BipWrapper*. Data control can be regenerated every time after modifications in the wrapper code (caution: to regenerate the data control the name of the data control must be identical).

(4) Using the generated data control the user interface will be build in project *BipInterface*. The task flow *bip-taskflow.xml* consists of only two activities

- » a method call login
- » a page fragment *bip.jsff* with UI components.



Figure 3 - BI Publisher task flow

To get the desired results the task flow has four mandatory parameters:

Input Parameter	Description	Example	Data Type
BipAdminUserName	BIP Admin user name	administrator	String
BipAdminPassword	BIP Admin password	weblogic1	String
BipUser	BIP user to impersonate. This is usually a normal user that exists in BIP. For example, each ADF application user has an account in BIP, and in order for each user to view the reports he has privilege to view, the logged in ADF user can be passed.	This could be static: appuser Or dynamic: #{SecurityContext.userName}	String
BipFolderPath	Reports path as specified in BI Publisher which resides under Shared Folders	/Samples/Human Resources Note: make sure path starts with /	String

(5) Because a page fragment cannot be executed on its own a test *page test.jsf* is provided as part of the base application. The task flow is implemented as a static region inside the test page. When the task flow is dragged into the page values for the parameters have to be provided.

The parameters and the generated report are processed in the Java bean *oracle.bip.ui.jsf.beans.BipBean.java* which is registered in *bip-taskflow* as a managed bean with page flow scope.

What still has to be done

(1) In the downloaded sample application the endpoint URLs have to be changed to an existing instance of BI Publisher. The best way is to use the search functionality in JDeveloper (*Search => Find in Files*) to find all occurrences of *xmlpserver* and replace host name and port number to valid values.

(2) If the page *test.jsf* is used for testing the parameter values have to be changed to the existing BI Publisher environment by editing the task flow binding for *taskFlow - biptaskflow1*. You have to provide valid values for (see page 8):

- » name and password of BIP admin user
- » name of BIP user
- » name of shared BI Publisher folder

	taski	=low - biptaskflow1
	Edit Ta	sk Flow Binding X
Task Flow:	/WEB-INF/T	askFlows/bip-taskflow.xml#bip-taskflow
Input Param	eters Map:	~
Input Para	imeters	
Name		Value 🔻
BipAdmi	nUserName *	administrator
BipAdmi	nPassword *	weblogicl
BipUser	*	appuser
BipFolde	rPath *	/Samples/Human Resources
* = Requ	iired	

Figure 4 - - Setting parameter values in task flow binding

You can test the page test.jsf simply by running it using the Integrated WebLogic server of JDeveloper.

(3) To consume the developed functionality in other applications is has to be exported as an ADF library jar file.

This should be done by executing the context menu option *Deploy* => *BipClient-taskflow* for the project *BipInterface*. The generated jar file can be found in the directory *./BipClientApplication/BipInterface/deploy* and should be copied afterwards into the directory *./Resources*.

Applications	×	🔕 test.jsf 🗡			
🔁 BipClientA	Application 🔹 💌	🚱 💌 Design	True 🛛 🔁 Foc	us 📃 🖸 📋 🖬 🏮 👻 😻 🗸	B
🖃 Projects	■ @ • \ 7 • \= •			BI Publisher Integration	Using ADF
🕀 🛅 BipInter	New				osing Abi
⊞ <mark>⊡</mark> BipWeb	Edit Pr <u>oj</u> ect Source Paths		,		
	🔀 De <u>l</u> ete Project			🔺 Region	
	Version Project			This region cannot be rendered at design time.	
	Find Project Files				
	<u>S</u> how Classpath				
	Sho <u>w</u> Overview		,		
	Deplo <u>v</u>		•	<u>1</u> BipClient-taskflow to JAR file	
	Find <u>U</u> sages		Ctrl+Alt-U	<u>B</u> ipClient-taskflow	
	🚵 Ma <u>k</u> e BipInterface.jpr		Ctrl-F9	<u>N</u> ew Deployment Profile	
	Re <u>b</u> uild BipInterface.jpr		Alt-F9		
	▶ <u>R</u> un				
	🌞 <u>D</u> ebug				
	Refa <u>c</u> tor		•		
Application F Data Contro	Co <u>m</u> pare With		•		
🗉 🛃 BipWrap	Replace Wit <u>h</u>		•		
	Restore <u>f</u> rom Local History				
	Refresh <u>A</u> DF Library Dependenci	es in BipInterface.jpr			
+ Recent Files	Conver <u>t</u> to Facelets				
	Q Project Properties				
BipInterface.jp	or - Structure × 🖃	star	t i		

Figure 5 - Exporting BI Publisher task flow

Consuming Application

What was already done for you

The application *BIPublisherApp* contains a single page *runBIP.jsf* based on the three-column template from Oracle. The page is prepared to integrate the BI Publisher task flow produced in the base application from previous step.

What still has to be done

(1) In this step we will add the library with the ADF task flow to the *ViewController* project of the application. Because the library is a Java Archive file (jar) we first need to create a file system connection in JDeveloper.

The library *BipClient-taskflow.jar* contains all the necessary functionality to call existing BI Publisher reports using BI Publisher web services to

- » Login the specified user
- » Querying the existing reports for that user
- » Providing list of values for the available formats and parameters for that report
- » Executing the report.

(1.1) First it is necessary to add the BI Publisher library to the *ViewController* project of the consuming application.

Open the application *BIPublisherApp* in JDeveloper.

If not already visible open the Window *Resources* by selecting *Window* => *Resources* from the top menu of JDeveloper.



Figure 6 - Window Resources

(1.2) Create a new File System Connection by open the New Gallery with *File => New => From Gallery* from the top menu of JDeveloper.

Select *General* => *Connections* at the left side of the New Gallery dialogue. Select *File System Connection* at the right side of the New Gallery dialogue.



Figure 7 - Create file system connection

Enter the necessary properties for the connection.

Property	Value
Connection Name	BIPublisherDir
Directory Path	<your_working_dir>\Resources</your_working_dir>

Test the connection and accept these entries by clicking OK.

Create File System Connection 🛛 🕹 🗙
Choose Application Resources to create a file system connection owned by and deployed with the current application (BipClientApplication.jws). Choose IDE Connections to create a connection that can be added to any application.
Create Connection in: IDE Connections
BIPublisherDir
Directory Path: pment/JDev_mywork/BIPublisherAppOOW/JDev1213_Neu/Resources
Test Connection Status:
Success!
Help OK Cancel

Figure 8 - Enter file system connection details

Opening the new created file system connection *BIPublisherFolder* a file *BipClient-taskflow.jar* is visible. This jar file contains an ADF task flow *bip-taskflow* which was developed in the base application and is provided as a reusable asset.

Components	Resources ×	
😭 - 🔍 Nam	e	
🗄 My Catalogs		
∃ IDE Connection	ns	
🖽 📲 Application	n Server	
🗄 🖬 Database		
	m ichorDir	
Er Birubii	Client-taskflow	iar
ora	logo small.gif	,
url		
		^ ^

Figure 9 - Task flow in ADF library

(1.3) Add the library file *BipClient-taskflow.jar* to the *ViewController* project:

Select the ViewController project in the Application Navigator

Select the library jar file *BipClient-taskflow.jar* in the window *Resources* and execute the option *Add to Project* from the context menu.

Confirm the alert that you want to add the library to the ViewController project.

Bie Ede Yuw Aselkaten Refigure Search Novigete Buld Bun Teim Tool Yudow Hele Constructions Applications Provects		Oracle JDeveloper 12c - BIPublisherApp.jws : ViewController.jpr	_ 🗆 X
Applications Appli	<u>File Edit View Application Refactor Search</u>	<u>N</u> avigate <u>B</u> uild <u>R</u> un Tea <u>m</u> <u>T</u> ools <u>W</u> indow <u>H</u> elp	
Applications Projects Imagine down of the second	🚯 🕒 - 🖸 🗐 🗐 🔍 🚱 - 🚱	 - A M M M P · A · A 	Q- Search
Application Resources Data Controls Data Cont	Applications ■ BiPublisherApp ■ Projects ■ College WewController		Resources
BipClienttaskflow.jar - Structure BipClienttaskflow.jar - Structure BipClienttaskflow.jar - Structure Messages - Log Live Issues BipClienttaskflow.jar - Issues Messages - Log Live Issues BipClienttaskflow.jar - Issues Messages - Log Live Issues BipClienttaskflow.jar - Issues Messages - Log Live Issues BipClienttaskflow.jar - Issues Messages - Log Live Issues BipClienttaskflow.jar - Issues Messages	Application Resources Date Controls Recent Files		Add to Project. Add to Project. Bemove from Project Leite Delet
Messages -Log Live issues: BipClient-taskflow jar - issues Image: Comparison of the issues: BipClient-taskflow jar - issues No Structure Feb 12, 2015 8:43:04 Mi oracle.security.jps.util.jpsUtil disableAudit Image: Comparison of the issues: BipClient-taskflow jar - issues No Structure Feb 12, 2015 8:43:04 Mi oracle.security.jps.util.jpsUtil disableAudit Image: Comparison of the issues: BipClient-taskflow jar - issues No Structure Feb 12, 2015 0:65:43 Mi oracle.security.jps.util.jpsUtil disableAudit Image: Comparison of the issues: BipClient-taskflow jar - issues Feb 12, 2015 10:05:19 Mi oracle.security.jps.util.jmspector.multi.MultiObjectModel_updateProperties Image: Comparison of the issues: BipClient-taskflow jar - issues No: SelectionModel has no selected items Feb 12, 2015 10:23:54 Mi oracle.bali.imspector.TransactedPIMultiModel_updateProperties News/menge/Development/Dev mywork/BIPublisherApp/NewController/NewController.jpr	≗ BipClient-taskflow.jar - Structure × ⊂		Processes
home/imenge/Development/Dev mywork/BlPublisherAppO0W/Dev1213 Neu/BlPublisherApp/ViewController/ViewController jpr	No Structure	Messages-Log Lve Issues: BipClenttaskflow jar-Issues Feb 12, 2015 6:43:04 AM oracle.security.jps.util.jpsUtil disableAudit TMF0: jpsUtil isAuditDashed set to true Feb 12, 2015 6:43:10 AM oracle.security.jps.util.jpsUtil disableAudit TMF0: jpsUtil isAuditDashed set to true Feb 12, 2015 0:43:10 AM oracle.security.jps.util.jpsUtil disableAudit TMF0: Splettil isAuditDashed set to true Feb 12, 2015 0:43:40 and oracle.ball.inspector.multi.MultiDbjectModel _updateProperties FMF0: Splettil Dis25 54 AM oracle.ball.win_outbas.inspector.TransactedPIPWltHX0del _updateProperties FMF12, 2015 0:23:54 AM oracle.ball.win_outbas.inspector.TransactedPIPWltHX0del _updateProperties	
	/home/jmenge/Development/JDev_mywork/BIPublish	erAppOOW/jDev1213_Neu/BiPublisherApp/ViewController/ViewController.jpr	·

Figure 10 - Adding ADF library to the ViewController project

(1.4) Save all by pressing the floppy disks symbol in the top menu of JDeveloper.

(2) In this step we will integrate the ADF task flow from the library in the page. At runtime the task flow will be executed in a region as part of the page. Because the task flow is parameterized we have to provide values for these parameters.

(2.1) Follow these steps

Open the page runBIP.jsf in the workspace in Design View.

Select the window Resources.

Drag&drop the *bip-taskflow* into the facet center of the page.

Select the option Region to display the task flow as a static region inside the page.



Figure 11 - Drag&Drop BIP task flow into the page

In the dialogue *Edit Task Flow Binding* we have to provide values for the mandatory parameters of the task flow. Enter the following values:

Name	Value	Comment
BipAdminUserName	<admin name="" user=""></admin>	BI Publisher administrator
BipAdminPassword	<admin password=""></admin>	Password of administrator
BipUser	<bip name="" user=""></bip>	BI Publisher user. We use a static value here but it can be derived from the security context of the current user.
BipFolderPath	<bip folder=""></bip>	Bl Publisher folder

The example actually utilizes BIP *impersonate* capabilities, as only the administrator credentials are authenticated, and once successful, admin impersonate the privileges of the user defined in parameter BipUser to fetch the reports he has access too.

	Edit Task Flow Binding	X
Task Flow: /W Input Parameters Map:	/EB-INF/TaskFlows/bip-taskflow.xml#bip-taskflow	~
Input Parameters		
Name	Value	-
BipAdminUserName *	administrator	\sim
BipAdminPassword *	weblogicl	
BipUser *	appuser	
BipFolderPath *	/Samples/Human Resources	
* = Required Help	OK Can	icel

Figure 12 - Providing parameters for the BIP task flow

As a result a region is created inside the page but cannot be rendered at design time because the content is provided by the ADF Task Flow from the reusable library.

■ runBiP.jsf ×	
🚯 🗸 Design True 🔲 Focus 🛄 🔲 🔲 💶 🗍 🕶 🥼	A
BI Publisher Integration Using ADE Task Flow	status 👝
Start Totas a consider on each of the construction of the con	end
Design Source History Bindings	\rightarrow

Figure 13 - Region inside the page

(2.2) Save all by pressing the floppy disks symbol in the top menu of JDeveloper.

Run and test the ADF Application

(1) In this section we will run and test the application using the integrated WebLogic Server in Oracle JDeveloper.

(1.1) To see what happens open the window *Processes*. If not already visible open the window *Processes* by selecting *Window* => *Processes* from the top menu of JDeveloper. Normally there should be no running processes at this time.

Properties	Processes ×	
⊕ - 💽 Proces	ses	

Figure 14 - Window Processes

(1.2) Open the file *adfc-config.xml* in the *Diagram View*. Select the view *runBIP* and choose the option *Run* from the context menu.

Oracle JDeveloper 12c - BIPublisherApp.jws : \	/iewController.jpr : /home/jmenge/Development/JDev_myw	ork/BIPublisherAppOOW/JDev1213_Neu/BIP	ublisherApp/Vie	wController/p	ublic_html/V	VEB-INF/ad 🔄 🗌	
<u>File Edit View Application Refactor Search</u>	<u>Navigate Build Run Diagram Team Tools Window H</u> elp						
🚓 🕒 - 🗁 🗐 💷 🥬 🔍 🔾 - 🕥 -	* 🔒 * 🎄 😫 🚜 🕨 * 🌞 *				Q+(Search		
Applications ×	runBIP.jsf × 🔀 adfc-config.xml ×			Components ×	Resources	1	
🔁 BIPublisherApp 🔹 💌	🍳 🔍 100% 🔷 Show - 🐧 🐴 🖳 😓 🔂 👪	1	::::::::::::::::::::::::::::::::::::::	Q* (0
🖃 Projects 💽 🍓 • 🍸 • 💯 •	Unbounded Task Flow			ADF Task Flow			-
I Model					nts		
ViewController Application Sources				⊒ Components			
B- Web Content				Activities			- 6
E WEB-INF				500	٠		
adte-config.xml				Method Call	Router	Save Point	
				mta	1	inc store	
web.xml	(m			Task Flow	IIPI View	View	
		p. II		Call			
🖃 Data Controls 🛛 🔞 🍸 💯 •		Insert Before View		Control Flow			
🗷 - 🛃 BipWrapper		Insert After View			4		-
	runs	Surround With		Control	Wildcard		-
± Recent Files		Refactor >		🗈 Diagram Anno	tations		
adfc-config xml - Structure × Thumbnail		Cut Ctrl-X		view runPID	Dreportion	Brazana v	-
95 dB		Ctri-C			rioperdes	Processes ~	
ADE Task Flow		Paste Ctrl-V	Ŭ				
i⊞-i view - runBIP		X Delete Delete		Processe	15		
		Show Thumbnail					
		Expand Thumbnail					
		Go to Source					
	Diagram Overview Source History	Go to Properties Go to Declaration Ctri-Period	×				
	Messages - Log × Live Issues: adfc-config.xml - Issues	Tongla Breaknoint ES					
	TED 12, 2010 11.07.20 APT GRACE Dati Inspector Mutti Put	A set all all all all all all all all all al	^				
	Feb 12, 2015 11:57:38 AM oracle.bali.inspector.multi.Mu	Make Ctrl+Shift-F9 Alt+Shift-F9					
	INFO: SelectionModel has no selected items Feb 12, 2015 11:57:52 AM oracle.bali.inspector.multi.Mul	Ruo					
	INFO: SelectionModel has no selected items Eab 12, 2015 11:58:08 AM oracle bali increator multi Mul	2 Debug					
	INFO: SelectionModel has no selected items	Mark Arthuby	0				
	10	Unmark Activity					
/home/jmenge/Development/JDev_mywork/BIPublish	erAppOOW/IDev1213 Neu/BIPublisherApp/ViewController/public html/	🖷 Extract Task Flow					

Figure 15 - Run page from adfc-config.xml

(1.3) View in the *Processes* window.

First the integrated WebLogic Server will be started.

After WLS was succesful started our application will be deployed and started.

Properties	Processes ×	
BIP	ses :gratedWebLogicServer ublisherApp (Running on IntegratedWebLogicServer)	
+)	•

Figure 16 - Processes window after starting application

(1.4) A browser window should be opened with our page. During the next steps we will stay on this page. Only the content of the embedded region is updated by a partial page refresh in the browser.

The first list-of-values for the Report Name displays the available reports from the specified folder.

When a report is selected the second list-of-values for the *Template* will be updated automatically to show the templates which are available for that report.

When a template is selected the third list-of-values for the *Format* will be refreshed automatically to show the formats which are allowed for that template.

If the selected report has associated parameters they will displayed too.

Every operation in the page triggers a web service call to the BI Publisher server, the delivered response will be displayed in the region.

ORACLE	BI Publisher I	ntegration Usir	ng ADF Ta	ask Flow	0
	Reports Report Name Template Format Run	Salary Report Financial Style PDF	¥ ¥ ¥	Report Parameters Department Employee *	1
			_		

Figure 17 - Selection of report, template, format and report parameters

(1.5) Now the report can be generated by pressing the green arrow button Run.

The report output will be produced on the server and downloaded to the client.

Depending from the selected format, the browser settings and your choice the result will be displayed in the browser (for example HTML), displayed using a helper application (for example PDF) or saved in the file system.

Salary Report - Adobe Reader											
File Edit View Document Tools Window Help											
Salary Report 🗵											
1 🗗 📋 🔬 -	· 💱 🛧 🔶 🛽	/ 4 💿 💌 83,69 - 🥁 🙀	Find	•							
						- 88					
						- 88					
						- 88					
				Brought Forward	.00	- 88					
						- 88					
	Employee Salary Report										
	Name	Job Title	Manager	Department Name	Salary						
	Sundita Kumar	Sales Representative	Gerald Cambrault	Sales	6 100 00						
	Elizabeth Bates	Sales Representative	Gerald Cambrault	Sales	7.300.00						
	William Smith	Sales Representative	Gerald Cambrault	Sales	7,400.00						
	Tayler Fox	Sales Representative	Gerald Cambrault	Sales	9,600.00						
	Harrison Bloom	Sales Representative	Gerald Cambrault	Sales	10,000.00						
	Lisa Ozer	Sales Representative	Gerald Cambrault	Sales	11,500.00						
	Matthew Weiss	Stock Manager	Steven King	Shipping	8,000.00						
	Adam Fripp	Stock Manager	Steven King	Shipping	8,200.00						
	Payam Kaulling Shanta Vollman	Stock Manager	Steven King	Shipping	7,900.00						
	Kevin Mourgos	Stock Manager	Steven King	Shipping	5,800.00						
	Gerald Cambrault	Sales Manager	Steven King	Sales	11 000 00						
	Eleni Zlotkev	Sales Manager	Steven King	Sales	10,500.00						
	John Russell	Sales Manager	Steven King	Sales	14,000.00						
	Karen Partners	Sales Manager	Steven King	Sales	13,500.00						
	Alberto Errazuriz	Sales Manager	Steven King	Sales	12,000.00						
	Den Raphaely	Purchasing Manager	Steven King	Purchasing	11,000.00						
	Michael Hartstein	Marketing Manager	Steven King	Marketing	13,000.00						
	Neena Kochnar	Administration Vice President	Steven King	DOAG	17,000.00						
	Hermann Baer	Sales Representative	Neena Kochhar	Public Relations	10,000,00						
	Susan Mavris	Human Resources Representative	Neena Kochhar	Human Resources	6,500.00						
	Nancy Greenberg	Finance Manager	Neena Kochhar	Finance	12,000.00						
	Jennifer Whalen	Administration Assistant	Neena Kochhar	Administration	4,400.00						
	Shelley Higgins	Accounting Manager	Neena Kochhar	Accounting1	12,000.00						
	Alexander Hunold	Programmer	Lex De Haan	IT	9,000.00						
	David Austin	Programmer	Alexander Hunold		4,800.00						
	Valli Pataballa	Programmer	Alexander Hunold		4,800.00						
	Diana Lorentz	Programmer	Alexander Hunold		4,200.00						
	Diruce Ellist	riogrammer	Alexandel Hunolu	11	0,000.00						
	-			Total Salami cr. Dorro	381,000,00						
				Carry Ecoward	-: 201,000.00						
-				Carry Porward	. 201,000.00						
- Of -											

Figure 18 - Generated report (PDF)

Summary

This example shows how to integrate BI Publisher reporting capabilities in ADF applications. It especially shows the power of ADF task flows. The *bip-taskflow* completely encapsulates the required functionality to call BI Publisher web services. The advantages of this approach are obvious:

- » The **developer for the model** (data control) provides an interface to BI Publisher server. He needs to know the web service API of BI Publisher and is responsible for the implementation.
- » The **UI developer** can simply consume the task flow in his application and does not need to know the underlying web service implementation.
- » The **task flow** can be used in all ADF applications which require this reporting functionality by simply adding the task flow as a library. Through parameterization the task flow can be reused in a very different context.

There are many ways this example can be extended:

- » Specific **report parameters** like employee id or department number are implemented in the example as input text fields. It would be more comfortable to offer a list-of-values to the end user to select the values from. There are appropriate web service methods available to fetch this information from BI Publisher server
- » There could be additional **scheduling parameters** to deliver a report immediately or at a certain time using mail or a printer.
- » There are very specific **report formats** (PDF/Z and interactive) which are not supported in the example. It could also be useful to provide a **file extension** along with the file name when generating a report to allow saving the report as a file.



CONNECT WITH US

blogs.oracle.com/oracle

facebook.com/oracle

twitter.com/oracle

oracle.com

Oracle Corporation, World Headquarters 500 Oracle Parkway

Redwood Shores, CA 94065, USA

Worldwide Inquiries Phone: +1.650.506.7000 Fax: +1.650.506.7200

Hardware and Software, Engineered to Work Together

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.0115

Integrating Oracle BI Publisher Reports into Oracle ADF Applications March 2015 Authors: Tamer Qumhieh, Jürgen Menge Contributing Authors:

