NetBeans 6.5.1, GlassFish v 2.1, Web Space Server 10 **Creating a Healthcare Facility JSR286-compliant Portlet**

Michael.Czapski@sun.com June 2009

Abstract

SOA is sometimes shown as a series of 4 layers with Presentation Layer (SOA 1) at the top. Each layer consumes services exposed by layers under it. Service interfaces are described using WSDL. Web Services are the means to decouple functional layers. Functionality in one layer can be swapped in and out without disturbing other layers. Presentation layer is often implemented as JSR-168-compliant or JSR-286-compliant Portlets, exposed through a standards-based Portal.

Here I will walk through development of a JSR-286-compliant Visual Web JSF Portlet, which will use a Web Service as a data provider. I use the NetBeans 6.5.1 IDE, part of the GlassFish ESB v2.1 installation, the Portal Pack 3.0.1 NetBeans Plugin and the JSF Portal Bridge provided by the Web Space Server 10. The Portlet will use JSF components provided by Project Woodstock. The technology will be introduced in a practical manner.

This is not a tutorial on JavaServer Faces, Visual Web JSF, Project Woodstock or Portlet development.

Introduction

In some views SOA is represented as a series of 4 layers: Presentation Layer (SOA 1), Business Process Layer (SOA 2), Business Service Layer (SOA 3) and Technical Layer (SOA 4). Typically each layer higher up in the hierarchy consumes services exposed by the layer under it. So the Presentation Layer would consume services provided by the Business Process or Business Service Layers. Service interfaces are described using Web Services Description Language (WSDL), sheltering service consumers from details of service implementation. Web Services are seen as the technical means to implement the decoupled functional layers in a SOA development. Decoupling allows implementations of business functionality at different layers to be swapped in and out without disturbing other layers in the stack. The SOA 1, Presentation Layer, is often implemented as JSR-168-compliant or JSR-286-complaint Portlets, exposed through a standards-based Portal.

The business idea is that patients are looked after in various healthcare facilities. Applications need to allow selection of a facility and to access facility details for display to human operators. A relational holds details of facilities which are a part of the healthcare enterprise. Facility list and details are available through a web service. This web service will be used to construct the JSR-286-comliant Portlet that provides a user view into the facilities and facility details. This Portlet will be deployed to the Sun FOSS Web Space Server 10 Portal.

Previous documents in this series, "GlassFish ESB v 2.1 - Creating a Healthcare Facili ty Web Service Provider", at <u>http://blogs.sun.com/javacapsfieldtech/entry/glassfish_es</u> <u>b_v_2_1</u> and "NetBeans 6.5.1 and GlassFish v 2.1 - Creating a Healthcare Facility Visual Web Application", <u>http://blogs.sun.com/javacapsfieldtech/entry/netbeans_6_5_1_and</u>, walked the reader through the process of implementing a GlassFish ESB v2.1-

based web service which returns facility list and facility details, and a Visual Web JSF Web Application which used that Web Service to display facility list and details.

In this document I will walk through the process of developing a JSR-286-compliant Visual Web JSF Portlet, deployed to the Sun Web Space Server 10 Portal, which will use the Web Service as a data provider. We will use the NetBeans 6.5.1 IDE, which comes as part of the GlassFish ESB v2.1 installation, the Portal Pack 3.0.1 NetBeans Plugin and the JSF Portal Bridge infrastructure provided by the Web Space Server 10. The Portlet will be implemented as a Visual Web JavaServer Faces Portlet using JSF components provided by Project Woodstock. The Portlet will introduce the technology in a practical manner and show how a web service can be used as a data provider, decoupling the web application from the data stores and specifics of data provision.

Note that this document is not a tutorial on JavaServer Faces, Visual Web JSF, Project Woodstock components or Portlet development. Note also that all the components and technologies used are either distributed as part of the NetBeans 6.5, as part of the GalssFish ESB v2.1, as part of the Web Space Server 10 or are readily pluggable into the NetBeans IDE. All are free and open source.

Prerequisites

To work through this material certain pre-requisites have to be met.

The GlassFish ESB v2.1 installation must be available. Install GlassFishESB v2.1, <u>https://open-esb.dev.java.net/Downloads.html</u>, following standard installation steps detailed at the OpenESB site.

Sun Web Space Server 10 functionality must be added to the GlassFish ESB v 2.1 installation. This is discussed in "Adding Sun WebSpace Server 10 Portal Server functionality to the GlassFish ESB v2.1 Installation", <u>http://blogs.sun.com/javacapsfield tech/entry/adding sun webspace server 10</u>. This implies availability of the MySQL RDBMS installation, also discussed in that document.

To allow web services and portlets to coexist in the same GalssFish Application Server instance the infrastructure must be configured with an alternate portal servlet context root. This is discussed in "Making Web Space Server And Web Services Play Nicely In A Single Instance Of The Glassfish Application Server", <u>http://blogs.sun.com/javacapsfieldtech/entry/making_web_space_server_and</u>.

The multi-operation web service, which provides faculty list and facility details, must be developed and deployed. This is discussed in "GlassFish ESB v 2.1 - Creating a Healthcare Facility Web Service Provider", at <u>http://blogs.sun.com/javacapsfieldtech/e</u> ntry/glassfish_esb_v_2_1.

Determining Service Endpoint and WSDL Location

This document assumes that the portlet will use the web service developed elsewhere as a data provider. To make it possible we need to know the endpoint location of the service and the location of the WSDL. This information is available if one knows where to look.

Let's open the FacilitySvc.wsdl document in project FacilitySvc_BM and inspect the properties of the soap:address node under the FacilitySvcService node.



Note the Location property value:

http://localhost:\${HttpDefaultPort}/FacilitySvcService/FacilitySvcPort

The HttpDefaultPort is the port which SOAP/HTTP BCs use. At CA deployment time this variable gets replaced with the actual port. To find out what this port is let's switch to the Services tab in Netbeans, expand Servers, expand JBI, expand Binding Components, right-click sun-http-binding and choose Properties.



Observe the Default HTTP Port Number property value. For my installation this will be 29080. For a default installation it will be 9080. It can be changed.

🇊 sun-http-binding - Properties		×
General		
Description	HTTP binding component. Provides mes	
Name	sun-http-binding	
State	Started	
Туре	binding-component	
Identification		
Version	2.5.0	
Build Number	090515	
Configuration		
Number of Outbound Threads	10	
Number of Inbound Threads	100	
Number of InboundReply Threads	100	
Default HTTP Port Number	29080	
Default HTTPS Port Number	29181	
(SSL)Client authentication enabled		
Sun Access Manager Configuration Directory		
Sun Access Manager Classpath		
Proxy Type	DIRECT	•
Proxy Host		
Proxy Port	0	
Non-proxy Hosts	localhost 127.0.0.1	
Drovy Heer Name		
sun-http-binding		0
HTTP binding component. Provides message p JBI 1.0 compliant environment.	processing capabilities over HTTP protocol in	ıa
		_
	Close	

So, the final service endpoint URL, from the soap:address Location property earlier, will be:

http://localhost:29080/FacilitySvcService/FacilitySvcPort

This URL is the service location.

The WSDL for this service can be accessed, using the regular convention, at:

http://localhost:29080/FacilitySvcService/FacilitySvcPort?WSDL

With this knowledge we can use the service in related projects.

Making Service Available for use in a Portlet

If you created a Web Application, discussed in "NetBeans 6.5.1 and GlassFish v 2.1 - Creating a Healthcare Facility Visual Web Application", <u>http://blogs.sun.com/javacapsfieldtech/entry/netbeans 6 5 1 and</u>, the web service reference will already be available for use. You can skip this section. If not, follow this section to get the service in the right place in NetBeans.

To make a web service available for use as a data provider in a web application, or a Portlet, we must "introduce" it to the NetBeans IDE. Switch to the Services Tab, rightclick on the Web Services node and choose Create Group.



Name this group "__HC Services".

Right-click on the __HC Services node and choose Add Web Service...



Enter service WSDL URL and click OK. The service WSDL URL for me will be http://localhost:29080/FacilitySvcService/FacilitySvcPort?WSDL

🇊 Add Web Se	rvice	×
Specify the file	of the Web Service.	
O Local File:		Browse
⊙ <u>U</u> RL:	http://localhost:29080/FacilitySvcService/FacilitySvcPort?WSDL	Set <u>P</u> roxy
Package Name:	<click override="" to=""></click>	
	ОК	Cancel <u>H</u> elp

Expand ___HC Services -> FacilitySvcPort, wait for the WSDL to be loaded and appropriate classes to be generated, compiled and packaged.

🕴 🕀 💼	Zvents
ė- 🗅	HC Services
Ė	FacilitySvcPort
	🛄 🔟 Loading
l 🗄 🚳 🛌	n (n)

Expand the node tree all the way to the operations, right-click on opFacList and choose Test Method.

🖃 🧰HC Services	
🗄 🖳 🕵 FacilitySvcPort	
🖻 🔁 FacilitySvcPort	
🍖 opGetFacDe	tails
🔤 🖬 opGetfacList	L
Enterprise Beans (2.x)	Test Method
🗄 📲 Servers	

Click Submit button and, when execution is completed, inspect the results.

🇊 Test Web Service Method				×
	Test Web Ser	rvice Method: ers and press the "Submit" button.		
Туре	Na	ame	Va	alue
🗆 🗁 sun.michael_czapski.xsds.facility.FacLi	msgFacListReq		sun.michael_czapski.xsd	ds.facility.FacListReq
java.lang.String	dummyString			
			_	
ļ		_ /	10 C	
	Sub	omit 🖌		
Results:				
Туре			Value	
🗆 🖂 sun.michael_czapski.xsds.facility.FacList	Res	sun.michael_czapski.xsd	s.facility.FacListRes@a6	4a19
□ 🗁 java.util.List <sun.michael_czapski.xsds< td=""><td>.facility.FacListRes.Fa</td><td>[sun.michael_czapski.xs</td><td>ds.facility.FacListRes\$Fa</td><th>cList@134624b, sun</th></sun.michael_czapski.xsds<>	.facility.FacListRes.Fa	[sun.michael_czapski.xs	ds.facility.FacListRes\$Fa	cList@134624b, sun
sun.michael_czapski.xsds.facility.Fa	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@134624b
sun.michael_czapski.xsds.facility.Fa	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@2155a0
sun.michael_czapski.xsds.facility.FacListRes.FacList		List@7cadb4		
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@170433f
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@14ee504
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@9d7383
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@10aae2f
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@31b105
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@5cd1d3
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@1d39194
sun.michael_czapski.xsds.facility.Fac	ListRes.FacList	sun.michael_czapski.xsd	s.facility.FacListRes\$Fac	List@fa9dec

Now execute the test for the opGetFacDetails, providing facility code of STC.

🇊 Test Web Service Method			×
§r	Test Web Ser	rvice Method: ers and press the "Submit" but	bn
Туре	Na	ame	Value
🗆 🗁 sun.michael_czapski.xsds.fa	msgFacDetailsReq		sun.michael_czapski.xsds.facility.Fa
java.lang.String	facCode		STC
]			
	Sub	omit	
<u>R</u> esults:			
Туре			Value
💷 🗁 sun.michael_czapski.xsds.facili	ty.FacDetailsRes	sun.michael_czapsk	i.xsds.facility.FacDetailsRes@1e7a6b1
java.lang.String		STC	
java.lang.String		SYDNEY TECHNICA	L HOSPITAL
java.lang.String		404 HUNTINGTON [DRIVE
java.lang.String		MONROVIA	
java.lang.String		CA	
java.lang.String		91016	
java.lang.String		USA	
java.lang.String		A	
			<u>Close</u> <u>H</u> elp

This is yet another method of testing web services in NetBeans. We have valid references to service operations, ready to be used in web applications and portlets.

Add Web Space Server 10 to the list of Servers

Before we get to create a Portlet we must add the Web Space Server to the list of Servers in the Services Tab.

Switch to the Services Tab in NetBeans IDE, right-click Servers and choose Add Server.



Choose WebSpace 10.x (Porject WebSynergy) and click Next.

TAdd Server Instance		×
Steps	Choose Server	
1. Choose Server 2	Server: Sun Java System Portal Server 7.0/7.1 (Updates) Sun Java System Portal Server 7.2 Tomcat 5.0 Tomcat 5.5 Tomcat 6.0 WebSpace 10.x (Project WebSynergy)	×
	Name: WebSpace 10.x (Project WebSynergy)	
	< Back Next > Finish	Cancel <u>H</u> elp

Locate the GlassFish Home directory (this will be the GlassFish instance to which you added WebSpace Server functionality).



Once you have chosen the right directory all other details will be filled in, except admin password. Provide the admin password and click Next.

Choose Server	GammaTama	GlassEish / Sun Java S	vstem AppServer 9.x	1
	Serveriype			-
	GlassFish Home:	G:\GlassFishESBv21\glassf	îsh	
	Domain Dir:	G:\GlassFishESBv21\glassf	îsh \domains \domain 1	1449 T
	Domain:	domain 1		
	User Name:	admin		
	Password:	*****		
	Port:	28080		
and a	Admin Port:	24848		

Recall that in one of the pre-requisites we changed the servlet context root for the portal. We must reflect this in the service we are adding. Change the Portal Context from "/" to "/wss", or whatever you used for the portal context. Finish the wizard.

. Choose Server		WebSpace 10.x (Project WebSynergy)	
	Host:	localhost	
	Portal Context :	/wss	
	Auto Deploy Dir:		Get
	Portal Deploy Dir:	Jomains\domain1\applications\j2ee-modules\webspace	
· Tas	Directory Deploy	ment Enabled	
\sim			

This adds the WebSpace 10.x server to the list. This server is where we will be deploying the portlet we will create shortly.

Create Visual Web JSF Portlet

Of the variety of methods available in NetBeans to develop portlets I have chosen the Visual Web JSF Portlet method. The reason is that it is visual, easy and quick, if one knows what one is doing. I know enough to be dangerous but not enough to help you out if you get into trouble with JSF, Visual Web JSF or Portlet design. There is a plethora of material on the Internet on the different aspects of these technologies. Do what I do – research and experiment. This is a practical cookbook for the specific portlet I built. Feel free to learn the technology and ad-lib.

Switch back to the Projects tab.

Create a New Project -> Java Web -> Web Application, FacilitySvcVWJSFP.

Steps	Name and Location	
1. Choose Project 2. Name and Location	Project Name: FacilitySvcVWJSFP	
8. Server and Settings 8. Frameworks	Project Location: G:\GlassFishESBv21Projects\FacilitySvcProjGrp	Browse
., realizatione	Project Eolder: G:\GlassFishESBv21Projects\FacilitySvcProjGrp\FacilitySvcVW	/JSFP
	libraries (see Help for details). ✓ Set as Main Project	
	Profe Navet Same Same	Survey Hole

Choose WebSpace 10.x as Server and accept defaults for other settings in the panel.

Steps	Server and Settings
 Choose Project Name and Location Server and Settings Frameworks 	Add to Enterprise Application: <none></none>
	Java EE Version: Java EE 5
. 102	Context Path: //FacilitySvcVWJSFP
<u>~~~</u>	

Choose portlet support but do not create a portlet.

Choose Project Name and Location Server and Settings	Select the frameworks you	want to use in your w				
Name and Location Server and Settings		Select the frameworks you want to use in your web application.				
	CMS FrameWork					
Frameworks	Portlet Support					
	Simple API For Workf	flow(SAW) Support				
	Visual Web JavaServ	er Faces				
	J T C Shrind Web MVC 7-5	C				
	Portlet Support Configuratio	n				
		59/0				
	Portlet Version:	2.0 🔻	Create Portlet	🔽 Greate Jsps		
	Source Root:			7		
	Package:	com.test				
	Portlet Class Name:	FacilitySvcVWJSF	p			
	Portlet Namer	Esellitu@ueA0W.1CE	P			
	For Dectivatile.	r-BeintySvevvv33r	r.			
	Portlet Display Name:	FacilitySvcVVUSF	P			
	Portlet Description:	FacilitySvcVWJSF	P			
	Portlet Title:	FacilitySvcVWJSF	P			
	Portlet Short Title:	FacilitySv¢VWJSF	P			
	Portlet Mode:	View	🔽 Edit	🔽 Help		
		600		Read 2		

Close index.jsp - we will not be using it.



Right-click on the project name and create New -> Visual Web JSF Portlet Page ...



Name the page FacilityLookupPortlet, provide Portlet Name (which must be an Identifier) and modify description and title text if you feel so inclined.

steps	Name and Location		
. Choose File Type . Name and Location	File Name: FacilityLookup	Portlet	
	Project: FacilitySvcV	NJSFP	
	Folder: web	Brows	e
	Created File: FishESBy21	Projects/EacilitySycProjGrn/EacilitySycVW1SEP/weh/EacilityLookupPor	tlet
	7		
	Default Java Package:	facilitysvcvwjsfp	
		Greate a New Portlet	
	Portlet Name:	FacilityLookupPortlet	
	Portlet Display Name:	Facility Lookup Portlet	
	Portlet Description:	Facility Lookup Portlet	
	Portlet Title:	Facility Lookup Portlet	
1	Portlet Short Title:	Facility Lookup Portlet	

Finish.

Change the Height property of the portlet to 200px. We can get away with less still. Experiment.

FacilityLookupPortlet - Properties				
General				
id	FacilityLookupPortle	t		
Appearance				
Page Layout	Grid Layout	-		
Width	400px			
Height	200px)			

Drag the Woodstock Basic -> Label component onto the canvas and change its text property to read "Choose Facility".

FacilityLoo	kupPortlet *	* ×								Palette	
Design	JSP	Java	<u> </u>	2		Any Size			-	<u>W</u> oodstock Basic	_
						******				1999 Label	D
										I Text Field	E
										🕑 Button	6
						· ·				😫 Image Hyperlink	E
: : :	· ·	· ·	: :	:	:	· ·	: :	:	: :	🗐 Listbox	B
										Checkbox Group	0
							· · · · ·			 Radio Button Group 	5
	· · · · · · · · · · · · · · · · · · ·	· · · · · · · ·					· · · · ·			Table	
				••••				• • • •			-
] FacilityLookupPortlet -	Pro
										General	
FacilityLo	ookupPortl	et * ×									
Design	JSP	Jav	a	<u>@</u>	Z						
Choo	se Faci	ility 🖥									
		- <u>-</u>									

Drag Woodstock Basic -> Drop Down List component to the canvas to the right of the label. Change the General -> Id property of the component to ddFacilities.

acilityLoo	kupPortlet	* x	< F - E	Palette	
Design	1SP	lava	Any Size		
o congri		TI	Trut see	Label	A Static Text
hoose	Facilit	y 🗄 🔨		Text Field	Text Area
		1 1 1 1 1 1 1 1 1 1 1		Button	📾 Hyperlink
19 19	11 - 11 11 - 11	411 - 411 411 - 411		Tmage Hyperlink ····	Drop Down List
1			 		



Right-click on the drop down component and choose Auto-Submit On Change.

[FacilityLoo	kupPortlet * ×	
	Design	JSP Java	🐻 🞯 📴 Any Size
	Choos	e Facility 📲	e Edit Java Source
I			Auto-Submit on Change
I			Bind to Data
I			Configure Default Options
	t		Add .

Save the project.

Switch from the Project view to Services view, locate the opGetFacList operation on the web service we added to NetBeans view of web services, drag it onto the FacilityLookupPortlet JSF page canvas and drop it right over the top of the drop down component.



Right-click the drop down component and choose Add Binding Attribute.

FacilityLoo	FacilityLookupPortlet * ×					
Design	JSP	Java	🐼 😂 💶 Any Size			
Chase						
Choose	e Facilit	у	Edit Java Source			
		: :	Auto-Submit on Change			
			Bind to Data			
			Add			
			Property Bindings			
			Add Binding Attribute			
		· ·	Edit JSP Source 😽			

Right-click the drop down component and choose Bind to Data.



Make sure facCode is selected in the Value field list and description is selected in Display field list, the click OK.

🇊 Bind to Data - ddFacilities	×
Current Items property setting	
#{FacilityLookupPortlet.facilitySvcPortOpGetfacList1.options['facCode	e,description']}
Bind to Data Provider Bind to an Object	
Choose a Data Provider to bind to ddFacilities:	
facilitySvcPortOpGetfacList1 (FacilityLookupPortlet)	T
<u>V</u> alue field:	Digplay field:
<none></none>	<use field="" value=""></use>
description String	description String
facCode String	facCode String
ОК	Cancel <u>Apply</u> <u>Help</u>

Switch to the project view, right-click the name of the project and choose Deploy.

Pro I X Files	Services	FacilityLoc	okupPortlet	t* ×		
⊞ 😤 FacilitySvc_BM	<u> </u>	Design	JSP	Java		🔊 rf
🗄 🖓 FacilitySvc_CA						
E G FacilitySvcVWJSFA		Choos	e Facil	ity _l	_	
FacilitySvcVWJSF	n.			· · · · · · · · · · ·	· · · · · ·	
🖨 🕝 Web Pages	New		•			
😟 💼 WEB-INF	Build					
	Clean and Build			· ·		
	Clean					
index.jsp	Concento Jourd					
🖻 🗔 Configuration Fil	Generate Javau	UC .				
MANIFEST.N	Run					:
faces-config	Deploy				•••••	•••••
🔤 🔛 liferay-displa	Debug				• • • • •	

Once the deployment completes successfully you will see messages similar to the following.



Start your favorite web browser, for example Mozilla Firefox, connect to <u>http://localhost:28080/wss</u> (change port if you use a different one, change portal context if you use different one) and sign in as a portal user. If you have no other user created there is always <u>admin@example.com</u> with password admin.

🥹 Welcome - example.com - Mozilla Firel	fox
<u>File Edit View History Bookmarks T</u>	ools <u>H</u> elp
🔇 🕑 - C 🗙 🏠 💊 h	ttp://localhost:28080/wss/web/guest;jsessionid=000205a1cc4
🙆 Most Visited 🏶 Getting Started <u>N</u> Late	st Headlines
🚸 Welcome - example.com	🔄 🔷 Welcome - example.com 🛛 🔯
Welcome	
admin@example.com	
Password	
Remember Me	
Sign In	
👉 <u>OpenID</u> 🙇 Create Accou	nt 🧕 Forgot Password

Click on Add Page.



Name the page "HC Portlet" and click Save.

	un systems	
Welcome	HC Portlet	✓ Save
🔑 Sign In		_
You are signed	in as <u>Admin User</u> .	

Click on the HC Portlet tab to switch to that page.

	UN osystems		
Welcome	HC PSrtlet		

Pull down the menu and choose Add Application



Expand User Portlets menu item and drag the Facility Lookup Portlet to the canvas.



You may need to right-click on the portlet and choose Reload to get the portlet to display properly.



Our portlet is now visible with no facilities in the drop down.





Note that the drop-down is empty. We need to initialize the call to the web service by providing dummy string value.

Remove the portlet so we can add it again later, once we modified it.

Facility Lookup Portlet	ဓဓဓ္ရု
Choose Facility	

Switch to NetBeans, switch to Java Tab and scroll down to the prerender() method.



Into the prerender method insert the following code.

```
init();
FacListReq flReq = new FacListReq();
```

```
flReq.setDummyString("dummystring");
facilitySvcPortOpGetfacList1.setMsgFacListReq(flReq);
```

Right-click anywhere in the source window and choose Fix Imports, then choose Format. This will add appropriate imports and reformat the source. We added a call to init() method because, unlike in a Visual Web JSF Application, in the Visual Web JSP Portlet it does not get executed. The only method of interest, created by default in the JSF, that is executed is the prerender() method. Init() method contains invocation of various methods generated by the addition of the web service data provider so we must execute it to initialize all there is to be initialized.



Before we deploy the portlet let's switch to the JSP view, locate the line that reads themeLinks, scroll to the far right and remove the string 'webuiAll="true". This is necessary if the web browser is a Firefox, as it is in my case. If we don't do this an obscure and annoying bug is triggered when displaying this portlet in Firefox.

FacilityLoo	kupPortlet	* ×									K	
Design	JSP	Java	10	-	-	ج ک	· 🖓	E 4	ۍ ج		0	
io	n="1.0"	encod	ling=	"UTF-	8"?>							
Ę												
nt	: Fa	cility	Look	upPor	tlet							
d	on : 24	/06/20	09, 3	11:51	:48 A	М						
1.1.1	: mc	zapski										
🛛 🖓 ve	rsion="	2.1" x	mlns	:f="h	ttp:/	/java	.sur	n.com	/jsf/	core	" xm]	lns:h="
p=	"http:/	/java.	sun.	com/j	sf/po	rtlet	/cor	npone	nts"	xmln	s:wel	buijsf=
lir	ective.	page o	onte	ntTyp	e="te	xt/ht	ml;c	chars	et=UI	F-8"	page	eEncodi
<u></u>												
[<u></u> ₽	ortletP	age>										
	<div st<="" th=""><th>yle="h</th><th>eigh</th><th>t: 20</th><th>0px;</th><th>posit</th><th>ion</th><th>: rel</th><th>ative</th><th>wi</th><th>dth:</th><th>400px;</th></div>	yle="h	eigh	t: 20	0px;	posit	ion	: rel	ative	wi	dth:	400px;
		buijsf	:the	meLin	ks bi	nding	="#+	[Faci	lityI	ooku	pPort	tlet.th
	<we< th=""><th>buijsf</th><th>:for</th><th>m bin</th><th>ding=</th><th>"#{Fa</th><th>cili</th><th>ityLo</th><th>okupI</th><th>Portl</th><th>et.fo</th><th>orm1}"</th></we<>	buijsf	:for	m bin	ding=	"#{Fa	cili	ityLo	okupI	Portl	et.fo	orm1}"
		<webu< th=""><th>ijsf</th><th>:labe</th><th>l id=</th><th>"labe</th><th>11"</th><th>styl</th><th>e="le</th><th>ft:</th><th>0px;</th><th>top: 0</th></webu<>	ijsf	:labe	l id=	"labe	11"	styl	e="le	ft:	0px;	top: 0
		<webu< th=""><th>ijsf</th><th>:drop</th><th>Down</th><th>bindi</th><th>.ng='</th><th>#{Fa</th><th>cilit</th><th>YLOO</th><th>kupPo</th><th>ortlet.</th></webu<>	ijsf	:drop	Down	bindi	.ng='	#{Fa	cilit	YLOO	kupPo	ortlet.
		i	tems	="#{F	acili	tyLoc	kupl	Portl	et.fa	cili	tySvo	cPortOp
	<th>ebuijs</th> <th>f:fo</th> <th>rm></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>	ebuijs	f:fo	rm>								



Let's now deploy the portlet again, add it to the page and see what we get.

As before, you may need to reload the page after adding the portlet. Once the portlet displays the facility list will be populated.

acility Lookup	Portlet	0000
Choose Facilit	y A AT A Blu Ne Sp Tre Go ICF NC SY TH	RED MEDICAL CENTRE RED MEDICAL CENTRE BIG HOSPITAL Je Mountains Hospital pean Hospital ringwood Hospital esilian overnor Phillip PMR ORTH SYDNEY MEDICAL CENTRE 'DNEY TECHNICAL HOSPITAL E BIG HOSPITAL

Let's remove the portlet for the next round of changes.

Now that we have the drop down being populated we would like to display details of the selected facility each time facility changes in the dropdown.

Let's switch to Design view, then add 4 labels and 5 static fields to contain descriptions and values of the various facility properties.

Let's add labels for "Facility Code", "Description", "Address" and "Status", select all labels, right-click and choose Align and choose Right.

FacilityLookupPo	rtlet * ×	
Design JSI	P Java 🐼 🐔	🔋 📴 Any Size
Choose Fa	cility	· · · · · · · · · ·
	Edit Java Source	· · · · · · · · · · · · · · · · · · ·
	Bind to Data	
pescription	<u>1</u> E. <u>A</u> dd	: : : : .
Address	Property Bindings.	: : : :
	Edit JSP Source	
	Edit Inline	
		· · · · · · · · · · · · · · · ·
	Select Parent	
	Edit Event Handler	r
	Set Initial Focus	
	Virtual Forms	
	Align	Snap to Grid
	Bring to Front	Тор
	Send to Back	Middle
	Customize	Bottom
		Left
	Cut	Ctrl+X Center
	Сору	Ctrl+C Right
	<u>P</u> aste	C#I+V

Let's now drag 5 static fields and give them ID of stFacCode, stDescription, stAddress1, stAddress2 and stStatus respectively.

FacilityLookupPortlet * ×	♦ ► ■ Palette		
Design JSP Java 😿 🔮	Any Size	dstock Basic	
	Lab	al A Static Text	t
Shoose Facility 🛛 🛄	Geo Нур	erlink 🔠 Image Hyp	perlink
Facility Code Static Te	9Xt 🔃 🚺 Che	ckbox Group 💿 Radio Butt	ion
Description Static Te	ext 🔢 📊 Tab	le Column 🔠 Table Row	Group
Address _Static Te	ext_ 🛛 👔 File	Upload 🛛 🖁 Tree	
Static Te	exter 🔅 🖉 Mes	sage Group	
Status		dstock Layout	
	🛄 Grid	Panel 📴 Group Pan	el
	Tab	🛁 Page Frag	ment Box
	Prop	oerty 🕘 Form	
		dstock Composite	
] stAddr	ess2 - Properties	
	Gener	al	
	id id	stAddress2	

Now right-click on each static field in turn and choose Add Binding Attribute.

FacilityLookupPortlet * >	د]		Palette
Design JSP .	Java 👌	😿 🥺 👜 Any Size	= <u>W</u> oo
Choose Facility			🔛 Labe 📾 Hyp
Facility Code	_ <mark>_</mark> Sta	Fdit Java Source	7 Che
Description	Sta	Bind to Data	Tabl
Address	Sta	Add	File
	Sta	Property Bindings	Mes
Status	Sta	Add Binding Attribute	<u>W</u> oo
	•••••	Edit JSP Source	Grid
		Edit Text	l Tab

This will allow us to manipulate contents of the fields at runtime.

So far nothing happens if we change the value of the ddFacilities drop-down. Let's add a handler to be executed when the value changes. Right-click the drop down component, choose Edit Event Handler and choose processValueChange.

FacilityLookupPortlet ×	:]] Palette
Design JSP	Java	ன 🞯 🟳 Any Size	🖃 <u>W</u> oodstock Basic
			🕫 🔎 Label
Choose Facility	·	Edit Java Source	📾 Hyperlink
Facility Code		Auto-Submit on Change	Checkbox Group
Description		Bind to Data	Table Column
Address		Add	Eile Upload
		Property Bindings	(a) Magazara Crown
	·	Remove Binding Attribute	BP Message Group
Status		Edit JSP Source	- Woodstock Layout
	:	Edit Label	Grid Panel
	: • • _		– 🔚 Tab
	: · · ·	Select Parent	Property
	.	Edit Event Handler	processValueChange
	·	Set Initial Focus	validate

This switches display to the Java source code mode, inserts a skeleton ddFacilities_processValueChange() method and allows us to add custom Java code.



What we need to do here is to look at the new value of the ddFacilty (facility code), use this code to look up facility details and populate facility details-related static fields with the property values for the designated facility.

Let's switch to the Services Tab, locate the opGetFacDetails web service operation and drag it onto the Java canvas inside the ddFacility_processValueChange method.



Accept defaults in the dialog box that results.

🗊 Customize opGetFacDetails Saas Service								
input Parameters:								
Name	Туре	Default Value						
msgFacDetailsReq	sun.michael_czapski.xsds.facility.FacDe	<not set=""></not>						
		OK Cancel						

A slab of boilerplate code will be added. I re-formatted it for better visibility.

```
public void ddFacilities_processValueChange(ValueChangeEvent event) {
    try {
        sun.michael_czapski.xsds.facility.FacDetailsReq msgFacDetailsReq
            = null;
        sun.michael_czapski.wsdls.facilitysvc.FacilitySvcService service
            = new sun.michael_czapski.wsdls.facilitysvc.FacilitySvcService();
        sun.michael_czapski.wsdls.facilitysvc.FacilitySvcPortType port
            = service.getFacilitySvcPort();
        // TODO process result here
        sun.michael_czapski.xsds.facility.FacDetailsRes result
            = port.opGetFacDetails(msgFacDetailsReg);
        System.out.println("Result = " + result);
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

Let's remove sun.michael_czapski.xsds.facility and sun.michael_czapski.wsdld.facility svc strings to minimize clutter and use NetBeans IDE facilities to resolve references through the Java import mechanism. Right-click anywhere inside the Java source window and choose Fix Imports.



The brokenness will be resolved and appropriate import statements will be added.



Looking at the statement FacDetailsReq msgFacDetailsReq = null; we deduce that we have an opportunity to populate the request message with the facility code and that we will get a response containing facility details 4 statements later. Let's populate the request message with the new facility code chosen in the drop down.

```
public void ddFacilities_processValueChange(ValueChangeEvent event) {
    try {
        FacDetailsReq msgFacDetailsReq = new FacDetailsReq();
        msgFacDetailsReq.setFacCode ({String}event.getNewValue());
        FacilitySvcService service = new FacilitySvcService();
        FacilitySvcPortType port = service.getFacilitySvcPort();
        // TODO process result here
        FacDetailsRes result = port.opGetFacDetails(msgFacDetailsReq);
        System.out.println("Result = " + result);
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

Now the service execution (port.opGetFacDetails(...)) will give us details for the chosen facility. We can use the response message to populate all the static fields we created earlier.

As you enter this code note how NetBeans helps through code completion and other clever tricks.





Our ddFacility_processValueChange() method body is completed. Each time we change the value in the drop down we will get the details of the facility displayed.

Let's deploy the portlet, as it stands now, add it to the page and see what it looks like and how it behaves.

Notice that we have the populated drop down, as we did before, but no facility details for the selected facility on initial display.

Welcome	HC Portlet		
Facility Looku	p Portlet		$\bigcirc \bigcirc $
Choose Facil	lity A F	RED MEDICAL CENTRE	r
Facility Co	ode		NE
Descrip	otion		
Ad	dress		
	Status		

Let's choose a different facility and see what happens.

This time details for the new facility got displayed.

Welcome	HC Portl	et					
Facility Lookup Portlet							
Choose Facil Facility Co	lity ode	THE	E BIG HOSPITAL	V			
Descrip	ption	THE	BIG HOSPITAL				
Ad	dress	WHC ANY	OKNOWES WHERE ST				
	Status						

The reason we did not see the details when the web page first appeared was because the ddFacility_provessValueChange was not executed the first time around. To make it execute we need to add some code to the prerender() method to first establish whether this is the first time that the page is displayed, then to work out what the facility code is the first time around and finally to force ddFacility_processValueChange method to get executed so the facility details get displayed.

Let's do these things a bit at a time.

Let's switch to the Design mode, right-click on the drop down and choose Add Binding Attribute. If the text reads Remove Binding Attribute we don't need to do this.

Choose Facility	a	Edit Java Source
Facility Code	Stati	✓ Auto-Submit on Change Bind to Data
Description	Stati	Add
Address	Stati	Property Bindings
	 Stati	Add Binding Attribute
Status	Stati	Edit JSP Source りん

Let's switch to Java source mode and scroll down to the prerender() method.

At the end of the code already in the pre-render method let's add the following:

log("===>>> " + ddFacilities.getValue());

Deploy the portlet again, then add it to the page. Look at the server.log to see what this statement displays the first time the application is run, the change facility in the drop down and see what this statement displays the second time around.

On both occasions I see:



In the portlet I can not tell, by looking at the content of ddFacilities, whether this is the first time the portlet was shown or not. We do know that the ddFacilities_processValueChange() method was invoked because this is where the static fields get populated and wee see values in the static fields when we change the facility selection. Let's confirm the value of the selected facility by adding a log statement to see what the new value is each time the method is executed.

Let's add the following at the end of the ddFacilities_processValueChange():



Let's deploy the portlet, add it to the page, change the facility to Sydney Technical Hospital and see what the server.log says.



The first highlighted log entry comes from the ddFacilities_processValueChange() method. The second highlighted log entry comes from the prerender() method. Clearly, at one point we know what the chosen facility is but we "forget" it by the time we get to render the page.

Unlike a JSF web application, which gets rendered in a single pass, the portlet gets rendered in two passes (or so I read). This makes it 'forget' local variable values. We need to resort to session variables to pass values of interest between render passes.

Let's add the following code to the ddFacilityies_processValueChange() method, following the log statement:

The setValue() method will add a variable with specific name and value to the appropriate scope. The value of this variable can then retrieved elsewhere, possibly in rendering modified magic different pass. the incantation а "#{sessionScope.varName}" so that the variable name includes the package and class name, This way I don't have to worry about making sure the variable names are unique across portlets deployed to the same page. I still need to worry about adding the same portlet to the same page more then once but this is an issue that you can address if it worries you.

Let's now go back to the prerender() method, get the value of the sessionScope variable we are setting in the ddFacilities_processValueChange() method and display this value in the log to see what it is at different times.

```
log("===>>> " + ddFacilities.getValue());
String sFacCode
    = (String) getValue("#{sessionScope.facilitysvcvwjsfp_FacilityLookupPortlet_facCode}");
log("===>>> prerender: FacCode " + sFacCode);
```

Let's deploy the portlet and add it again to the page, assuming all the while that we are removing the portlet from the page after each test.

First time around, after deployment, we see log entries from the prerender method():

[#|2009-06-24T16:27:41.343+1000|INFO|sun-appserver2.1|java ervletContext.log():===>>> null|#]

[#|2009-06-24T16:27:41.343+1000|INF0|sun-appserver2.1|java
ervletContext.log():===>>> null|#]

[#|2009-06-24T16:27:41.343+1000|INF0|sun-appserver2.1|debu

[#|2009-06-24T16:27:41.343+1000|INF0|sun-appserver2.1|debu

[#|2009-06-24T16:27:41.343+1000|INF0|sun-appserver2.1|java
ervletContext.log():===>>> prerender: FacCode null|#]

[#|2009-06-24T16:27:41.343+1000|INF0|sun-appserver2.1|java
ervletContext.log():===>>> prerender: FacCode null|#]

In both tracers facility code (one from the ddFacility.getValue() and one from getValue("#{sessionScope...}) are null.

Let's choose Sydney Technical Hospital from the drop down and see what the log says.

[#|2009-06-24T16:28:43.906+1000|INFO|sun-appserver2.1|javax.enterprise.system.container.we VebModule[/FacilitySvcVWJSFP] ServletContext.log():===>>> New Facility STC[#]

[#12009-06-24T16:28:43.906+1000|INFO[sun-appserver2.1|debug.com.sun.portal.portletcontaine

[#|2009-06-24T16:28:43.937+1000|INFO|sun-appserver2.1|debug.com.sun.portal.portletcontaine itySvcVWJSFP;FacilityLookupPortlet;FacilityLookupPortlet_WAR_FacilitySvcVWJSFP_INSTANCE_U3 ?, PortletName:FacilityLookupPortlet, PortletWindowName:FacilityLookupPortlet_WAR_Facility

[#|2009-06-24T16:28:43.937+1000|INFO|sun-appserver2.1|javax.enterprise.system.container.we
VebModule[/FacilitySvcVWJSFP] ServletContext.log():===>>> null|#]

[#/2009-06-24T16:28:43.937+1000|INF0|sun-appserver2.1|debug.com.sun.portal.portletcontaine 28080-1; |===>>> null|#]

[#|2009-06-24T16:28:43.937+1000|INF0|sun-appserver2.1|javax.enterprise.system.container.we
WebModule[/FacilitySvcVWJSFP] ServletContext.log():===>>> prerender: FacCode STC[#]

[#|2009-06-24T16:28:43.937+1000|INFO|sun-appserver2.1|debug.com.sun.portal.portletcontaine 28080-1;|===>>> prerender: FacCode STC|#]

The first highlighted entry comes from the ddFacilities_processValueChange() method. We know at that point what the selected facility code is.

The method sets the session-scoped variable to that value.

In the second highlighted log statement, in prerender() method, we get a null from ddFacilities.getValue(). We don't know what the facility code just selected is.

In the third highlighted log statement, still in prerender() method we get the sessionscoped variable to work out what the facility code was in the ddFacilities_processValueChange().

As we can see the firs time the prerender() method is run the session-scoped variable will be null. On subsequent renders it will contain the value of the most recently selected facility code.

Knowing that this is happening allows us to determine whether we are displaying the first time or the subsequent time.

First time around, as we noticed, we have no idea what the "selected" code is. We may or may not be aware that at this point in the code the opGetFacList operation was executed and the list of facilities is available. We do know that the first time around the first facility will be the one "selected" in the drop down. We can exploit both of these to work out what the facility code is.

Let's add the following slab of code to the end of the prerender() method.

```
String sFacCode
  = (String) getValue("#{sessionScope.facilitysvcvwjsfp_FacilityLookupPortlet_facCode}");
log("===>>> prerender: FacCode " + sFacCode);
if (sFacCode == null) {
    // get the list of facility objects already assembled
   // get the first item in the list
    // get facility code for first item
   Object[] objFacListAry = facilitySvcPortOpGetfacList1.getResultObjects();
    Object oBjFirstFac = objFacListAry[0];
    FacListRes.FacList fAcR = (FacListRes.FacList) oBjFirstFac;
    // force the ddFacilities_processValueChange() method
    // ti execute using the facility code as new value
    ValueChangeEvent event =
           new ValueChangeEvent(ddFacilities, null, fAcR.getFacCode());
    ddFacilities processValueChange(event);
}
```

The facilitySvcPortOpGetfacList1.getResultObjects() is a reference to the web servicereturned list of facility objects. The first item in that list will give us access to the facility code. We create a new event in which we are setting the new value to that code and explicitly executing ddFacility_processValueChange method. This causes the static fields to be populated.

Here is the complete prerender() method.

```
public void prerender() {
    init();
    FacListReq flReq = new FacListReq();
    flReg.setDummyString("dummystring");
    facilitySvcPortOpGetfacList1.setMsgFacListReq(flReq);
    log("===>>> " + ddFacilities.getValue());
    String sFacCode
      = (String) getValue("#{sessionScope.facilitysvcvwjsfp_FacilityLookupPprtlet_facCode}");
    log("===>>> prerender: FacCode " + sFacCode);
    if (sFacCode == null) {
        // get the list of facility objects already assembled
        // get the first item in the list
        // get facility code for first item
        Object[] objFacListAry = facilitySvcPortOpGetfacList1.getResultObjects();
        Object oBjFirstFac = objFacListAry[0];
        FacListRes.FacList fAcR = (FacListRes.FacList) oBjFirstFac;
        // force the ddFacilities processValueChange() method
        // ti execute using the facility code as new value
        ValueChangeEvent event =
              new ValueChangeEvent(ddFacilities, null, fAcR.getFacCode());
        ddFacilities_processValueChange(event);
    3
3
```

Let's deploy the portlet, add it to the page again, and see what we get the first time around and what we get when we change the facility selection.

First time around:

Welcome HC P	ortiet	
acility Lookup Portlet		0000
Choose Facility	A RED MEDICAL CENTRE	•
Facility Code	ARMC	
Description	A RED MEDICAL CENTRE	
Address	SOMEWHERE ST	
	SOMEWHERE CITY, NSW, 2060	

Choosing Sydney Technical Hospital:

Welcome	HC Port	let						
Facility Lookup	Facility Lookup Portlet							
Choose Facilit	у	SYDNEY TECHNICAL HOSPITAL	•					
Facility Coo	le	STC						
Descripti	ion	SYDNEY TECHNICAL HOSPITAL						
Add	ress	404 HUNTINGTON DRIVE						
		MONROVIA, CA, 91016						
S	tatus	CA						

The portlet is now complete and behaves the way we expect it to, both the first time it is displaye and each time a different facility is chosen.

Note that the portlet was showing up in the "User Portlets" group of Applications. We can change that to a group with a name of our own choosing.

Expand the project FacilitySvcVWJSFP, expand Configuration Files and open liferaydisplay.xml configuration file in the editor. Change the text in name attribute of the category tag.



For example changing the name to "HC Portlets" will make the portlet appear in a new category, with that name, once it is next deployed. Deploy the portlet to effect the change.

Note the portlet in the new group.



This is it. This is what it took to develop a portlet that used the Facilities web service as a data provider and deploy it to the Web Space Server 10 Portal.

Summary

In this document we created, deployed and exercised a JSR 286-copliant portlet that provided a list of Healthcare Facilities as a drop down and details of a specific Facility when chosen.

We used the NetBenas 6.5.1 IDE, included with the GlassFish ESB v 2.1 infrastructure, Portal Pack 3.0.1 and Web Space Server 10. We used the Visual Web JavaServer Faces Portlet and JSF Portal Bridge technologies, Project Woodstock JSF components and JBI-based multi-operation web service. The NetBeans IDE tooling assisted in rapidly developing the web application with minimum of custom Java code. This web application, a component in SOA 1, Presentation Layer, consumes SOA 3, Business Service service in a loosely coupled manner.