BEA WebLogic Workshop™ Help

Version 8.1 SP4
December 2004

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Table of Contents

controls Samples

This section contains source code for the following samples.

Samples Included in This Section

ControlsController.jpf Sample

database Samples

Index.jsp Sample

webservice Samples

ControlsController.jpf Sample

This topic inludes the source code for the ControlsController.jpf Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/

Sample Source Code

```
package controls;
import com.bea.wlw.netui.pageflow.Forward;
import com.bea.wlw.netui.pageflow.PageFlowController;

/**
 * @jpf:controller
 * @jpf:view-properties view-properties::
 * <!-- This data is auto-generated. Hand-editing this section is not recommended. -->
 * <view-properties>
 * <pageflow-object id="pageflow:/controls/controlsController.jpf"/>
 * <pageflow-object id="action:begin.do">
 *   <property value="80" name="x"/>
 *   <property value="100" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:toDatabase.do">
 *   <property value="340" name="x"/>
 *   <property value="60" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:toHelloWorld.do">
 *   <property value="360" name="x"/>
 *   <property value="120" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:toPolling.do">
 *   <property value="340" name="x"/>
 *   <property value="180" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:toPageFlowSamples.do">
 *   <property value="220" name="x"/>
 *   <property value="220" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:index.jsp@#@action:toPolling.do@">
 *   <property value="276,290,290,304" name="elbowsX"/>
 *   <property value="103,103,172,172" name="elbowsY"/>
 *   <property value="East_2" name="fromPort"/>
 *   <property value="West_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:index.jsp@#@action:toDatabase.do@">
 *   <property value="276,290,290,304" name="elbowsX"/>
 *   <property value="81,81,52,52" name="elbowsY"/>
 *   <property value="East_0" name="fromPort"/>
 *   <property value="West_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:index.jsp@#@action:toPageFlowSamples.do@">
 *   <property value="240,240,220,220" name="elbowsX"/>
 *   <property value="144,160,160,176" name="elbowsY"/>
```

```
*   <property value="South_1" name="fromPort"/>
*   <property value="North_1" name="toPort"/>
* </pageflow-object>
* <pageflow-object id="action-call:@page:index.jsp@#@action:toHelloWorld.do@">
*   <property value="276,300,300,324" name="elbowsX"/>
*   <property value="92,92,112,112" name="elbowsY"/>
*   <property value="East_1" name="fromPort"/>
*   <property value="West_1" name="toPort"/>
* </pageflow-object>
* <pageflow-object id="page:index.jsp">
*   <property value="240" name="x"/>
*   <property value="100" name="y"/>
* </pageflow-object>
* <pageflow-object id="external-jpf:database/databaseController.jpf">
*   <property value="480" name="x"/>
*   <property value="60" name="y"/>
* </pageflow-object>
* <pageflow-object id="external-jpf:webservice/helloworld/HelloWorldController.jpf">
*   <property value="480" name="x"/>
*   <property value="120" name="y"/>
* </pageflow-object>
* <pageflow-object id="external-jpf:webservice/polling/PollingController.jpf">
*   <property value="460" name="x"/>
*   <property value="180" name="y"/>
* </pageflow-object>
* <pageflow-object id="external-jpf:/Controller.jpf">
*   <property value="220" name="x"/>
*   <property value="340" name="y"/>
* </pageflow-object>
* <pageflow-object id="forward:path#success#index.jsp#@action:begin.do@">
*   <property value="116,160,160,204" name="elbowsX"/>
*   <property value="92,92,92,92" name="elbowsY"/>
*   <property value="East_1" name="fromPort"/>
*   <property value="West_1" name="toPort"/>
*   <property value="success" name="label"/>
* </pageflow-object>
* <pageflow-object id="forward:path#success#database/databaseController.jpf#@action:toDatabase
*   <property value="376,410,410,444" name="elbowsX"/>
*   <property value="52,52,52,52" name="elbowsY"/>
*   <property value="East_1" name="fromPort"/>
*   <property value="West_1" name="toPort"/>
*   <property value="success" name="label"/>
* </pageflow-object>
* <pageflow-object id="forward:path#success#webservice/helloworld/HelloWorldController.jpf#@ac
*   <property value="396,420,420,444" name="elbowsX"/>
*   <property value="112,112,112,112" name="elbowsY"/>
*   <property value="East_1" name="fromPort"/>
*   <property value="West_1" name="toPort"/>
*   <property value="success" name="label"/>
* </pageflow-object>
* <pageflow-object id="forward:path#success#webservice/polling/PollingController.jpf#@action:t
*   <property value="376,400,400,424" name="elbowsX"/>
*   <property value="172,172,172,172" name="elbowsY"/>
*   <property value="East_1" name="fromPort"/>
*   <property value="West_1" name="toPort"/>
*   <property value="success" name="label"/>
* </pageflow-object>
* <pageflow-object id="forward:path#success#/Controller.jpf#@action:toPageFlowSamples.do@">
*   <property value="220,220,220,220" name="elbowsX"/>
*   <property value="264,280,280,296" name="elbowsY"/>
*   <property value="South_1" name="fromPort"/>
```

```
 *   <property value="North_1" name="toPort"/>
 *   <property value="success" name="label"/>
 * </pageflow-object>
 * </view-properties>
 * ::
 */
public class controlsController extends PageFlowController
{

    /**
     * @jpf:action
     * @jpf:forward name="success" path="index.jsp"
     */
    protected Forward begin()
    {
        return new Forward("success");
    }

    /**
     * @jpf:action
     * @jpf:forward path="database/databaseController.jpf" name="success"
     */
    protected Forward toDatabase()
    {
        return new Forward("success");
    }

    /**
     * @jpf:action
     * @jpf:forward path="webservice/helloworld/HelloWorldController.jpf" name="success"
     */
    protected Forward toHelloWorld()
    {
        return new Forward("success");
    }

    /**
     * @jpf:action
     * @jpf:forward path="webservice/polling/PollingController.jpf" name="success"
     */
    protected Forward toPolling()
    {
        return new Forward("success");
    }

    /**
     * @jpf:action
     * @jpf:forward path="/Controller.jpf" name="success"
     */
    protected Forward toPageFlowSamples()
    {
        return new Forward("success");
    }

}
```

database Samples

This section contains source code for the following samples.

Samples Included in This Section

DatabaseController.jpf Sample

List_users.jsp Sample

UsersDBControl.jcx Sample

DatabaseController.jpf Sample

This topic inludes the source code for the DatabaseController.jpf Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/database/

Sample Source Code

```
package controls.database;

import com.bea.wlw.netui.pageflow.*;

/**
 * This Page Plow queries a database by calling methods on the
 * the database control file UsersDBControl.jcx.  Each method
 * on the control sends a SQL query to the database.  There are methods for
 * selecting, inserting, updating, and deleting records.
 *
 * This Page Plow provides a user interface for the database control,
 * invokes the control methods, and displays the results
 * on the JSP page list_users.jsp.
 */

/**
 * @jpf:view-properties view-properties::
 * <!-- This data is auto-generated. Hand-editing this section is not recommended. -->
 * <view-properties>
 * <pageflow-object id="pageflow:/controls/database/databaseController.jpf"/>
 * <pageflow-object id="action:begin.do">
 *   <property name="x" value="60"/>
 *   <property name="y" value="200"/>
 * </pageflow-object>
 * <pageflow-object id="action:addUser.do#controls.database.databaseController.UserForm">
 *   <property value="200" name="x"/>
 *   <property value="60" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:deleteUser.do">
 *   <property name="x" value="380"/>
 *   <property name="y" value="200"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:list_users.jsp@#@action:addUser.do#controls.database.
 *   <property value="200,200,200,200" name="elbowsX"/>
 *   <property value="156,130,130,104" name="elbowsY"/>
 *   <property value="North_1" name="fromPort"/>
 *   <property value="South_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:list_users.jsp@#@action:deleteUser.do@">
 *   <property value="236,290,290,344" name="elbowsX"/>
 *   <property value="192,192,192,192" name="elbowsY"/>
 *   <property value="East_1" name="fromPort"/>
 *   <property value="West_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="page:list_users.jsp">
```

```
 *   <property name="x" value="200"/>
 *   <property name="y" value="200"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#list_users.jsp#@action:begin.do@">
 *   <property name="elbowsY" value="192,192,192,192"/>
 *   <property name="toPort" value="West_1"/>
 *   <property name="elbowsX" value="96,130,130,164"/>
 *   <property name="label" value="success"/>
 *   <property name="fromPort" value="East_1"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#list_users.jsp#@action:addUser.do#controls.databas
 *   <property value="200,200,200,200" name="elbowsX"/>
 *   <property value="104,130,130,156" name="elbowsY"/>
 *   <property value="South_1" name="fromPort"/>
 *   <property value="North_1" name="toPort"/>
 *   <property value="success" name="label"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#list_users.jsp#@action:deleteUser.do@">
 *   <property name="elbowsY" value="192,192,192,192"/>
 *   <property name="toPort" value="East_1"/>
 *   <property name="elbowsX" value="344,290,290,236"/>
 *   <property name="label" value="success"/>
 *   <property name="fromPort" value="West_1"/>
 * </pageflow-object>
 * <pageflow-object id="control:controls.database.UsersDBControl#users">
 *   <property value="26" name="x"/>
 *   <property value="34" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="formbeanprop:controls.database.databaseController.UserForm#username#jav
 * <pageflow-object id="formbeanprop:controls.database.databaseController.UserForm#password#jav
 * <pageflow-object id="formbeanprop:controls.database.databaseController.UserForm#message#java
 * <pageflow-object id="formbean:controls.database.databaseController.UserForm"/>
 * </view-properties>
 * ::
 *
 */
public class databaseController extends PageFlowController
{
    /**
     * Declare the database control.
     * You can now call methods on the database control through
     * the 'users' object.
     *
     * @common:control
     */
    private UsersDBControl users;


    /**
     * This method queries the USERS table with the SQL query: SELECT * FROM USERS
     */
    public UsersDBControl.User[] getAllUsers()
    {
        return users.getAllUsers();
    }

    /**
     * When this Page Flow is first invoked, it tries to create and populate the
     * table USERS.  This ensures that the database control does not query
     * a non-existent table.
     *
```

```
     * @jpf:action
     * @jpf:forward name="success" path="list_users.jsp"
     */
    public Forward begin()
    {
        try
        {
            users.createUsersTable();
            users.insertUser( "JohnS", "1234" );
            users.insertUser( "SteveH", "5678" );
        }
        catch( Exception e )
        {
            // ignore -- the table might already have been created,
            // which will cause an exception.
        }

        return new Forward( "success" );
    }

    /**
     * This method adds a record to the USERS table.
     *
     * @jpf:action
     * @jpf:forward name="success" path="list_users.jsp"
     */
    public Forward addUser( UserForm form )
    {
       if ( form.getUsername().length() == 0 || form.getPassword().length() == 0 )
       {
           form.setMessage( "Please enter both a username and a password." );
           return new Forward( "success", form );
       }

       users.insertUser( form.getUsername(), form.getPassword() );
       return new Forward( "success", new UserForm() );
    }

    /**
     * This method deletes a record from the USERS table.
     *
     * @jpf:action
     * @jpf:forward name="success" path="list_users.jsp"
     */
    public Forward deleteUser()
    {
        users.deleteUser( getRequest().getParameter( "userToDelete" ) );
        return new Forward( "success", new UserForm() );
    }

    public static class UserForm extends FormData
    {
       private String _username;
       private String _password;
       private String _message;

       public String getUsername()
       {
           return _username;
       }
```

```
        public void setUsername( String username )
        {
            _username = username;
        }

        public String getPassword()
        {
            return _password;
        }

        public void setPassword( String password )
        {
            _password = password;
        }

        public String getMessage()
        {
            return _message;
        }

        public void setMessage( String message )
        {
            _message = message;
        }
    }
}
```

List_users.jsp Sample

This topic inludes the source code for the List_users.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/database/

Sample Source Code

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>

<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
    <head>
        <netui:base/>
        <title>Java Control: Called from a Page Flow</title>
        <link href="/WebApp/resources/css/style.css" type="text/css" rel="stylesheet"/>
    </head>

    <body>
        <jsp:include page="/resources/jsp/header.jsp"/>
        <h3>Calling a Database Control</h3>
        <p>This JSP page is part of a Page Flow that calls a Java control.
        Instead of embedding the business logic in this JSP, the Page
        Flow's Controller file is used to call the separate Java control, UsersDBControl.jcx.</
        <hr>
        <blockquote>

        <netui:label
        value="Add or Remove User Accounts" style="font-size : 12px; font-family : Verdana, Ari
        font-weight : bold; color: black; text-align: left;
        padding-left: 5px;" />

        <!--This netui-data:repeater element calls the getAllUsers method
        on the page flow callJavaControlController.jpf via the XScript
        expression {pageFlow.allUsers}.  "pageFlow" refers to
        callJavaControlController.jpf; "allUsers" refers to the getter
        method getAllUsers in callJavaControlController.jpf -->
        <netui-data:repeater dataSource="{pageFlow.allUsers}">
            <netui-data:repeaterHeader>
                <table border=1>
                <tr><td class="header-text">Username</td>
                <td class="header-text">Password</td></tr>
            </netui-data:repeaterHeader>
            <netui-data:repeaterItem>
                <tr>
                 <td class="row-text"><netui:label value="{container.item.username}"/></td>
                 <td class="row-text"><netui:label value="{container.item.password}"/></td>
                 <td class="row-text">
                  <netui:anchor action="deleteUser">
```

```
                    <netui:parameter name="userToDelete" value="{container.item.username}"/>Del
                  </netui:anchor>
                 </td>
                </tr>
            </netui-data:repeaterItem>
            <netui-data:repeaterFooter>
                </table>
            </netui-data:repeaterFooter>
        </netui-data:repeater>

        <p>
        <netui:form action="addUser">
            <font color="red"><netui:label value="{actionForm.message}" /></font>
            <br>
            <table>
                <tr><td class="label">Username:</td><td><netui:textBox dataSource="username" />
                <tr><td class="label">Password:</td><td><netui:textBox dataSource="password" />
            </table>
            <br>
            <netui:button type="Submit">Add New Account</netui:button>
        </netui:form>

        </blockquote>

        <hr>
        <p><netui:anchor href="/WebApp/controls/controlsController.jpf">Back to Control Samples

    </body>
</html>
```

UsersDBControl.jcx Sample

This topic inludes the source code for the UsersDBControl.jcx Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/database/

Sample Source Code

```
package controls.database;

import java.io.Serializable;
import java.sql.SQLException;

import com.bea.control.DatabaseControl;

/**
 * @jc:connection data-source-jndi-name="cgSampleDataSource"
 */
public interface UsersDBControl extends com.bea.control.ControlExtension, DatabaseControl
{
    /**
     * @jc:sql statement="create table users (username VARCHAR(50), password VARCHAR(50))"
     */
    public void createUsersTable() throws SQLException;

    /**
     * @jc:sql statement::
     *   INSERT INTO USERS (username, password)
     *   VALUES ({username}, {password})
     * ::
     */
    public int insertUser( String username, String password );

    /**
     * @jc:sql statement::
     * SELECT * FROM USERS WHERE USERNAME = {username}
     * ::
     */
    public User lookupUser( String username );

    /**
     * @jc:sql statement::
     * UPDATE USERS SET USERNAME = {username}, PASSWORD = {password} WHERE USERNAME = {username
     * ::
     */
    public int updateUser( String username, String password );

    /**
     * @jc:sql statement::
     * SELECT * FROM USERS
     * ::
     */
    public User[] getAllUsers();
```

```
    /**
     * @jc:sql statement::
     * DELETE FROM USERS WHERE USERNAME = {username}
     * ::
     */
    public int deleteUser( String username );


    public static class User implements Serializable
    {
        public String username;
        public String password;

        public User()
        {
        }

        public User( String username, String password )
        {
            this.username = username;
            this.password = password;
        }
    }
}
```

Index.jsp Sample

This topic inludes the source code for the Index.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/

Sample Source Code

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>
<%@ taglib uri="netui-tags-template.tld" prefix="netui-template"%>
<netui:base />
<netui:html>
    <head>
        <title>
            Web Application Samples: Controls
        </title>
        <link href="/WebApp/resources/css/style.css" type="text/css" rel="stylesheet"/>
    </head>
    <body>
        <jsp:include page="/resources/jsp/header.jsp"/>
        <h3>Web Application Samples: Using Controls</h3>
        <p>These samples show how to use controls to access resources.</p>
        <p>A theme of all these samples is the separation between data <i>presentation</i>
        (handled by the JSP pages)
        and data <i>processing</i> (handled by the Controller files).  None of the JSP pages
        containing any of the control-querying logic. Instead the logic
        is contained in the Controller files. This helps maintain a strong seperation between d
        and data processing.
        <blockquote>
        <hr>
        <p><netui:anchor action="toDatabase">Calling a Database</netui:anchor>
        <p>This sample shows how to (1) query a database using a database control and (2) displ
        <hr>
        <p><netui:anchor action="toHelloWorld">Calling a Synchronous Web Service</netui:anchor>
        <p>This sample shows how to query a synchronous Web Serivce using a control.
        <hr>
        <p><netui:anchor action="toPolling">Calling an Asynchronous Web Service</netui:anchor>
        <p>This sample shows how to query an asynchronous Web Service using a control.
        </blockquote>

        <hr>
        <p><netui:anchor action="toPageFlowSamples">Back to Web Application Samples Home</netui
    </body>
</netui:html>
```

webservice Samples

This section contains source code for the following samples.

Samples Included in This Section

helloworld Samples

polling Samples

helloworld Samples

This section contains source code for the following samples.

Samples Included in This Section

HelloWorldController.jpf Sample

Index.jsp Sample

Result.jsp Sample

syncWebService Samples

HelloWorldController.jpf Sample

This topic inludes the source code for the HelloWorldController.jpf Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/helloworld/

Sample Source Code

```
package controls.webservice.helloworld;

import com.bea.wlw.netui.pageflow.FormData;
import com.bea.wlw.netui.pageflow.Forward;
import com.bea.wlw.netui.pageflow.PageFlowController;

import controls.webservice.helloworld.syncWebService.HelloWorldControl;

/**
 * This Page Flow calls a web service control.  It invokes the method "HelloWorld"
 * and displays the result on the JSP page "result.jsp".
 *
 * @jpf:view-properties view-properties::
 * <!-- This data is auto-generated. Hand-editing this section is not recommended. -->
 * <view-properties>
 * <pageflow-object id="pageflow:/controls/webservice/helloworld/HelloWorldController.jpf"/>
 * <pageflow-object id="action:begin.do">
 *   <property name="x" value="60"/>
 *   <property name="y" value="60"/>
 * </pageflow-object>
 * <pageflow-object id="action:HelloWorld.do">
 *   <property name="x" value="60"/>
 *   <property name="y" value="160"/>
 * </pageflow-object>
 * <pageflow-object id="page:index.jsp">
 *   <property name="x" value="180"/>
 *   <property name="y" value="60"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:result.jsp@#@action:begin.do@">
 *   <property value="144,120,120,96" name="elbowsX"/>
 *   <property value="152,152,52,52" name="elbowsY"/>
 *   <property value="West_1" name="fromPort"/>
 *   <property value="East_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="page:result.jsp">
 *   <property name="x" value="180"/>
 *   <property name="y" value="160"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#index.jsp#@action:begin.do@">
 *   <property name="elbowsY" value="52,52,52,52"/>
 *   <property name="elbowsX" value="96,120,120,144"/>
 *   <property name="toPort" value="West_1"/>
 *   <property name="fromPort" value="East_1"/>
 *   <property name="label" value="success"/>
 * </pageflow-object>
```

```
 * <pageflow-object id="forward:path#success#result.jsp#@action:HelloWorld.do@">
 *   <property name="elbowsY" value="152,152,152,152"/>
 *   <property name="elbowsX" value="96,120,120,144"/>
 *   <property name="toPort" value="West_1"/>
 *   <property name="fromPort" value="East_1"/>
 *   <property name="label" value="success"/>
 * </pageflow-object>
 * <pageflow-object id="control:controls.webservice.helloworld.syncWebService.HelloWorldControl
 *   <property value="28" name="x"/>
 *   <property value="34" name="y"/>
 * </pageflow-object>
 * </view-properties>
 * ::
 *
 */
public class HelloWorldController extends PageFlowController
{

    public String result;

    /**
     *  @common:control
     */
    private HelloWorldControl myControl;


    /**
     * @jpf:action
     * @jpf:forward name="success" path="index.jsp"
     */
    protected Forward begin()
    {
        return new Forward( "success" );
    }



    /**
     * This method invokes the control method "HelloWorld"
     *
     * @jpf:action
     * @jpf:forward name="success" path="result.jsp"
     */
    public Forward HelloWorld()
    {
        try
        {
            result = myControl.HelloWorld();

            return new Forward( "success" );

        }
        catch( Throwable ex )
        {
            ex.printStackTrace();
        }
        return null;
    }

}
```

Index.jsp Sample

This topic inludes the source code for the Index.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/helloworld/

Sample Source Code

```
<!--Generated by Weblogic Workshop-->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>
<html>
  <head>
    <netui:base/>
    <link href="../../../resources/css/style.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <jsp:include page="../../../resources/jsp/header.jsp"/>
    <h3>Calling a (Synchronous) Web Service from a Page Flow</h3>
    <p>This sample demonstrates how to call a (synchronous) web service from a Page Flow.
    <p>Clicking the link below invokes the HelloWorld.jws web service.
    <blockquote>

    <p><netui:anchor action="HelloWorld.do">
      Get message from the HelloWorld web service</netui:anchor>

    </blockquote>
    <hr>
    <p><netui:anchor href="/WebApp/controls/controlsController.jpf">Back to Control Samples</ne


  </body>
</html>
```

Result.jsp Sample

This topic inludes the source code for the Result.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/helloworld/

Sample Source Code

```
<%@page contentType="text/html;charset=UTF-8" language="java"%>
<%@taglib prefix="netui" uri="netui-tags-html.tld"%>
<html>
  <head>
    <title>Result from Synchronous Web Service</title>
    <netui:base/>
    <link href="/WebApp/resources/css/style.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <jsp:include page="/resources/jsp/header.jsp"/>
    <br>
    <br>

    <blockquote>

        <p>The message from the HelloWorld web service is:</p>

        <p><netui:label style="font-weight:bold;" value="{pageFlow.result}" />


    </blockquote>
    <hr>
    <p><netui:anchor action="begin">Re-run this sample</netui:anchor></p>
    <p><netui:anchor href="/WebApp/controls/controlsController.jpf">Back to Control Samples</ne

    </body>
</html>
```

syncWebService Samples

This section contains source code for the following samples.

Samples Included in This Section

HelloWorld.jws Sample

HelloWorldControl.jcx Sample

HelloWorld.jws Sample

This topic inludes the source code for the HelloWorld.jws Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/helloworld/syncWebService/

Sample Source Code

```
package controls.webservice.helloworld.syncWebService;

public class HelloWorld implements com.bea.jws.WebService
{
    /**
     * @common:operation
     */
    public String HelloWorld()
    {
        return "Hello, World!";
    }
}
```

HelloWorldControl.jcx Sample

This topic inludes the source code for the HelloWorldControl.jcx Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/helloworld/syncWebService/

Sample Source Code

```
package controls.webservice.helloworld.syncWebService;

/**
 * @jc:location http-url="HelloWorld.jws" jms-url="HelloWorld.jws"
 * @jc:wsdl file="#HelloWorldWsdl"
 * @editor-info:link autogen-style="java" source="HelloWorld.jws" autogen="true"
 */
public interface HelloWorldControl extends com.bea.control.ControlExtension, com.bea.control.Se
{

    public java.lang.String HelloWorld ();

    static final long serialVersionUID = 1L;
}

/** @common:define name="HelloWorldWsdl" value::
    <?xml version="1.0" encoding="utf-8"?>
    <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:conv="http://www.openuri.org/20
      <types>
        <s:schema elementFormDefault="qualified" targetNamespace="http://www.openuri.org/" xmln
          <s:element name="HelloWorld">
            <s:complexType>
              <s:sequence/>
            </s:complexType>
          </s:element>
          <s:element name="HelloWorldResponse">
            <s:complexType>
              <s:sequence>
                <s:element name="HelloWorldResult" type="s:string" minOccurs="0"/>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="string" nillable="true" type="s:string"/>
        </s:schema>

      </types>
      <message name="HelloWorldSoapIn">
        <part name="parameters" element="s0:HelloWorld"/>
      </message>
      <message name="HelloWorldSoapOut">
        <part name="parameters" element="s0:HelloWorldResponse"/>
      </message>
      <message name="HelloWorldHttpGetIn"/>
      <message name="HelloWorldHttpGetOut">
        <part name="Body" element="s0:string"/>
```

```
</message>
<message name="HelloWorldHttpPostIn"/>
<message name="HelloWorldHttpPostOut">
  <part name="Body" element="s0:string"/>
</message>
<portType name="HelloWorldSoap">
  <operation name="HelloWorld">
    <input message="s0:HelloWorldSoapIn"/>
    <output message="s0:HelloWorldSoapOut"/>
  </operation>
</portType>
<portType name="HelloWorldHttpGet">
  <operation name="HelloWorld">
    <input message="s0:HelloWorldHttpGetIn"/>
    <output message="s0:HelloWorldHttpGetOut"/>
  </operation>
</portType>
<portType name="HelloWorldHttpPost">
  <operation name="HelloWorld">
    <input message="s0:HelloWorldHttpPostIn"/>
    <output message="s0:HelloWorldHttpPostOut"/>
  </operation>
</portType>
<binding name="HelloWorldSoap" type="s0:HelloWorldSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
  <operation name="HelloWorld">
    <soap:operation soapAction="http://www.openuri.org/HelloWorld" style="document"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
<binding name="HelloWorldHttpGet" type="s0:HelloWorldHttpGet">
  <http:binding verb="GET"/>
  <operation name="HelloWorld">
    <http:operation location="/HelloWorld"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
</binding>
<binding name="HelloWorldHttpPost" type="s0:HelloWorldHttpPost">
  <http:binding verb="POST"/>
  <operation name="HelloWorld">
    <http:operation location="/HelloWorld"/>
    <input>
      <mime:content type="application/x-www-form-urlencoded"/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
</binding>
<service name="HelloWorld">
  <port name="HelloWorldSoap" binding="s0:HelloWorldSoap">
```

```
        <soap:address location="http://localhost:7001/controls/webservice/helloworld/HelloWor
      </port>
      <port name="HelloWorldHttpGet" binding="s0:HelloWorldHttpGet">
        <http:address location="http://localhost:7001/controls/webservice/helloworld/HelloWor
      </port>
      <port name="HelloWorldHttpPost" binding="s0:HelloWorldHttpPost">
        <http:address location="http://localhost:7001/controls/webservice/helloworld/HelloWor
      </port>
    </service>
  </definitions>
* ::
*/
```

polling Samples

This section contains source code for the following samples.

Samples Included in This Section

asyncWebService Samples

Index.jsp Sample

PollingController.jpf Sample

Result.jsp Sample

asyncWebService Samples

This section contains source code for the following samples.

Samples Included in This Section

HelloWorldAsync.jws Sample

HelloWorldAsyncControl.jcx Sample

HelloWorldAsync.jws Sample

This topic inludes the source code for the HelloWorldAsync.jws Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/polling/asyncWebService/

Sample Source Code

```
package controls.webservice.polling.asyncWebService;

/**
 * This originally asynchronous web service (copied and modified from
 * SamplesApp/WebServices/async/HelloWorldAsych.jws) has been supplemented with a polling
 * interface. If the client specifies that it cannot hear callbacks (via the
 * useCallback parameter on the HelloAsync() method), the result is not sent
 * in a callback, instead it is stored in the member variable m_message.
 *
 * Two methods have been added to the original web service:
 *
 * checkStatus()
 * getMessageResponse()
 *
 * Three member variables have been added:
 *
 * m_useCallbacks
 * m_messageIsReceived
 * m_message
 *
 * A parameter has been added to the requestMessage() method:
 *
 * useCallbacks.
 *
 * A web service that uses a TimerControl to delay sending a response back to the client.
 *
 * The timer simulates waiting for a slow back end service to complete work for us.  We
 * use a callback to asynchronously notify the client when the simulated operation is
 * complete.
 */

public class HelloWorldAsync implements com.bea.jws.WebService
{
    /**
     * @jc:timer timeout="2 seconds"
     * @common:control
     */
    private com.bea.control.TimerControl helloDelay;

    /*
     * This member variable stores the client choice to be sent a callback or not.
     */
    public boolean m_useCallback;

    /*
```

```
 * When the callback handler is fired, this boolean is set to true.
 * Clients that don't want callbacks check this boolean to see if their result is ready.
 */
public boolean m_messageIsReceived;

/*
 * This member variable stores the result for clients that don't want callbacks
 */
public String m_message = "";


/**
 * <p>callback is the variable that represents the client connection.  It's used
 * to enable our service to send messages back to the client asynchronously.
 * callback is type Callback, which is defined next;</p>
 */
public Callback callback;

/**
 * <p>the Callback interface is the definition of which messages the client will
 * accept from us via the callback variable.</p>
 */
public interface Callback extends com.bea.control.ServiceControl.Callback
{
    /**
     * <p>HelloResult is the message we will send back to the client some time after
     * the client initiates the operation.</p>
     *
     * <p>We mark the callback as finishing the conversation because once we
     * invoke the callback the interaction between client and this service is
     * complete.  If we didn't use the @common:conversation tag here to finish
     * the conversation, we would have to call finishConversation() on this
     * service's context object.</p>
     *
     * @jws:conversation phase="finish"
     */
    public void onHelloResult(String hello);
}

/**
 * <p>The client starts the interaction by calling HelloAsync.</p>
 *
 * <p>The client sends this, and some time later our web service replies by calling
 * callback.onHelloResult.</p>
 *
 * <p>Because we need to remember which client called us, this is a "conversation start"
 * method.  This means the system will automatically track which clients have called
 * us, and where to send the result for each.  This is known as <i>correlation</i>.</p>
 *
 * <p>Because it uses a TimerControl method that might throw an Exception, the
 * method must declare that it might throw an Exception.</p>
 *
 * @common:operation
 * @jws:conversation phase="start"
 */
public void requestMessage(boolean useCallback)
{
    m_useCallback = useCallback;

    // Start the delay timer.
    helloDelay.start();
```

```
    return;
}

/**
 * Clients that don't want callbacks call this method to see if the results are ready
 * to be retrieved.
 *
 * @common:operation
 * @jws:conversation phase="continue"
 */
public boolean checkStatus()
{
    return m_messageIsReceived;
}

/**
 * Clients that don't want callbacks call this method to get the result
 *
 * @common:operation
 * @jws:conversation phase="finish"
 */
public String getMessageResponse()
{
    return m_message;
}



/**
 * <p>The handler for helloTimer's onTimeout event.</p>
 *
 * <p>When the timer expires, this method will be invoked.  When that
 * happens, we'll send the result back to the client who asked for it (if the
 * client specifies they want a callback).
 *
 * If the client has specified no callbacks, then we store the result in the
 * member variable m_message.</p>
 */
private void helloDelay_onTimeout(long time)
{
    String hello = "Hello, World!";

    /*
     * If the client doesn't want callbacks, store the result in the member variable
     * m_messageIsReceived.
     */
    if(m_useCallback == false)
    {
        m_messageIsReceived = true;

        m_message = hello;
    }
    else
    {
    // If the client wants a callback, send the client a hello message.
    callback.onHelloResult(hello);
    }

    // we don't want any more timer events for this conversation.
    helloDelay.stop();
```

```
        return;
    }

}
```

HelloWorldAsyncControl.jcx Sample

This topic inludes the source code for the HelloWorldAsyncControl.jcx Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/polling/asyncWebService/

Sample Source Code

```
package controls.webservice.polling.asyncWebService;

/**
 * This originally asynchronous web service (copied and modified from SamplesApp/WebServices/as
 * @jc:location http-url="HelloWorldAsync.jws" jms-url="HelloWorldAsync.jws"
 * @jc:wsdl file="#HelloWorldAsyncWsdl"
 * @editor-info:link autogen-style="java" source="HelloWorldAsync.jws" autogen="true"
 */
public interface HelloWorldAsyncControl extends com.bea.control.ControlExtension, com.bea.contr
{
    public static class StartHeader
        implements java.io.Serializable
    {
        public java.lang.String conversationID;
        public java.lang.String callbackLocation;
    }

    public static class ContinueHeader
        implements java.io.Serializable
    {
        public java.lang.String conversationID;
    }

    public static class CallbackHeader
        implements java.io.Serializable
    {
        public java.lang.String conversationID;
    }


    public interface Callback extends com.bea.control.ServiceControl.Callback
    {
        /**
         * <p>HelloResult is the message we will send back to the client some time after the cl
         * @jc:conversation phase="finish"
         */
        public void onHelloResult (java.lang.String hello);

    }

    /**
     * <p>The client starts the interaction by calling HelloAsync.</p> <p>The client sends this
     * @jc:conversation phase="start"
     */
    public void requestMessage (boolean useCallback);
```

```
    /**
     * Clients that don't want callbacks call this method to see if the results are ready to be
     * @jc:conversation phase="continue"
     */
    public boolean checkStatus ();

    /**
     * Clients that don't want callbacks call this method to get the result
     * @jc:conversation phase="finish"
     */
    public java.lang.String getMessageResponse ();

    static final long serialVersionUID = 1L;
}

/** @common:define name="HelloWorldAsyncWsdl" value::
    <?xml version="1.0" encoding="utf-8"?>
    <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:conv="http://www.openuri.org/20
      <types>
        <s:schema elementFormDefault="qualified" targetNamespace="http://www.openuri.org/" xmln
          <s:element name="onHelloResultResponse">
            <s:complexType>
              <s:sequence/>
            </s:complexType>
          </s:element>
          <s:element name="onHelloResult">
            <s:complexType>
              <s:sequence>
                <s:element name="hello" type="s:string" minOccurs="0"/>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="requestMessage">
            <s:complexType>
              <s:sequence>
                <s:element name="useCallback" type="s:boolean"/>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="requestMessageResponse">
            <s:complexType>
              <s:sequence/>
            </s:complexType>
          </s:element>
          <s:element name="checkStatus">
            <s:complexType>
              <s:sequence/>
            </s:complexType>
          </s:element>
          <s:element name="checkStatusResponse">
            <s:complexType>
              <s:sequence>
                <s:element name="checkStatusResult" type="s:boolean"/>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="boolean" type="s:boolean"/>
          <s:element name="getMessageResponse">
            <s:complexType>
              <s:sequence/>
```

```
      </s:complexType>
    </s:element>
    <s:element name="getMessageResponseResponse">
      <s:complexType>
        <s:sequence>
          <s:element name="getMessageResponseResult" type="s:string" minOccurs="0"/>
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="string" nillable="true" type="s:string"/>
  </s:schema>

  <s:schema elementFormDefault="qualified" targetNamespace="http://www.openuri.org/2002/0
    <s:element name="StartHeader" type="conv:StartHeader"/>
    <s:element name="ContinueHeader" type="conv:ContinueHeader"/>
    <s:element name="CallbackHeader" type="conv:CallbackHeader"/>
    <s:complexType name="StartHeader">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="conversationID" type="s:string"/>
        <s:element minOccurs="0" maxOccurs="1" name="callbackLocation" type="s:string"/>
      </s:sequence>
    </s:complexType>
    <s:complexType name="ContinueHeader">
      <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="conversationID" type="s:string"/>
      </s:sequence>
    </s:complexType>
    <s:complexType name="CallbackHeader">
      <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="conversationID" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:schema>
</types>
<message name="onHelloResultSoapIn">
  <part name="parameters" element="s0:onHelloResultResponse"/>
</message>
<message name="onHelloResultSoapOut">
  <part name="parameters" element="s0:onHelloResult"/>
</message>
<message name="requestMessageSoapIn">
  <part name="parameters" element="s0:requestMessage"/>
</message>
<message name="requestMessageSoapOut">
  <part name="parameters" element="s0:requestMessageResponse"/>
</message>
<message name="checkStatusSoapIn">
  <part name="parameters" element="s0:checkStatus"/>
</message>
<message name="checkStatusSoapOut">
  <part name="parameters" element="s0:checkStatusResponse"/>
</message>
<message name="getMessageResponseSoapIn">
  <part name="parameters" element="s0:getMessageResponse"/>
</message>
<message name="getMessageResponseSoapOut">
  <part name="parameters" element="s0:getMessageResponseResponse"/>
</message>
<message name="onHelloResultHttpGetIn"/>
<message name="onHelloResultHttpGetOut">
  <part name="hello" type="s:string"/>
```

```
</message>
<message name="requestMessageHttpGetIn">
  <part name="useCallback" type="s:string"/>
</message>
<message name="requestMessageHttpGetOut"/>
<message name="checkStatusHttpGetIn"/>
<message name="checkStatusHttpGetOut">
  <part name="Body" element="s0:boolean"/>
</message>
<message name="getMessageResponseHttpGetIn"/>
<message name="getMessageResponseHttpGetOut">
  <part name="Body" element="s0:string"/>
</message>
<message name="onHelloResultHttpPostIn"/>
<message name="onHelloResultHttpPostOut">
  <part name="hello" type="s:string"/>
</message>
<message name="requestMessageHttpPostIn">
  <part name="useCallback" type="s:string"/>
</message>
<message name="requestMessageHttpPostOut"/>
<message name="checkStatusHttpPostIn"/>
<message name="checkStatusHttpPostOut">
  <part name="Body" element="s0:boolean"/>
</message>
<message name="getMessageResponseHttpPostIn"/>
<message name="getMessageResponseHttpPostOut">
  <part name="Body" element="s0:string"/>
</message>
<message name="StartHeader_literal">
  <part name="StartHeader" element="conv:StartHeader"/>
</message>
<message name="ContinueHeader_literal">
  <part name="ContinueHeader" element="conv:ContinueHeader"/>
</message>
<message name="CallbackHeader_literal">
  <part name="CallbackHeader" element="conv:CallbackHeader"/>
</message>
<portType name="HelloWorldAsyncSoap">
  <operation name="onHelloResult">
    <documentation>&lt;p&gt;HelloResult is the message we will send back to the client so
    <output message="s0:onHelloResultSoapOut"/>
    <input message="s0:onHelloResultSoapIn"/>
  </operation>
  <operation name="requestMessage">
    <documentation>&lt;p&gt;The client starts the interaction by calling HelloAsync.&lt;/
    <input message="s0:requestMessageSoapIn"/>
    <output message="s0:requestMessageSoapOut"/>
  </operation>
  <operation name="checkStatus">
    <documentation>Clients that don't want callbacks call this method to see if the resul
    <input message="s0:checkStatusSoapIn"/>
    <output message="s0:checkStatusSoapOut"/>
  </operation>
  <operation name="getMessageResponse">
    <documentation>Clients that don't want callbacks call this method to get the result</
    <input message="s0:getMessageResponseSoapIn"/>
    <output message="s0:getMessageResponseSoapOut"/>
  </operation>
</portType>
<portType name="HelloWorldAsyncHttpGet">
```

```
    <operation name="onHelloResult">
      <documentation>&lt;p&gt;HelloResult is the message we will send back to the client so
      <output message="s0:onHelloResultHttpGetOut"/>
      <input message="s0:onHelloResultHttpGetIn"/>
    </operation>
    <operation name="requestMessage">
      <documentation>&lt;p&gt;The client starts the interaction by calling HelloAsync.&lt;/
      <input message="s0:requestMessageHttpGetIn"/>
      <output message="s0:requestMessageHttpGetOut"/>
    </operation>
    <operation name="checkStatus">
      <documentation>Clients that don't want callbacks call this method to see if the resul
      <input message="s0:checkStatusHttpGetIn"/>
      <output message="s0:checkStatusHttpGetOut"/>
    </operation>
    <operation name="getMessageResponse">
      <documentation>Clients that don't want callbacks call this method to get the result</
      <input message="s0:getMessageResponseHttpGetIn"/>
      <output message="s0:getMessageResponseHttpGetOut"/>
    </operation>
  </portType>
  <portType name="HelloWorldAsyncHttpPost">
    <operation name="onHelloResult">
      <documentation>&lt;p&gt;HelloResult is the message we will send back to the client so
      <output message="s0:onHelloResultHttpPostOut"/>
      <input message="s0:onHelloResultHttpPostIn"/>
    </operation>
    <operation name="requestMessage">
      <documentation>&lt;p&gt;The client starts the interaction by calling HelloAsync.&lt;/
      <input message="s0:requestMessageHttpPostIn"/>
      <output message="s0:requestMessageHttpPostOut"/>
    </operation>
    <operation name="checkStatus">
      <documentation>Clients that don't want callbacks call this method to see if the resul
      <input message="s0:checkStatusHttpPostIn"/>
      <output message="s0:checkStatusHttpPostOut"/>
    </operation>
    <operation name="getMessageResponse">
      <documentation>Clients that don't want callbacks call this method to get the result</
      <input message="s0:getMessageResponseHttpPostIn"/>
      <output message="s0:getMessageResponseHttpPostOut"/>
    </operation>
  </portType>
  <binding name="HelloWorldAsyncSoap" type="s0:HelloWorldAsyncSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="onHelloResult">
      <soap:operation soapAction="http://www.openuri.org/onHelloResult" style="document"/>
      <cw:transition phase="finish"/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
        <soap:header wsdl:required="true" message="s0:CallbackHeader_literal" part="Callbac
      </output>
    </operation>
    <operation name="requestMessage">
      <soap:operation soapAction="http://www.openuri.org/requestMessage" style="document"/>
      <cw:transition phase="start"/>
      <input>
        <soap:body use="literal"/>
```

```
      <soap:header wsdl:required="true" message="s0:StartHeader_literal" part="StartHeade
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
  <operation name="checkStatus">
    <soap:operation soapAction="http://www.openuri.org/checkStatus" style="document"/>
    <cw:transition phase="continue"/>
    <input>
      <soap:body use="literal"/>
      <soap:header wsdl:required="true" message="s0:ContinueHeader_literal" part="Continu
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
  <operation name="getMessageResponse">
    <soap:operation soapAction="http://www.openuri.org/getMessageResponse" style="documen
    <cw:transition phase="finish"/>
    <input>
      <soap:body use="literal"/>
      <soap:header wsdl:required="true" message="s0:ContinueHeader_literal" part="Continu
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
<binding name="HelloWorldAsyncHttpGet" type="s0:HelloWorldAsyncHttpGet">
  <http:binding verb="GET"/>
  <operation name="onHelloResult">
    <http:operation location="/onHelloResult"/>
    <cw:transition phase="finish"/>
    <input>
      <mime:mimeXml part="Body"/>
    </input>
    <output>
      <http:urlEncoded/>
    </output>
  </operation>
  <operation name="requestMessage">
    <http:operation location="/requestMessage"/>
    <cw:transition phase="start"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output/>
  </operation>
  <operation name="checkStatus">
    <http:operation location="/checkStatus"/>
    <cw:transition phase="continue"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
  <operation name="getMessageResponse">
    <http:operation location="/getMessageResponse"/>
```

```
    <cw:transition phase="finish"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
</binding>
<binding name="HelloWorldAsyncHttpPost" type="s0:HelloWorldAsyncHttpPost">
  <http:binding verb="POST"/>
  <operation name="onHelloResult">
    <http:operation location="/onHelloResult"/>
    <cw:transition phase="finish"/>
    <input>
      <mime:mimeXml part="Body"/>
    </input>
    <output>
      <mime:content type="application/x-www-form-urlencoded"/>
    </output>
  </operation>
  <operation name="requestMessage">
    <http:operation location="/requestMessage"/>
    <cw:transition phase="start"/>
    <input>
      <mime:content type="application/x-www-form-urlencoded"/>
    </input>
    <output/>
  </operation>
  <operation name="checkStatus">
    <http:operation location="/checkStatus"/>
    <cw:transition phase="continue"/>
    <input>
      <mime:content type="application/x-www-form-urlencoded"/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
  <operation name="getMessageResponse">
    <http:operation location="/getMessageResponse"/>
    <cw:transition phase="finish"/>
    <input>
      <mime:content type="application/x-www-form-urlencoded"/>
    </input>
    <output>
      <mime:mimeXml part="Body"/>
    </output>
  </operation>
</binding>
<service name="HelloWorldAsync">
  <documentation>This originally asynchronous web service (copied and modified from Sampl
  <port name="HelloWorldAsyncSoap" binding="s0:HelloWorldAsyncSoap">
    <soap:address location="http://localhost:7001/controls/webservice/polling/asyncWebSer
  </port>
  <port name="HelloWorldAsyncHttpGet" binding="s0:HelloWorldAsyncHttpGet">
    <http:address location="http://localhost:7001/controls/webservice/polling/asyncWebSer
  </port>
  <port name="HelloWorldAsyncHttpPost" binding="s0:HelloWorldAsyncHttpPost">
    <http:address location="http://localhost:7001/controls/webservice/polling/asyncWebSer
  </port>
```

```
      </service>
    </definitions>
* ::
*/
```

Index.jsp Sample

This topic inludes the source code for the Index.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/polling/

Sample Source Code

```
<!--Generated by Weblogic Workshop-->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="netui-tags-databinding.tld" prefix="netui-data"%>
<%@ taglib uri="netui-tags-html.tld" prefix="netui"%>
<html>
  <head>
    <netui:base/>
    <link href="/WebApp/resources/css/style.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <jsp:include page="/resources/jsp/header.jsp"/>
    <h3>Calling an Asynchronous Web Service from a Page Flow</h3>

    <p>This sample demonstrates calling an asynchronous web service from a Page Flow.
    <p>To call an asynchronous web service, the Page Flow must "poll" the web service.
    Polling a web service involves three steps:
    <ol>
    <li>Invoking the asynchronous method on the web service
    <li>Waiting a period of time for the web service to process the request
    <li>Invoking the result collection method on the web service
    </ol>
    <p>Clicking the link below will start the polling process:
    <blockquote>

    <p><netui:anchor action="requestMessage.do">
      Request Message
    </netui:anchor>

    </blockquote>
    <hr>
    <p><netui:anchor href="/WebApp/controls/controlsController.jpf">Back to Control Samples</ne

  </body>
</html>
```

PollingController.jpf Sample

This topic inludes the source code for the PollingController.jpf Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/polling/

Sample Source Code

```
package controls.webservice.polling;

import com.bea.wlw.netui.pageflow.FormData;
import com.bea.wlw.netui.pageflow.Forward;
import com.bea.wlw.netui.pageflow.PageFlowController;

/**
 * PollingController.jpf demonstrates how a Page Flow can communicate with a
 * asynchronous web service by polling the web service.
 *
 * Polling a web service involves three steps:
 *
 * (1) Invoking the asynchronous method on the web service.
 * (2) Waiting a period of time for the web service to process the request.
 * (3) Invoking another method on the web service to collect the result.
 *
 * These three stages are encapsulated in the method requestMessage() below.
 *
 * This sample also shows how to modify a web service, originally designed to send callbacks,
 * so that non-callbackable clients can use the web service. A polling interface has been
 * added to the web service (HelloWorldAsync.jws) allowing this Page Flow (which cannot
 * hear callbacks from web services) to retrieve data from the web service. The web
 * service's polling interface exists alongside the original callback interface in
 * the same file.
 *
 * Note that the Workshop issues compiler warnings in this file, indicated by green underscorin
 * Workshop issues a warning anytime that a Page Flow declares a web service with a callback in
 * because callback interfaces cannot be used by Page Flows. In this case, the compiler warning
 * should be ignored since the web service contains *both* a callback
 * interface (which Page Flows cannot use) and a polling interface (which Page Flows can use).
 *
 * @jpf:view-properties view-properties::
 * <!-- This data is auto-generated. Hand-editing this section is not recommended. -->
 * <view-properties>
 * <pageflow-object id="pageflow:/controls/webservice/polling/PollingController.jpf"/>
 * <pageflow-object id="action:begin.do">
 *   <property value="60" name="x"/>
 *   <property value="60" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action:requestMessage.do">
 *   <property value="200" name="x"/>
 *   <property value="260" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="page:index.jsp">
 *   <property value="60" name="x"/>
```

```
 *   <property value="260" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="action-call:@page:result.jsp@#@action:begin.do@">
 *   <property value="164,130,130,96" name="elbowsX"/>
 *   <property value="52,52,52,52" name="elbowsY"/>
 *   <property value="West_1" name="fromPort"/>
 *   <property value="East_1" name="toPort"/>
 * </pageflow-object>
 * <pageflow-object id="page:result.jsp">
 *   <property value="200" name="x"/>
 *   <property value="60" name="y"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#index.jsp#@action:begin.do@">
 *   <property value="104,160,160,216" name="elbowsY"/>
 *   <property value="North_1" name="toPort"/>
 *   <property value="60,60,60,60" name="elbowsX"/>
 *   <property value="success" name="label"/>
 *   <property value="South_1" name="fromPort"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#success#result.jsp#@action:requestMessage.do@">
 *   <property value="216,160,160,104" name="elbowsY"/>
 *   <property value="South_1" name="toPort"/>
 *   <property value="200,200,200,200" name="elbowsX"/>
 *   <property value="success" name="label"/>
 *   <property value="North_1" name="fromPort"/>
 * </pageflow-object>
 * <pageflow-object id="forward:path#failure#index.jsp#@action:requestMessage.do@">
 *   <property value="164,130,130,96" name="elbowsX"/>
 *   <property value="252,252,252,252" name="elbowsY"/>
 *   <property value="West_1" name="fromPort"/>
 *   <property value="East_1" name="toPort"/>
 *   <property value="failure" name="label"/>
 * </pageflow-object>
 * <pageflow-object id="control:controls.webservice.polling.asyncWebService.HelloWorldAsyncCont
 *   <property value="28" name="x"/>
 *   <property value="34" name="y"/>
 * </pageflow-object>
 * </view-properties>
 * ::
 *
 */
public class PollingController extends PageFlowController
{
    /**
     * The green underscoring under 'myControl', indicates a compiler warning.
     * Note that this compiler warning is expected whenever you declare a callback-able
     * web service control on a page flow file.  (See the note above for details.)
     *
     * @common:control
     */
    private controls.webservice.polling.asyncWebService.HelloWorldAsyncControl myControl;

    /*
     * This member variable stores the message retreived from the web service.
     */
    public java.lang.String m_message = "";


    /**
     * @jpf:action
     * @jpf:forward name="success" path="index.jsp"
```

```
 */
protected Forward begin()
{
    return new Forward( "success" );
}


/**
 * This Action polls the HelloWorldAsync.jws web service.
 *
 * @jpf:action
 * @jpf:forward name="success" path="result.jsp"
 * @jpf:forward name="failure" path="index.jsp"
 */
public Forward requestMessage()
{
    try
    {
        /*
         * Request a message from the web service.
         */
        myControl.requestMessage(false);

        /*
         * Check 10 times to see if the message is ready to be retreived.
         * from the web service.
         */
        for(int i=0; i<10; i++)
        {
            /*
             * When the message is ready,
             * load the it into the member variable m_message, and
             * forward the user to the response.jsp page.
             */
            if(myControl.checkStatus() == true)
            {
                 m_message = myControl.getMessageResponse();
                 return new Forward( "success" );
            }
            else
            {
                /*
                 * Wait one second between checks.
                 */
                Thread.sleep(1000,0);
            }
        }

        /*
         * If, after 10 seconds, the message is not ready, load an error message into the m
         * variable m_message, and forward the user to the response.jsp page.
         */
        m_message = "The message was not received in the time allowed.";
        return new Forward( "failure" );

    }
    catch( Throwable ex )
    {
        ex.printStackTrace();
    }
    return null;
```

```
    }


}
```

Result.jsp Sample

This topic inludes the source code for the Result.jsp Sample.

Sample Location

This sample is located in the following directory in your WebLogic Workshop installation:

BEA_HOME/weblogic81/samples/workshop/SamplesApp/WebApp/controls/webservice/polling/

Sample Source Code

```
<%@page contentType="text/html;charset=UTF-8" language="java"%>
<%@taglib prefix="netui" uri="netui-tags-html.tld"%>
<html>
  <head>
    <title>Result from Asynchronous Web Service</title>
    <netui:base/>
    <link href="/WebApp/resources/css/style.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <jsp:include page="/resources/jsp/header.jsp"/>

    <blockquote>

        <p>Your message has arrived: </p>

        <p><netui:label style="font-weight:bold;" value="{pageFlow.m_message}"></netui:label></

    </blockquote>

    <hr>
    <p><netui:anchor action="begin">Re-run this sample</netui:anchor>
    <p><netui:anchor href="/WebApp/controls/controlsController.jpf">Back to Controls Samples</n

    </body>
</html>
```