JSF - Facelets Tags

JSF - template tags

Templates in a web application defines a common interface layout and style. For example, a same banner, logo in common header and copyright information in footer. JSF provides following facelets tags to provide a standard web interface layout.

S.N.	Tag & Description
1	**ui:insert** Used in template file. It defines contents to be placed in a template. ui:define tag can replaced its contents.
2	**ui:define** Defines the contents to be inserted in a template.
3	**ui:include** Includes contents of one xhtml page into another xhtml page.
4	**ui:composition** Loads a template using **template** attribute. It can also define a group of components to be inserted in xhtml page.

Creating Template

Creating template for a web application is a step-by-step procedure. Let's follow the following steps to create a sample template.

Step 1: Create Header file: header.xhtml

Use **ui:composition** tag to define a default content of Header section.

```
<ui:composition>
   <h1>Default Header</h1>
```

```
</ui:composition>
```

Step 2: Create Footer file: footer.xhtml

Use **ui:composition** tag to define a default content of Footer section.

```
<ui:composition>
   <h1>Default Footer</h1>
</ui:composition>
```

Step 3: Create Content file: contents.xhtml

Use **ui:composition** tag to define a default content of Content section.

```
<ui:composition>
   <h1>Default Contents</h1>
</ui:composition>
```

Step 4: Create a Template: common.xhtml

Use **ui:insert** and **ui:include** tag to include header/footer and content file in template file. Name each section in **ui:insert** tag.

name attribute of **ui:insert** tag will be used to replace contents of corresponding section.

```
<h:body>
   <ui:insert name="header" >
      <ui:include src="header.xhtml" />
   </ui:insert>
   <ui:insert name="content" >
      <ui:include src="contents.xhtml" />
   </ui:insert>
   <ui:insert name="footer" >
      <ui:include src="footer.xhtml" />
   </ui:insert>
</h:body>
```

Step 5a: Use Template with default contents: home.xhtml

load common.xhtml, a template using **ui:composition** tag in any xhtml page.

```
<h:body>
   <ui:composition template="common.xhtml">
</h:body>
```

Step 5b: Use Template and set own contents: home.xhtml

load common.xhtml, a template using **ui:composition** tag in any xhtml page. Use **ui:define**tag to override default values.

```
<h:body>
   <ui:composition template="templates/common.xhtml">
      <ui:define name="content">
         <h:link value="Page 1" outcome="page1" />
          
         <h:link value="Page 2" outcome="page2" />
      </ui:define>
   </ui:composition>
</h:body>
```

Example Application

Let us create a test JSF application to test the template tags in JSF.

Step	Description
1	Create a project with a name *helloworld* under a package *com.kore.test* as explained in the *JSF - First Application* chapter.
2	Create *templates* folder under *src* > *main* > *webapp* folder.

3	Create *header.xhtml*, *footer.xhtml*, *contents.xhtml* and *common.xhtml* files under*src* > *main* > *webapp* > *templates* folder. Modify them as explained below.
3	Create *page1.xhtml* and *page2.xhtml* files under *src* > *main* > *webapp* folder. Modify them as explained below.
4	Modify *home.xhtml* as explained below. Keep rest of the files unchanged.
5	Compile and run the application to make sure business logic is working as per the requirements.
6	Finally, build the application in the form of war file and deploy it in Apache Tomcat Webserver.
7	Launch your web application using appropriate URL as explained below in the last step.

header.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

   xmlns:ui="http://java.sun.com/jsf/facelets">

   <body>

      <ui:composition>

         <h1>Default Header</h1>

      </ui:composition>

   </body>

</html>
```

footer.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <body>
      <ui:composition>
         <h1>Default Footer</h1>
      </ui:composition>
   </body>
</html>
```

contents.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <body>
      <ui:composition>
         <h1>Default Content</h1>
      </ui:composition>
   </body>
</html>
```

common.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:head></h:head>
   <h:body>
      <div style="border-width:2px; border-color:green; border-style:solid;">
```

```
         <ui:insert name="header" >
            <ui:include src="/templates/header.xhtml" />
         </ui:insert>
      </div>
      <br/>
      <div style="border-width:2px; border-color:black; border-style:solid;">
         <ui:insert name="content" >
            <ui:include src="/templates/contents.xhtml" />
         </ui:insert>
      </div>
      <br/>
      <div style="border-width:2px; border-color:red; border-style:solid;">
         <ui:insert name="footer" >
            <ui:include src="/templates/footer.xhtml" />
         </ui:insert>
      </div>
   </h:body>
</html>
```

page1.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:body>
      <ui:composition template="templates/common.xhtml">
         <ui:define name="header">
            <h2>Page1 header</h2>
         </ui:define>
         <ui:define name="content">
            <h2>Page1 content</h2>
```

```
            <h:link value="Back To Home" outcome="home" />
         </ui:define>
         <ui:define name="footer">
            <h2>Page1 Footer</h2>
         </ui:define>
      </ui:composition>
   </h:body>
</html>
```

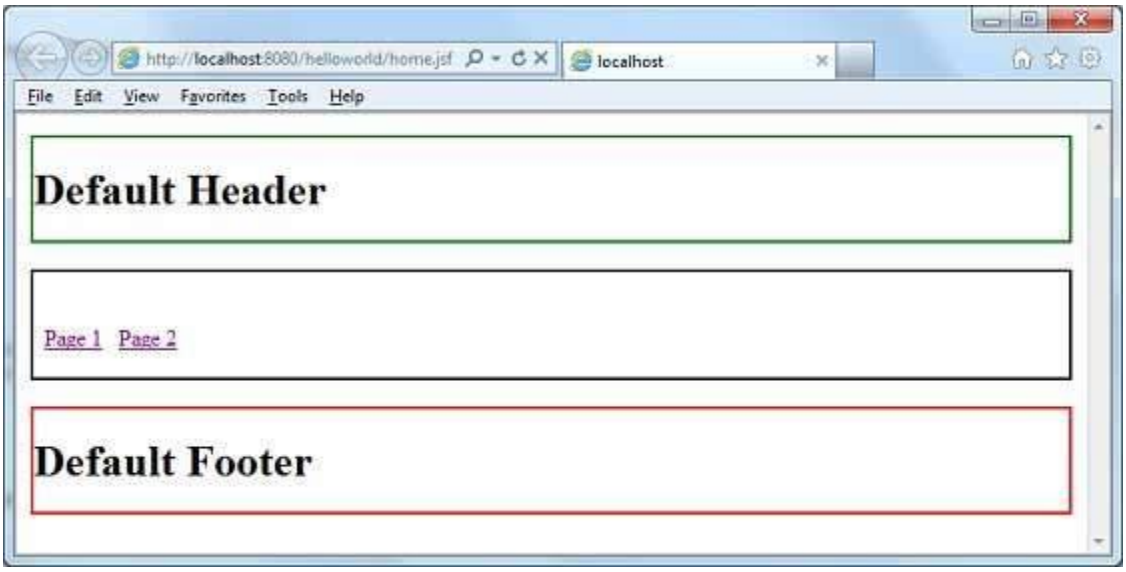
page2.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:body>
      <ui:composition template="templates/common.xhtml">
         <ui:define name="header">
            <h2>Page2 header</h2>
         </ui:define>
         <ui:define name="content">
            <h2>Page2 content</h2>
             <h:link value="Back To Home" outcome="home" />
         </ui:define>
         <ui:define name="footer">
            <h2>Page2 Footer</h2>
         </ui:define>
      </ui:composition>
   </h:body>
</html>
```

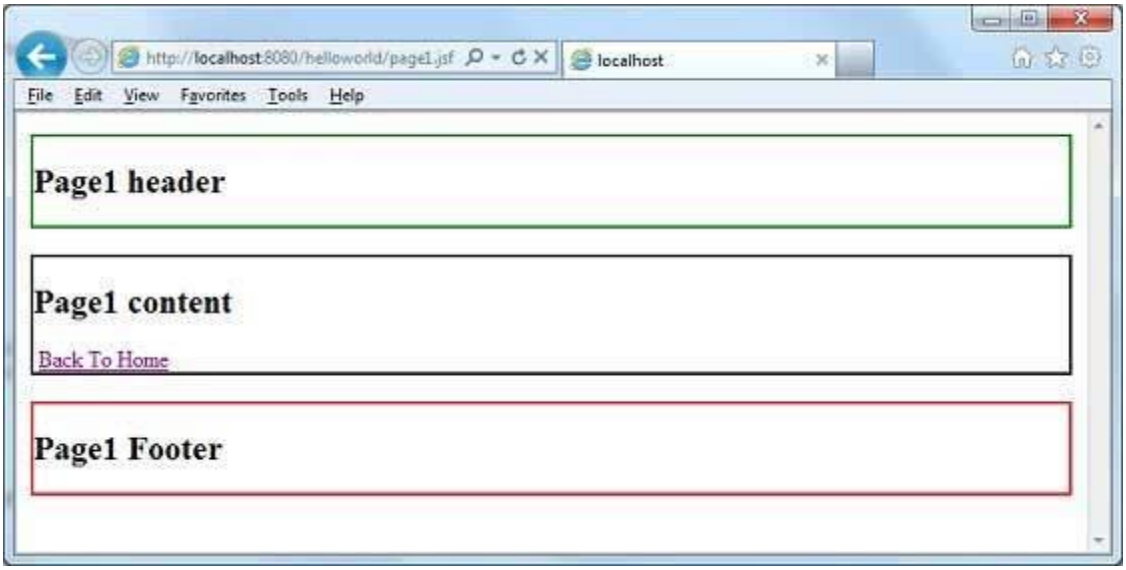
home.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:body>
      <ui:composition template="templates/common.xhtml">
         <ui:define name="content">
            <br/><br/>
             <h:link value="Page 1" outcome="page1" />
             <h:link value="Page 2" outcome="page2" />
            <br/><br/>
         </ui:define>
      </ui:composition>
   </h:body>
</html>
```

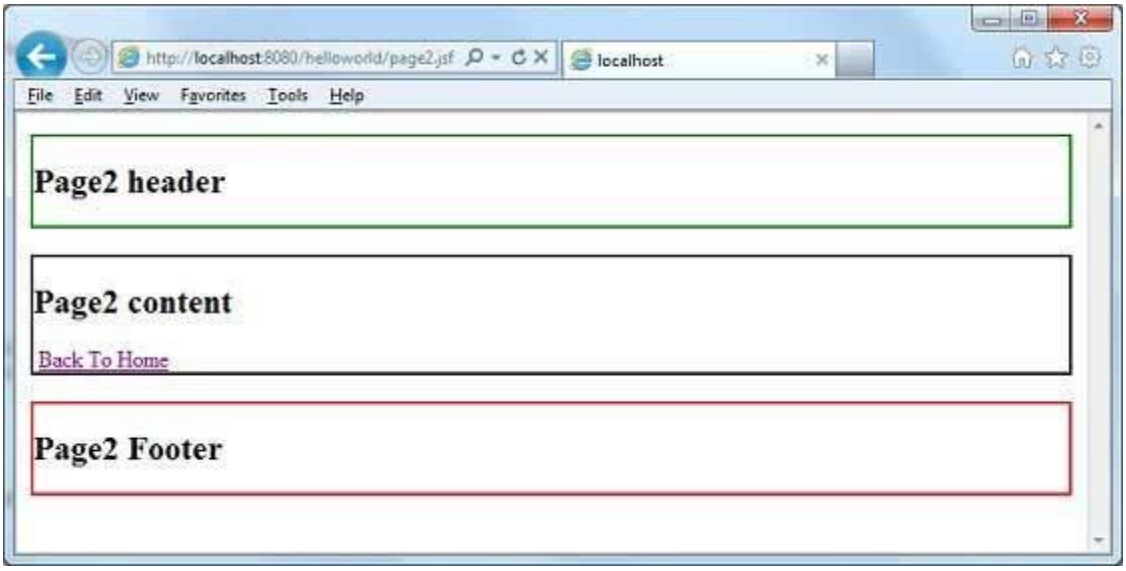
Once you are ready with all the changes done, let us compile and run the application as we did in JSF - First Application chapter. If everything is fine with your application, this will produce following result:



Click **Page1** link and you'll see the following result.



Or Click **Page2** link and you'll see the following result.



JSF - ui:param Tag

Using ui:param tag, we can pass parameters to template file or an included file.

In *JSF - template tags* chapter we've learned how to create and use template tags. We defined various section such as header, footer, content and a template combining all the sections.

Now we'll learn

- how to pass parameter(s) to various section of a template
- how to pass parameter(s) to a template

Parameter to section of a template

Create parameter : common.xhtml

Add parameter to ui:include tag.Use **ui:param** tag to define a parameter containing a value to be passed to Header section.

```
<ui:insert name="header" >
   <ui:include src="/templates/header.xhtml" >
      <ui:param name="defaultHeader" value="Default Header" />
   </ui:include>
</ui:insert>
```

Use parameter : header.xhtml

```
<ui:composition>
   <h1>#{defaultHeader}</h1>
</ui:composition>
```

Parameter to template

Create parameter : home.xhtml

Add parameter to ui:composition tag.Use **ui:param** tag to define a parameter containing a value to be passed to template.

```
<ui:composition template="templates/common.xhtml">
   <ui:param name="title" value="Home" />
</ui:composition>
```

Use parameter : common.xhtml

```
<h:body>

   <h2>#{title}</h2>

</h:body>
```

Example Application

Let us create a test JSF application to test the template tags in JSF.

Step	Description
1	Create a project with a name *helloworld* under a package *com.kore.test*as explained in the *JSF - Templates Tag* chapter.
2	Modify *header.xhtml*,and *common.xhtml* files under *src* > *main* > *webapp* > *templates* folder. Modify them as explained below.
3	Modify *home.xhtml* as explained below. Keep rest of the files unchanged.
4	Compile and run the application to make sure business logic is working as per the requirements.
5	Finally, build the application in the form of war file and deploy it in Apache Tomcat Webserver.
6	Launch your web application using appropriate URL as explained below in the last step.

header.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

   xmlns:ui="http://java.sun.com/jsf/facelets">
```

```
   <body>
      <ui:composition>
         <h1>#{defaultHeader}</h1>
      </ui:composition>
   </body>
</html>
```

common.xhtml

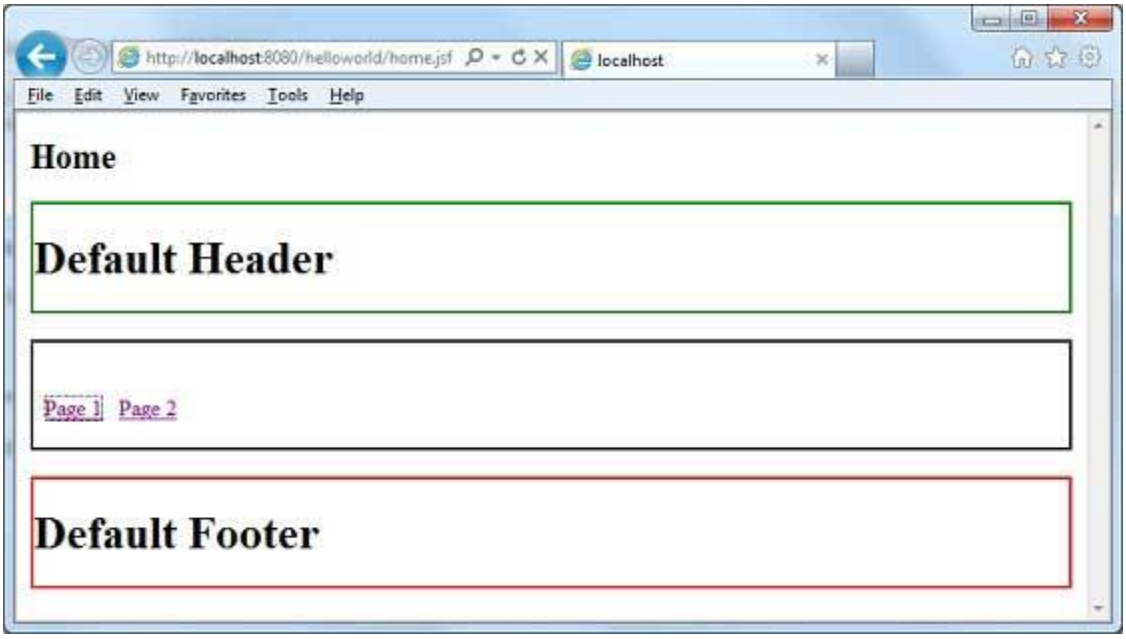
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:head></h:head>
      <h2>#{title}</h2>
      <div style="border-width:2px; border-color:green; border-style:solid;">
         <ui:insert name="header" >
            <ui:include src="/templates/header.xhtml" >
               <ui:param name="defaultHeader" value="Default Header" />
            </ui:include>
         </ui:insert>
      </div>
      <br/>
      <div style="border-width:2px; border-color:black; border-style:solid;">
         <ui:insert name="content" >
            <ui:include src="/templates/contents.xhtml" />
         </ui:insert>
      </div>
      <br/>
      <div style="border-width:2px; border-color:red; border-style:solid;">
         <ui:insert name="footer" >
            <ui:include src="/templates/footer.xhtml" />
```

```
         </ui:insert>
      </div>
   </h:body>
</html>
```

home.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   <h:body>
      <ui:composition template="templates/common.xhtml">
         <ui:param name="title" value="Home" />
         <ui:define name="content">
            <br/><br/>
             <h:link value="Page 1" outcome="page1" />
             <h:link value="Page 2" outcome="page2" />
            <br/><br/>
         </ui:define>
      </ui:composition>
   </h:body>
</html>
```

Once you are ready with all the changes done, let us compile and run the application as we did in JSF - First Application chapter. If everything is fine with your application, this will produce following result:


http://localhost:8080/helloworld/home.jsf
localhost
File Edit View Favorites Tools Help
Home
Default Header
Page 1 Page 2
Default Footer

JSF - Custom Tag

JSF provides developer a powerful capability to define own custom tags which can be used to render custom contents.

Defining a custom tag in JSF is a three step process

Step No.	Description
1a	Create a xhtml file and define contents in it using **ui:composition** tag
1b	Create a tag library descriptor (.taglib.xml file) and declares the above custom tag in it.
1c	Register the tag libray descriptor in web.xml

Step 1a: Define custom tag contents : buttonPanel.xhtml

```
<h:body>
   <ui:composition>
      <h:commandButton type="submit" value="#{okLabel}" />
      <h:commandButton type="reset" value="#{cancelLabel}" />
   </ui:composition>
</h:body>
```

Step 1b: Define a tag library : kore.taglib.xml

As name mentions a Tag library is a library of tags. Following table describes important attributes of a tag library.

S.N.	Node & Description
1	**facelet-taglib**

	Contains all the tags.
2	**namespace** Namespace of the tag library and should be unique.
3	**tag** **Contains a single tag**
4	**tag-name** Name of the tag
5	**source** tag implementation

```
<facelet-taglib>
   <namespace>http://kore.com/facelets</namespace>
   <tag>
      <tag-name>buttonPanel</tag-name>
      <source>com/kore/buttonPanel.xhtml</source>
   </tag>
</facelet-taglib>
```

Step 1c: Register the tag library :web.xml

```
<context-param>
   <param-name>javax.faces.FACELETS_LIBRARIES</param-name>
   <param-value>/WEB-INF/kore.taglib.xml</param-value>
</context-param>
```

Using a custom tag in JSF is a two step process

Step	Description

No.	
2a	Create a xhtml file and use custom tag library's namespace.
2b	Use the custom tag as normal JSF tags

Step 2a: Use Custom Namespace: home.xhtml

```
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets">
   xmlns:tp="http://kore.com/facelets">
```

Step 2b: Use Custom Tag: home.xhtml

```
<h:body>
   <tp:buttonPanel okLabel="Ok" cancelLabel="Cancel" />
</h:body>
```

Example Application

Let us create a test JSF application to test the template tags in JSF.

Step	Description
1	Create a project with a name *helloworld* under a package *com.kore.test*as explained in the *JSF - First Application* chapter.
2	Create *com* folder under *WEB-INF* directory.
3	Create *kore* folder under *WEB-INF* > *com* directory.
4	Create *buttonPanel.xhtml* file under *WEB-INF* > *com* > *kore* folder. Modify it as explained below.

5	Create *kore.taglib.xml* file under *WEB-INF* folder. Modify it as explained below.
6	Modify *web.xml* file under *WEB-INF* folder as explained below.
7	Modify *home.xhtml* as explained below. Keep rest of the files unchanged.
8	Compile and run the application to make sure business logic is working as per the requirements.
9	Finally, build the application in the form of war file and deploy it in Apache Tomcat Webserver.
10	Launch your web application using appropriate URL as explained below in the last step.

buttonPanel.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

   xmlns:h="http://java.sun.com/jsf/html"

   xmlns:ui="http://java.sun.com/jsf/facelets">

   <h:body>

      <ui:composition>

         <h:commandButton type="submit" value="#{okLabel}" />

         <h:commandButton type="reset" value="#{cancelLabel}" />

      </ui:composition>

   </h:body>

</html>
```

kore.taglib.xml

```
<?xml version="1.0"?>

<!DOCTYPE facelet-taglib PUBLIC
```

```
"-//Sun Microsystems, Inc.//DTD Facelet Taglib 1.0//EN"
"http://java.sun.com/dtd/facelet-taglib_1_0.dtd">
<facelet-taglib>
   <namespace>http://kore.com/facelets</namespace>
   <tag>
      <tag-name>buttonPanel</tag-name>
      <source>com/kore/buttonPanel.xhtml</source>
   </tag>
</facelet-taglib>
```

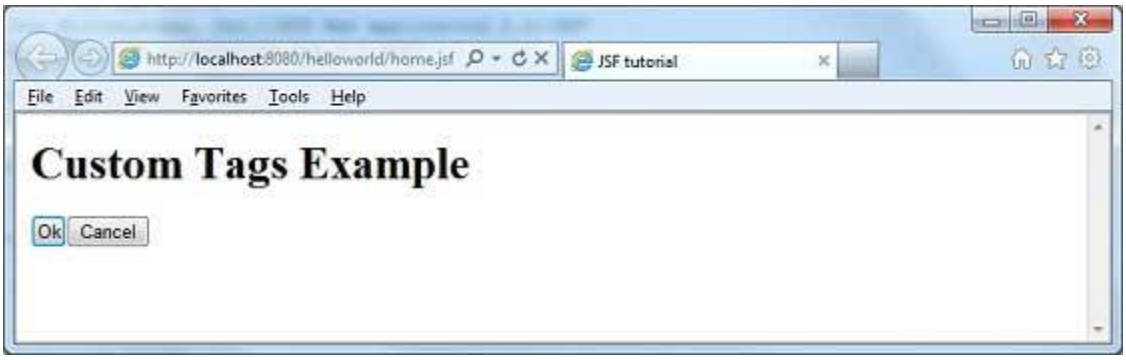
web.xml

```
<!DOCTYPE web-app PUBLIC
"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd" >
<web-app>
   <display-name>Archetype Created Web Application</display-name>
   <context-param>
      <param-name>javax.faces.PROJECT_STAGE</param-name>
      <param-value>Development</param-value>
   </context-param>
   <context-param>
      <param-name>javax.faces.FACELETS_LIBRARIES</param-name>
      <param-value>/WEB-INF/kore.taglib.xml</param-value>
   </context-param>
   <servlet>
      <servlet-name>Faces Servlet</servlet-name>
      <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
   </servlet>
   <servlet-mapping>
      <servlet-name>Faces Servlet</servlet-name>
      <url-pattern>*.jsf</url-pattern>
   </servlet-mapping>
</web-app>
```

home.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:ui="http://java.sun.com/jsf/facelets"
   xmlns:tp="http://kore.com/facelets">
   <h:head>
      <title>JSF tutorial</title>
   </h:head>
   <h:body>
      <h1>Custom Tags Example</h1>
      <tp:buttonPanel okLabel="Ok" cancelLabel="Cancel" />
   </h:body>
</html>
```

Once you are ready with all the changes done, let us compile and run the application as we did in JSF - First Application chapter. If everything is fine with your application, this will produce following result:



JSF - ui:remove Tag

ui:remove tag is used the prevent the JSF specific code to be rendered on client side. It is used especially to prevent commented out code to be rendered on client side.

JSF Tag commented out using HTML comment

```
<!-- JSF code commented out -->
<!--
<h:commandButton value="Ok" />
-->
```

Rendered Output

```
<!-- JSF code commented out -->
<!--
&lt;h:commandButton value=&quot;Ok&quot; /&gt;
-->
```

Now using remove tag we'll see the following change in rendered output.

JSF Tag commented out using remove tag

```
<!-- JSF code commented out -->
<ui:remove>
   <h:commandButton value="Ok" />
</ui:remove>
```

Rendered Output

```
<!-- JSF code commented out -->
```

Example Application

Let us create a test JSF application to test the template tags in JSF.

Step	Description
1	Create a project with a name *helloworld* under a package *com.kore.test*as explained in

	the *JSF - First Application* chapter.
2	Modify *home.xhtml* as explained below. Keep rest of the files unchanged.
3	Compile and run the application to make sure business logic is working as per the requirements.
4	Finally, build the application in the form of war file and deploy it in Apache Tomcat Webserver.
5	Launch your web application using appropriate URL as explained below in the last step.

home.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

   xmlns:h="http://java.sun.com/jsf/html"

   xmlns:ui="http://java.sun.com/jsf/facelets">

   <h:head>

      <title>JSF tutorial</title>

   </h:head>

   <h:body>

      <ui:remove>

         <h:commandButton value="Ok" />

      </ui:remove>

      <!--

         <h:commandButton value="Cancel" />

      -->

   </h:body>

</html>
```

Once you are ready with all the changes done, let us compile and run the application as we did in JSF - First Application chapter. If everything is fine with your application, you'll see an empty page.

View source of page and you will see the following html text.

home.jsf

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<title>JSF tutorial</title>
</head>
<body>
   <!--
      &lt;h:commandButton value=&quot;Cancel&quot; /&gt;
   -->
</body>
</html>
```