

Chapter 4 STD JSF Tags

1. Styles

➔ One can use styles, either inline(style) or classes(styleClass) to influence how components are rendered.

```
<h:outputText value="#{customer.name}" styleClass="emphasis"/>
```

```
<h:outputText value="#{customer.id}" style="border: thin solid blue"/>
```

➔ CSS style attributes can be value expressions that give programmatic control over style.

2. Resources

➔ One can place all style sheets, java script files, images and other files into 'resources' directory in the root of web app.

➔ Sub-directories of 'resources' directories are called libraries. One can create libraries as one like. Ex css, images, java script etc.

➔ To include style sheet use tag

```
<h:outputStylesheet library="css" name="styles.css"/>
```

The tag adds a link of the form

```
<link href="/context-root/faces/javax.faces.resource/styles.css?ln=css" rel="stylesheet" type="text/css"/>
```

➔ To include java script use tag <h:outputScript>

```
<h:outputScript name="jsf.js" library="javascript" target="head"/>
```

➔ Here 'target' could be 'head' or 'body'. The script is appended to the 'head' or 'body' facet of the root component, which means that it appears at the end of the head or body in the generated HTML. If there is no target the script will be inserted in the current location.

➔ To include image library use <h:graphicImage> tag

```
<h:graphicImage name="logo.png" library="images"/>
```

➔ One can provide version for library

Ex.

```
resources/css/1_0_2
```

```
resources/css/1_1
```

Then latest version resources/css/1_1 will be used.

➔ One can even provide version of file also where one need to replace the resource with directory of same name and then use version name as file name. Example is shown below

```
resources/css/styles.css/1_0_2.css
```

```
resources/css/styles.css/1_1.css
```

3. DHTML Events

➔ Dynamic HTML is supported by nearly all JSF HTML tags.

➔ Some DHTML attributes are shown below(there are many such attributes).

DHTML Event Attributes^a

Attribute	Description
onblur (16)	Element loses focus
onchange (11)	Element's value changes
onclick (17)	Mouse button is clicked over the element
ondblclick (21)	Mouse button is double-clicked over the element

4. Panels

- We have `<h:panelGrid>` tag which generates HTML markup for laying out components in rows and columns.
- If we have 4 components and `<h:panelGrid>` specified with 2 columns then one will have two rows with 2 components each.

→ Ex

1. `<h:panelGrid columns="2">`



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/MessagePjt/` and the page title "login Page". The main content area has the heading "Welcome to Login Page". Below the heading, there are two input fields: "Enter Your Name :" followed by a text input field, and "Enter Your EmpID :" followed by a text input field containing the value "0". A "submit" button is located below the input fields.

2. `<h:panelGrid columns="3">`



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/MessagePjt/` and the page title "login Page". The main content area has the heading "Welcome to Login Page". Below the heading, there are two input fields side-by-side: "Enter Your Name :" followed by a text input field, and "Enter Your EmpID :" followed by a text input field containing the value "0". A "submit" button is located below the input fields.

➔ <h:panelGrid> is often used with <h:panelGroup> with which two or more components are grouped and then they are treated as one

```
<h:panelGrid columns="1">
  <h:panelGroup>
    <h:inputText id='name' value=#{user.name}'>
    <h:message for="name"/>
  </h:panelGroup>
</h:panelGrid>
```

➔ If no. of columns are not mentioned then by default it is '1'.

5. Text Fields and Text Areas




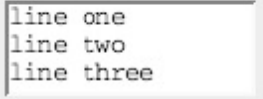
➔ JSF supports three varieties of text inputs

```
h:inputText
h:inputSecret
h:inputTextarea
```

h:inputText and h:inputSecret Examples	
Example	Result
<code><h:inputText value="#{form.testString}" readonly="true"/></code>	<input type="text" value="12345678901234567890"/>
<code><h:inputSecret value="#{form.passwd}" redisplay="true"/></code>	<input type="password" value="*****"/> (shown after an unsuccessful form submit)
<code><h:inputSecret value="#{form.passwd}" redisplay="false"/></code>	<input type="password"/> (shown after an unsuccessful form submit)
<code><h:inputText value="inputText" style="color: Yellow; background: Teal;"/></code>	<input type="text" value="inputText"/>
<code><h:inputText value="1234567" size="5"/></code>	<input type="text" value="123456"/>
<code><h:inputText value="1234567890" maxlength="6" size="10"/></code>	<input type="text" value="123456"/>

Note: 'size' attribute specifies the number of visible characters in a text field. But we can see in example 5 where size=5 is defined still 123456 i.e. 6 characters appears in text field. Hence 'size' attribute is not precise where 'maxlength' attribute is precise and gives perfect result as shown in last example.

➔ <h:inputTextArea> examples

h:inputTextArea Examples	
Example	Result
<code><h:inputTextArea rows="5"/></code>	
<code><h:inputTextArea cols="5"/></code>	
<code><h:inputTextArea value="123456789012345" rows="3" cols="10"/></code>	
<code><h:inputTextArea value="#{form.dataInRows}" rows="2" cols="15"/></code>	

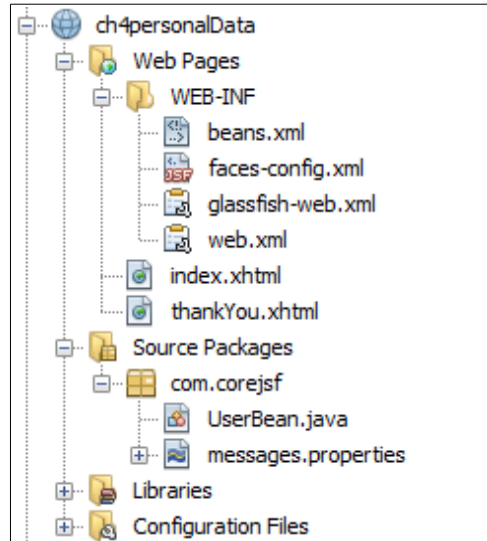
➔ We can see in third example if we place long string as 'value' then the whole string will be displayed in one line. Also to keep data in separate lines one can insert new line character(/n) to force a line break. In last example dataInRows property of backing bean is implemented as shown below.

```
private String dataInRows = "line one\nline two\nline three";  
public void setDataInRows(String newValue) {  
    dataInRows = newValue;  
}  
public String getDataInRows() {  
    return dataInRows;  
}
```

➔ Example for text field and text area tags

```
h:inputText  
h:inputSecret  
h:inputTextArea
```

Lets create directory for project 'ch4personalData' as shown below



UserBean.java

```
package com.corejsf;

import java.io.Serializable;

import javax.inject.Named;
// or import javax.faces.bean.ManagedBean;
import javax.enterprise.context.SessionScoped;
// or import javax.faces.bean.SessionScoped;

@Named("user") // or @ManagedBean(name="user")
@SessionScoped
public class UserBean implements Serializable {
    private String name;
    private String password;
    private String aboutYourself;

    public String getName() { return name; }
    public void setName(String newValue) { name = newValue; }

    public String getPassword() { return password; }
    public void setPassword(String newValue) { password = newValue; }

    public String getAboutYourself() { return aboutYourself; }
    public void setAboutYourself(String newValue) { aboutYourself = newValue; }
}
```

messages.properties

indexWindowTitle=Using Textfields and Textareas

thankYouWindowTitle=Thank you for submitting your information

thankYouPageTitle=Thank you!

indexPageTitle=Please enter the following personal information

namePrompt=Name:

passwordPrompt=Password:

tellUsPrompt=Please tell us about yourself:

aboutYourselfPrompt=Some information about you:

submitPrompt=Submit your information

index.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">
  <h:head>
    <title>#{msgs.indexWindowTitle}</title>
  </h:head>
  <h:body>
    <h:outputText value="#{msgs.indexPageTitle}"
                  style="font-style: italic; font-size: 1.5em"/>
    <h:form>
      <h:panelGrid columns="2">
        <h:inputText value="#{user.name}"/>
        <h:inputSecret value="#{user.password}"/>
        <h:inputTextarea value="#{user.aboutYourself}" rows="5" cols="35"/>
      </h:panelGrid>
      <h:commandButton value="#{msgs.submitPrompt}" action="thankYou"/>
    </h:form>
  </h:body>
</html>
```

thankYou.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">
  <h:head>
    <title>#{msgs.thankYouWindowTitle}</title>
  </h:head>
  <h:body>
    <h:outputText value="#{msgs.namePrompt}" style="font-style: italic"/>
    <h:outputText value="#{user.name}" style="font-style: italic"/>
    <br/>
    <h:outputText value="#{msgs.aboutYourselfPrompt}" style="font-style: italic"/>
    <h:outputText value="#{user.aboutYourself}" style="font-style: italic"/>
    <br/>
    <pre>#{user.aboutYourself}</pre>
  </h:body>
</html>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
                             http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         version="2.5">
  <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>/faces/*</url-pattern>
  </servlet-mapping>
</web-app>
```

```

</servlet-mapping>
<welcome-file-list>
  <welcome-file>faces/index.xhtml</welcome-file>
</welcome-file-list>
<context-param>
  <param-name>javax.faces.PROJECT_STAGE</param-name>
  <param-value>Development</param-value>
</context-param>
</web-app>

```

faces-config.xml

```

<?xml version="1.0"?>
<faces-config xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-facesconfig_2_0.xsd"
  version="2.0">
  <application>
    <resource-bundle>
      <base-name>com.corejsf.messages</base-name>
      <var>msgs</var>
    </resource-bundle>
  </application>
</faces-config>

```

Output

index.xhtml

Please enter the following personal information

Name:

Password:

Please tell us about yourself:

thankYou.xhtml

Name: Jayant

Some information about you:

I am software professional