

JSF - H:SELECTONERADIO

http://www.tutorialspoint.com/jsf/jsf_selectoneradio_tag.htm

Copyright © tutorialspoint.com

The h:selectOneRadio tag renders a set of HTML input element of type "radio", and format it with HTML table and label tags.

JSF Tag

```
<h:selectOneRadio value="#{userData.data}">
  <f:selectItem itemValue="1" itemLabel="Item 1" />
  <f:selectItem itemValue="2" itemLabel="Item 2" />
</h:selectOneRadio>
```

Rendered Output

```
<table>
  <tr>
    <td><input type="radio" checked="checked" name="j_idt6:j_idt8"
      />
      <label for="j_idt6:j_idt8:0"> Item 1</label></td>
    <td><input type="radio" name="j_idt6:j_idt8"
      />
      <label for="j_idt6:j_idt8:1"> Item 2</label></td>
  </tr>
</table>
```

Tag Attributes

S.N.	Attribute & Description
1	id Identifier for a component
2	binding Reference to the component that can be used in a backing bean
3	rendered A boolean; false suppresses rendering
4	styleClass Cascading stylesheet CSS class name
5	value A component's value, typically a value binding
6	valueChangeListener A method binding to a method that responds to value changes
7	converter Converter class name

8	validator	Class name of a validator that's created and attached to a component
9	required	A boolean; if true, requires a value to be entered in the associated field
10	accesskey	A key, typically combined with a system-defined metakey, that gives focus to an element
11	accept	Comma-separated list of content types for a form
12	accept-charset	Comma- or space-separated list of character encodings for a form. The accept-charset attribute is specified with the JSF HTML attribute named acceptcharset .
13	alt	Alternative text for nontextual elements such as images or applets
14	charset	Character encoding for a linked resource
15	coords	Coordinates for an element whose shape is a rectangle, circle, or polygon
16	dir	Direction for text. Valid values are ltr <i>lefttoright</i> and rtl <i>righttoleft</i> .
17	disabled	Disabled state of an input element or button
18	hreflang	Base language of a resource specified with the href attribute; hreflang may only be used with href .
19	lang	Base language of an element's attributes and text
20	maxlength	Maximum number of characters for text fields
21	readonly	

Read-only state of an input field; text can be selected in a readonly field but not edited

22 **rel**

Relationship between the current document and a link specified with the **href** attribute

23 **rev**

Reverse link from the anchor specified with **href** to the current document. The value of the attribute is a space-separated list of link types.

24 **rows**

Number of visible rows in a text area. **h:dataTable** has a **rows** attribute, but it's not an HTML pass-through attribute.

25 **shape**

Shape of a region. Valid values: **default, rect, circle, poly**. *default signifies the entire region*

26 **style**

Inline style information

27 **tabindex**

Numerical value specifying a tab index

28 **target**

The name of a frame in which a document is opened

29 **title**

A title, used for accessibility, that describes an element. Visual browsers typically create tooltips for the title's value

30 **type**

Type of a link; for example, **stylesheet**

31 **width**

Width of an element

32 **onblur**

Element loses focus

33 **onchange**

Element's value changes

34 **onclick**

Mouse button is clicked over the element

35 **ondblclick**
 Mouse button is double-clicked over the element

36 **onfocus**
 Element receives focus

37 **onkeydown**
 Key is pressed

38 **onkeypress**
 Key is pressed and subsequently released

39 **onkeyup**
 Key is released

40 **onmousedown**
 Mouse button is pressed over the element

41 **onmousemove**
 Mouse moves over the element

42 **onmouseout**
 Mouse leaves the element's area

43 **onmouseover**
 Mouse moves onto an element

44 **onmouseup**
 Mouse button is released

45 **onreset**
 Form is reset

46 **onselect**
 Text is selected in an input field

47 **disabledClass**
 CSS class for disabled elements

48 **enabledClass**
 CSS class for enabled elements

49 layout

Specification for how elements are laid out: `lineDirection` *horizontal* or `pageDirection` *vertical*

50 border

border of the element

Example Application

Let us create a test JSF application to test the above tag.

Step	Description
1	Create a project with a name <i>helloworld</i> under a package <i>com.tutorialspoint.test</i> as explained in the <i>JSF - First Application</i> chapter.
2	Modify <i>home.xhtml</i> as explained below. Keep rest of the files unchanged.
3	Create <i>result.xhtml</i> in the webapps directory as explained below.
4	Create <i>UserData.java</i> as a managed bean under package <i>com.tutorialspoint.test</i> as explained below.
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.

UserData.java

```
package com.tutorialspoint.test;

import java.io.Serializable;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

@ManagedBean(name = "userData", eager = true)
@SessionScoped
public class UserData implements Serializable {

    private static final long serialVersionUID = 1L;

    public String data = "1";

    public String getData() {
        return data;
    }

    public void setData(String data) {
        this.data = data;
    }
}
```

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:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
  <head>
    <title>JSF Tutorial!</title>
  </head>
  <h:body>
    <h2>h:selectManyCheckbox example</h2>
    <hr />
    <h:form>
      <h3>Radio Button</h3>
      <h:selectOneRadio value="#{userData.data}">
        <f:selectItem itemValue="1" itemLabel="Item 1" />
        <f:selectItem itemValue="2" itemLabel="Item 2" />
        <f:selectItem itemValue="3" itemLabel="Item 3" />
        <f:selectItem itemValue="4" itemLabel="Item 4" />
        <f:selectItem itemValue="5" itemLabel="Item 5" />
      </h:selectOneRadio>
      <h:commandButton value="Submit" action="result" />
    </h:form>
  </h:body>
</html>

```

result.xhtml

```

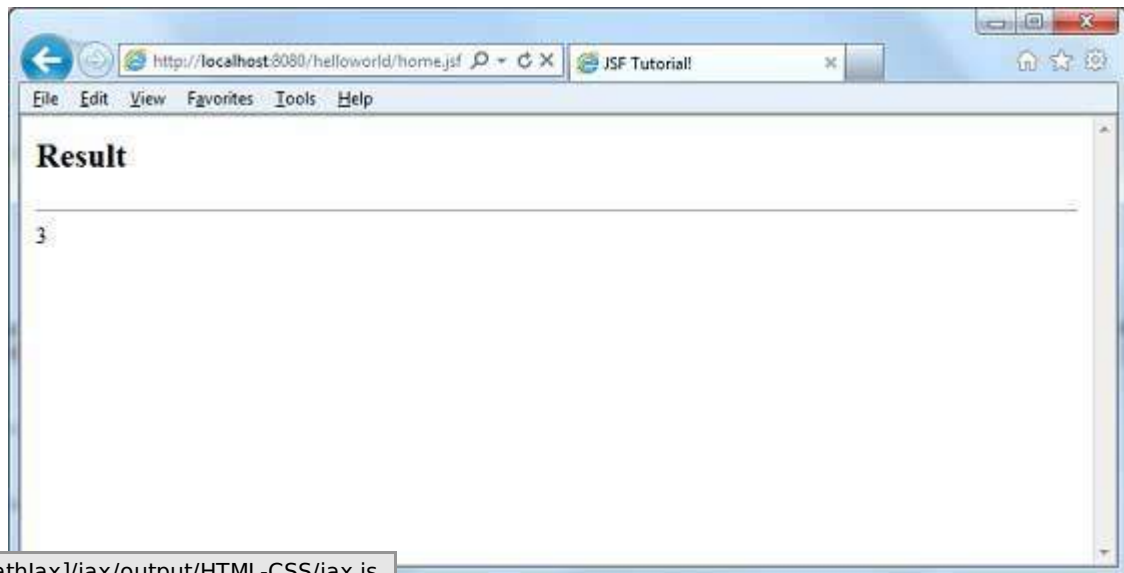
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ui="http://java.sun.com/jsf/facelets">
  <head>
    <title>JSF Tutorial!</title>
  </head>
  <h:body>
    <h2>Result</h2>
    <hr />
    #{userData.data}
  </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:



Select any option and press **Submit** button. We've selected item 3. You will see the selected results.



Loading [Mathjax]/jax/output/HTML-CSS/jax.js