**Question 1**

We learned that all HTTP requests for a JSF application are handled by the FacesServlet.

For example, if you sent an HTTP request from the browser with URL:

**http://localhost:8080/MyJSFApp/main.xhtml** , then this HTTP request will execute the

service() method of the FacesServlet. Explain clearly how the web container knows for which

HTTTP requests it should call the service() method of the FacesServlet.

In web.xml we specify which url’s should be handled by the FacesServlet

**<servlet>**

**<servlet-name>Faces Servlet</servlet-name>**

**<servlet-class>javax.faces.webapp.FacesServlet</servlet-class>**

**<load-on-startup>1</load-on-startup>**

**</servlet>**

**<servlet-mapping>**

**<servlet-name>Faces Servlet</servlet-name>**

**<url-pattern>\*.xhtml</url-pattern>**

**</servlet-mapping>**

**Question 2**

Explain clearly the difference between action, actionlistener and valuechangelistener

attributes. When would you use which attribute?

Action: when you want to execute some code when you click a button or a hyperlink.

Action is also used for navigation.

Actionlistener: when you want to execute some code when you click a button or a

hyperlink. You also have access to the action event, so you can use send parameters.

Cannot be used for navigation.

Valuechangelistener: : when you want to execute some code when the value of a

control changes.

**Question 3**

b) Explain clearly what redirection means with respect to JSF navigation. Give the

advantages and disadvantages of redirection in JSF.

With redirection, the server send a response to the browser to change the URL. The

Browser changes the URL and sends a new request to the server.

Advantage: the URL is always correct.

Disadvantage: extra roundtrip and we loose the original request values.

**Question 4**

We learned that all HTTP requests for a JSF application are handled by the

FacesServlet. This FacesServlet will start a sequence of steps that are executed so

that everything works correctly. We call this the JSF lifecycle. Explain clearly the

different steps that are started by the FacesServlet, and explain the work that is done

in every step.

**Phase Specific task of this phase**

1. **Restore view Create/retrieve component tree**
2. **Apply request The values of the components in the component tree are**

**updated with the values from the request**

1. **Process validations Validate the values in the component tree**

**Update model values Find the backin**

**gbean and update the values in the backing**

**Bean**

1. **Invoke application Call the registered action listeners**
2. **Render response Create the Http response**

**Question 7**

Explain clearly the meaning of the following 4 different facelets tags:

a) ui:insert

b) ui:define

c) ui:include

d) ui:composition

1. **ui:insert** Inserts content into a template. That content is defined inside the

tag that loads the template.

1. **ui:define** Defines content that is inserted into a template with a matching

ui:insert.

1. **ui:include** Includes content from another XML file.
2. **ui:composition** When used without a template attribute, a composition is a

sequence of elements that can be inserted somewhere else. The

composition can have variable parts (specified with ui:insert

children).

**Question 8**

a. Suppose you need to validate an input field of a form to check if it is a number

between 90 and 100. How would you do that? Give some example code that shows

how this will work.

**<h:inputText id=*"inputNumber" value="#{validationBackingBean.number}">***

**<f:validateLongRange maximum=*"10" minimum="0"/>***

**</h:inputText>**

**Question 9**

b. Suppose you need to validate an input field of a form to check if the entered text

starts with 3 digits followed by a forward slash and then followed by 3 letters. How

would you do that? Give some example code that shows how this will work.

**<h:inputText id=*"phone" value="#{validationBackingBean.text}">***

**<f:validateRegex pattern=*"(([0-9]{3})/[a-z]{3})" />***

**</h:inputText>**

**Question 10**

c. Suppose you need to convert the entered text of an input field of a form to a

java.util.Date object. How would you do that? Give some example code that shows

how this will work.

**<h:inputText value="#{dateBackingBean.dateofbirth}">**

**<f:convertDateTime pattern=*"dd/MM/yyyy"/>***

**</h:inputText>**

**Question 11**

d. Suppose you need to convert the entered text of an input field of a form to a domain

specific object, for example a phone number class. You need to use this conversion

logic at a number of different places in your application. How would you do that?

Give some example code that shows how this will work.

**@FacesConverter( value="phoneConverter" )**

**public class Phoneconverter implements Converter {**

**public Object getAsObject(FacesContext context, UIComponent component, String**

**value) {**

**if ("".equals(value)){**

**return null;**

**}**

**PhoneNumber phone = new PhoneNumber();**

**String countryCode = value.substring(value.indexOf('(')+1,value.indexOf(')'));**

**String areaCode = value.substring(value.indexOf(')')+1,value.indexOf('-'));**

**String number = value.substring(value.indexOf('-')+1,value.length());**

**phone.setCountryCode(countryCode);**

**phone.setAreaCode(areaCode);**

**phone.setNumber(number);**

**return phone;}**

**public String getAsString(FacesContext context, UIComponent component, Object**

**value) {**

**return value.toString();**

**}**

**}**

**<h:inputText value="#{phoneBackingBean.phonenumber}">**

**<f:converter converterId=*"phoneConverter"/>***

**</h:inputText>**