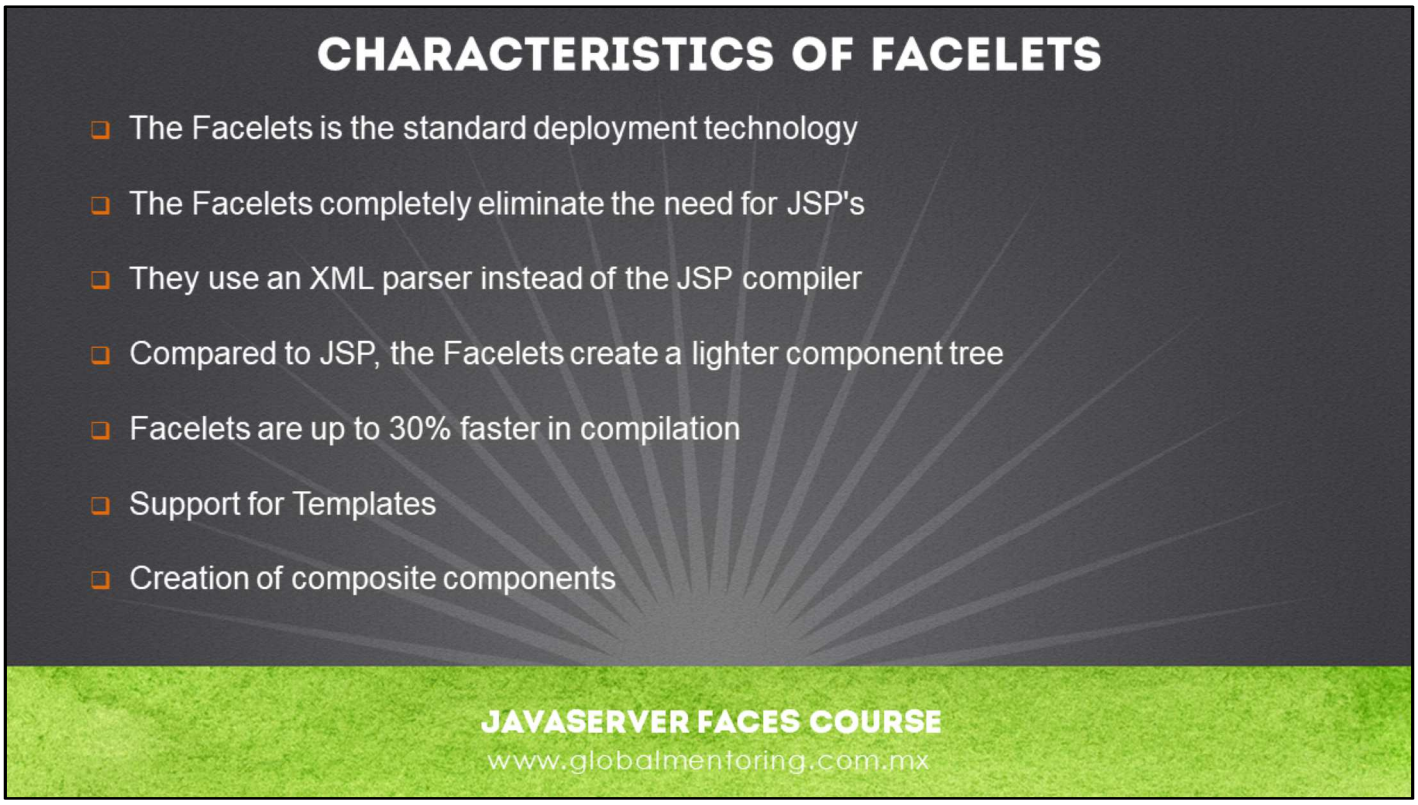
Hello, Ubaldo Acosta greets you again. I hope you're ready to start with this lesson ..

We are going to study the topic of Facelets in JavaServer Faces.

Are you ready? Let's go!



The Facelets is a new technology that takes into account the life cycle of the JSF, so from the JSF 2.0 version is the deployment standard.

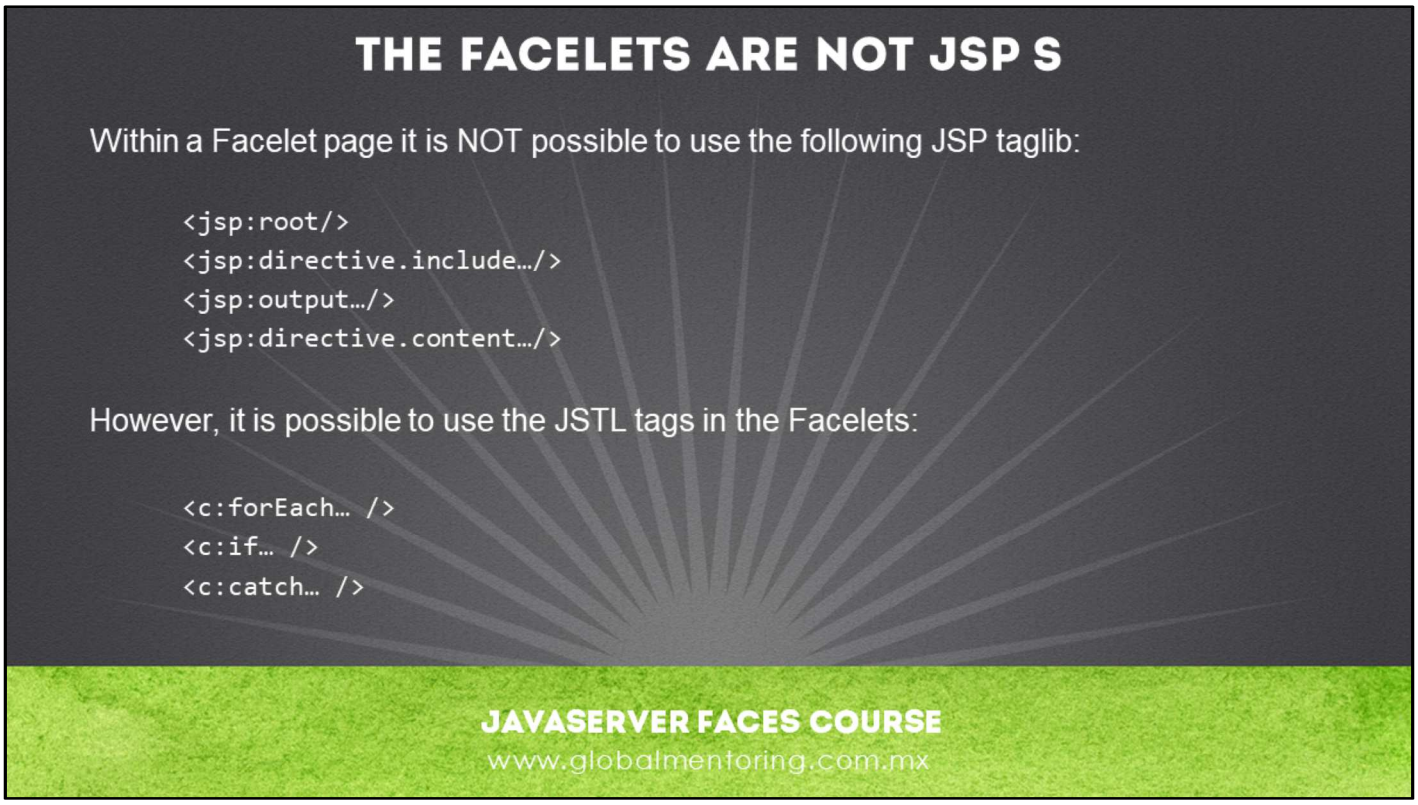
The Facelets are much more precise when it comes to indicating errors in our pages. They also allow easy access to Managed Beans with the help of EL (Expression Language) from any part of the page.

A new Facelets library allows you to create components more easily:

```
<ui:include />
<ui:composition />
<ui:define />
<ui:decorate />
<ui:repeat />
```

The Facelets allow the dynamic management of the component tree, both to add or remove components. For example:

```
<ui:include src=”#{bean.filePath}” />
```



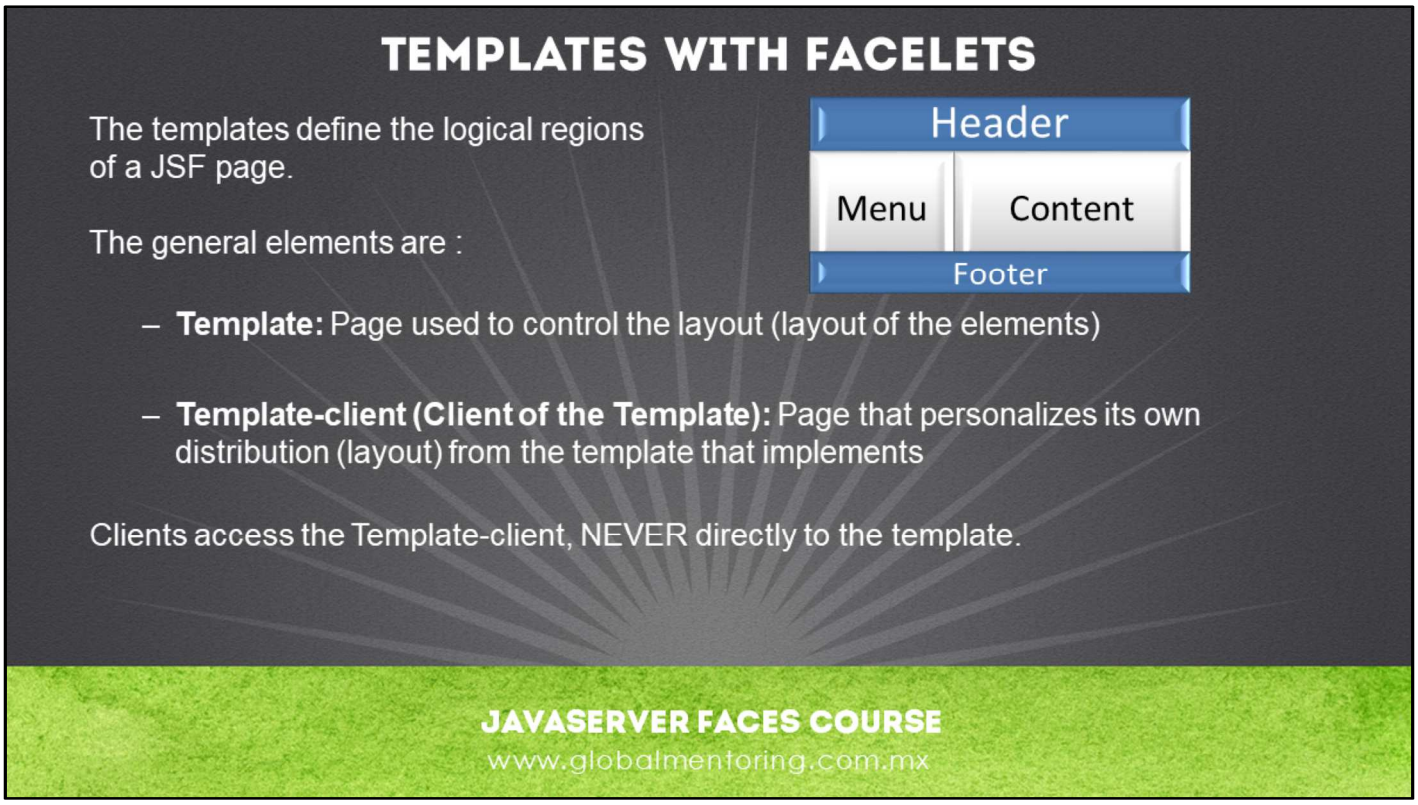
Within a Facelet page it is NOT possible to use the following JSP taglib:

```
<jsp:root/>
<jsp:directive.include…/>
<jsp:output…/>
<jsp:directive.content…/>
```

This is due to the fact that the translation mechanism of the JSP's and their life cycle are different from those of the Facelets.

However, it is possible to use the JSTL tags in the Facelets:

```
<c:forEach… />
<c:if… />
<c:catch… />
```



The Facelets provide an intuitive mechanism to define logical regions in a JSF page. They are used to separate the layout of the page and the data of the same

Template:

The logical areas of the page are defined using **<ui:include name="..." />**

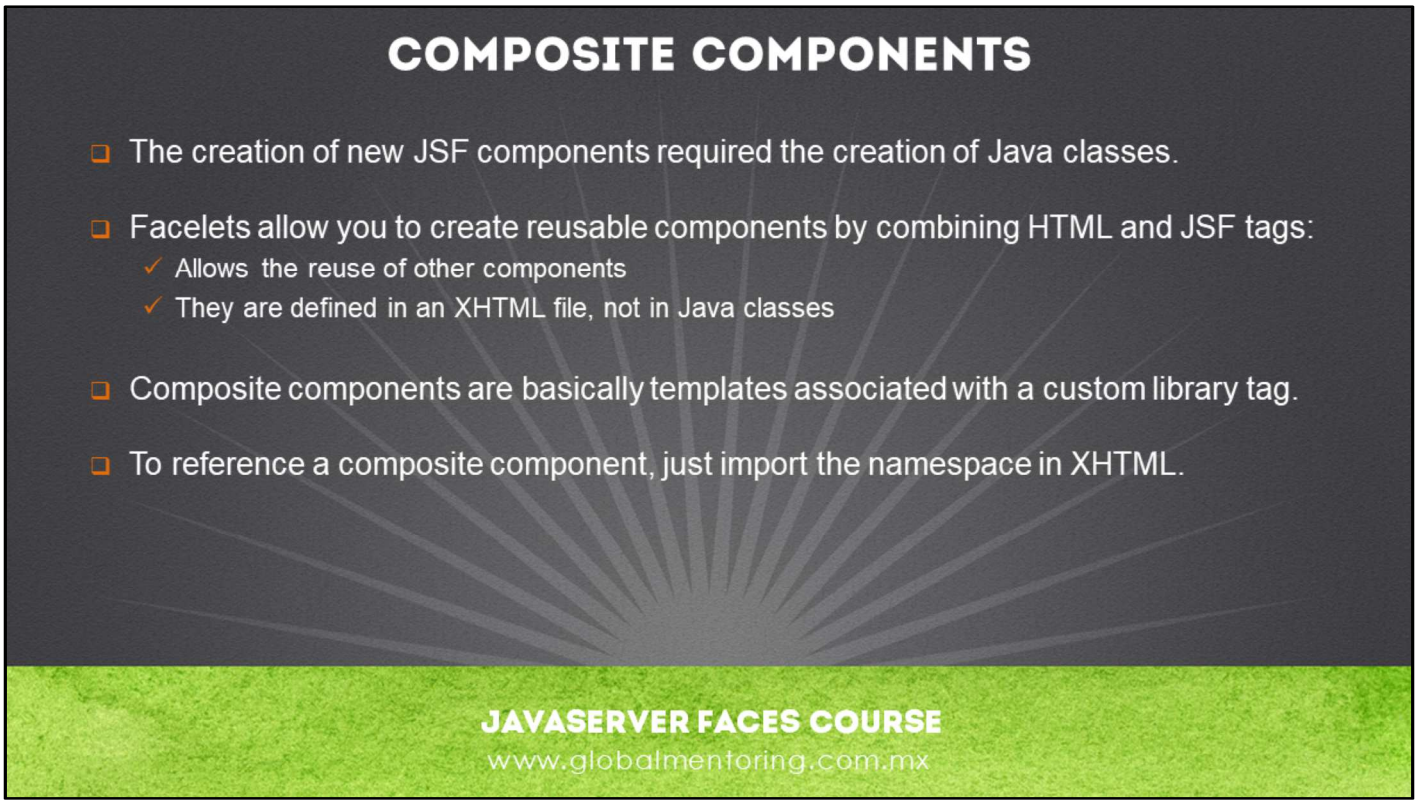
The default content can be added as internal elements to

<ui:include />

Template-client:

The pages that use the templates must modify the regions to be customized. In case of not modifying them, the default values from the template will be used.

The template-client defines which template will be used as a base and personalizes the elements with the tag **<ui:define />**



The main idea of creating composite components is reuse. For example, if we consider a code similar to this one:

```
<h:column>
 <f:facet name="header">
          <h:outputText value="#{msgs.title}" />
 </f:facet>
 <h:outputText value="#{book.title}" />
</h:column>
```

By creating several columns of the style shown in a table, the XHTML file would grow too large. The composite components allow us to encapsulate this complexity and reduce the code on our pages with Facelets. The result would be similar to:

```
<myLib:column columnTitle="#{msgs.title}"  columnValue="#{book.title}">
```

In broad strokes to achieve this code, we must create a facelet library and create the respective .jar file, finally add the namespace in the Facelet page.