Composite Components

Composite Components

- JavaServer Faces technology offers the concept of composite components with Facelets. A composite component is a special type of template that acts as a component.

Composite Components

- A composite component *consists of a collection of markup tags and other existing components.*

- This reusable, user-created component has a customized, defined functionality and can have validators, converters, and listeners attached to it like any other component.
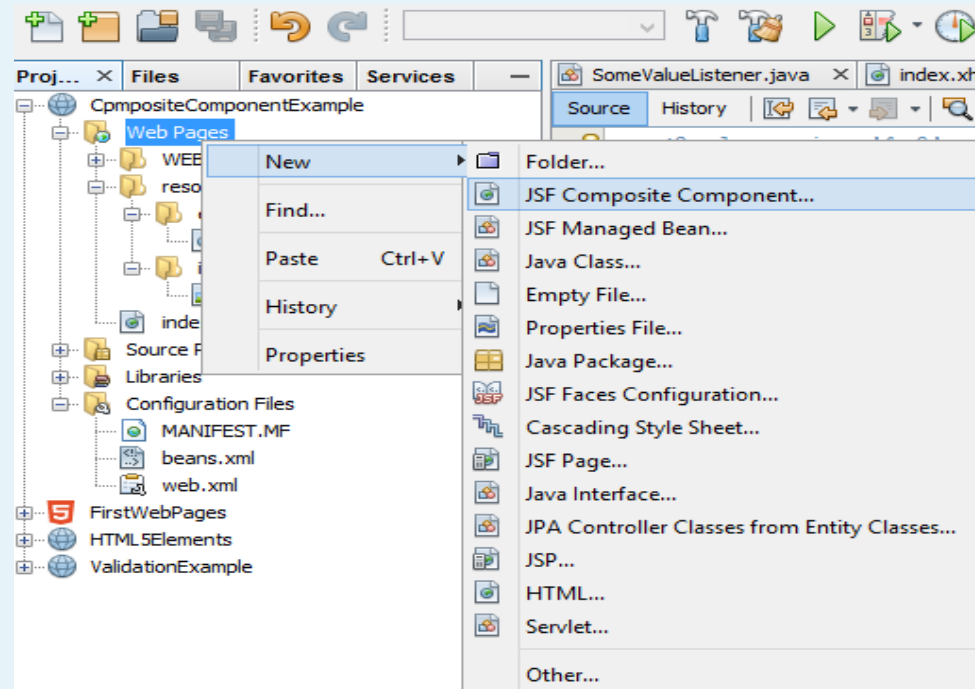
Composite Components

- With Facelets, any XHTML page that contains markup tags and other components can be converted into a composite component.

- Using the resources facility, the composite component can be stored in a library that is available to the application from the defined resources location.

Composite Component Tags

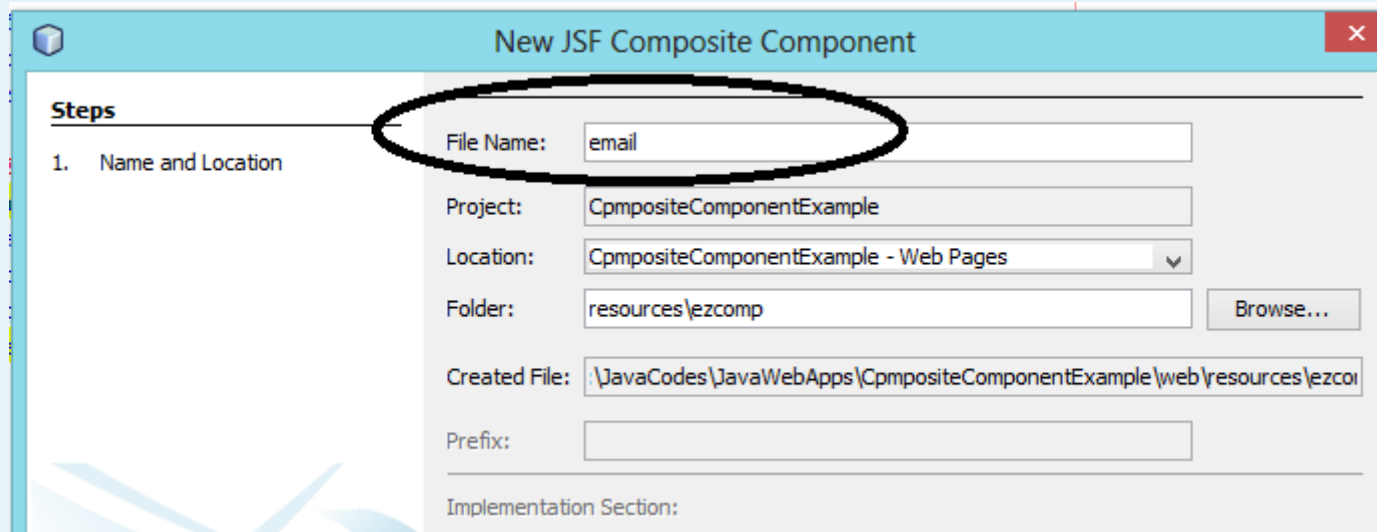| Tag	Description
`composite:interface`	Declares the usage contract for a composite component. The composite component can be used as a single component whose feature set is the union of the features declared in the usage contract.
`composite:implementation`	Defines the implementation of the composite component. If a composite:interface element appears, there must be a corresponding composite:implementation.
`composite:attribute`	Declares an attribute that may be given to an instance of the composite component in which this tag is declared.
`composite:insertChildren`	Any child components or template text within the composite component tag in the using page will be reparented into the composite component at the point indicated by this tag's placement within the `composite:implementation` section.
`composite:valueHolder`	Declares that the composite component whose contract is declared by the `composite:interface` in which this element is nested exposes an implementation of `ValueHolder` suitable for use as the target of attached objects in the using page.
`composite:editableValueHolder`	Declares that the composite component whose contract is declared by the `composite:interface` in which this element is nested exposes an implementation of `EditableValueHolder` suitable for use as the target of attached objects in the using page.
`composite:actionSource`	Declares that the composite component whose contract is declared by the composite:interface in which this element is nested exposes an implementation of ActionSource2 suitable for use as the target of attached objects in the using page.

Creating an Composite Component

- In project tree right click over Web Pages.

- Select New->JSF Composite Component...

Creating an Composite Component

- Give name for your component and press Finsih
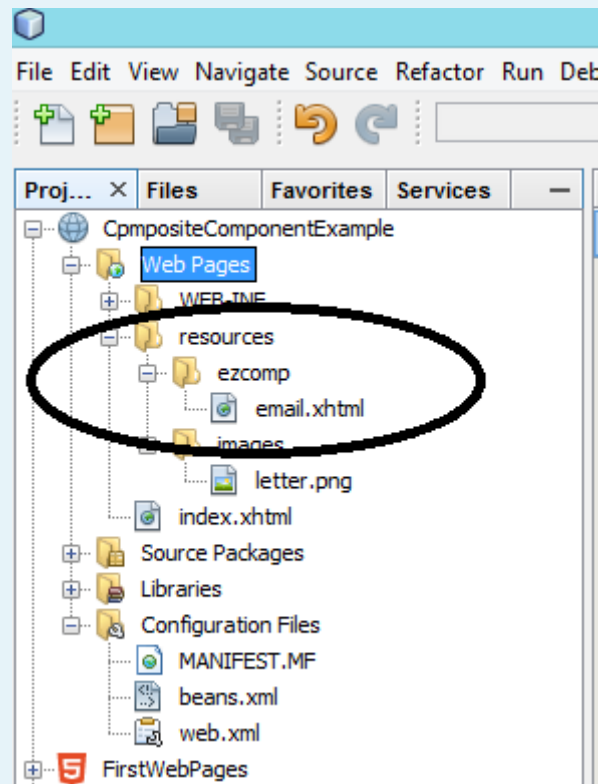
Creating an Composite Component

- Your project tree should be now as follow

Example Implementation

```
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:cc="http://xmlns.jcp.org/jsf/composite"
    xmlns:h="http://xmlns.jcp.org/jsf/html">

    <!-- INTERFACE -->
    <cc:interface>
        <cc:attribute name="email_addr" required="false"></cc:attribute>
    </cc:interface>

    <!-- IMPLEMENTATION -->
    <cc:implementation>
        <h:graphicImage alt="wave" library="images" name="letter.png"></h:graphicImage><br/>
        <h:outputLabel value="Enter your email:"></h:outputLabel>
        <h:inputText value="#{cc.attrs.email_addr}"></h:inputText>
    </cc:implementation>
</html>
```

Using Component

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "htt
<html xmlns="http://www.w3.org/1999/xhtml"
        xmlns:h="http://xmlns.jcp.org/jsf/html"
        xmlns:em="http://xmlns.jcp.org/jsf/composite/ezcomp">
    <h:head>
        <title>Using Composite Components</title>
    </h:head>
    <h:body>
        <h:form>
            <em:email email_addr="Enter your email" />
        </h:form>
    </h:body>
</html>
```

Enter your email: Enter your email