



Exadel Studio Pro

Getting Started Guide for Creating a JSF Application

We are going to show you how to create a simple JSF application using the Exadel Studio Pro plug-in for Eclipse. The completed application will ask a user to enter a name and click a button. The resulting new page will display the familiar message, “Hello <name>!”

This document will show you how to create such an application from the beginning, along the way demonstrating some of the powerful features of Exadel Studio. You will design the JSF application and then run the application from inside Exadel Studio.

We’ll assume that you have already launched Eclipse with Exadel Studio Pro installed and also that the Exadel Studio perspective is the current perspective. (If not, make it active by selecting **Window/Open Perspective/Exadel Studio** from the menu bar or by selecting **Window/Open Perspective/Other...** from the menu bar and then selecting **Exadel Studio** from the **Select Perspective** dialog box.)

Setting Up the Project

We are first going to create a new project for the application.

1. Go to the menu bar and select **File/New/Project...**
2. Select **Exadel Studio/JSF Project** in the **New Project** dialog box.
3. Click **Next >**.
4. Enter `jsfHello` as the project name.
5. Leave everything else as is, and click **Finish**.

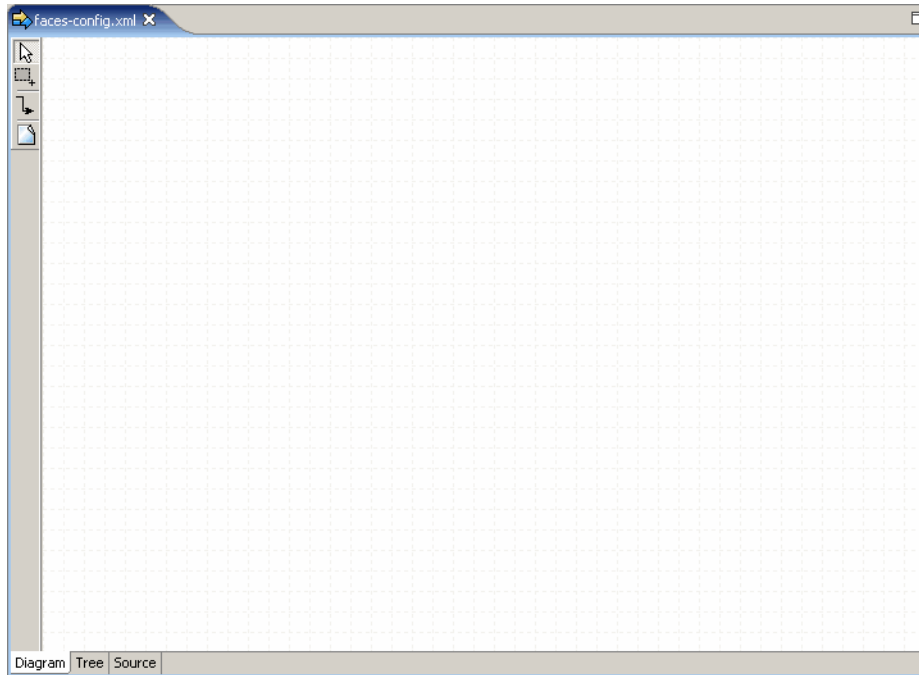
The JSF Application Configuration File

A **jsfHello** node should appear in the upper-left Package Explorer view.

6. Click the plus sign next to **jsfHello** to reveal the child nodes.
7. Click the plus sign next to **WebContent** under **jsfHello**.
8. Click the plus sign next to **WEB-INF** under **WebContent**.

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

9. Then double-click on the **faces-config.xml** node to display the JSF application configuration file editor.



Adding Navigation to the Application

In our simple application, the flow is defined as a single navigation rule connecting two views (presentation files). At this point, we will create the placeholders for the two JSP presentation files and then the navigation rule to connect them as views. Later, we will complete the coding for the JSP presentation files. With Exadel Studio Pro, we can do all of this in the Diagram mode of the configuration file editor.

Adding Two Views (JSP Pages)

10. Right-click anywhere on the diagram and select **New View...** from the pop-up menu.
11. In the dialog box, type `pages/inputname` as the value for **From-view-id**
12. Leave everything else as is.
13. Click **Finish**.

If you look in the Package Explorer view you should see a **pages** folder under **WebContent**. Opening it will reveal the JSP file you just created.

14. Back on the diagram, right-click anywhere and select **New View...** from the pop-up menu.
15. In the dialog box, type `pages/greeting` as the value for **From-view-id**

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

16. Leave everything else as is.
17. Click **Finish**.

Creating the Transition (Navigation Rule)

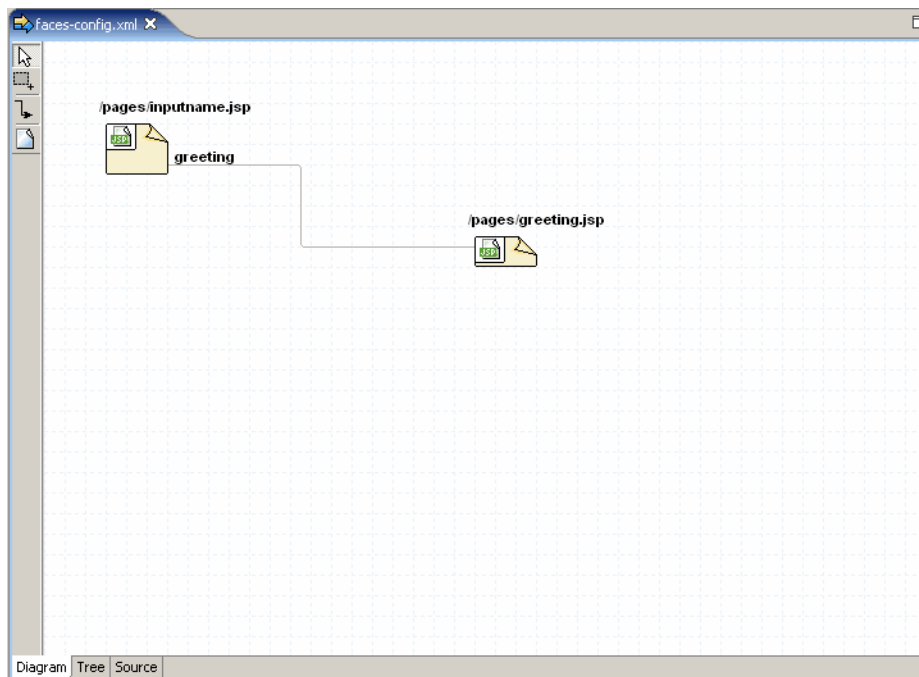
18. In the diagram, select the connection icon third from the top along the upper left side of the diagram



to get an arrow cursor with a two-pronged plug at the arrow's bottom.

19. Click on the **pages/inputname** page icon and then click on the **pages/greeting** page icon.

A transition should appear between the two icons.



20. Select **File/Save** from the menu bar.

Adding a Managed Bean to the Application

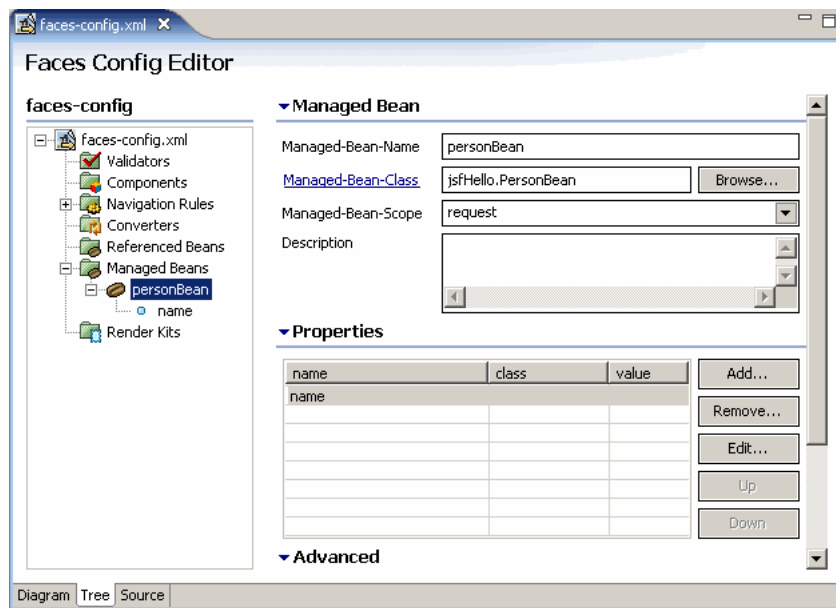
To store data in the application, we will use a managed bean.

21. Click on the **Tree** tab at the bottom of the editing window.
22. Select the **Managed Beans** node and then click the **Add...** button displayed along the right side of the editor window.
23. Type in `jsfHello.PersonBean` for **Class** and `personBean` for **Name**. Leave **Scope** as is and **Generate Source Code** as is (checked).

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

24. Click **Finish**.
25. **personBean** will now be selected and three sections of information, **Managed Bean**, **Properties**, and **Advanced**, will be displayed about it. Under the **Properties** section, click the **Add...** button.
26. For **Property-Name** type in name. Leave everything else as is. (When **Property-Class** is not filled in, `String` is the assumed type.)
27. Click **Finish**.
28. Select the **personBean** node in the tree.

You should see this now:



29. Select **File/Save** from the menu bar.

You have now registered the managed bean and created a stub-coded class file for it.

Editing the JSP View Files

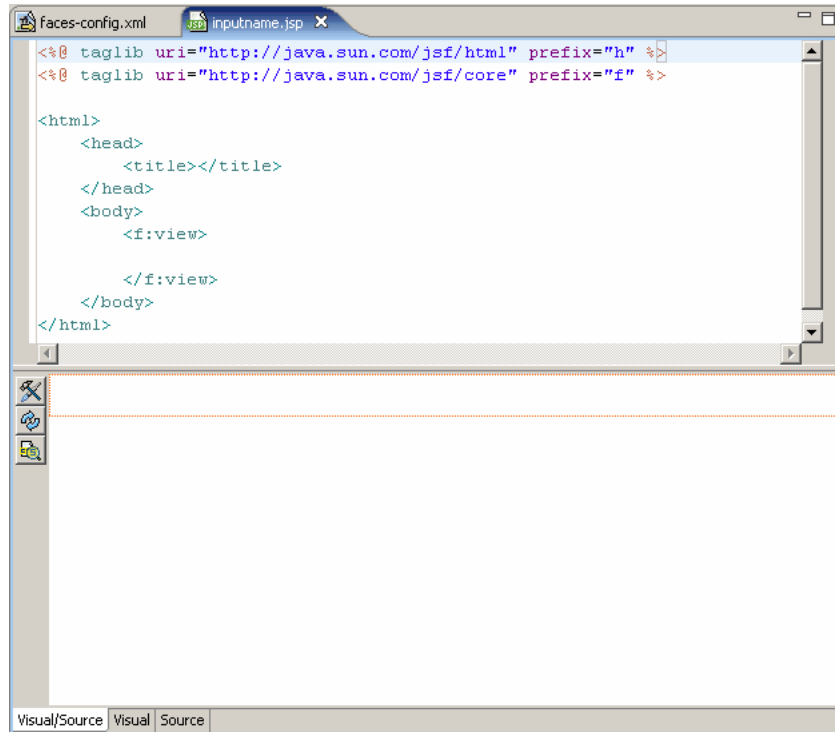
Now we will finish editing the JSP files for our two “views” using Exadel’s JSP Visual Page Editor.

inputname.jsp

30. Click on the **Diagram** tab for the configuration file editor.
31. Open the editor for this first JSP file by double-clicking on the **/pages/input-name.jsp** icon.

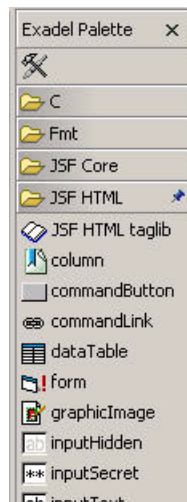
Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

The Visual Page Editor will open in a screen split between source code along the top and a WYSIWIG view along the bottom:



Some JSF code will already be in the file because we selected a template when creating the page.

32. Select the **Visual** tab, so we can work with the editor completely in its WYSIWYG mode.
33. To the right of the editor, in the Exadel Palette, expand the **JSF HTML** palette folder by selecting it.



34. Click on **form** within this folder, drag the cursor over to the editor, and drop it inside the red box in the editor.

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

Another red box will appear inside the first red box.

35. Right-click on the innermost box and select **<h:form> Attributes** from the menu.
36. In the value field next to **id**, type `greeting` and click on the **Close** button.
37. Type `Please enter name:` inside the boxes.
38. Select **inputText** within the **JSF HTML** palette folder and drag it into the innermost box in the editor after “Please enter name:”.
39. In the attributes dialog, click in the value field next to the **value** attribute and click on the ... button.
40. Then, select the **Managed Beans/personBean/name** node and click on the **Ok** button
41. Back in the attributes dialog, select the **Advanced** tab, type in `name` as the value for the **id** attribute, and then click on the **Finish** button.
42. Select **commandButton** within the **JSF HTML** palette folder and drag it into the innermost box in the editor after the input box.
43. In the attributes dialog, click in the value field next to the **action** attribute and click on the ... button.
44. Then, select the **View Actions/greeting** node and click on the **Ok** button.
45. Back in the attributes dialog box, type in `Say Hello` as the value for the **value** attribute (“Say Hello”) and then click on the **Finish** button.

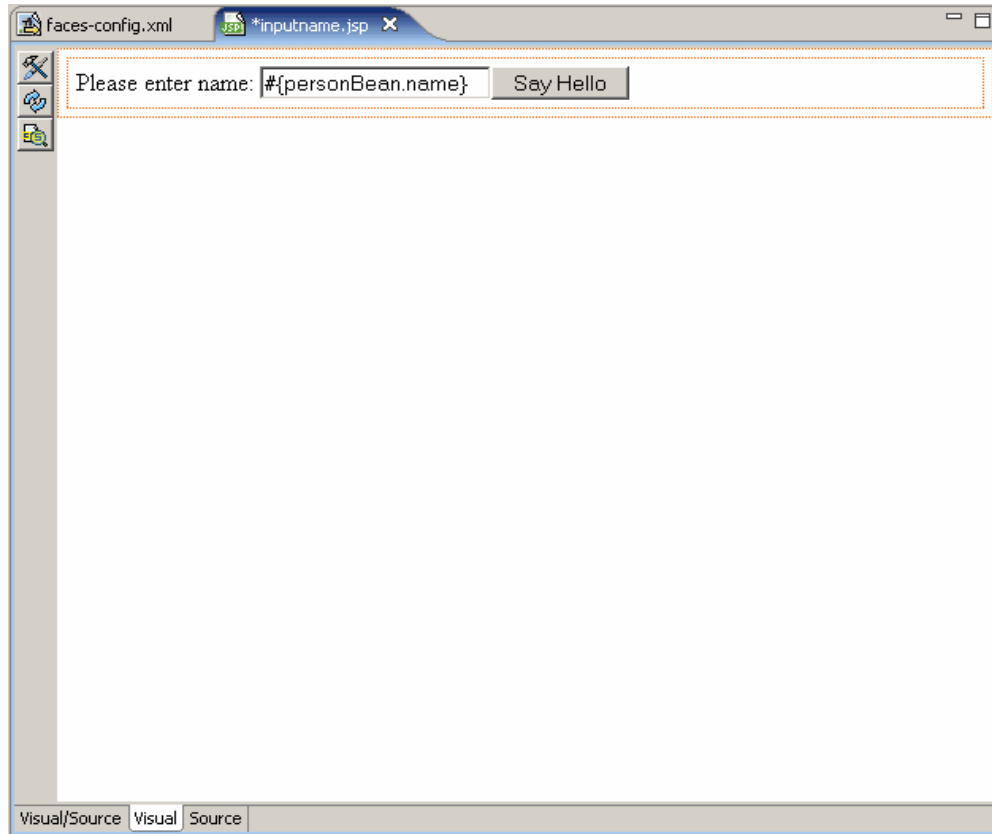
The source coding should be something like this now:

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>

<html>
  <head>
    <title></title>
  </head>
  <body>
    <f:view>
      <h:form id="greeting">
        Please enter a name:
        <h:inputText id="name" value="#{personBean.name}"/>
        <h:commandButton value=" Say Hello " action="greeting"/>
      </h:form>
    </f:view>
  </body>
</html>
```

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

The editor should look like this:



46. Save the file by selecting **File/Save** from the menu bar.

greeting.jsp

47. Click on the **faces-config.xml** tab to bring the diagram back.
48. Open the editor for the second file by double-clicking on the **/pages/greeting.jsp** icon.
49. Select the **Visual** tab, so we can work with the editor completely in its WYSIWYG mode.
50. Type `He11o` (note space after hello) into the box.
51. Select **outputText** within the **JSF HTML** palette folder and drag it into the innermost box in the editor after "Hello".
52. In the attributes dialog, click in value field next to the `value` attribute and click on the **...** button.
53. Then, select the **Managed Beans/personBean/name** node, click on the **Ok** button, and then click on the **Finish** button.
54. Right after the output field, type an exclamation point (!).

Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

The source coding should be something like this now:

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>

<html>
  <head>
    <title></title>
  </head>
  <body>
    <f:view>
      Hello <h:outputText value="#{personBean.name}"/>!
    </f:view>
  </body>
</html>
```

55. Save the file.

Creating the Start Page

You also need to create a start page as an entry point into the application.

56. In the Package Explorer view to the left, right-click **jsfHello/WebContent** and select **New/JSP File**.
57. For **Name** type in `index`, for **Template** select **JSPRedirect**, and click **Finish**.

A JSP editor will open up on the newly created file.

58. In the Source part of the split screen, type `/pages/inputname.jsf` in between the quotes for the `page` attribute.

The source coding should look like this now:

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head></head>
<body>
  <jsp:forward page="/pages/inputname.jsf" />
</body>
</html>
```

Note the `.jsf` extension for the file name. This is a mapping defined in the `web.xml` file for the project for invoking JavaServer Faces when you run the application.

59. Select **File/Save** from the menu bar.

Running the Application

Everything is now ready for running our application—without having to leave Exadel Studio—by using the Tomcat engine that comes with the Exadel Studio plug-in. For controlling Tomcat within JSF Studio, the toolbar contains a special panel.



Exadel Studio Pro: Getting Started Guide for Creating a JSF Application

60. Start up Tomcat by clicking on the first icon from left. (If Tomcat is already running, stop it by clicking on the third icon from the left and then start it again. Remember, the JSF run-time requires restarting the servlet engine when any changes have been made.)

After the messages in the **Console** tabbed view stop scrolling, Tomcat is available.

61. Click on the Exadel run icon in the toolbar.



This is the equivalent of launching the browser and typing `http://localhost:8080/jsfHello` into your browser. Our JSF application should now appear.