

# Explicit navigation

## faces-con fig.xml

- Location
  - WEB-INF/faces-con fig.xml
- Purposes
  - Declare navigation rules
    - Map return conditions to results pages
  - Declare beans
    - Map bean names to bean classes
  - Inject bean properties
  - Define properties files
  - Declare Locales
  - Register validators and renderers
  - Register custom components

## Navigation rules

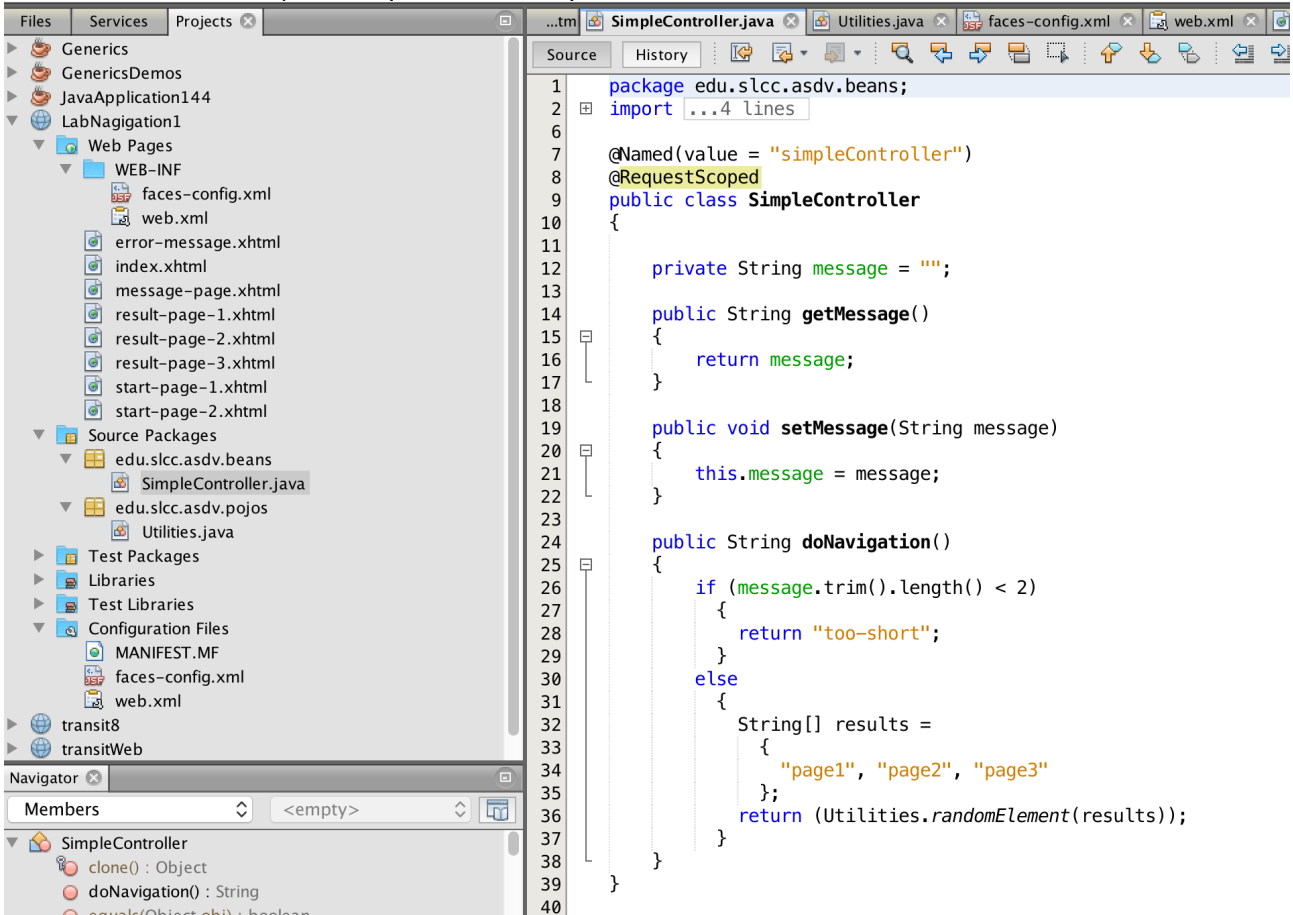
```
<navigation-rule>
  <from-view-id>/some-start-page.xhtml</from-view-id>

  <navigation-case>
    <from-outcome>return-condition-1from controller</from-outcome>
    <to-view-id>/result-page-1.xhtml</to-view-id>
  </navigation-case>
  .....
  More navigation-case entries for other conditions
</navigation-rule>
```

## Practice Example

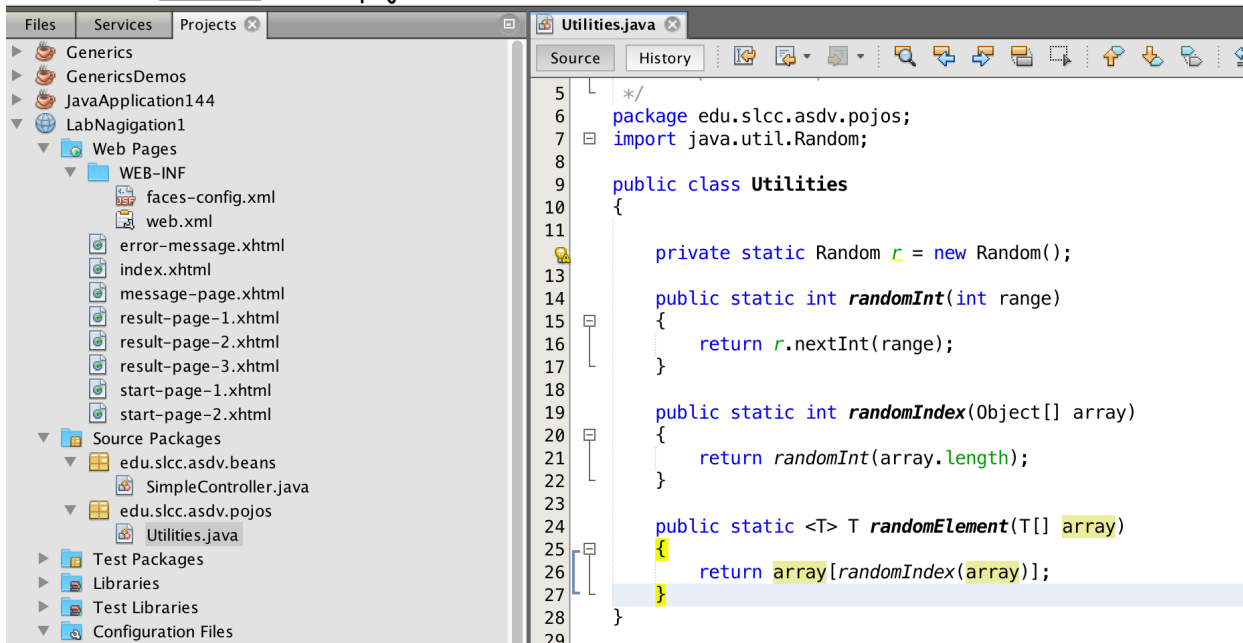
- What we will do in this practice
  - Collect a message from user, have the controller (in bean) generate a random page as a result . Then use the random page to navigate to one of three possible navigation- pages which will display the message.
  - If the message is missing or only one character long, show an error page
- Implementation
  - Action controller returns four possible strings: page1, page2, page3, and too-short
  - Navigation rules in faces-con fig.xml **map** each of those returns of controller to exact results page

1. Create a new Web App, LabNavigation1
2. Create the RequestScoped bean that produces some result



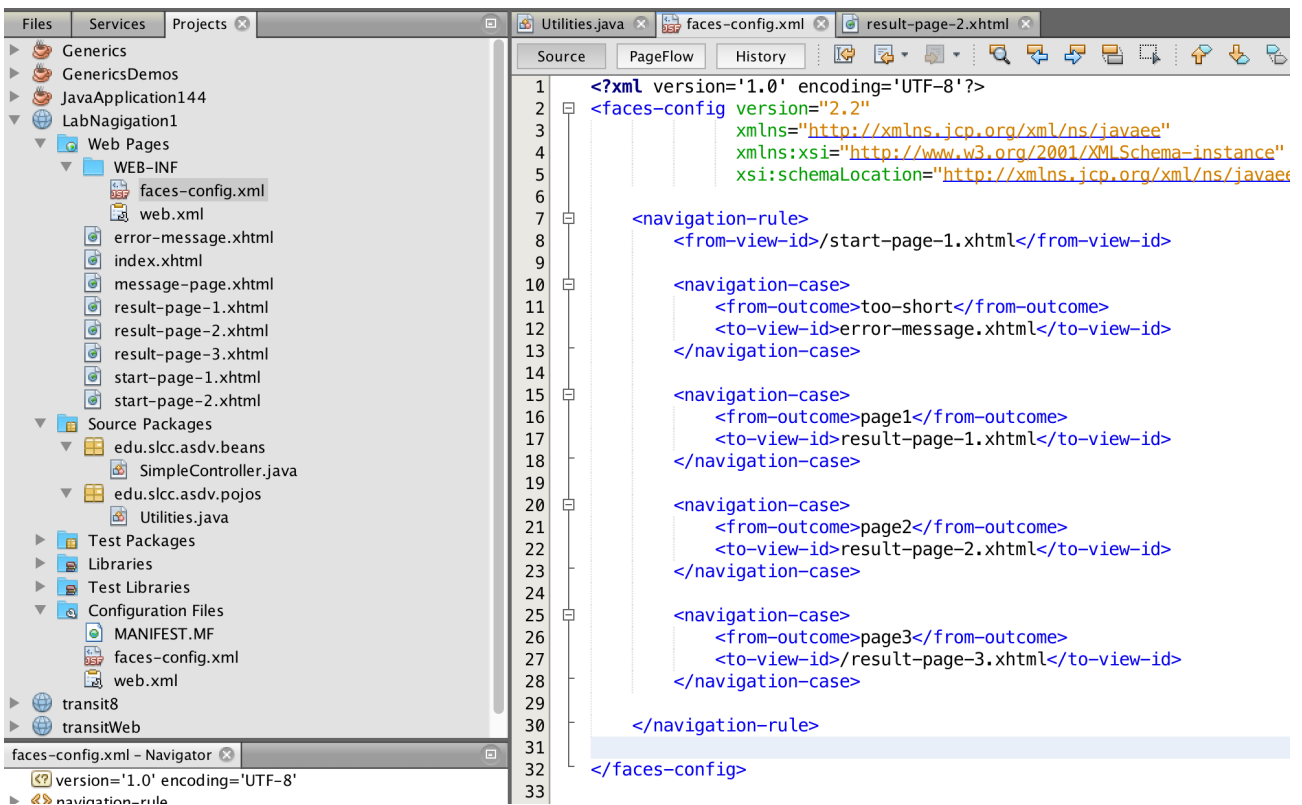
```
1 package edu.slcc.asdv.beans;
2 import ...4 lines
6
7 @Named(value = "simpleController")
8 @RequestScoped
9 public class SimpleController
10 {
11
12     private String message = "";
13
14     public String getMessage()
15     {
16         return message;
17     }
18
19     public void setMessage(String message)
20     {
21         this.message = message;
22     }
23
24     public String doNavigation()
25     {
26         if (message.trim().length() < 2)
27         {
28             return "too-short";
29         }
30         else
31         {
32             String[] results =
33             {
34                 "page1", "page2", "page3"
35             };
36             return (Utilities.randomElement(results));
37         }
38     }
39 }
40
```

3. Add a Utilities class a pojo.

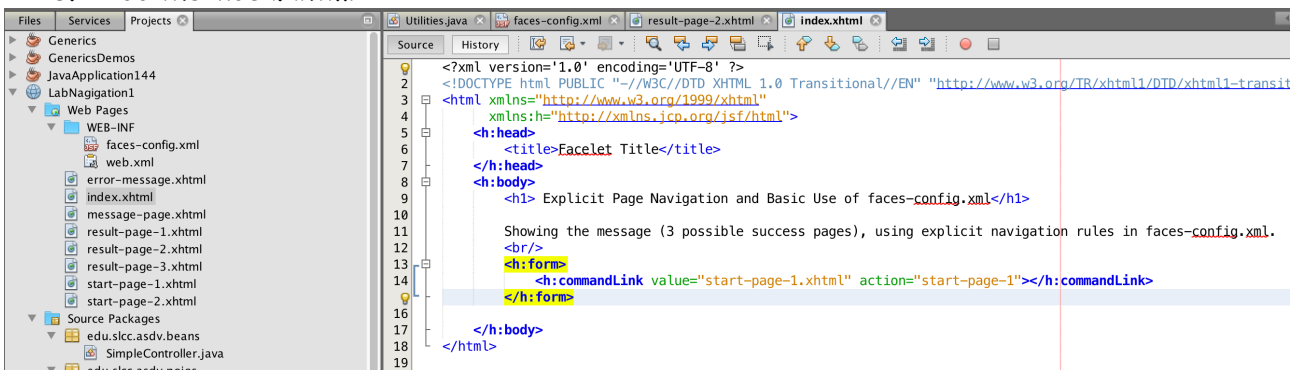


```
5 /*
6 package edu.slcc.asdv.pojos;
7 import java.util.Random;
8
9 public class Utilities
10 {
11
12     private static Random r = new Random();
13
14     public static int randomInt(int range)
15     {
16         return r.nextInt(range);
17     }
18
19     public static int randomIndex(Object[] array)
20     {
21         return randomInt(array.length);
22     }
23
24     public static <T> T randomElement(T[] array)
25     {
26         return array[randomIndex(array)];
27     }
28 }
29
```

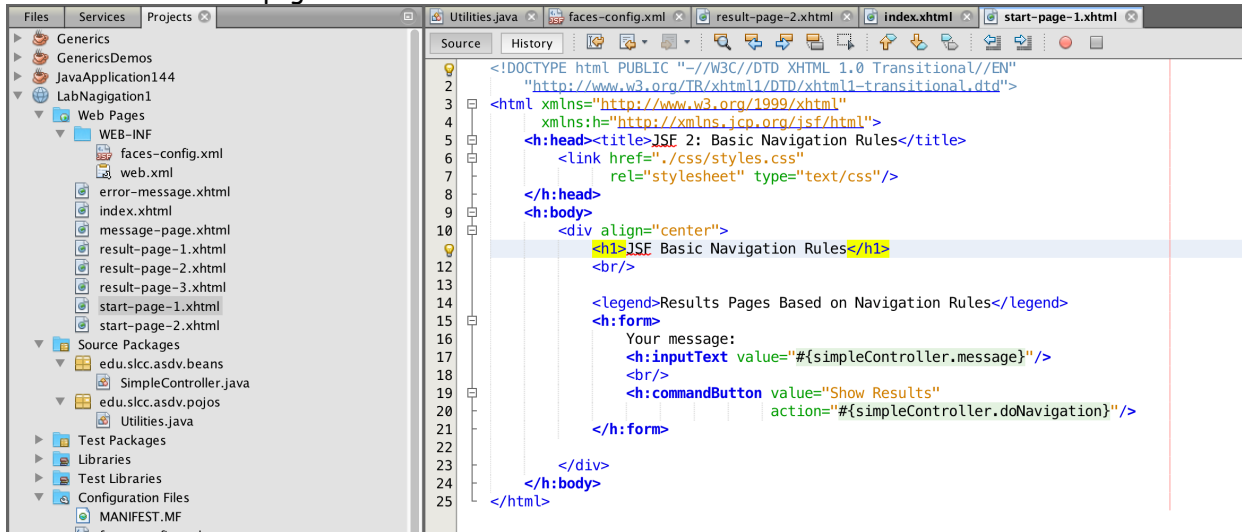
4. Create the `faces-config.xml` file and add the navigation shown.  
Click: ( New/Other/JavaServer Faces/JSF Pages Configuration)



5. Add the `index.xhtml`

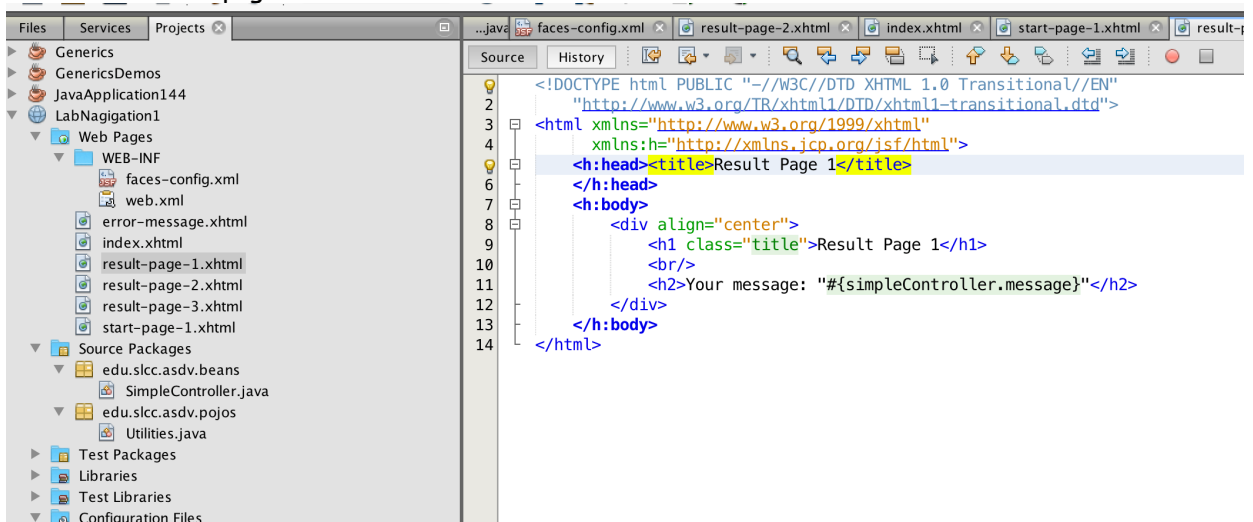


## 6. Add the start-page-1.xhtml



```
<!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:h="http://xmlns.jcp.org/jsf/html">
<h:head><title>JSF 2: Basic Navigation Rules</title>
<link href="/css/styles.css"
rel="stylesheet" type="text/css"/>
</h:head>
<h:body>
<div align="center">
<h1>JSF Basic Navigation Rules</h1>
<br/>
<legend>Results Pages Based on Navigation Rules</legend>
<h:form>
Your message:
<h:inputText value="#{simpleController.message}"/>
<br/>
<h:commandButton value="Show Results"
action="#{simpleController.doNavigation}"/>
</h:form>
</div>
</h:body>
</html>
```

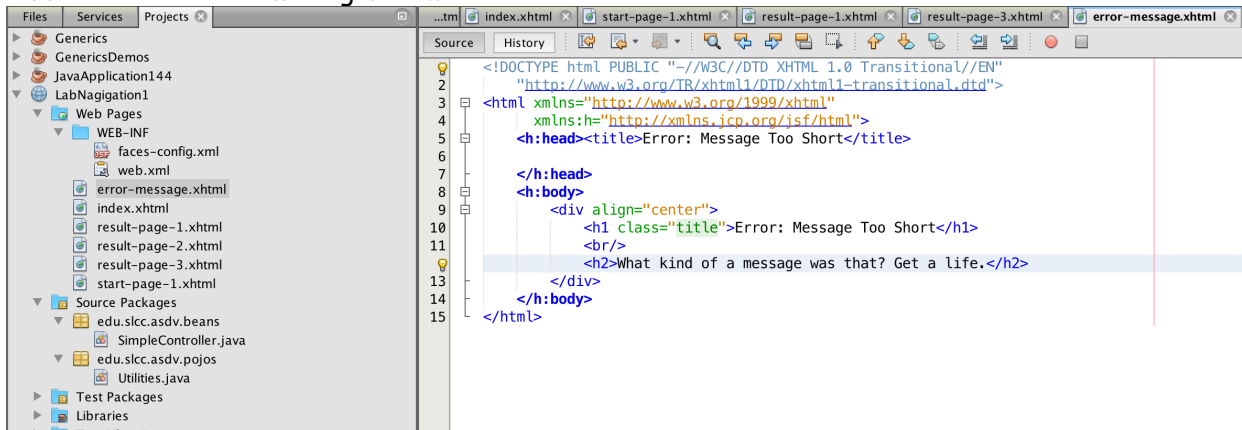
## 7. Add result-page-1.xhtml



```
<!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:h="http://xmlns.jcp.org/jsf/html">
<h:head><title>Result Page 1</title>
</h:head>
<h:body>
<div align="center">
<h1 class="title">Result Page 1</h1>
<br/>
<h2>Your message: "#{simpleController.message}"</h2>
</div>
</h:body>
</html>
```

## 8. Similarly, add result-page-2.xhtml and result-page-3.xhtml

## 9. Add the error-message.xhtml



```
<!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:h="http://xmlns.jcp.org/jsf/html">
<h:head><title>Error: Message Too Short</title>
</h:head>
<h:body>
<div align="center">
<h1 class="title">Error: Message Too Short</h1>
<br/>
<h2>What kind of a message was that? Get a life.</h2>
</div>
</h:body>
</html>
```

## 10. Clean build and run.