



Java™ Education & Technology Services

Java Server Faces (JSF)



Table of Contents

- **Chapter 1:** JSF Introduction
- **Chapter 2:** Understanding Managed Beans
- **Chapter 3:** Page Navigation
- **Chapter 4:** Standard JSF Tags
- **Chapter 5:** Facelets
- **Chapter 6:** Data Tables
- **Chapter 7:** Conversion and Validation
- **Chapter 8:** AJAX & JSF 2.0



Chapter 3

Page Navigation



Chapter 3 Outline

- ☐ What is Navigation?
- ☐ Types of Navigation
- ☐ Mapping Outcomes to View IDs
- ☐ Wildcards
- ☐ from-action
- ☐ Conditional Navigation Cases
- ☐ Dynamic Target View IDs
- ☐ Redirection

What is Navigation?

- **Navigation:** is a set of rules for choosing the next page to be displayed after a **button** or **hyperlink** is clicked.
- The selection of the next page is determined by:
 - The page that is currently displayed.
 - The **action method** invoked by the **action** property of the component that generated the event.
 - The outcomes of decisions in the business logic.
- The navigation handler is responsible for selecting the next JSF page.



Types of Navigation

- **Types of Navigation:**
 - Static Navigation
 - Dynamic Navigation

Types of Navigation (Cont'd)

- Static Navigation

In /index.xhtml

```
<h:commandButton label="Login" action="welcome"/>
```

View ID -- JSF page ← mapped outcome



- The outcome is transformed into a view ID through:
 - If it doesn't have a file extension, then append the extension of the **current view**.
 - If it doesn't start with a /, then goes as the path of the **current view**.

Outcome: **welcome** → View ID: **/welcome.xhtml**

Types of Navigation (Cont'd)

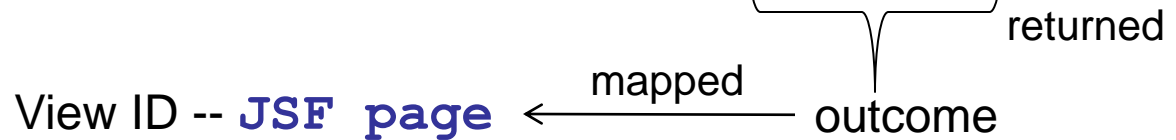
- Dynamic Navigation

In /index.xhtml

```
<h:commandButton label="Login" action="#{loginBean.verify}"/>
```

View ID -- JSF page ← mapped — outcome

returned



- Method in loginBean:

```
public String verify() { // It can have any return type
    if (...)
        return "success";
    else
        return "failure";
}
```




Types of Navigation (Cont'd)

- **Dynamic Navigation (Cont'd)**

- **Steps of Dynamic Navigation:**

1. The specified bean is retrieved.
2. The referenced method is called.
3. The outcome string is transformed into a view ID.
4. The navigation handler looks up the page corresponding to the view ID.



Mapping Outcomes to View IDs

- The *method* that computes the *outcome* should not have to know the exact names of the web pages.
- JSF2.0 provides a mechanism for mapping *logical outcomes* to actual *web pages*.
- Add *navigation-rule* entries into *faces-config.xml*.



Mapping Outcomes to View IDs (Cont'd)

<navigation-rule>

<from-view-id> /index.xhtml </from-view-id>

<navigation-case>

<from-outcome> success </from-outcome>

<to-view-id> /welcome.xhtml </to-view-id>

</navigation-case>

</navigation-rule>

Mapping Outcomes to View IDs (Cont'd)

<navigation-rule>

// no from-view-id : This rule applies to all pages

<navigation-case>

<from-outcome> **success** </from-outcome>

<to-view-id> **/welcome.xhtml** </to-view-id>

</navigation-case>

</navigation-rule>

Mapping Outcomes to View IDs (Cont'd)

<navigation-rule>

<from-view-id> /index.xhtml </from-view-id>

<navigation-case>

<from-outcome> success </from-outcome>

<to-view-id> /welcome.xhtml </to-view-id>

</navigation-case>

<navigation-case>

<from-outcome> failure </from-outcome>

<to-view-id> /newuser.xhtml </to-view-id>

</navigation-case>

</navigation-rule>

<navigation-rule>

<from-view-id> /secure/* </from-view-id>

<navigation-case>

.....

</navigation-case>

</navigation-rule>

<navigation-rule>

<from-view-id> /* </from-view-id>

.....

</navigation-rule>

<navigation-rule>

<from-view-id> * </from-view-id>

.....

</navigation-rule>

<navigation-rule>

<from-view-id> /login.xhtml</from-view-id>

<navigation-case>

<from-action>#{quizBean.answerAction}</from-action>

<from-outcome>again</from-outcome>

<to-view-id>/again.xhtml</to-view-id>

</navigation-case>

<navigation-case>

<from-action>#{quizBean.startAction}</from-action>

<from-outcome>again</from-outcome>

<to-view-id>/Home.xhtml</to-view-id>

</navigation-case>

</navigation-rule>

`<navigation-case>`

`<from-outcome>previous</from-outcome>`

`<if> #{quizBean.currentQuestion != 0} </if>`

`<to-view-id> /main.xhtml </to-view-id>`

`</navigation-case>`



Dynamic Target View IDs

<navigation-case>

.....

<to-view-id> #{quizBean.nextViewID} </to-view-id>

</navigation-case>

- JSF can send an HTTP redirect to the client.
- The redirect response tells the client which URL to use for the next page.
- The client then makes a GET request to that URL.
- Redirecting is slow because another round trip to the browser is involved.

```
<h:commandButton label="Login" action="welcome?faces-redirect=true"/>
```

Or

```
<navigation-case>
  <from-outcome>success</from-outcome>
  <to-view-id>/welcome.xhtml</to-view-id>
  <redirect/>
</navigation-case>
```

- **Redirection and the Flash.**
- **RESTful Navigation and Bookmarkable URLs**