JSF ПРАВИЛА ЗА НАВИГАЦИЯ

Съдържание

Част 1:

- JSF въведение
- **JSF правила за навигация**
- JSF управляеми компоненти
- JSF компоненти на потребителски интерфейс

Концепция за изглед

Изгледа може да бъде:

- *слой от MVC архитектурата*;

от друга страна изглед се нарича и:

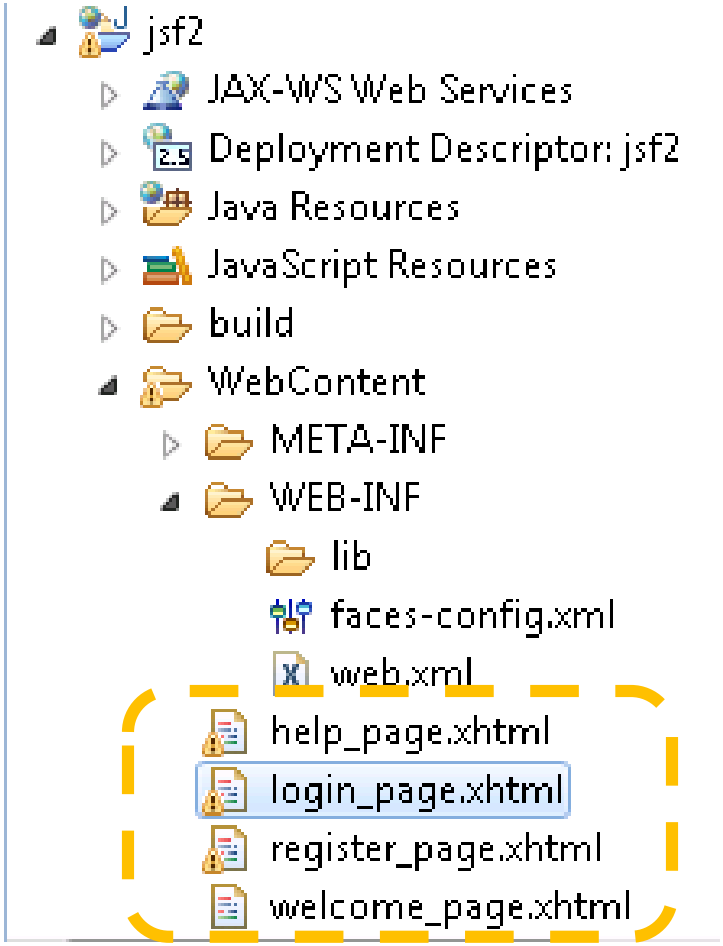
- *отделна страница, даваща възможност на потребителя да взаимодейства с приложението*;

- В изглед / страница могат да се разполагат набор от компоненти.

Графични компоненти – основни характеристики

- За основните тагове от HTML са създадени **графични компоненти**, които им съответстват. Например: компонент за вход от html форма **(<input type=...>)**
- Предоставена е възможност за **създаване на допълнителни тагове от потребителя** (например, за взимане на дата)
- Предоставенавена е възможност към графичните компоненти да се **свързват свойства на бизнес компоненти** (JavaBeans), валидатори, конвертори и други.
- Графичните компоненти **могат да съхраняват стойността си при последователни заявки** към съответният изглед / страница (stateful)

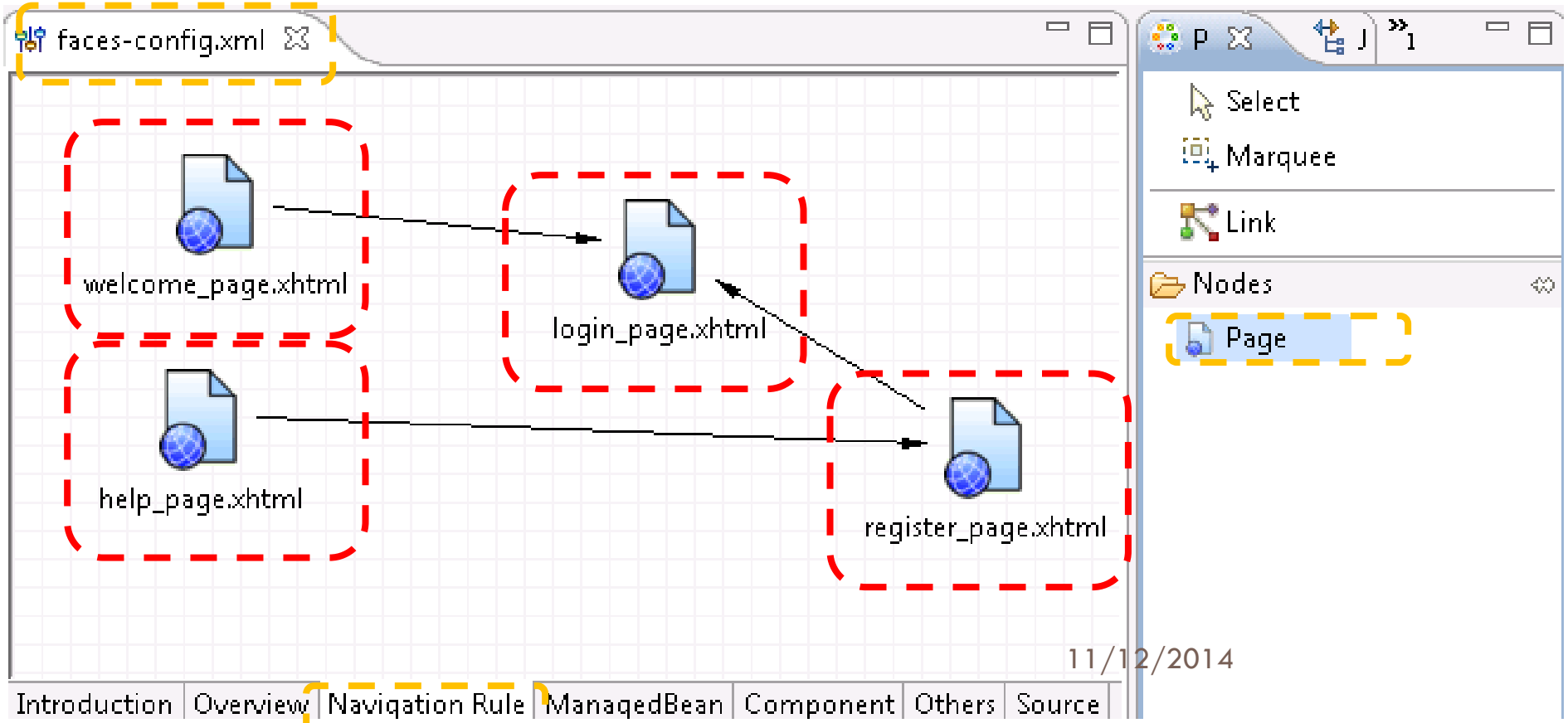
Създаване на изгледите



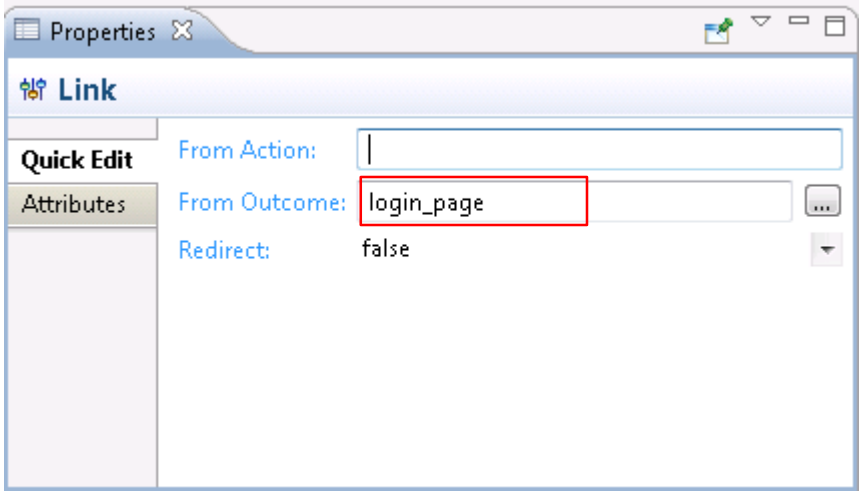
- welcome_page
- help_page
- register_page
- login_page

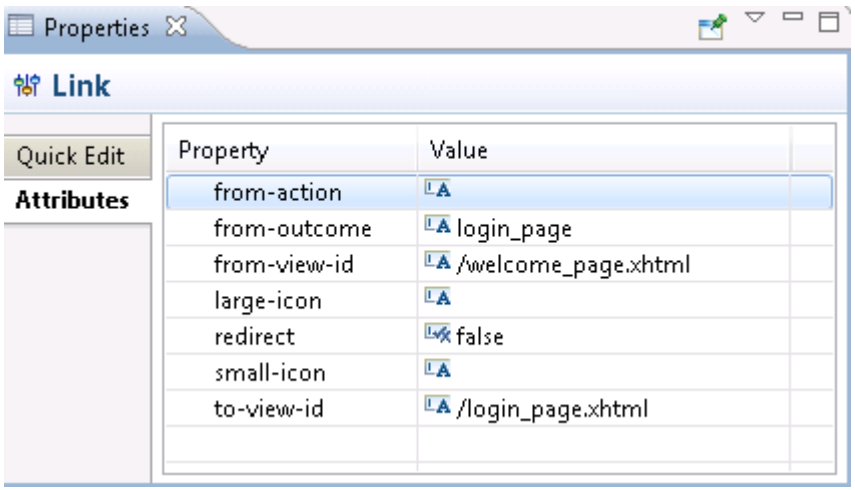
Визуално създаване на правила за навигация в Eclipse (3.7.2)

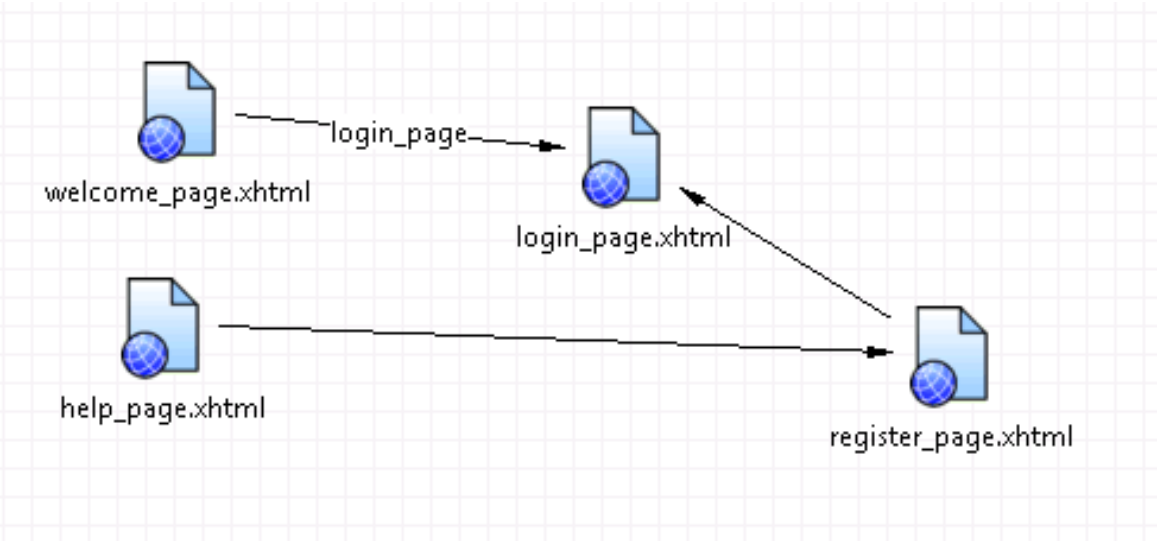
- Стъпка 2: връзки между страниците



Дефиниране на навигация по резултат







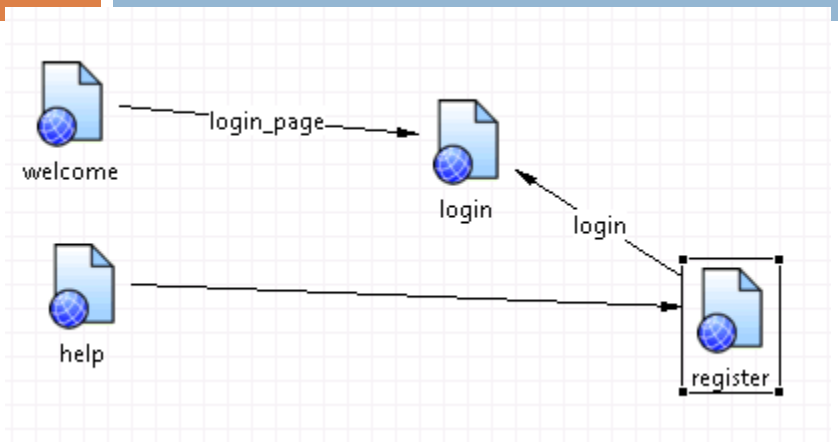
Изходен код

faces-config.xml

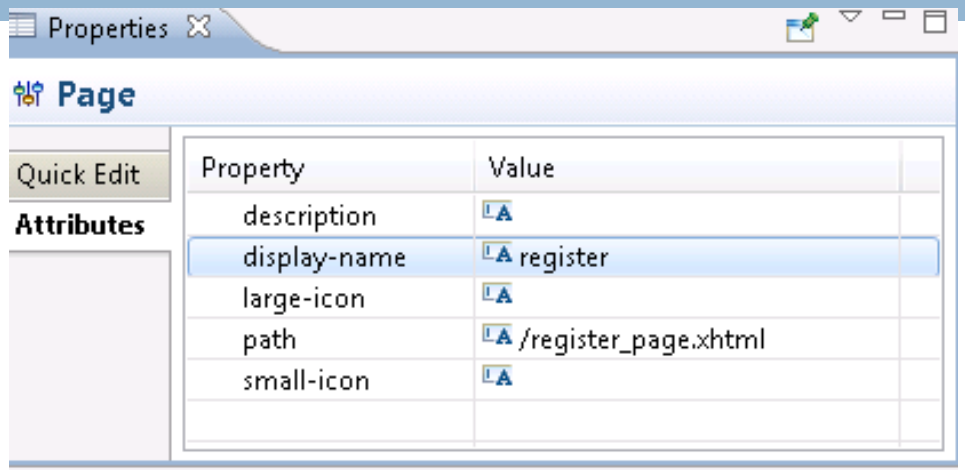
```
<?xml version="1.0" encoding="UTF-8"?>

<faces-config
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http:
    version="2.0">
    <navigation-rule>
        <display-name>welcome_page.xhtml</display-name>
        <from-view-id>/welcome_page.xhtml</from-view-id>
        <navigation-case>
            <from-outcome>login_page</from-outcome>
            <to-view-id>/login_page.xhtml</to-view-id>
        </navigation-case>
    </navigation-rule>
    <navigation-rule>
        <display-name>help_page.xhtml</display-name>
        <from-view-id>/help_page.xhtml</from-view-id>
        <navigation-case>
            <to-view-id>/register_page.xhtml</to-view-id>
        </navigation-case>
    </navigation-rule>
    <navigation-rule>
        <display-name>register_page.xhtml</display-name>
        <from-view-id>/register_page.xhtml</from-view-id>
        <navigation-case>
            <to-view-id>/login_page.xhtml</to-view-id>
        </navigation-case>
    </navigation-rule>

</faces-config>
```

Introduction | Overview | Navigation Rule | ManagedBean | Component | Others | Source

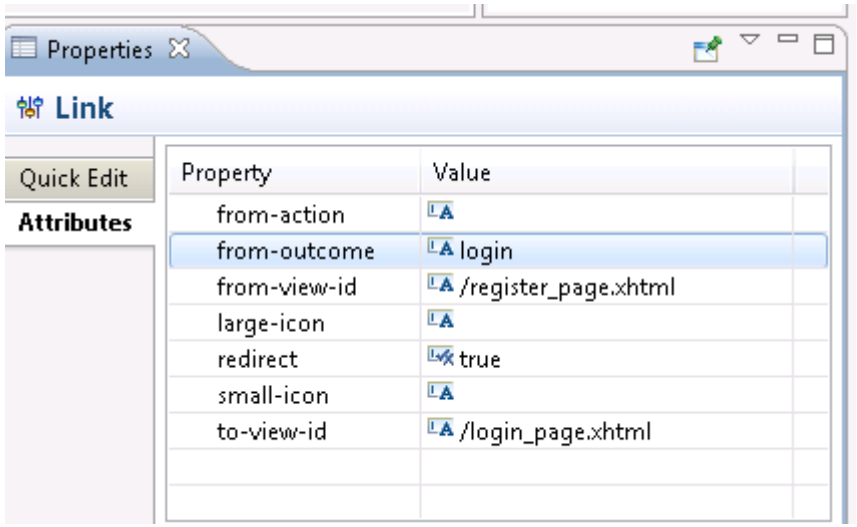
Задаване на пренасочване



1. Навигация в сайта



2. Свойства на изглед

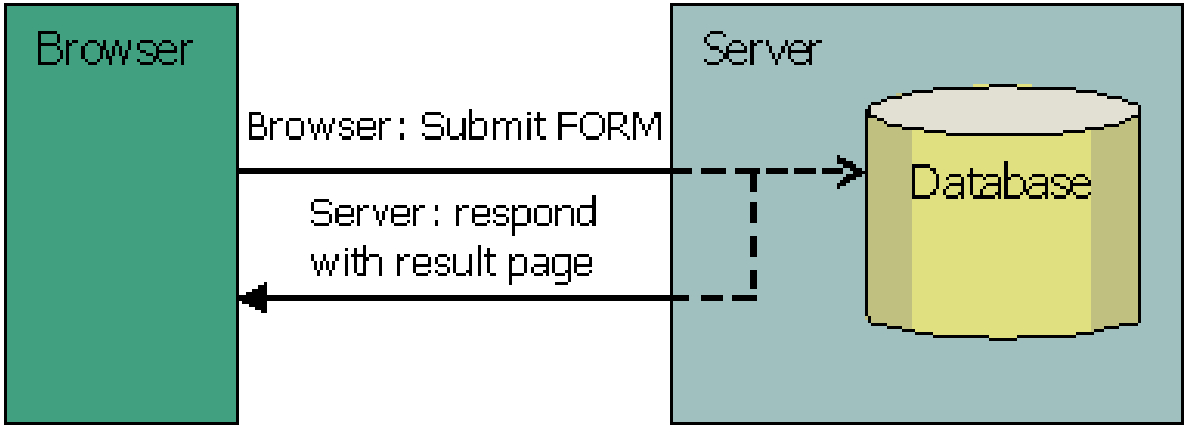


3. Свойства на навигационно правило 'login'

```
<navigation-rule>
    <display-name>register</display-name>
    <from-view-id>/register_page.xhtml</from-view-id>
    <navigation-case>
        <from-outcome>login</from-outcome>
        <to-view-id>/login_page.xhtml</to-view-id>
        <redirect />
    </navigation-case>
</navigation-rule>
```

4. Фрагмент от изходен код за навигационно правило с redirect

Шаблон PRG



- The answer to double submit problem is redirection. This is a known technique, but it has not become a standard for "after-POST" results yet. As far as I know it does not have a well-known name. I suggest calling it PRG pattern for POST-REDIRECT-GET.
- PRG pattern splits one request into two. Instead of returning a result page immediately in response to POST request, server responds with redirect to result page. Browser loads the result page as if it were an separate resource. After all, there are two different tasks to be done. First is to POST input data to the server. Second is to GET output to the client.

изт.: Michael Jouravlev, *Redirect After Post*, 2004 | TheServerSide

Проблеми: **Double Submit problem**

- Интерактивните програми предоставят две основни функции: прочитане на потребителският вход и визуализация на резултата;
- В уеб приложенията това става посредством двата основни HTTP метода съответно - POST и GET.
- Поради проблеми с браузърите правилното извикване на методите може да доведе до неконсистетно състояние на приложението;

source: http://www.theserverside.com/news/1365146/Redirect-After-Post

Пример за изглед

```
<h:form id="login">
   <h:panelGrid border="1" columns="2">
<h:outputLabel for="username" value="username" />
<h:inputText   id="username" value="#{loginReq.username}" />
<h:inputSecret id="password" value="#{loginReq.password}" />
   </h:panelGrid>
</h:form>
```

Навигация между страниците (1/2)

- Всяка **една препратка (hyperlink)** или **бутон** дефинира **резултатен низ (outcome)** или задава метод, който извършва дадено действие и връща този низ като резултат;

Навигация между страниците (2/2)

- Съществуват два вида правила за навигация:
  1. **явна навигация** – *където навигацията между изгледите е предварително дефинирана;*
  2. **неявна** – *чрез правила по подразбиране;*

Явни правила за навигация (1/4)

- Когато потребител натисне върху препратка или бутон се извиква съответният метод и резултатния низ от него се сравнява с предварително дефинирани правила от вида:
  - Резултатен низ (outcome) – нова страница
  - При което се преминава към съответната нова страница
- Ако резултатният низ е **null** се остава на текущата страница
- Правилата се дефинират във файла **faces-config.xml**

```
<h:commandButton action="#{loginRequest.login}"
                 value="Login" />
<h:commandLink action="register" value="here" />
```

Извиква се методът `login()` на `loginRequest`

Резултатният низ е константа (не се извиква метод)

Дефиниране на правила за навигация (2/4)

```
<navigation-rule>
   <from-view-id>/login.jsp</from-view-id>
   <navigation-case>
      <from-outcome>posts</from-outcome>
      <to-view-id>/posts.jsp</to-view-id>
   </navigation-case>
</navigation-rule>

<navigation-rule>
   <from-view-id>/login.jsp</from-view-id>
   <navigation-case>
      <from-outcome>register</from-outcome>
      <to-view-id>/register.jsp</to-view-id>
   </navigation-case>
</navigation-rule>
```

От коя страница

При какъв резулатен низ (**outcome**)

Следваща страница

ДЕФИНИРАНЕ НА ПРАВИЛА ЗА НАВИГАЦИЯ (3/4)

- Правилата могат да се комбинират:

```
<navigation-rule>
   <from-view-id>/login.jsp</from-view-id>
   <navigation-case>
      <from-outcome>posts</from-outcome>
      <to-view-id>/posts.jsp</to-view-id>
   </navigation-case>
   <navigation-case>
      <from-outcome>register</from-outcome>
      <to-view-id>/register.jsp</to-view-id>
   </navigation-case>
</navigation-rule>
```

От коя страница

При какъв резулатен низ (**outcome**)

Следваща страница

Второро правило от страница `login.jsp`

ДЕФИНИРАНЕ НА ПРАВИЛА ЗА НАВИГАЦИЯ (4/4)

•Общи правила – без значение на коя страница се намира в момента потребителят при получаване на даден резултатен низ (outcome) се преминава към определена страница:

```
<navigation-rule>
   <from-view-id>*</from-view-id>
   <navigation-case>
      <from-outcome>login</from-outcome>
      <to-view-id>/login.jsp</to-view-id>
   </navigation-case>
</navigation-rule>
```

От всяка страница

При какъв резулатен низ (outcome)

Следваща страница

Неявни правила за навигация

- В JSF 2+ е създадена алтернатива на явната навигация, чрез използването на следното правило:
  - Ако резултата от навигацията е даден идентификатор (например error) и нямаме явно дефинирано правило за такъв резултат, то този резултат автоматично се свързва със страница със същото име и разширение такова, каквото е във входната страница (в случая error.xhtml);
  - *outcome -> outcome.xhtml*

Пример за неявни правило (1/3)

- Създайте страница: **lecture_index.xhtml**
- <?xml version="1.0" encoding="UTF-8"?> <html xmlns="http://www.w3.org/1999/xhtml" lang="en" xmlns:h="http://java.sun.com/jsf/html">

<head> <title>J2EE Course – лекция 3</title> </head>

<body>

```
  <h:form id="course_form">
      <h:commandLink
                action="#{courseBean.nextLecture}"> See next lecture
       </h:commandLink>
  </h:form>
</html>
```

Направете метод, който ако съществува файл lecture04.xhtml да се връща **lecture04,** в противен случай да връща **lecture_not_found**

lecture04.xhtml (2/3)

```
<?xml version="1.0" encoding="UTF-8"?> <!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://java.sun.com/jsf/html" xmlns:f="http://java.sun.com/jsf/core">
    <head> <title>J2EE Course – лекция 3</title></head>
    <body>
        <h1> #{lectureBean.lectureTitle}</h1>
    </body>
</html>
```

Пример за неявни правило (3/3)

```
@ManagedBean
public class CourseBean {
    private String lectureTitle; //TODO1: write getLectureTitle()..
    private int currentLecture = 0;
    private boolean next = true; //if has next -> show link next
    public String nextLecture () {
       ++currentLecture;
       //TODO3: lectureTitle -> setTitle(…);
        //TODO4: load hasNext and hasPrev properties (according to file existance
in web-folder
       //TODO2: проверка дали файла същесвува
       return ("lecture" + ((currentLecture>9)? "":"0"))+currentLecture);
    }
}
```

Трябва за примера да е 3

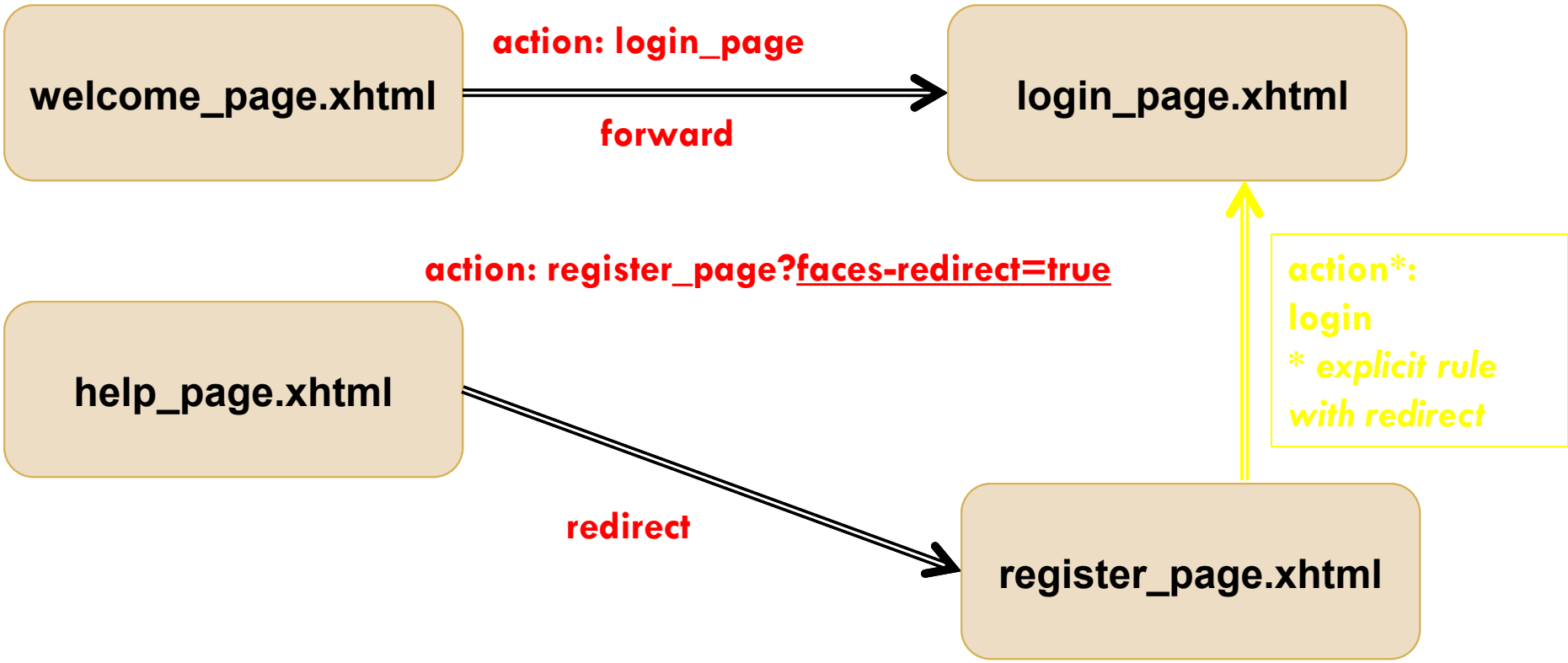
Алгоритъм на работа на JSF forward

- Браузъра изпраща “**GET**” заявка към URL : http://localhost:8080/JavaServerFaces/faces/start.xhtml.
- JSF приема заявката и връща страница “**start.xhtml**“.
- Браузърът визуализира страницата “**start.xhtml**“.
- Потребителят натиска бутон с резултат **page1**
- JSF приема действието и прави вътрешно препращане **(internal page forward)** към то “**page1.xhtml**” в сървъра.
- JSF връща страницата “**page1.xhtml**“.
- Браузърът визуализира страницата “**page1.xhtml**“.
- При препращане на страници (page forward), адресът на браузърът не се променя.

Алгоритъм на работа на JSF redirect

- Браузърът изпраща заявка “**GET**” с адреса URL : http://localhost:8080/JavaServerFaces/faces/start.xhtml.
- JSF приема заявката и връща “**start.xhtml**“.
- Браузърът визуализира “**start.xhtml**“.
- Потребителят натиска бутон.
- JSF приема действието и връща команда за пренасочване(“**redirect**”) към “**page1.xhtml**” обратно към браузърът
- Браузърът приема тази заявка и изпраща нова “**GET**” заявка към новият адрес URL : http://localhost:8080/JavaServerFaces/faces/page1.xhtml.
- JSF приема заявката и връща страницата “**page1.xhtml**“.
- Браузърът визуализира страницата “**page1.xhtml**“, и адресът в браузъра се обновява с новият адрес.

Навигиране към следваща страница



Пренасочване и препращане на заявка (1/2)

- forward & redirect:

Page forward: welcome_page.xhtml.

```
<h:form>
   <h:commandButton action="login_page" value="Go to login page" />
</h:form>
```

Page redirection:

help_page.xhtml

```
<h:form>
  <h:commandButton action=" register_page?faces-redirect=true"
value="Register here" />
</h:form>
```

Дефиниране на действие за forward

Пренасочване и препращане на заявка (2/2)

register_page.xhtml

```
<h:form>
   <h:commandButton action="login" value="Go to login page" />
</h:form>
```

file: faces-config.xml

```
<navigation-rule>
  <!– ENABLE redirect -->
  <from-view-id>register_page.xhtml</from-view-id>
  <navigation-case>
      <from-outcome>login</from-outcome>
      <to-view-id>login_page.xhtml</to-view-id>
       <redirect />
  </navigation-case>
</navigation-rule>
```

Смяна на действието по подразбиране чрез задаване на правило redirect

Включване на параметри при препращане (redirect)

```
<f:metadata>
 <f:viewParam name=“param1" value="#{bean.parameter1}"/>
....
</f:metadata>
...
<h:commandButton value="Enroll" action=“enroll_page?faces-redirect=true&amp;includeViewParams=true" />
...
```

В лекцията бяха разгледани

- Концепция за изглед
- Основните харктеристики на графичните компоненти
- Пример за графичен компонент
- Навигация между изгледи
- Графично представяне на правила за навигация
- Дефиниране на правила за навигация
  - Явни дефиниции на навигационни правила
  - Неявни правила за навигация
- Препращане и пренасочване (forward и redirect) в JSF 2.

Задача

- **Задача 1**
  - опишете кратък/сценарии и изисквания за разработка на собствено приложение (lab03_requirements.docx).
  - дефинирайте собствен фрагмент от faces-config.xml файл с няколко изгледа (страници) и няколко правила за навигация и различни (възможни) изходи;
  - Копирайте навигационните правила от faces-config.xml файла в .docx файл на документацията и
  - направете screenshot на визуализираната навигация (от графичният изглед на NetBeans или Eclipse).

Задача - упътване

Примери за навигация

- **Login management:**
  - login.xhtml/logout.xhtml/error_not_logged.xhtml/register.xhtml/unregister.xhtml/confirm_account.xhtml
- **User management:**
  - show_user_info.xhtml/edit_user_info.xhtml/change_password.xhtml/change_email.xhtml (with confirmation)/reset_password.xhtml/view_last_login.xhtml