Java EE:

JavaServer Faces

SoftUni Team

Technical Trainers

Software University

SoftUni Foundation

Software University

http://softuni.bg

Table of Contents

Questions

Software University

sli.do

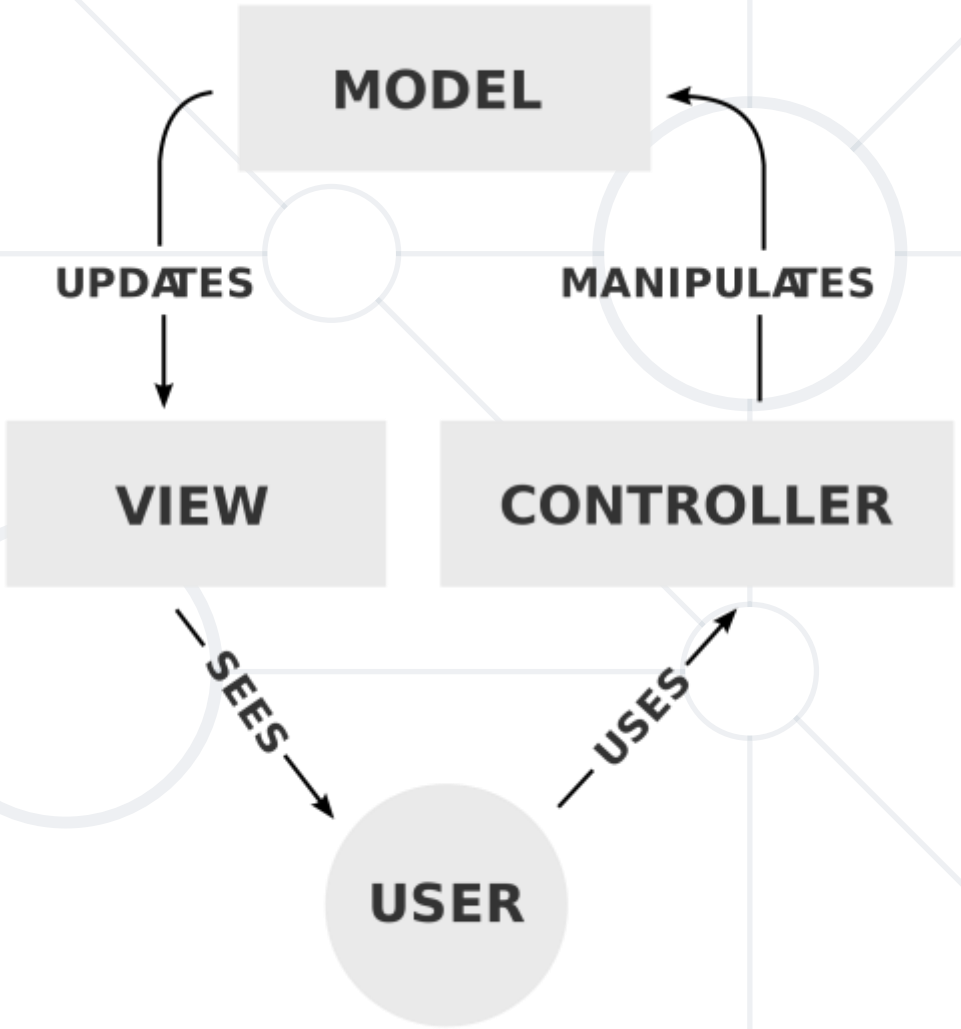
#java-web

Model-View-Controller

Overview

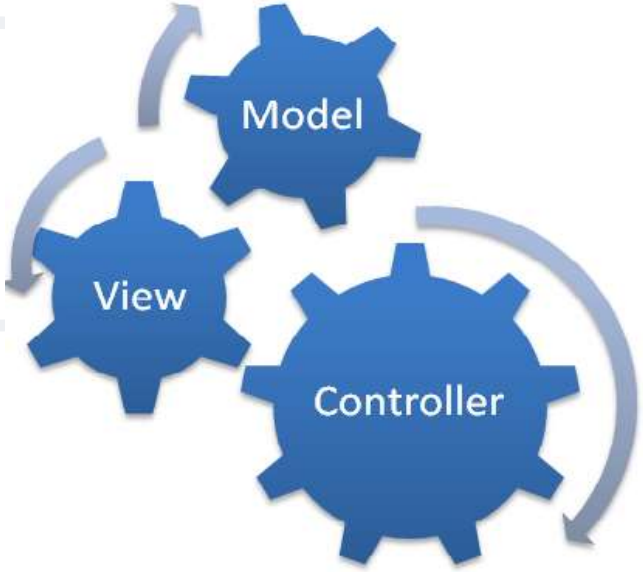
What is Model-View-Controller?(1)

- MVC:
  - Design pattern
  - Uses three separate modules:
    - Model - carries data
    - View - shows user interface
    - Controller - handles processing of an application



What is Model-View-Controller?(2)

- Purpose of MVC design pattern:
  - To separate model and presentation
  - Web designers have to concentrate only on view layer
  - Developers - on model and controller layer



JavaServer Faces

Overview

What are JavaServer Faces?(1)

Software University

- JavaServer Faces (JSF):
  - Java-based web application framework
  - Intended to simplify development integration of web-based user interfaces
  - Standardized display technology
  - Formalized in a specification through the Java Community Process:
    - https://jcp.org/en/jsr/detail?id=372

What are JavaServer Faces?(2)

- JavaServer Faces (JSF):
  - MVC web framework
  - Simplifies the construction of User Interfaces (UI) for server-based applications
  - Uses reusable UI components in a page
  - Provides a facility to connect UI widgets:
    - With data sources
    - Server-side event handlers

What are JavaServer Faces?(3)

Software University

- The JSF specification defines:
  - Set of standard UI components
  - Provides an Application Programming Interface (API) for developing components
- JSF enables:
  - Reuse of UI components
  - Extension of the existing standard UI components

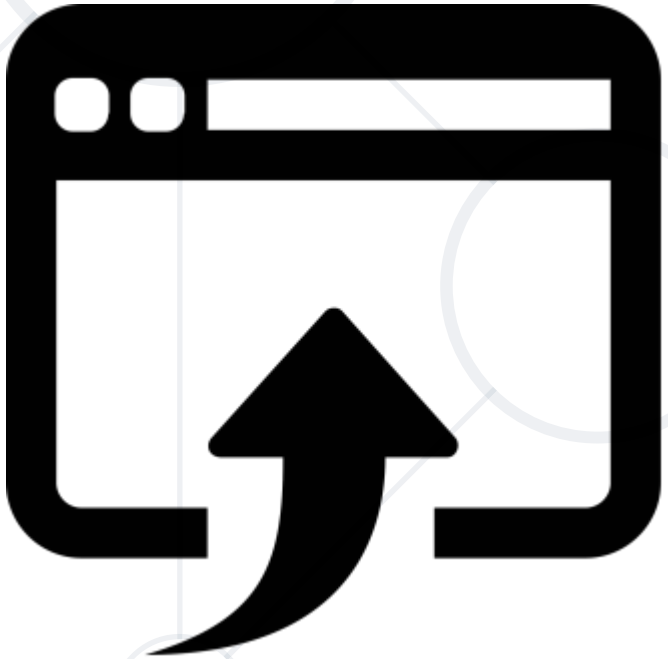
What are JavaServer Faces?(4)

Software University

- JSF reduces the effort in:
  - creating applications
  - maintaining applications
- JSF facilitates Web application development by:
  - Providing reusable UI components
  - Making easy data transfer between UI components
  - Managing UI state across multiple server requests
  - Enabling implementation of custom components
  - Wiring client-side event to server-side application code

What are JavaServer Faces?(5)

- JSF provides the developers with the capability to create Web application from collections of UI components that can render themselves in different ways for multiple client types:
  - HTML browser.
  - Wireless device.
  - WAP device.



What are JavaServer Faces?(6)

- JSF provides:
  - Core library.
  - A set of base UI components - standard HTML input elements.
  - Extension of the base UI components to create additional UI component libraries or to extend existing components.
  - Multiple rendering capabilities that enable JSF UI components to render themselves differently depending on the client types.

JSF Architecture(1)

Software University

- JSF application is:
  - Similar to any other Java technology-based web application
  - Runs in a Java servlet container
  - Contains:
    - JavaBeans components as models containing application-specific functionality and data
    - A custom tag library for:
      - Representing event handlers and validators
      - Rendering UI components

JSF Architecture(2)

- JSF application also contains:
  - UI components represented as stateful objects on the server
  - Server-side helper classes
  - Validators, event handlers, and navigation handlers
  - Application configuration resource file for configuring application resources

Managed Beans

Managed Beans

- Managed Bean:
  - Regular Java Bean class registered with JSF
  - Managed by JSF framework
  - Contains:
    - Getter and setter methods
    - Business logic
    - Backing bean - a bean that contains all the HTML form values
  - Works as Model for UI component
  - Can be accessed from JSF page

JSF and Managed Beans

Examples

Hello World!(1)

Software University

web.xml

```
<?xml version = "1.0" encoding = "UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
   http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         id="WebApp_ID" version="2.5">
    <welcome-file-list>
        <welcome-file>faces/index.xhtml</welcome-file>
    </welcome-file-list>
    <servlet>
        <servlet-name>Faces Servlet</servlet-name>
        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    …
```

Hello World!(2)

Software University

web.xml

```
    …
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.xhtml</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.jsf</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.faces</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>/faces/*</url-pattern>
    </servlet-mapping>
</web-app>
```

Hello World!(3)

HelloWorldBean.java

```
@Named
@RequestScoped
public class HelloWorld implements Serializable {
    private String message;
    public HelloWorld() {
        this.message = "Hello World!";
    }

    public String getMessage() {
        return this.message;
    }

    public void setMessage(String message) {
        this.message = message;
    }
}
```

Hello World!(4)

Software University

index.xhtml

```
<?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">

<h:head>
    <title>JSF Demo</title>
</h:head>

<h:body>
    <h1>#{helloWorld.message}</h1>
</h:body>

</html>
```

Register Users!(1)

RegisterUserBean.java

```
@Named
@RequestScoped
public class UserRegisterBean {
    private String username;
    private String password;
    private final UserRepository userRepository;
    @Inject
    public UserRegisterBean(UserRepository userRepository) {
        this.userRepository = userRepository;
    }

    // Getters and Setters

    public void register() {
        User user = new User(this.username, this.password);
        this.userRepository.save(user);
    }
}
```

Register Users!(2)

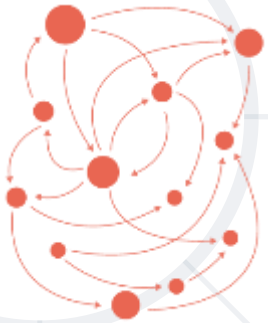
Software University

index.xhtml

```
<?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">
<h:head>
    <title>JSF Demo</title>
</h:head>
<h:body>
    <label for="usernameInput" />
    <h:outputText id="usernameInput" value="#{userRegisterBean.username}"/>
    <label for="passwordInput" />
    <input type="password" id="passwordInput" jsfc="h:inputSecret"
value="#{userRegisterBean.password}" />
    <h:commandButton value="Register" action="#{userRegisterBean.register}"/>
</h:body>
</html>
```

Summary

- MVC:
  - Model
  - View
  - Controller
- JavaServer Faces:
  - MVC web framework
- Managed Beans:
  - Java bean managed by JSF framework



Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

SoftUni Diamond Partners

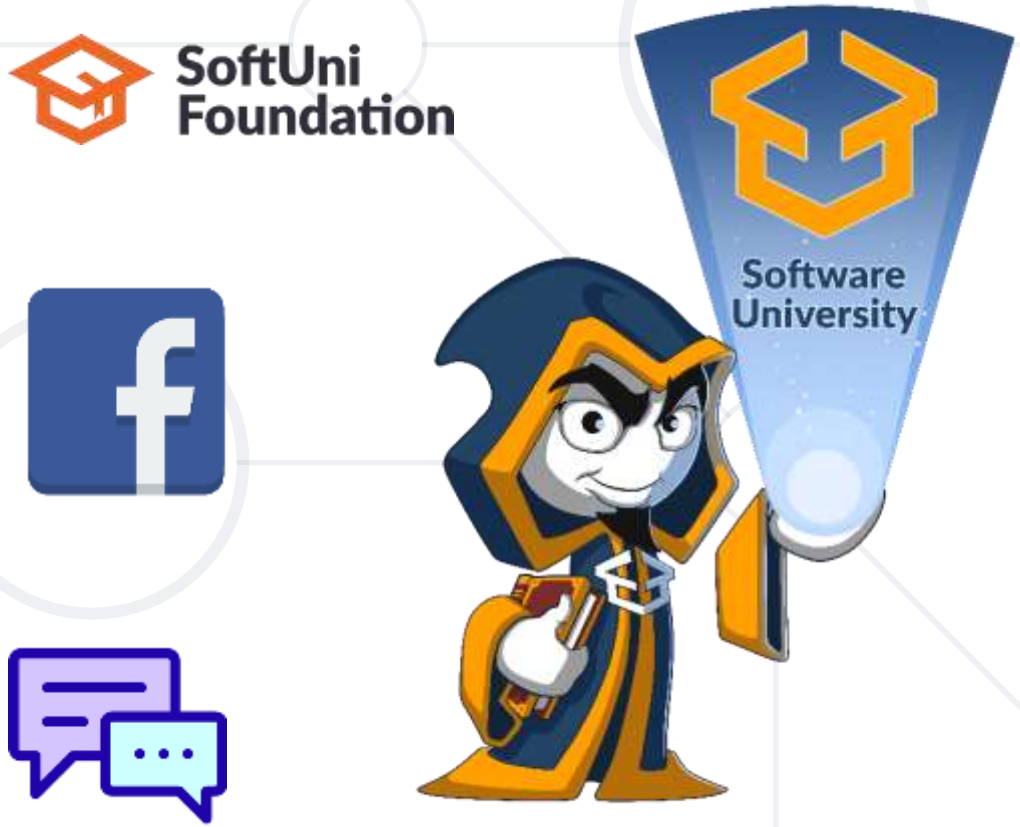
Software University

SoftUni Organizational Partners

Software University

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg



License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license