Basic Web Project

MVC, Spring and Thymeleaf

SoftUni Team

Technical Trainers

Software University

Software University

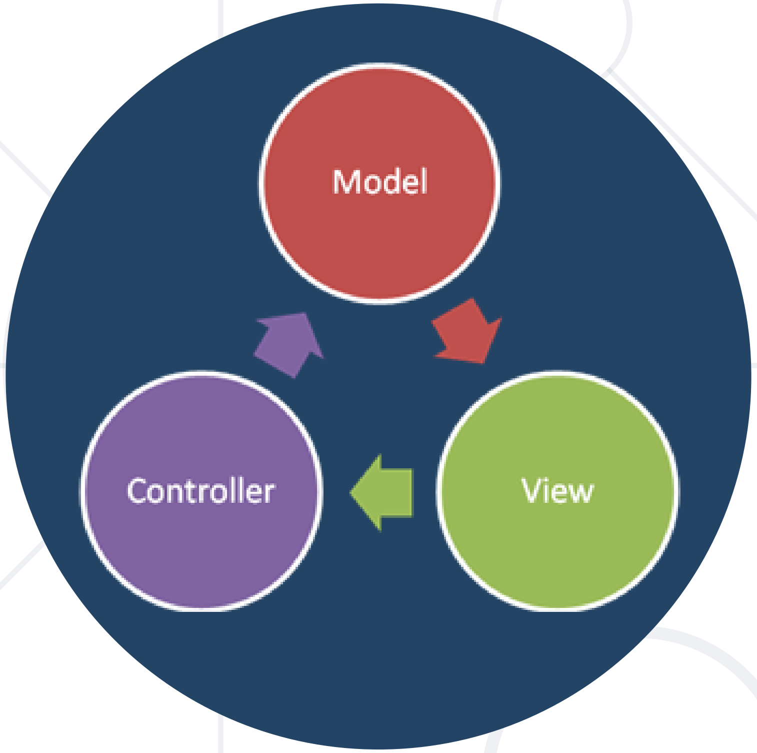
https://softuni.bg

Have a Question?

sli.do

#fund-java

Table of Contents



MVC

Model-View Controller

What is Model-View Controller

- **MVC** == Model-View-Controller
- **Views** (presentation / UI)
  - Render UI (produce HTML)
- **Controllers** (logic)
  - Prepare UI (presentation logic)
  - Update database (business logic)
- **Models** (data)
  - Data access classes or ORM

Model (Data)

- Set of **classes** that describes the **data** we are working with
- Rules for **how** the data can be **changed** and **manipulated**
- May contain **data validation rules**
- Often **encapsulates** data stored in a database

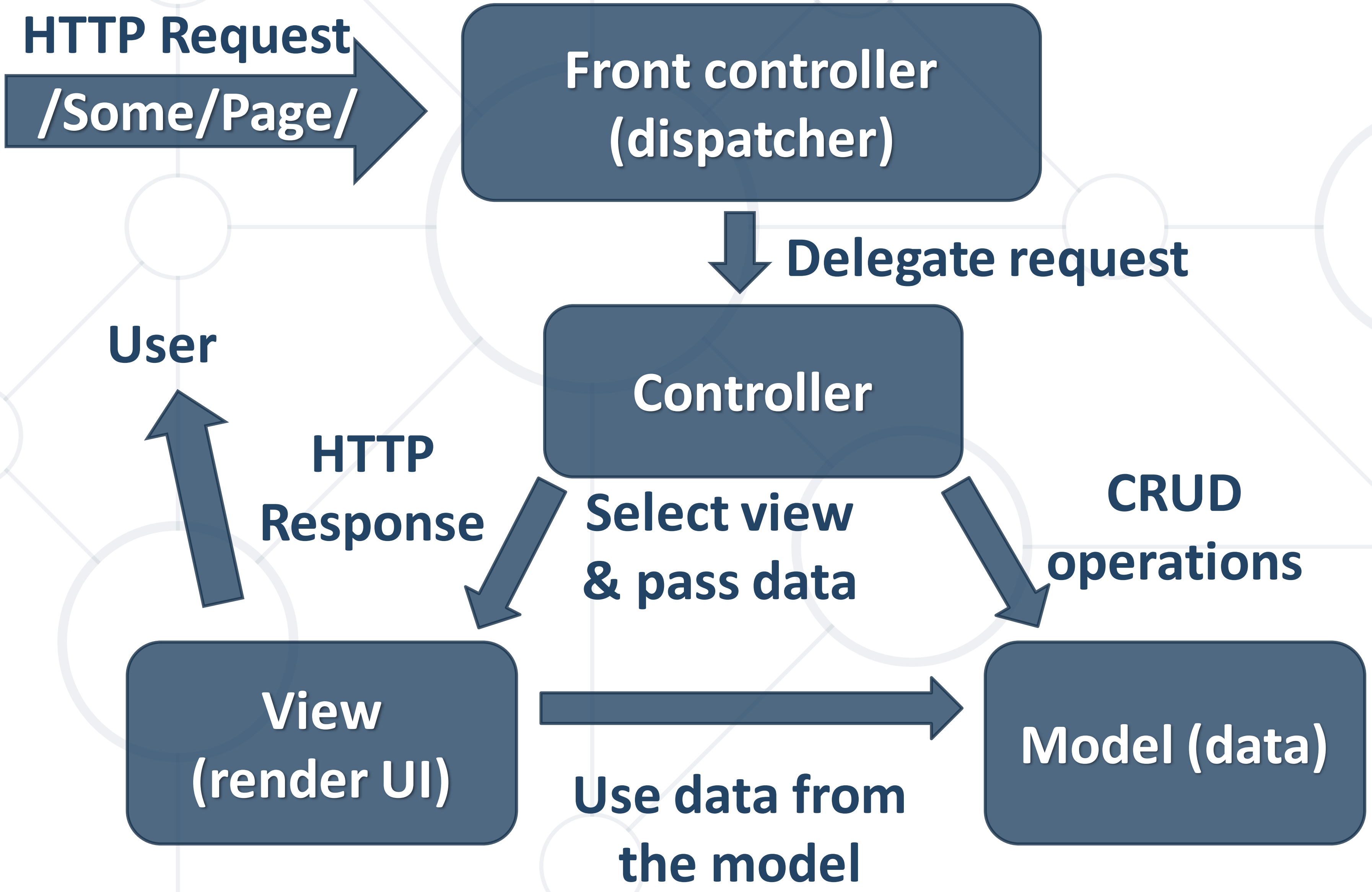
View

- Defines how the application's **user interface** (UI) will be displayed
- May support master views (**layouts**)
- May support sub-views (**partial views** or controls)
- May use **templates** to **dynamically generate** HTML

Controller

- The **core** MVC component - holds the **logic**
- Process the requests
- A set of classes that handles
  - **Communication** from the user
  - Overall application **flow**
  - Application-specific **logic** (business logic)
- Every controller has one or more "**actions**"

The MVC Pattern



Spring MVC

Spring MVC

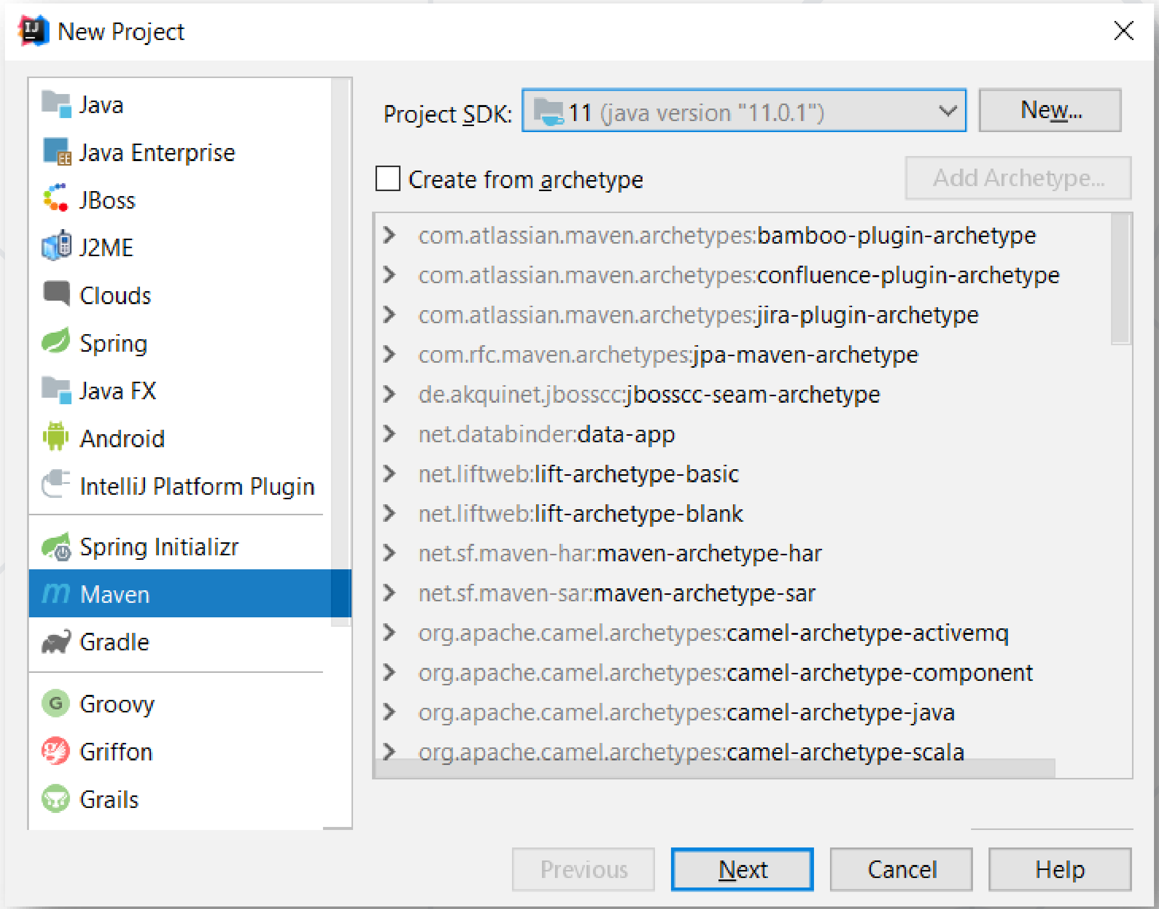
- Spring MVC == open source Web MVC framework for Java
  - Developed by Pivotal Software
  - https://spring.io
- Built top of Java Servlet API

Spring Boot

- Simplifies building Spring applications
- Convention-over-configuration
  - Rapid application development with Spring
  - Create production-grade applications that you can "**just run**"
  - **Automatically** configure Spring Framework
- Built-in Web server (Tomcat)
- Integrates Spring MVC, Spring Data and other Spring technologies

Starting with Spring Boot

- Create a new Maven-based Java project

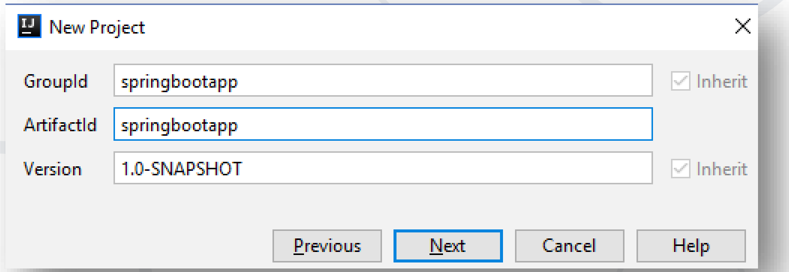


New Project

Project SDK: 11 (java version "11.0.1")

- [ ] Create from archetype

- com.atlassian.maven.archetypes:bamboo-plugin-archetype
- com.atlassian.maven.archetypes:confluence-plugin-archetype
- com.atlassian.maven.archetypes:jira-plugin-archetype
- com.rfc.maven.archetypes:jpa-maven-archetype
- de.akquinet.jbosscc:jbosscc-seam-archetype
- net.databinder:data-app
- net.liftweb:lift-archetype-basic
- net.liftweb:lift-archetype-blank
- net.sf.maven-har:maven-archetype-har
- net.sf.maven-sar:maven-archetype-sar
- org.apache.camel.archetypes:camel-archetype-activemq
- org.apache.camel.archetypes:camel-archetype-component
- org.apache.camel.archetypes:camel-archetype-java
- org.apache.camel.archetypes:camel-archetype-scala



New Project

GroupId	springbootapp	Inherit
ArtifactId	springbootapp	
Version	1.0-SNAPSHOT	Inherit

Starting with Spring Boot

pom.xml

```xml
<parent>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-parent</artifactId>
   <version>2.0.4.RELEASE</version>
</parent>
<dependencies>
   …
</dependencies>
<properties><java.version>11</java.version></properties>
```

Starting with Spring Boot

pom.xml

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

Spring Boot Application Class

src/main/java/app/MvcAppExample.java

```java
package app;

import org.springframework.boot.*;
import org.springframework.boot.autoconfigure.*;

@SpringBootApplication
public class MvcAppExample {
  public static void main(String[] args) {
    SpringApplication.run(MvcAppExample.class, args);
  }
}
```

Spring Annotations

- Spring uses strongly-typed annotations
  - Syntax highlighting + error checking
  - Describe the code below them

```
@Controller
public class HomeController {
    …
}
```

```
@GetMapping("/hello")
public ModelAndView hello() {
    …
}
```

Spring Controllers

- MVC **controllers** hold **actions**, mapped to **URL** by **annotations**
- Defined with **@Controller** annotations

```
@Controller
public class HomeController {
    …
}
```

- Controllers can hold multiple actions on different routes

```
@GetMapping("home")
public ModelAndView home (ModelAndView modelAndView) { … }
```

Mapped to http://localhost:8080/hello

Controller Actions

- GetMapping – GET Request

```
@GetMapping("/home")
public ModelAndView home(ModelAndView modelAndView) {
   …
}
```

- PostMapping – POST Request

```
@PostMapping("/register")
public ModelAndView register(ModelAndView modelAndView) {
   …
}
```

Spring Controller: Example

- Create Web controller + action /hello + view hello.html

```
@Controller
public class GreetingController {
   @GetMapping("/hello")
   public ModelAndView home(ModelAndView modelAndView) {
      modelAndView.setViewName("hello.html");
      return modelAndView;
   }
}
```

HTML File in resources/templates/hello.html

Thymeleaf

Template Engine

Thymeleaf

- Thymeleaf is a view engine used in **Spring MVC**
  - Natural templates – HTML with additional attributes to add view logic
- Thymeleaf allows us to:
  - Use **variables** / **collections** in our views
  - Execute **operations** on our variables
  - **Iterate** over **collections**

Thymeleaf Tags and Attributes

- All Thymeleaf tags and attributes begin with **th:**
- Example of Thymeleaf attribute

```
<p th:text="Example">…</p>
```

- **th:block** is an attribute container that disappears in the HTML

```
<th:block>
  …
</th:block>
```

Thymeleaf Variable Expressions

- Variable Expressions are executed on the context variables

```
${ … }
```

- Examples:

```
${title}
```

```
${article.title}
```

```
${article.author.name}
```

Thymeleaf Link Expressions

- Link Expressions are used to build URLs

```
@{ … }
```

- Example:

```
<a th:href="@{/register}">Register</a>
```

- You can also pass query string parameters

```
<a th:href="@{/details(id=${game.id})}">Details</a>
```

- Create dynamic URLs

```
<a th:href="@{/games/{id}/edit(id=${game.id})}">Edit</a>
```

Forms in Thymeleaf

- In Thymeleaf you can create HTML forms:

```
<form th:action="@{/user}" th:method="post">
   <input type="number" name="id"/>
   <input type="text" name="name"/>
   <input type="submit">
</form>
```

- You can parse the input as an object

```
@PostMapping("/user")
public ModelAndView register(@ModelAttribute User user) { … }
```

Conditional Statements in Thymeleaf

- You can use if statements in thymeleaf using **th:if**

```
<div th:if="${…}">
  <p>The statement is true"</p>
</div>
```

- You can create inverted if statements using **th:unless**

```
<div th:unless="${…}">
  <p>The statement is false"</p>
</div>
```

Loops in Thymeleaf

- For loop

```
<div th:each="element :
                  ${#numbers.sequence(start, end, step)}">
   <p th:text="${element}"></p>
</div>
```

- Example:

```
<div th:each="element : ${#numbers.sequence(1, 5, 1)}">
   <p th:text="${element}"></p>
</div>
//1 2 3 4 5
```

Loops in Thymeleaf

- For-each loop

```
<div th:each="item : ${collection}">
   <p th:text="${item.property}"></p>
</div>
```

- Example

```
<div th:each="book : ${books}">
   <p th:text="${book.name}"></p>
   <p th:text="${book.author}"></p>
   <p th:text="${book.price}"></p>
</div>
```

Passing Attributes to View

- Passing a string to the view

```
<body>
   <p>Hello, <span th:text="${name}"></span></p>
</body>
```

```
@GetMapping("/hello")
public ModelAndView hello(ModelAndView modelAndView) {
   modelAndView.setViewName("hello");
   modelAndView.addObject("name", "Peter");
   return modelAndView;
}
```

Passing Attributes to View

- Passing a collection to the view

```
<div th:each="book : ${books}">
    <p th:text="${book.name}"></p>
</div>
```

```
@GetMapping("/all")
public ModelAndView listBooks(ModelAndView modelAndView) {
   …
   modelAndView.addObject("books", books);
   return modelAndView;
}
```

Summary

- Implementing MVC Pattern
- Spring MVC
  - Open Source Framework for Java
- Spring Boot
  - Configures and simplifies Spring apps
- Thymeleaf
  - Powerful view engine
  - Expressions, Conditions and Iterations

Software University

Questions?

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

Finance Academy

SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SOFTWARE GROUP

DXC TECHNOLOGY

createX

Postbank
Решения за твоето утре

SUPER HOSTING .BG

DRAFT KINGS

VIVACOM

INDEAVR
Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License