

## **BÀI 1: GIỚI THIỆU SPRING MVC**

Giảng viên: Ths Nguyễn Trung Hiếu

Email: [hieunt.tg@ptithcm.edu.vn](mailto:hieunt.tg@ptithcm.edu.vn)

Mobile/Zalo:098.305.1825

Cách tính điểm :

Chuyên cần:10%

Điểm kiểm tra : 7 bài Lab.

Thi lần 1: Báo cáo project.

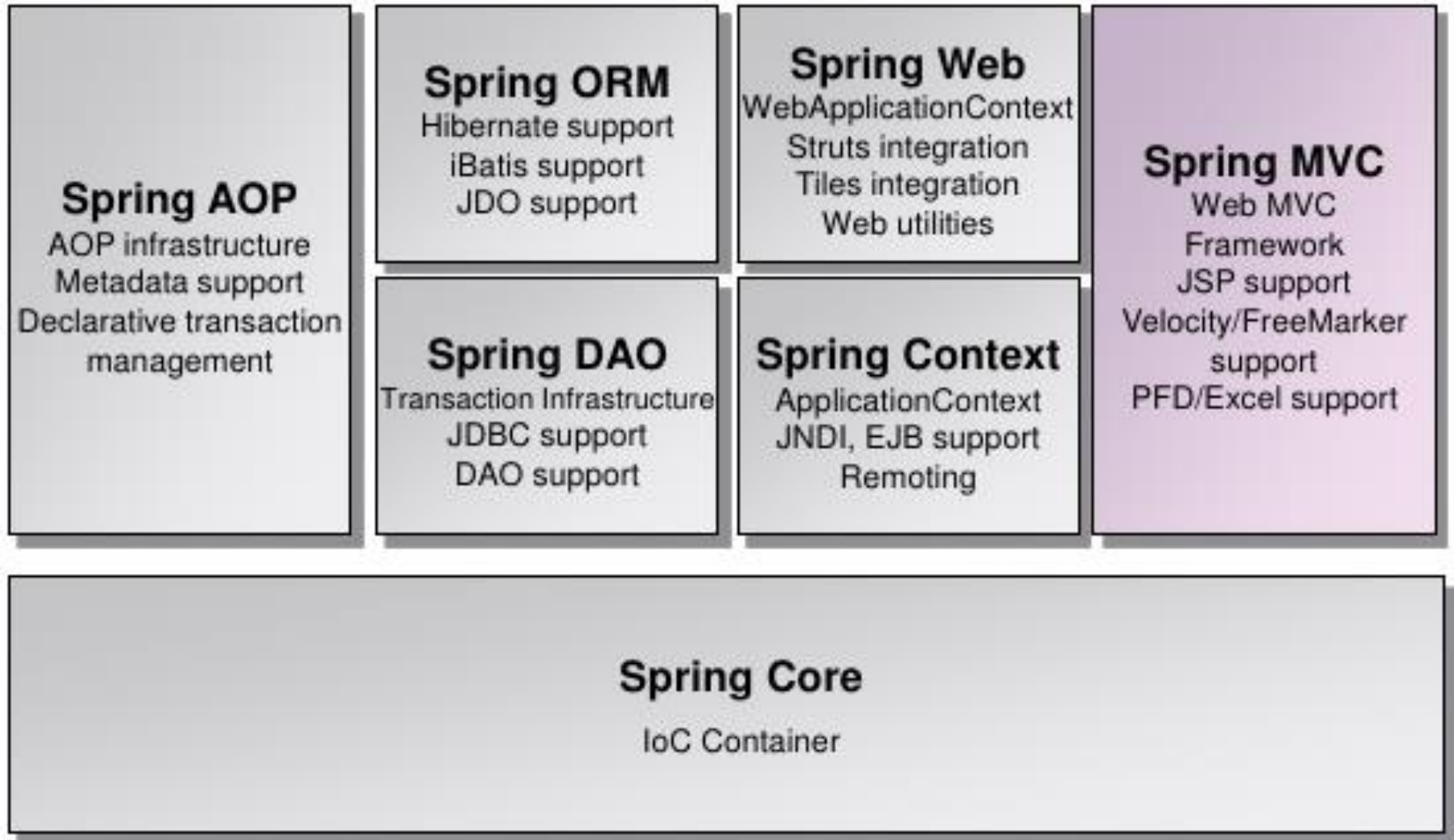
Thi lần 2: Làm bài test 60' ở phòng thực hành.

- ⦿ Hiểu Spring Framework
- ⦿ Nắm mô hình hoạt động Spring MVC
- ⦿ Thiết lập môi trường
- ⦿ Tạo dự án Spring MVC
  - ⦿ Tạo Controller
  - ⦿ Tạo View
  - ⦿ Cấu hình ứng dụng
- ⦿ Làm việc với các đối tượng web
- ⦿ Truyền dữ liệu từ Controller sang View

# GIỚI THIỆU SPRING FRAMEWORK

- ❑ Spring framework là nền tảng mã nguồn mở. Nó cung cấp cơ sở hạ tầng toàn diện để phát triển ứng dụng Java một cách mạnh mẽ, rất dễ dàng và nhanh chóng.
- ❑ Spring framework được tạo bởi Rod Johnson và được giới thiệu vào tháng 6 năm 2003.
- ❑ Spring là framework phát triển ứng dụng Java phổ biến nhất đối với doanh nghiệp.
- ❑ Spring Framework được hàng triệu nhà phát triển ứng dụng trên toàn thế giới sử dụng để tạo ra các sản phẩm phần mềm với hiệu suất cao, dễ dàng kiểm chứng, tái sử dụng mã.

# KIẾN TRÚC SPRING FRAMEWORK



# THÀNH PHẦN SPRING FRAMEWORK

## ☐ Spring Core

- ❖ Cung cấp nền tảng cơ bản của hệ thống ứng dụng Spring

## ☐ Spring AOP

- ❖ Cung cấp nền tảng cho lập trình hướng khía cạnh

## ☐ Spring DAO

- ❖ Cung cấp dụng cụ đối tượng truy xuất dữ liệu

## ☐ Spring Context

- ❖ Cung cấp dịch vụ truy cập từ xa như JNDI, EJB...

## ☐ Spring MVC

- ❖ Nền tảng ứng dụng web theo mô hình MVC

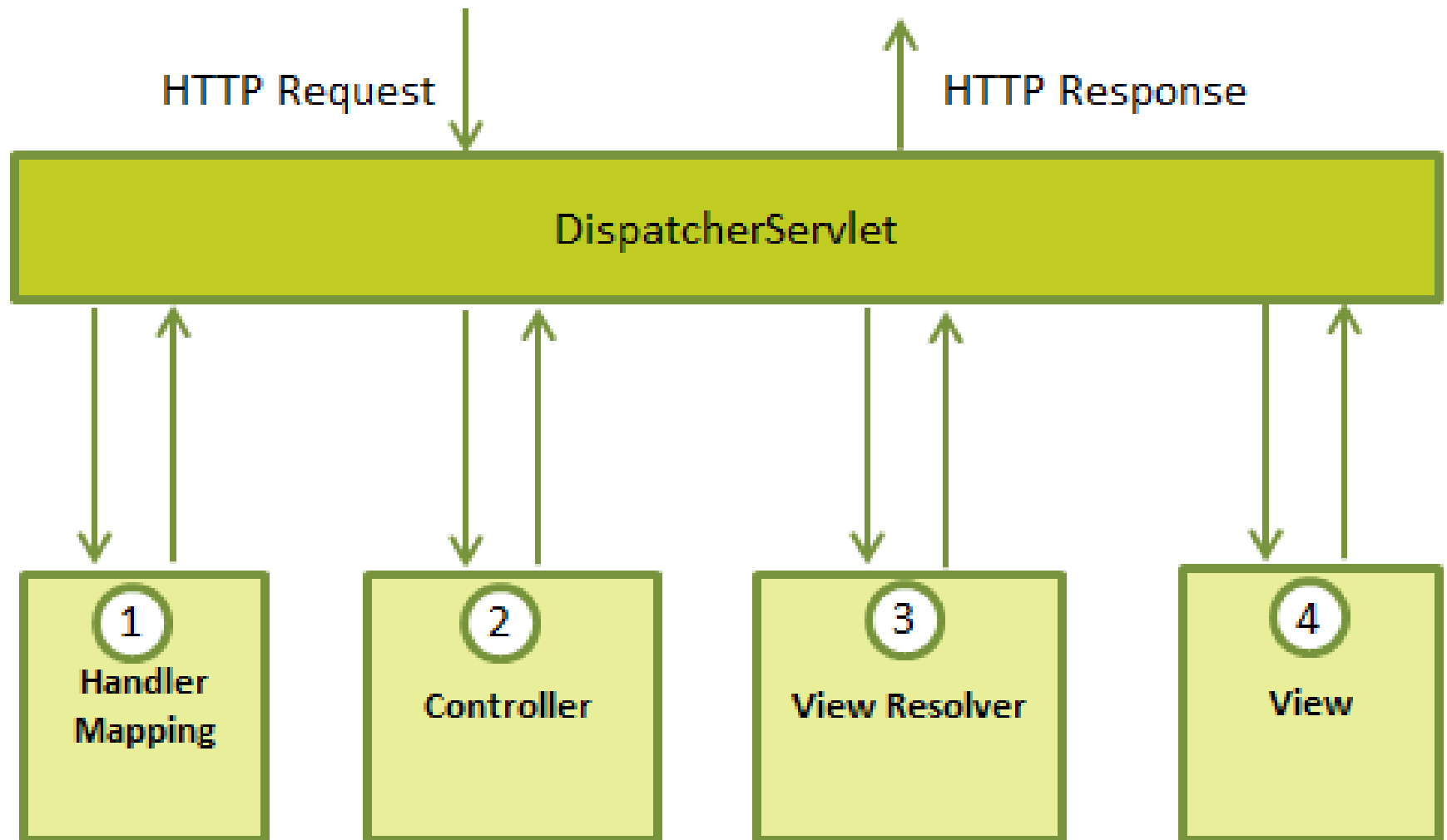
## ☐ Spring ORM

- ❖ Cung cấp dịch vụ ánh xạ đối tượng quan hệ dữ liệu

## ☐ Spring Web

- ❖ Cung cấp dịch vụ tích hợp các framework web khác

# XỬ LÝ REQUEST TRONG SPRING MVC



## DIỄN GIẢI QUI TRÌNH XỬ LÝ REQUEST

- ❑ DispatcherServlet tiếp nhận mọi yêu cầu từ người dùng và thực hiện điều phối qua 4 bước sau
  1. Chuyển URL cho bộ phận **Handler Mapping** để lấy **action method** muốn gọi
  2. Gọi **action method** trong **Controller** và nhận kết quả
  3. Chuyển kết quả cho bộ phận **ViewResolver** để lấy đường dẫn **View**
  4. Gọi **View** để kết xuất kết quả cho client

# THIẾT LẬP MÔI TRƯỜNG PHÁT TRIỂN

## □ Môi trường cần thiết cho khóa học

- ❖ **JDK 7+** là nền tảng bắt buộc cho việc phát triển và chạy ứng dụng Java
- ❖ **Eclipse for JavaEE developer** là một IDE được sử dụng phổ biến nhất ở các doanh nghiệp sản xuất phần mềm để phát triển ứng dụng web với Java
- ❖ **Tomcat 8x** là web server được sử dụng để triển khai ứng dụng web
- ❖ **SQL Server 2008+** là hệ quản trị CSDL quan hệ được sử dụng để lưu trữ và quản lý dữ liệu



# THIẾT LẬP MÔI TRƯỜNG

## ❑ Download JDK và cài đặt

❖ <http://download.oracle.com/otn-pub/java/jdk/8u112-b15/jdk-8u112-windows-x64.exe>

## ❑ Download SQL Server Express và cài đặt

❖ [http://download.microsoft.com/download/8/D/D/8DD7BDBA-CEF7-4D8E-8C16-D9F69527F909/ENU/x64/SQLManagementStudio\\_x64\\_ENU.exe](http://download.microsoft.com/download/8/D/D/8DD7BDBA-CEF7-4D8E-8C16-D9F69527F909/ENU/x64/SQLManagementStudio_x64_ENU.exe)

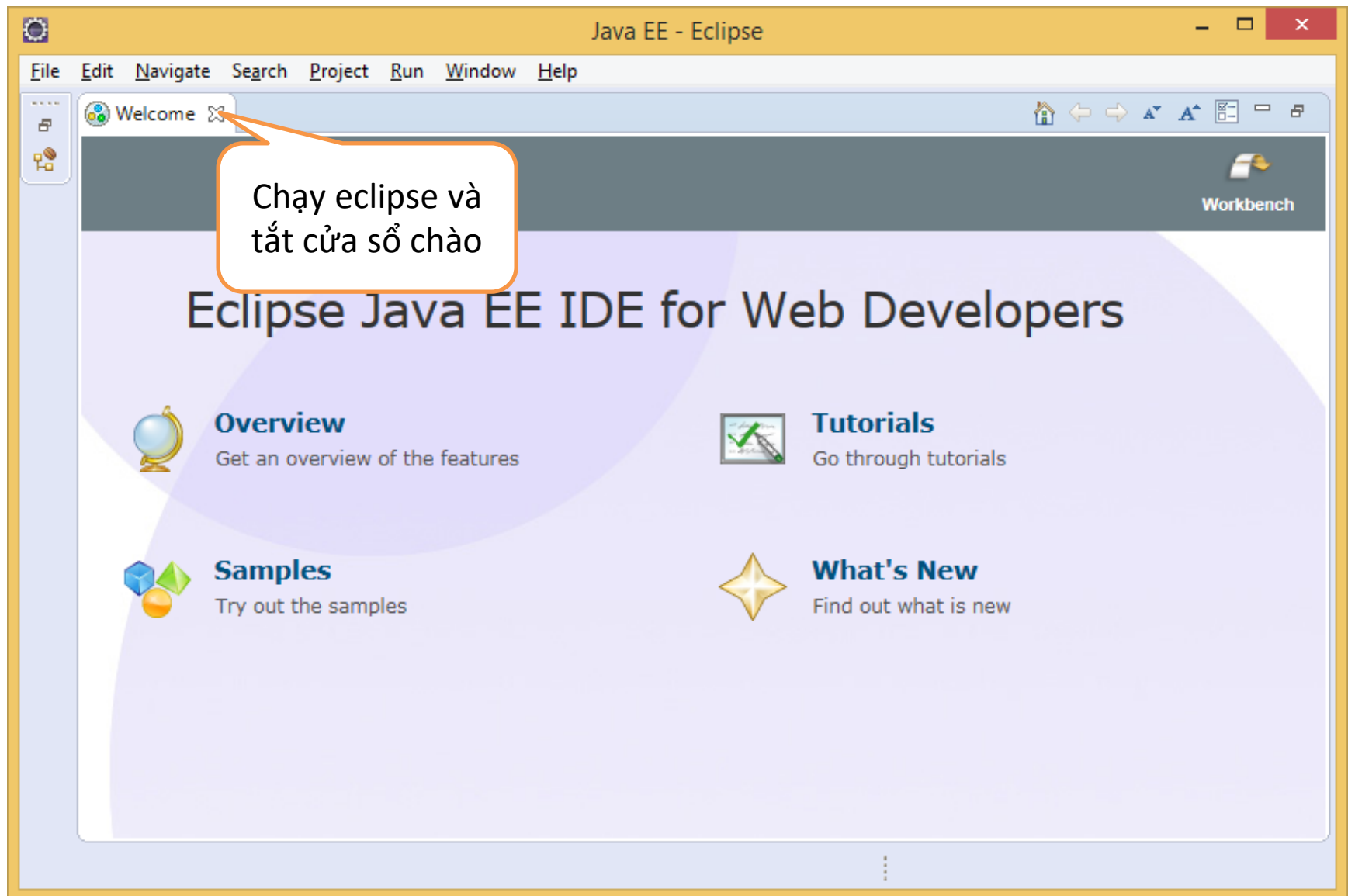
## ❑ Download Eclipse và giải nén vào thư mục thích hợp

❖ [http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/mars/R/eclipse-jee-mars-R-win32-x86\\_64.zip&mirror\\_id=448](http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/mars/R/eclipse-jee-mars-R-win32-x86_64.zip&mirror_id=448)

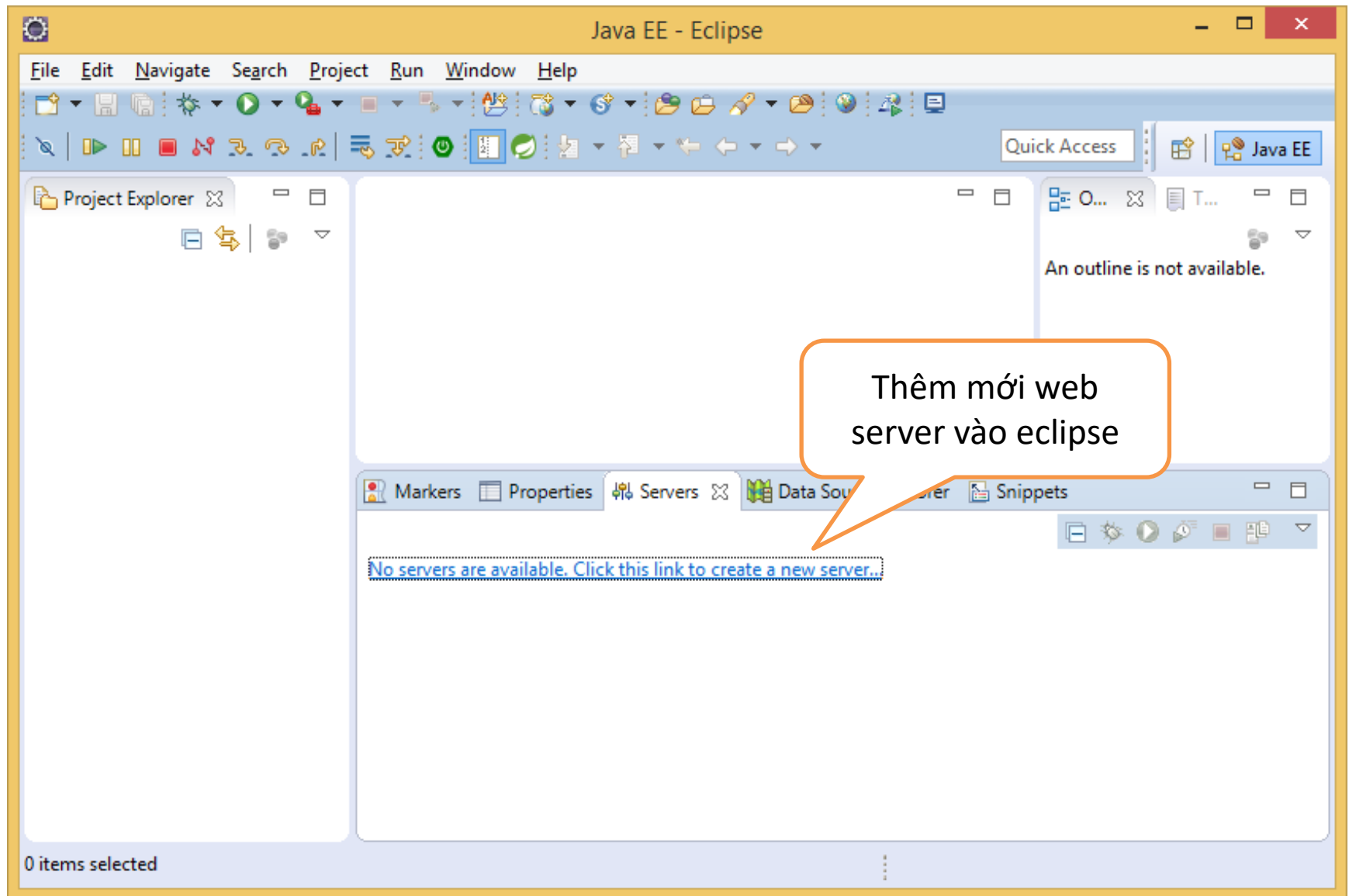
## ❑ Download Tomcat và giải nén vào thư mục thích hợp

❖ <http://www-us.apache.org/dist/tomcat/tomcat-8/v8.5.8/bin/apache-tomcat-8.5.8-windows-x64.zip>

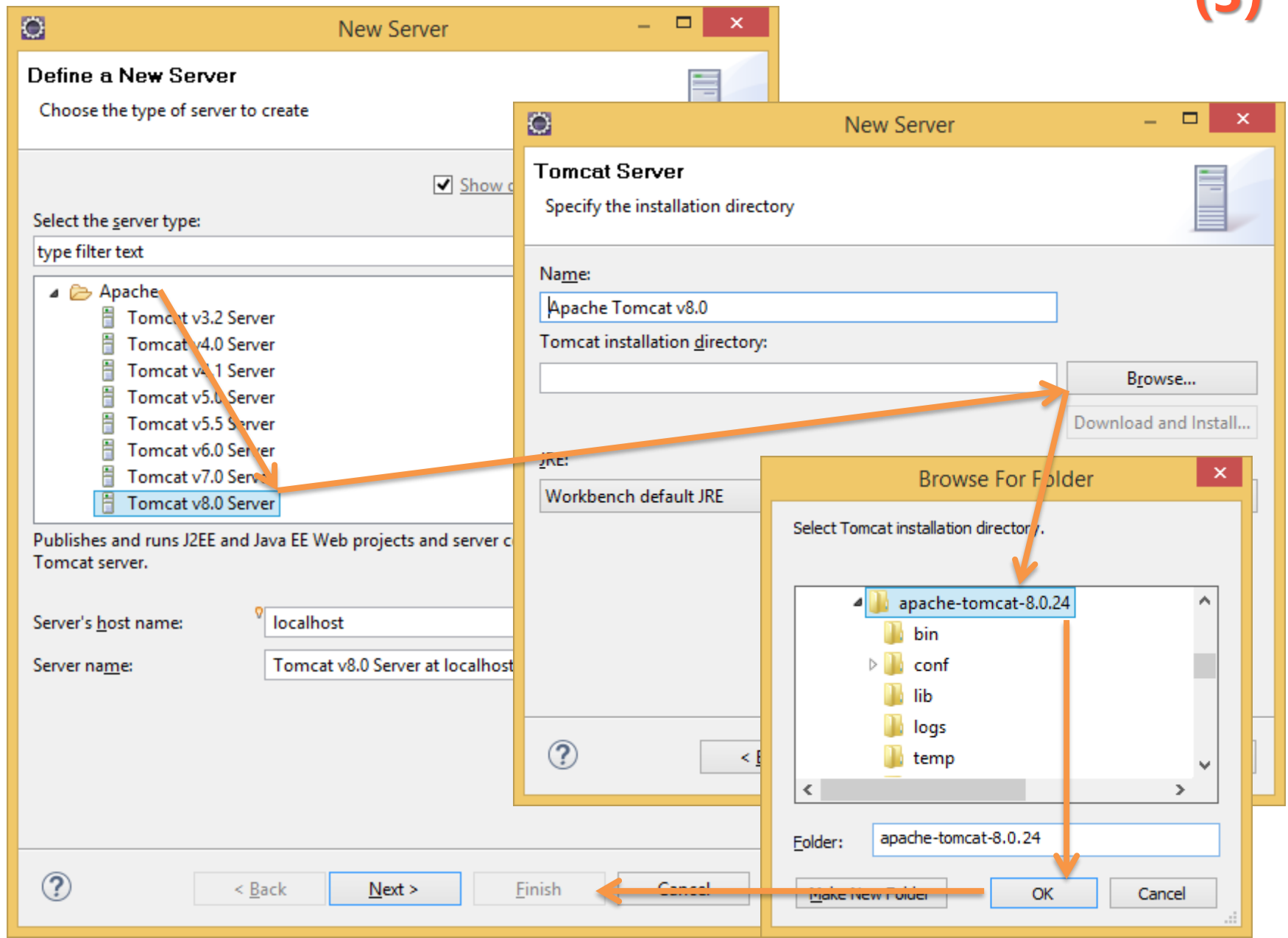
# TÍCH HỢP TOMCAT VÀO ECLIPSE IDE (1)



# TÍCH HỢP TOMCAT VÀO ECLIPSE IDE (2)

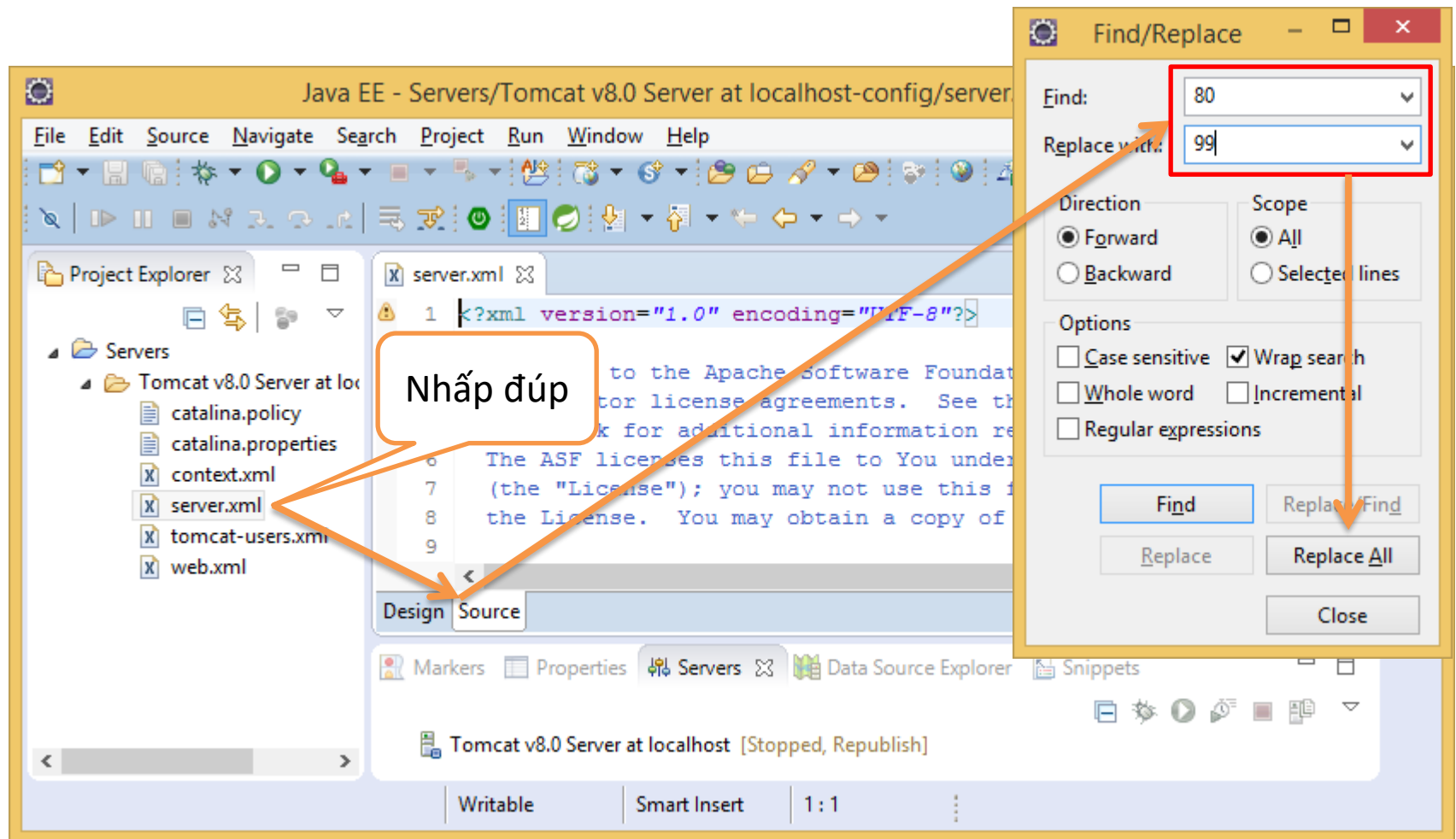


# TÍCH HỢP TOMCAT VÀO ECLIPSE IDE (3)

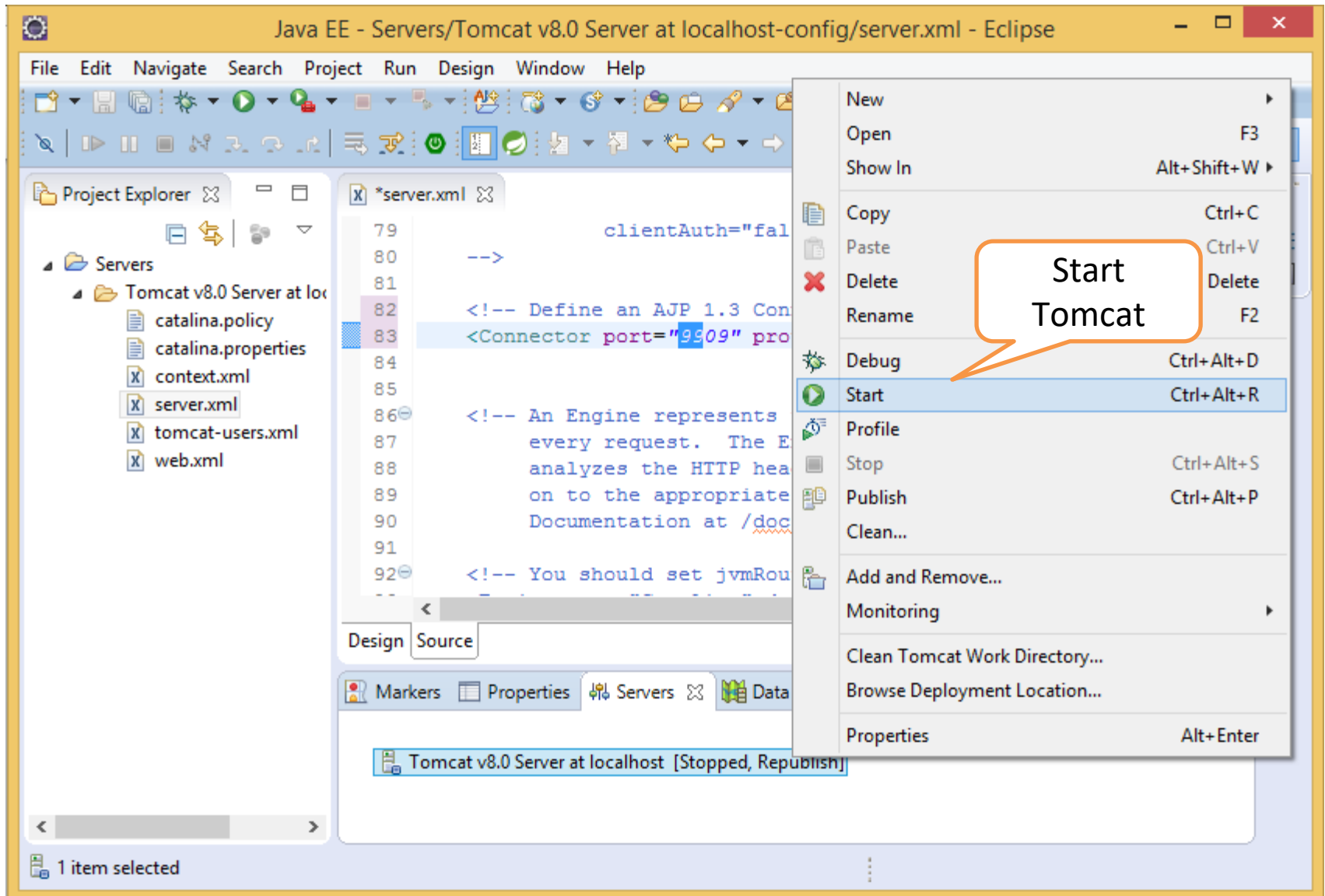


## TÍCH HỢP TOMCAT VÀO ECLIPSE IDE (4)

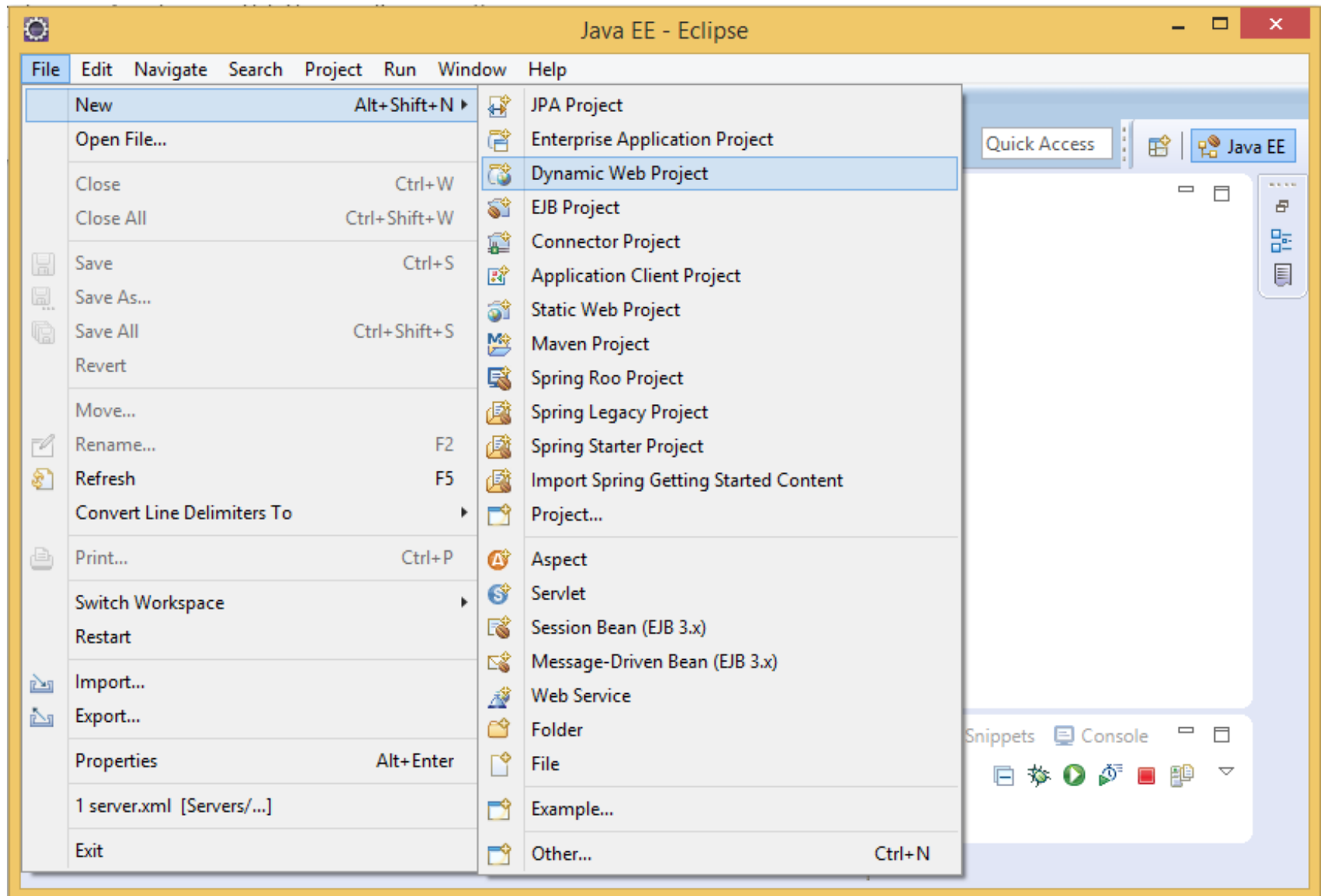
- ## ❑ Thay thế port tomcat tránh đụng port khi chạy



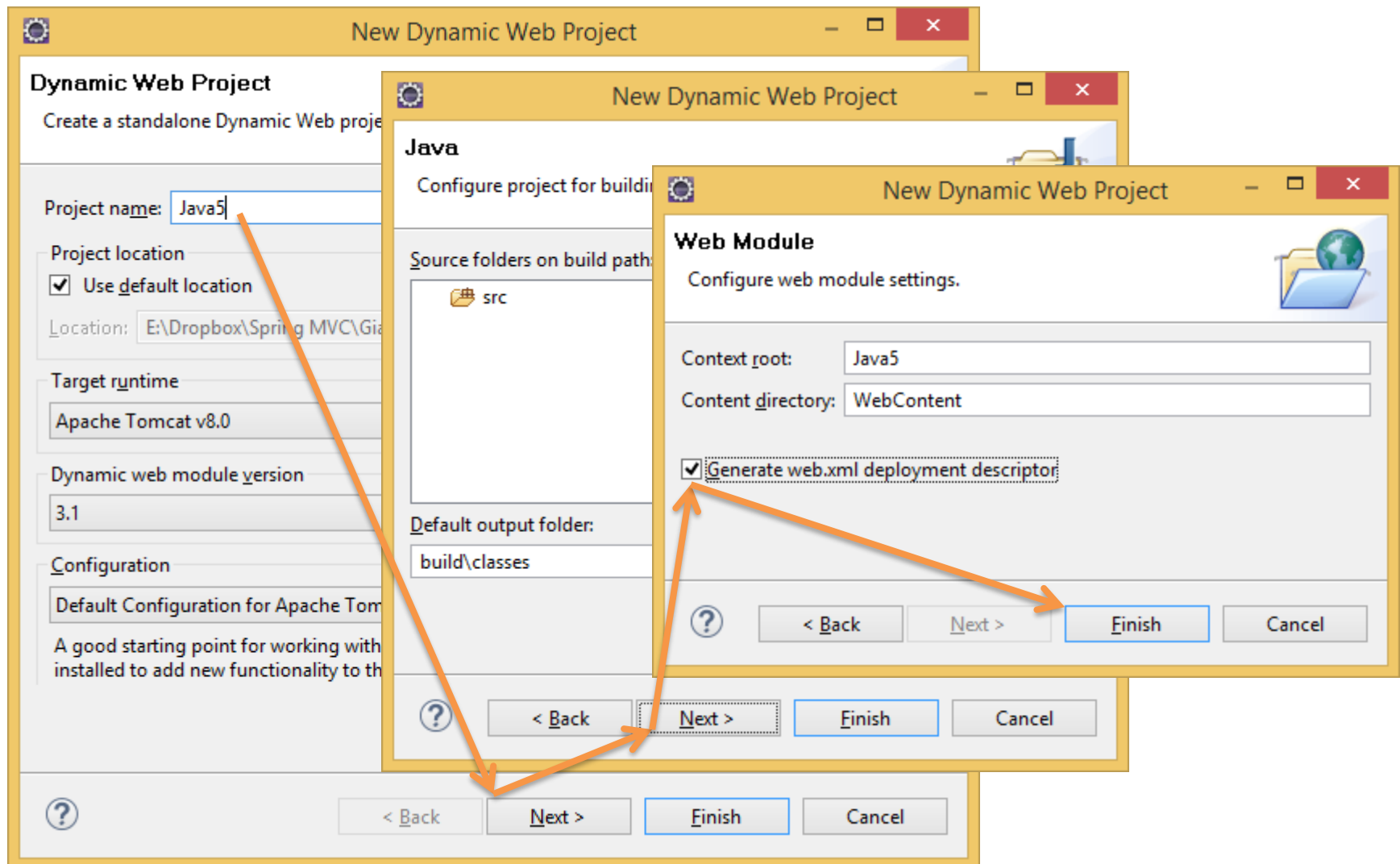
# TÍCH HỢP TOMCAT VÀO ECLIPSE IDE (5)



# TẠO DỰ ÁN WEB (1)

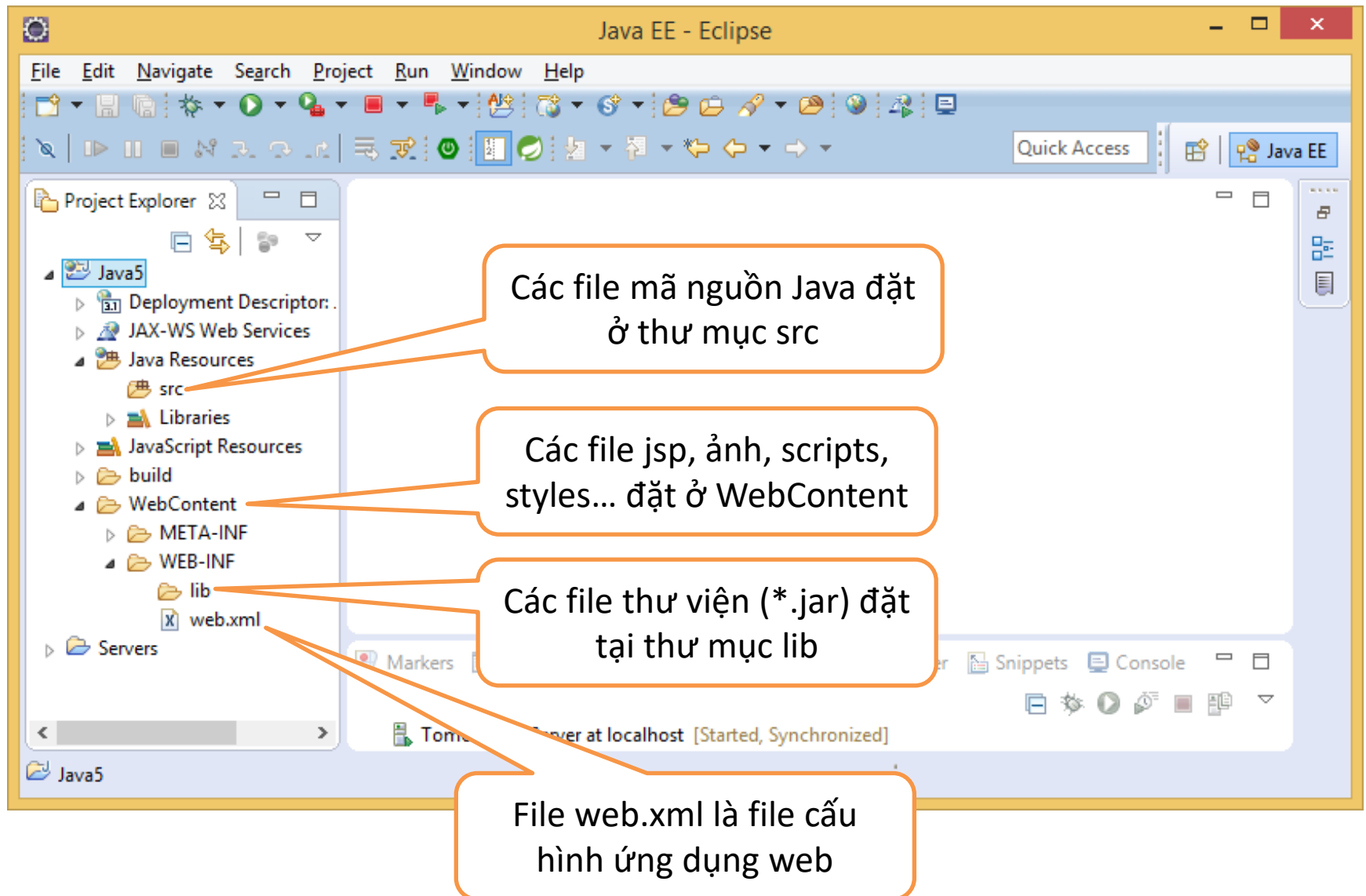


## TẠO DỰ ÁN WEB (2)

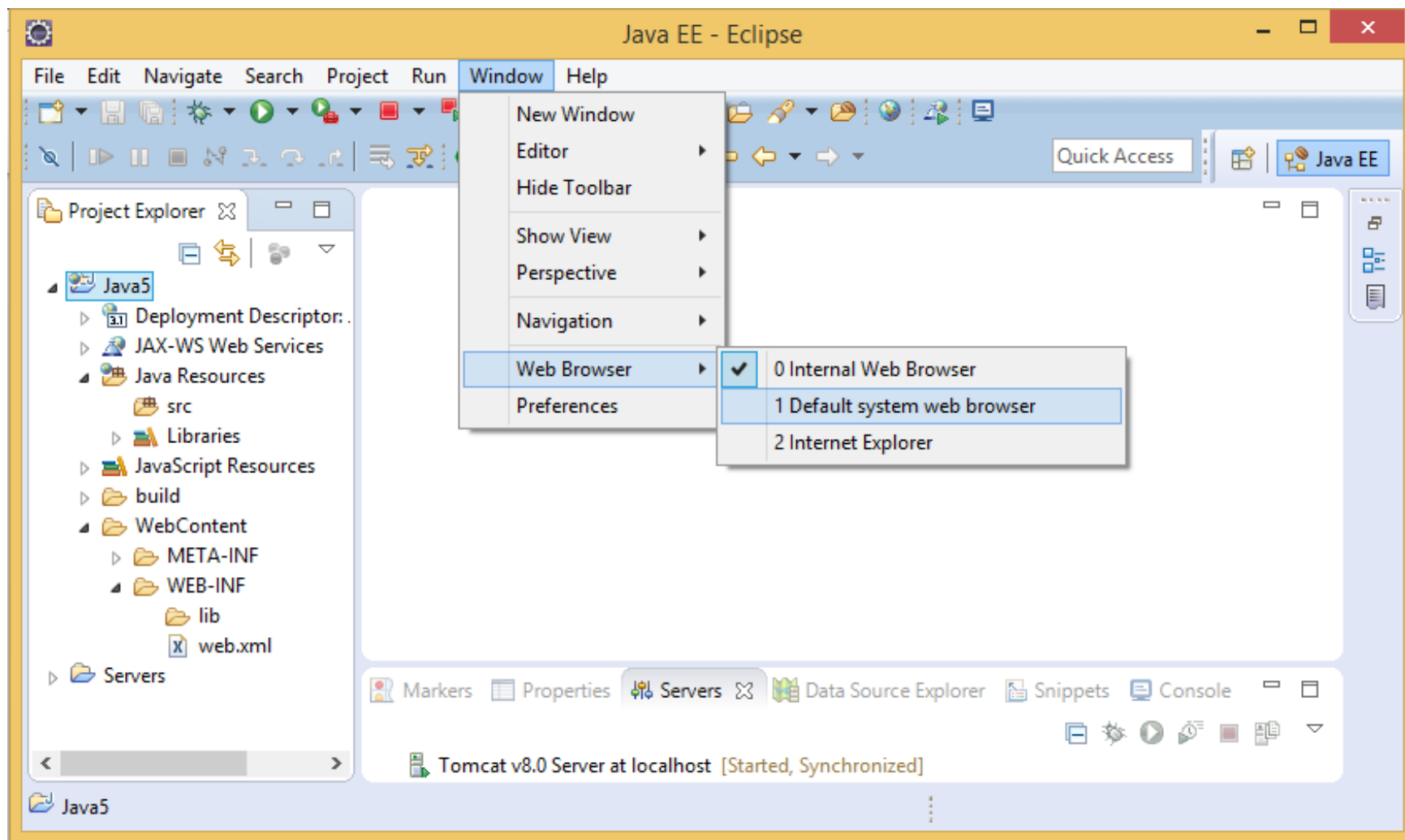




# TỔ CHỨC DỰ ÁN WEB

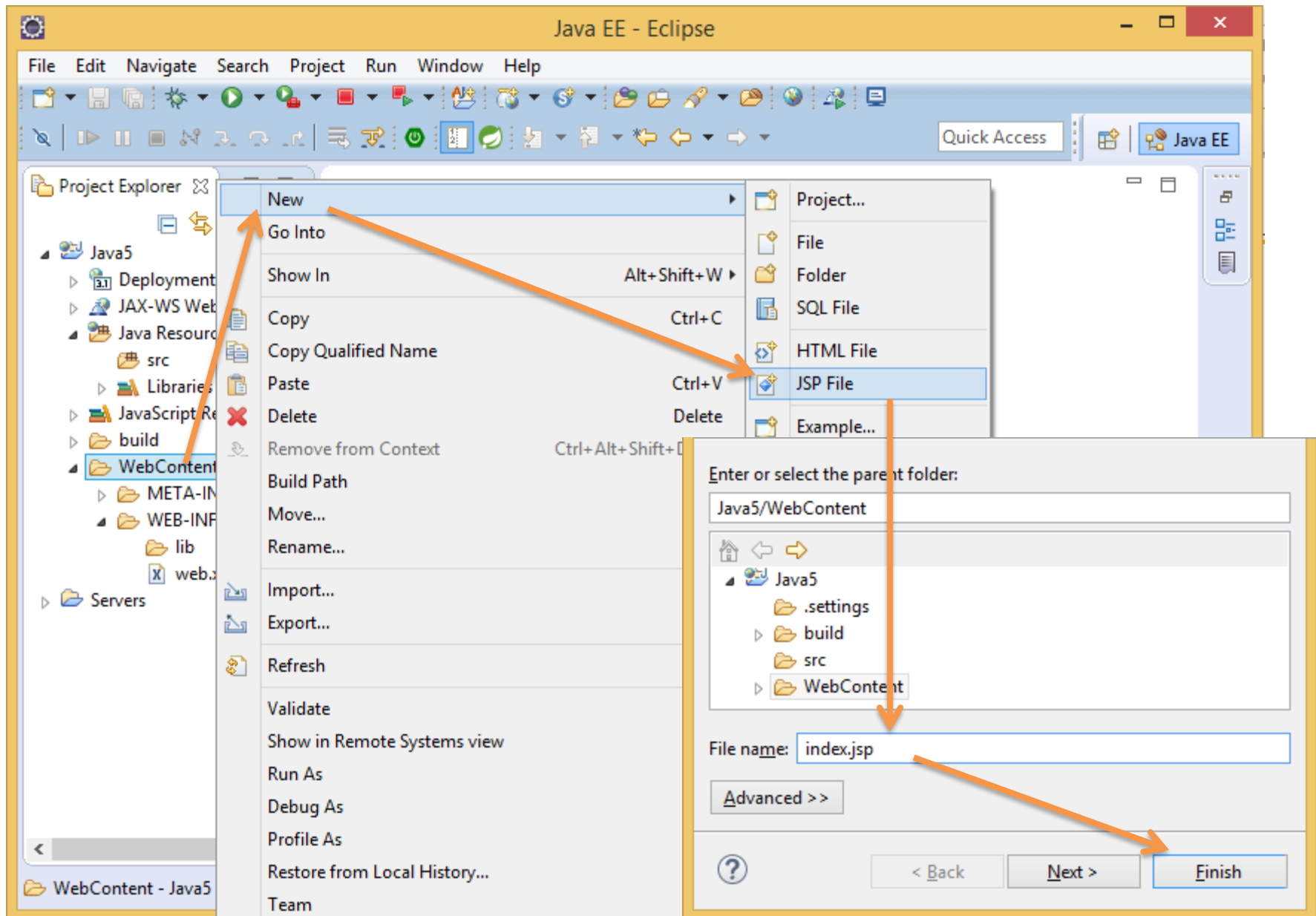


# CHỌN TRÌNH DUYỆT NGOÀI

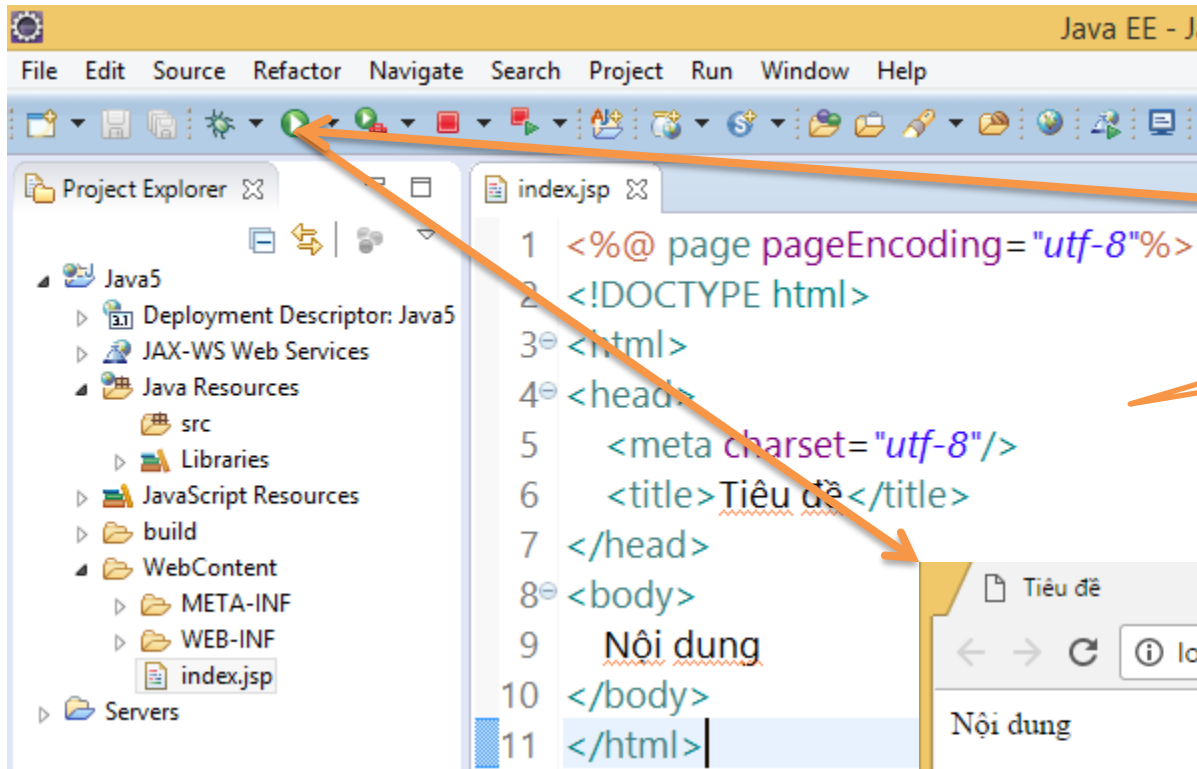


- ❑ Mặc định eclipse sử dụng trình duyệt nội bộ (không đủ mạnh để xử lý css và javascript)

# TẠO TRANG JSP

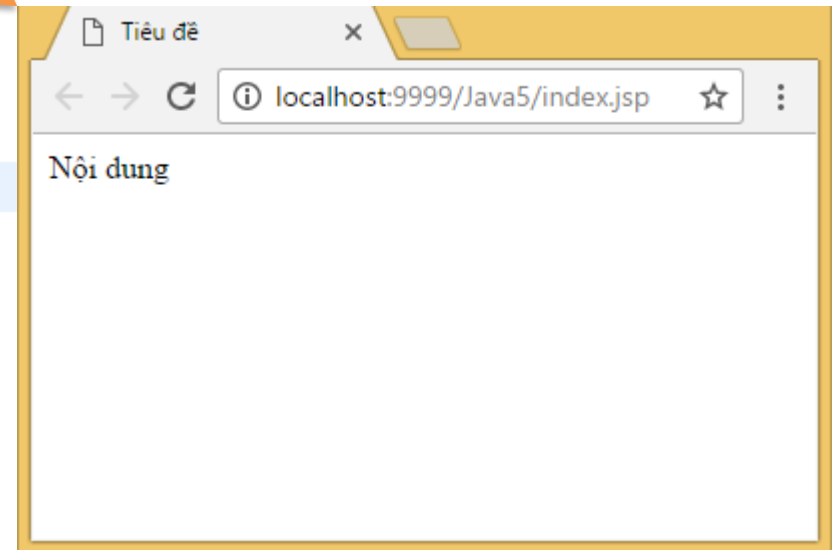


# CHẠY TRANG JSP



Hiệu chỉnh mã jsp theo  
chuẩn HTML5

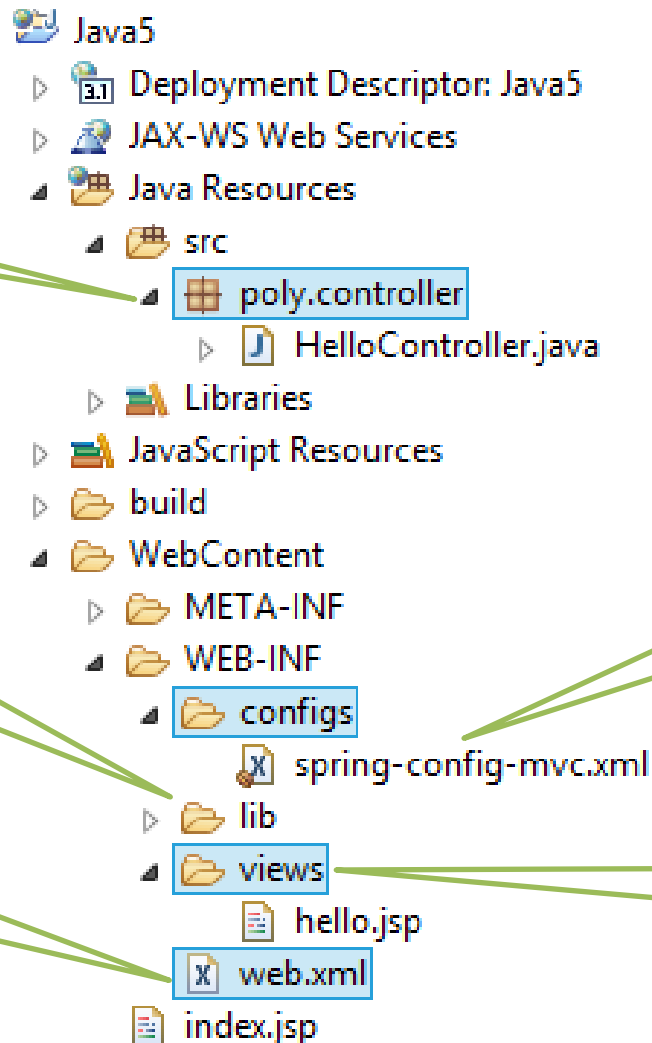
- ❑ Có thể chạy jsp bằng cách  
nhấp phải chuột lên trang  
jsp sau đó chọn  
**Run as > Run on server**



# DỰ ÁN SPRING MVC

- ❑ Để dự án hoạt động theo Spring MVC cần
  - ❖ Các thư viện liên quan (\*.jar)
  - ❖ Cấu hình đúng (\*.xml)
  - ❖ Viết mã theo đúng qui ước

# TỔ CHỨC DỰ ÁN SPRING MVC



Controller

Thư viện

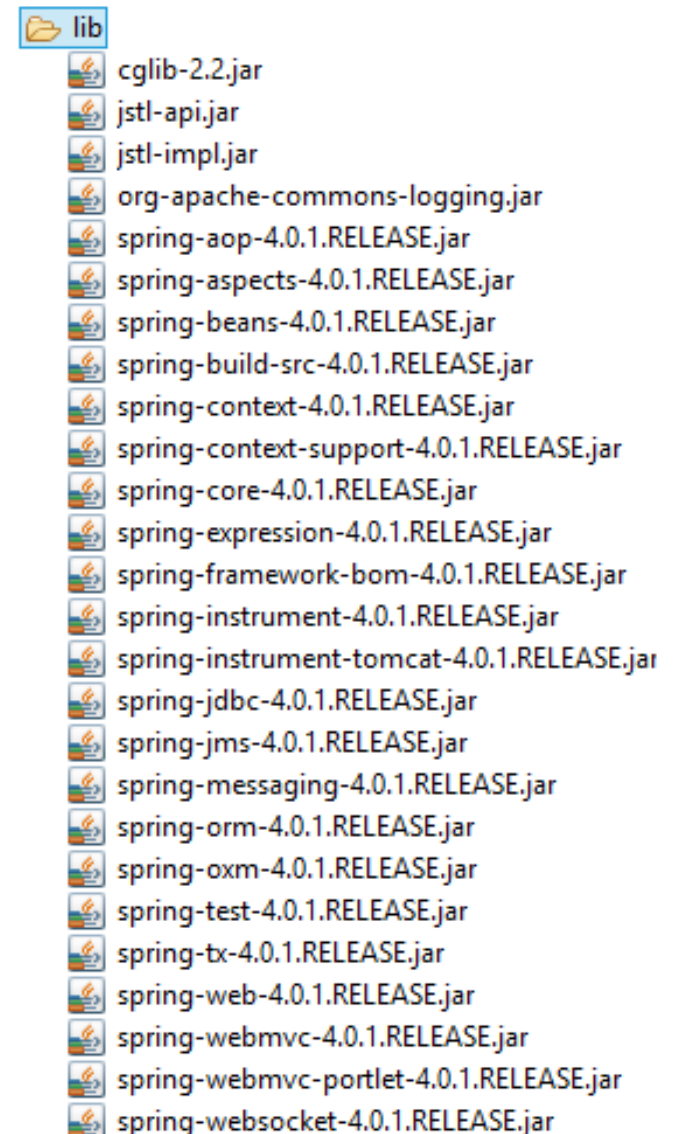
Cấu hình web

Cấu hình Spring MVC

View

# THƯ VIỆN SPRING MVC

❑ Thư viện cần thiết cho ứng dụng web nói chung và Spring MVC nói riêng phải được đặt trong thư mục **/WEB-INF/lib**



# CẤU HÌNH DỰ ÁN SPRING MVC

❑ **web.xml** là file cấu hình ứng dụng web

- ❖ Khai báo DispatcherServlet

  - Tiếp nhận và điều phối yêu cầu từ người dùng

- ❖ Khai báo CharacterEncodingFilter

  - Xử lý chế độ mã hóa ký tự

- ❖ Khai báo spring-config-mvc.xml

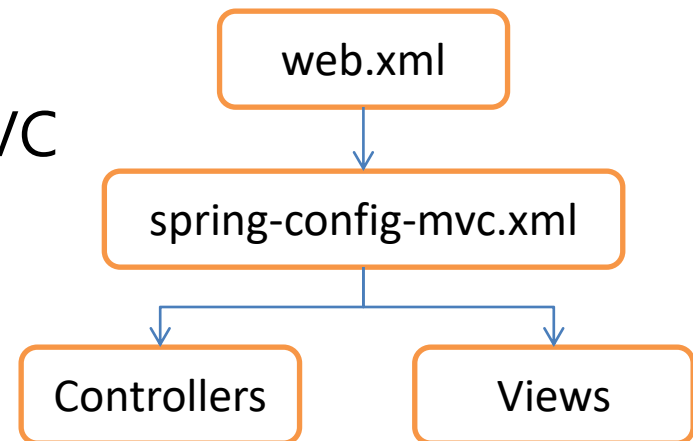
  - Cấu hình Spring MVC

❑ **spring-config-mvc.xml** là file cấu hình Spring MVC

- ❖ Cấu hình ứng dụng Spring MVC

- ❖ Khai báo Controller

- ❖ Khai báo ViewResolver





# CẤU HÌNH ỨNG DỤNG WEB

web.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app xmlns:xsi="http://www.w3.org/200
3    <display-name>Java5 </display-name>
4  <!-- DispatcherServlet -->
5  <servlet>..
14 <servlet-mapping>..
18
19 <!-- CharacterEncodingFilter -->
20 <filter>..
32 <filter-mapping>..
36
37 <welcome-file-list>
38   <welcome-file>index.jsp</welcome-file>
39 </welcome-file-list>
40 </web-app>
```

Khai báo  
**DispatcherServlet**

Khai báo  
**CharacterEncodingFilter**

# KHAI BÁO DISPATCHERSERVLET

```
<servlet>
  <servlet-name>spring</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/configs/*.xml</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>spring</servlet-name>
  <url-pattern>*.htm</url-pattern>
</servlet-mapping>
```

Sử dụng dấu \* để chỉ ra rằng tất cả các file xml đặt vào thư mục **/WEB-INF/configs** đều được xem như là file cấu hình Spring và được nạp vào ứng dụng

Tất cả các URL kết thúc bởi **.htm** đều được DispatcherServlet tiếp nhận và xử lý

# KHAI BÁO CHARACTERENCODINGFILTER

```
<filter>
  <filter-name>utf8</filter-name>
  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
  <init-param>
    <param-name>encoding</param-name>
    <param-value>UTF-8</param-value>
  </init-param>
  <init-param>
    <param-name>forceEncoding</param-name>
    <param-value>true</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>utf8</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

CharacterEncodingFilter cho phép ứng dụng web làm việc với utf-8 (tiếng Việt)

# CẤU TRÚC FILE CẤU HÌNH SPRING

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

```
<beans
```

```
xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:p="http://www.springframework.org/schema/p"
xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:mvc="http://www.springframework.org/schema/mvc"
xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context.xsd
    http://www.springframework.org/schema/tx
    http://www.springframework.org/schema/tx/spring-tx.xsd
    http://www.springframework.org/schema/mvc
    http://www.springframework.org/schema/mvc/spring-mvc.xsd">
```

```
<!-- Nội dung khai báo cấu hình Spring -->
```

```
</beans>
```

Các namespace và schema qui định cú pháp thẻ trong file cấu hình

# SPRING-CONFIG-MVC.XML

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

```
<beans ...>
```

```
<!-- Cấu hình Spring MVC Annotation -->
```

```
<context:annotation-config />
```

```
<mvc:annotation-driven/>
```

Cho phép sử dụng Annotation trong ứng dụng Spring

View = prefix + viewname + suffix

```
<!-- Cấu hình ViewResolver -->
```

```
<bean id="viewResolver"
```

```
    p:prefix="/WEB-INF/views/" p:suffix=".jsp"
```

```
    class="org.springframework.web.servlet.view.InternalResourceViewResolver"/>
```

```
<!-- Cấu hình package chứa các controller -->
```

```
<context:component-scan base-package="poly.controller"/>
```

```
</beans>
```

Chỉ rõ gói chứa các Controller. Sử dụng dấu phẩy để phân cách các gói

# HELLOCONTROLLER

Chú thích lớp  
Controller

```
package poly.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;

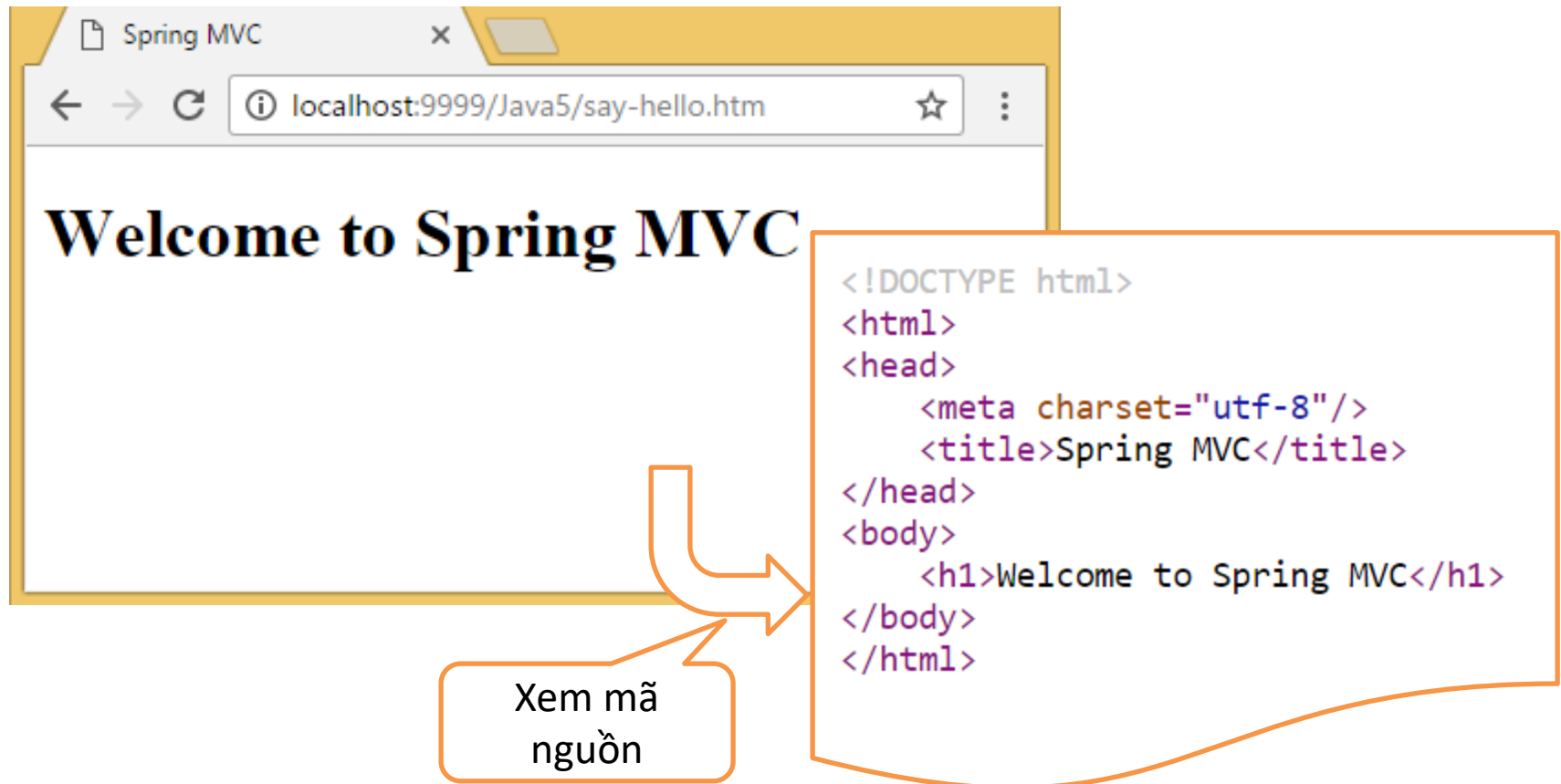
@Controller
public class HelloController {
    @RequestMapping("say-hello")
    public String sayHello() {
        return "hello";
    }
}
```

Tên giao dịch

Tên view

```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <title>Spring MVC</title>
</head>
<body>
    <h1>Welcome to Spring MVC</h1>
</body>
</html>
```

- ❑ Chạy index.jsp sau đó nhập lại url như sau
  - ❖ <http://localhost:9999/Java5/say-hello.htm>
- ❑ Sau đây là kết quả phản hồi



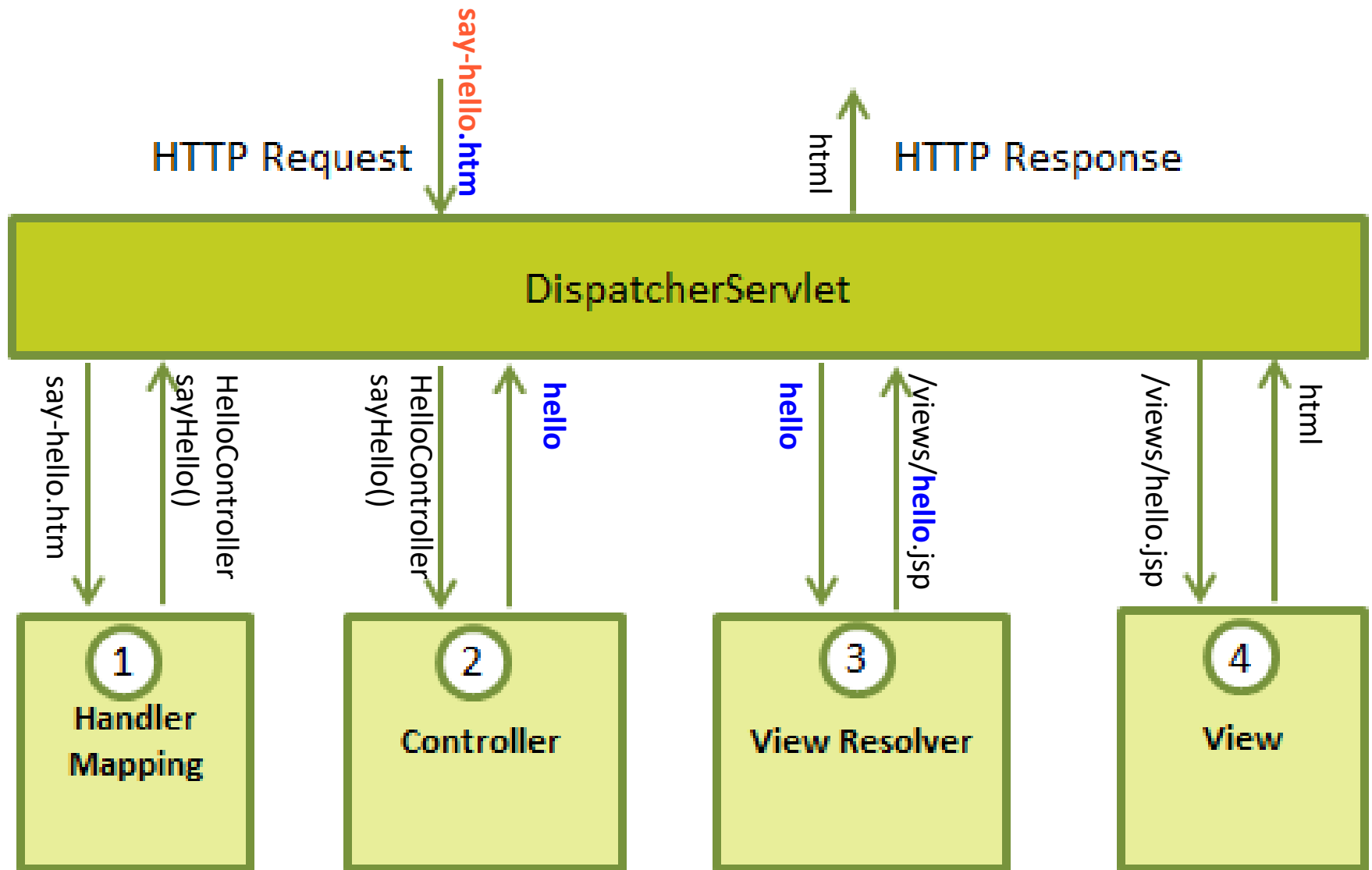
The image shows a web browser window with the title 'Spring MVC' and the address bar displaying 'localhost:9999/Java5/say-hello.htm'. The main content of the page is 'Welcome to Spring MVC'. An orange arrow points from the text 'Xem mã nguồn' (View source code) to a box containing the HTML source code of the page.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"/>
  <title>Spring MVC</title>
</head>
<body>
  <h1>Welcome to Spring MVC</h1>
</body>
</html>
```

Xem mã nguồn



# QUI TRÌNH XỬ LÝ SAY-HELLO.HTM



# LƯU Ý VIEWRESOLVER

```
@Controller  
public class HelloController {  
    @RequestMapping("say-hello")  
    public String sayHello() {  
        return "hello";  
    }  
}
```

```
<bean id="viewResolver"  
    p:prefix="/WEB-INF/views/" p:suffix=".jsp"  
    class="org.springframework.web.servlet.view.InternalResourceViewResolver"/>
```

prefix + view + suffix  
/WEB-INF/views/hello.jsp

## QUI TRÌNH XỬ LÝ SAY-HELLO.HTM

❑ DispatcherServlet sẽ nhận request với URL kết thúc .htm

1. Chuyển **say-hello**.htm cho Handler Mapping và sẽ nhận được **sayHello()** của **HelloController** (do phương thức này được map với tên say-hello)
2. Gọi sayHello() của HelloController và nhận được **"hello"** (do phương thức này return "hello")
3. Chuyển **"hello"** cho ViewResolver và nhận được **"/WEB-INF/views/hello.jsp"** (do ghép nối prefix + hello + suffix)
4. Gọi hello.jsp và nhận kết quả HTML sau cùng là phản hồi cho người dùng

## PHẦN 2

□ Trong lập trình Servlet/JSP chúng ta đã được làm việc với các thành phần web sau

❖ **HttpServletRequest**

- Gói dữ liệu gửi từ client và chia sẻ cho nhiều Servlet/JSP hoạt động trên một request

❖ **HttpServletResponse**

- Gói dữ liệu chuyển về client

❖ **HttpSession**

- Phạm vi chia sẻ dữ liệu theo từng phiên làm việc khác nhau

❖ **ServletContext**

- Phạm vi chia sẻ dữ liệu trên toàn ứng dụng

# LÀM VIỆC VỚI CÁC ĐỐI TƯỢNG WEB TRONG SPRING MVC

- ❑ Trong Spring MVC bạn có thể truy xuất các đối tượng web một cách dễ dàng bằng cách định nghĩa chúng như những đối số của action method hoặc sử dụng @Autowire.

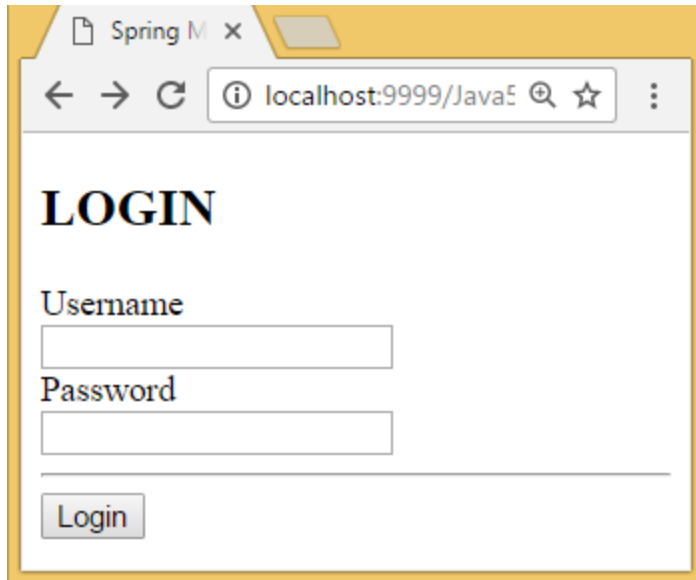
Đối với ServletContext bạn sử dụng @Autowired để tham chiếu đến

Khi bạn muốn làm việc với đối tượng nào bạn chỉ việc khai báo đối tượng đó như đối số của action method.

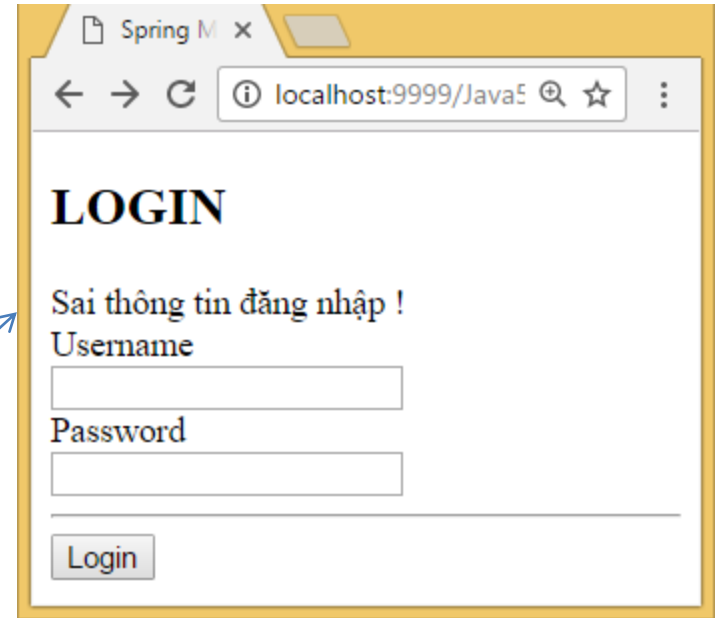
```
@Controller
public class UserController {
    @Autowired
    ServletContext application;

    @RequestMapping("say-hello")
    public String sayHello(
        HttpServletRequest request,
        HttpServletResponse response,
        HttpSession session) {
        System.out.println("index");
        return "user";
    }
}
```

# TÌNH HUỐNG ĐĂNG NHẬP



A screenshot of a web browser window with the title "Spring M x". The address bar shows "localhost:9999/Java5". The page content includes a "LOGIN" heading, "Username" and "Password" labels with corresponding input fields, and a "Login" button at the bottom.



A screenshot of a web browser window with the title "Spring M x". The address bar shows "localhost:9999/Java5". The page content includes a "LOGIN" heading, an error message "Sai thông tin đăng nhập !", "Username" and "Password" labels with corresponding input fields, and a "Login" button at the bottom. Two blue arrows point from the left screenshot to this one, indicating a transition or comparison.

# XÂY DỰNG USERCONTROLLER

@Controller

**public class** UserController{

@RequestMapping("/user/form")

**public** String showForm() {

**return** "user/login";

}

@RequestMapping("/user/login")

**public** String login(HttpServletRequest request) {

String id = request.getParameter("id");

String pw = request.getParameter("password");

**if**(id.equals("fpt") && pw.equals("polytechnic")){

request.setAttribute("uid", id);

request.setAttribute("pwd", pw);

**return** "user/info";

}

request.setAttribute("message", "Sai thông tin đăng nhập!");

**return** "user/login";

}

}

View này chứa form

Sử dụng request để nhận tham số và chia sẻ dữ liệu

Nhận tham số

Chia sẻ dữ liệu

View này hiển thị thông tin user



# XÂY DỰNG CÁC VIEW

Hiển thị dữ liệu  
truyền từ action

```
<h2>LOGIN</h2>
${message}
<form action="user/login.htm" method="post">
  <div>Username</div>
  <input name="id"/>

  <div>Password</div>
  <input name="password"/>

  <hr>
  <button>Login</button>
</form>
```

**user/login.jsp**

Hiển thị dữ liệu  
truyền từ action

```
<h3>USER INFO</h3>
<ul>
  <li>User Name: ${uid}</li>
  <li>Password: ${pwd}</li>
</ul>
```

**user/info.jsp**

# TRUYỀN DỮ LIỆU TỪ CONTROLLER SANG VIEW

- ❑ Bạn có thể sử dụng `request.setAttribute(name, value)` để truyền dữ liệu cho View
- ❑ Trong Spring MVC bạn có phương án khác chuẩn tắc hơn là sử dụng `ModelMap` làm đối số action method thay vì sử dụng `HttpServletRequest`

```
@RequestMapping("say-hello1")
public String sayHello1(HttpServletRequest request) {
    request.setAttribute("name", "Nguyễn Văn Tèo");
    return "hello";
}
```

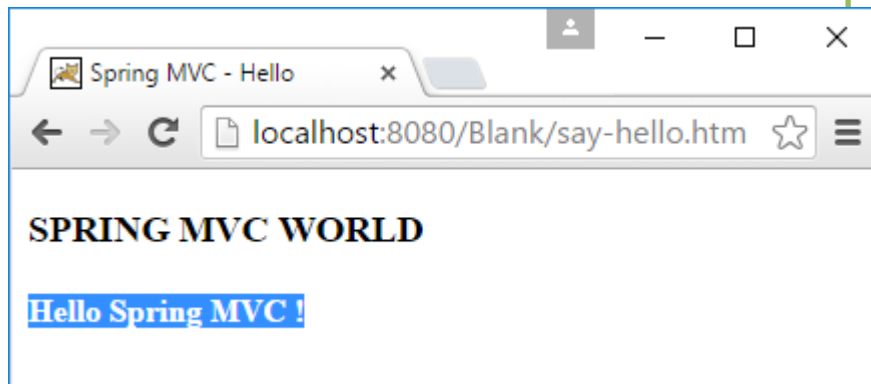
```
@RequestMapping("say-hello2")
public String sayHello1(ModelMap model) {
    model.addAttribute("name", "Nguyễn Văn Tèo");
    return "hello";
}
```

Trong JSP bạn có thể sử dụng `<%=request.getAttribute("name")%>` để truy xuất hoặc có thể sử dụng biểu thức EL `${name}` để truy xuất

# TRUYỀN DỮ LIỆU CHO VIEW

```
@Controller
public class HelloController {
    @RequestMapping(value="/say-hello")
    public String sayHello(ModelMap model) {
        model.addAttribute("message", "Hello Spring MVC !");
        return "hello";
    }
}
```

```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Spring MVC - Hello</title>
</head>
<body>
    <h3>SPRING MVC WORLD</h3>
    <h4>${message}</h4>
</body>
</html>
```



- ❑ Hiệu chỉnh action login theo hướng dẫn sau
  - ❖ Thêm đối số ModelMap model
  - ❖ Thay request.setAttribute() bằng model.addAttribute()

```
@RequestMapping("user/login")
public String login(ModelMap model, HttpServletRequest request) {
    String id = request.getParameter("id");
    String pw = request.getParameter("password");
    if(id.equals("fpt") && pw.equals("polytechnic")){
        model.addAttribute("uid", id);
        model.addAttribute("pwd", pw);
        return "user/info";
    }
    model.addAttribute("message", "Sai thông tin đăng nhập!");
    return "user/login";
}
```

# TỔNG KẾT NỘI DUNG BÀI HỌC

- ☑ Giới thiệu Spring Framework
- ☑ Xử lý request trong Spring MVC
- ☑ Thiết lập hệ thống phát triển ứng dụng web
- ☑ Tích hợp tomcat vào eclipse IDE
- ☑ Tạo dự án web
- ☑ Dự án Spring MVC
- ☑ Cấu hình ứng dụng
- ☑ Tạo Controller
- ☑ Tạo JSP
- ☑ Làm việc với các đối tượng web
- ☑ Truyền dữ liệu từ Controller sang View