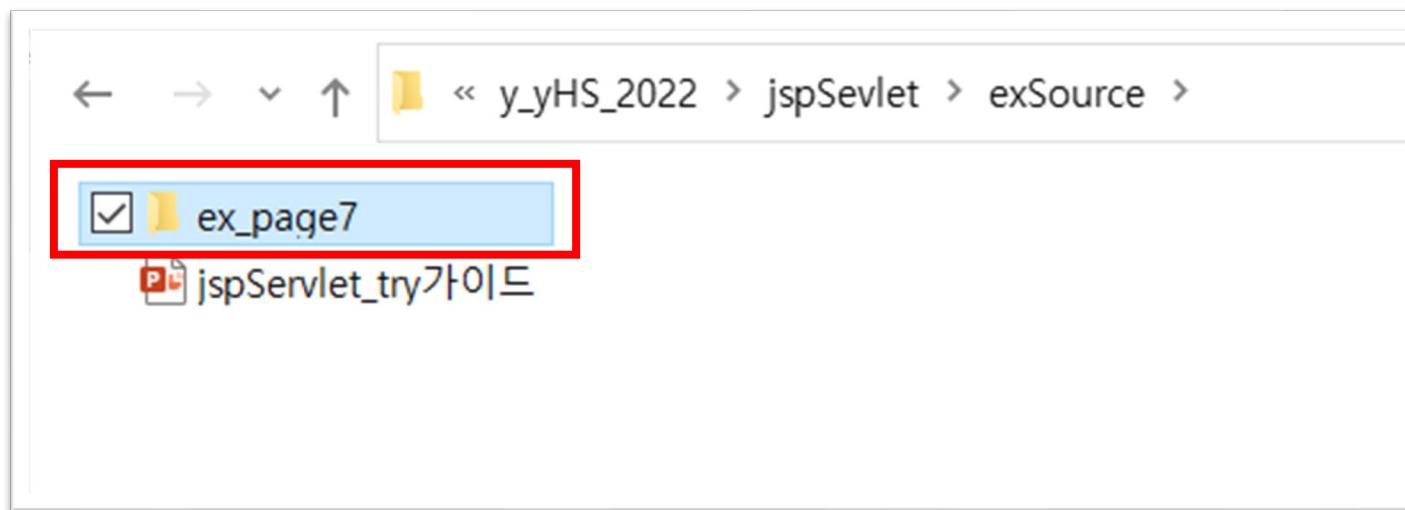


# Try~ Java Web

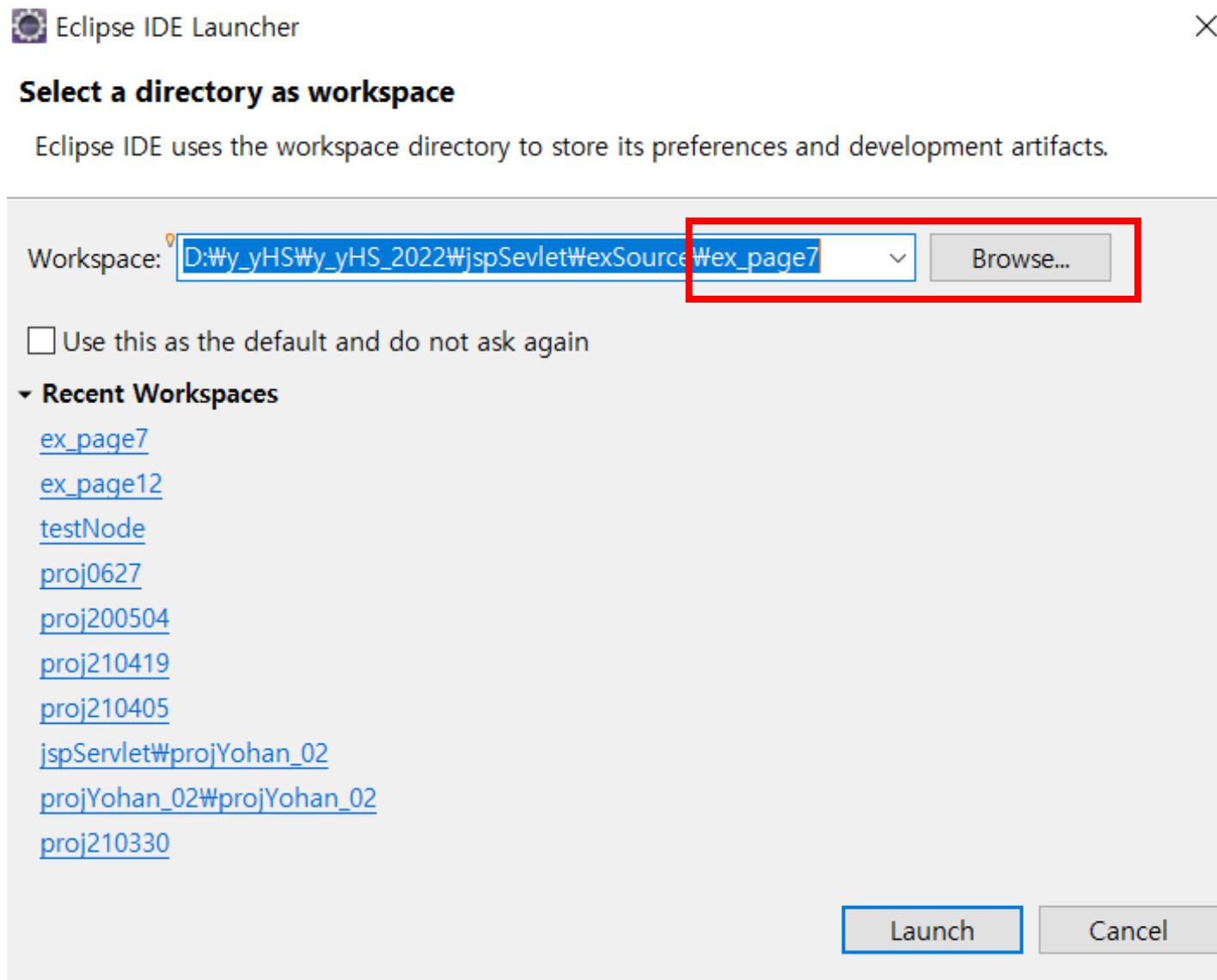
- A. 프로젝트 폴더 설정
- B. 이클립스에 WAS설정
- B\_2. Tomcat 포트번호 변경
- C. Dynamic프로젝트 시작
- C\_2. 이클립스 한글설정\_UTF-8
- C\_3. jsp페이지 작성 및 실행
- C\_4. sevlet페이지 작성
- C\_5. xml 매팅과 실행

교사 최선경

# A. 프로젝트 폴더 생성



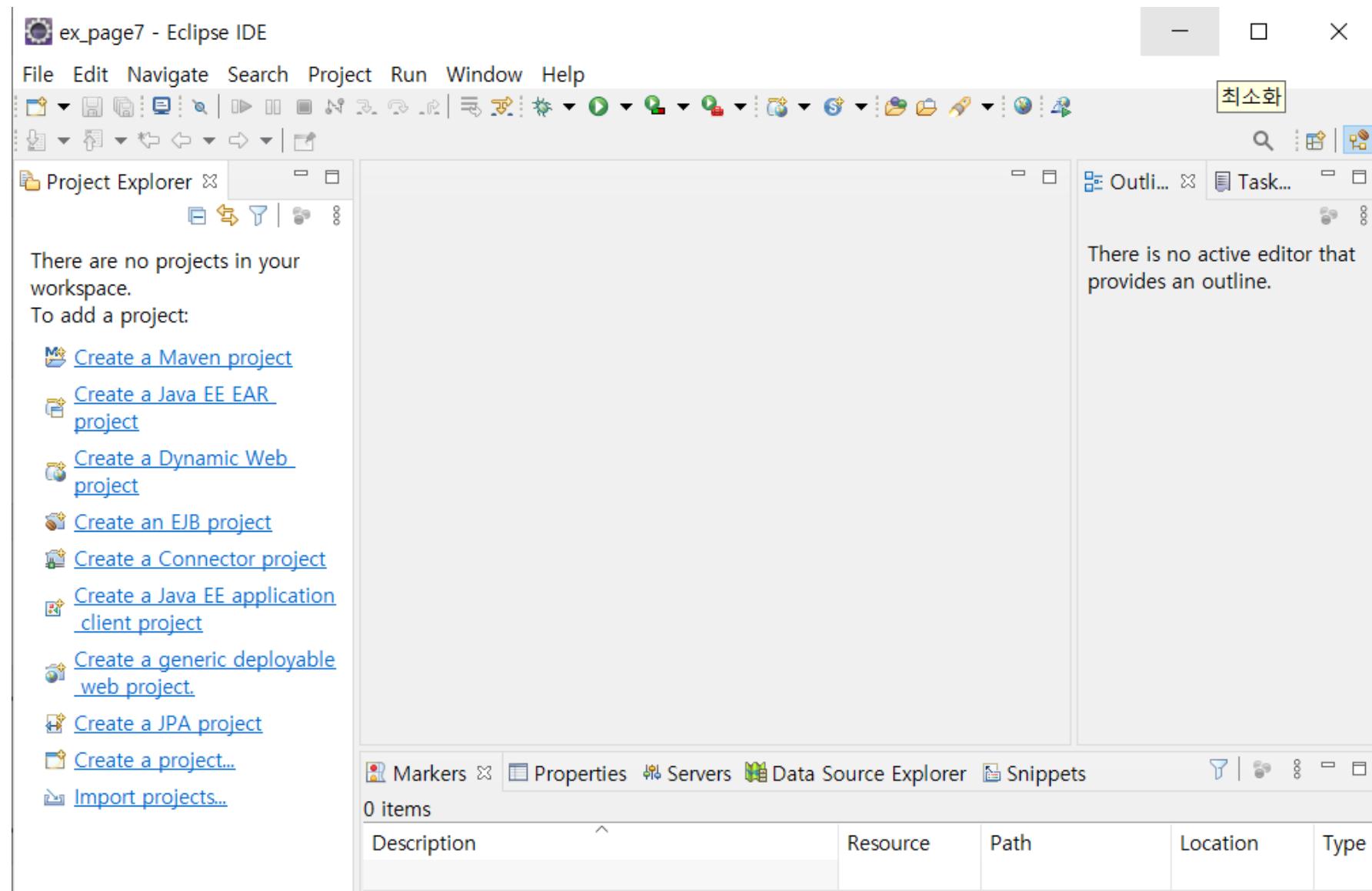
# 경로에 맞춰서 이클립스 실행



프로젝트폴더안의 환경파일을 확인한다.

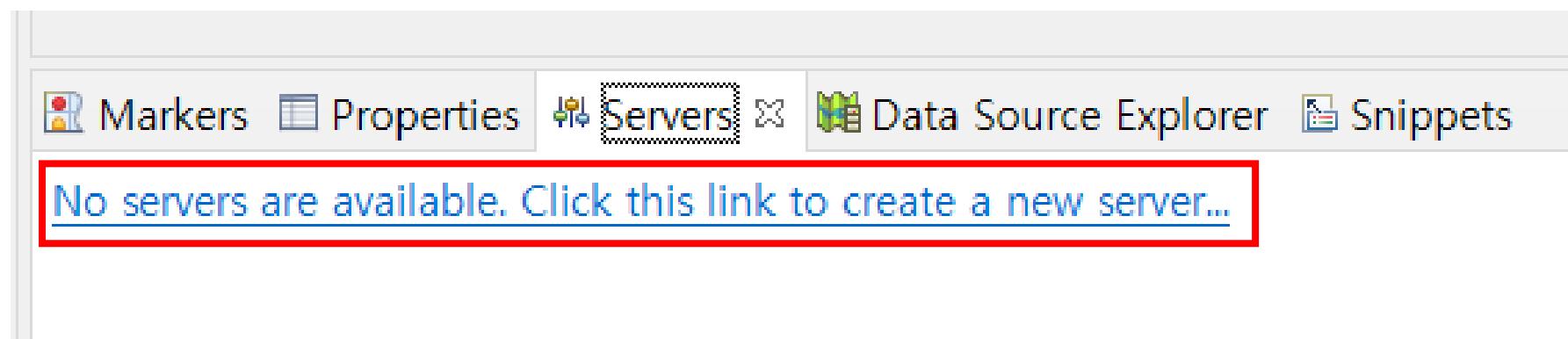


# 이클립스 OPEN.



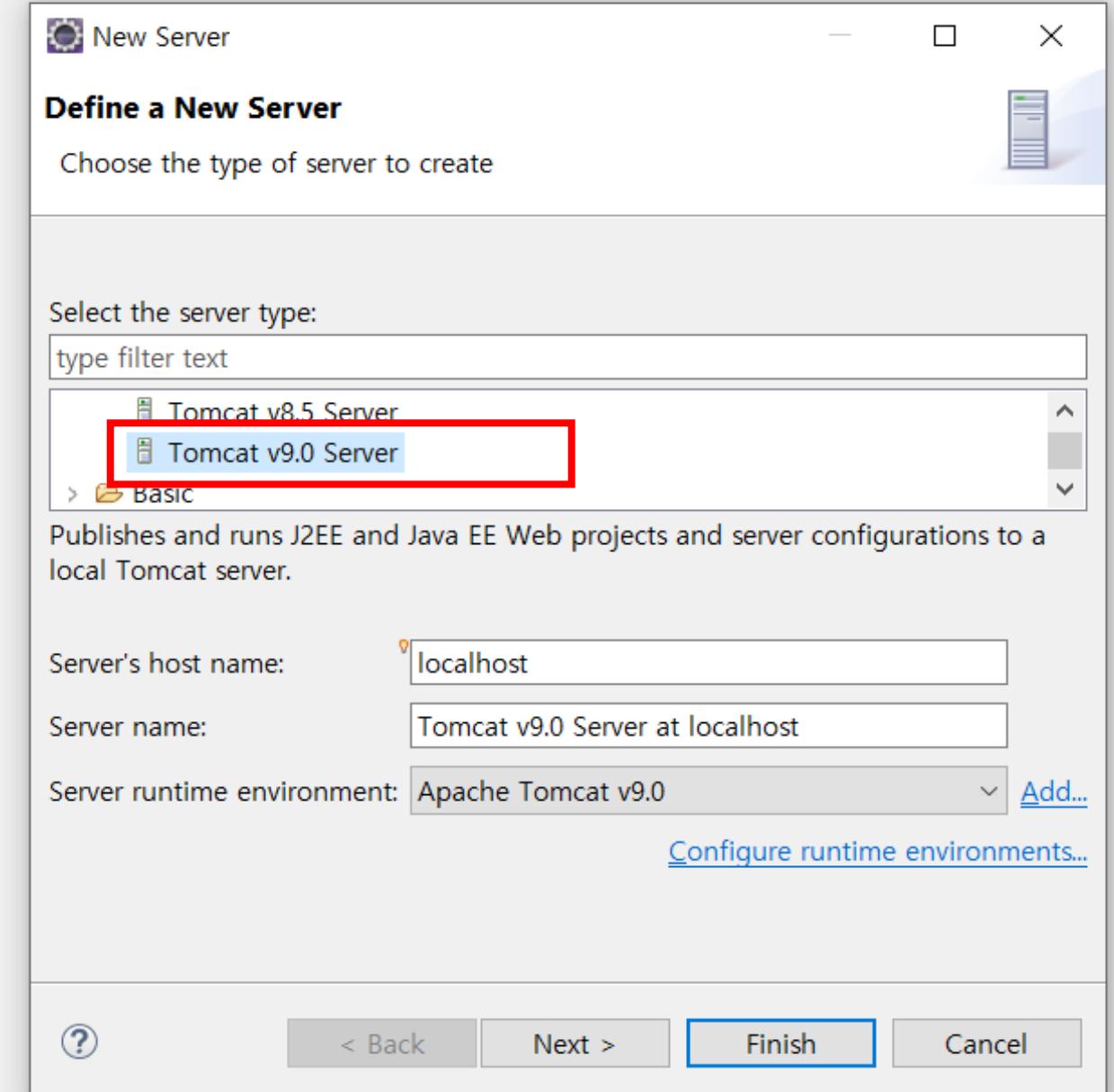
## B. 자바웹을 개발하기 위한 WAS설정.

1. WebApplicationServer 설치 \_ ex. Tomcat9
2. 이클립스에 Tomcat을 서버로 연결



# 이클립스

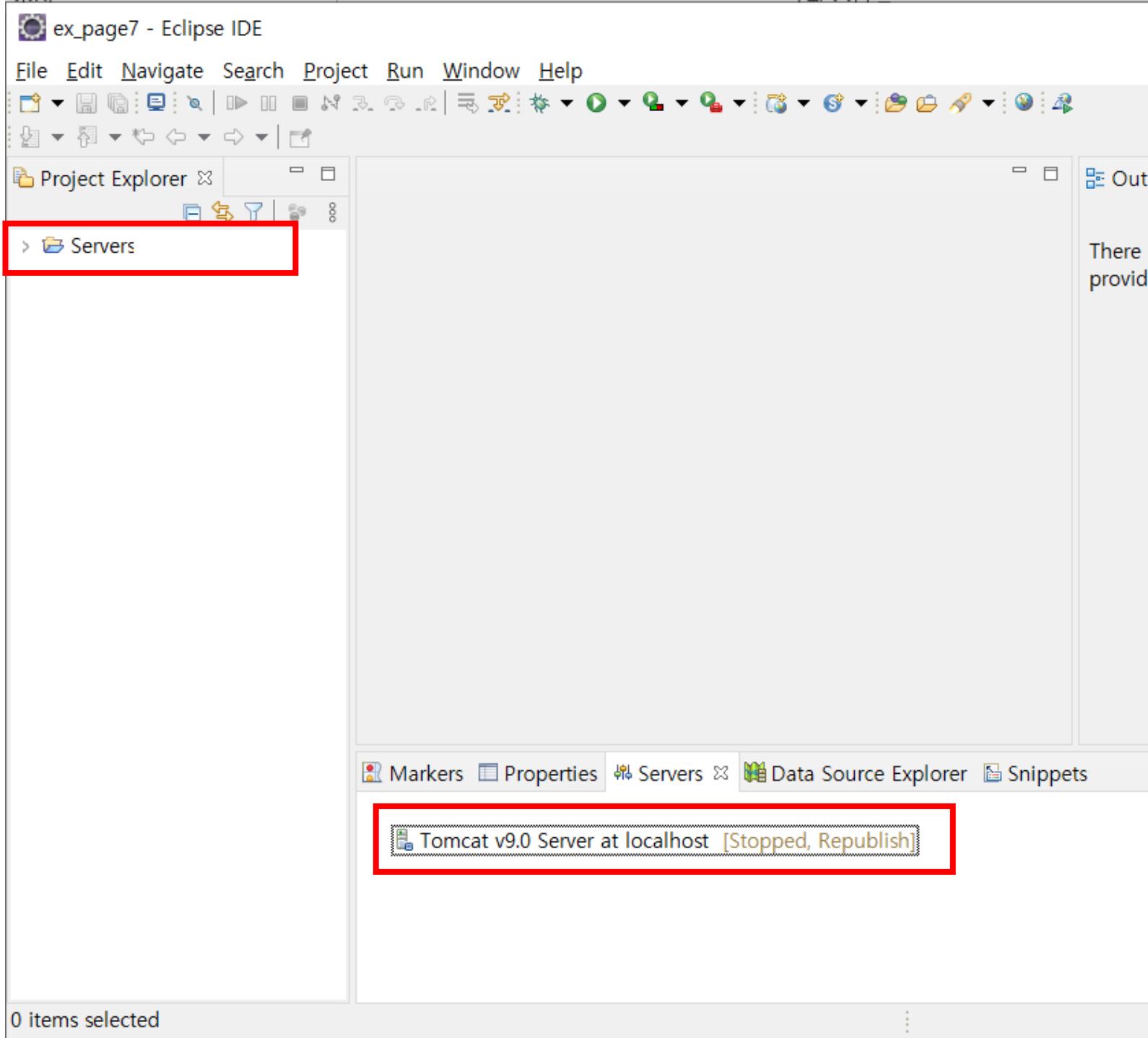
## 하단 Servers



Markers Properties Servers Data Source Explorer Snippets

No servers are available. Click this link to create a new server...

# 이클립스에 서버설정 확인



# B\_2. Tomcat 포트 Port 번호변경\_충돌방지

\*Tomcat v9.0 Server at localhost

## Overview

**General Information**  
Specify the host name and other common settings.

Server name: Tomcat v9.0 Server at localhost

Host name: localhost

Runtime Environment: Apache Tomcat v9.0

Configuration path: /Servers/Tomcat v9.0 Server at localhost-config | Browse...

[Open launch configuration](#)

**Server Locations**  
Specify the server path (i.e. catalina.base) and deploy path. Server must be published with no modules present to make changes.

Use workspace metadata (does not modify Tomcat installation)

Use Tomcat installation (takes control of Tomcat installation)

Use custom location (does not modify Tomcat installation)

Server path: .metadata\plugins\org.eclipse.wst.server.core\tmp0 | Browse...

[Set deploy path to the default value \(currently set\)](#)

Deploy path: wtpwebapps | Browse...

**Publishing**

**Timeouts**

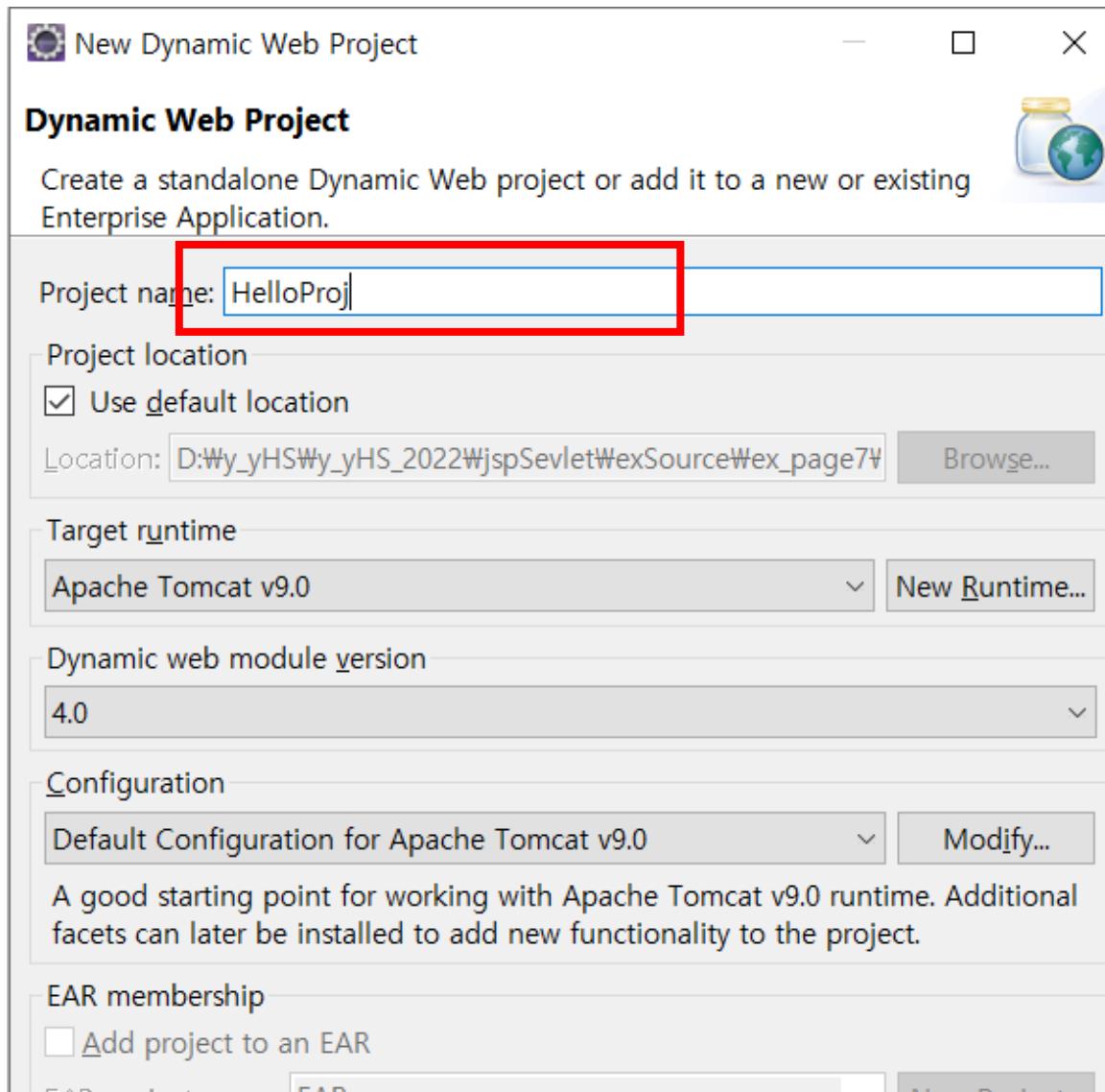
**Ports**  
Modify the server ports.

Port Name	Port Number
Tomcat admin port	9005
HTTP/1.1	8090

**MIME Mappings**

Enter settings for the server.

# Dynamic Web Project 생성



# Next버튼-> Web Module에서 체크

New Dynamic Web Project

**Dynamic Web Project**

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: HelloProj

Project location

Use default location

Location: D:\y\_yHS\y\_yHS\_2022\jspSevlet\exSource\ex\_page7\HelloProj

Target runtime

Apache Tomcat v9.0

Dynamic web module version

4.0

Configuration

Default Configuration for Apache Tomcat v9.0

A good starting point for working with Apache Tomcat v9.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

Add project to an EAR

EAR project name: HelloProjEAR

Working sets

Add project to working sets

New Dynamic Web Project

**Web Module**

Configure web module settings.

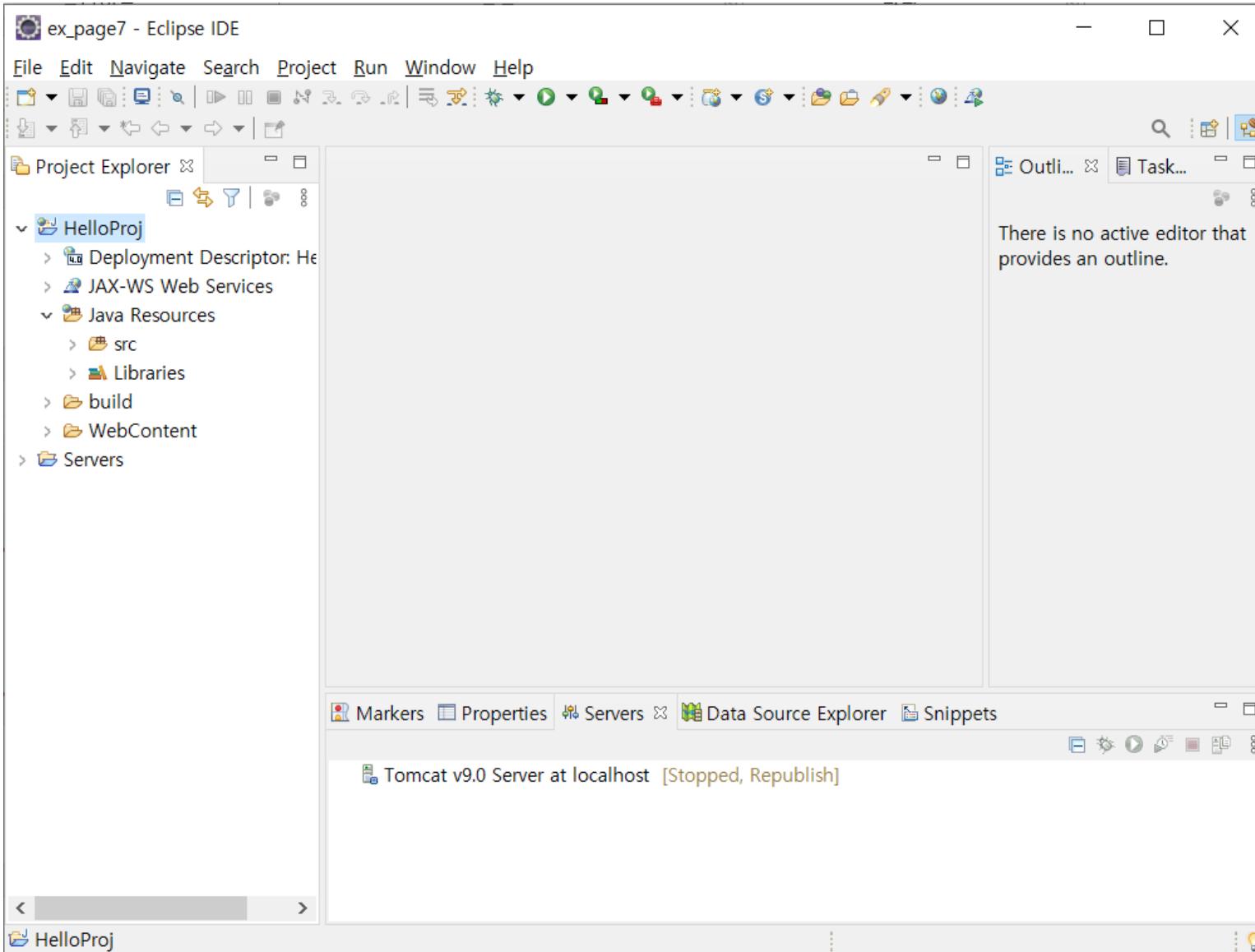
Context root: HelloProj

Content directory: WebContent

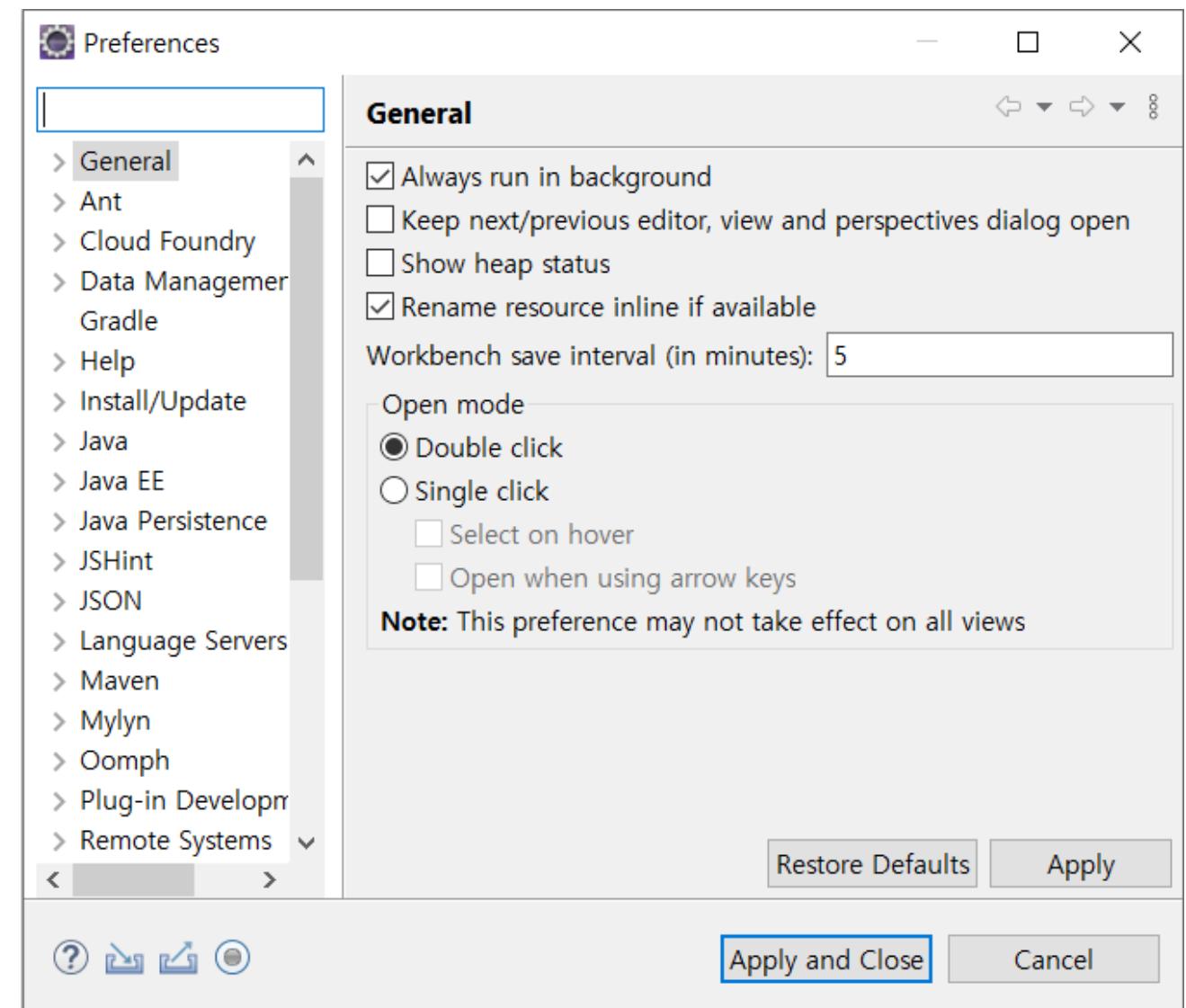
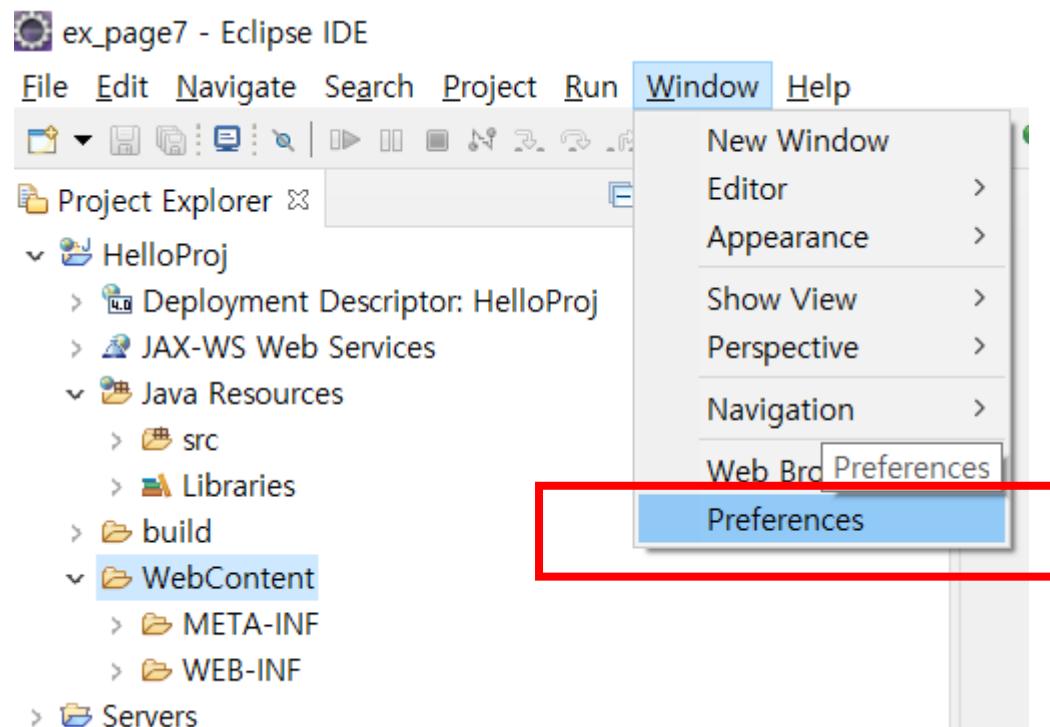
Generate web.xml deployment descriptor

Xml 환경파일을 자동으로 만들어줌

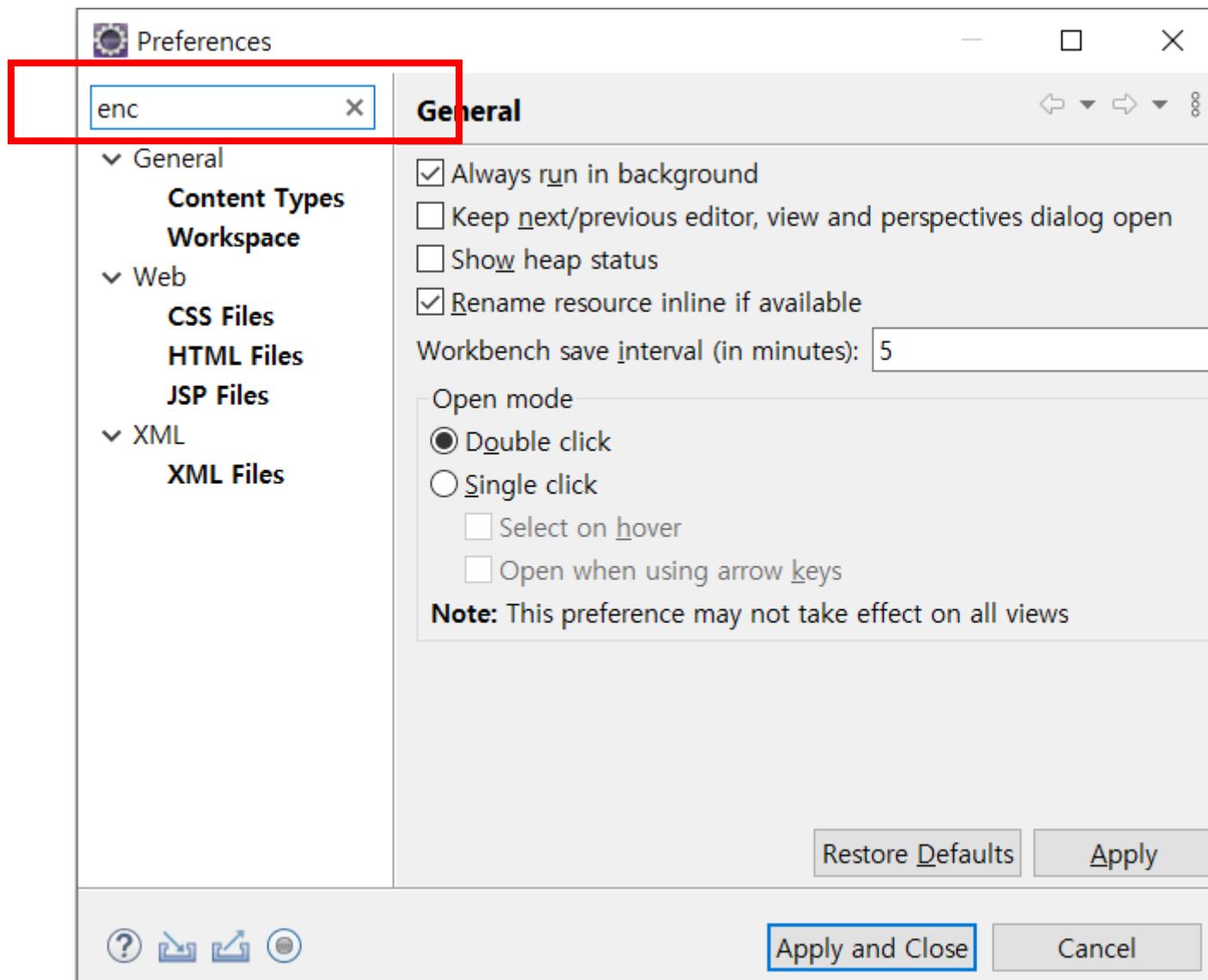
# C. Dynamic 프로젝트 시작.



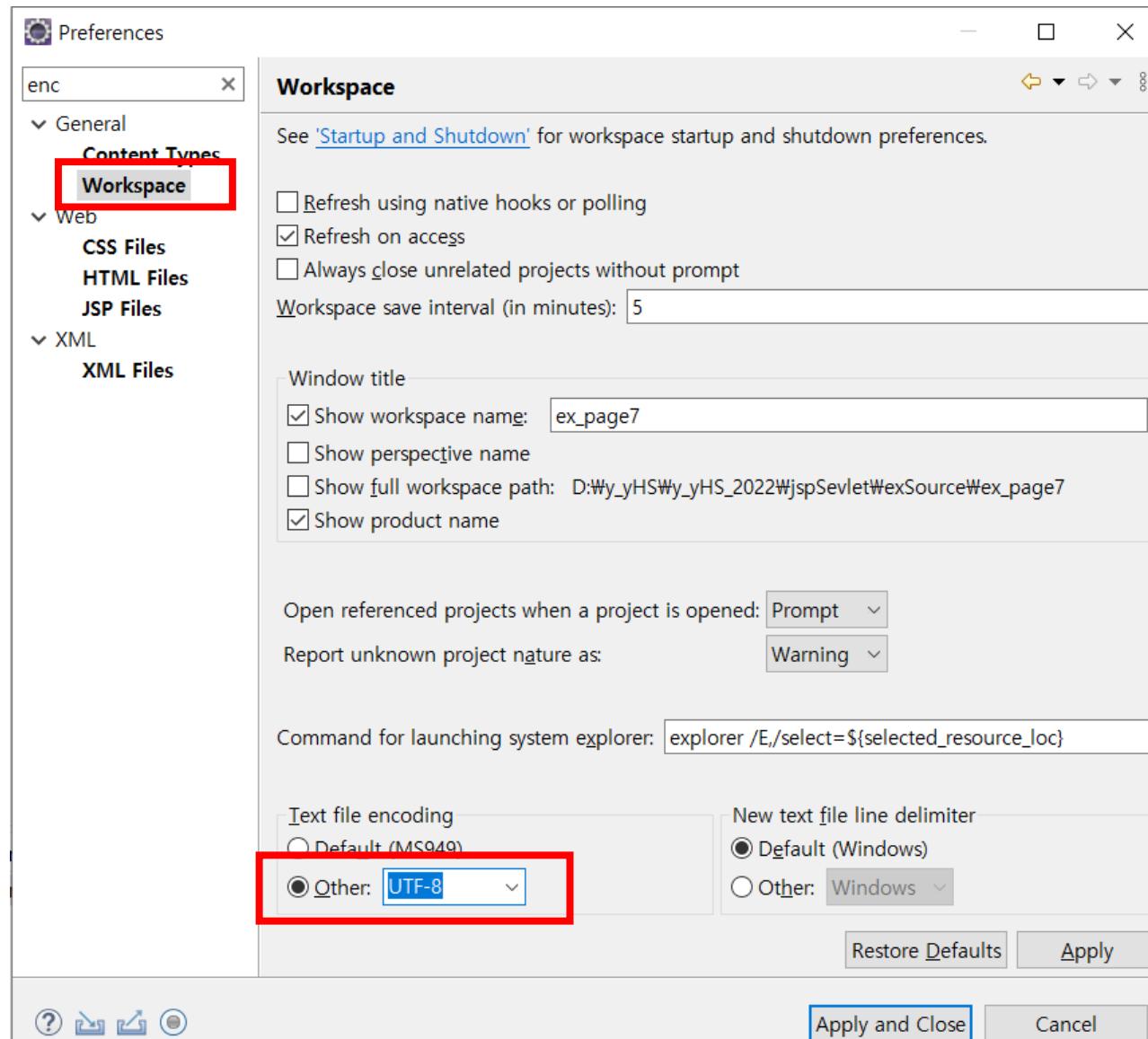
# C\_2. 이클립스 한글설정



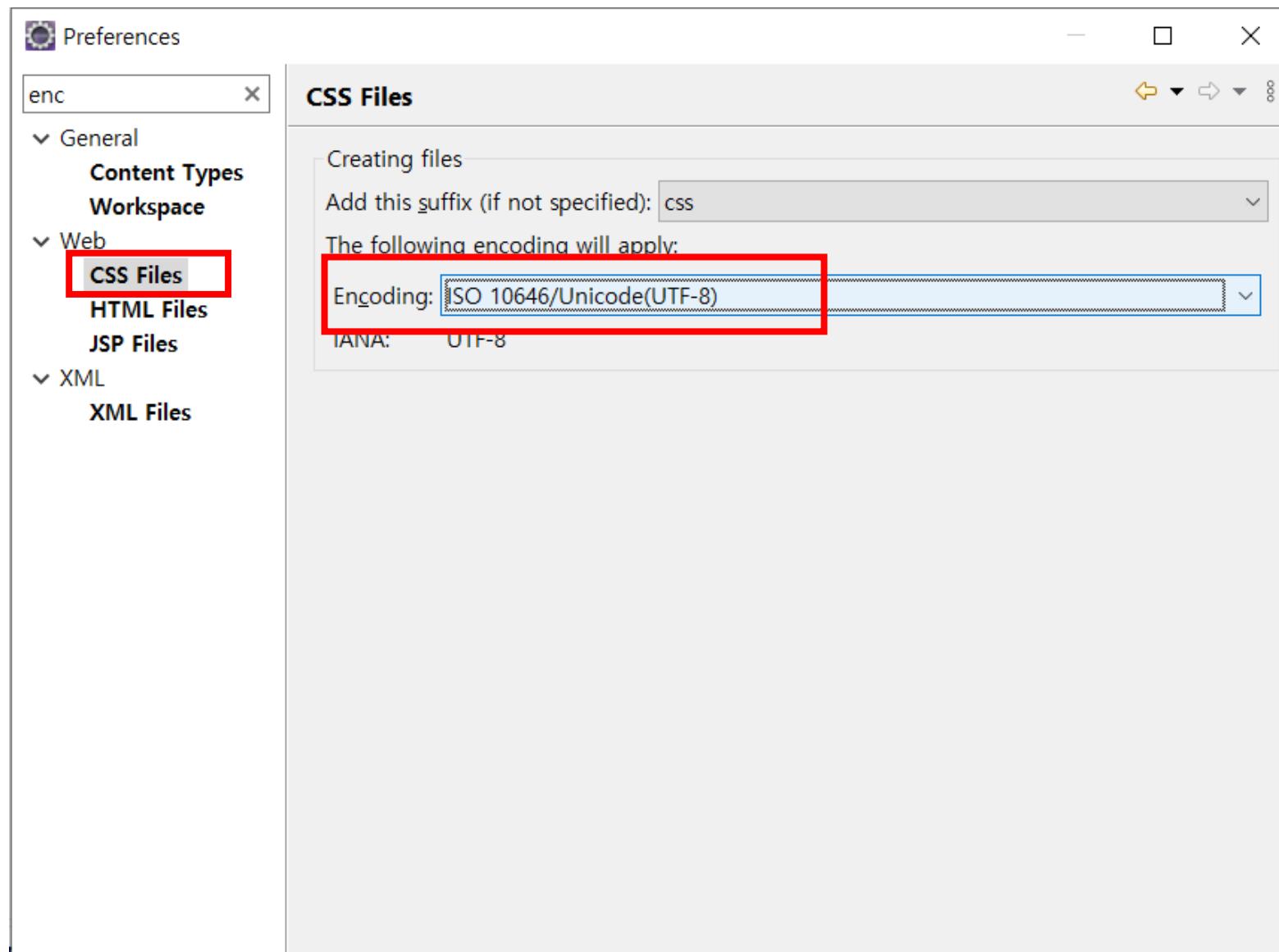
# 검색란에 enc 입력



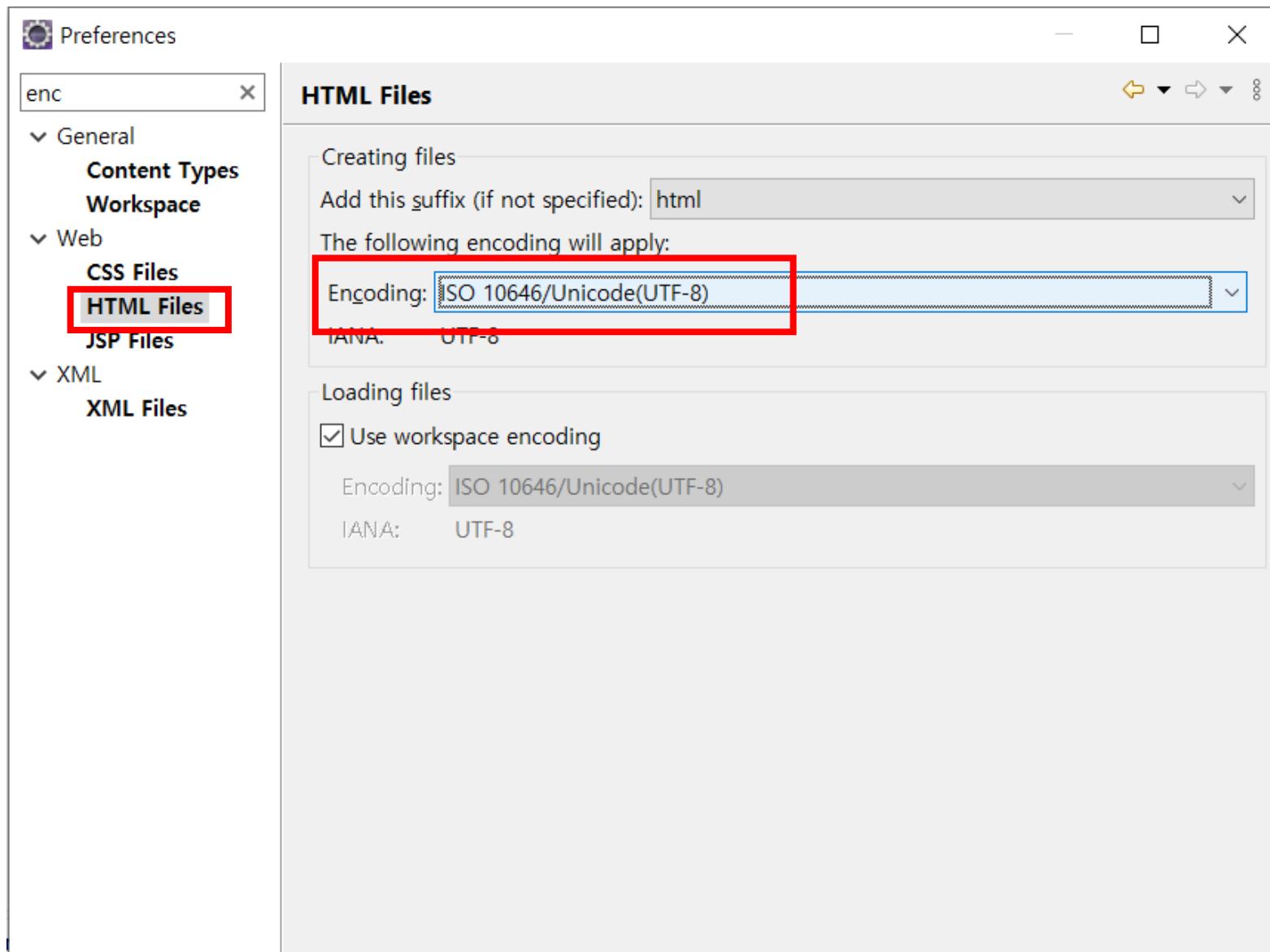
# 코딩영역 한글깨짐방지.



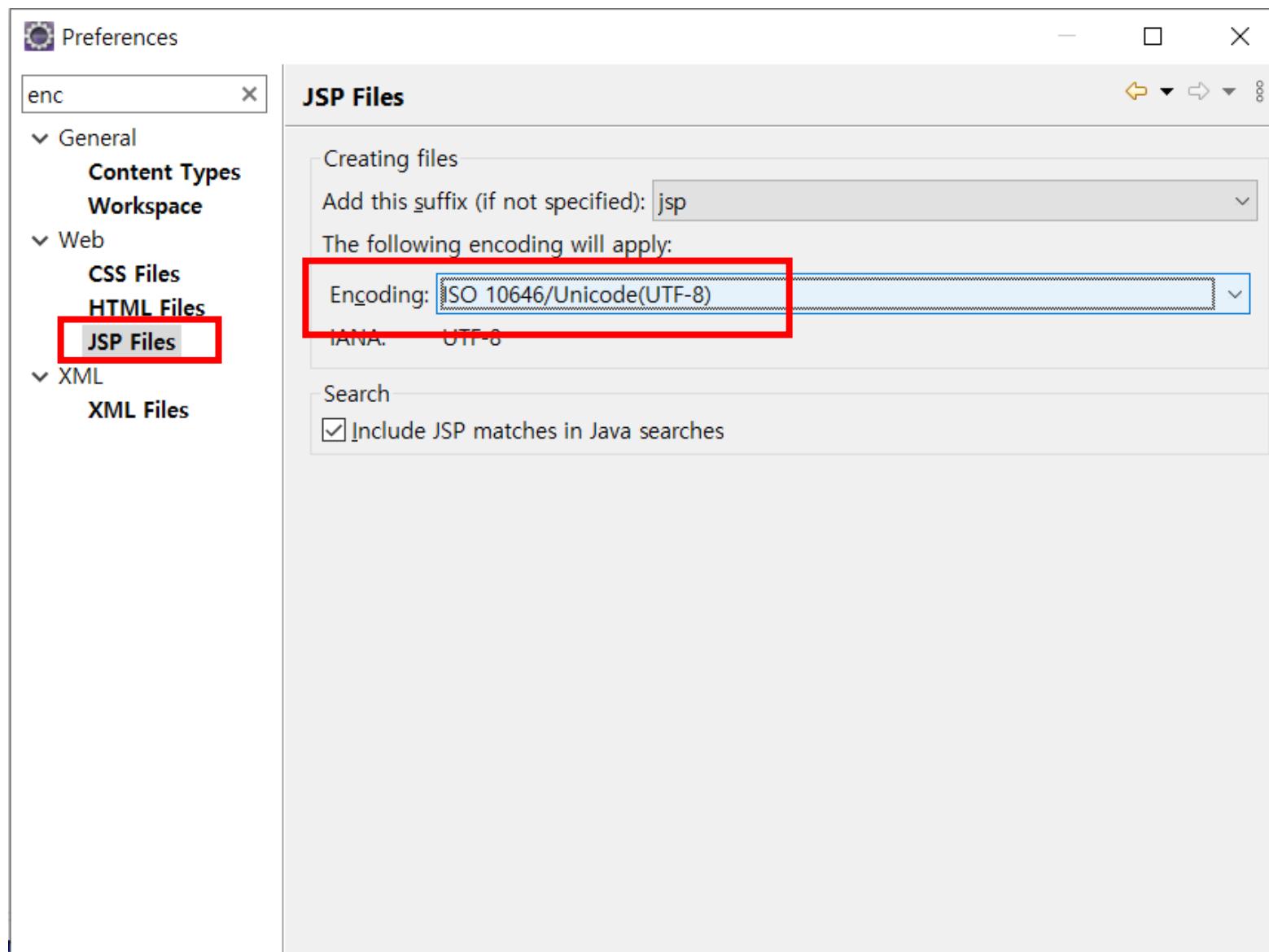
# 스타일시트 한글깨짐방지.



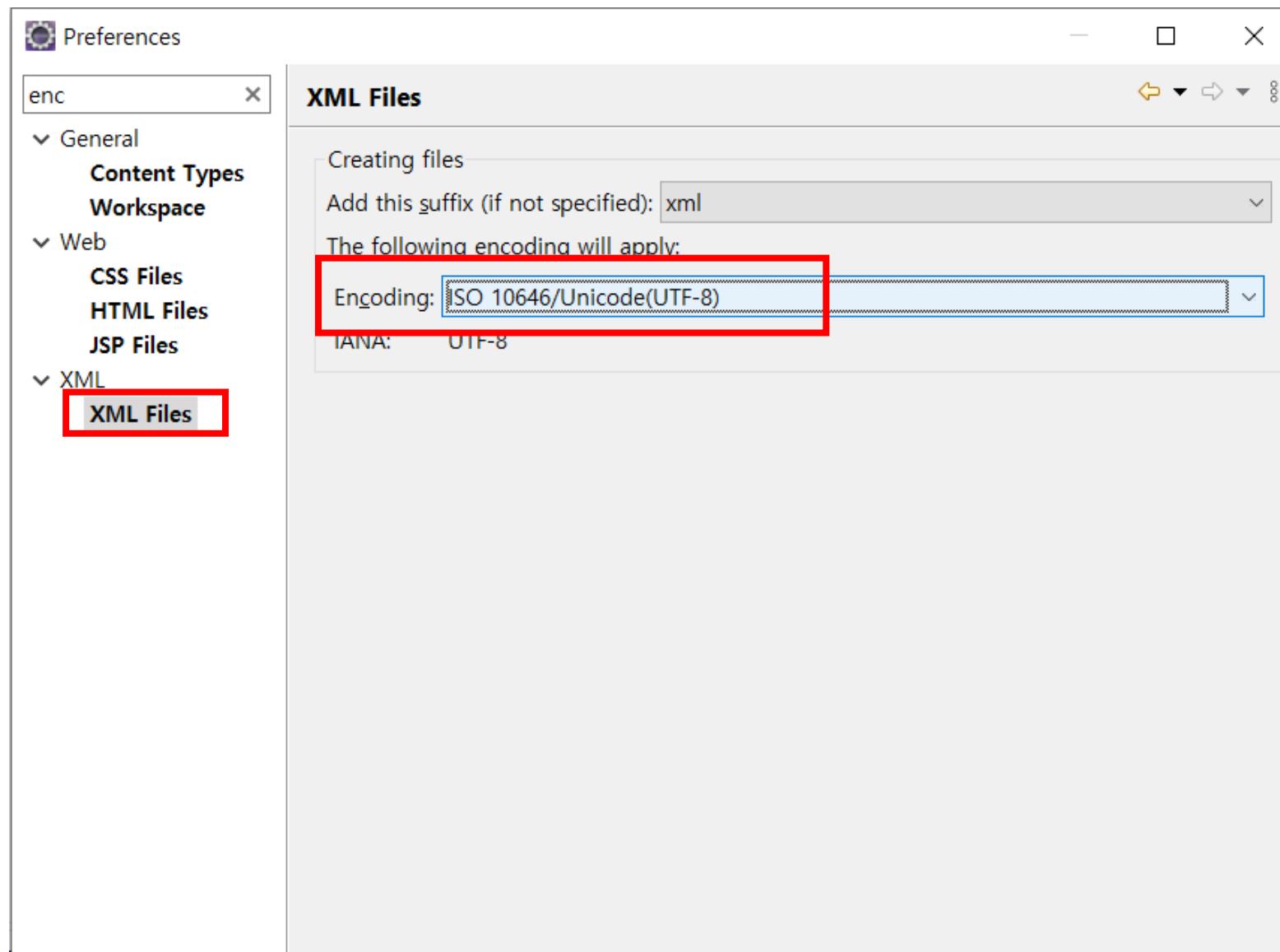
# html 파일 한글깨짐방지.



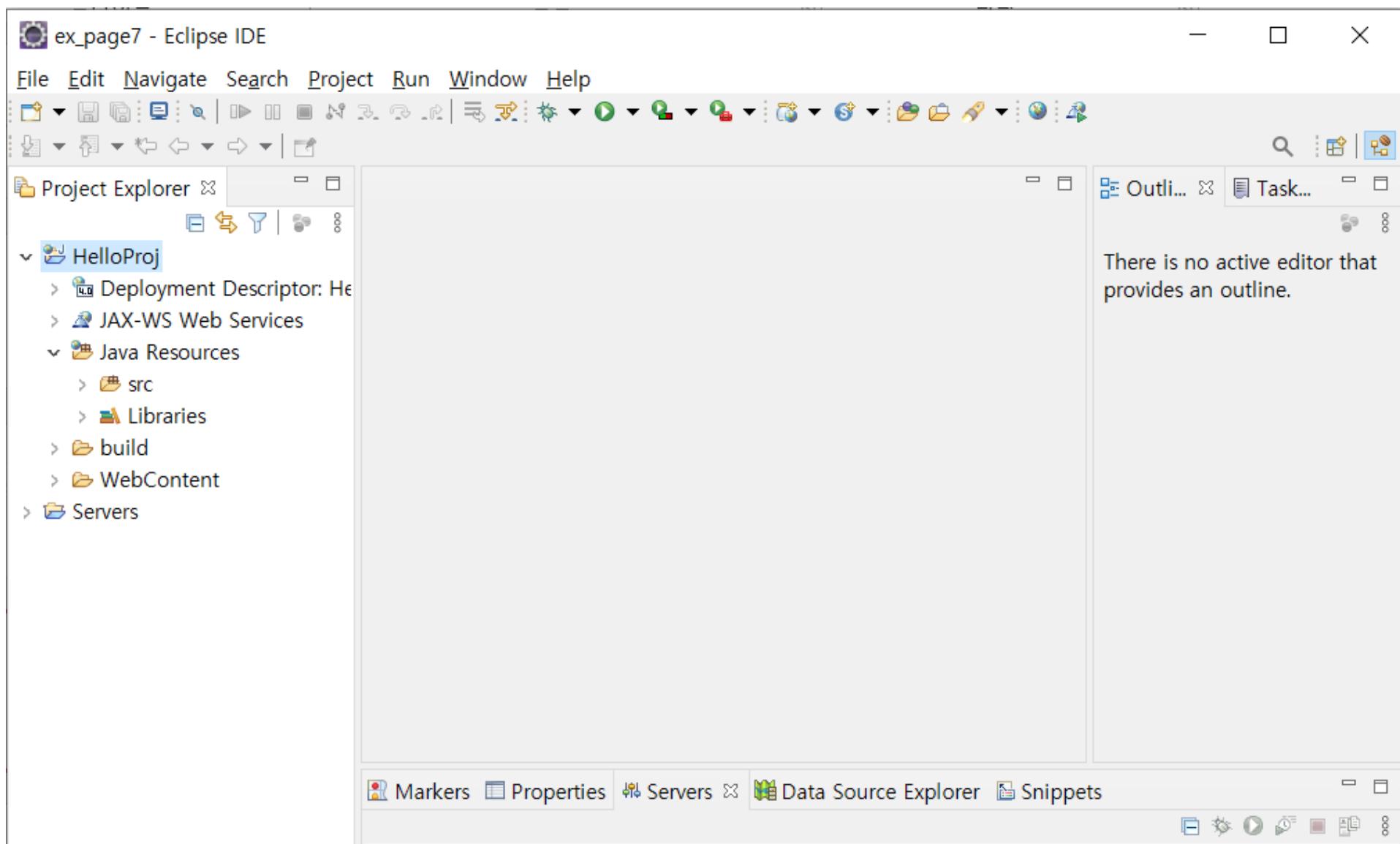
# Jsp파일 한글깨짐방지



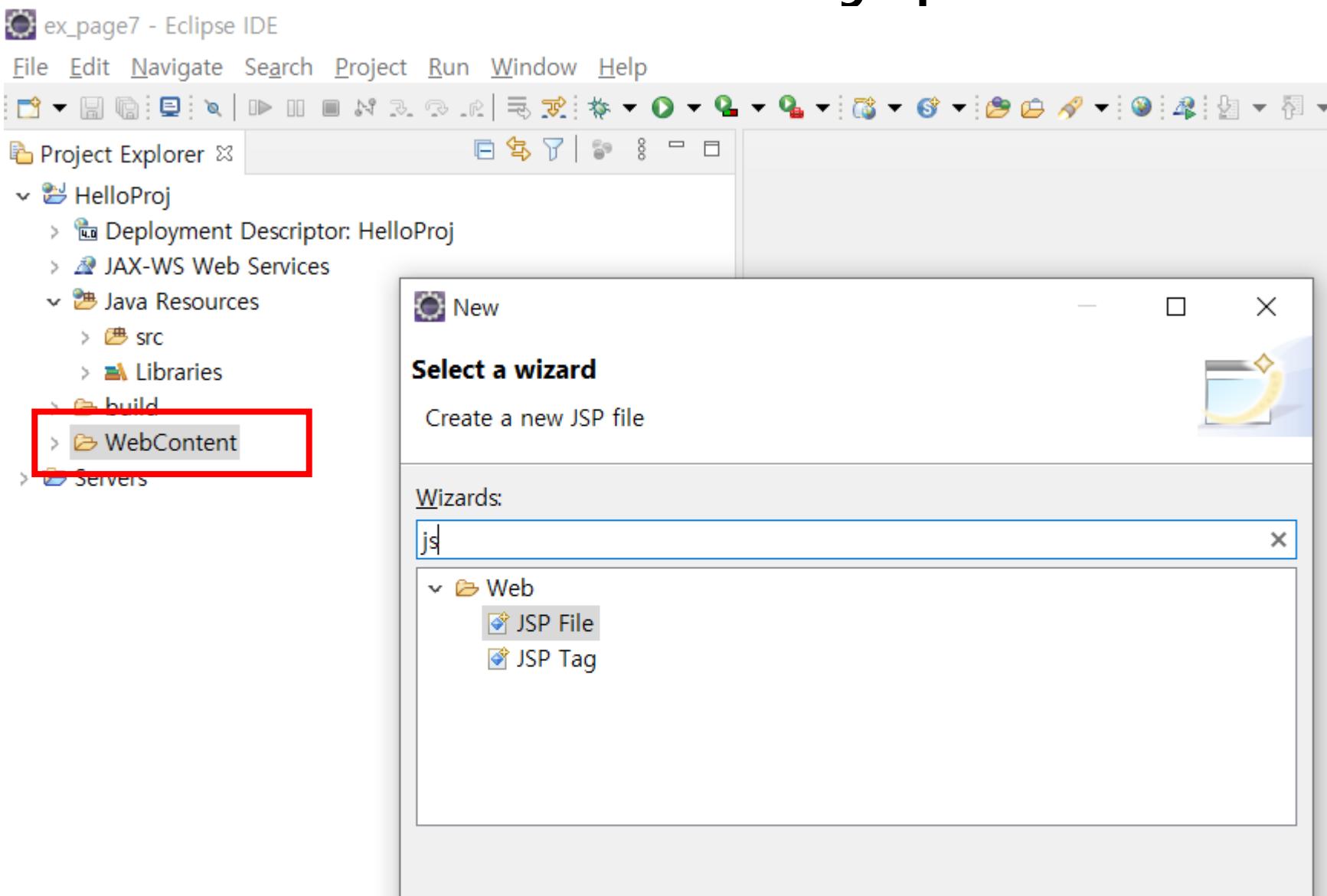
# xml 파일 한글깨짐 방지



# C. Dynamic 프로젝트 작성 시작



# C\_3. WebContent에 hello.jsp 생성



# Hello.jsp

ex\_page7 - HelloProj/WebContent/Hello.jsp - Eclipse IDE

File Edit Navigate Search Project Run Window Help

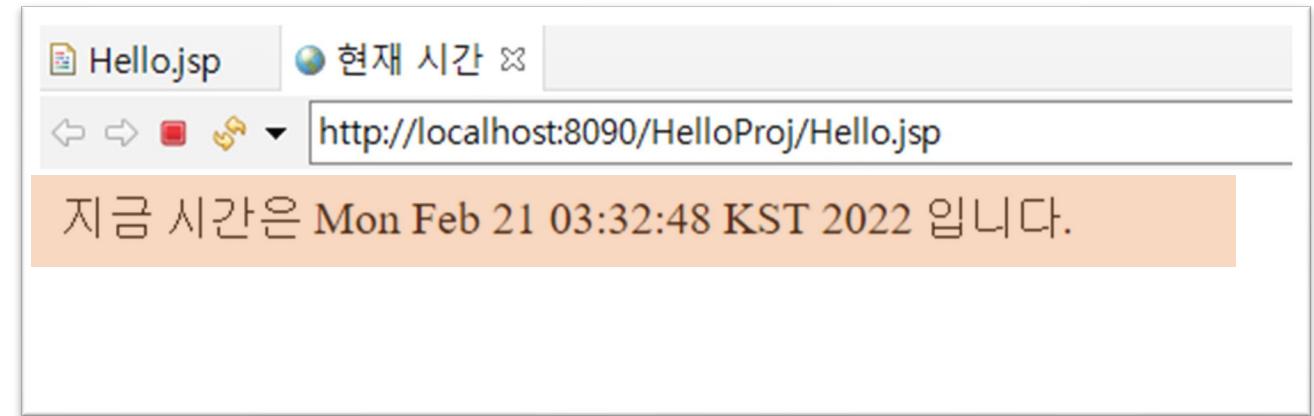
Project Explorer    Hello.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

Servers

The screenshot shows the Eclipse IDE interface. The title bar says 'ex\_page7 - HelloProj/WebContent/Hello.jsp - Eclipse IDE'. The menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The 'Project Explorer' view on the left shows a project named 'HelloProj' with its structure: Deployment Descriptor: HelloProj, JAX-WS Web Services, Java Resources (src, Libraries), build, WebContent (META-INF, WEB-INF, Hello.jsp), and Servers. The 'Hello.jsp' file is selected in the Project Explorer and is also open in the central editor area. The editor shows the JSP code with line numbers 1 through 12. A red box highlights the 'Hello.jsp' file in the Project Explorer.

# Hello.jsp 작성 및 실행



```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>현재 시간</title>
8 </head>
9 <body>
10    지금 시간은 <%= new java.util.Date() %> 입니다.
11 </body>
12 </html>
```

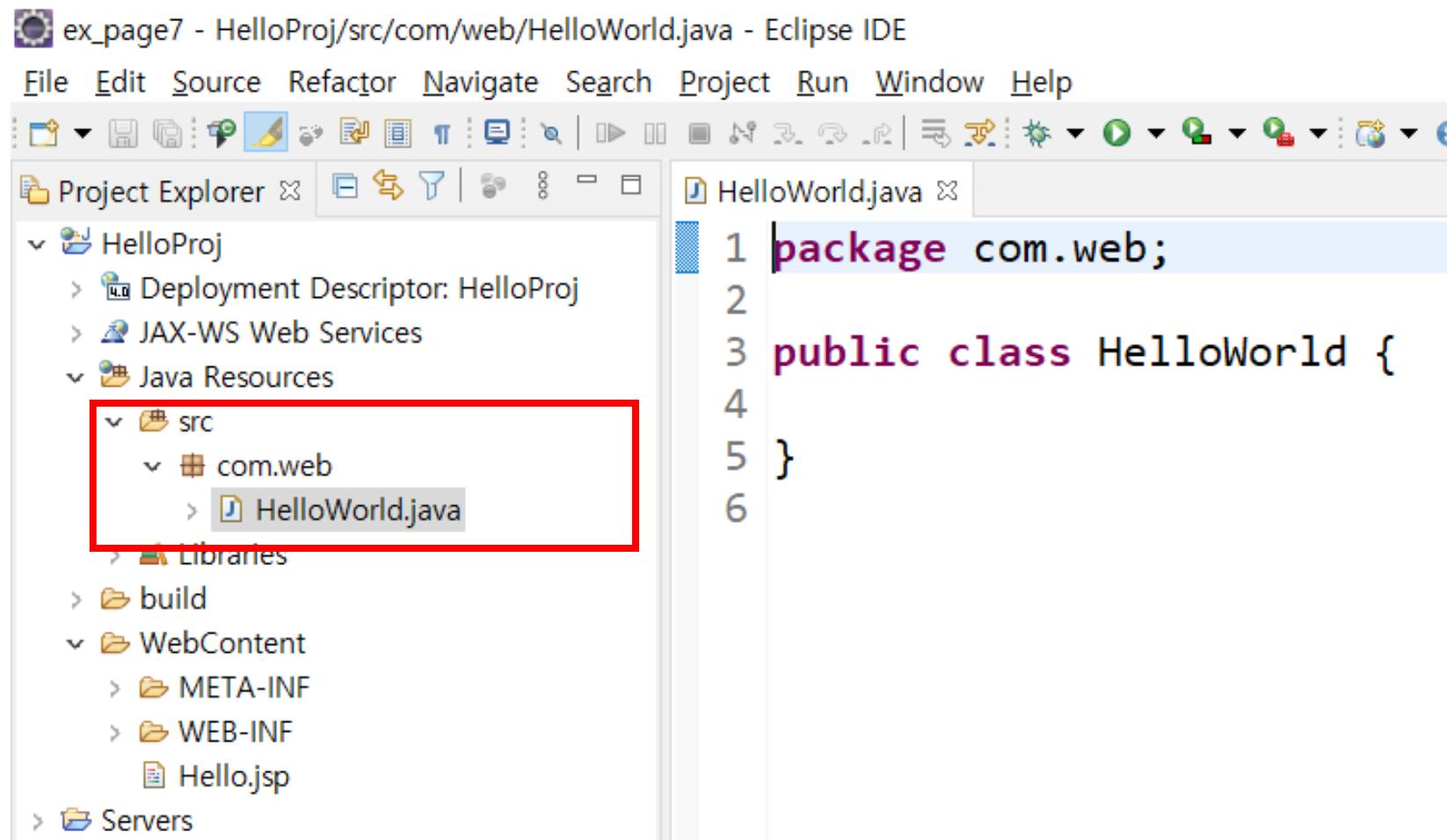
# C\_4. servlet 페이지 작성

The screenshot shows the Eclipse IDE interface with the title "ex\_page7 - Eclipse IDE". The menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The Project Explorer view on the left shows a project named "HelloProj" with a "src" folder highlighted by a red box. The "Java Resources" section contains "src", "Libraries", "build", "WebContent" (with "META-INF" and "WEB-INF" subfolders), and "Hello.jsp". The Servers view is also present.

In the center, a "New" dialog is open under the "Select a wizard" category, which is set to "Create a Java class". The "Wizards:" dropdown shows "cl" selected. Below it, a tree view lists "Class" under "Java", "Class" under "Java EE", "Application Client Project" under "Java EE", and "JAXB Classes from Schema" under "JAXB".

To the right, the "New Java Class" dialog is displayed. It has fields for "Source folder" (set to "HelloProj/src"), "Package" (set to "com.web"), and "Name" (set to "HelloWorld"). Under "Modifiers", the "public" radio button is selected. The "Superclass" field is set to "java.lang.Object". At the bottom, there's a section for "Which method stubs would you like to create?" with three checkboxes: "public static void main(String[] args)" (unchecked), "Constructors from superclass" (unchecked), and "Inherited abstract methods" (checked). A note at the bottom asks if comments should be added, with a link to configuration templates.

# HelloWorld.java



The screenshot shows the Eclipse IDE interface with the title bar "ex\_page7 - HelloProj/src/com/web/HelloWorld.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run.

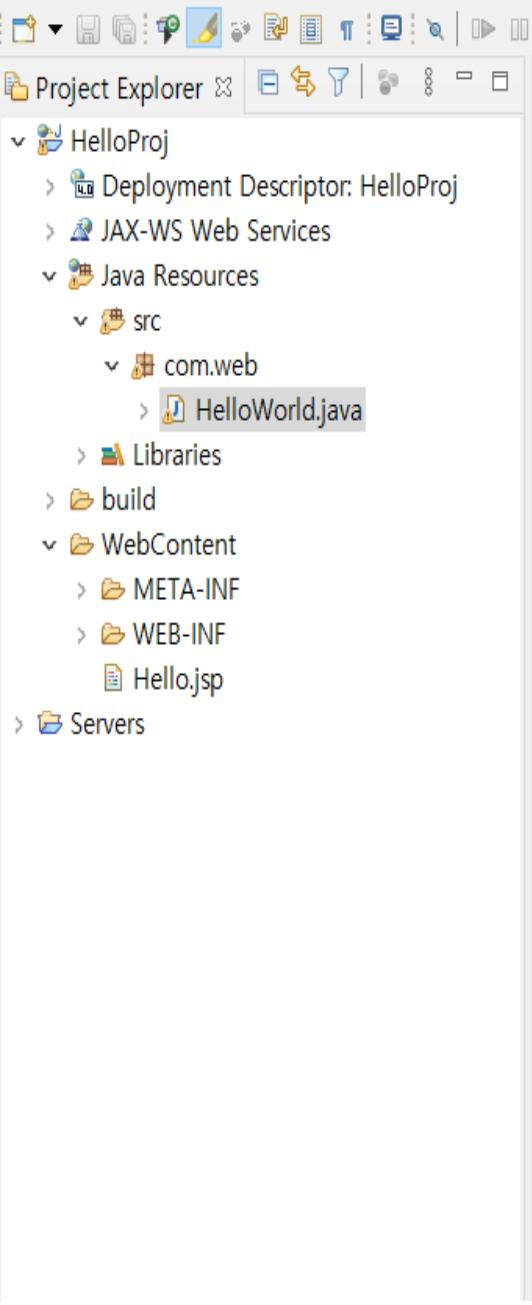
The Project Explorer view on the left shows the project structure:

- >HelloProj
  - Deployment Descriptor: HelloProj
  - JAX-WS Web Services
  - Java Resources
    - src
      - com.web
        - HelloWorld.java
    - Libraries
    - build
  - WebContent
    - META-INF
    - WEB-INF
    - Hello.jsp
  - Servers

A red box highlights the "src/com.web/HelloWorld.java" entry in the Project Explorer.

The right-hand editor window displays the code for "HelloWorld.java":

```
1 package com.web;
2
3 public class HelloWorld {
4
5 }
6
```



# HelloWorld 서블릿페이지 작성

# HelloWorld 서블릿페이지 실행

The screenshot shows a web browser window with the following details:

- Tab: HelloWorld.java (selected)
- Tab: HTTP 상태 404 – 찾을 수 없음
- Address bar: http://localhost:8090/HelloProj/servlet/com.web.HelloWorld

**HTTP 상태 404 – 찾을 수 없음**

---

**타입** 상태 보고

**메시지** 요청된 리소스 [/HelloProj/servlet/com.web.HelloWorld]은(는) 가용하지 않습니다.

**설명** Origin 서버가 대상 리소스를 위한 현재의 representation을 찾지 못했거나, 그것이 존재하는지를 밝히려 하지 않습니다.

---

**Apache Tomcat/9.0.55**

# C\_4. 서블릿 페이지의 실행을 설정

The screenshot shows the Eclipse IDE interface with the title "ex\_page7 - HelloProj/WebContent/WEB-INF/web.xml - Eclipse IDE". The menu bar includes File, Edit, Source, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The Project Explorer view on the left shows the project structure:

- HelloProj
  - Deployment Descriptor: HelloProj
  - JAX-WS Web Services
  - Java Resources
    - src
      - com.web
        - HelloWorld.java
      - Libraries
    - build
    - WebContent
      - META-INF
      - WEB-INF
        - lib
        - web.xml
- Servers

The "web.xml" file is selected in the Project Explorer and is displayed in the code editor. The code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/XMLSchema-instance
  http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
  version="2.5">
  <display-name>HelloProj</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

A red box highlights the "web.xml" file in the Project Explorer.

**페이지를 열어보자~**

**수정은 다음장에 기록...->**

# Web.xml 수정하기

ex\_page7 - HelloProj/WebContent/WEB-INF/web.xml - Eclipse IDE

File Edit Navigate Search Project Run Design Window Help

The screenshot shows the Eclipse IDE interface. The Project Explorer view on the left displays a project named 'HelloProj' with its structure: Deployment Descriptor (HelloProj), JAX-WS Web Services, Java Resources (src, com.web, HelloWorld.java, Libraries), build, WebContent (META-INF, WEB-INF, lib, web.xml), and Hello.jsp. The 'web.xml' file is selected in the Project Explorer. The code editor on the right shows the XML configuration for the web application. The URL pattern '/Hello' is highlighted with a blue selection bar. The code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
          http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
          version="2.5">
    <display-name>HelloProj</display-name>
    <servlet>
        <servlet-name>HelloServlet</servlet-name>
        <servlet-class>com.web.HelloWorld</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloServlet</servlet-name>
        <url-pattern>/Hello</url-pattern>
    </servlet-mapping>
</web-app>
```

```
1 package com.web;
2
3 import java.io.IOException;
4
5
6 public class HelloWorld extends HttpServlet {
7
8     @Override
9     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
10
11         //resp.setContentType("text/html;charset=UTF-8");
12         resp.setContentType("text/html");
13         PrintWriter out = resp.getWriter();
14         out.println("<html>");
15         out.println("<body>");
16         out.println("Hello Java Server Page ~ !!!");
17         out.println("자바 웹페이지 안녕~ !!!");
18         out.println("</body></html>");
19         out.close();
20
21     }
22
23 }
24
25
26
27 }
28
```

# 서블릿 작성 및 실행 1

http://localhost:8090/HelloProj>Hello

Hello Java Server Page ~ !!! ?? ???? ??~ !!!

HelloWorld.java

```
1 package com.web;
2
3+import java.io.IOException;□
10
11 public class HelloWorld extends HttpServlet {
12
13@Override
14 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
15
16     resp.setContentType("text/html;charset=UTF-8");
17     //resp.setContentType("text/html");
18     PrintWriter out = resp.getWriter();
19     out.println("<html>");
20     out.println("<body>");
21     out.println("Hello Java Server Page ~ !!!");
22     out.println("자바 웹페이지 안녕~ !!!");
23     out.println("</body></html>");
24     out.close();
25 }
26
27 }
28
```

## 서블릿 작성 및 실행 2

http://localhost:8090/HelloProj>Hello

The screenshot shows a browser window with the URL "http://localhost:8090/HelloProj>Hello". The page content is "Hello Java Server Page ~ !!! 자바 웹페이지 안녕~ !!!".

```
http://localhost:8090/HelloProj>Hello
http://localhost:8090/HelloProj>Hello
Hello Java Server Page ~ !!! 자바 웹페이지 안녕~ !!!
```