

# **WEB PROGRAMMING**

01/22 16:00 PM

# Java Server Programming

# Server Side Script Language

---

## ➤ JSP, Servlet

- Java 언어를 기반으로 한 웹 프로그래밍 기술
- 장점: MVC 구현  
(Model: Java, View:JSP, Controller:Servlet)
- 서버: Tomcat

## JSP (Java Server Page)와 Servlet

---

- JSP와 Servlet은 동일한 역할을 하지만, 작성되는 코드 스타일이 다름
- JSP : HTML 문서 안에 Java 코드가 스크립트 형식으로 추가됨
- Servlet : Java 코드 안에 HTML 태그가 문자열 형식으로 추가됨

# Server Side Script Language

---

## ➤ JSP

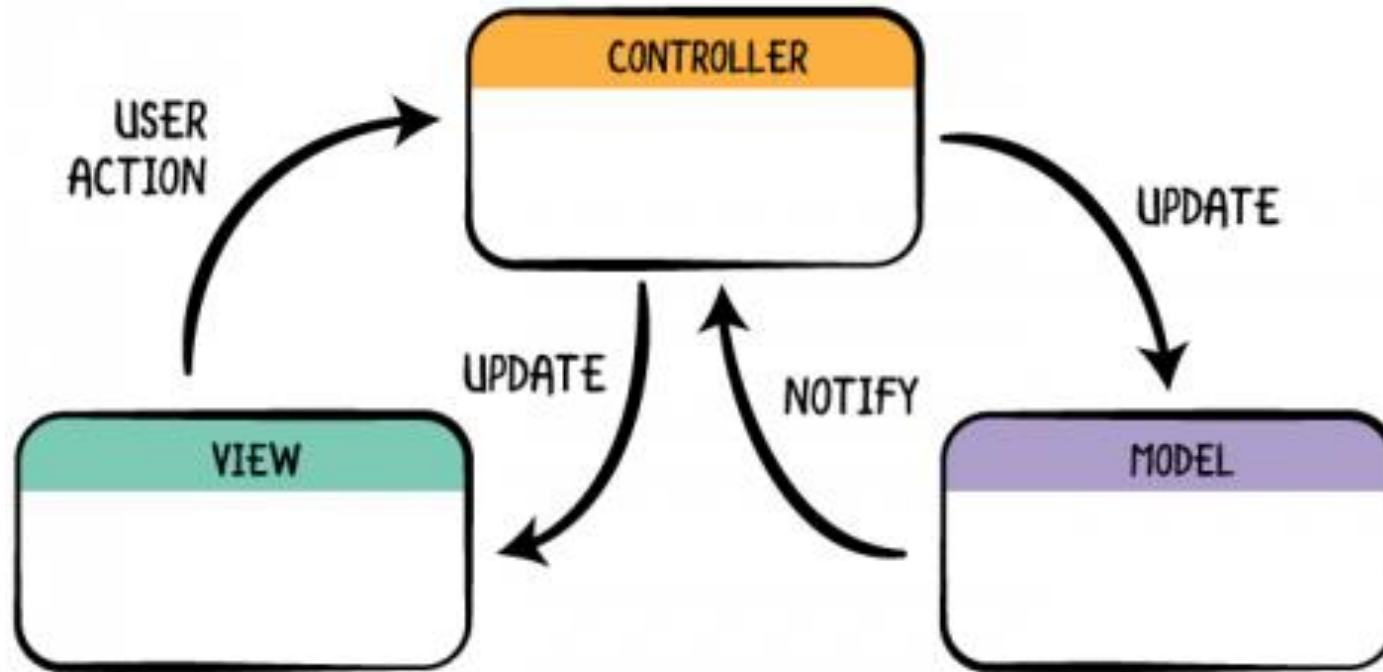
- .jsp 확장자
- 클라이언트의 요청에 동적으로 작동
- 응답은 HTML 이용
- 웹 서버와 통신 시에 자동으로 Servlet으로 변환됨
- MVC 패턴에서 View로 이용

## ➤ Servlet

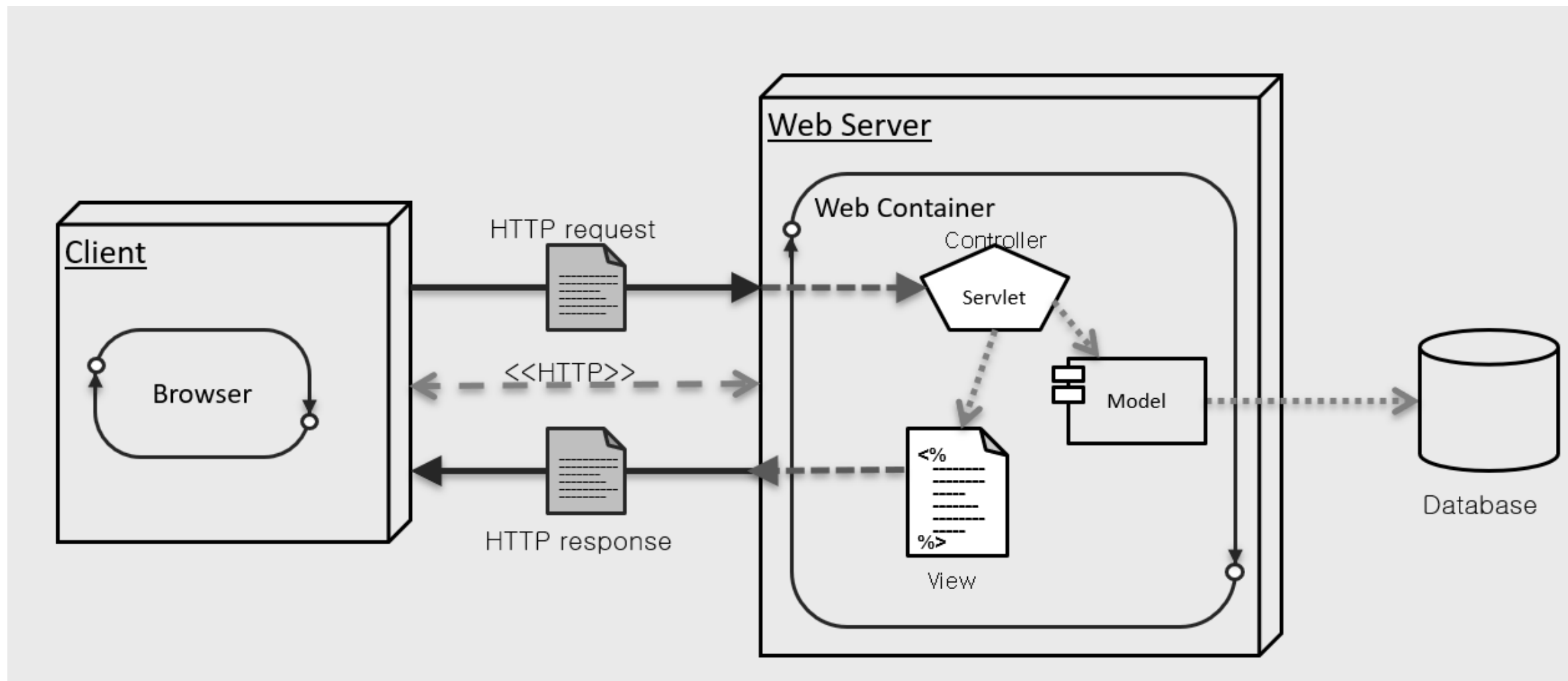
- .java 확장자
- 클라이언트의 요청에 동적으로 작동
- 응답은 HTML 이용
- java thread 이용하여 동작
- MVC 패턴에서 Controller로 이용

# MVC, JSP 아키텍처

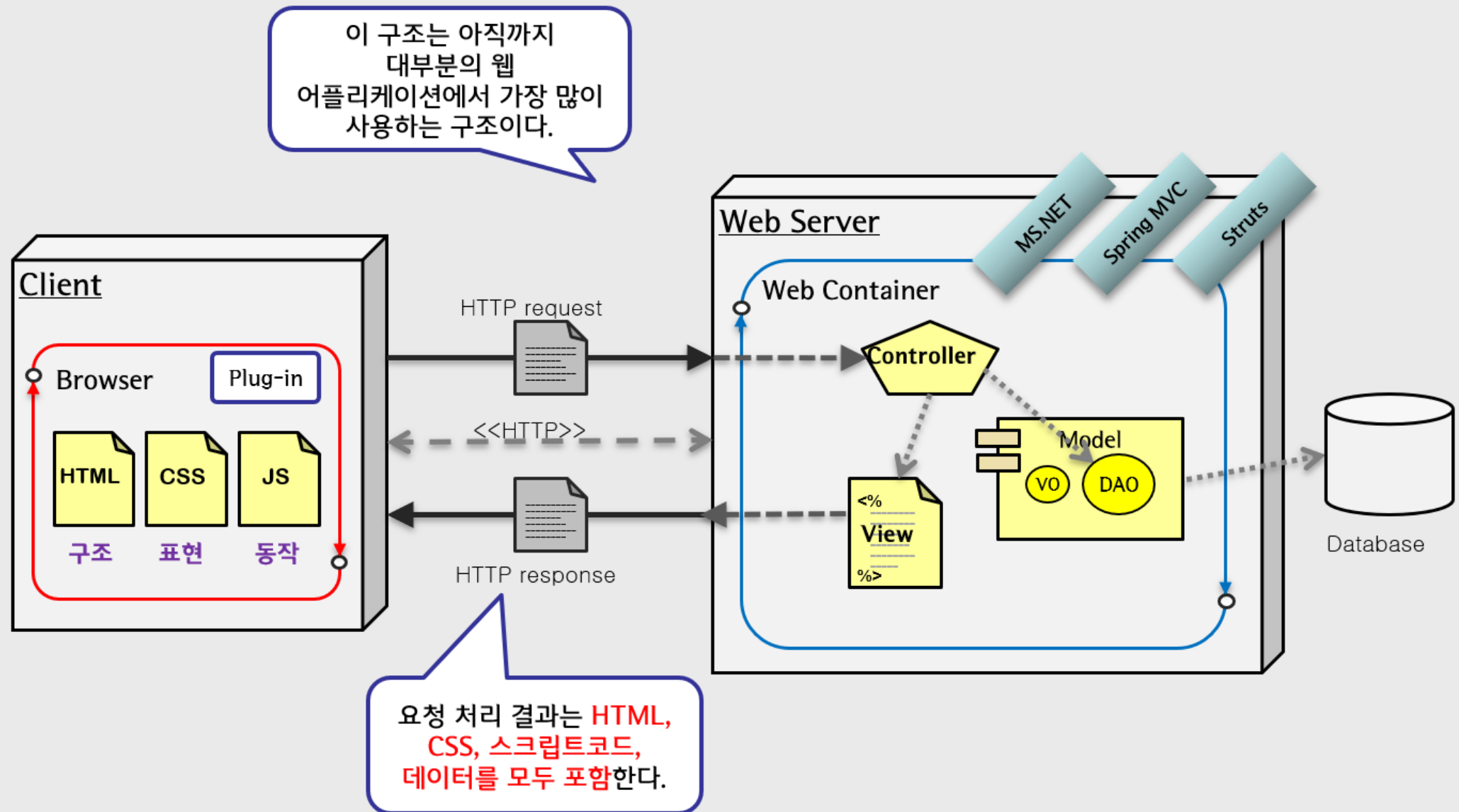
- Model : Controller의 요청에 따라 DB와 작업 (java class file)
- View : UI (jsp file)
- Controller : View와 Model 연결 및 조율 (jsp/servlet file)



# MVC Model 2 아키텍처



# Web Application Architecture



# MVC, JSP 아키텍처

## ➤ DAO Class (Data Access Object)

- DB에 접속하여 데이터 추가, 수정, 삭제 등의 작업을 하는 클래스
  - 테이블로부터 데이터를 읽음 -> 자바 객체로 변환
  - 자바 객체의 값을 읽음 -> 테이블에 저장
- 유지 보수 및 코드 모듈화를 위해 별도의 DAO 클래스를 만들어 사용
- 한 개의 DB 테이블마다 한 개의 DAO 클래스 작성
- 테이블로부터 데이터를 읽음 -> 자바 객체로 변환
- 자바 객체의 값을 읽음 -> 테이블에 저장

## ➤ VO Class (Value Object)

- DAO Class를 이용하여 DB의 데이터를 관리할 때, 테이블의 컬럼과 매핑되는 값을 갖는 Java Bean 클래스
- DB와 관련된 변수들의 모음 역할



# MVC, JSP 아키텍처

---

- JSP 아키텍처
  - JSP 파일 실행(요청) → 웹 서버에서 JSP 파일을 Java 코드로 변환
    - 해당 파일 컴파일 → HTML로 응답
  - ex) index.jsp → index\_jsp.java (서버에서 서블릿화)
    - index\_jsp.class (servlet class compiled)

# Web Site 구조

---

- ✓ 웹 사이트 = HTML 페이지와 여러 미디어 파일의 모음
- ✓ 이 파일들은 서버에 여러 디렉토리에 나누어 저장
- ✓ 웹 사이트의 페이지에 접근 위한 형식
  - protocol://host:port/path/file
  - ex) <http://mypage:8181/page/index.html>
- ✓ index.html은 주소에 파일명 입력하지 않아도 됨

# 개발 환경 구축

# Apache Tomcat 웹 서버 설치

- ✓ tomcat.apache.org/
- ✓ tomcat 8.0 설치

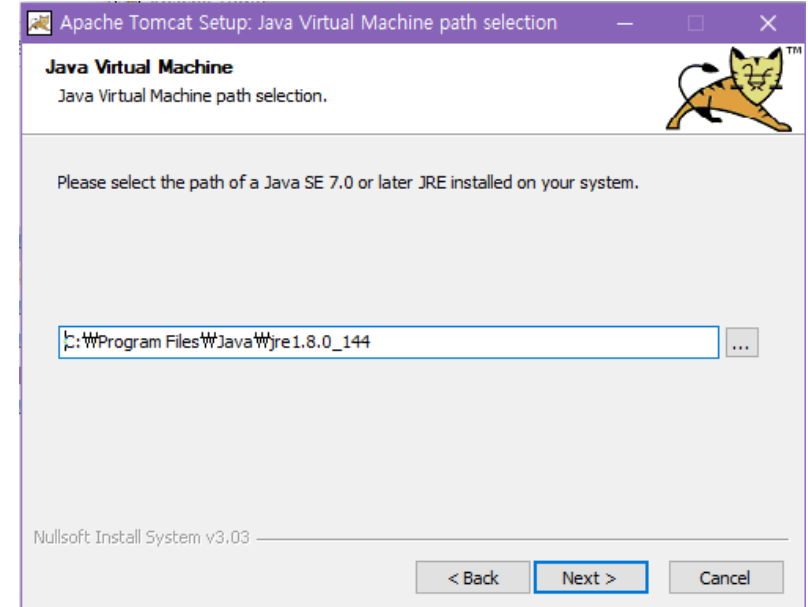
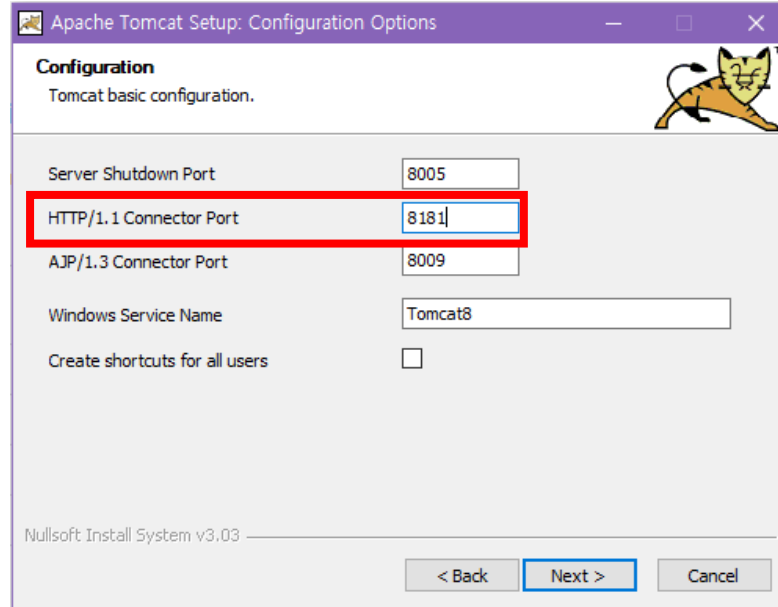
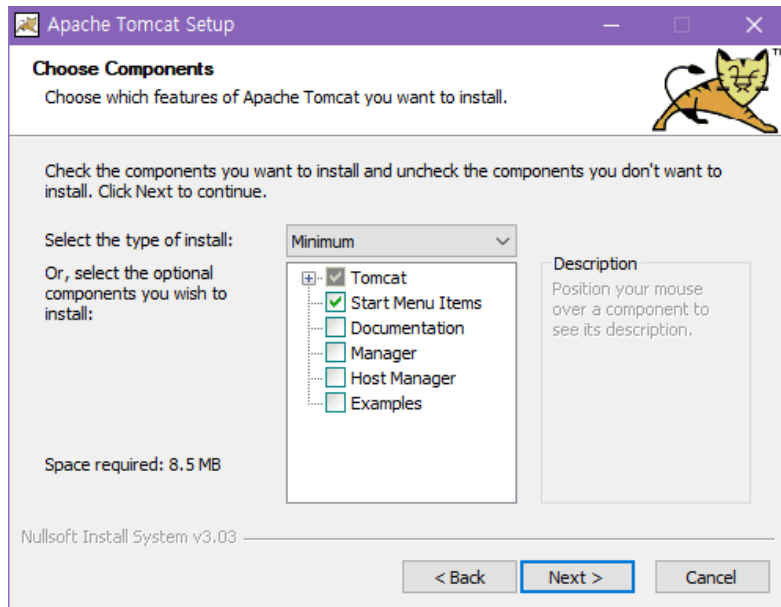
## 8.5.37

Please see the [README](#) file for packaging information. It explains what every distribution contains.

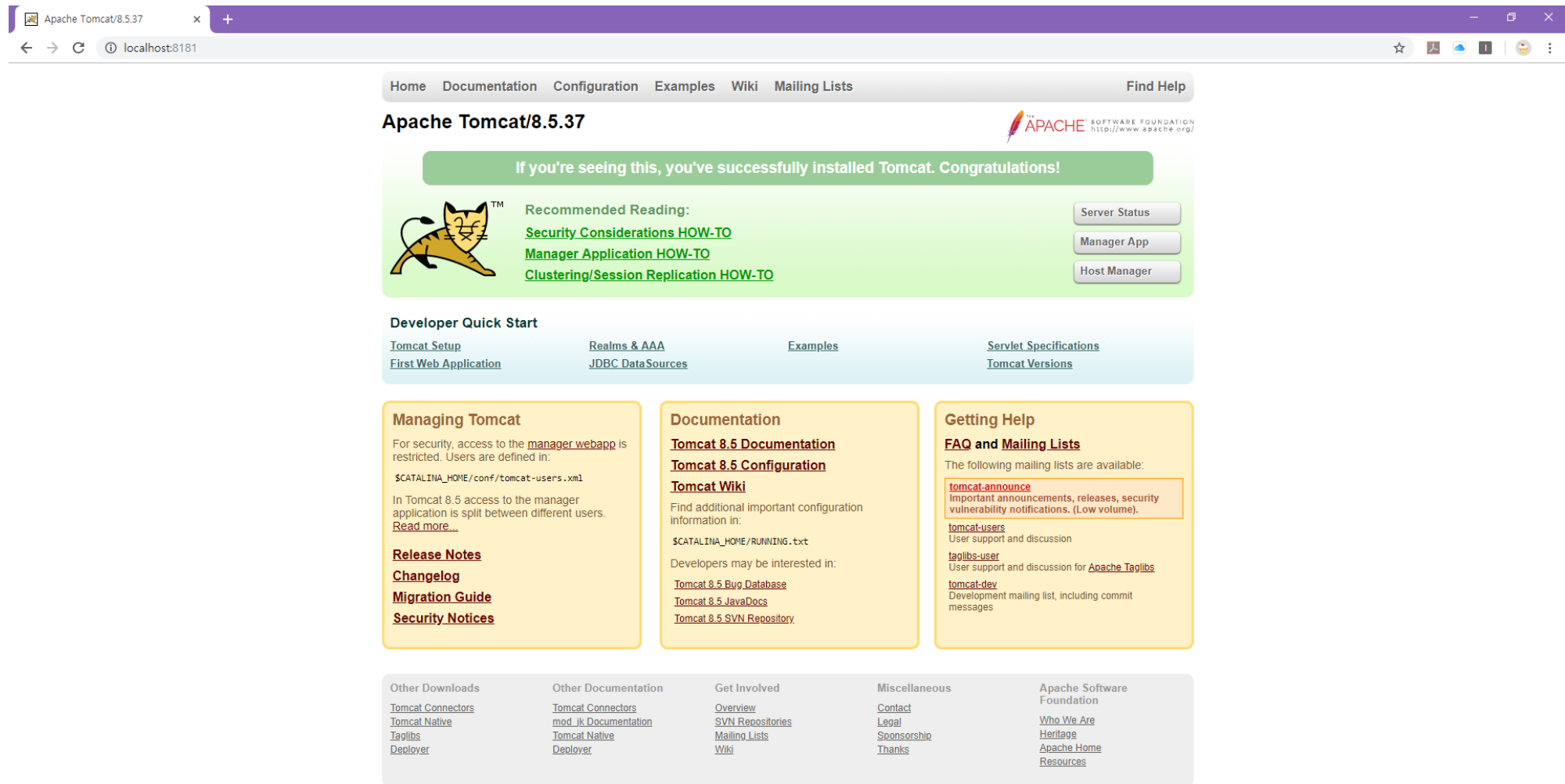
### Binary Distributions

- Core:
  - [zip](#) ([pgp](#), [sha512](#))
  - [tar.gz](#) ([pgp](#), [sha512](#))
  - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
  - [64-bit Windows zip](#) ([pgp](#), [sha512](#))
  - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))
- Full documentation:
  - [tar.gz](#) ([pgp](#), [sha512](#))
- Deployer:
  - [zip](#) ([pgp](#), [sha512](#))
  - [tar.gz](#) ([pgp](#), [sha512](#))
- Extras:
  - [JMX Remote jar](#) ([pgp](#), [sha512](#))
  - [Web services jar](#) ([pgp](#), [sha512](#))
- Embedded:
  - [tar.gz](#) ([pgp](#), [sha512](#))
  - [zip](#) ([pgp](#), [sha512](#))

# Apache Tomcat 웹 서버 설치



✓ <http://localhost:8181> 접속 시 고양이 뜨면 설치 완료




Apache Tomcat/8.5.37

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

**Apache Tomcat/8.5.37**

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:

- [Security Considerations HOW-TO](#)
- [Manager Application HOW-TO](#)
- [Clustering/Session Replication HOW-TO](#)

Server Status  
Manager App  
Host Manager

**Developer Quick Start**

- [Tomcat Setup](#)
- [First Web Application](#)
- [Realms & AAA](#)
- [JDBC DataSources](#)
- [Examples](#)
- [Servlet Specifications](#)
- [Tomcat Versions](#)

**Managing Tomcat**

For security, access to the [manager webapp](#) is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 8.5 access to the manager application is split between different users.  
[Read more...](#)

[Release Notes](#)  
[Changelog](#)  
[Migration Guide](#)  
[Security Notices](#)

**Documentation**

[Tomcat 8.5 Documentation](#)  
[Tomcat 8.5 Configuration](#)  
[Tomcat Wiki](#)

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

- [Tomcat 8.5 Bug Database](#)
- [Tomcat 8.5 JavaDocs](#)
- [Tomcat 8.5 SVN Repository](#)

**Getting Help**

**FAQ and Mailing Lists**

The following mailing lists are available:

- [tomcat-announce](#)  
Important announcements, releases, security vulnerability notifications. (Low volume).
- [tomcat-users](#)  
User support and discussion
- [taglibs-user](#)  
User support and discussion for [Apache Taglibs](#)
- [tomcat-dev](#)  
Development mailing list, including commit messages

**Other Downloads**

- [Tomcat Connectors](#)
- [Tomcat Native](#)
- [Taglibs](#)
- [Deployer](#)

**Other Documentation**

- [Tomcat Connectors Overview](#)
- [mod\\_jk Documentation](#)
- [Tomcat Native](#)
- [Deployer](#)

**Get Involved**

- [Overview](#)
- [SVN Repositories](#)
- [Mailing Lists](#)
- [Wiki](#)

**Miscellaneous**

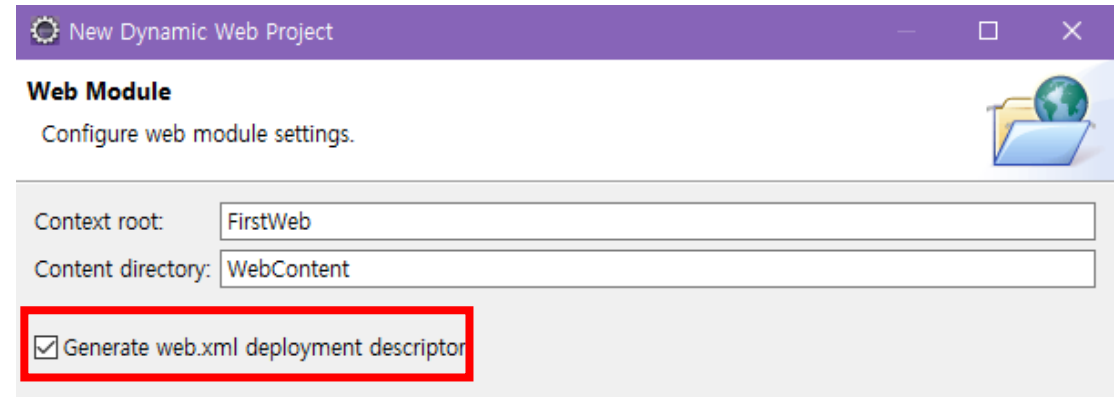
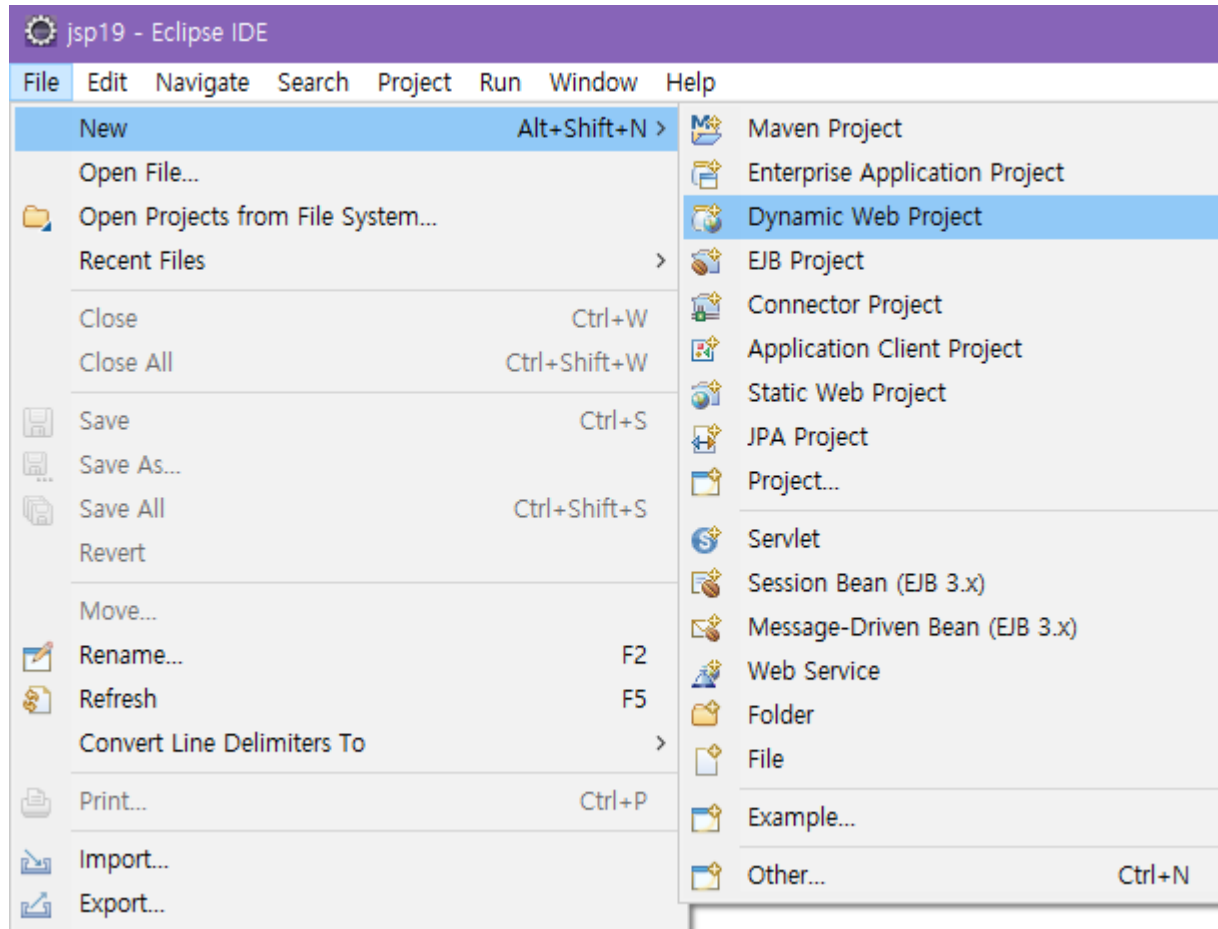
- [Contact](#)
- [Legal](#)
- [Sponsorship](#)
- [Thanks](#)

**Apache Software Foundation**

- [Who We Are](#)
- [Heritage](#)
- [Apache Home Resources](#)

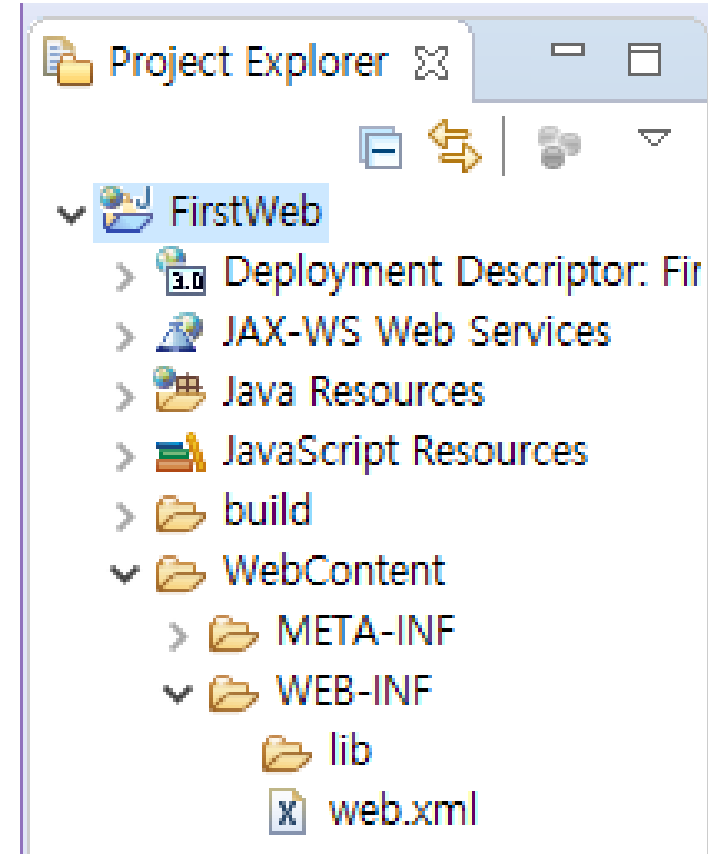
# Dynamic Web Project 생성

- File > New > New Dynamic Web Project > next, next > Generate web.xml deployment descriptor



# Dynamic Web Project 생성

- Java Resources : Java 파일
- JavaScript Resources : img 파일 등
- WebContent : jsp 파일

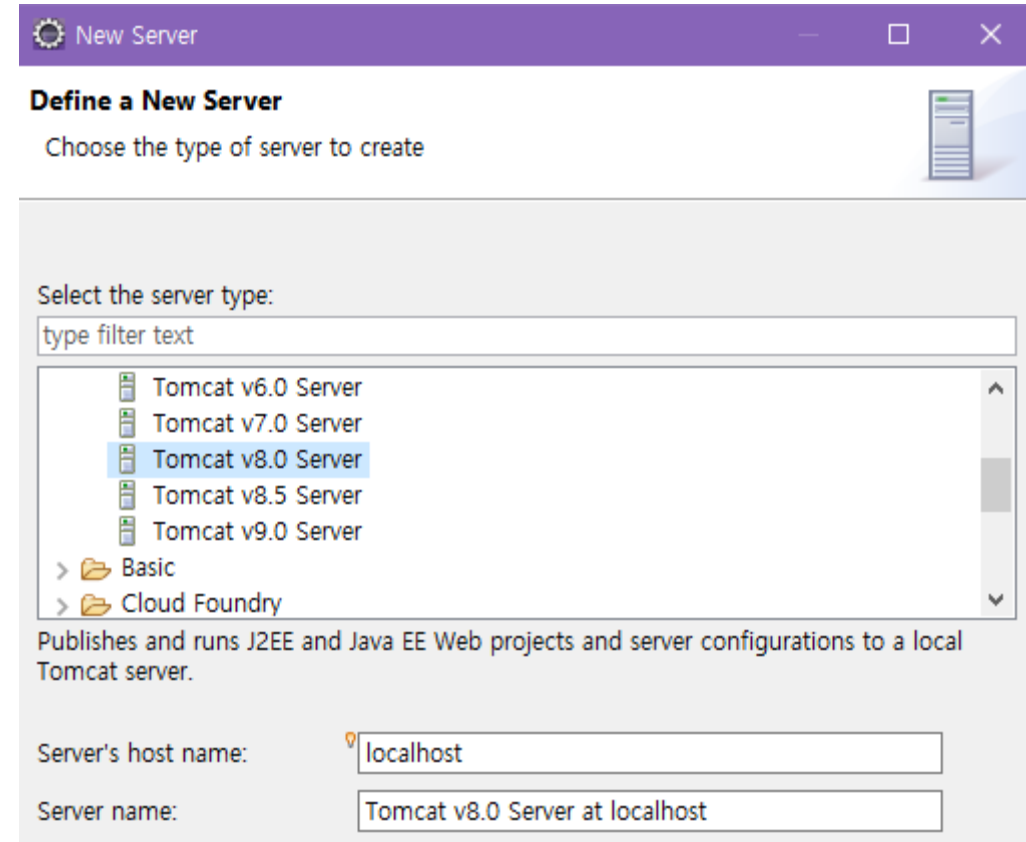


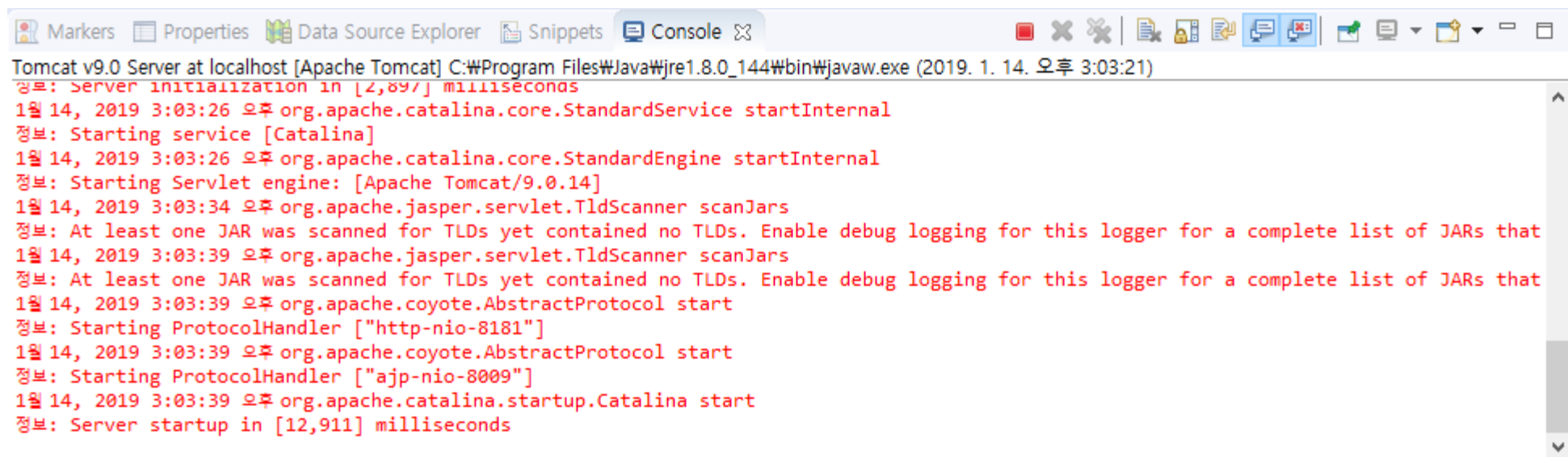


# Server 추가

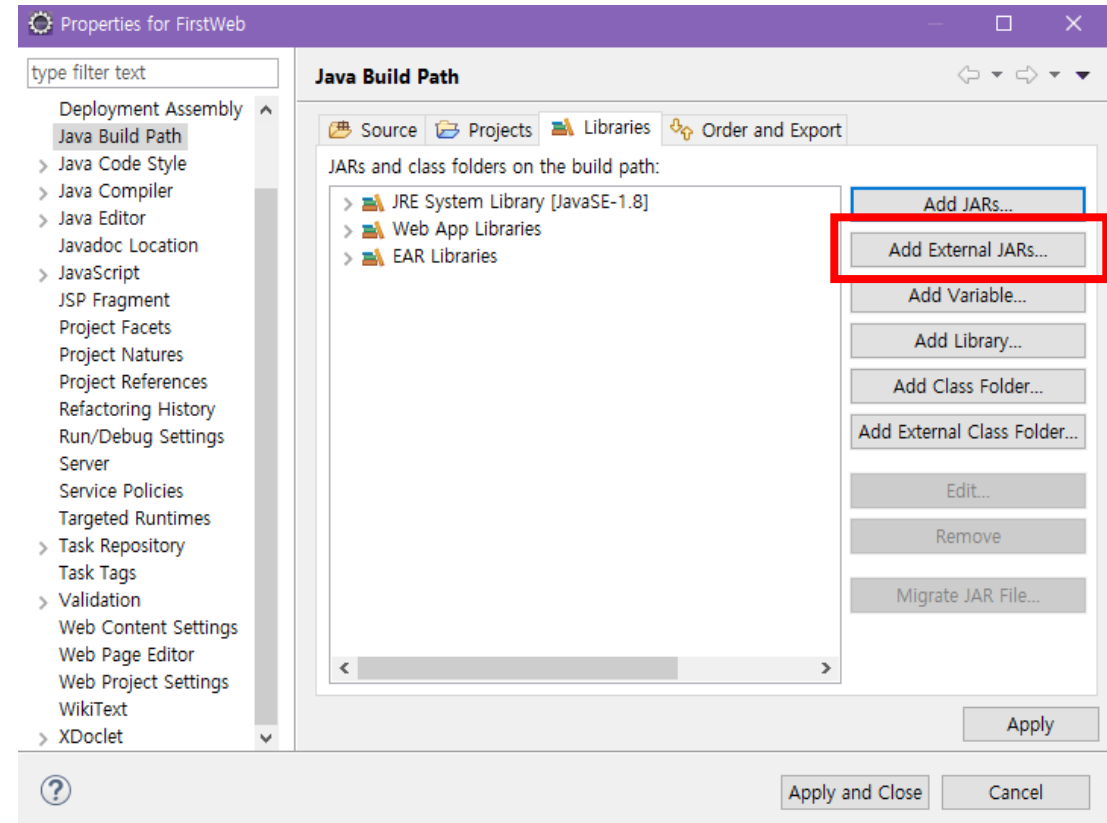
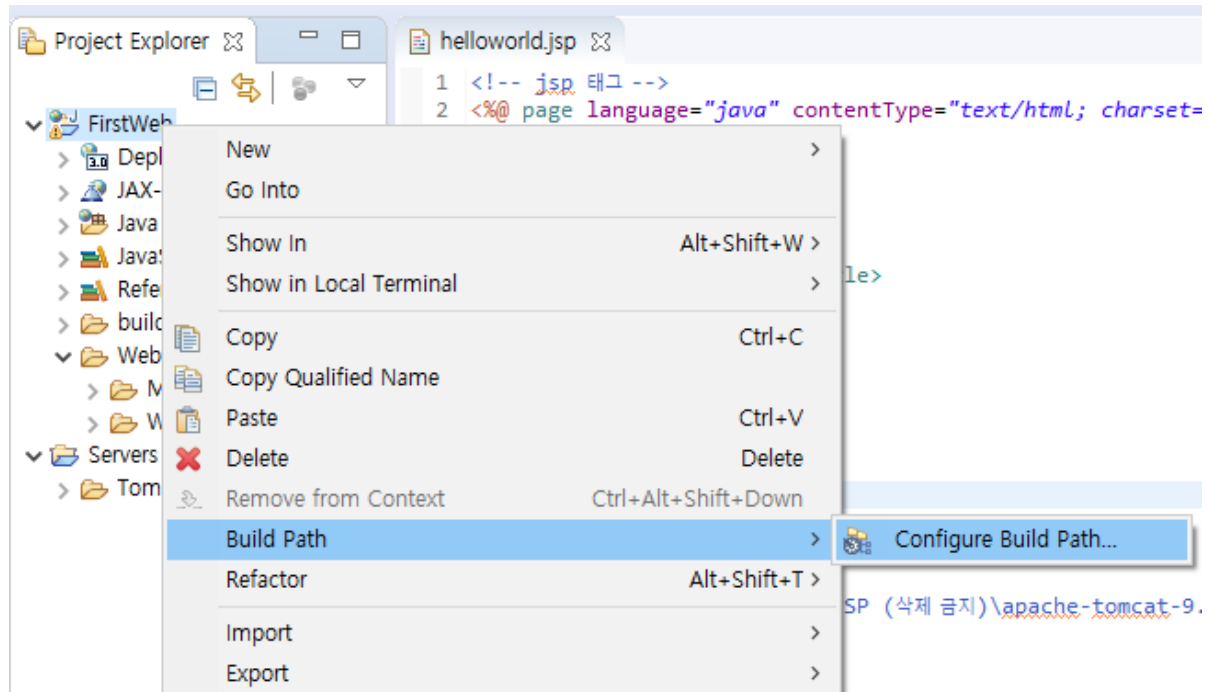


[No servers are available. Click this link to create a new server...](#)

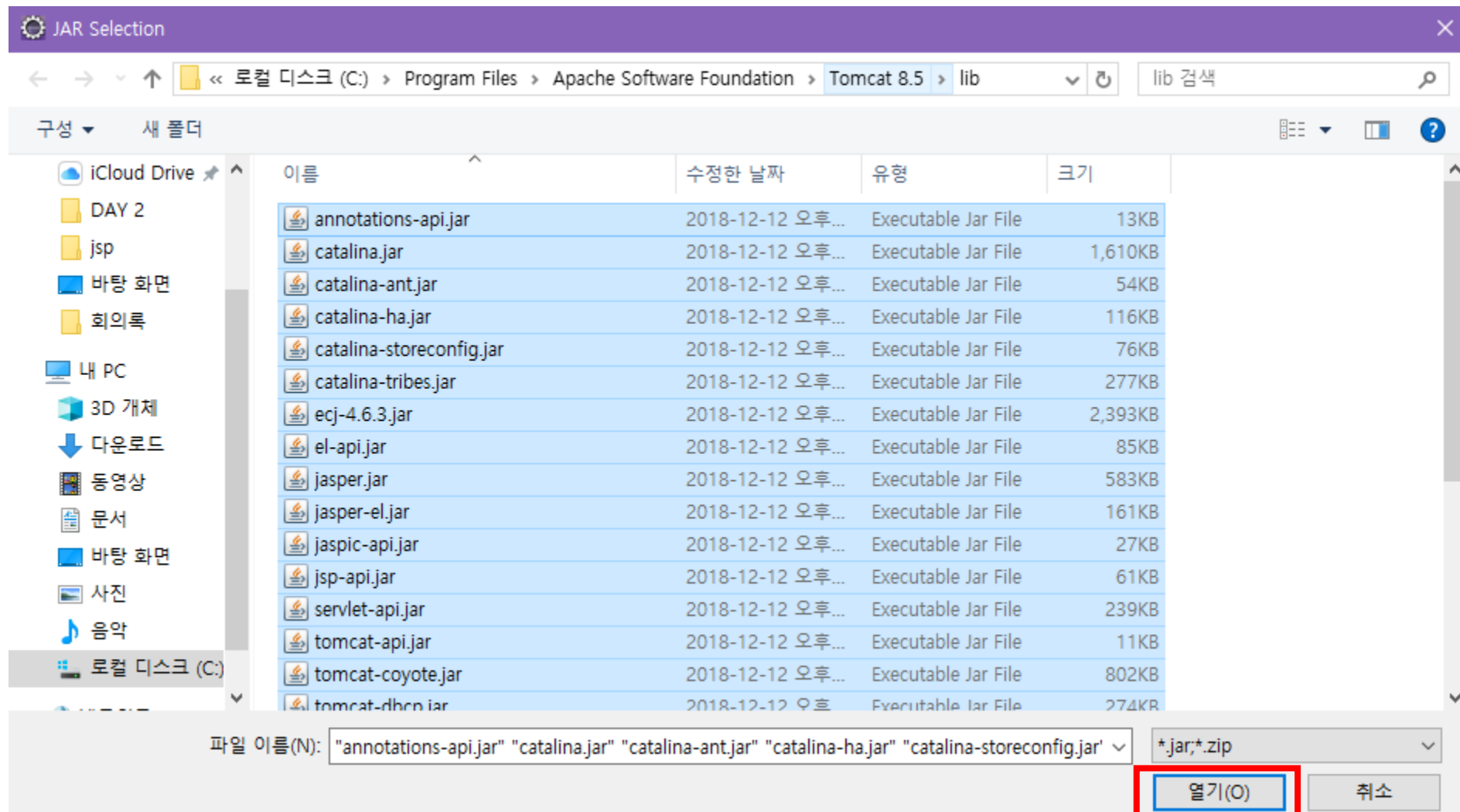




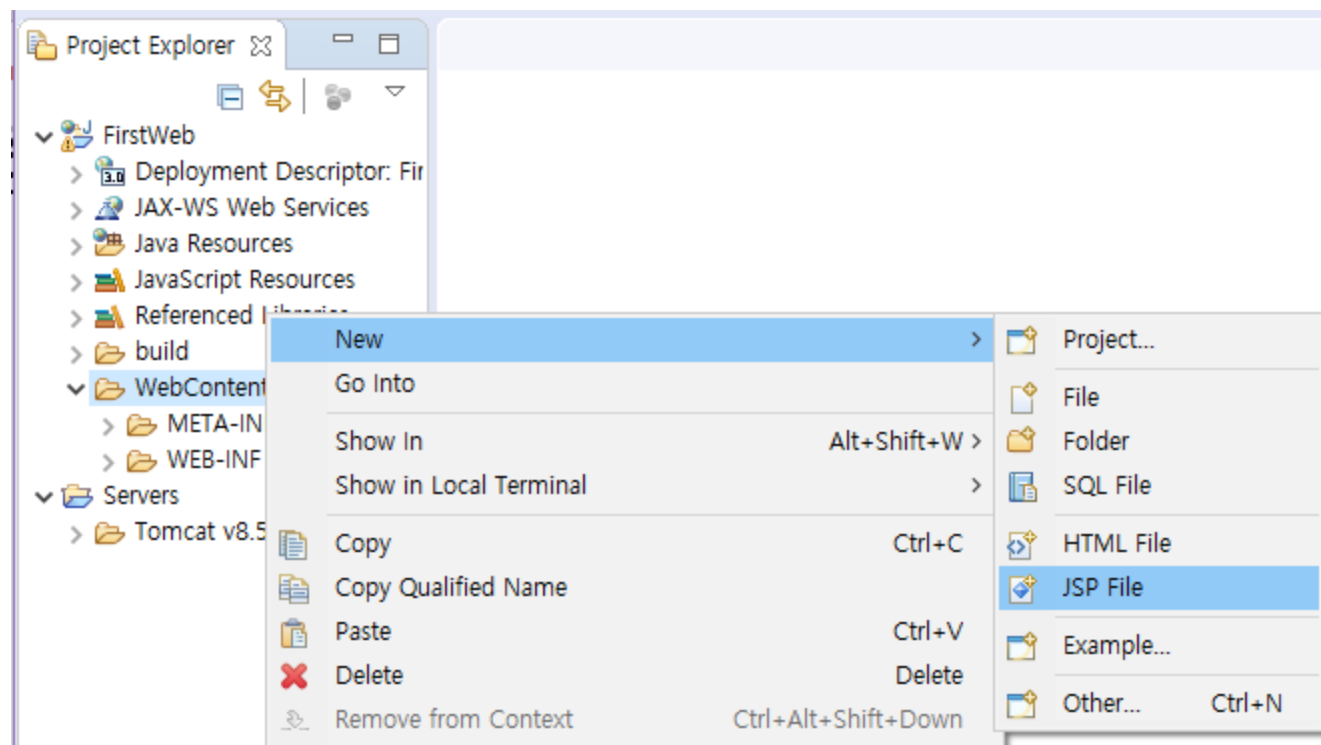
# Build Path



# Build Path



# JSP File 생성



# JSP File 생성

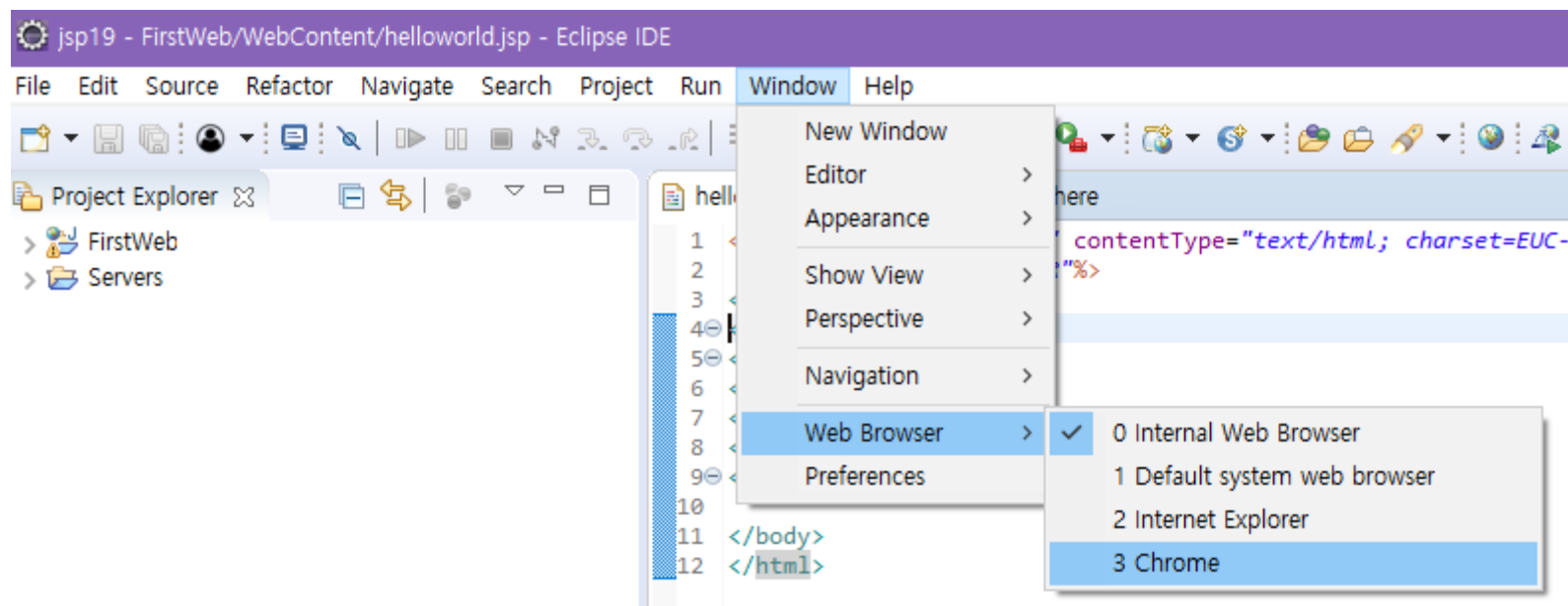
\*helloworld.jsp

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

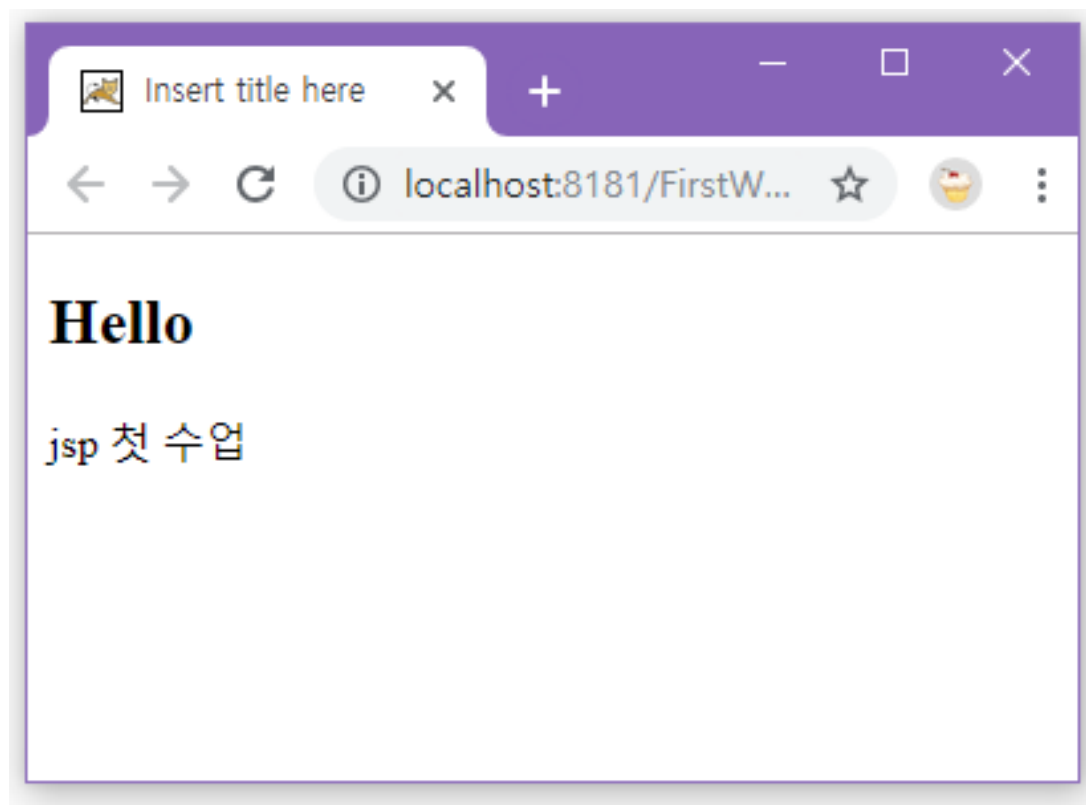
helloworld.jsp

```
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   <h2>Hello</h2>
11   jsp 첫 수업 <br>
12 </body>
13 </html>
```

# Web Browser 설정

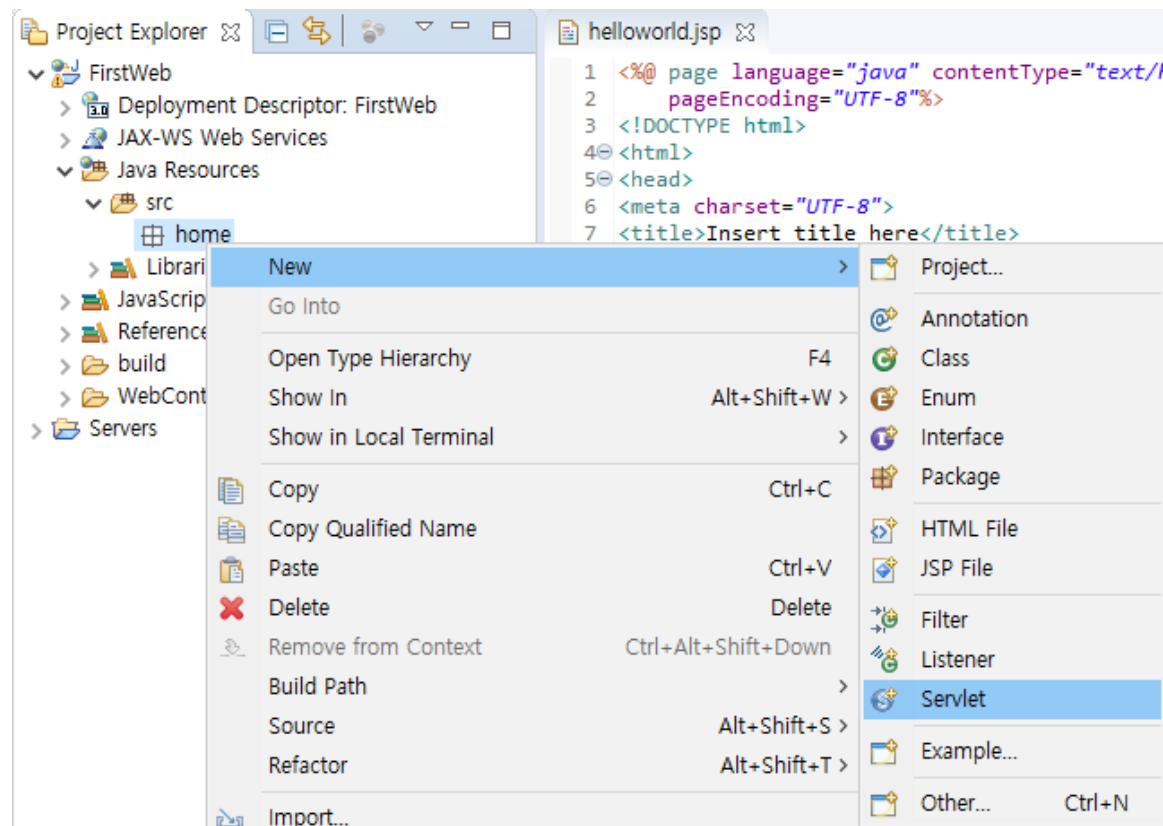
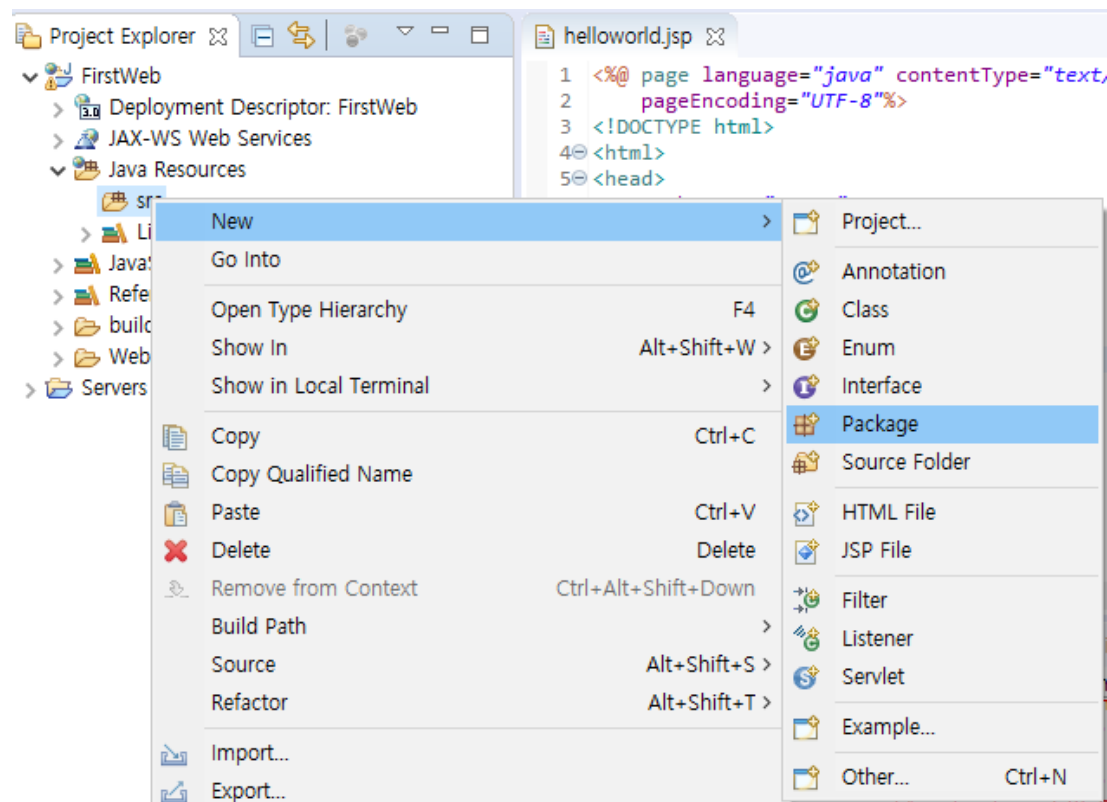


# Project 실행





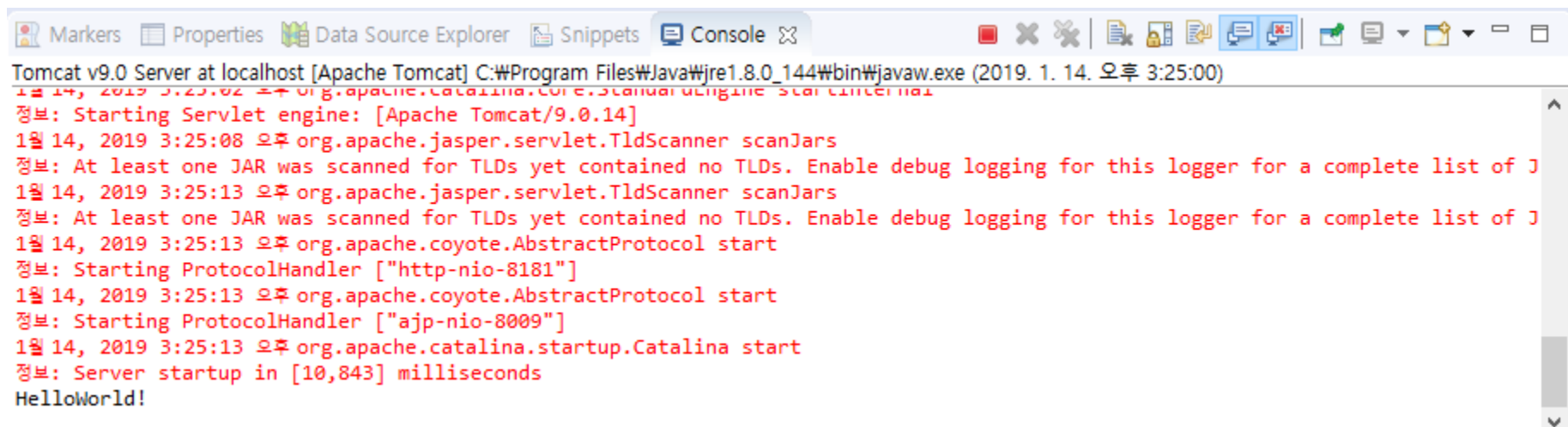
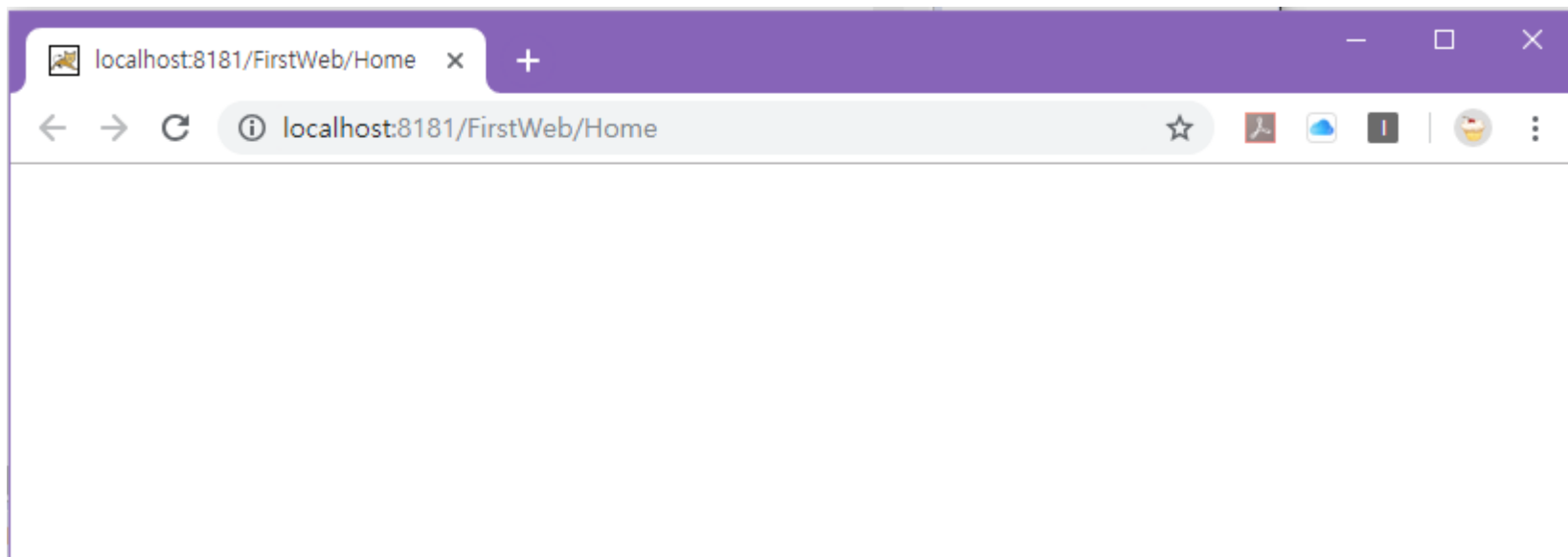
# Servlet 생성



# Servlet 생성

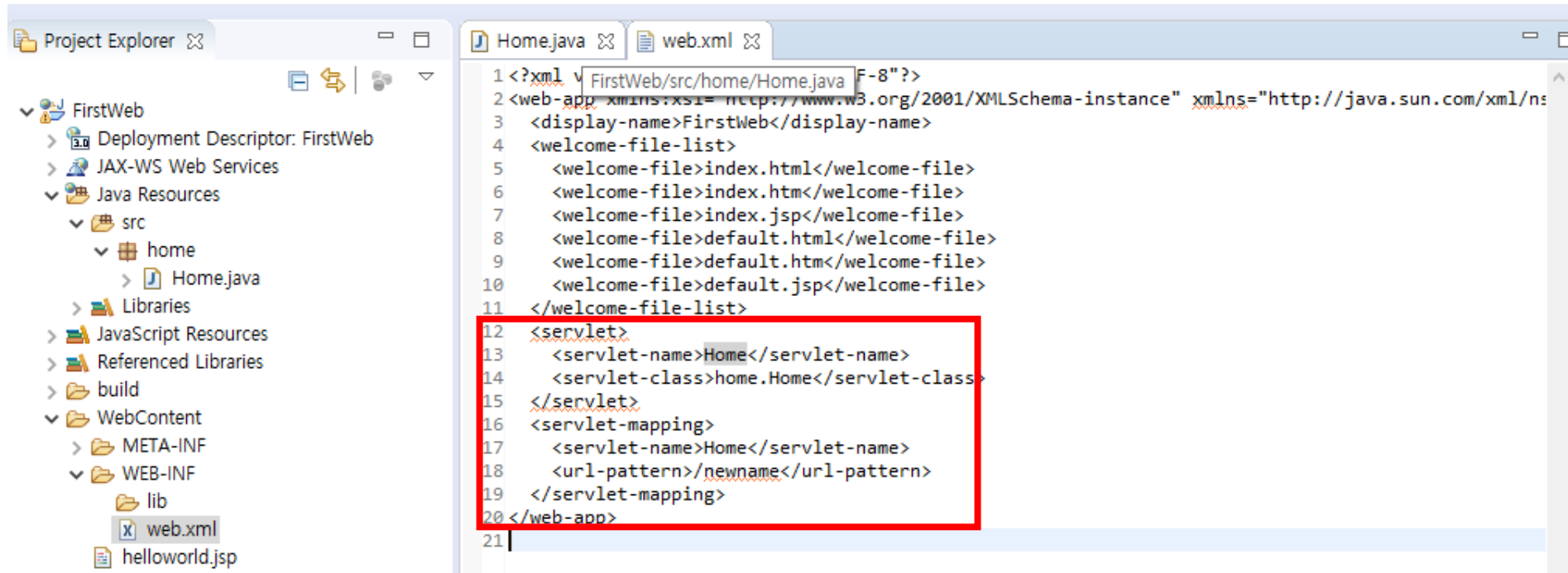
```
*Home.java
1 package home;
2
3 import java.io.IOException;
4
5
6 /**
7  * Servlet implementation class Home
8  */
9
10 @WebServlet("/Home")
11
12 public class Home extends HttpServlet {
13     private static final long serialVersionUID = 1L;
14
15     /**
16      * @see HttpServlet#HttpServlet()
17      */
18     public Home() {
19         super();
20         // TODO Auto-generated constructor stub
21     }
22
23     /**
24      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
25      */
26     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
27         // TODO Auto-generated method stub
28         response.getWriter().append("Served at: ").append(request.getContextPath());
29     }
30
31     /**
32      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
33      */
34     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
35         // TODO Auto-generated method stub
36         doGet(request, response);
37     }
38
39 }
40
41
42 }
```

# Servlet 실행



# URL Mapping

✓ URL 매핑을 하지 않으면 URL 주소가 너무 길어지고, 경로가 노출되어 보안에 위험 생김



# GET / POST 방식 브라우저 한글 처리

## ➤ GET

- server.xml 파일 수정
- <connector> 에 속성 값으로 URLEncoder="utf-8"

▼ Servers

▼ Tomcat v9.0 Server at

catalina.policy

catalina.properties

context.xml

63

<Connector connectionTimeout="20000" port="8181" protocol="HTTP/1.1" redirectPort="8443" URLEncoder="utf-8"/>

server.xml

tomcat-users.xml

web.xml

## ➤ POST

- post 방식 처리하는 메소드에 request.setCharacterEncoding("utf-8");

```
54 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
55  
56     request.setCharacterEncoding("utf-8"); //서버에서 한글 처리
```

# Servlet

# HTTP Request Methods

## ➤ GET

- 클라이언트가 서버에게 URL이 가리키는 웹 문서의 내용 전송하도록 요청
- 데이터(parameter)가 URL 주소(query string)에 묻어서 감  
ex) `www.mypage.com/index.jsp?id=abc1234&pw=123123`
- 전송했던 데이터는 브라우저의 히스토리에 접속했던 주소와 함께 남아 있음 => 보안성에 취약
- 서버의 정보 가지고 올 때 사용 (ex: 게시판 글 조회, 검색)
- 전송할 수 있는 최대 크기 정해져 있음

## ➤ POST

- 클라이언트가 서버에게 데이터를 전송하도록 함
- 전송되는 데이터가 URL에 묻어 나가지 않고, 전송 객체의 Body 통해 전달됨
- 브라우저에 전달되는 데이터가 남지 않음 ⇒ 보안성 강함
- private한 데이터를 서버에 전송할 때 주로 사용 (ex: 비밀번호, 주민등록번호)
- 데이터 양 제한 없음 ie. 대량 전송 가능



# Servlet

## ➤ HttpServlet 상속받아 Servlet 클래스 작성

- javax.servlet.http
- public void doXXX (Request / Response) throws ServletException, IOException{...} override

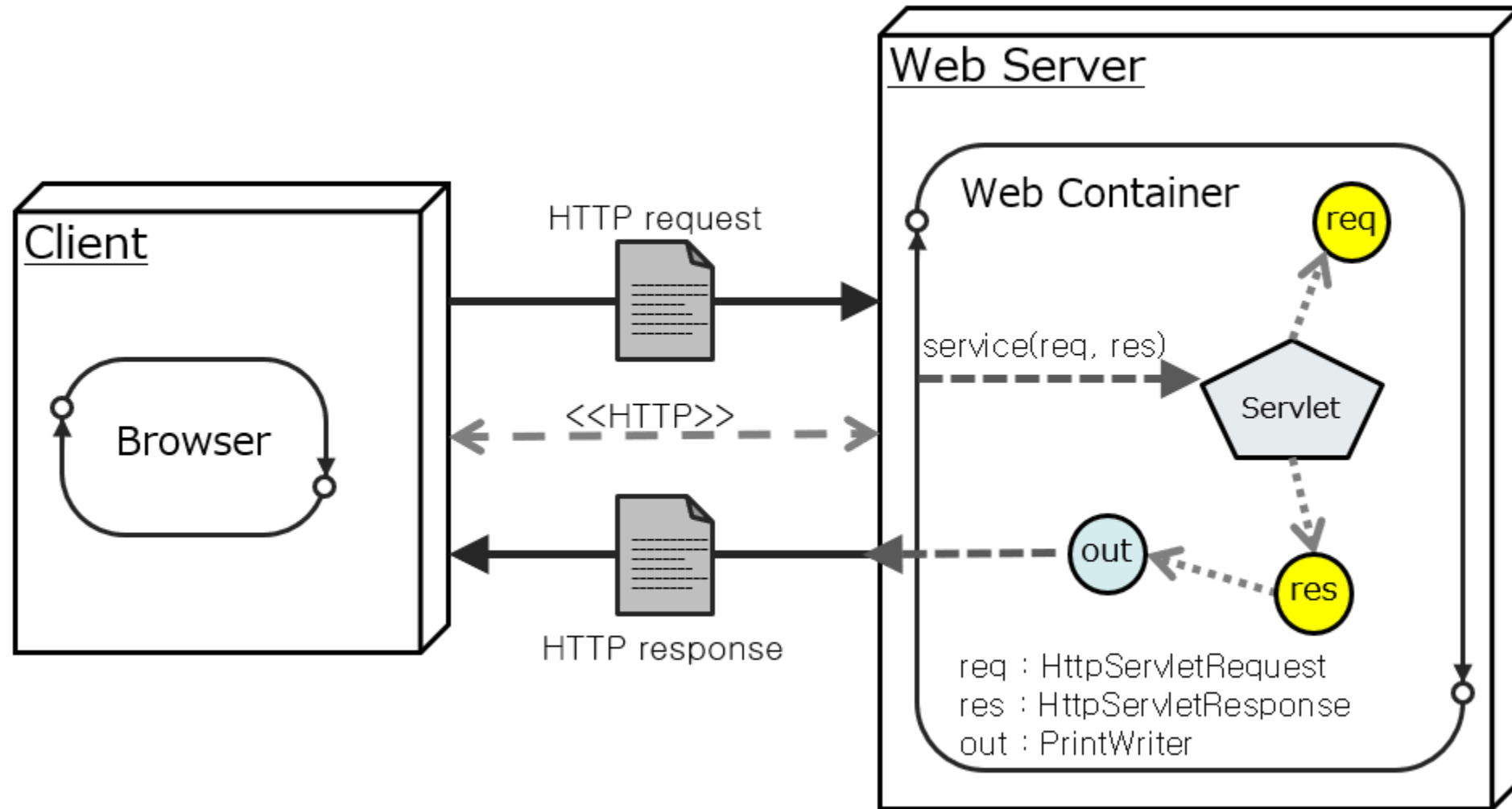
## ➤ URL Mapping

- @WebServlet("매핑하고 싶은 String")

## ➤ HTML 코드 작성 방법

- PrintWriter out = response.getWriter(); 사용하여 HTML 코드 삽입
- cf) JSP는 HTML 코드에 Java 언어 삽입하는 것으로, 이때 특정 JSP 태그 사용 필요

# Servlet HTTP Request & Response



# HTML Form Tag

- ✓ 데이터를 서버로 전송하기 위하여 사용
- ✓ `<form action="url" method="post/get"></form>`
  - action : 폼 전송할 url 주소
  - method : get/post

```
18 <form action = "" method = "">
19     NAME : <input type = "text"> <br>
20     ID : <input type = "text"> <br>
21     PW : <input type = "password"> <br>
22     PW CHECK : <input type = "password"> <br>
23     INTRO : <textarea rows = "5" cols = "50"></textarea><br>
24     <!-- &nbsp; 는 스페이스바 기능 수행 -->
25     SEX : <input type = "radio" name = "gender"> M &nbsp; &nbsp; &nbsp;
26     <input type = "radio", name = "gender"> F <br>
27     AREA : <select>
28         <option>SEOUL</option>
29         <option>BUSAN</option>
30         <option>JEJU</option>
31         <option>OTHERS</option>
32     </select> <br>
33     HOBBY : <input type = "checkbox" name = "hobby"> BOOKS
34         <input type = "checkbox" name = "hobby"> MUSIC
35         <input type = "checkbox" name = "hobby"> MOVIES
36         <input type = "checkbox" name = "hobby"> OTHERS <br>
37     <!-- 버튼 안에 들어갈 문자는 value 속성에 적음 -->
38     <input type = "submit" value = "OK">
39     <input type = "reset" value = "CANCEL">
```

## HttpServletRequest 주요 method

---

- **request.getParameter(String name)**
  - form의 name에 해당하는 parameter의 값을 return
  - name 값은 요소의 name 속성과 값이 일치해야 함
- **request.getParameterValues(String name)**
  - name에 해당하는 parameter의 값을 return
  - 여러 값을 가지고 있을 경우 사용 (ex: form checkbox, list, radio)

## HttpServletResponse 주요 method

---

- **response.setContentType("text/html;charset=euc-kr");**
  - 콘텐츠 타입 지정 및 문자 인코딩 지정
- **response.sendRedirect("url")**
  - 페이지를 지정한 url로 리다이렉트

실습