

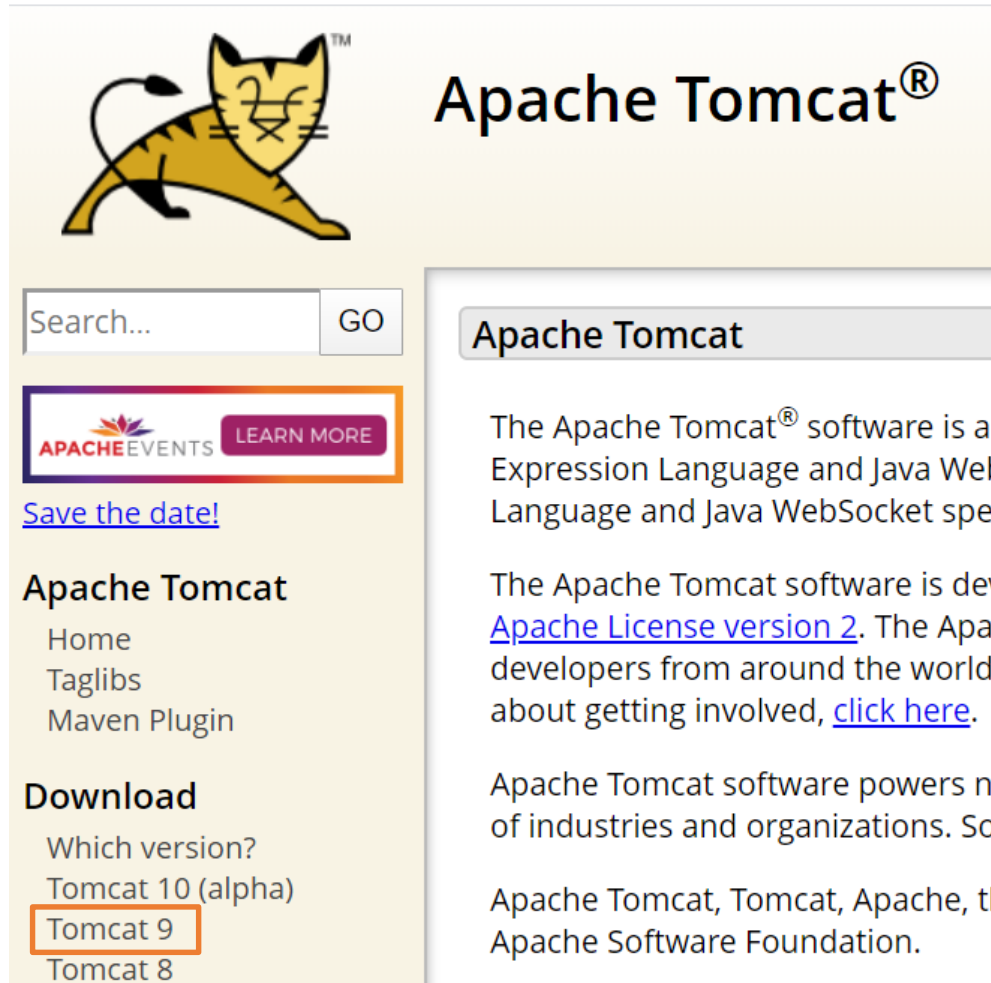
JSP



환경설정

톰캣 다운로드

❖ <https://tomcat.apache.org/>



The screenshot shows the Apache Tomcat homepage. At the top left is the Tomcat logo, a stylized orange and black cat. To its right is the text "Apache Tomcat®". Below the logo is a search bar with the text "Search..." and a "GO" button. Under the search bar is a banner for "APACHE EVENTS" with a "LEARN MORE" button. Below the banner is a link "Save the date!". To the left of the main content area, there is a sidebar with the heading "Apache Tomcat" and links for "Home", "Taglibs", and "Maven Plugin". Below this is a "Download" section with the text "Which version?" and a list of versions: "Tomcat 10 (alpha)", "Tomcat 9" (which is highlighted with an orange box), and "Tomcat 8". The main content area has a heading "Apache Tomcat" and three paragraphs of text. The first paragraph describes the software as an Expression Language and Java Web Language and Java WebSocket specification. The second paragraph states that the software is developed by Apache developers from around the world and provides a link "click here" for getting involved. The third paragraph mentions that the software powers many of industries and organizations. The fourth paragraph lists "Apache Tomcat, Tomcat, Apache, the Apache Software Foundation."

Apache Tomcat®

Search... GO

APACHE EVENTS LEARN MORE

[Save the date!](#)

Apache Tomcat

Home
Taglibs
Maven Plugin

Download

Which version?

Tomcat 10 (alpha)
Tomcat 9
Tomcat 8

Apache Tomcat

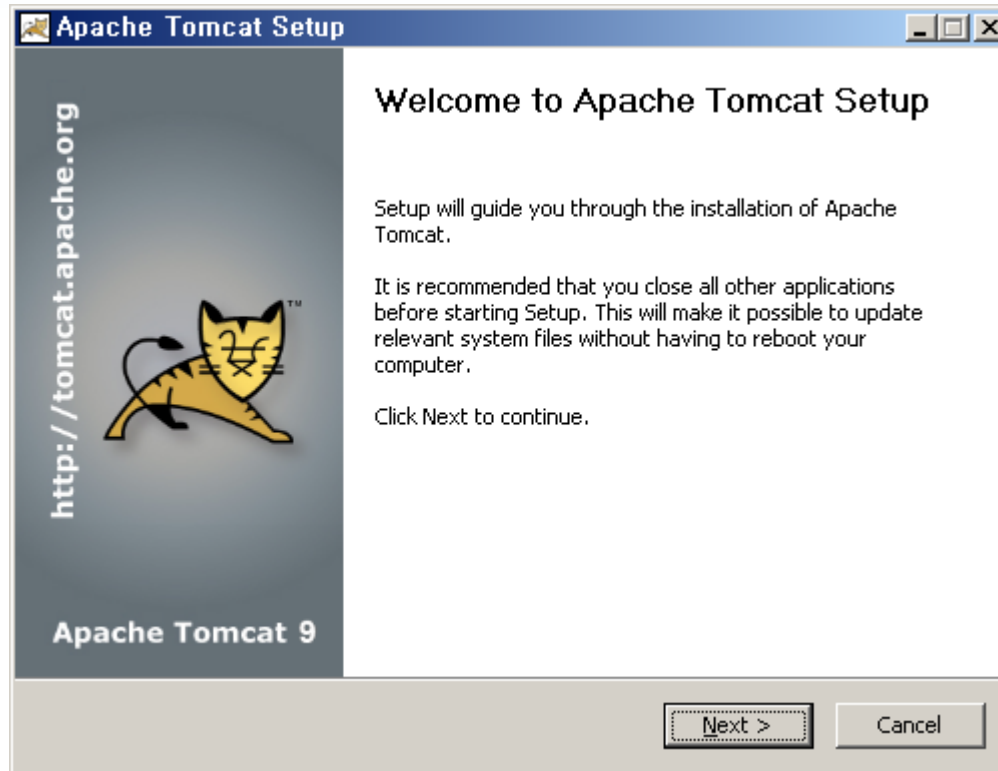
The Apache Tomcat® software is an Expression Language and Java Web Language and Java WebSocket specification.

The Apache Tomcat software is developed by Apache developers from around the world. about getting involved, [click here](#).

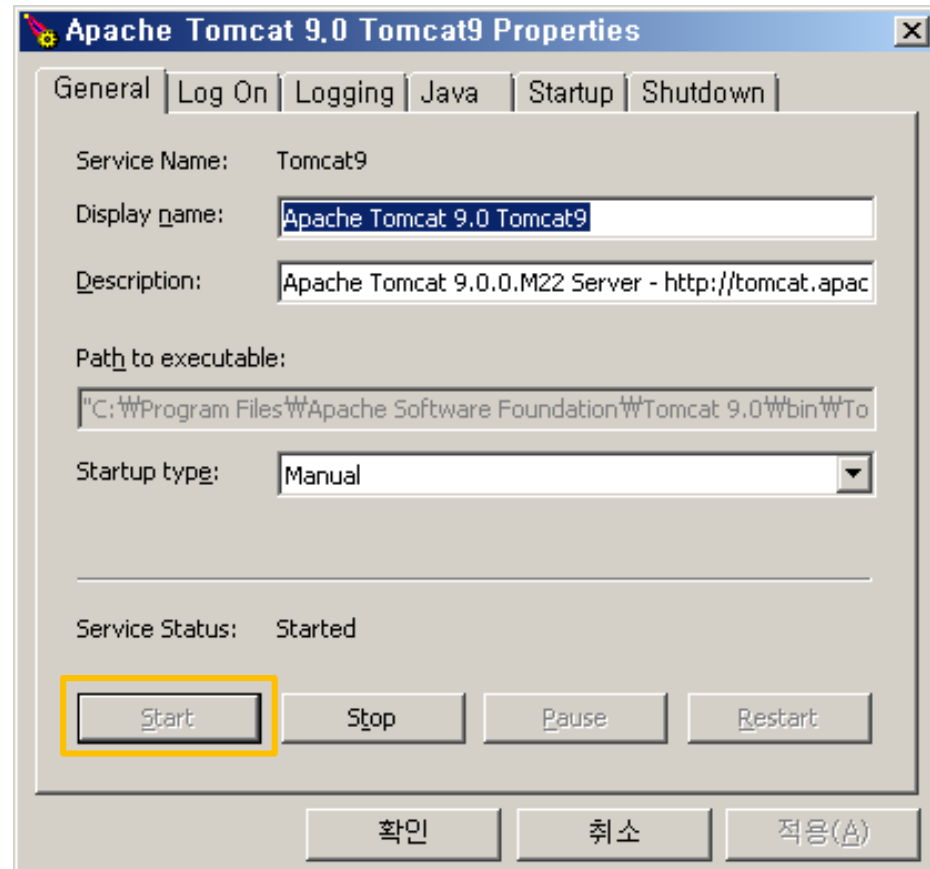
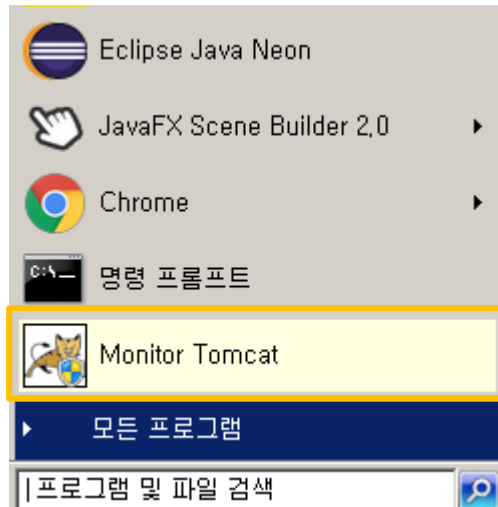
Apache Tomcat software powers many of industries and organizations. So

Apache Tomcat, Tomcat, Apache, the Apache Software Foundation.

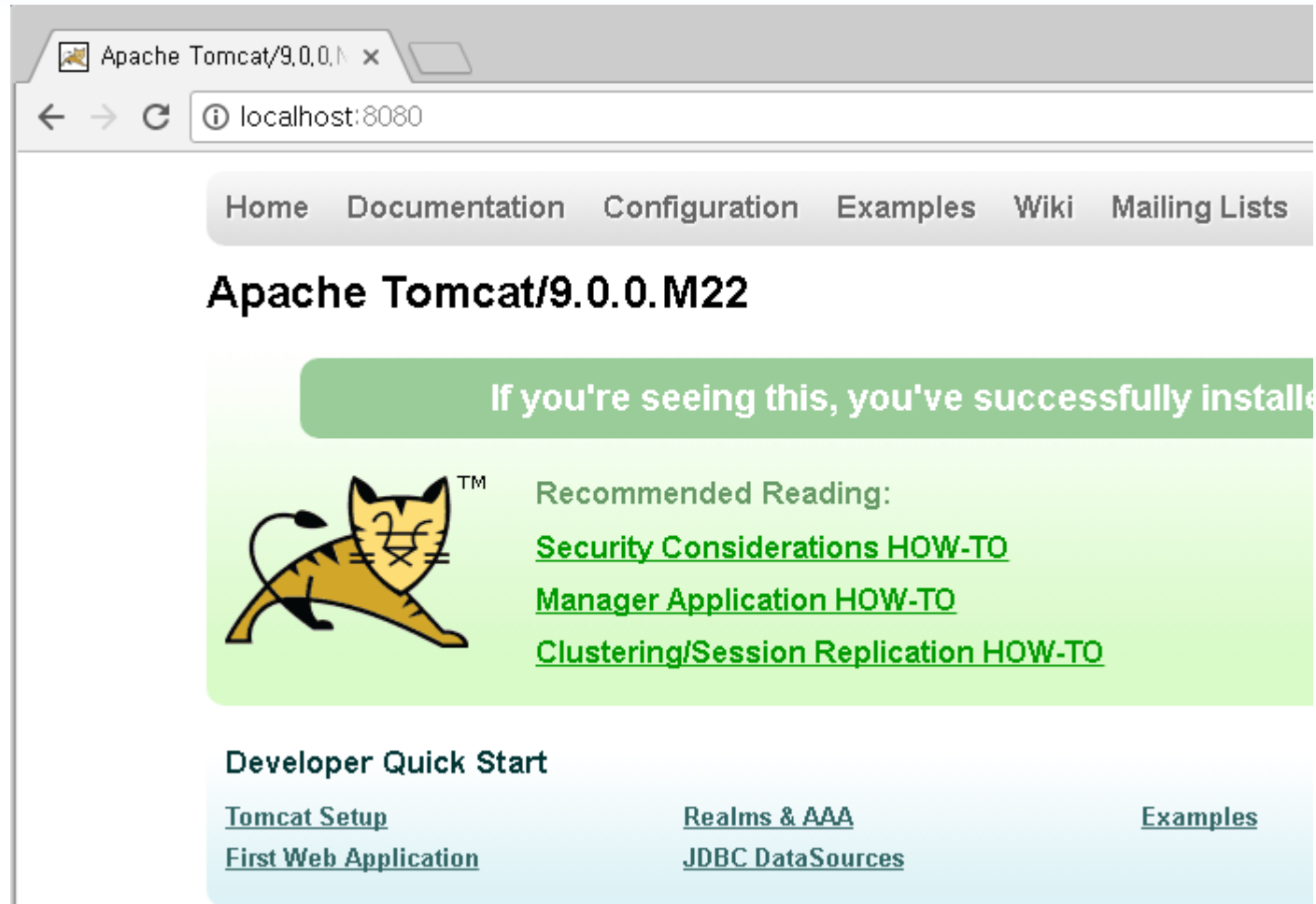
톰캣 설치



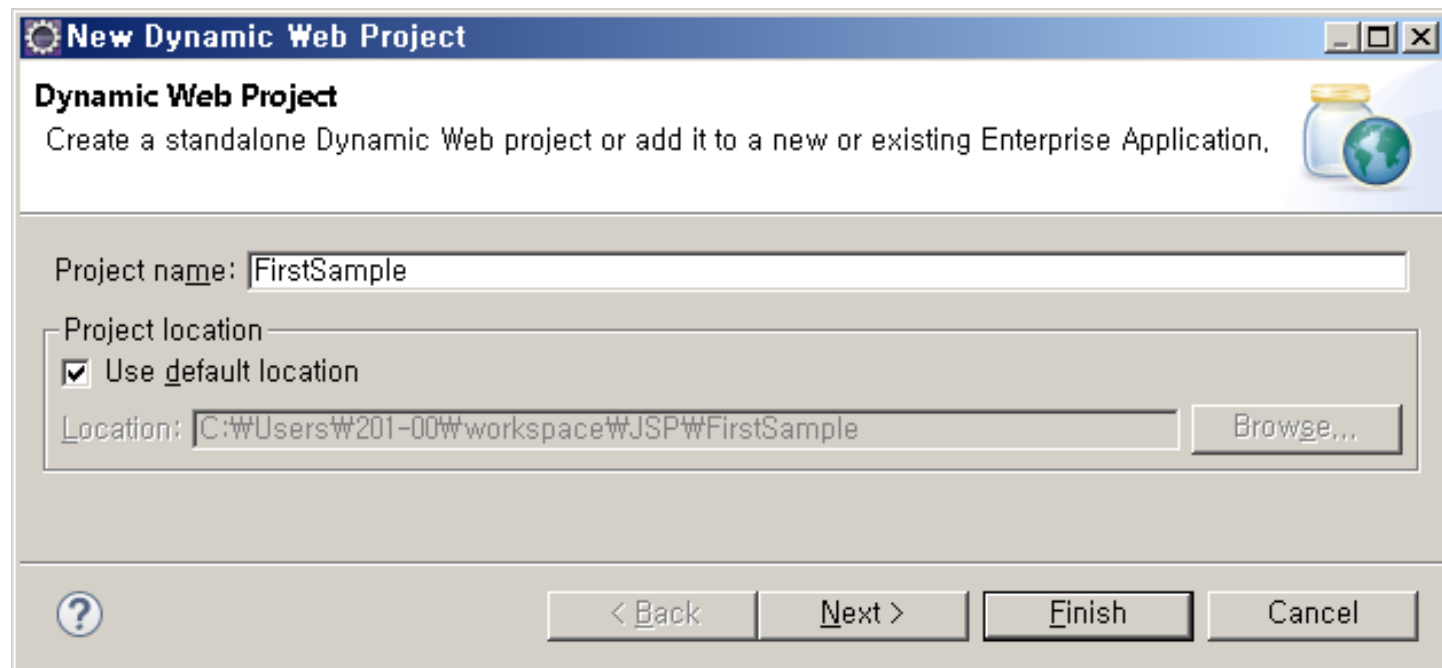
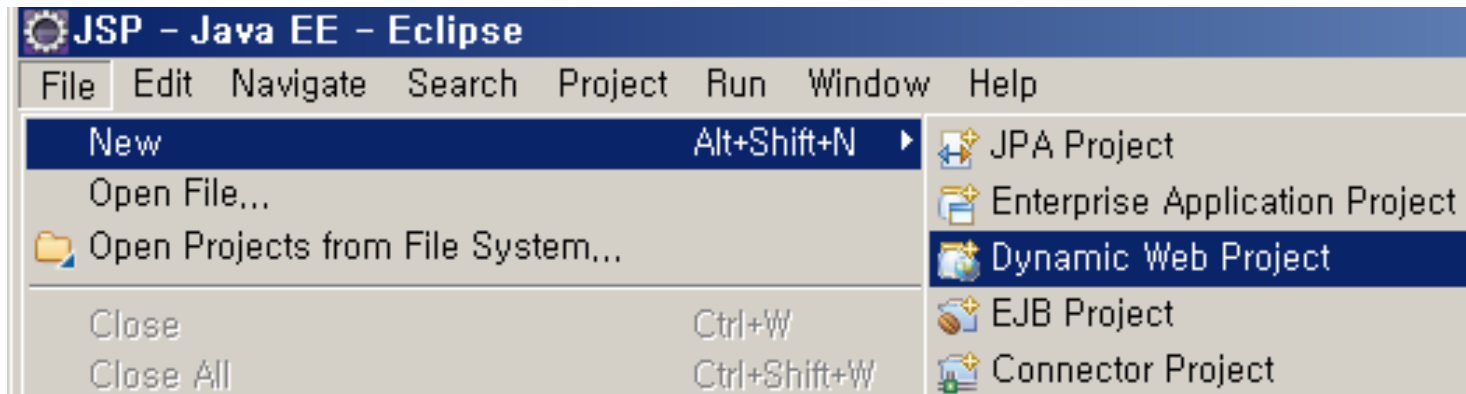
톰캣 실행



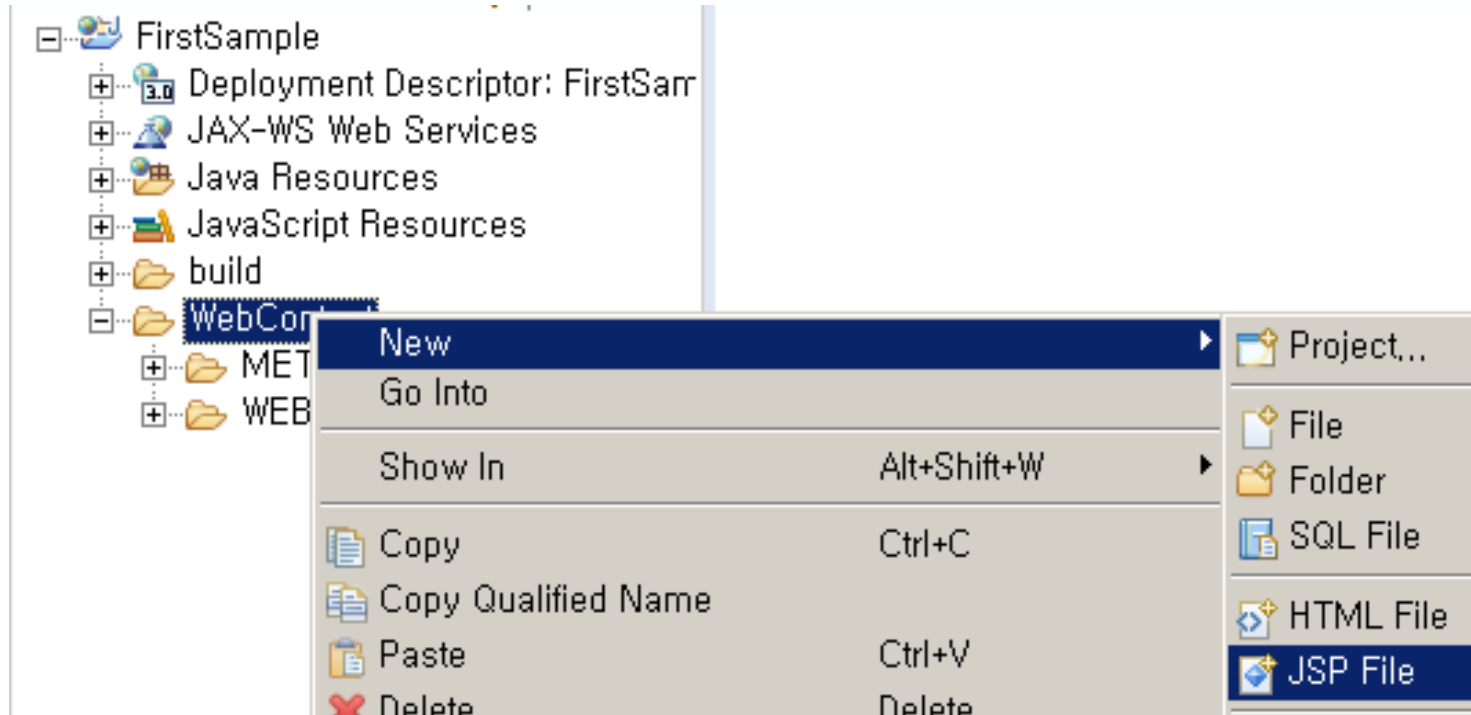
접속 확인 – localhost:8080



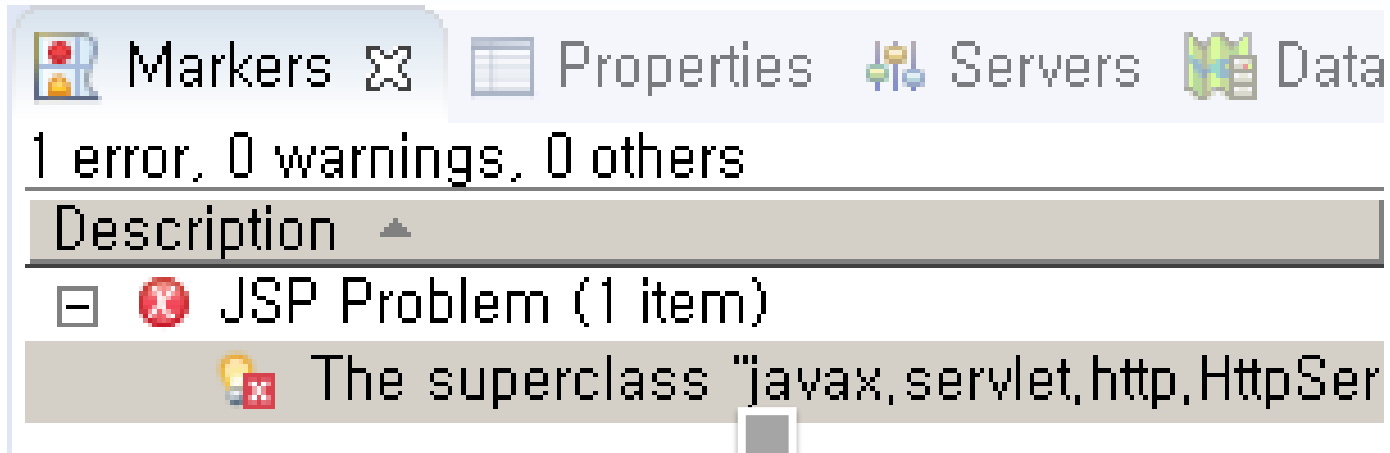
Hello JSP



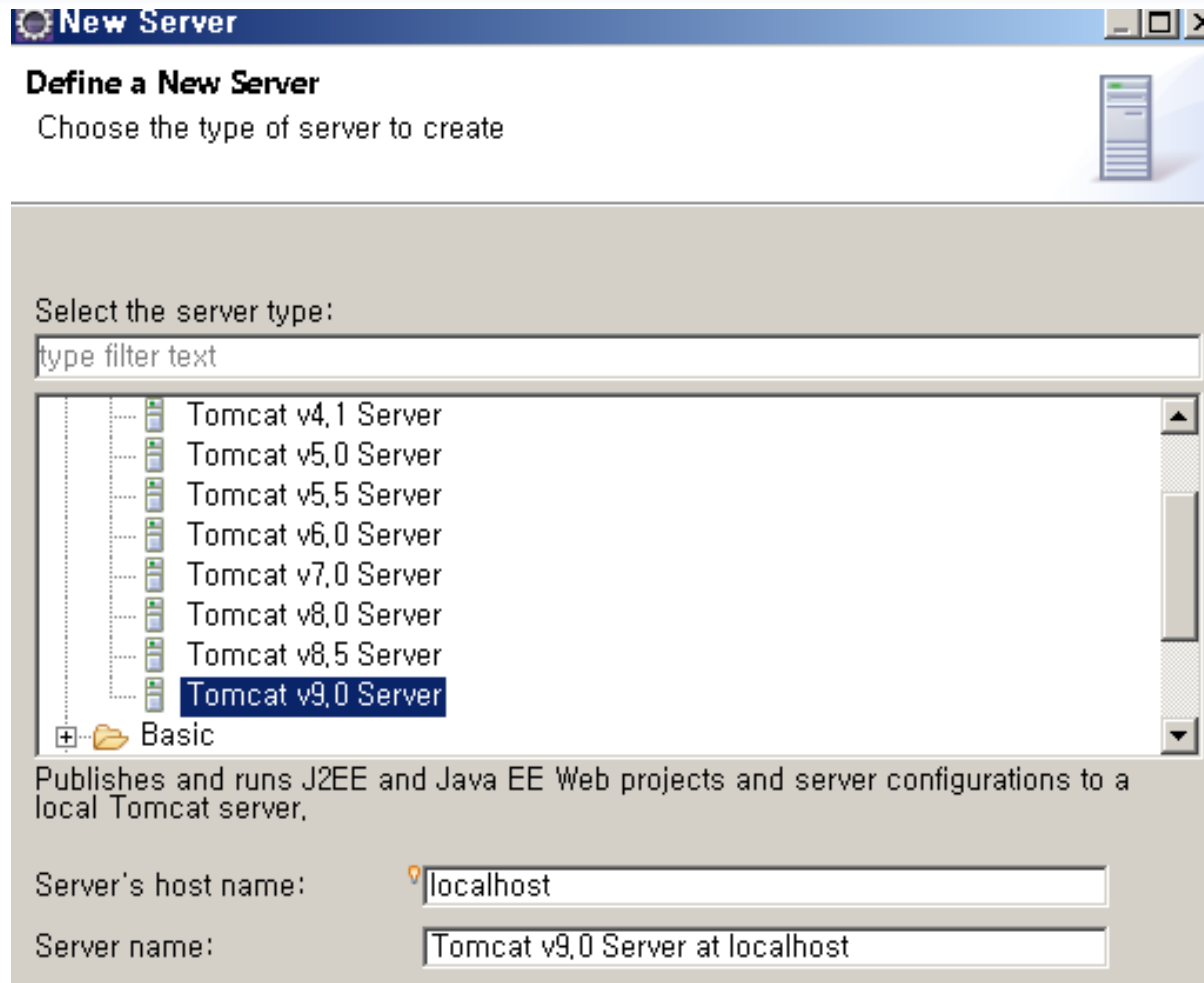
Hello JSP



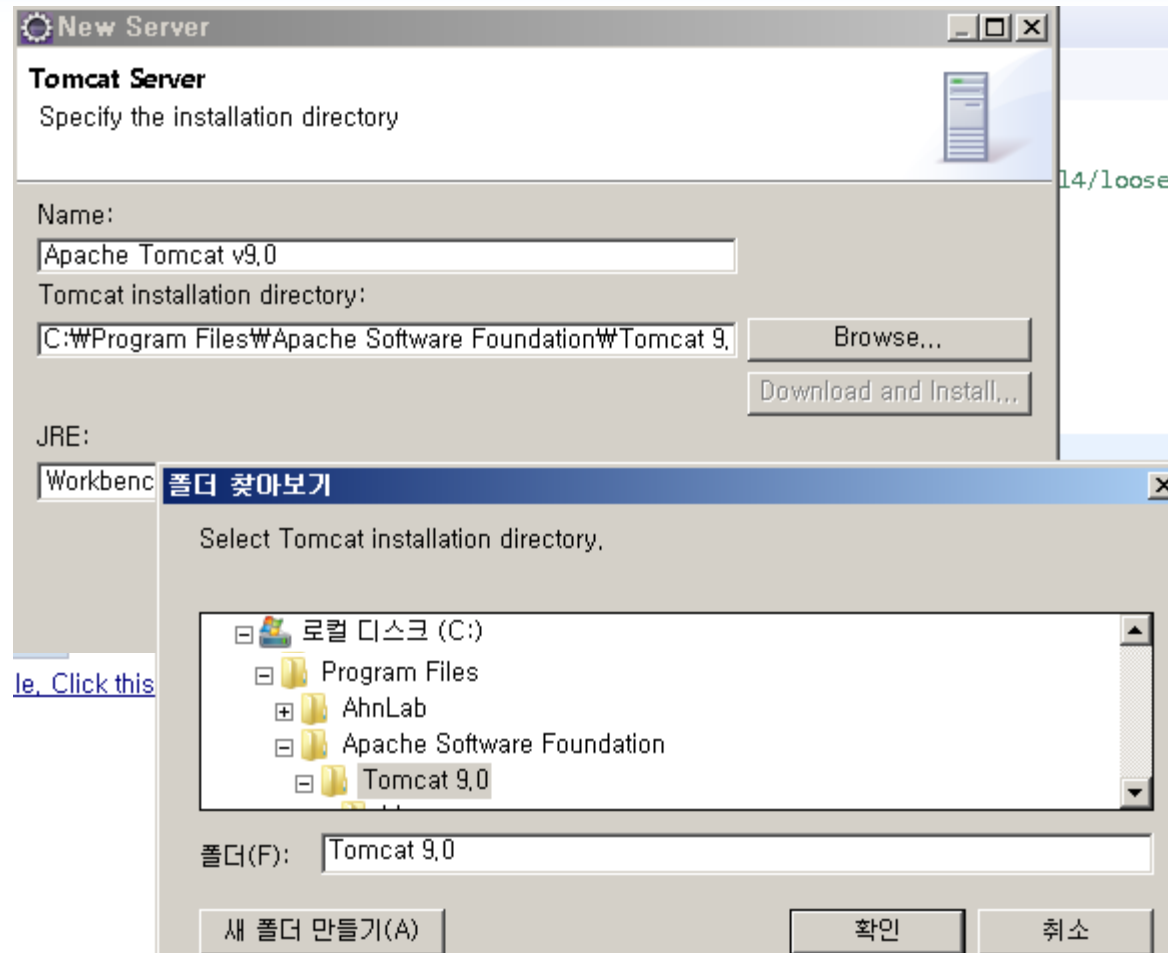
Hello JSP



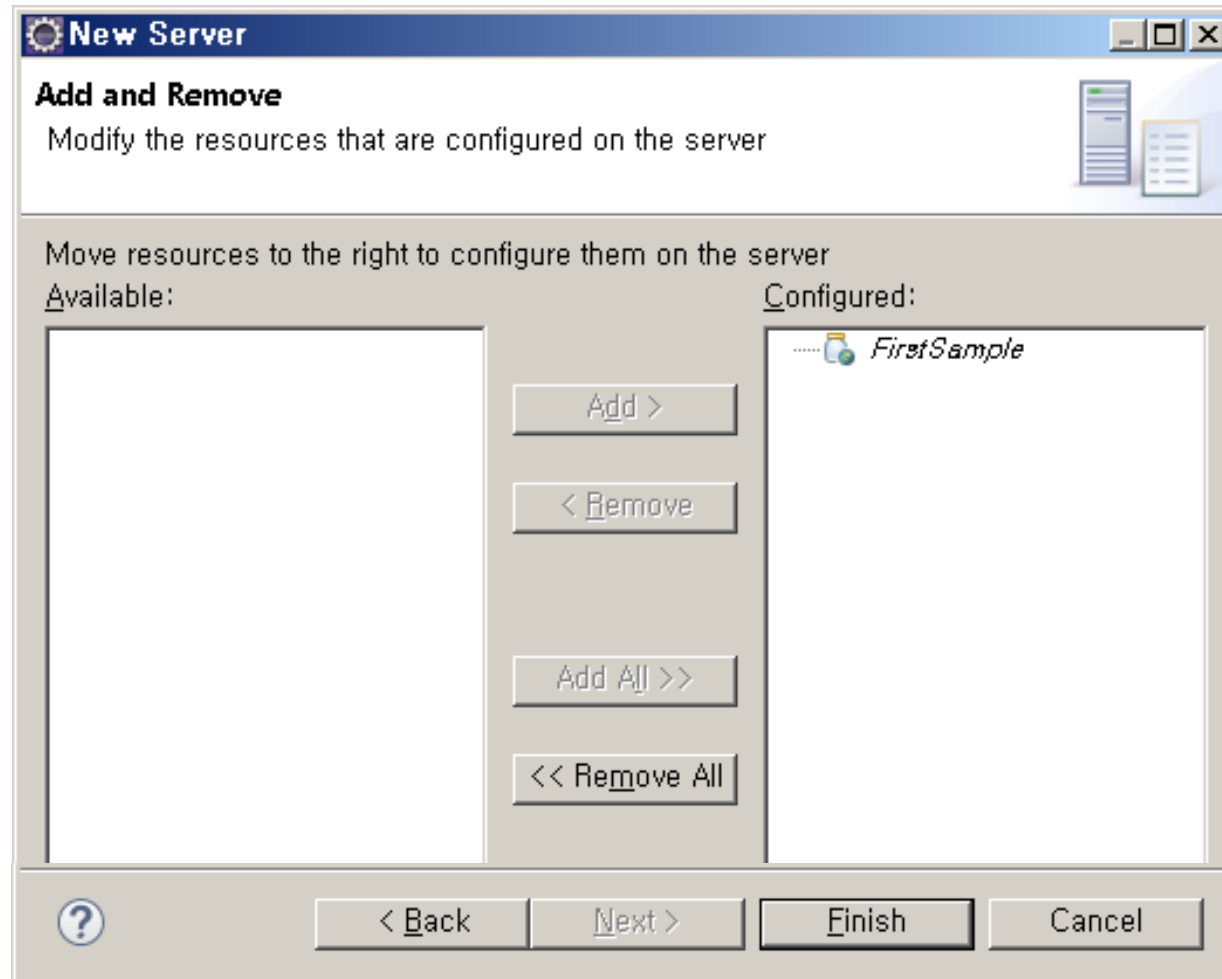
톰캣 설정



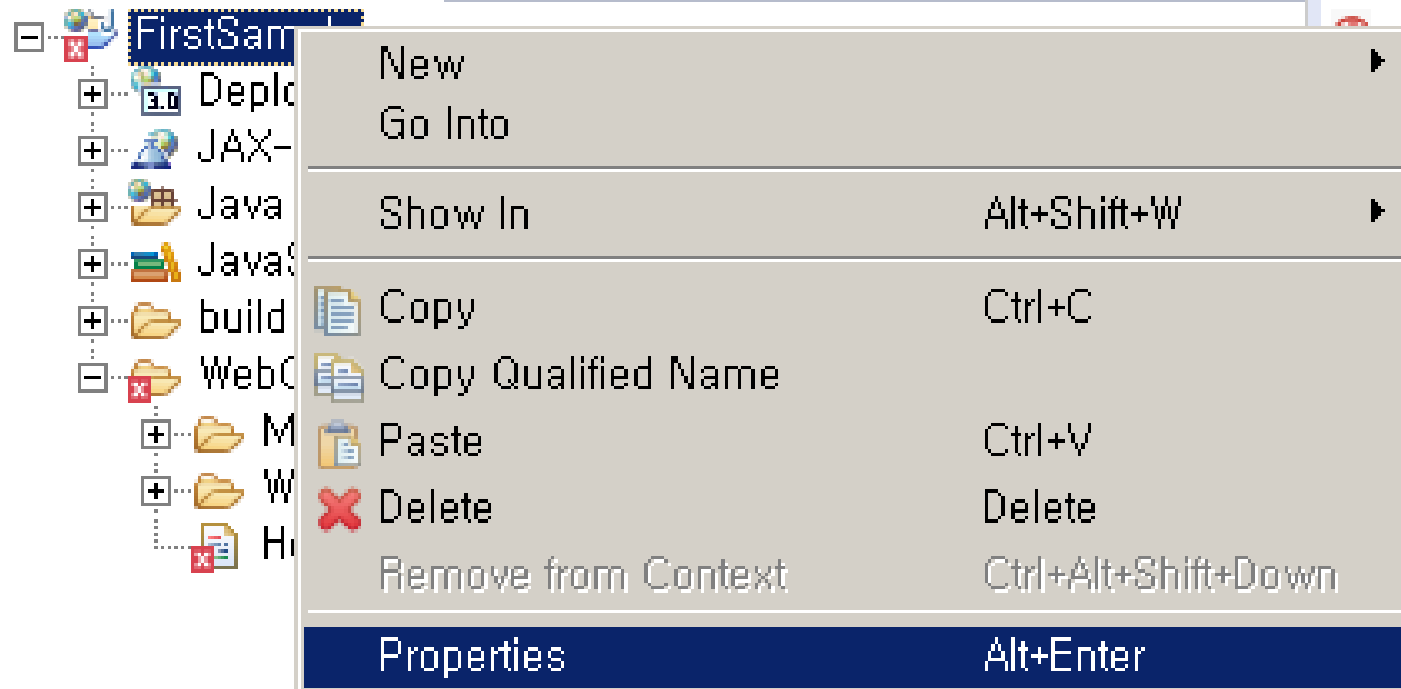
톰캣 설정



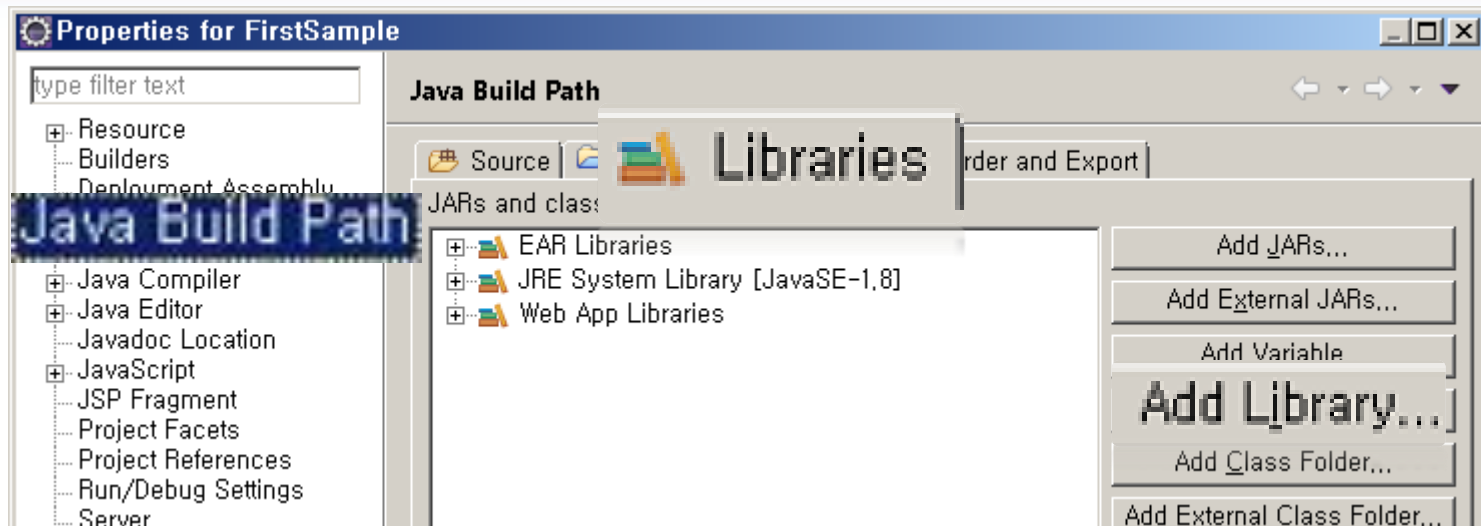
톰캣 설정



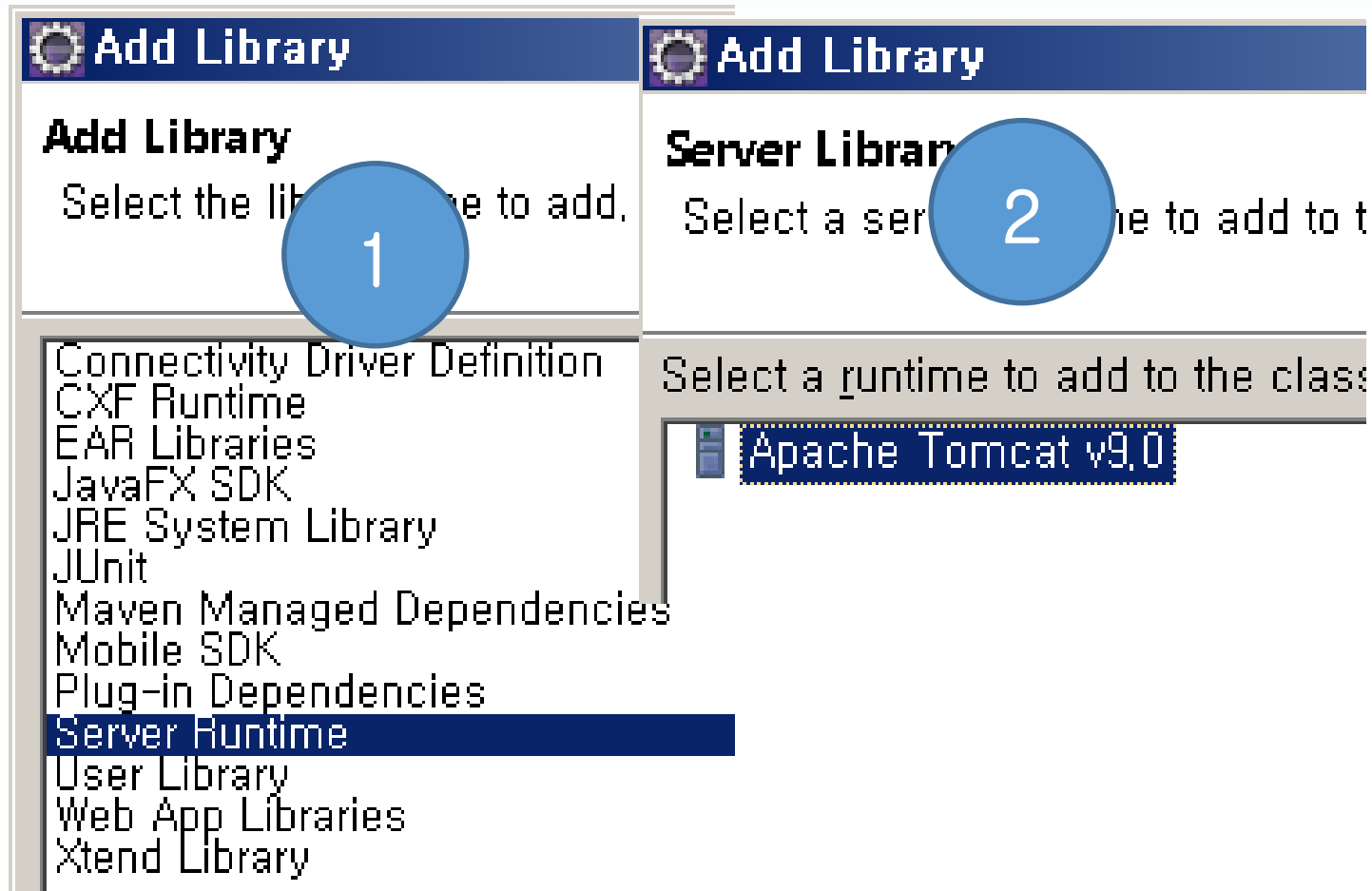
프로젝트 환경 설정



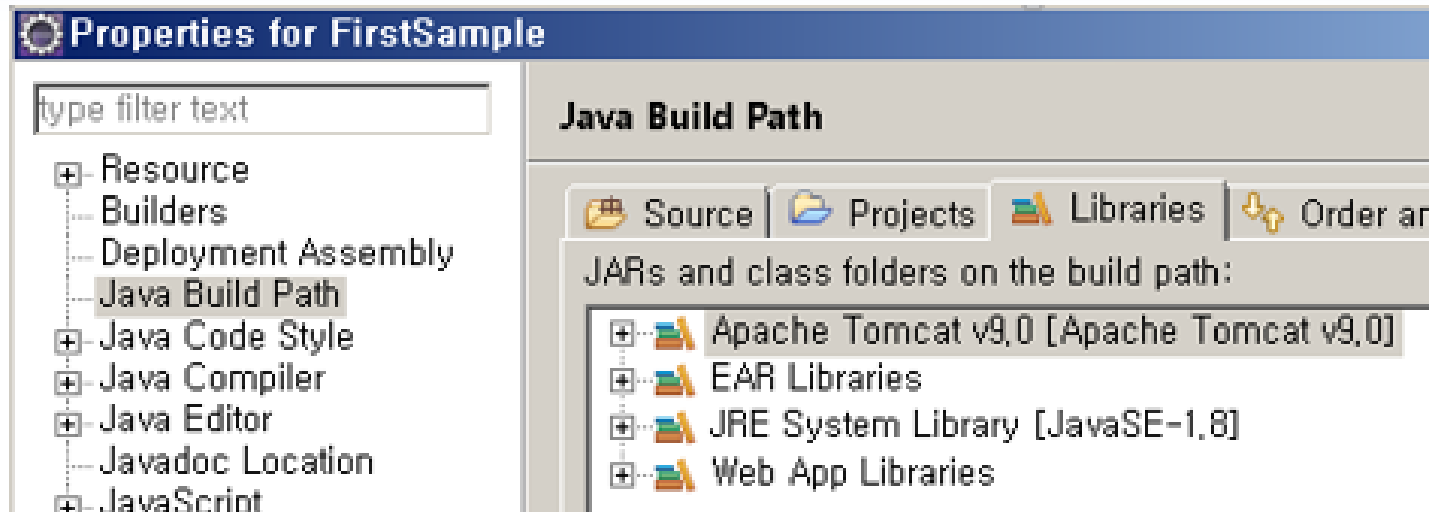
프로젝트 환경 설정



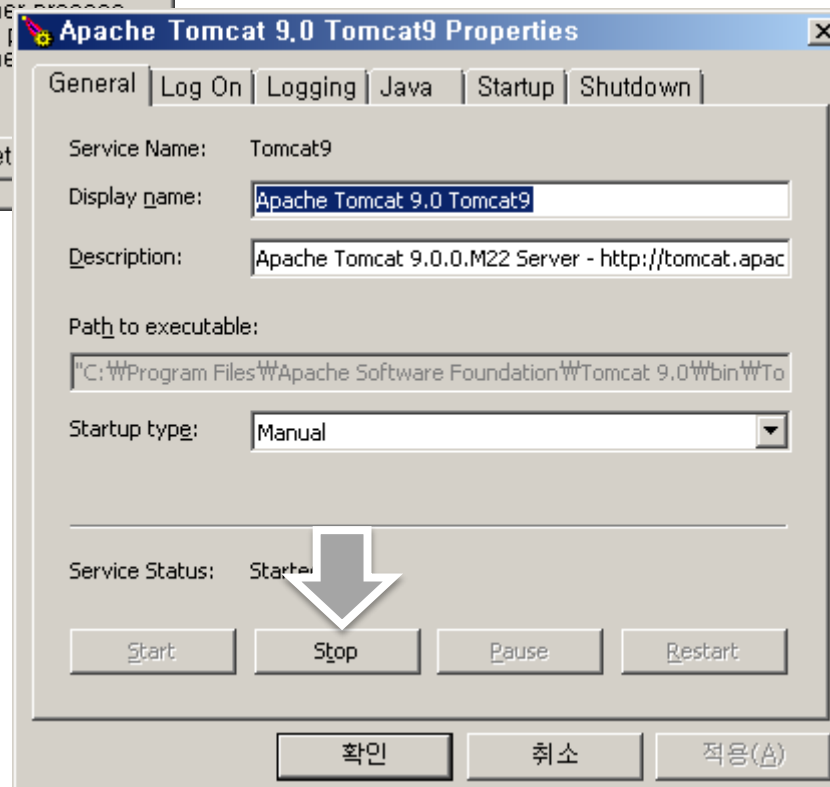
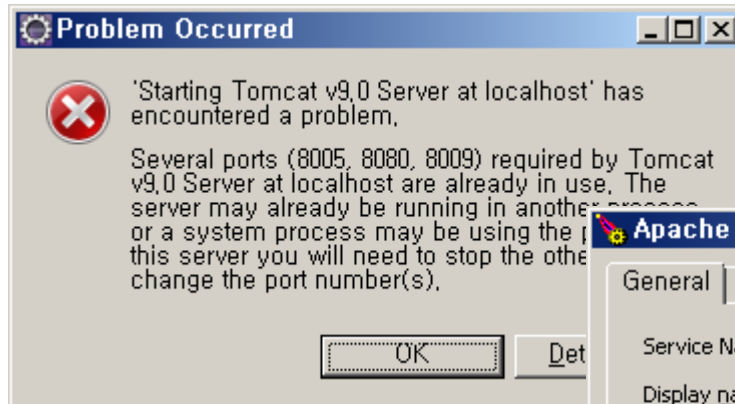
프로젝트 환경 설정



프로젝트 환경 설정



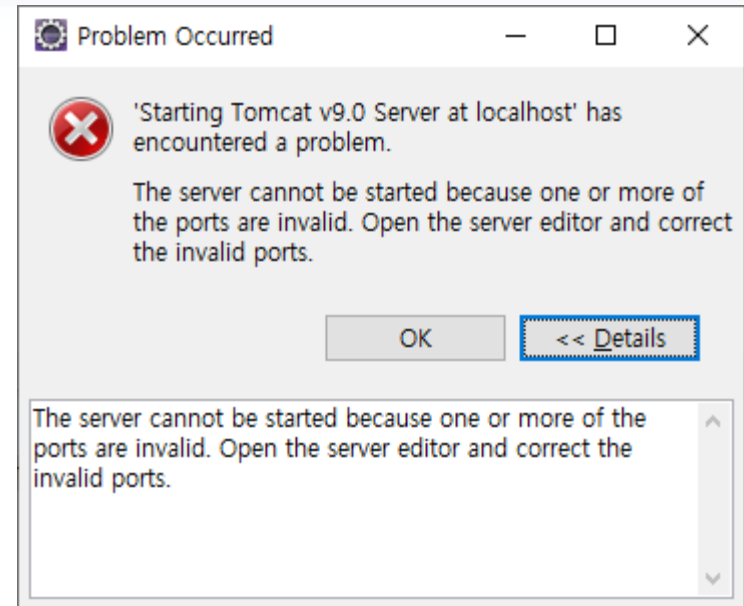
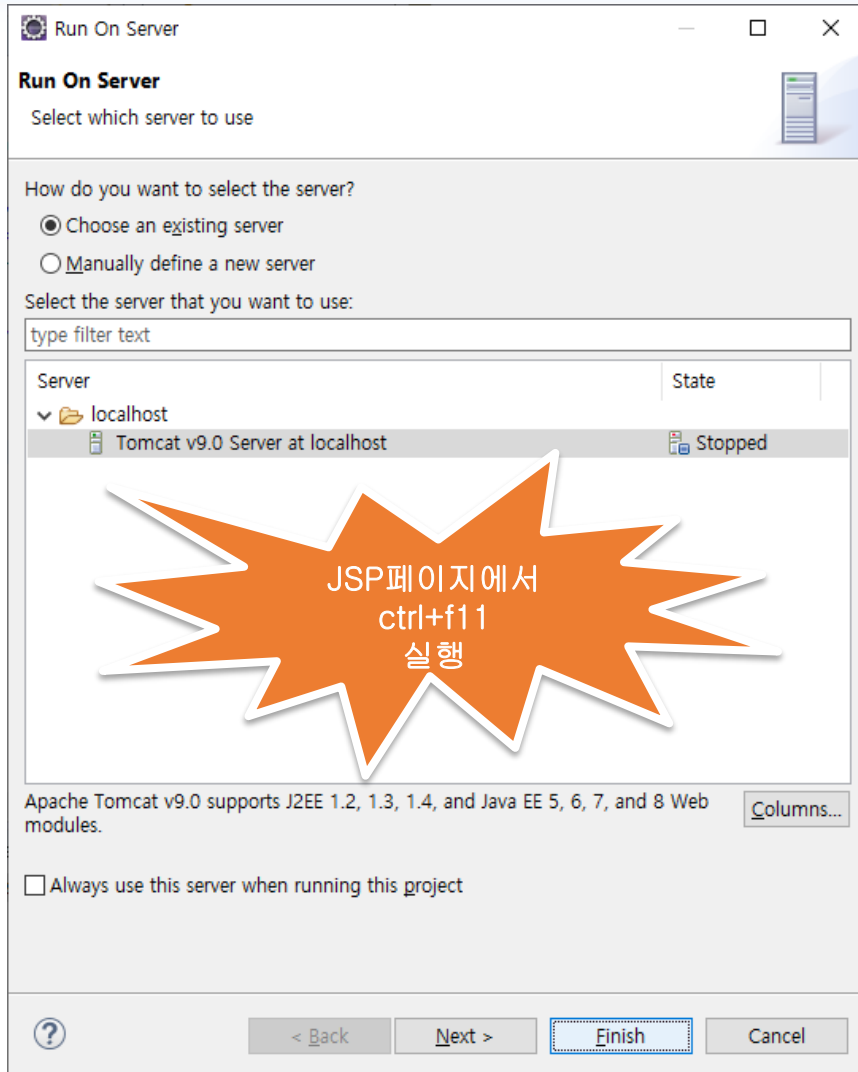
Error 발생



실습

```
<%@ page language="java" conten
    pageEncoding="EUC-KR"%>
<!DOCTYPE html PUBLIC "-//W3C//
<html>
<head>
<meta http-equiv="Content-Type"
<title>Insert title here</title
</head>
<body>
<h1>Hello JSP</h1>
</body>
</html>
```

에러발생



서버 포트 변경

Ex01_language.jsp Tomcat v9.0 Server at localhost

▶ Publishing

▶ Timeouts

▼ Ports

Modify the server ports.

Port Name	Port Number
Tomcat admin port	-
HTTP/1.1	8080

Overview Modules

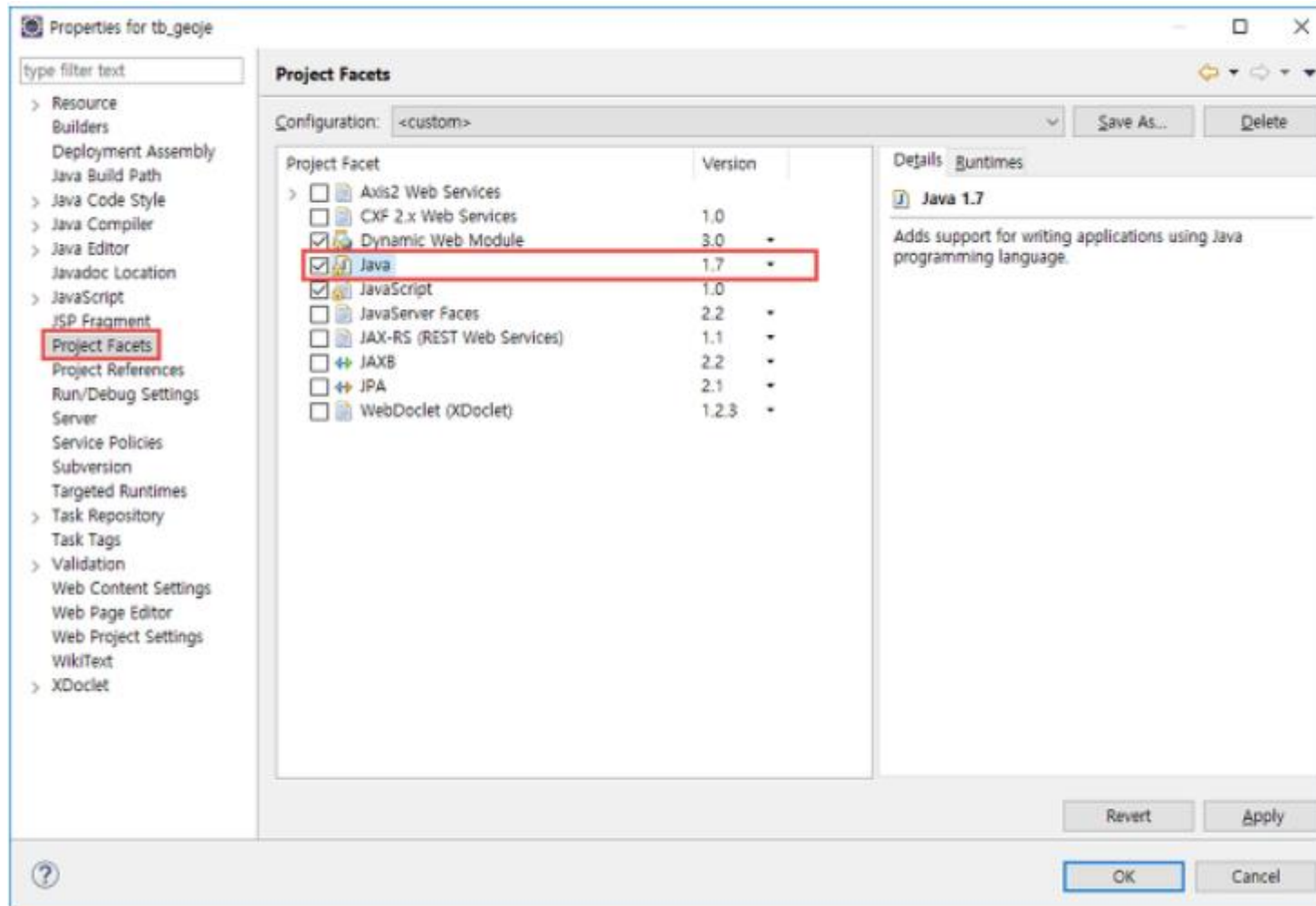
Markers Properties Servers Data Source Explorer Snippets

> Tomcat v9.0 Server at localhost [Stopped, Synchronized]

Tomcat admin
port번호를
8005로 변경 후
저장

프로젝트 버전 문제로 안될 경우

- ❖ 프로젝트 선택 후 마우스 우클릭 --> Properties --> Project Facets --> Java 버전 확인



기초 문법

디렉티브(Directive)

❖ JSP 페이지에 대한 설정 정보를 지정

❖ 디렉티브 구문

- `<%@ 디렉티브이름 속성1="값1" 속성2="값2" ... %>`
- 예, `<%@ page contentType = "text/html; charset=euc-kr" %>`

❖ 제공 디렉티브

- page : JSP 페이지에 대한 정보를 지정
 - 문서의 타입, 출력 버퍼의 크기, 에러 페이지 등 정보 지정
- taglib : 사용할 태그 라이브러리를 지정
- include : 다른 문서를 포함

page 디렉티브

❖ JSP 페이지에 대한 정보를 입력

- JSP가 생성할 문서의 타입, 사용할 클래스, 버퍼 여부, 세션 여부

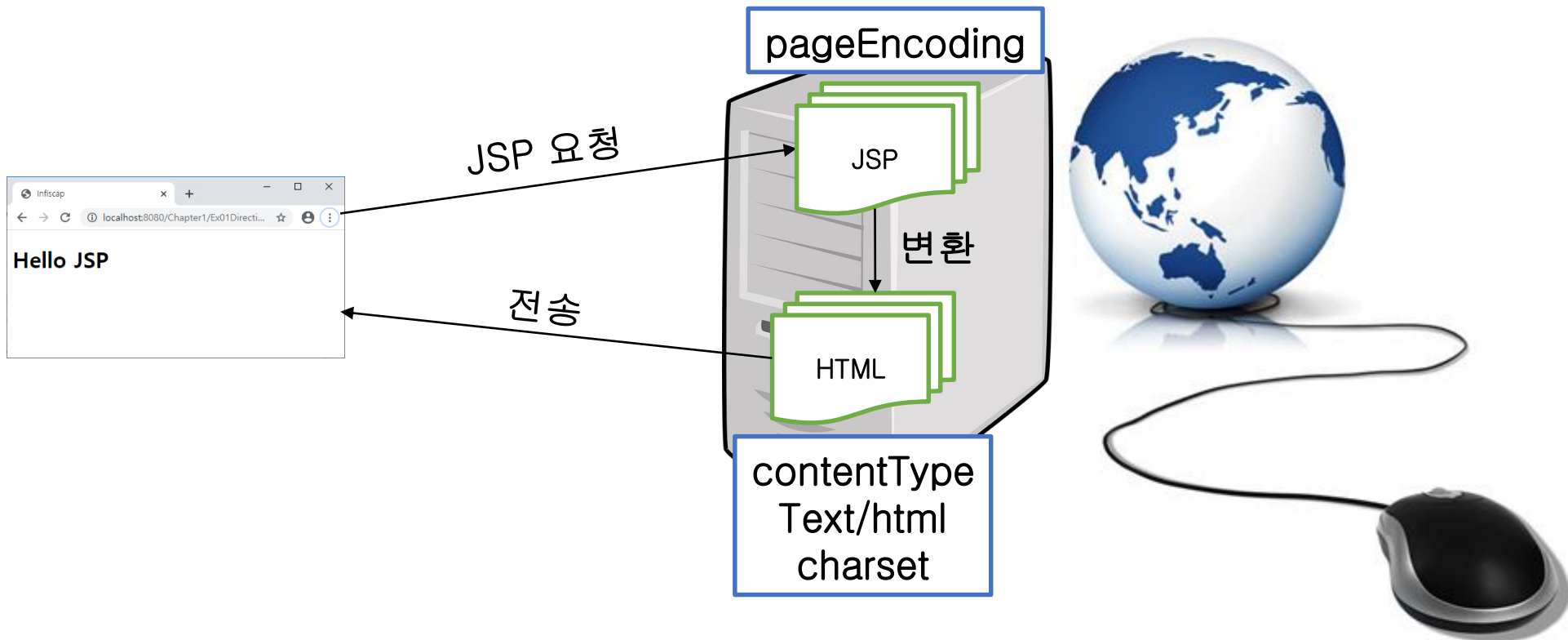
❖ JSP 디렉티브의 작성 예

- `<%@ page contentType="text/html; charset=euc-kr" %>`
- `<%@ page import="java.util.Date" %>`

❖ 주요 속성

- `contentType` : JSP가 생성할 문서의 타입을 지정
- `import` : JSP 페이지에서 사용할 자바 클래스를 지정
- `session` : JSP 페이지가 세션을 사용할 지의 여부를 지정
- `info` : JSP 페이지에 대한 설명을 입력한다.
- `errorPage` : 에러가 발생할 때 보여 줄 페이지를 지정
- `isErrorPage` : 에러 페이지인지의 여부를 지정

실습 – Directive 이해



스크립트 요소

- ❖ 요청을 처리하는 데 필요한 코드를 실행
- ❖ 동적으로 응답 결과를 생성하기 위해 사용
- ❖ 스크립트 요소 세 가지
 - 스크립트릿(Scriptlet)
 - 표현식(Expression)
 - 선언부(Declaration)

스크립트릿(Scriptlet)

- ❖ 자바 코드를 실행할 때 사용되는 코드의 블록
- ❖ 스크립트릿의 구조

```
<%  
    자바코드1;  
    자바코드2;  
    ....  
%>
```

표현식(Expression)

- ❖ 값을 출력 결과에 포함시키고자 할 때 사용
- ❖ 표현식 구문
 - `<%= 값 %>`

선언부(Declaration)

- ❖ 스크립트릿이나 표현식에서 사용할 수 있는 함수를 작성할 때 사용
- ❖ 선언부 형식

```
<%!  
    public 리턴타입 메서드이름(파라미터목록)  
{  
    자바코드1;  
    자바코드2;  
  
    ...  
    자바코드n;  
    return 값;  
}  
%>
```

실습



Quiz

1. 스크립트릿을 이용하여
다음과 같이 나타나도록
코딩하시오

Hello JSP

Hello JSP

Hello JSP

Hello JSP

Hello JSP

Hello JSP



Quiz

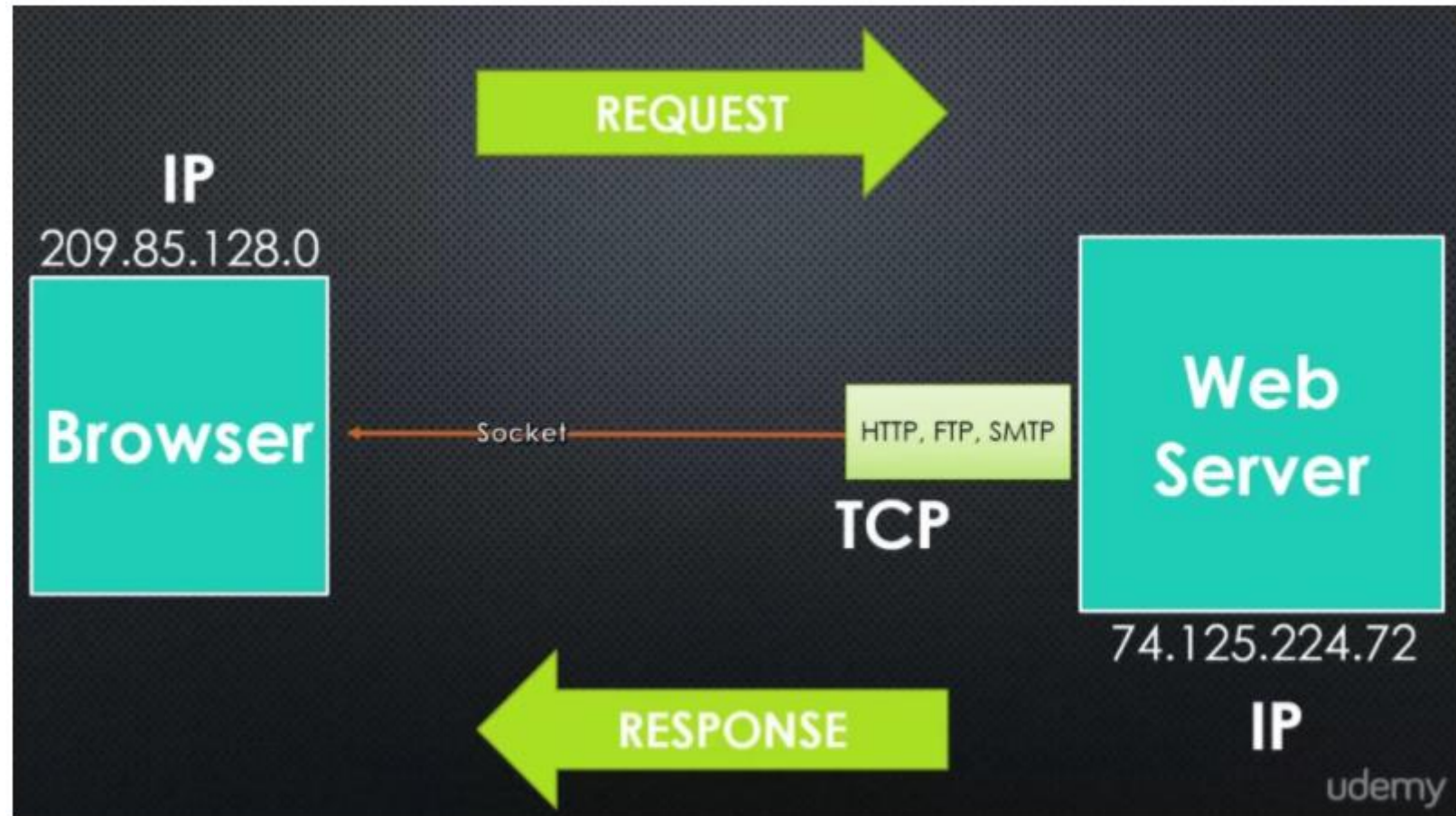
3과 9를 변수에 저장한 후
선언부에 선언되어 있는 메소드를
호출하여 아래와 같이 동작되도록
코딩하시오(Ex3 참조)

$$3 + 9 = 12$$

$$3 - 9 = -6$$



Request VS Response



<출처: Udemv Anthony Alicea - Understanding NodeJS>

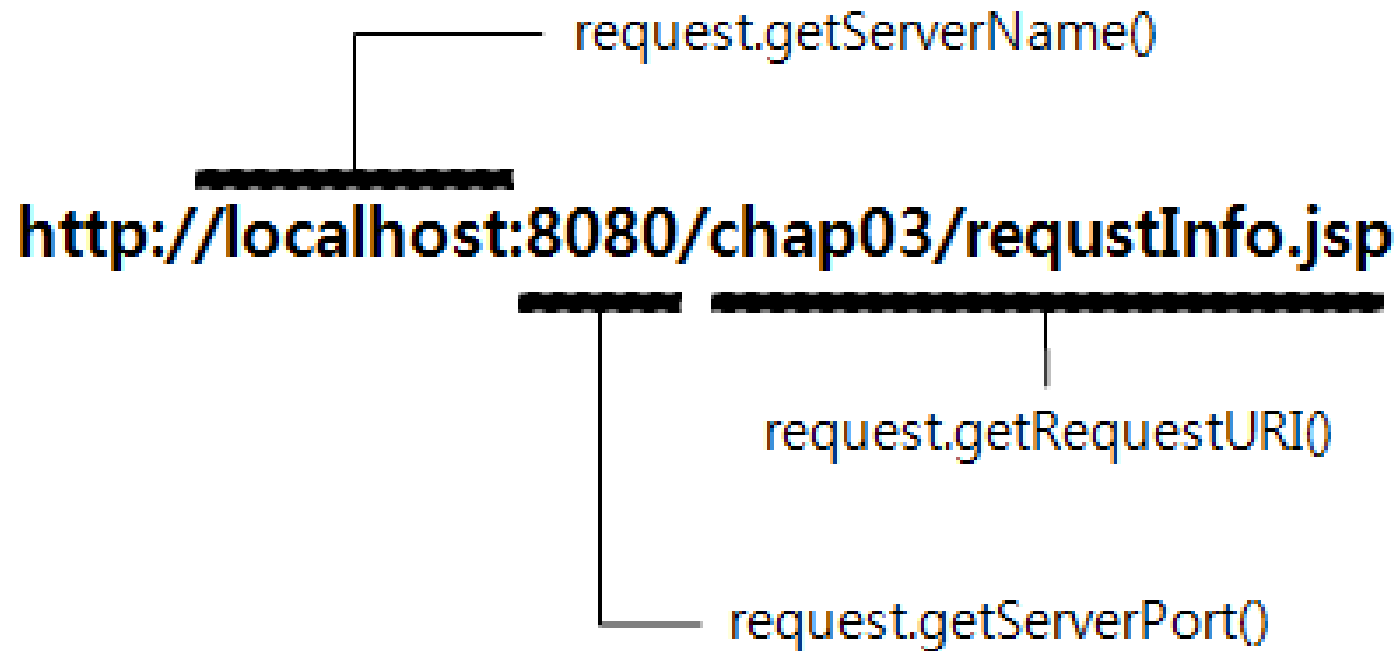
request

❖ 웹 브라우저가 웹 서버에 전송한 요청 관련 정보 제공

❖ 주요 기능

- 클라이언트(웹 브라우저)와 관련된 정보 읽기 기능
- 서버와 관련된 정보 읽기 기능
- 클라이언트가 전송한 요청 파라미터 읽기 기능
- 클라이언트가 전송한 요청 헤더 읽기 기능
- 클라이언트가 전송한 쿠키 읽기 기능
- 속성 처리 기능

request - 주요 정보 제공 메서드



request - 파라미터 읽기 메서드

메서드	리턴타입	설명
getParameter(String args)	String	name이 args인 파라미터의 값을 구한다. 존재하지 않을 경우 null을 리턴한다.
getParameterValues(String args)	String[]	name이 args인 모든 파라미터의 값을 배열로 구한다. 존재하지 않을 경우 null을 리턴한다.
getParameterNames()	java.util.Enumeration	웹 브라우저가 전송한 파라미터의 이름을 구한다.
getParameterMap()	java.util.Map	웹 브라우저가 전송한 파라미터의 맵을 구한다. 맵은 <파라미터 이름, 값> 쌍으로 구성된다.

한글 깨짐 현상(tomcat 설정)

Servers

Tomcat v9.0 Server at localhost-

- catalina.policy
- catalina.properties
- conf.xml
- server.xml
- tomcat-users.xml
- web.xml

```
<Connector
connectionTimeout="20000"
port="8080"
protocol="HTTP/1.1"
redirectPort="8443"
URIEncoding="EUC-KR"/>
```

추가

Find/Replace

Find: port="8080"

Replace with:

Direction: ☒ Forward ☐ Backward

Scope: ☒ All ☐ Selected lines

한글 깨짐 현상(tomcat 설정)

Servers

Tomcat v9.0 Server at localhost-

catalina.policy

catalina

주석 제거

context.xml

server.xml

tomcat-users.xml

web.xml

```
<!--
```

```
<filter>
```

```
<filter-name>setCharacterEnc
```

```
<filter-class>org.apache.cat
```

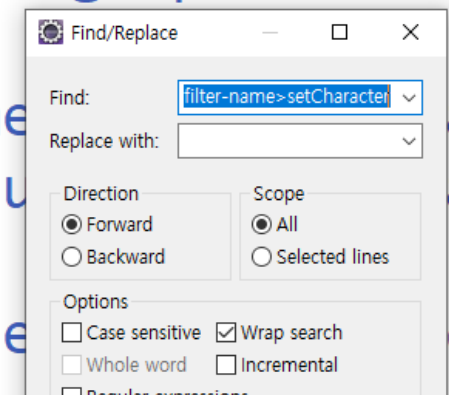
```
<init-param>
```

```
<param-name>
```

```
<param-value>
```

```
</init-param>
```

```
<async-supporte
```



실습



Quiz

이름:

주소:

좋아하는 동물: ☐ 강아지 ☒ 고양이 ☒ 돼지

입력한 정보

이름 : 홍길동

주소 : 산골짜기

동물 : [cat] [pig]



Get VS Post

❖ 정의

- GET방식은 전송 URL에 Form으로부터 입력받은 내용을 결합된 스트링쿼리로 보내는 방식이다.(아래 URL 참조)

❖ 사용 예

- <http://www.siteName.co.kr?id=hello&pw=1234>

❖ 특징

- FORM으로부터 입력 받은 내용이 URL에 표시된다.
- URL에 정보가 모두 나오기 때문에 보안성이 떨어진다.
- POST보다 속도가 빠르다.(처음 접근 후에, 재 요청시 빠르게 접근하기 위해서 데이터를 저장시켜 놓기 때문)
- 전송데이터의 한계가 있다. (255자)
- 게시판의 리스트나 페이지를 볼 때 사용하면 유용하다.
- 보통 SELECT 성향 즉, 데이터를 가져와 보여줄 때 사용한다.

Get VS Post

❖ 정의

- POST방식은 GET방식과 다르게 URL에 어떠한 내용도 붙여서 보내지 않는다. 대신 BODY에 데이터를 넣어서 전송한다.

❖ 특징

- 정보를 숨길 수 있어서 GET보다는 보안성이 높으나 그렇다고해서 안심해서는 안된다. (URL에서 보이지 않을 뿐..)
- GET에 비해 속도가 느리다.
- 전송 데이터양의 한계가 없다.
- Header에 BODY에 대한 설명을 명시해 줘야한다. (Content-Type...등)
- 보통 Update 성향 즉, 값의 상태를 바꿀 때 사용한다. (글의 내용 수정...등)

실습

```
<form method="get" action="/Chapter1/Ex03Request/viewParameter.jsp">
```

이름: 이름:

주소: 주소:

좋아하는 동물: ☒ 강아지 ☒ 고양이 ☐ 돼지

http://localhost:8080/Chapter1/
Ex03Request/viewParameter.jsp
?name=name&address=address
&pet=dog&pet=cat

```
<form method="post" action="/Chapter1/Ex03Request/viewParameter.jsp">
```

이름: 이름:

주소: 주소:

좋아하는 동물: ☒ 강아지 ☒ 고양이 ☐ 돼지

http://localhost:8080/Chapter1/
Ex03Request/viewParameter.jsp