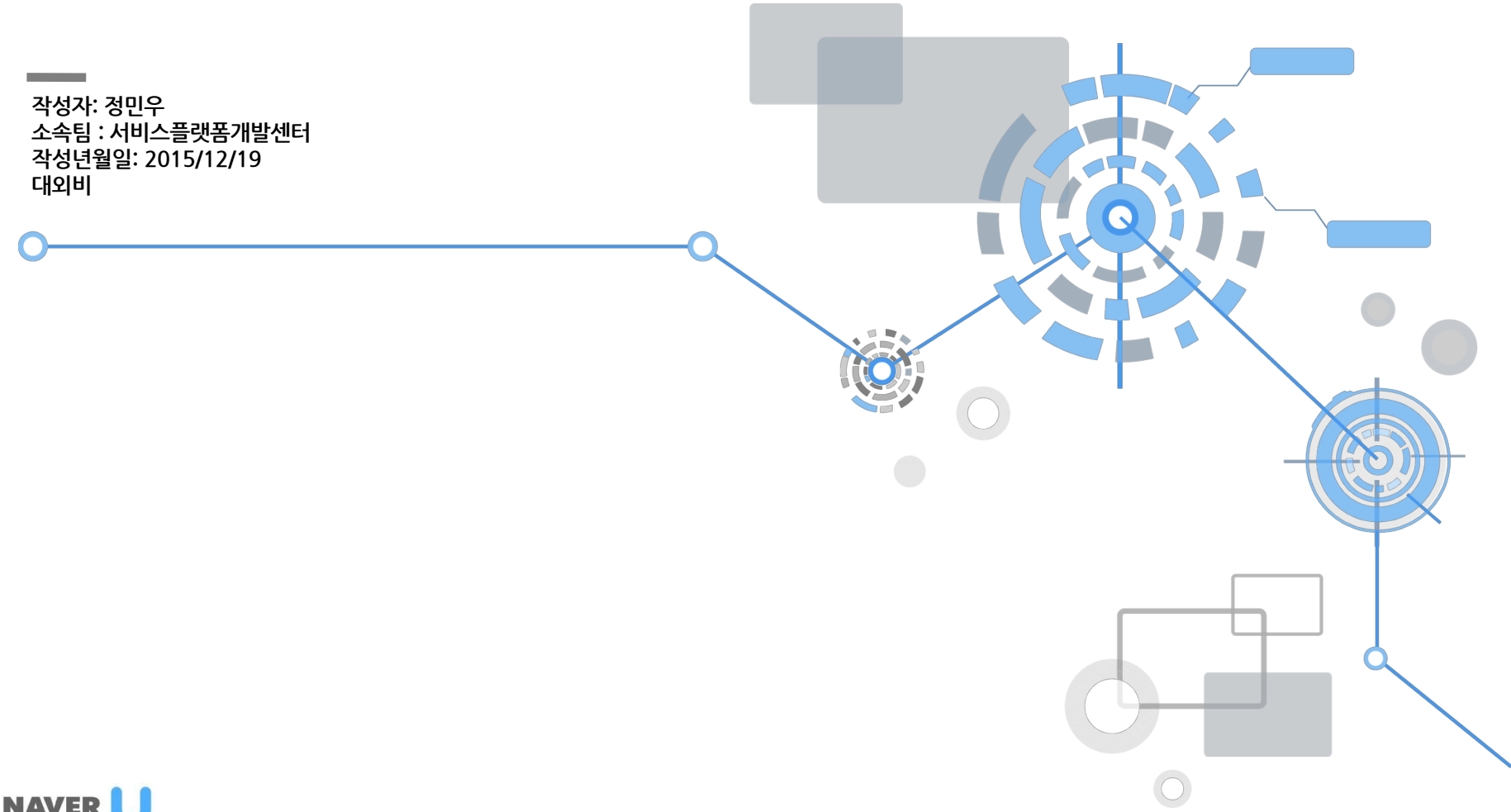
2016 NAVER / Works Mobile 신입기술교육

웹 개발의 이해

작성자: 정민우
소속팀 : 서비스플랫폼개발센터
작성년월일: 2015/12/19
대외비



NAVER

목표

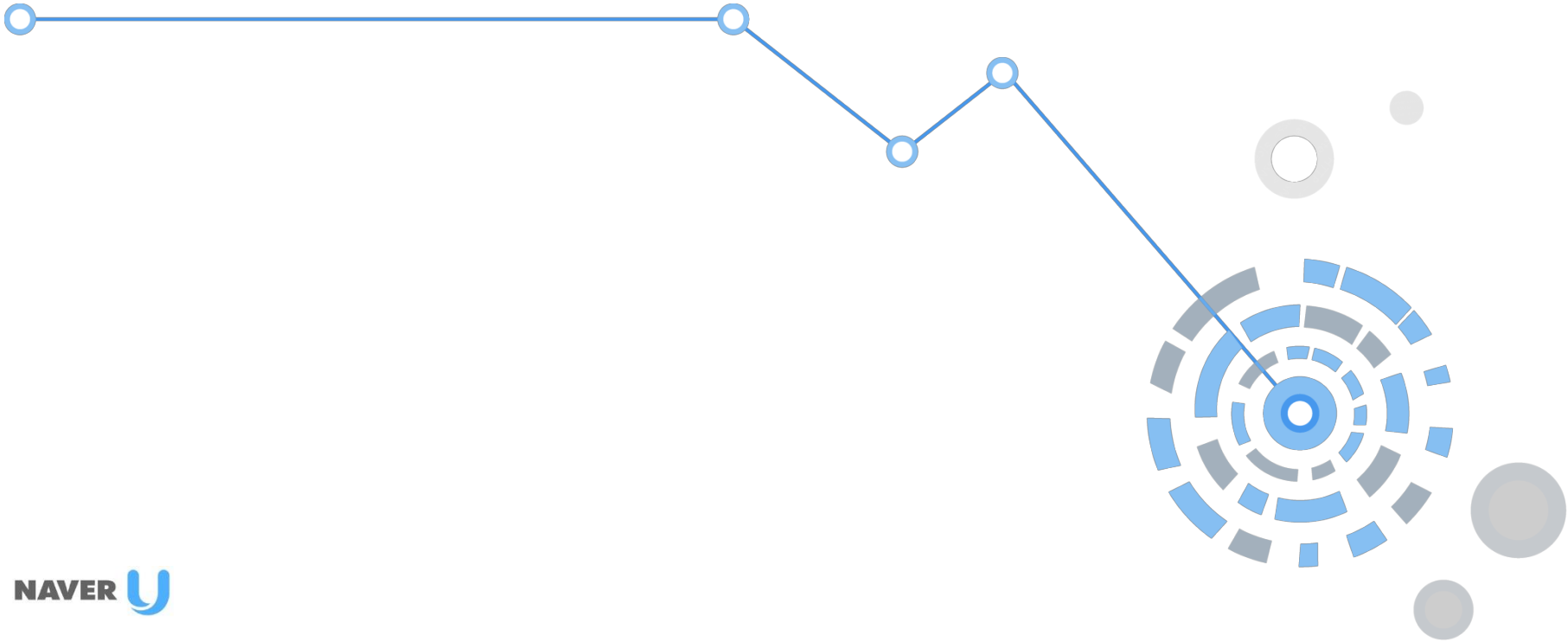
- **웹 서비스의 구조 이해**
  - 사내 웹서비스의 구조를 이해 한다.
- **Servlet & jsp 이해**
  - Servlet 과 jsp를 이해한다.
- **Error & Exception 분석 방법**
- **Logging**

NAVER U

목차

NAVER U

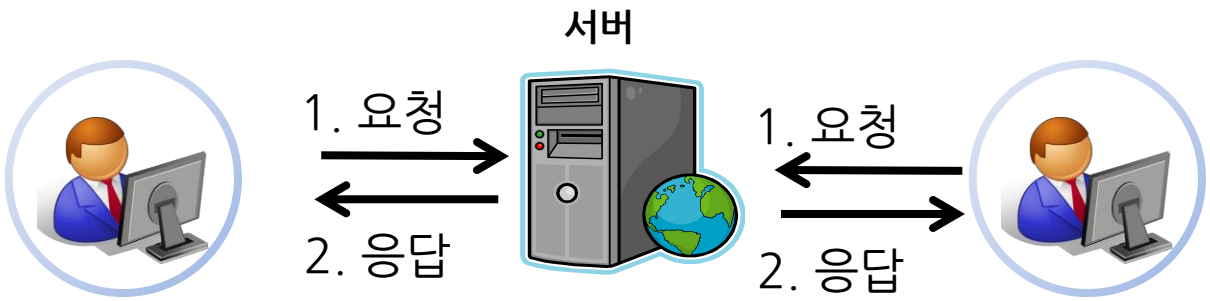
1. 웹서비스의 구조



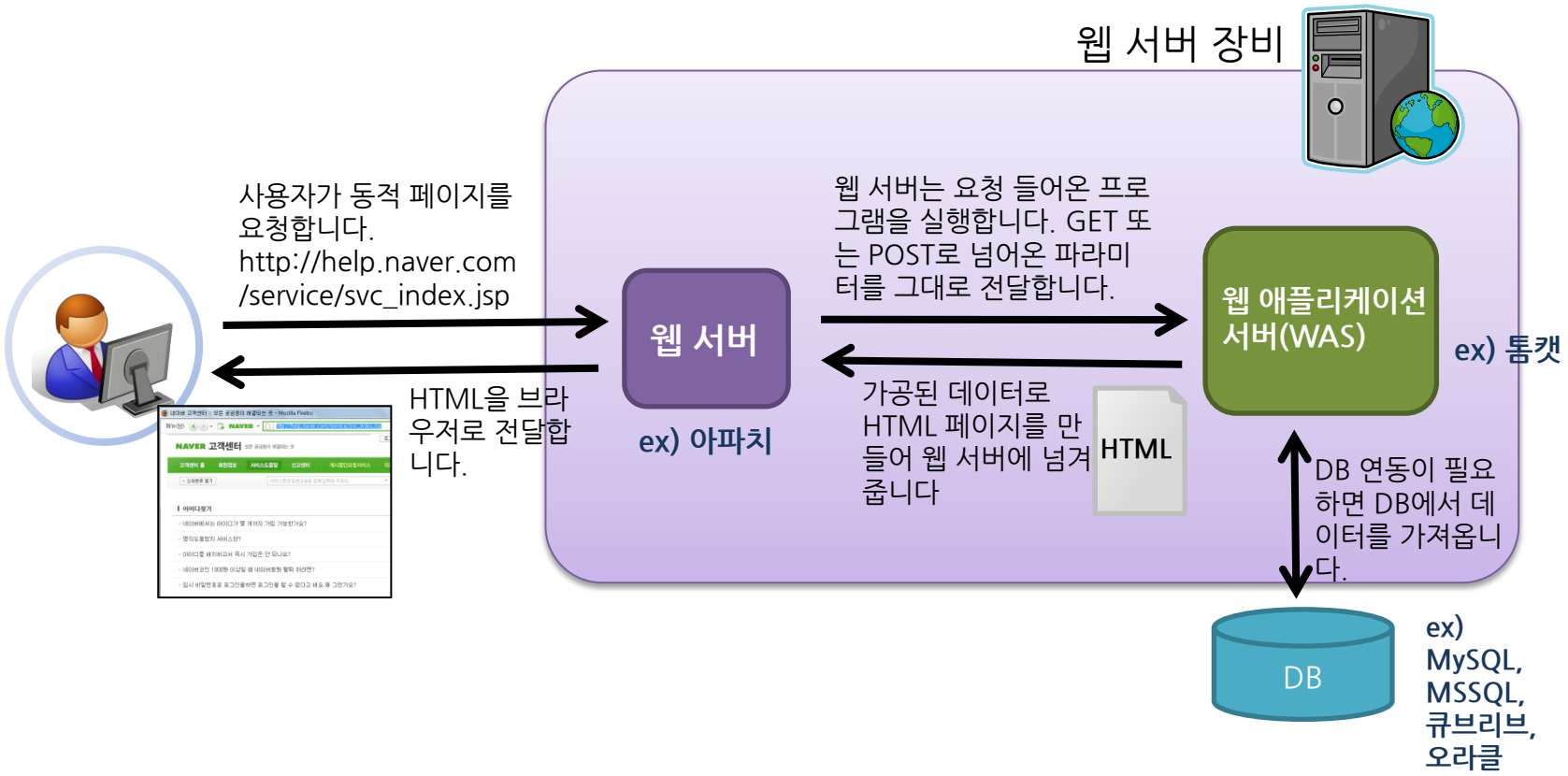
NAVER

- **Client-Server 시스템**

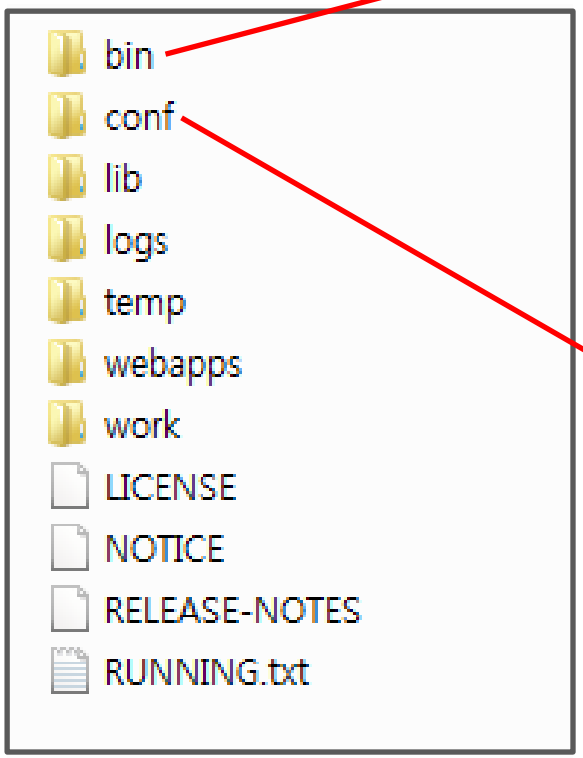
Client가 서비스 요구자로서 요청(Request)정보를 전송하면,
Server는 서비스 제공자로서 응답(Response)정보를 돌려주는 형태로
네트워크를 통해서 자원을 공유하는 시스템 형태

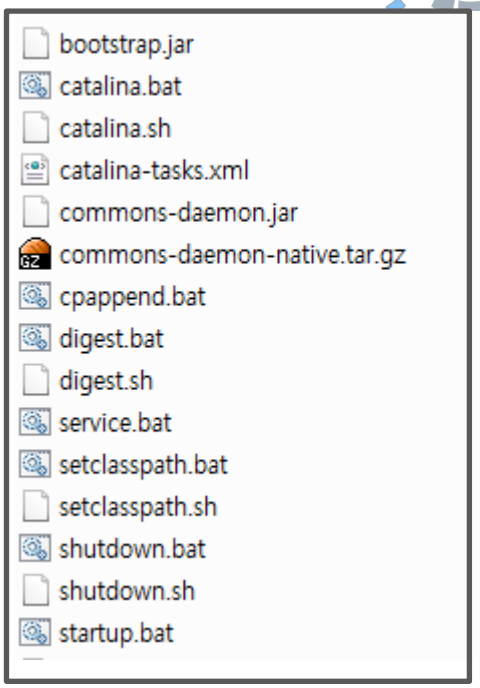


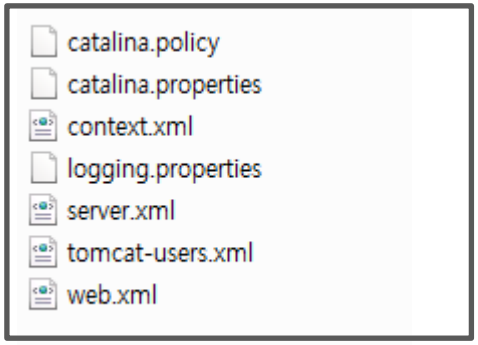
1. 웹 서비스의 구조


웹 서버 장비
사용자가 동적 페이지를 요청합니다.
http://help.naver.com/service/svc_index.jsp
웹 서버
ex) 아파치
웹 서버는 요청 들어온 프로그램을 실행합니다. GET 또는 POST로 넘어온 파라미터를 그대로 전달합니다.
웹 애플리케이션 서버(WAS)
ex) 톰캣
HTML을 브라우저로 전달합니다.
가공된 데이터로 HTML 페이지를 만들어 웹 서버에 넘겨줍니다
HTML
DB 연동이 필요하면 DB에서 데이터를 가져옵니다.
DB
ex) MySQL, MSSQL, 큐브리브, 오라클

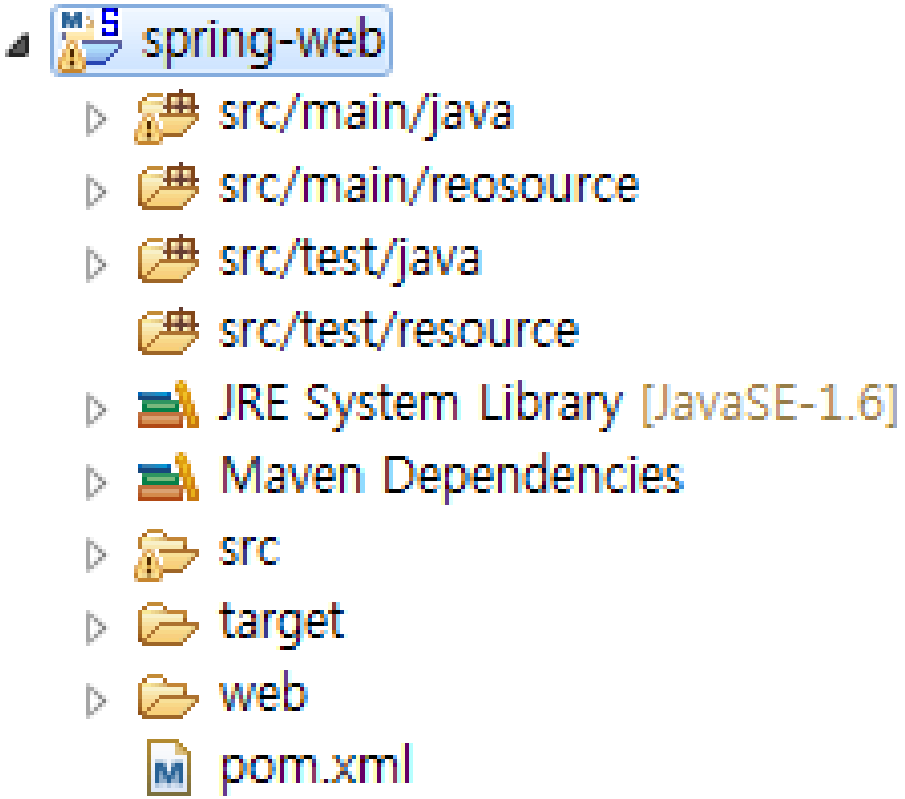
1.1 웹 서비스의 구조 - WAS



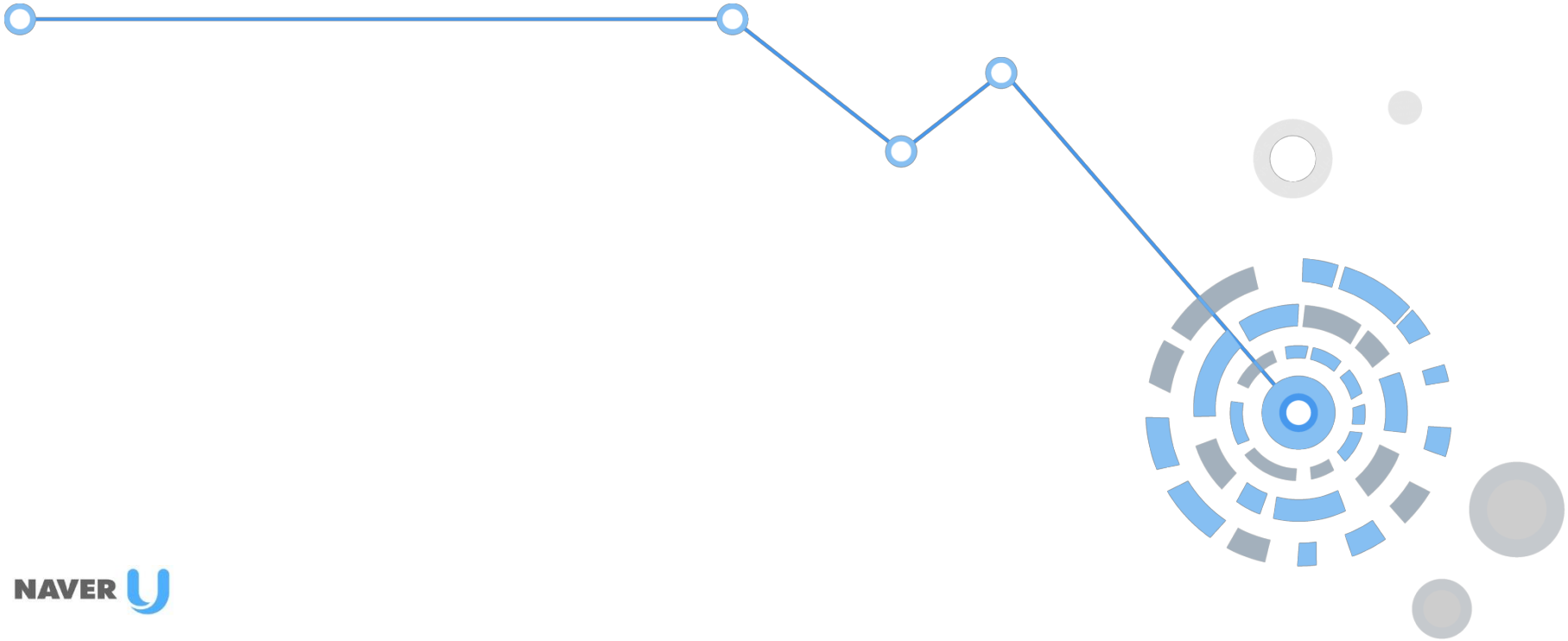




1.2 웹 서비스의 구조 - Java web Project

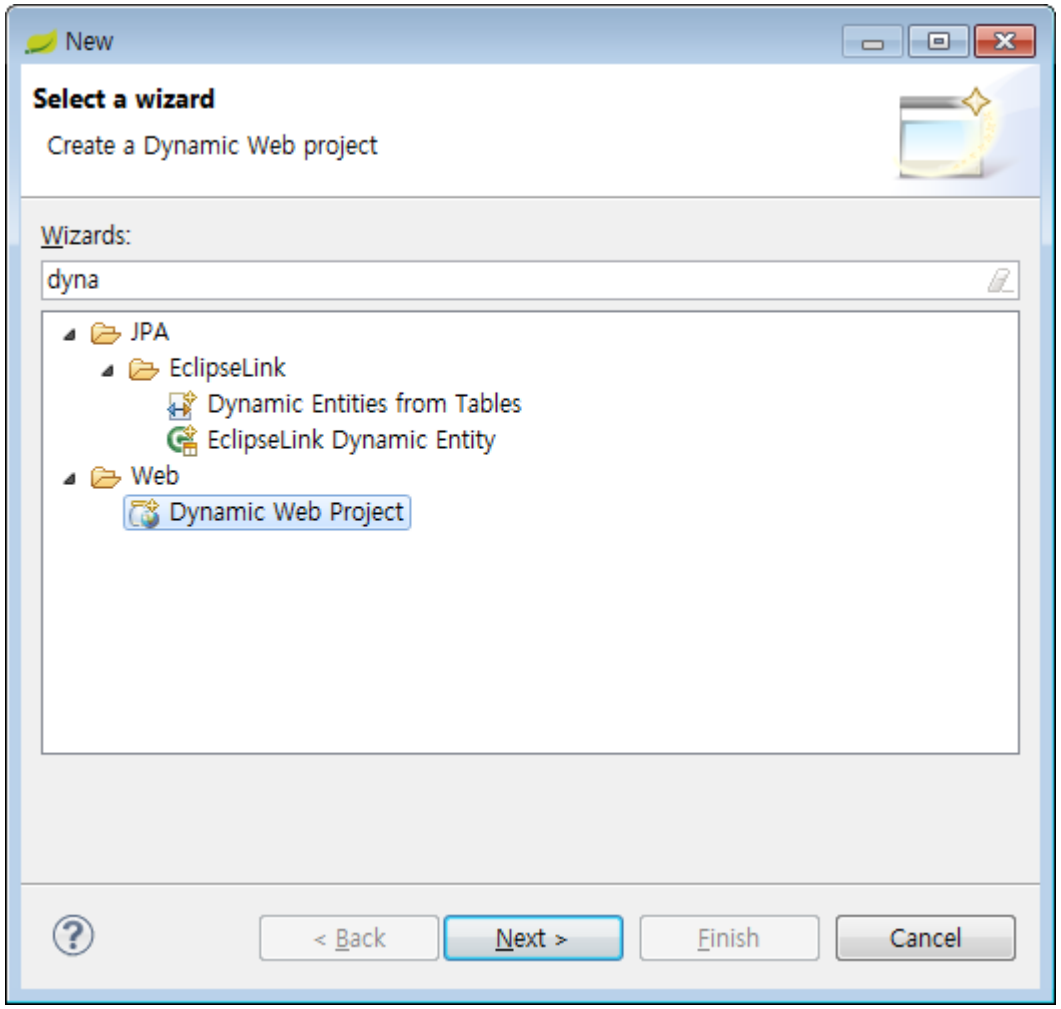


2. Servlet & Jsp 프로그래밍



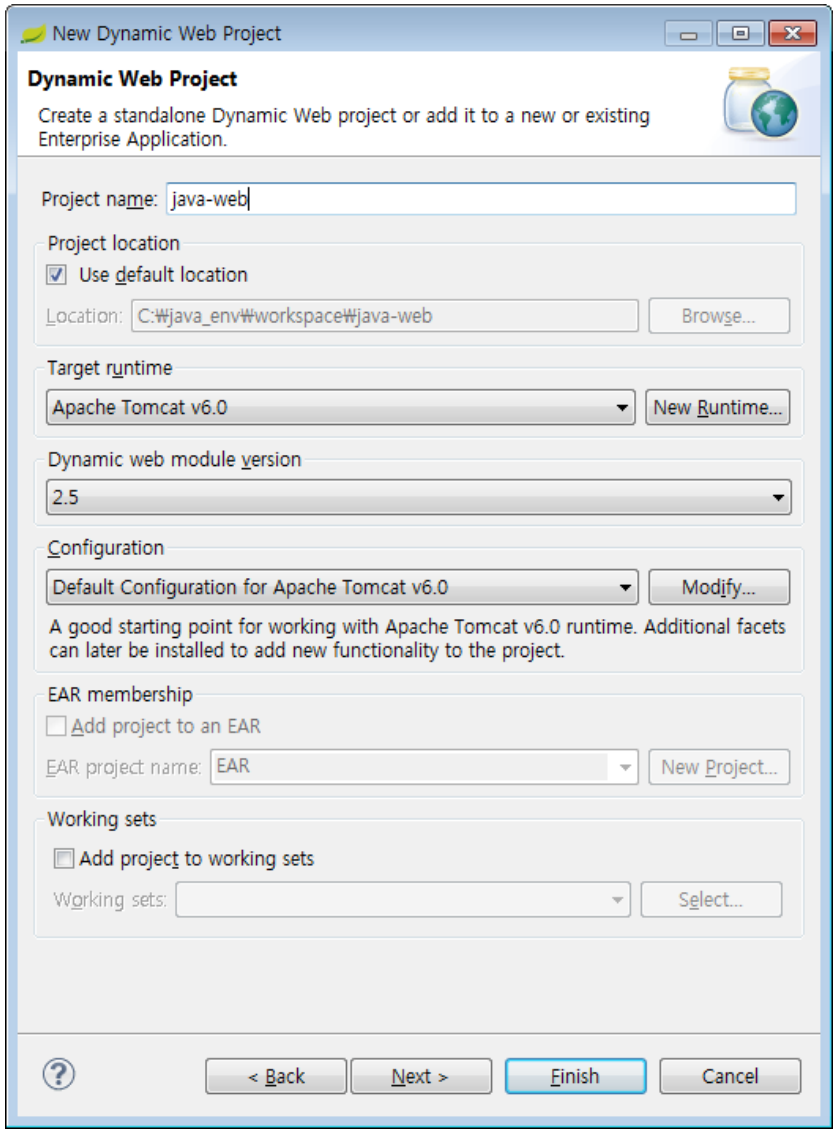
NAVER

2.1 Servlet & Jsp 프로그래밍 - 웹 프로젝트 만들기



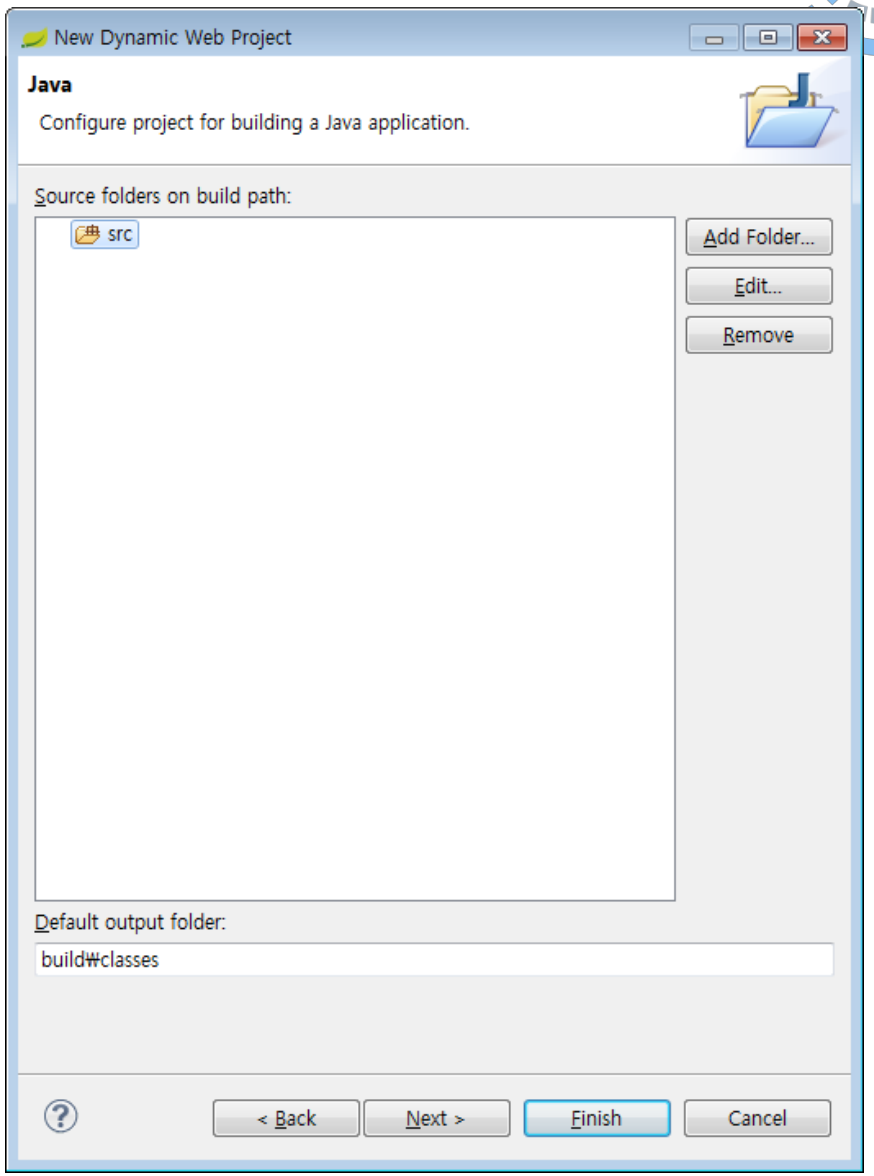
2.1 Servlet & Jsp 프로그래밍 - 웹 프로젝트 만들기

- 프로젝트 이름 및 tomcat 설정



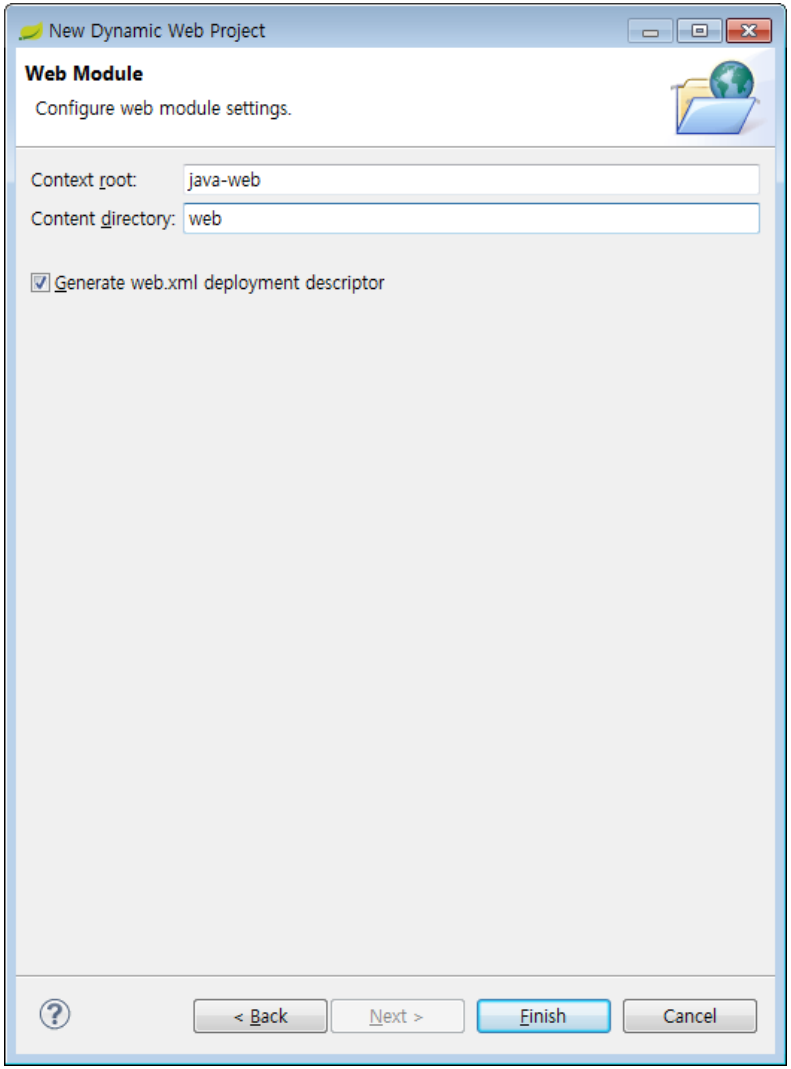
2.1 Servlet & Jsp 프로그래밍 - 웹 프로젝트 만들기

- java code 위치 지정

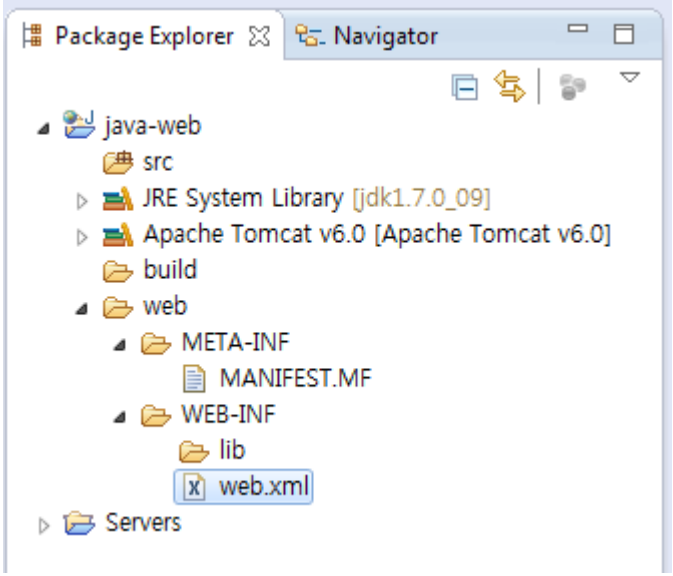


2.1 Servlet & Jsp 프로그래밍 - 웹 프로젝트 만들기

- web module 지정



프로젝트 생성 확인

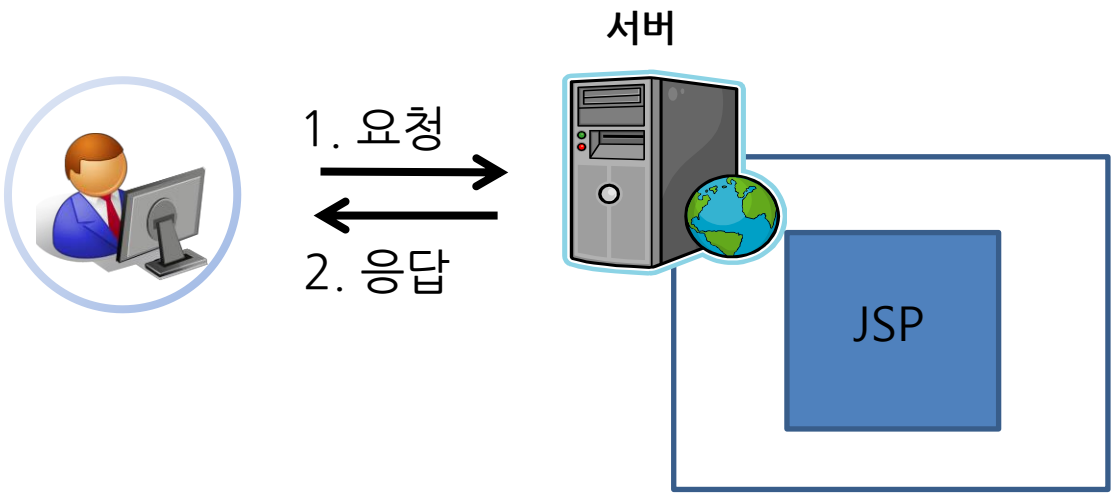

Package Explorer
Navigator
java-web
src
JRE System Library [jdk1.7.0_09]
Apache Tomcat v6.0 [Apache Tomcat v6.0]
build
web
META-INF
MANIFEST.MF
WEB-INF
lib
web.xml
Servers

web.xml

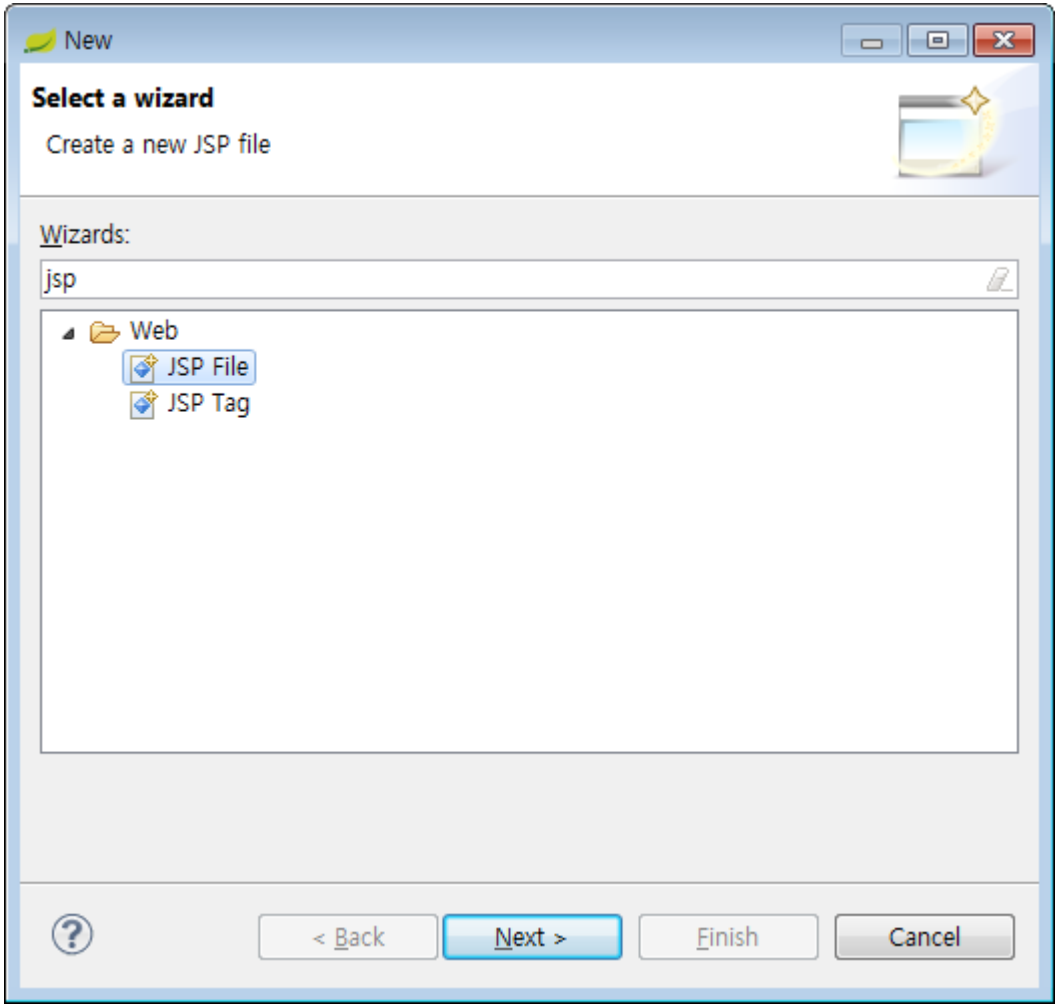
- web application의 기본설정
- 각종 servlet의 설정과 servlet 매핑, 필터, 인코딩 등을 담당

web/WEB-INF/web.xml

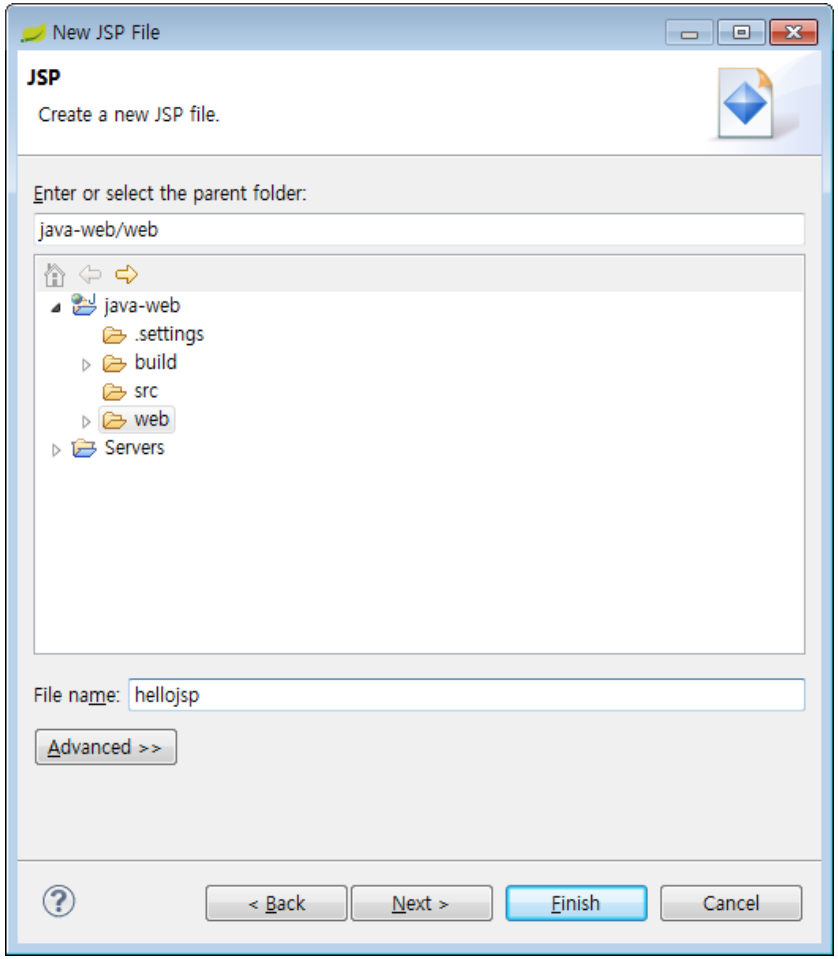
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
id="WebApp_ID" version="2.5">
  <display-name>java-web</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>
```


서버
1. 요청
2. 응답
JSP

2.2 Servlet & Jsp 프로그래밍 - jsp 파일 만들기



2.2 Servlet & Jsp 프로그래밍 - jsp 파일 만들기

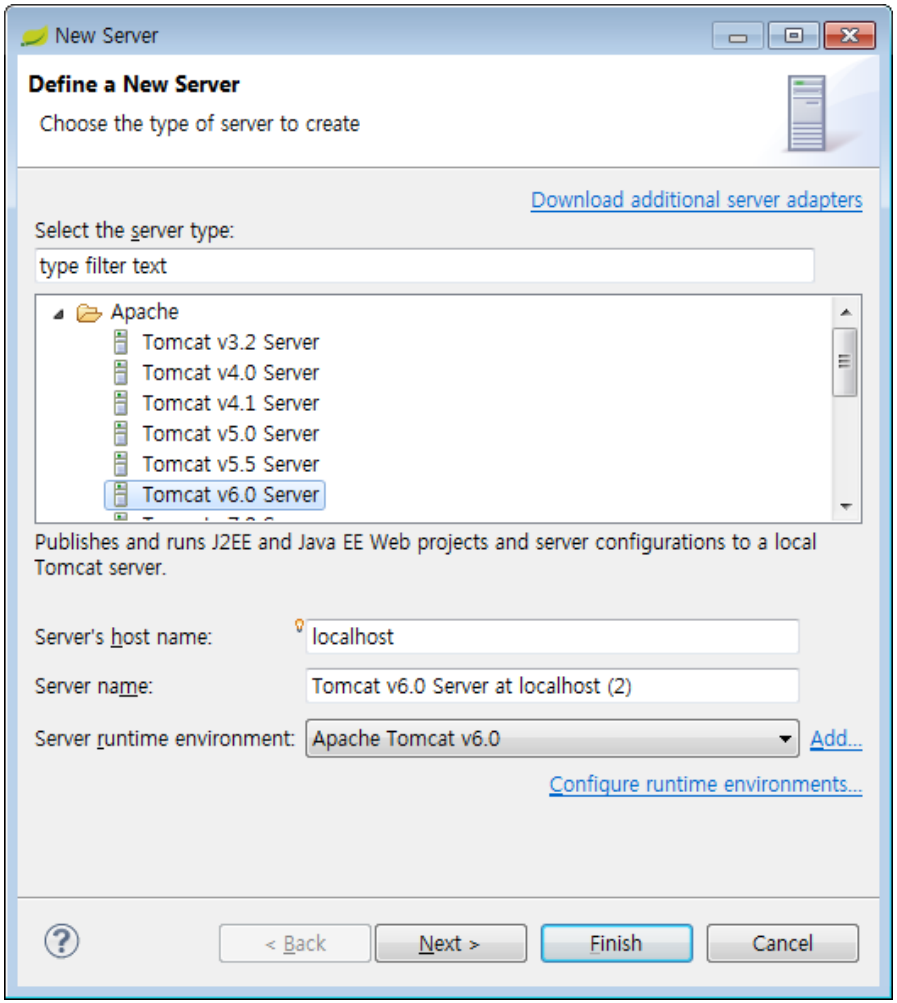


web/hello.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-
KR">
<title>hello nhn</title>
</head>
<body>
hello nhn!
</body>
</html>
```

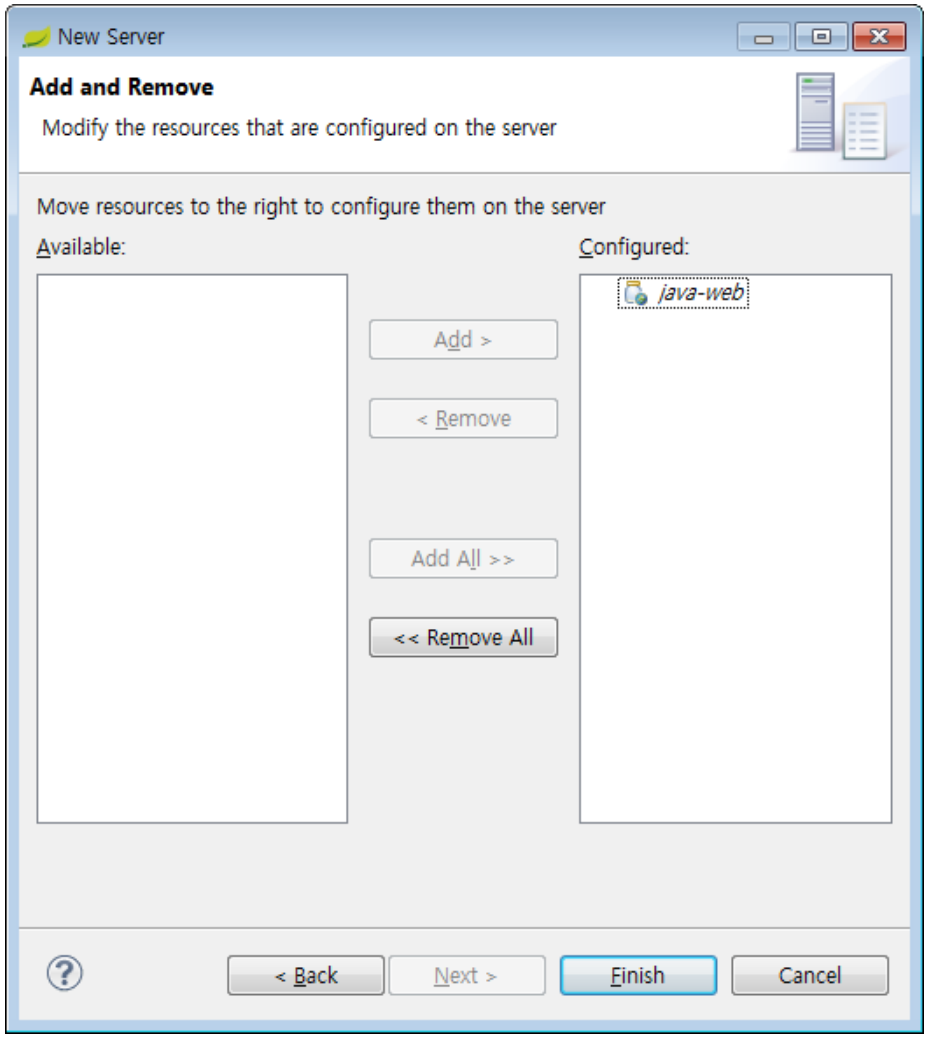
2.3 Servlet & Jsp 프로그래밍 - tomcat 연동

- server view에 tomcat 추가



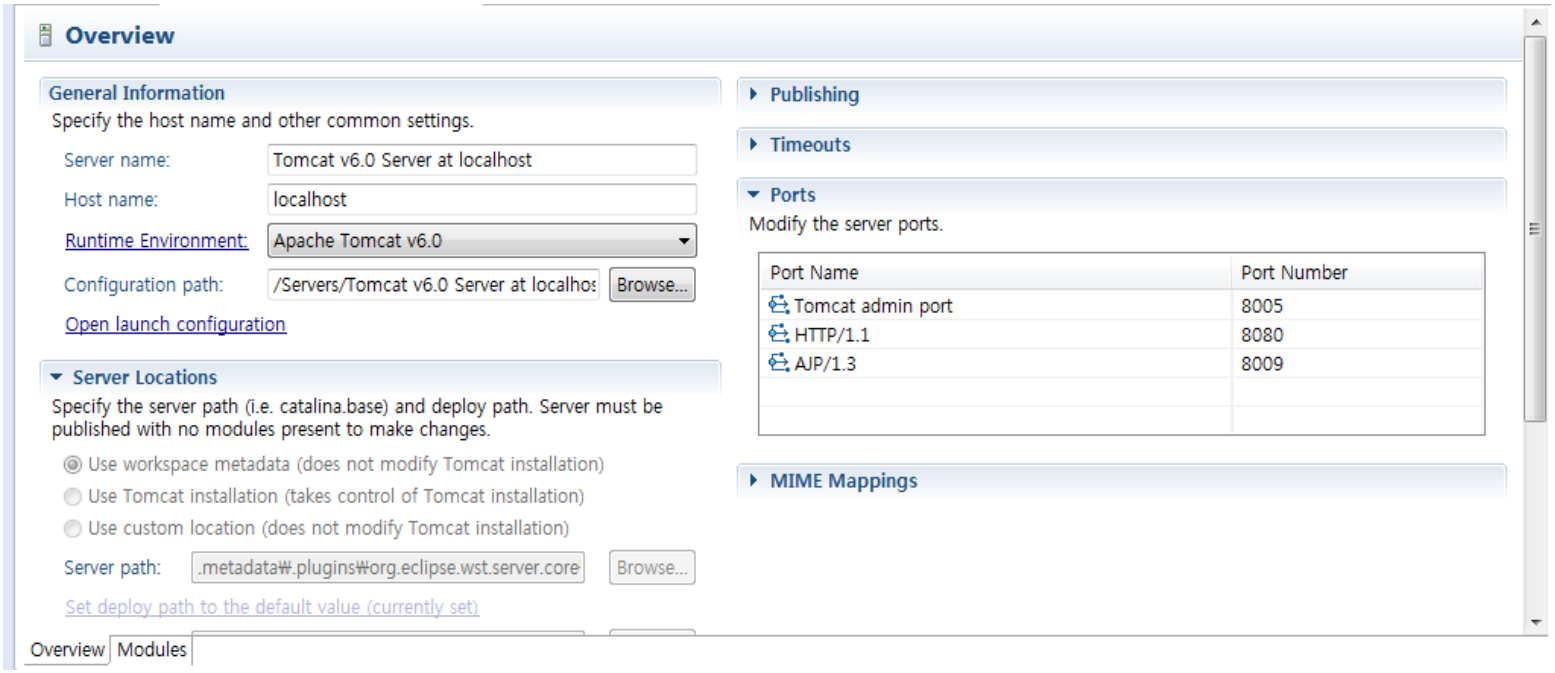
2.3 Servlet & Jsp 프로그래밍 - tomcat 연동

- tomcat 과 web project 연동



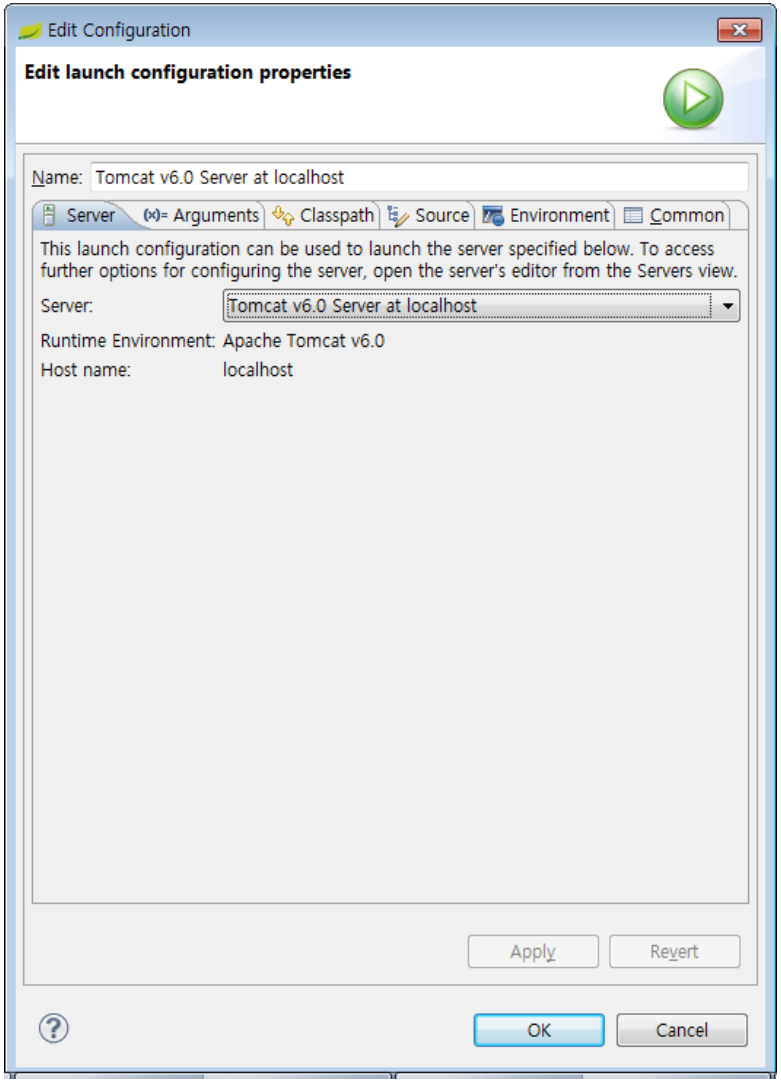
2.3 Servlet & Jsp 프로그래밍 - tomcat 연동

- tomcat 설정



2.3 Servlet & Jsp 프로그래밍 - tomcat 연동

- tomcat 실행 옵션 설정



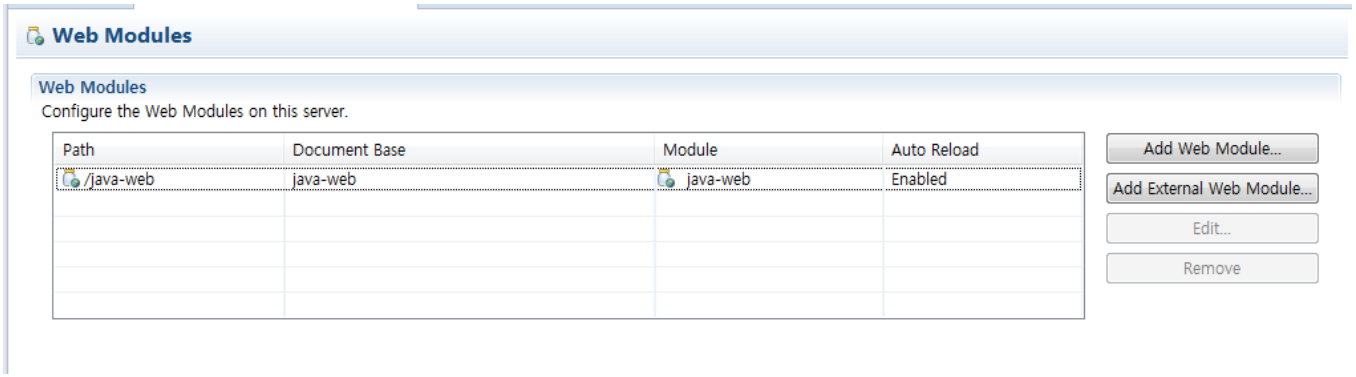
Tomcat 서버 실행 option 예제

irteam@ttest02.webp:~

```
----------------------------------------------------------------------------------
----------
----------------------------------------------------------------------------------
----------
[gw01:/home1/jmwiner] kinit
Password for jmwiner@NAVER.COM:
[gw01:/home1/jmwiner] rlogin -l irteam ttest02.webp
This rlogin session is encrypting all data transmissions.
Last login: Thu Jan  3 16:04:13 from 10.24.22.70
[irteam@ttest02.webp ~]$ ps -ef | grep java
irteam    4441     1  0 Jan03 pts/1    00:02:23 /home1/irteam/apps/jdk/bin/java
-Djava.util.logging.config.file=/home1/irteam/apps/apache-tomcat-6.0.35/conf/log
ging.properties -XX:+UseParNewGC -XX:+CMSParallelRemarkEnabled -XX:+UseConcMarkS
weepGC -XX:PermSize=256m -XX:MaxPermSize=256m -Xms512m -Xmx512m -server -Xloggc:
/logs/gc.log -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -
Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=13000 -Dcom.su
n.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.authenticate=fal
se -Djava.endorsed.dirs=/home1/irteam/apps/apache-tomcat-6.0.35/endorsed -classp
ath /home1/irteam/apps/apache-tomcat-6.0.35/bin/bootstrap.jar -Dcatalina.base=/h
ome1/irteam/apps/apache-tomcat-6.0.35 -Dcatalina.home=/home1/irteam/apps/apache-
tomcat-6.0.35 -Djava.io.tmpdir=/home1/irteam/apps/apache-tomcat-6.0.35/temp org.
apache.catalina.startup.Bootstrap start
irteam   29709 29676  0 16:30 pts/2    00:00:00 grep java
[irteam@ttest02.webp ~]$
```

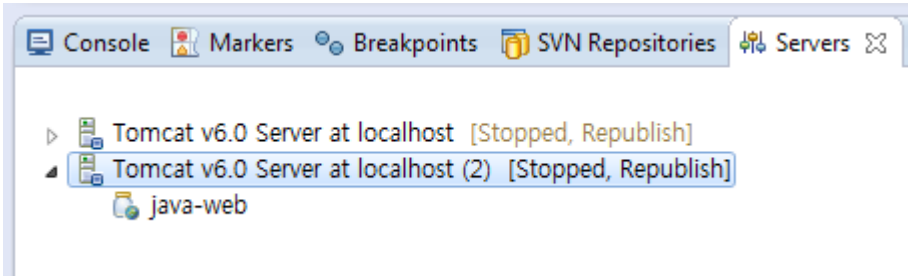
2.3 Servlet & Jsp 프로그래밍 - tomcat 연동

- web modules path 설정 화면



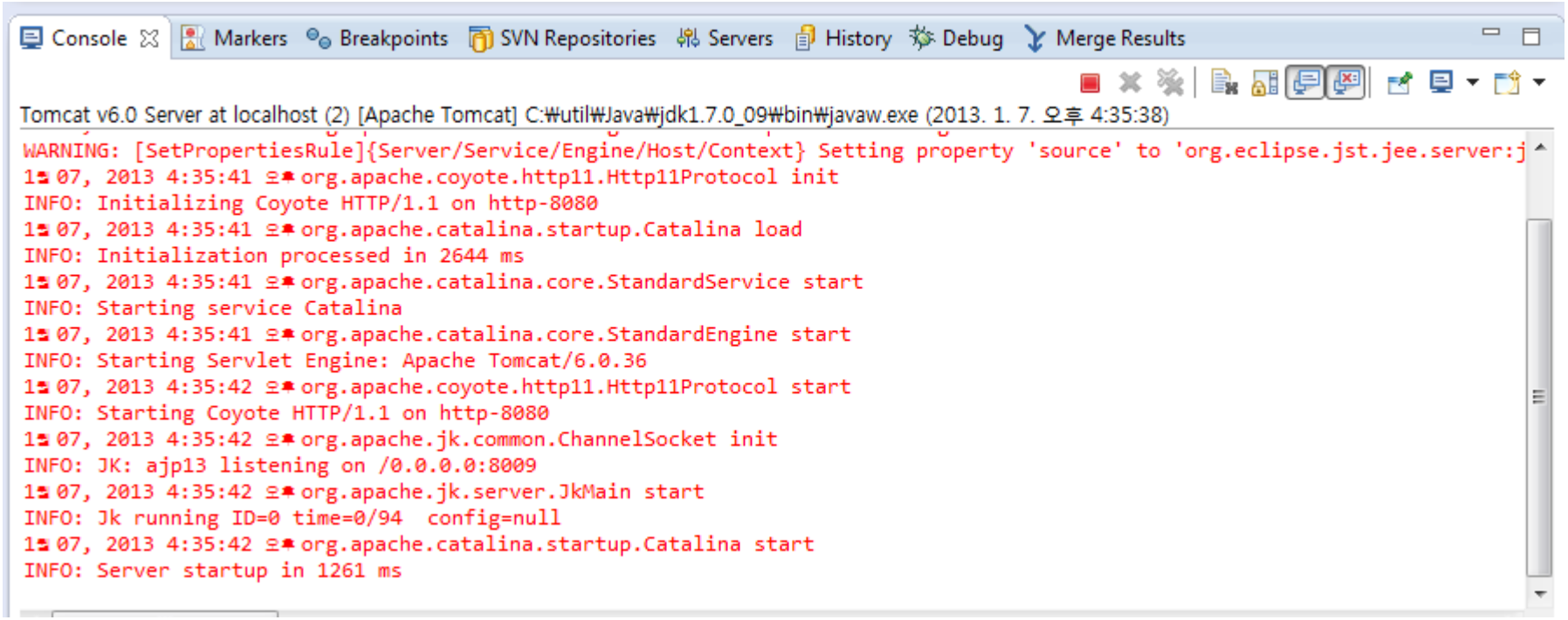
2.4 Servlet & Jsp 프로그래밍 - tomcat 실행

- tomcat 선택 후 실행



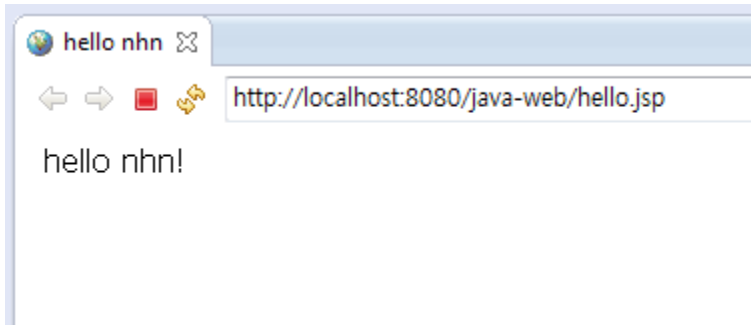
2.4 Servlet & Jsp 프로그래밍 - tomcat 실행

- tomcat 실행 작업 대한 log가 console에 출력 된다.

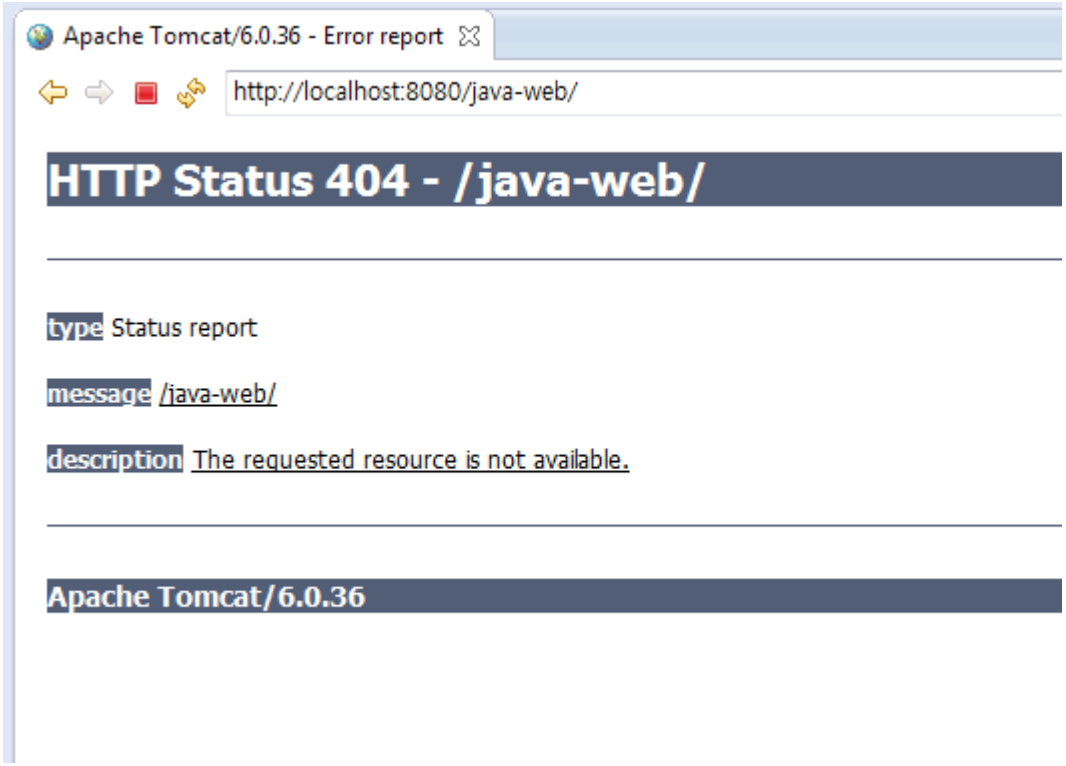


2.5 Servlet & Jsp 프로그래밍 - url 호출

- web project 호출



2.6 Servlet & Jsp 프로그래밍 - http status code

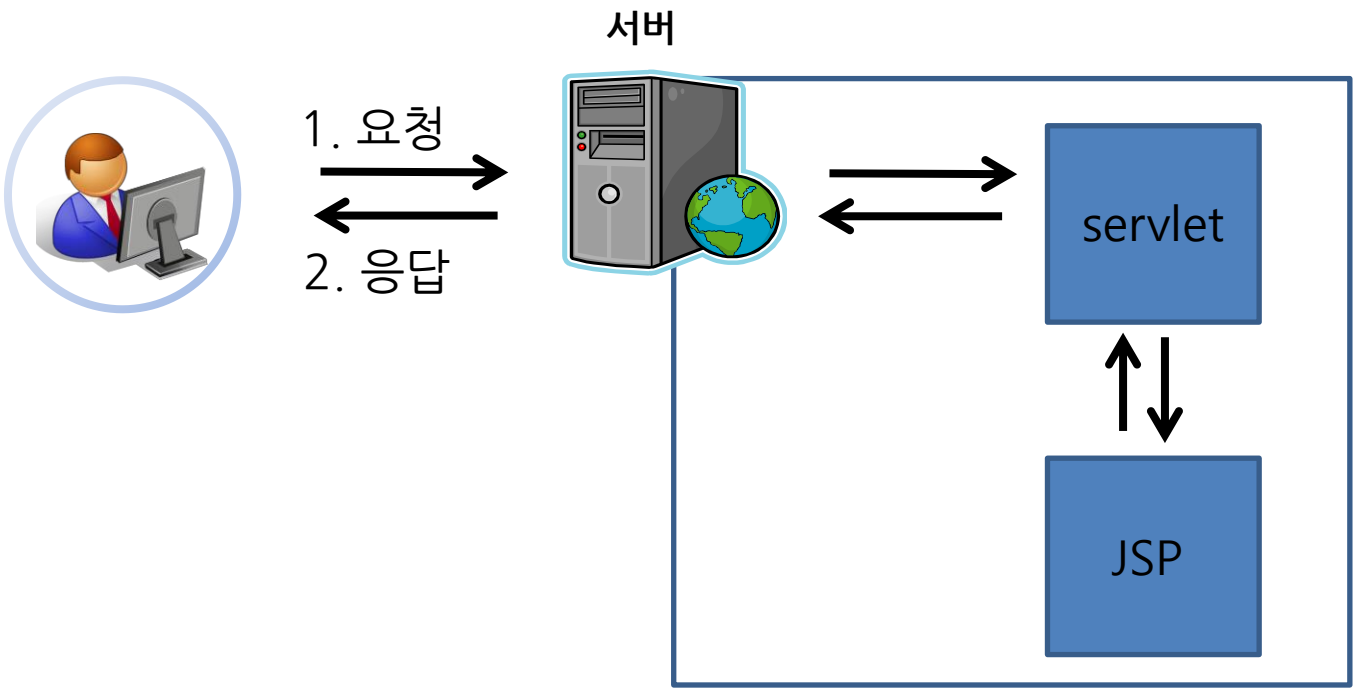


뭐지?

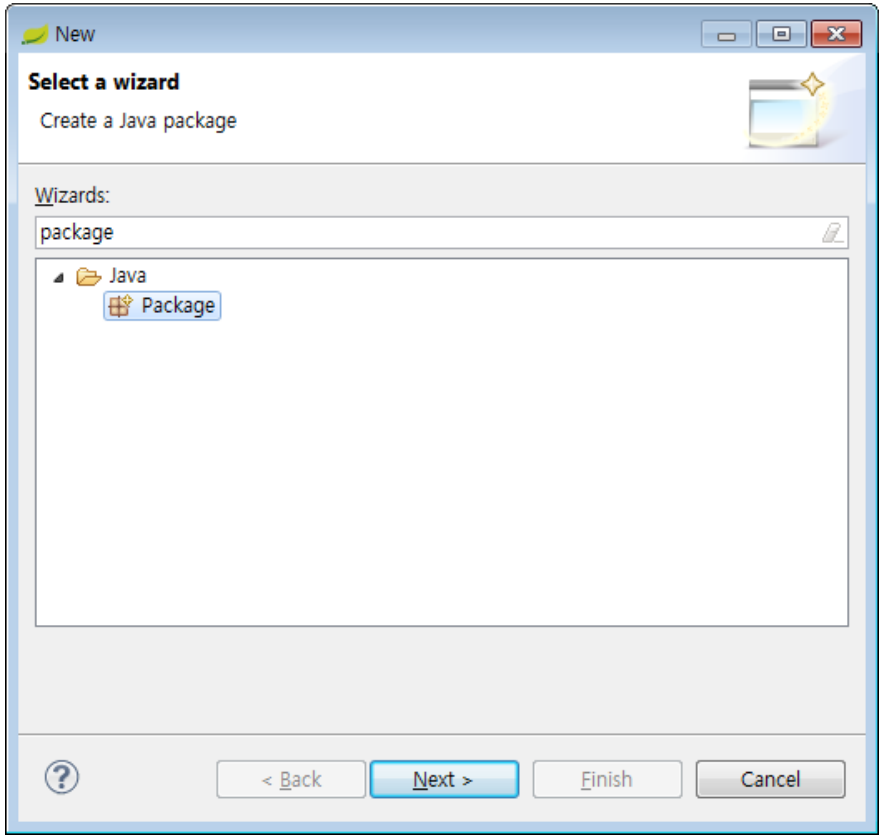
code	메시지	설명
200	OK	에러 없이 성공
404	Not found	문서를 찾을 수 없음. 서버가 요청한 파일이나 스크립트를 찾지 못함.
500	Internal Server Error	서버 내부 오류

참고 : http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

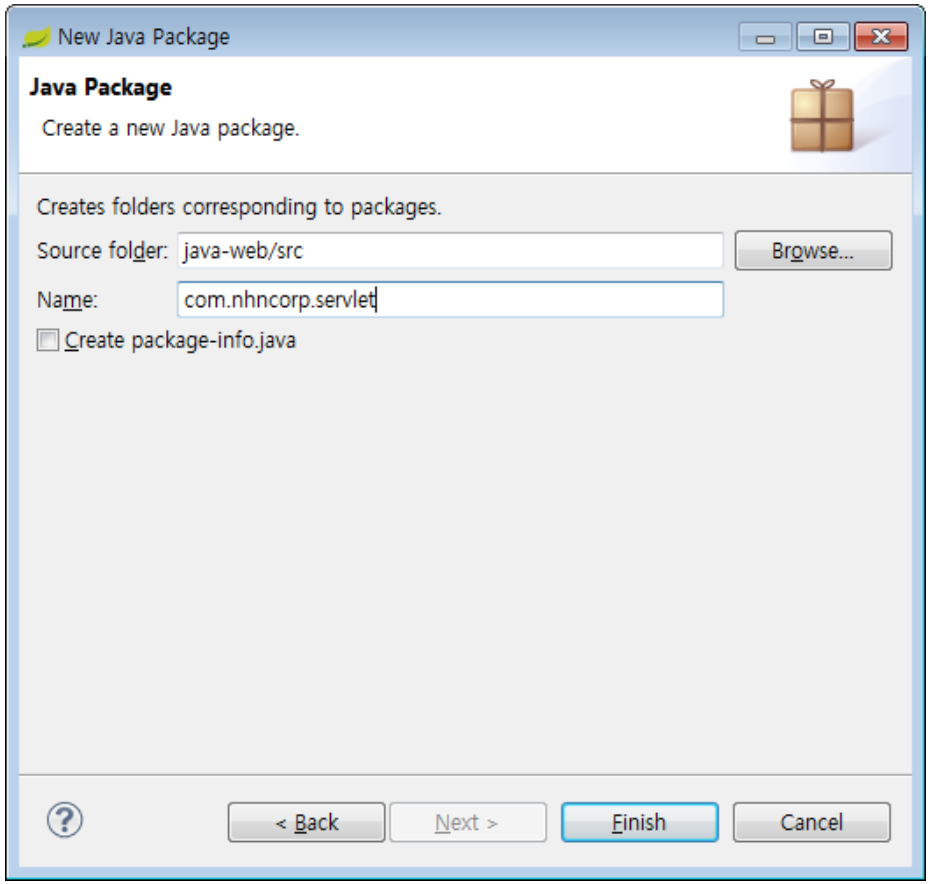
2.7 Servlet & Jsp 프로그래밍 - servlet 생성



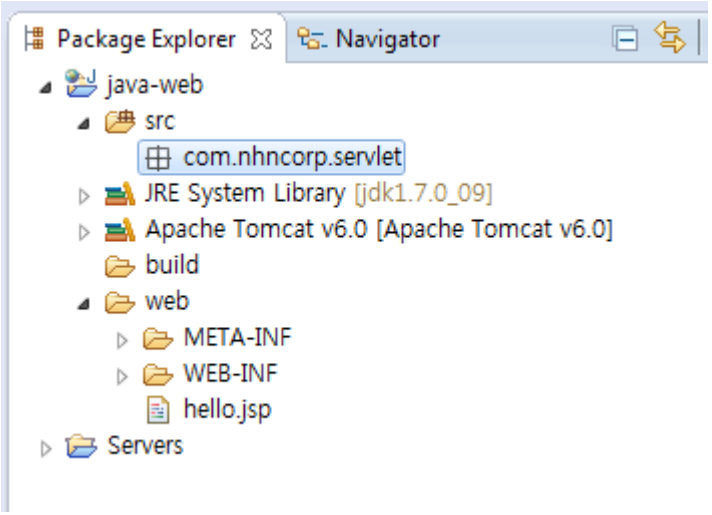
- **Package 추가**


New
Select a wizard
Create a Java package
Wizards:
package
Java
Package
< Back
Next >
Finish
Cancel

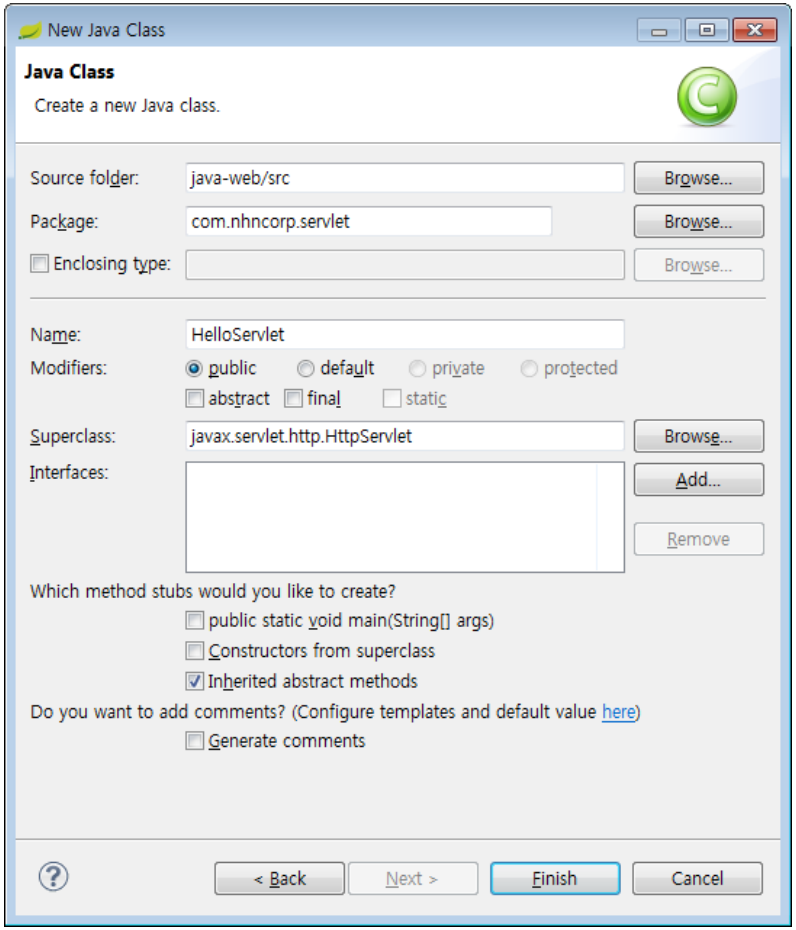
- **Package 추가**


New Java Package
Java Package
Create a new Java package.
Creates folders corresponding to packages.
Source folder: java-web/src
Browse...
Name: com.nhncorp.servlet
Create package-info.java
< Back
Next >
Finish
Cancel

- **Servlet 파일 추가**



- **Servlet 파일 추가**

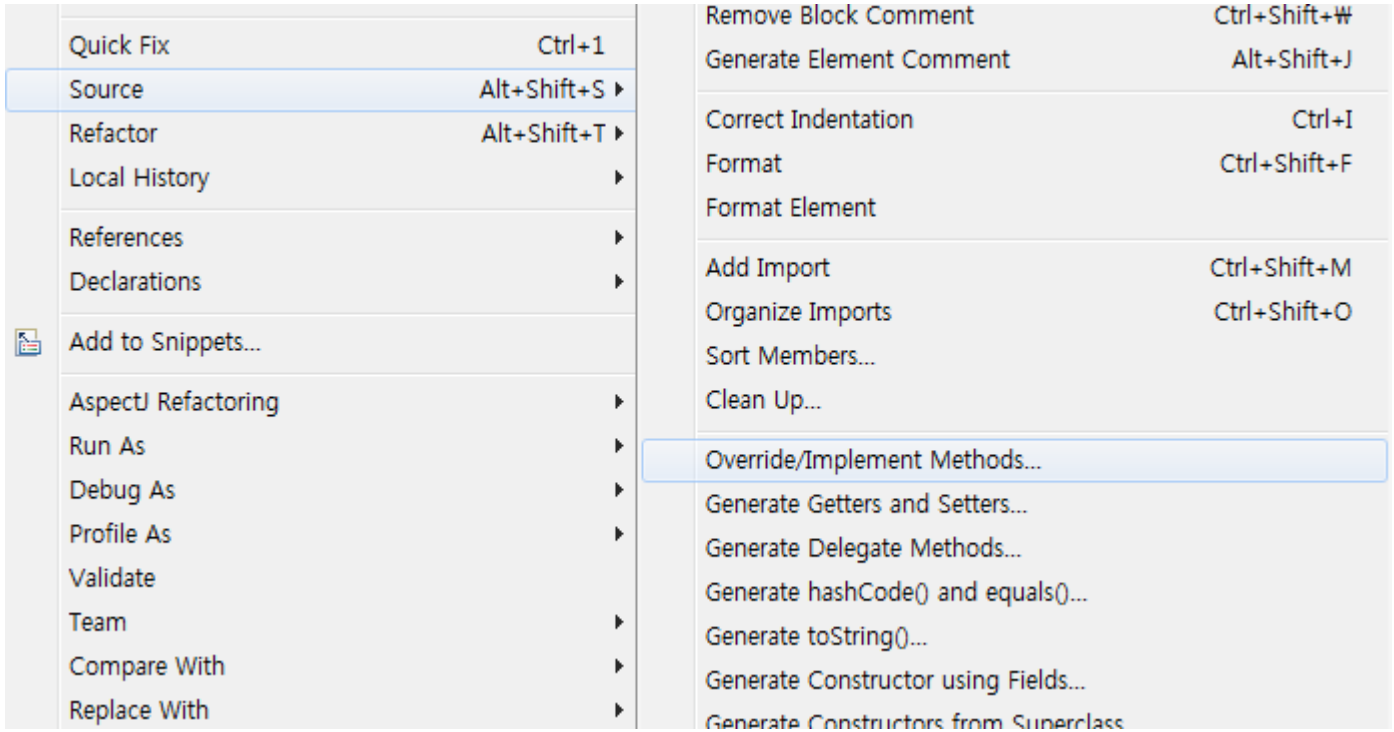


com/nhncorp/servlet/HelloServlet.java

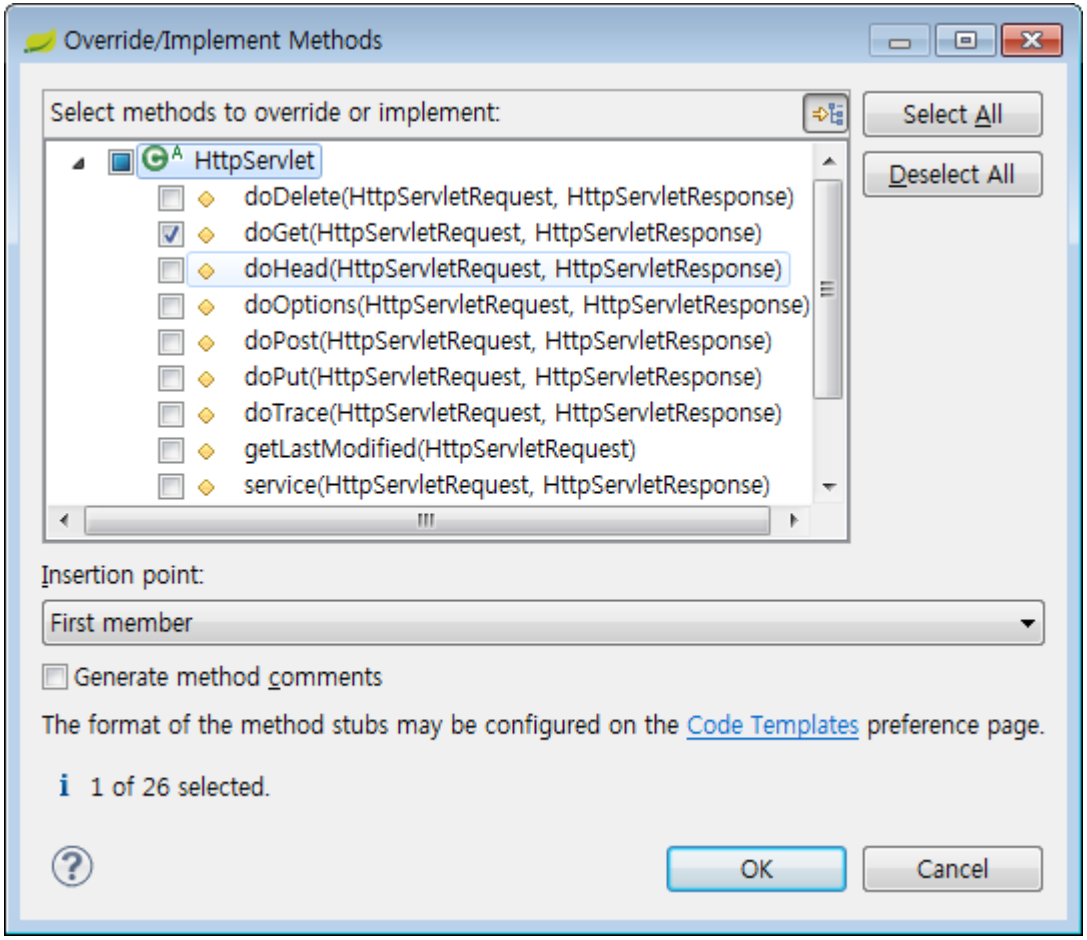
HelloServlet.java | Dashboard | HttpServlet.class

```
1  package com.nhncorp.servlet;
2
3  import javax.servlet.http.HttpServlet;
4
5  public class HelloServlet extends HttpServlet {
6
7  }
8
```

2.7 Servlet & Jsp 프로그래밍 - servlet 생성



2.7 Servlet & Jsp 프로그래밍 - servlet 생성



com/nhncorp/servlet/HelloServlet.java

```
public class HelloServlet extends HttpServlet {

@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException{
      String name = req.getParameter("name");

      if(name == null){
          name = "NHN";
      }

      String hello = "hello " + name;
      req.setAttribute("hello", hello);

      RequestDispatcher dispatcher = req.getRequestDispatcher("/WEB-INF/views/hello.jsp");
      dispatcher.forward(req, resp);
      }

}
```

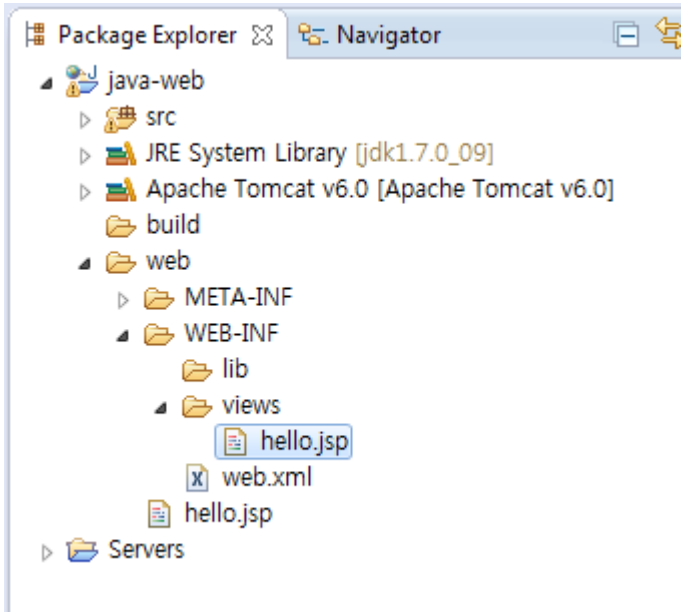
2.7 Servlet & Jsp 프로그래밍 - servlet 생성

web/WEB-INF/web.xml

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

NAVER

web/WEB-INF/hello.jsp

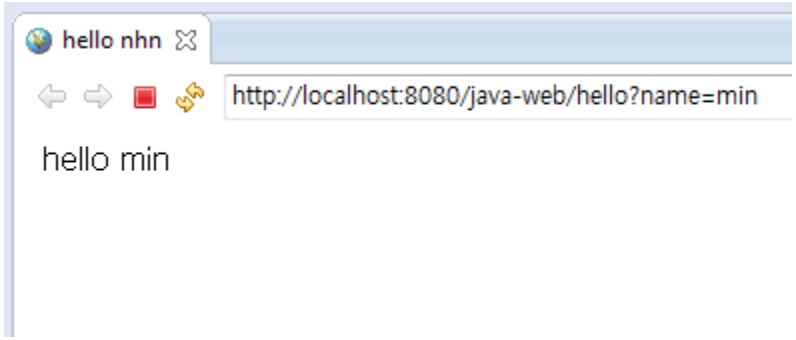


2.7 Servlet & Jsp 프로그래밍 - servlet 생성

web/WEB-INF/hello.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
<title>hello nhn</title>
</head>
<body>
${hello}
</body>
</html>
```

2.8 Servlet & Jsp 프로그래밍 - url 호출



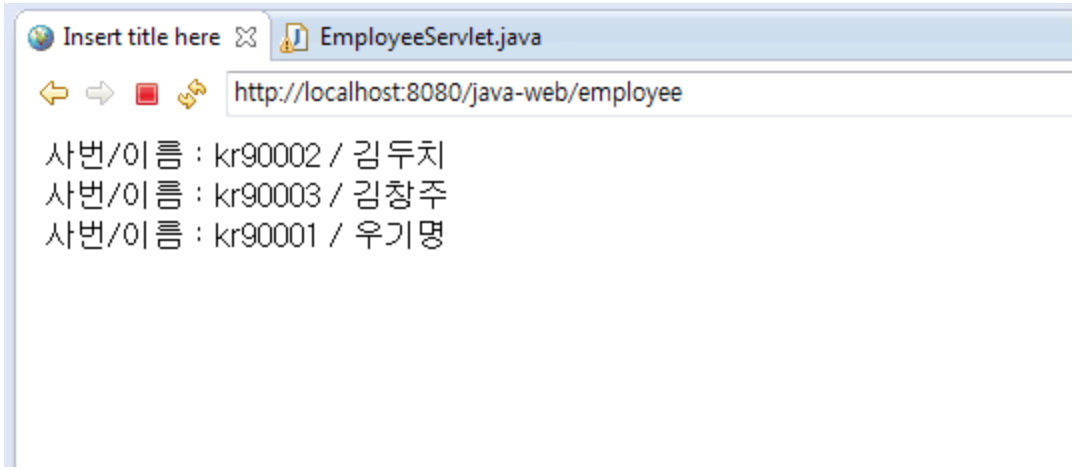
실습 1

모든 사원의 리스트를 출력하는 페이지 추가

- jsp 에서 C taglib를 사용해야함.
  - c taglib를 사용하기 위해서는 아래와 같은 선언이 필요함.
  - <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

- C taglib 사용을 위해서는 jstl, standard 라이브러리가 필요

2.8 Servlet & Jsp 프로그래밍 - 실습

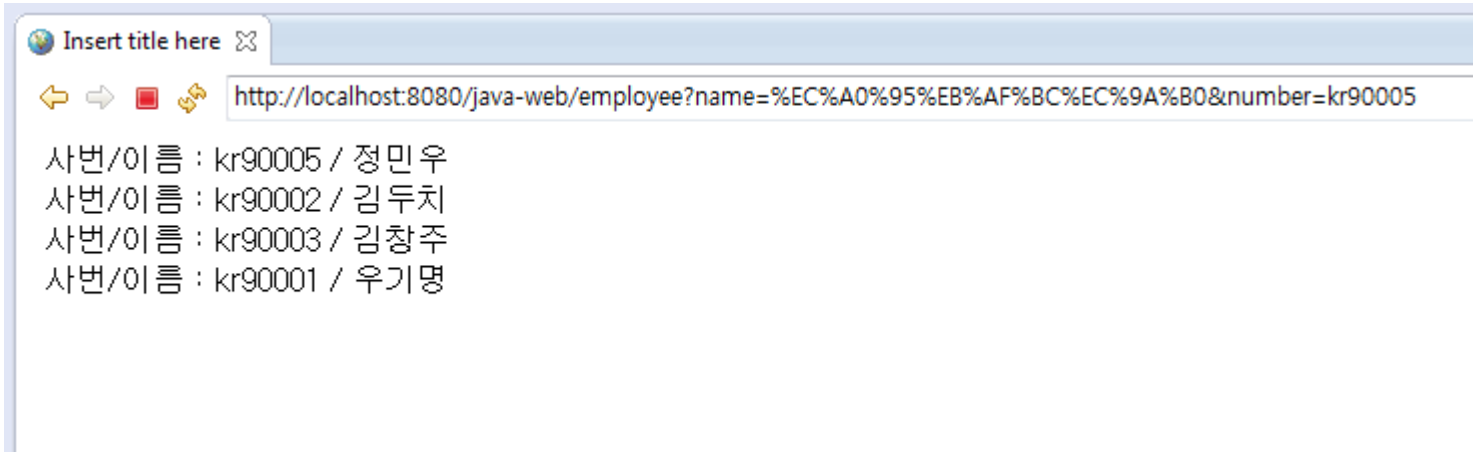


실습 2

사번/이름을 파라미터로 전달하고 사원을 추가하여 리스트를 출력

- 이름을 utf-8로 encoding 하여 전달 해야 함
- tomcat 설정 파일에 encoding 설정을 해야함.
- server.xml : <Connector connectionTimeout="20000" port="8080"
  protocol="HTTP/1.1" redirectPort="8443" URIEncoding="UTF-8" />

2.8 Servlet & Jsp 프로그래밍 - 실습



web/WEB-INF/web.xml

```
<servlet>
    <servlet-name>employee</servlet-name>
    <servlet-class>com.nhncorp.servlet.EmployeeServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>employee</servlet-name>
    <url-pattern>/employee</url-pattern>
</servlet-mapping>
```

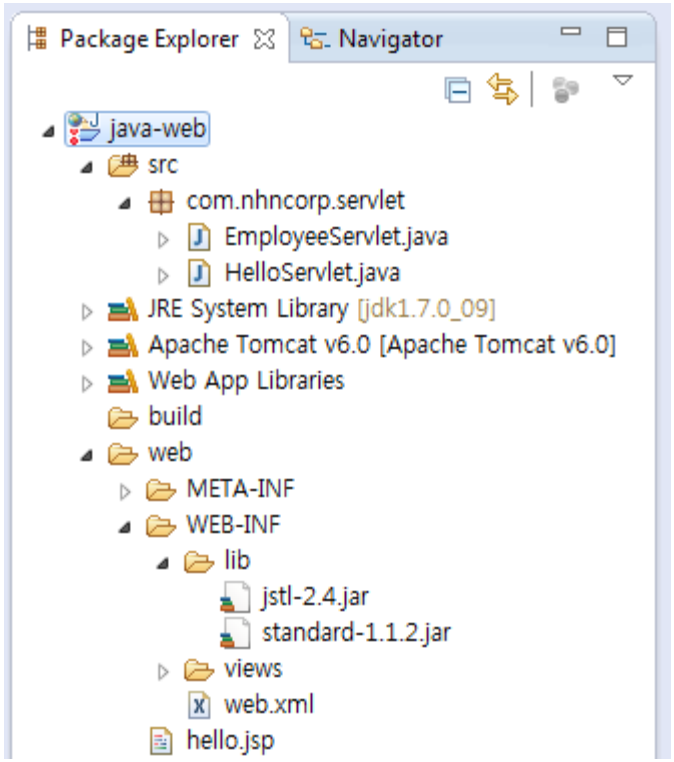
src/com/nhncorp/servlet/EmployeeServlet.java

```
public class EmployeeServlet extends HttpServlet {

	Map<String, String> employees = new HashMap<String, String>();

	public EmployeeServlet(){
		employees.put("kr90001", "우기명");
		employees.put("kr90002", "김두치");
		employees.put("kr90003", "김창주");
	}

	@Override
	protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
	IOException{
		String number = req.getParameter("number");
		String name = req.getParameter("name");

		if(number != null && name != null){
			employees.put(number,name);
		}

		req.setAttribute("employees", employees);
		RequestDispatcher dispatcher = req.getRequestDispatcher("/WEB-INF/views/employeeList.jsp");
		dispatcher.forward(req, resp);
	}

}
```

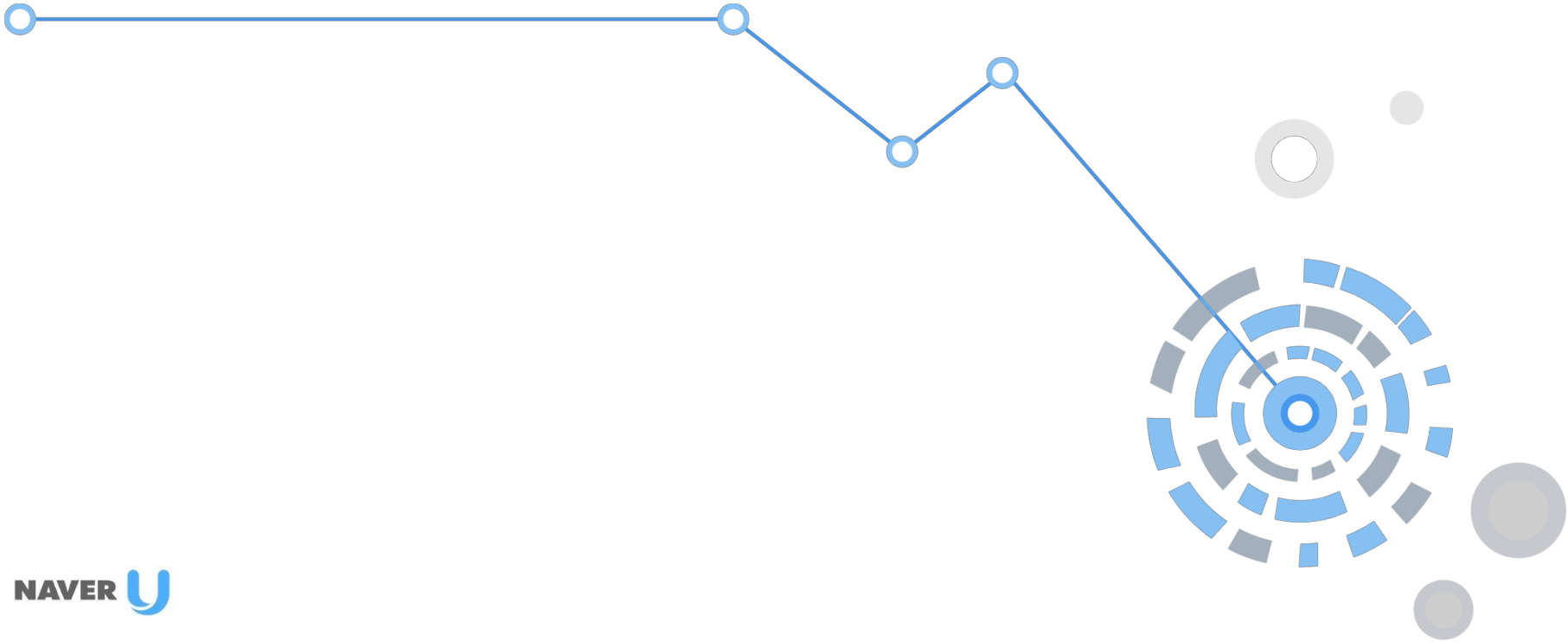
web/WEB-INF/views/employeeList.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">


<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
<title>Insert title here</title>
</head>
<c:forEach var="employee" items="${employees}">
   사번/이름 :  ${employee.key} / ${employee.value} <br/>
</c:forEach>
</html>
```

2.8 Servlet & Jsp 프로그래밍 - 실습



3. Error & Exception 분석



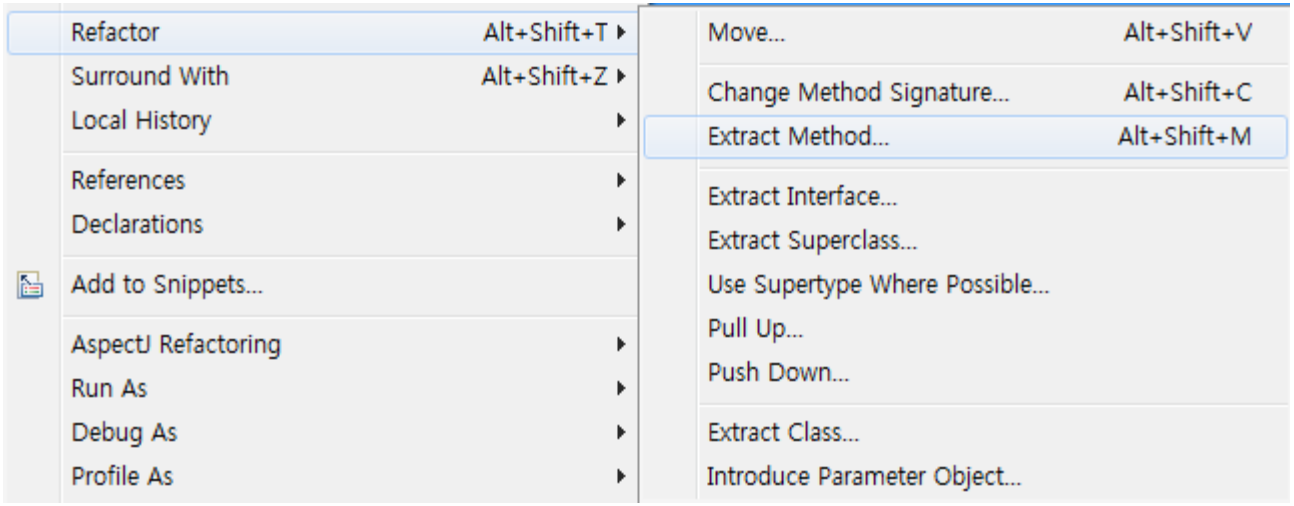
NAVER

코드 refactoring

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException{
    String number = req.getParameter("number");
    String name = req.getParameter("name");

    if(number != null && name != null)
    employees.put(number,name);

    req.setAttribute("employees", employees);
    RequestDispatcher dispatcher = req.getRequestDispatcher("/WEB-
    INF/views/employeeList.jsp");
    dispatcher.forward(req, resp);
}
```

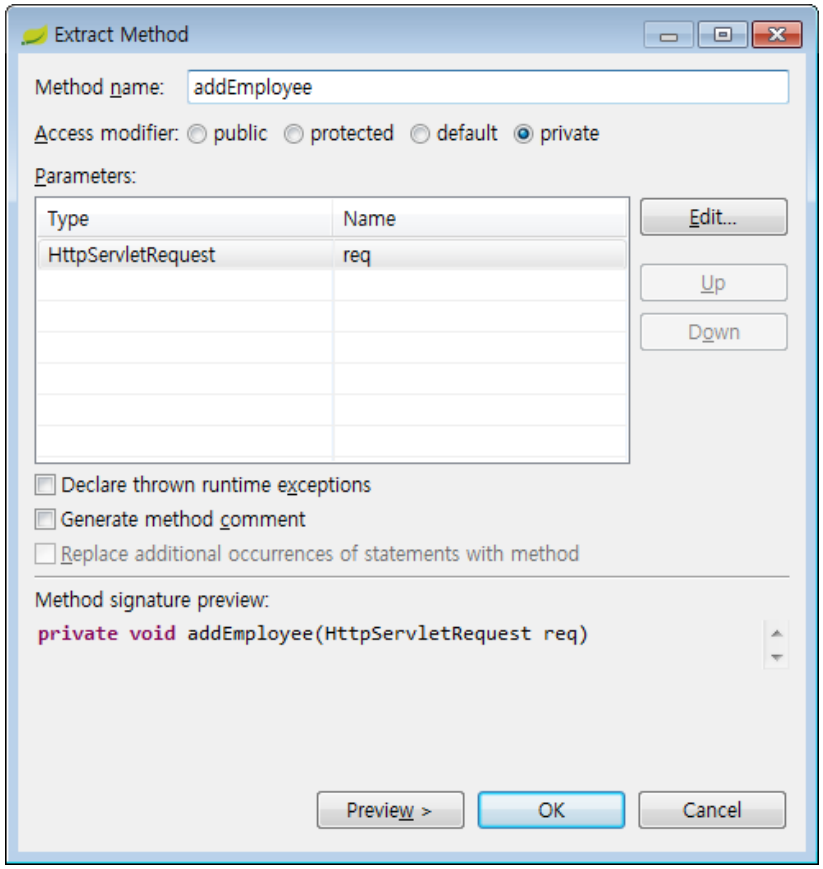
3. Error & Exception 분석하는 방법

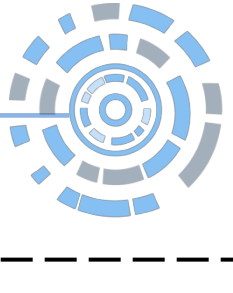
코드 refactoring



3. Error & Exception 분석하는 방법

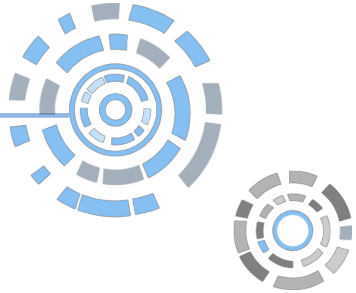
코드 refactoring





코드 refactoring

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException{
    addEmployee(req);
    RequestDispatcher dispatcher = req.getRequestDispatcher("/WEB-
    INF/views/employeeList.jsp");
    dispatcher.forward(req, resp);
}

private void addEmployee(HttpServletRequest req) {
    String number = req.getParameter("number");
    String name = req.getParameter("name");

    if(number != null && name != null)
        employees.put(number,name);

    req.setAttribute("employees", employees);
}
```



코드 refactoring

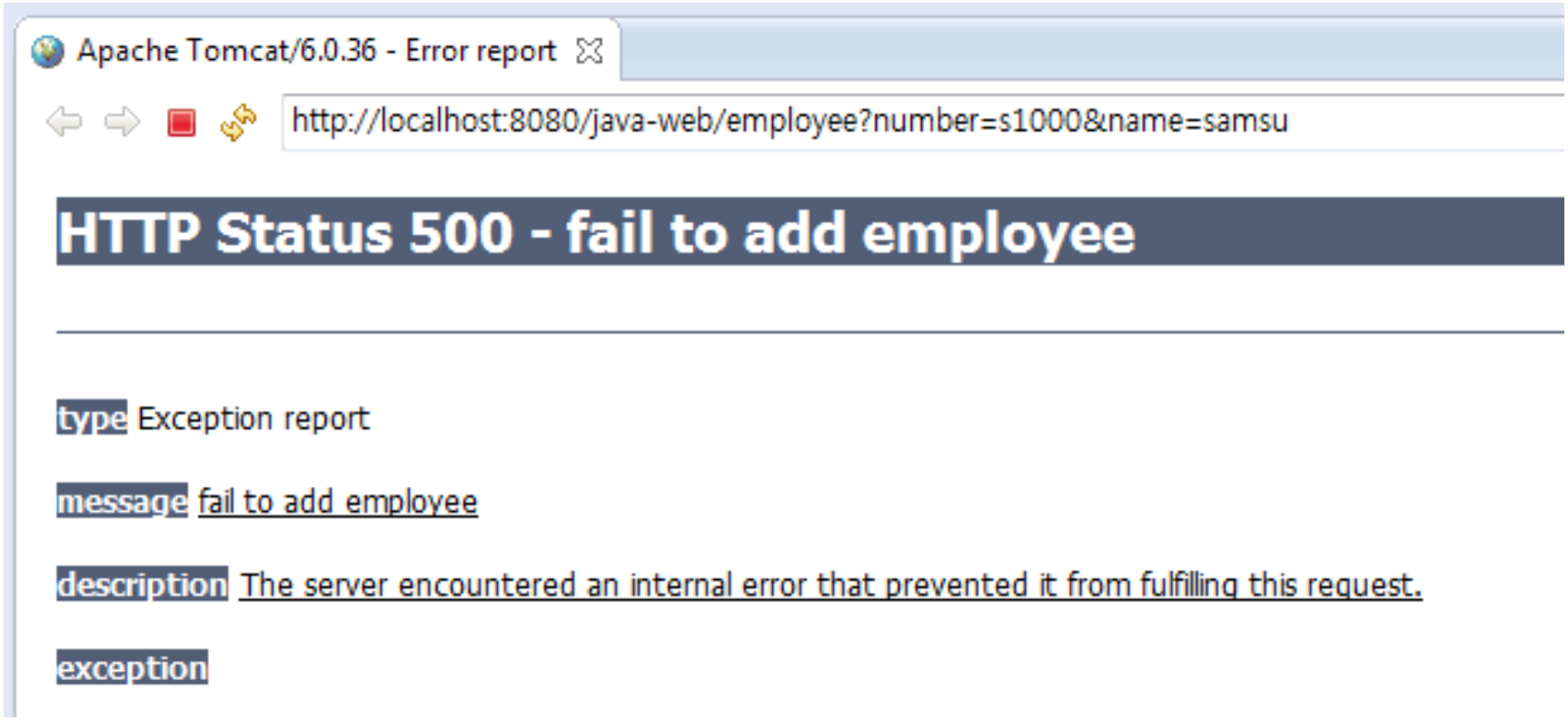
```
package com.nhncorp.exception;

public class ValidException extends Exception{
}
```

3. Error & Exception 분석하는 방법

코드 refactoring

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException{
      try {
      addEmployee(req);
      } catch (Exception e) {
            throw new RuntimeException("fail to add employee", e);
      }
      RequestDispatcher dispatcher = req.getRequestDispatcher("/WEB-INF/views/employeeList.jsp");
      dispatcher.forward(req, resp);
}

private void addEmployee(HttpServletRequest req) throws Exception {
      String number = req.getParameter("number");
      String name = req.getParameter("name");

      if(number != null && name != null){
            if(!number.startsWith("kr")){
                  throw new RuntimeException("The number is not valid");
            }
            employees.put(number,name);
      }
      req.setAttribute("employees", employees);
}
```

3. Error & Exception 분석하는 방법



3. Error & Exception 분석하는 방법

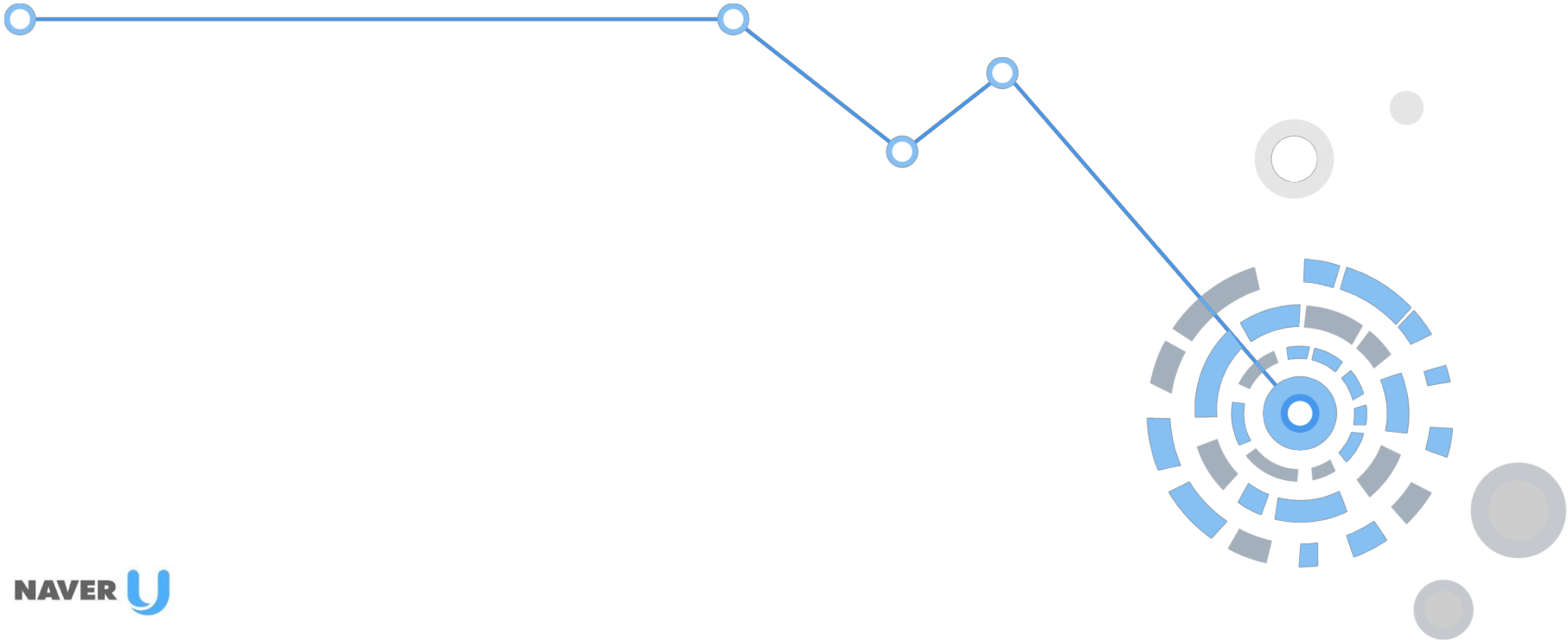
Error Log

```
java.lang.RuntimeException: fail to add employee
      at com.nhncorp.servlet.EmployeeServlet.doGet(EmployeeServlet.java:32)
      at javax.servlet.http.HttpServlet.service(HttpServlet.java:617)
      at javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
      at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:290)
      at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
      at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:233)
      at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:191)
      at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:127)
      at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:103)
      at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109)
      at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:293)
      at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:861)
      at org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.process(Http11Protocol.java:606)
      at org.apache.tomcat.util.net.JIoEndpoint$Worker.run(JIoEndpoint.java:489)
      at java.lang.Thread.run(Thread.java:722)
Caused by: java.lang.RuntimeException: The number is not valid
      at com.nhncorp.servlet.EmployeeServlet.addEmployee(EmployeeServlet.java:44)
      at com.nhncorp.servlet.EmployeeServlet.doGet(EmployeeServlet.java:28)
      ... 14 more
```

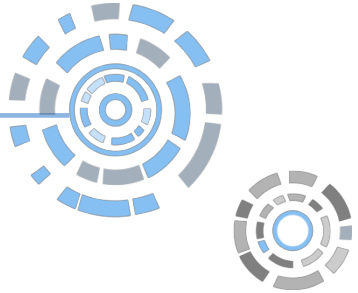
실습 3

앞에서 설명한 refactoring 코드를 적용하여 exception을 발생시켜보자

4. Logging



NAVER



✓ logging 이란?

시스템을 작동할 때 시스템의 작동상태의 기록 · 보존, 이용자의 습성조사 및 시스템 동작의 분석 등을 하기 위해 작동 중의 각종 정보를 기록하여 둘 필요가 있다. 이 기록을 만드는 것을 로깅이라 한다

✓ 필요성 : 문제 분석

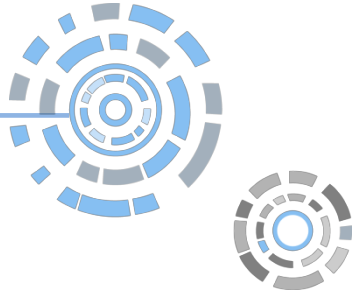
Log4j

자바 기반 로깅 유틸리티

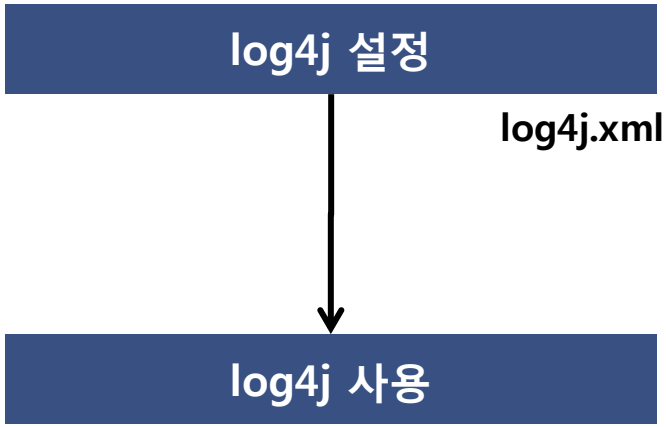
Log4j 예제 코드

```
Log log = LogFactory.getLog(클래스명.class);


log.info("In SimpleQuartzJob - executing its JOB at " + new Date() );
```

```
fatal()
error()
warn()
info()
debug()
```



로그 레벨	설명
fatal	아주 심각한 에러 시스템적으로 심각한 문제가 발생해서 어플리케이션 작동이 불가능할 경우가 해당하는데, 일반적으로는 어플리케이션에서는 사용할 일이 없다.
error	요청을 처리하는 중 문제가 발생한 상태
warn	요청을 처리하는 중 문제가 발생했지만, 불완전하게나마 처리가 가능한 상태
info	어플리케이션이 작동할 때 필요한 기본적인 정보
debug	디버깅, 즉 문제 해결을 하기 위한 상태 정보


log4j 설정
log4j.xml
log4j 사용

log4j 설정

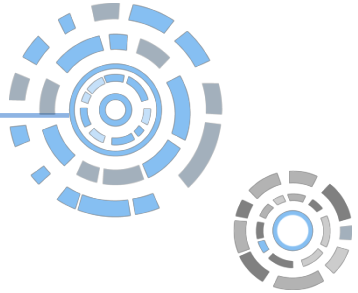
/src/main/resources/log4j.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j='http://jakarta.apache.org/log4j/'>
    <!--  appender -->
    <appender name="console" class="org.apache.log4j.ConsoleAppender">
        <layout class="org.apache.log4j.PatternLayout">
            <param name="ConversionPattern"
                   value="%d{yyyy-MM-dd HH:mm:ss} [%-5p](%F:%L) %m%n" />
        </layout>
    </appender>

    <!--  logger -->
    <logger name="com.nhncorp.lucy.edu.job">
        <level value="INFO" />
        <appender-ref ref="console" />
    </logger>

    <root>
        <level value="INFO" />
        <appender-ref ref="console" />
    </root>
</log4j:configuration>
```

log4j 사용

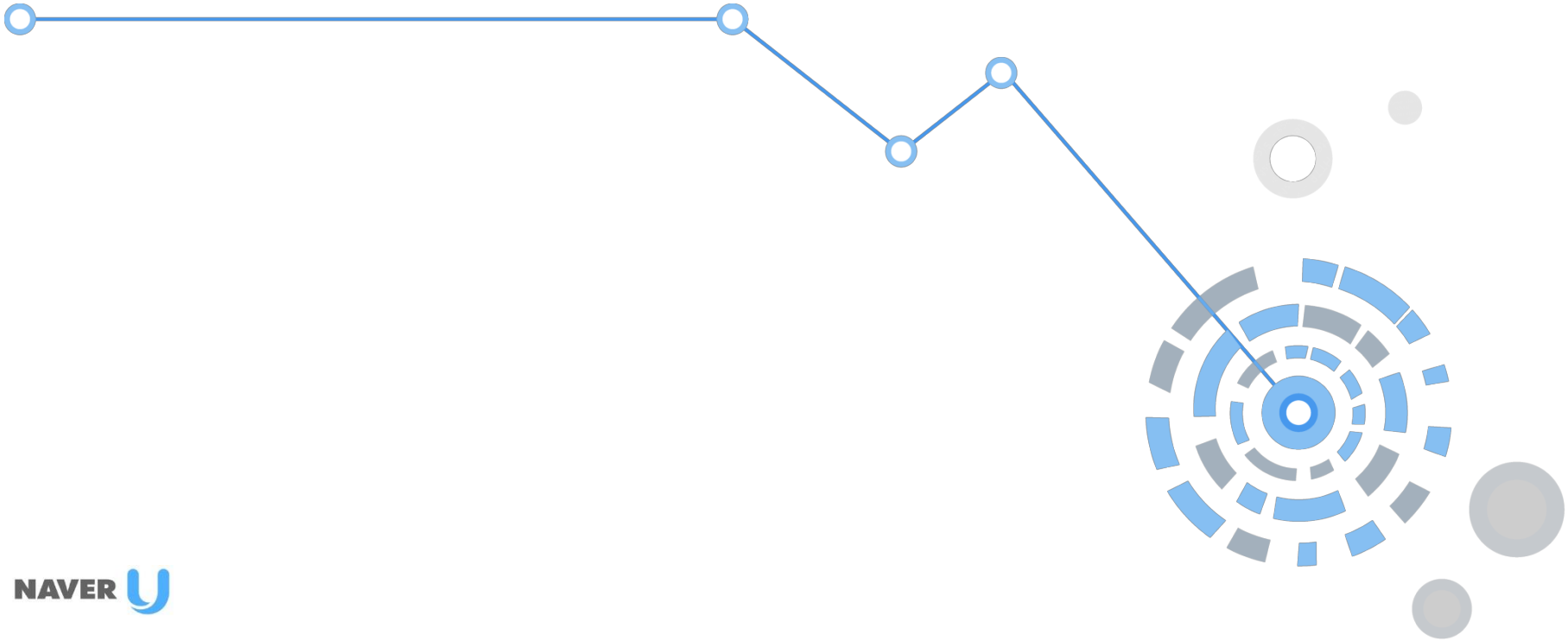
/src/main/java/com/nhncorp/lucy/edu/job/Log4jTestJob.java

```
package com.nhncorp.lucy.edu.job;

import java.util.Date;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.quartz.Job;
import org.quartz.JobExecutionContext;
import org.quartz.JobExecutionException;

public class Log4jTestJob implements Job {
    protected static Log log = LogFactory.getLog(Log4jTestJob.class);

    public void execute(JobExecutionContext ctx) throws JobExecutionException {
        log.info("In SimpleQuartzJob - executing its JOB at "
                + new Date() + " by " + ctx.getTrigger().getName());
    }
}
```



- 로그는 최대한 많이 찍으면 좋을까?
- 일반적으로 로그는 info level 이상으로 설정
- 로그를 통해서 문제를 분석
- 문제를 해결할 때 로그를 추가
- 로그를 파악할 때 동시성 문제 고려 필요
  - thread를 구분하기 위해서 thread id 출력

Thank you.



NAVER