

# 서버 프로그래밍 (스프링 프레임 워크)

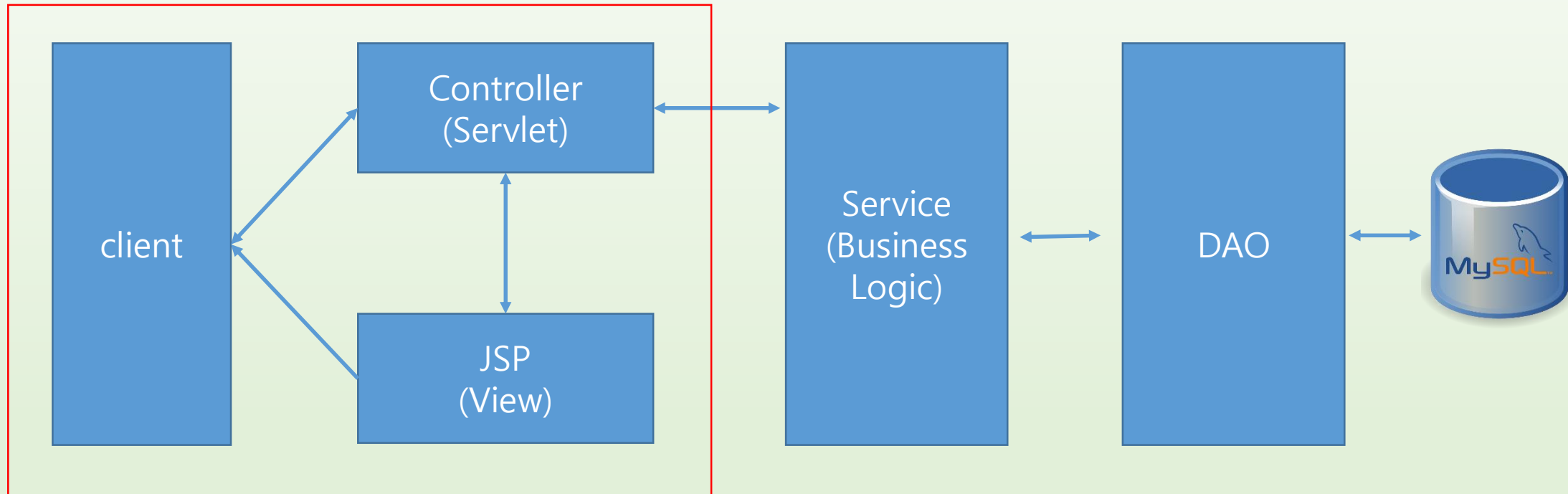
# 목차

- 지난 시간 내용 복습
- 스프링 MVC
- 화면과 연결하기
- 실습
- @RequestParam

## 복습문제

- 새로운 스프링 프로젝트를 생성해서 과제를 진행
- 주소에 gugudan 호출시
- 구구단 전체를 출력해주는 프로그램 작성

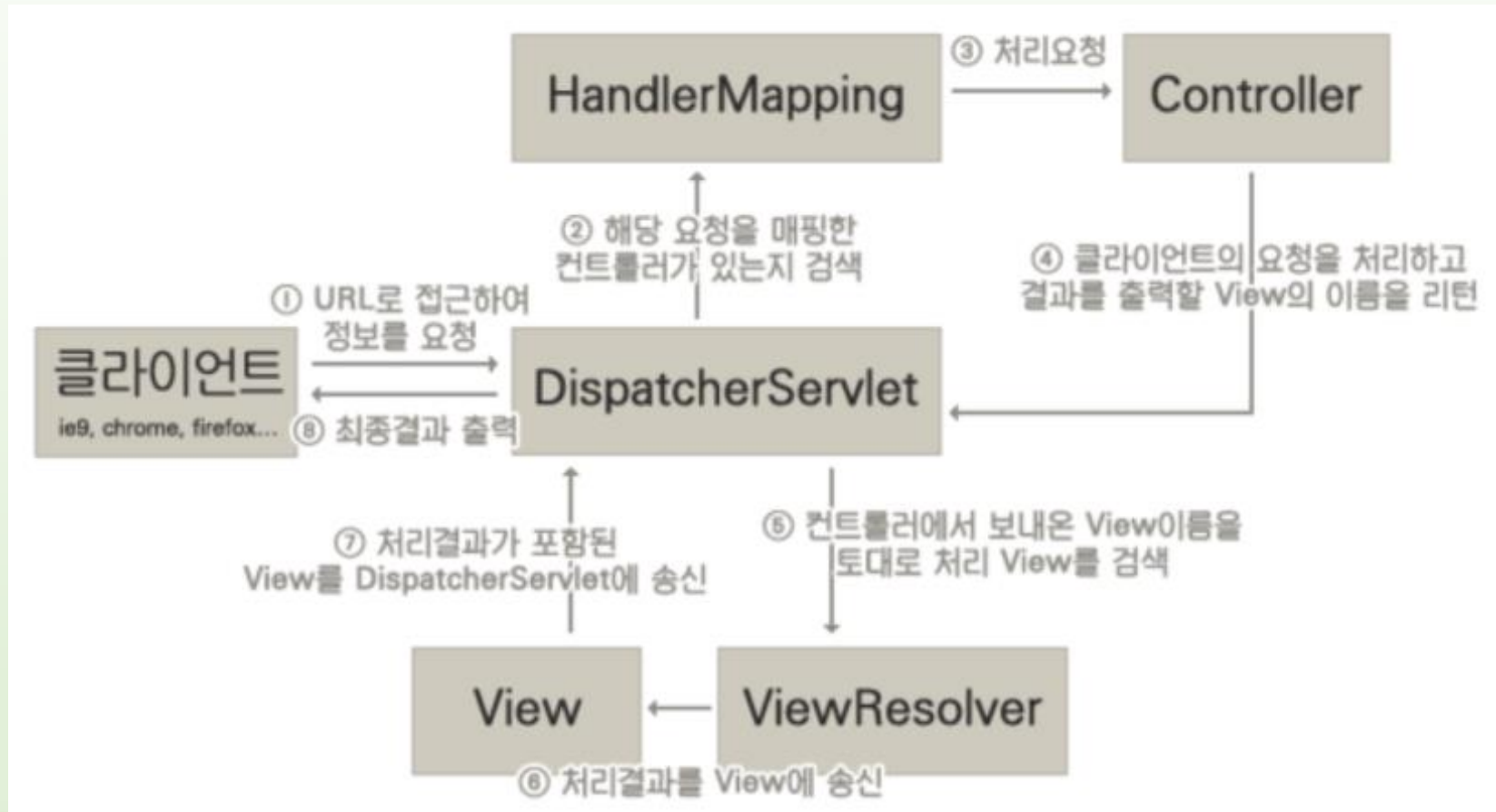
# Spring 프레임워크 구조



# Spring MVC 구성 주요 컴포넌트

- **DispatcherServlet** : 모든 요청을 최초로 받아 들이는 서블릿
- **HandlerMapping** : 클라이언트의 요청을 처리할 컨트롤러를 찾는 작업
- **Controller** : 클라이언트의 요청을 처리하
- **ModelAndView** : View에게 값을 전달 하기 위해 사용되는 객체
- **ViewResolver** : View를 찾는 작업을 처리
- **View** : 화면구성

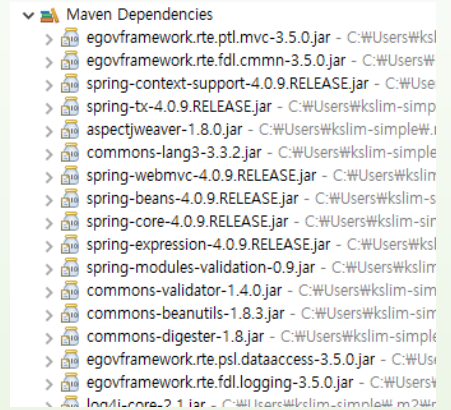
# Spring MVC 구성 주요 컴포넌트 구조



출처: <http://egloos.zum.com/springmvc/v/504151>

# 스프링의 동작원리

- 라이브러리 : 어플리케이션의 실행을 위해 특정 기능들을 미리 구현해 놓은 부분
- 설정 파일 : 라이브러리를 해당 프로젝트에 맞게 활용하기 위하여 설정되는 값 정보
- 어노테이션 : 자바코드에서 스프링프레임워크의 기능을 작동 시키기 위한 @로 시작하는 회색글자



```
@Controller
public class EgovSampleController {

    /** EgovSampleService */
    @Resource(name = "sampleService")
    private EgovSampleService sampleService;

    /** EgovPropertyService */
    @Resource(name = "propertiesService")
    protected EgovPropertyService propertiesService;

    /** Validator */
    @Resource(name = "beanValidator")
    protected DefaultBeanValidator beanValidator;
}
```

```
<?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"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd">

    <context:component-scan base-package="egovframework">
        <context:exclude-filter type="annotation" expression="org.springframework.stereotype.Controller" />
    </context:component-scan>

    <bean id="messageSource" class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
        <property name="basenames">
            <list>
                <value>classpath:/egovframework/message/message-common</value>
                <value>classpath:/egovframework/rte/fdl/idgnr/messages/idgnr</value>
                <value>classpath:/egovframework/rte/fdl/property/messages/properties</value>
            </list>
        </property>
    </bean>
</beans>
```

# 실습



# JSP와 컨트롤러 연결(pom.xml)

- 위치 주의!!!

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-test</artifactId>  
  <scope>test</scope>  
</dependency>
```

```
<!-- jsp 사용을 위한 추가 -->
```

```
<dependency>  
  <groupId>javax.servlet</groupId>  
  <artifactId>jstl</artifactId>  
</dependency>
```

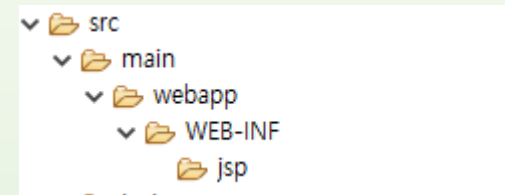
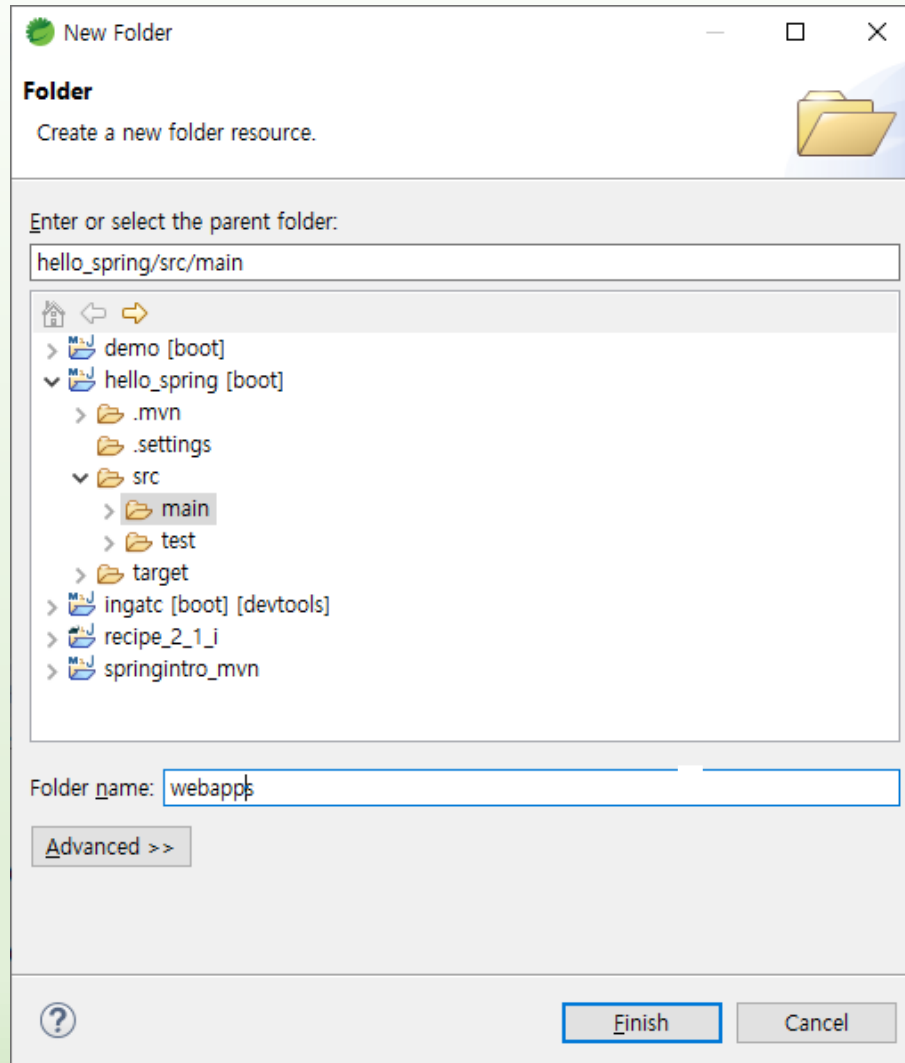
```
<dependency>  
  <groupId>org.apache.tomcat.embed</groupId>  
  <artifactId>tomcat-embed-jasper</artifactId>  
  <scope>provided</scope>  
</dependency>
```

## 설정 추가

- application.properties

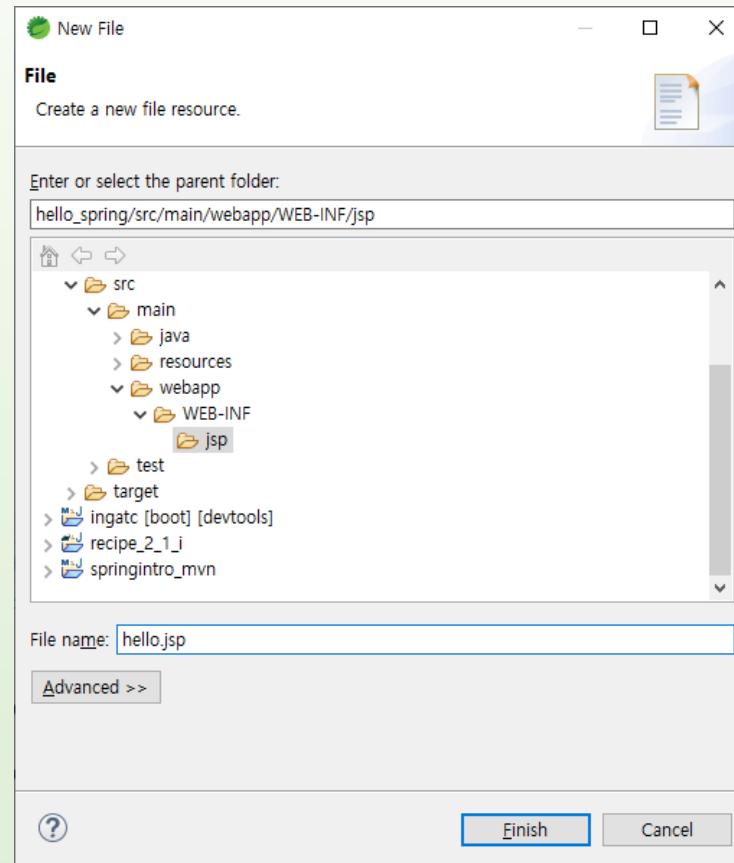
```
#jsp설정  
spring.mvc.view.prefix=/WEB-INF/jsp/  
spring.mvc.view.suffix=.jsp|
```

# 폴더 구조 생성



# 파일

- 파일생성



## 샘플 jsp 파일 작성

```
<%@ page language="java" contentType="text/html; charset=UTF-8"  
    pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="UTF-8">  
  <title>hello</title>  
</head>  
<body>  
  안녕하세요!!  
</body>  
</html>
```

# 컨트롤러 생성

**New Java Class**

**Java Class**  
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

---

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

☒ Inherited abstract methods

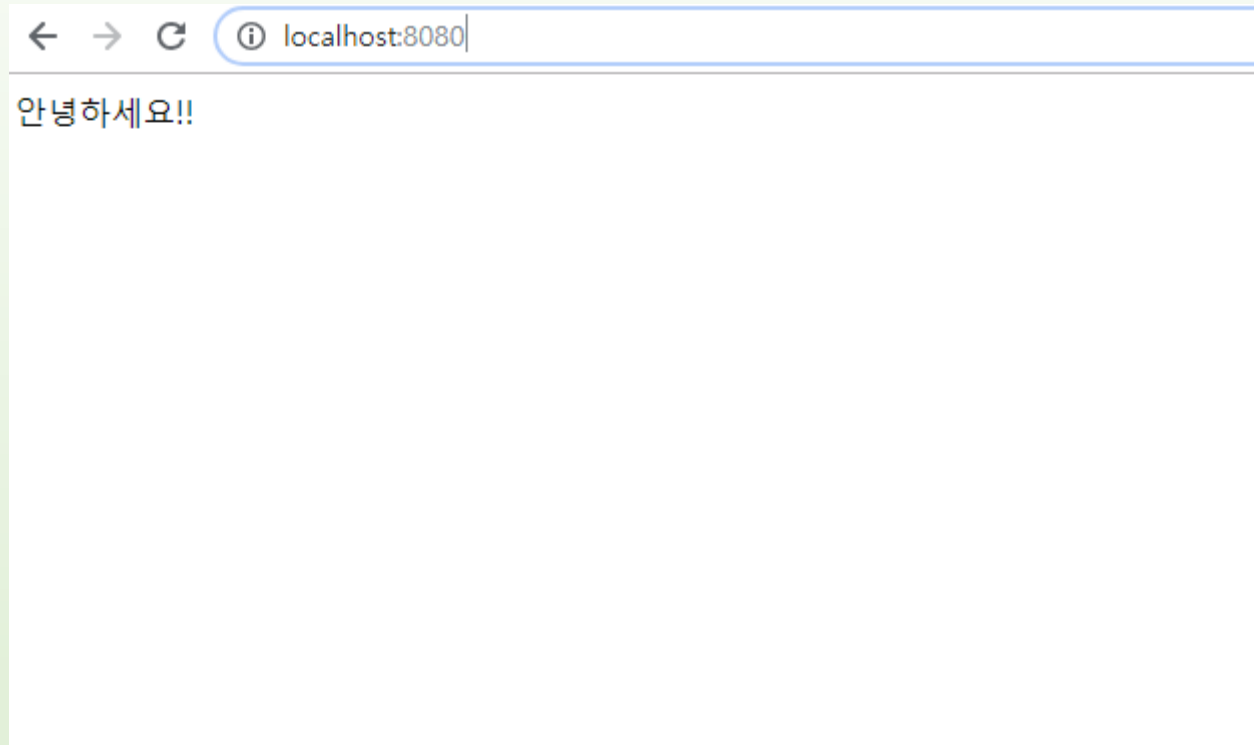
Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

# Controller Mapping 정보 작성

```
1 package kr.ac.inhatc.mvc.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5
6 @Controller
7 public class JspController {
8     @RequestMapping("/")
9     public String index() {
10         return "hello";
11     }
12 }
13
```

# Hello World!!

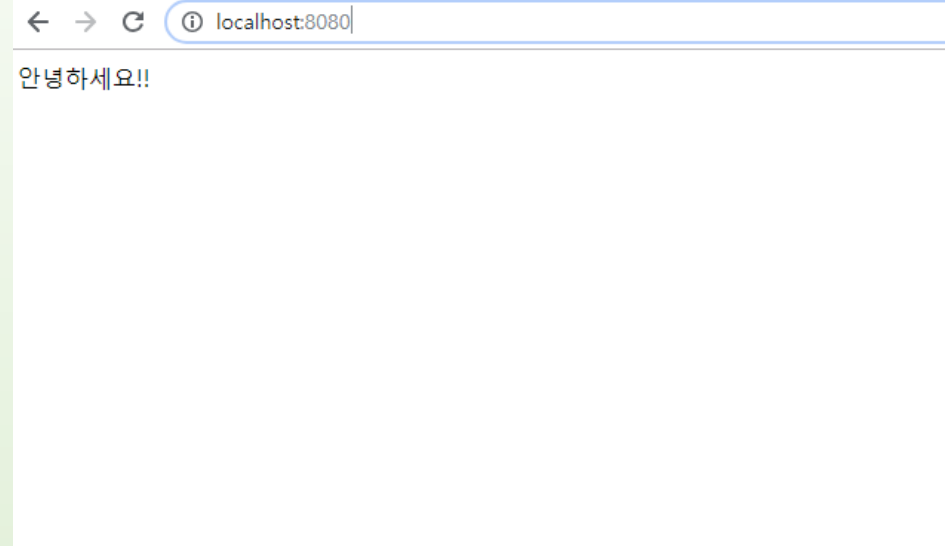




# 수정시 파일이 안바뀐다??

- 서버를 재기동

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>hello</title>
</head>
<body>
안녕하세요!!@@@
</body>
</html>
```



## 조금 더 편하게 하려면(pom.xml)

- 위치 주의

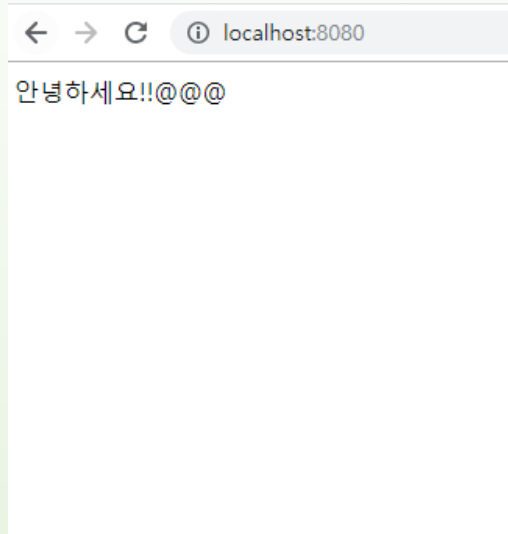
```
38      <!-- jsp 사용을 위한 추가 -->
39      <dependency>
40          <groupId>javax.servlet</groupId>
41          <artifactId>jstl</artifactId>
42      </dependency>
43
44      <dependency>
45          <groupId>org.apache.tomcat.embed</groupId>
46          <artifactId>tomcat-embed-jasper</artifactId>
47          <scope>provided</scope>
48      </dependency>
49
50      <!-- 자동리로드 -->
51      <dependency>
52          <groupId>org.springframework.boot</groupId>
53          <artifactId>spring-boot-devtools</artifactId>
54          <scope>true</scope>
55      </dependency>
56
57  </dependencies>
58
59  <build>
```

## 조금 더 편하게 하려면(application.properties)

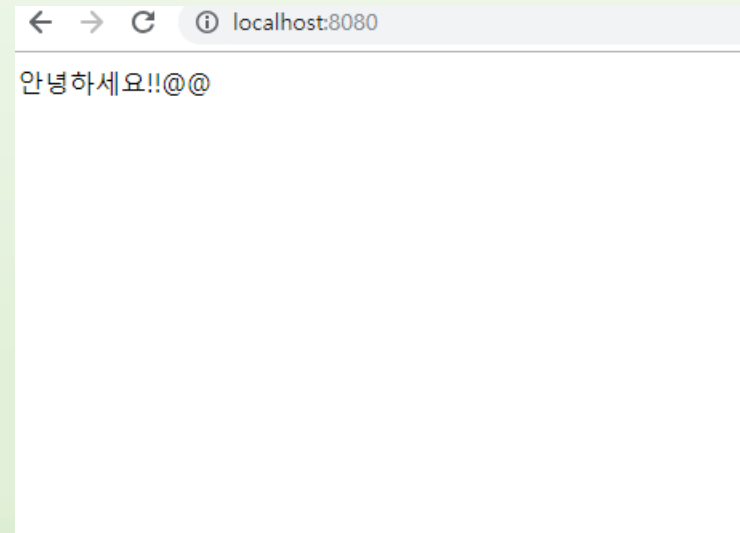
```
#jsp설정
spring.mvc.view.prefix=/WEB-INF/jsp/
spring.mvc.view.suffix=.jsp

#자동 리로드 설정
spring.devtools.livereload.enabled=true
spring.freemarker.cache=false
```

# 서버 재기동



```
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>hello</title>
8 </head>
9 <body>
10 안녕하세요!!@@
11 </body>
12 </html>
```



## 컨트롤러에서 view로 데이터 전달하기

- model을 활용한 컨트롤러에 데이터 전달방법
- Controller

```
@RequestMapping("board")  
public String board(Model model,String id) {  
  
    model.addAttribute("id",id);  
    return "board";  
}
```

- jsp

```
<title>${id}</title>
```

## 실습 1. 페이지에 구구단 출력

← → ↻ ⓘ localhost:8080/gugudan?num=2

구구단 : 2\*1=2 2\*2=4 2\*3=6 2\*4=8 2\*5=10 2\*6=12 2\*7=14 2\*8=16 2\*9=18

## 실습 2. 1부터 100까지의 합

1부터 100까지의 합 : 5050

## 실습3. 1부터 입력한 숫자까지의 합

1부터 100까지의 합 : 5050



# 정적파일 활용하기(1/3)





- 
- 
- 
-

과목 목록

Name ▼

[검색](#)

| No | 과목명                      | 학년  | 개설여부 | 설명   | 등록자 |
|----|--------------------------|-----|------|------|-----|
| 1  | <a href="#">C</a>        | 2학년 | Y    | 기초과목 | 관리자 |
| 2  | <a href="#">JAVA</a>     | 2학년 | Y    | 심화과목 | 관리자 |
| 3  | <a href="#">node</a>     | 2학년 | Y    | 응용과목 | 관리자 |
| 4  | <a href="#">자료구조</a>     | 2학년 | Y    | 고급과목 | 관리자 |
| 5  | <a href="#">서버프로그래밍</a>  | 3학년 | Y    | 필수과목 | 관리자 |
| 6  | <a href="#">진로상담</a>     | 3학년 | Y    | 일반과목 | 관리자 |
| 7  | <a href="#">심화C</a>      | 2학년 | Y    | 일반과목 | 관리자 |
| 8  | <a href="#">심화 JAVA</a>  | 1학년 | Y    | 일반과목 | 관리자 |
| 9  | <a href="#">고급프로그래밍</a>  | 2학년 | Y    | 일반과목 | 관리자 |
| 10 | <a href="#">서버프로그래밍2</a> | 1학년 | Y    | 일반과목 | 관리자 |

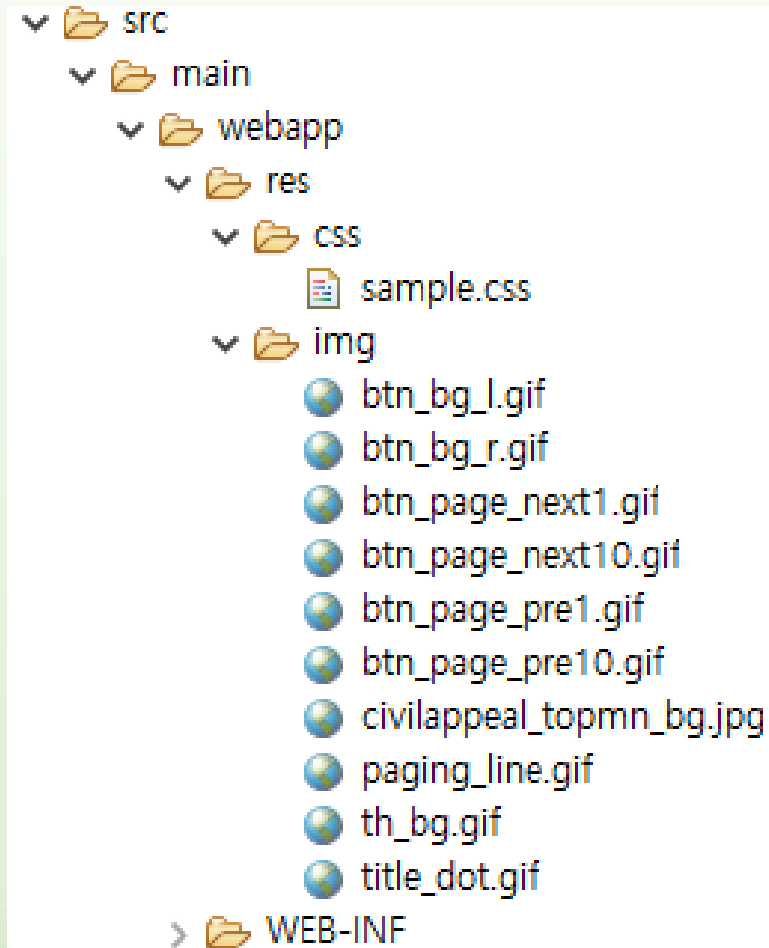


[1](#)
[2](#)
[3](#)
[4](#)
[5](#)
[6](#)
[7](#)
[8](#)
[9](#)
[10](#)



[등록](#)

-

## 정적 파일 활용하기(2/3)

- 구조 생성



## 정적 파일 활용하기(3/3)

- application.properties

#jsp설정

```
spring.mvc.view.prefix=/WEB-INF/jsp/
```

```
spring.mvc.view.suffix=.jsp
```

#자동 리로드 설정

```
spring.devtools.livereload.enabled=true
```

```
spring.freemarker.cache=false
```

#resource 설정

```
spring.mvc.static-path-pattern=/resource/**
```

# 적용화면

## 과목 목록

Name ▼

검색

| No | 과목명      | 학년  | 개설여부 | 설명   | 등록자 |
|----|----------|-----|------|------|-----|
| 1  | C        | 2학년 | Y    | 기초과목 | 관리자 |
| 2  | JAVA     | 2학년 | Y    | 심화과목 | 관리자 |
| 3  | node     | 2학년 | Y    | 응용과목 | 관리자 |
| 4  | 자료구조     | 2학년 | Y    | 고급과목 | 관리자 |
| 5  | 서버프로그래밍  | 3학년 | Y    | 필수과목 | 관리자 |
| 6  | 진로상담     | 3학년 | Y    | 일반과목 | 관리자 |
| 7  | 심화C      | 2학년 | Y    | 일반과목 | 관리자 |
| 8  | 심화 JAVA  | 1학년 | Y    | 일반과목 | 관리자 |
| 9  | 고급프로그래밍  | 2학년 | Y    | 일반과목 | 관리자 |
| 10 | 서버프로그래밍2 | 1학년 | Y    | 일반과목 | 관리자 |

<< < 1 2 3 4 5 6 7 8 9 10 > >>

등록

## @RequestParam

- 넘어오는 데이터를 받아오기 위해 사용되는 어노테이션
- 파라미터가 변수의 이름이 다를 경우도 활용
- 예외 방지를 위한 DefaultValue지정 가능
- 필수 여부를 선택가능

```
@RequestMapping("board")
public String board(Model model, @RequestParam("test") String id) {
    model.addAttribute("id",id);
    return "board";
}

@RequestMapping("board2")
public String board2(Model model, @RequestParam(value="test", required=false, defaultValue="100") String id) {
    model.addAttribute("id",id);
    return "board";
}
```

# Q & A