

# Spring MVC Setting

---

524730-1  
2021년 봄학기  
5/12/2021  
박경신

# Spring Framework

---

## □ Spring Framework

- 웹 서버를 구현하기 위한 어플리케이션 프레임워크
- IoC (Inversion of Control) 제어의 역행
  - IoC는 메소드나 객체의 호출을 개발자가 결정하는 것이 아니라 외부에서 결정되는 것을 의미
  - Spring IoC 컨테이너는 ApplicationContext 또는 BeanFactory 중 하나를 사용, Bean을 만들고, 의존성을 엮어주며 제공해주는 역할을 수행
- DI (Dependency Injection) 의존성 주입을 통한 객체 간의 관계
  - IoC가 일어날 때 스프링 내부에 있는 객체들(Beans) 간의 관계를 컨테이너가 자동적으로 연결해주는 것
- AOP (Aspect Oriented Programming) 관점 지향 프로그래밍
  - 시스템 보안, 로그, 트랜잭션 등 스프링에서는 이러한 '횡단 관심사'를 모듈로 분리해서 제작 가능. 공통 관심 사항을 프로그래밍하여 이 코드를 여러 코드에 짜 넣음(weave). AspectJ로 이미 사용되어 왔음.
- POJO (Plain Old Java Object) 기반 구조
  - 객체 간의 관계 구성 시 POJO (즉, Bean)의 구성만으로 가능
- MVC (Model-View-Controller) 구조

# Spring MVC Setting

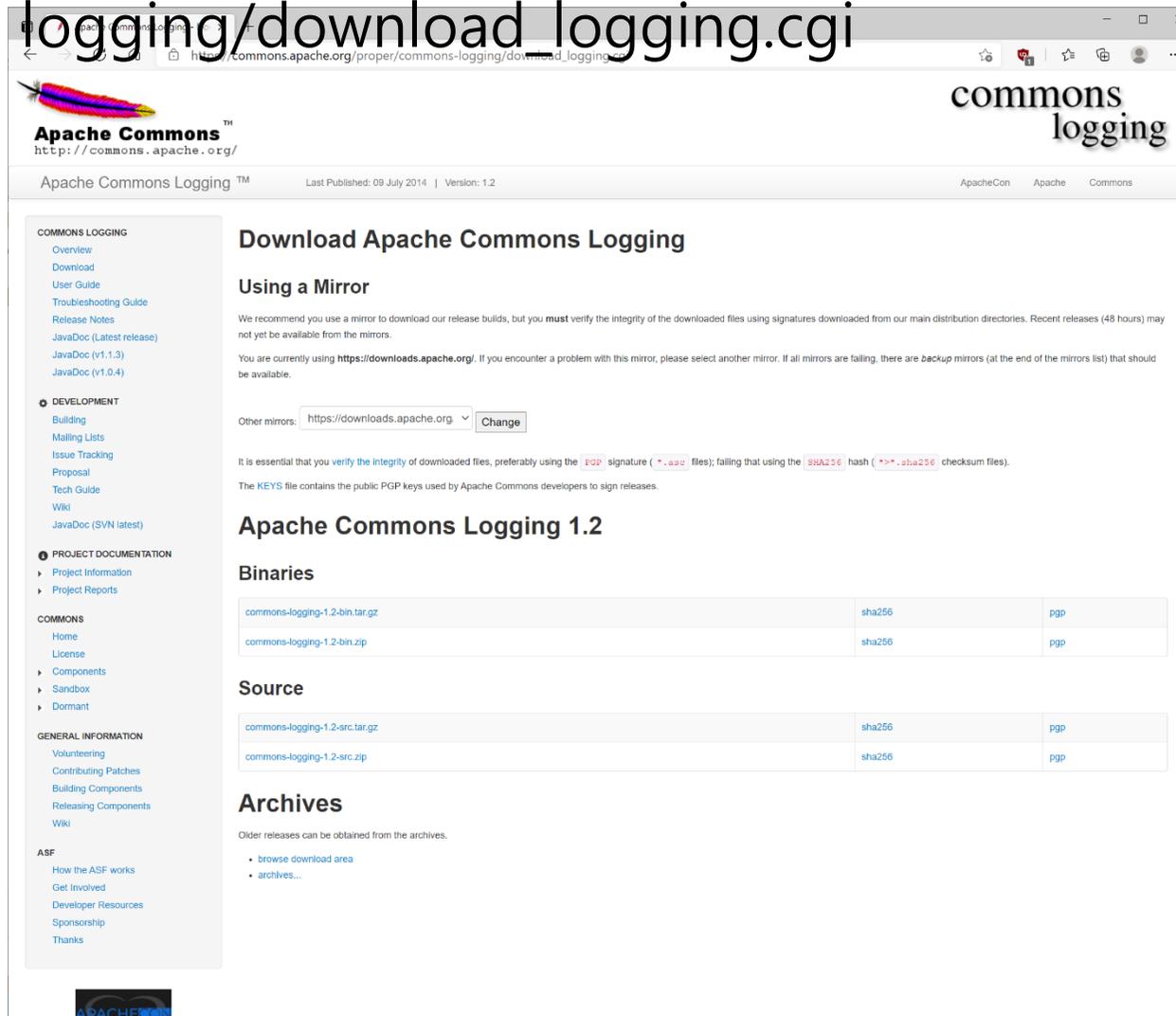
---

## □ Spring MVC Quick Start

- OpenJDK 15.0.2 & Eclipse IDE 2020-12
- Tomcat 9.0.43
  - Eclipse Server Setting: Eclipse -> Window -> Preferences -> Server -> Runtime Environments -> Tomcat9
- Common Logging 1.2
  - [https://commons.apache.org/proper/commons-logging/download\\_logging.cgi](https://commons.apache.org/proper/commons-logging/download_logging.cgi)
  - Download commons-logging-1.2-bin.zip
- Maven 3.6.3 프로젝트 관리 도구
  - <https://maven.apache.org/download.cgi>
  - Download apache-maven-3.6.3-bin.zip
- Spring Tool Suite (STS)
  - <https://spring.io/tools/> Spring Tools for Eclipse
  - Install Eclipse->Help->"Eclipse Marketplace" -> STS -> Spring Tools 3.9.14.RELEASE

# Commons Logging

- [https://commons.apache.org/proper/commons-logging/download\\_logging.cgi](https://commons.apache.org/proper/commons-logging/download_logging.cgi)



The screenshot shows the Apache Commons Logging download page. The page features a navigation sidebar on the left with categories like COMMONS LOGGING, DEVELOPMENT, PROJECT DOCUMENTATION, COMMONS, GENERAL INFORMATION, and ASF. The main content area is titled "Download Apache Commons Logging" and includes a "Using a Mirror" section with a mirror selection dropdown set to "https://downloads.apache.org". Below this is a "Binaries" table listing download links for tar.gz and zip files, along with their SHA256 hashes and PGP signatures. A "Source" table lists source code download links. The "Archives" section mentions that older releases can be obtained from archives.

## Download Apache Commons Logging

### Using a Mirror

We recommend you use a mirror to download our release builds, but you **must** verify the integrity of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet be available from the mirrors.

You are currently using <https://downloads.apache.org/>. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are *backup* mirrors (at the end of the mirrors list) that should be available.

Other mirrors:

It is essential that you [verify the integrity](#) of downloaded files, preferably using the **PGP** signature (`*.asc` files); falling that using the **SHA256** hash (`**.sha256` checksum files).

The **KEYS** file contains the public PGP keys used by Apache Commons developers to sign releases.

## Apache Commons Logging 1.2

### Binaries

<a href="#">commons-logging-1.2-bin.tar.gz</a>	sha256	pgp
<a href="#">commons-logging-1.2-bin.zip</a>	sha256	pgp

### Source

<a href="#">commons-logging-1.2-src.tar.gz</a>	sha256	pgp
<a href="#">commons-logging-1.2-src.zip</a>	sha256	pgp

### Archives

Older releases can be obtained from the archives.

- [browse download area](#)
- [archives...](#)

# Maven

□ <https://maven.apache.org/download.cgi>

The screenshot shows the Apache Maven Project website for downloading version 3.6.3. The page includes a navigation menu on the left, a main content area with a download button, system requirements, and a table of download links with checksums and signatures.

## Downloading Apache Maven 3.6.3

Apache Maven 3.6.3 is the latest release and recommended version for all users.

The currently selected download mirror is <https://downloads.apache.org/>. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are *backup* mirrors (at the end of the mirrors list) that should be available. You may also consult the [complete list of mirrors](#).

Other mirrors:

### System Requirements

<b>Java Development Kit (JDK)</b>	Maven 3.3+ require JDK 1.7 or above to execute - they still allow you to build against 1.3 and other JDK versions by <a href="#">Using Toolchains</a>
<b>Memory</b>	No minimum requirement
<b>Disk</b>	Approximately 10MB is required for the Maven installation itself. In addition to that, additional disk space will be used for your local repository. The size of your local repository will vary depending on usage but expect at least 500MB.
<b>Operating System</b>	No minimum requirement. Start up scripts are included as shell scripts and Windows batch files.

### Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [installation instructions](#). Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public **KEYS** used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	<a href="#">apache-maven-3.6.3-bin.tar.gz</a>	<a href="#">apache-maven-3.6.3-bin.tar.gz.sha512</a>	<a href="#">apache-maven-3.6.3-bin.tar.gz.asc</a>
Binary zip archive	<a href="#">apache-maven-3.6.3-bin.zip</a>	<a href="#">apache-maven-3.6.3-bin.zip.sha512</a>	<a href="#">apache-maven-3.6.3-bin.zip.asc</a>
Source tar.gz archive	<a href="#">apache-maven-3.6.3-src.tar.gz</a>	<a href="#">apache-maven-3.6.3-src.tar.gz.sha512</a>	<a href="#">apache-maven-3.6.3-src.tar.gz.asc</a>
Source zip archive	<a href="#">apache-maven-3.6.3-src.zip</a>	<a href="#">apache-maven-3.6.3-src.zip.sha512</a>	<a href="#">apache-maven-3.6.3-src.zip.asc</a>

- Release Notes
- Reference Documentation
- Apache Maven Website As Documentation Archive
- All current release sources (plugins, shared libraries,...) available at <https://downloads.apache.org/maven/>
- latest source code from source repository
- Distributed under the Apache License, version 2.0

### Previous Releases

It is strongly recommended to use the latest release version of Apache Maven to take advantage of newest features and bug fixes.

If you still want to use an old version you can find more information in the [Maven Releases History](#) and can download files from the [archives](#) for versions 3.0.4+ and [legacy archives](#) for earlier

# Maven

## □ Maven

The screenshot shows the Apache Maven Project website at <http://maven.apache.org/install.html>. The page title is "Installing Apache Maven". The Apache Maven Project logo is at the top left, and the "Maven" logo is at the top right. The breadcrumb navigation shows "Apache / Maven / Installing Apache Maven". The page content includes a sidebar with navigation links, a main heading "Installing Apache Maven", a description of the installation process, detailed steps for Windows and Unix-based systems, and specific tips for Windows and Unix-based operating systems.

### Installing Apache Maven

The installation of Apache Maven is a simple process of extracting the archive and adding the 'bin' folder with the 'mvn' command to the 'PATH'.

Detailed steps are:

- Ensure `JAVA_HOME` environment variable is set and points to your JDK installation
- Extract distribution archive in any directory

```
1. unzip apache-maven-3.6.3-bin.zip
```

or

```
1. tar xzvf apache-maven-3.6.3-bin.tar.gz
```

Alternatively use your preferred archive extraction tool.

- Add the `bin` directory of the created directory `apache-maven-3.6.3` to the `PATH` environment variable
- Confirm with `mvn -v` in a new shell. The result should look similar to

```
1. Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
2. Maven home: /opt/apache-maven-3.6.3
3. Java version: 1.8.0_45, vendor: Oracle Corporation
4. Java home: /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/jre
5. Default locale: en_US, platform encoding: UTF-8
6. OS name: "mac os x", version: "10.8.5", arch: "x86_64", family: "mac"
```

### Windows Tips

- Check environment variable value e.g.

```
1. echo %JAVA_HOME%
2. C:\Program Files\Java\jdk1.7.0_51
```

- Adding to `PATH`: Add the unpacked distribution's bin directory to your user PATH environment variable by opening up the system properties (WinKey + Pause), selecting the "Advanced" tab, and the "Environment Variables" button, then adding or selecting the `PATH` variable in the user variables with the value `C:\Program Files\apache-maven-3.6.3\bin`. The same dialog can be used to set `JAVA_HOME` to the location of your JDK, e.g. `C:\Program Files\Java\jdk1.7.0_51`
- Open a new command prompt (Winkey + R then type `cmd`) and run `mvn -v` to verify the installation.

### Unix-based Operating System (Linux, Solaris and Mac OS X) Tips

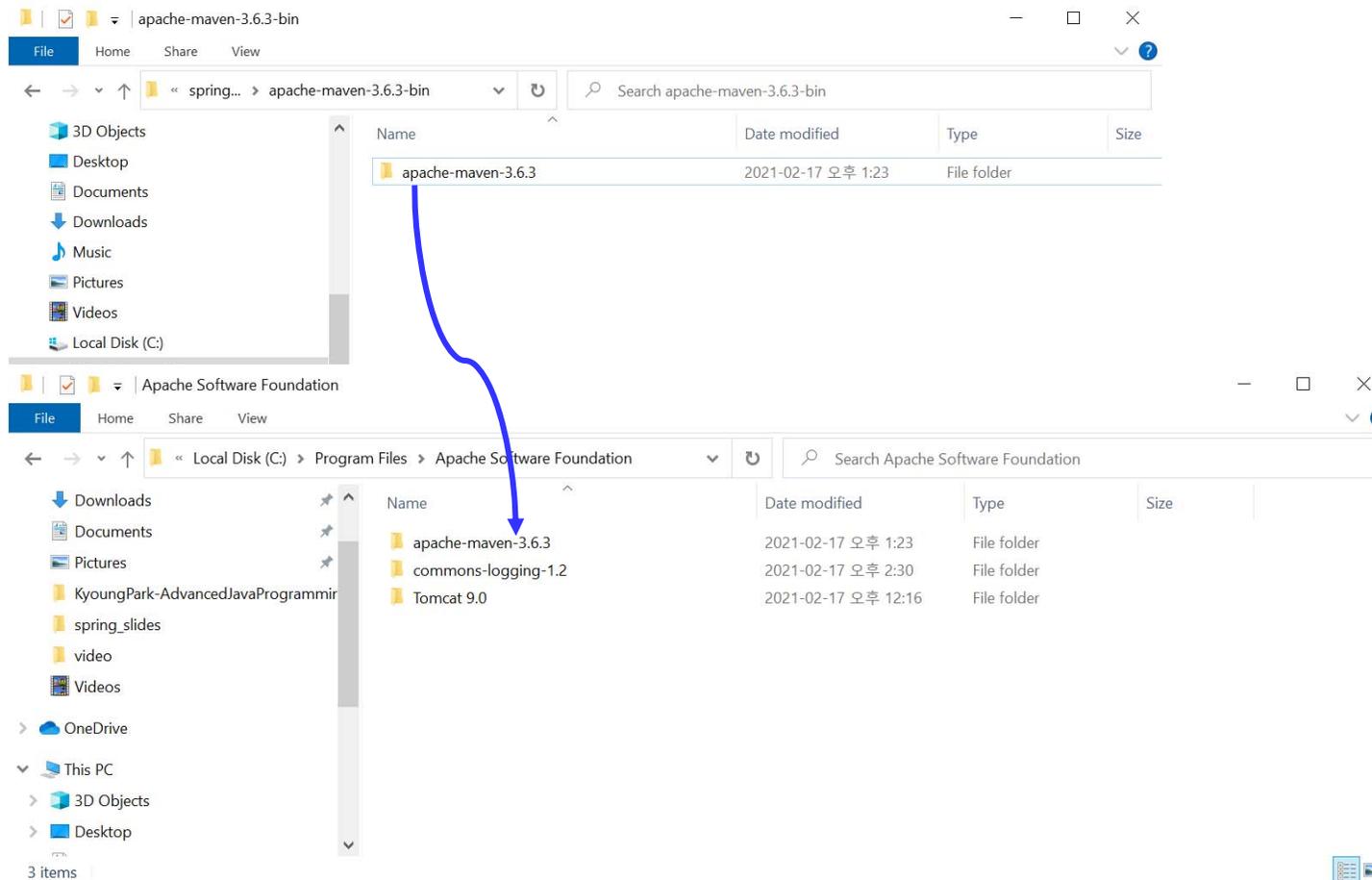
- Check environment variable value

```
1. echo $JAVA_HOME
2. /Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home
```

- Adding to `PATH`

# Maven

- Unzip apache-maven-3.6.3-bin.zip & Copy "apache-maven-3.6.3" to "C:\Program Files\Apache Software Foundation"



# Maven

## ▣ Set path environment variable

The image displays two screenshots of Windows system variable configuration dialogs.

The left dialog, titled "New System Variable", shows the configuration for a new system variable:

- Variable name: MAVEN\_HOME
- Variable value: C:\Program Files\Apache Software Foundation\apache-maven-3.6.3

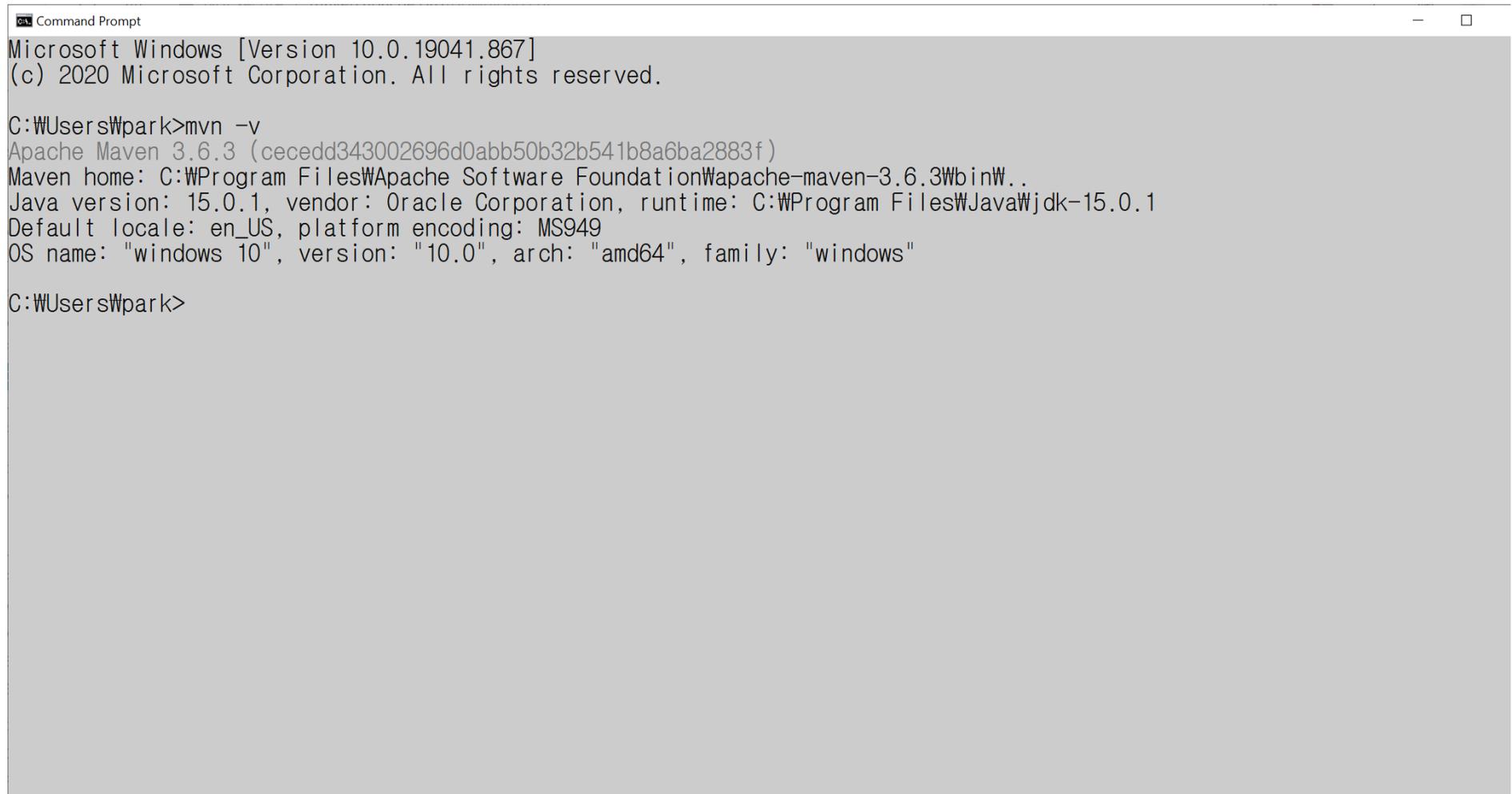
The right dialog, titled "Edit environment variable", shows a list of environment variables. The variable "%MAVEN\_HOME%\bin" is highlighted in blue. The list includes:

- C:\DevTools\Anaconda3
- C:\DevTools\Anaconda3\Library\mingw-w64\bin
- C:\DevTools\Anaconda3\Library\usr\bin
- C:\DevTools\Anaconda3\Library\bin
- C:\DevTools\Anaconda3\Scripts
- C:\Windows\system32
- C:\Windows
- C:\Windows\System32\Wbem
- C:\Windows\System32\WindowsPowerShell\v1.0\
- C:\Windows\System32\OpenSSH\
- C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
- %JAVA\_HOME%\bin
- C:\Users\park\AppData\Local\Android\Sdk\platform-tools
- C:\Program Files\Git\cmd
- C:\Program Files\dotnet\
- C:\Program Files (x86)\dotnet\
- %MAVEN\_HOME%\bin

# Maven

---

## □ Run "mvn -v"



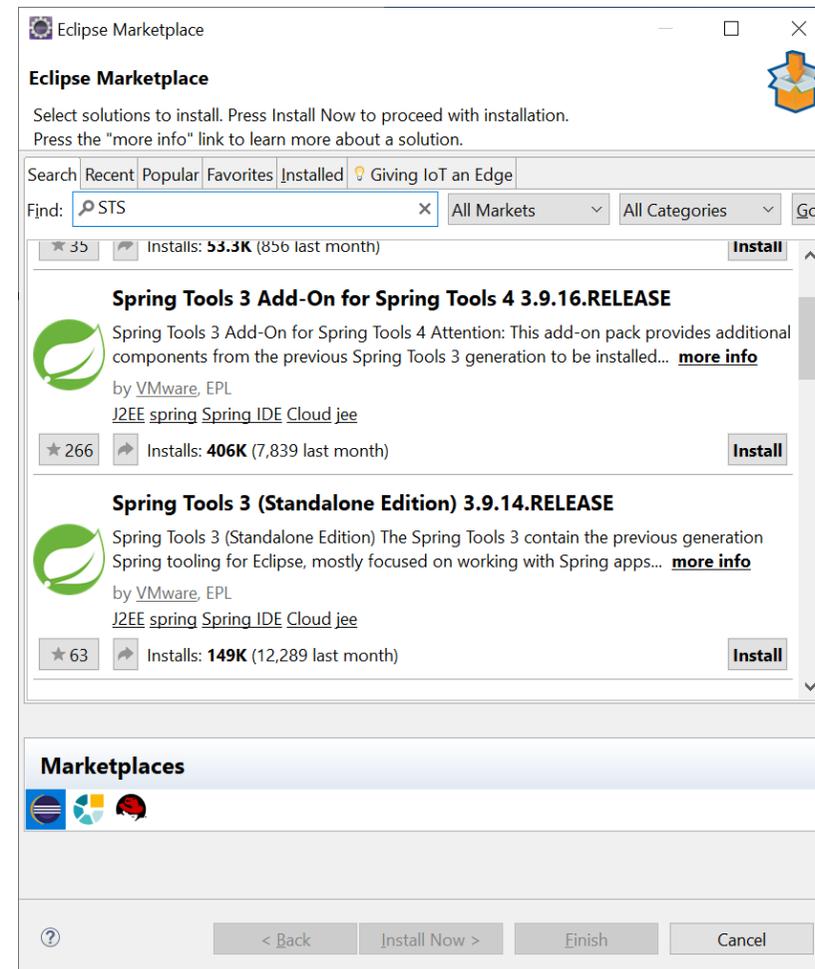
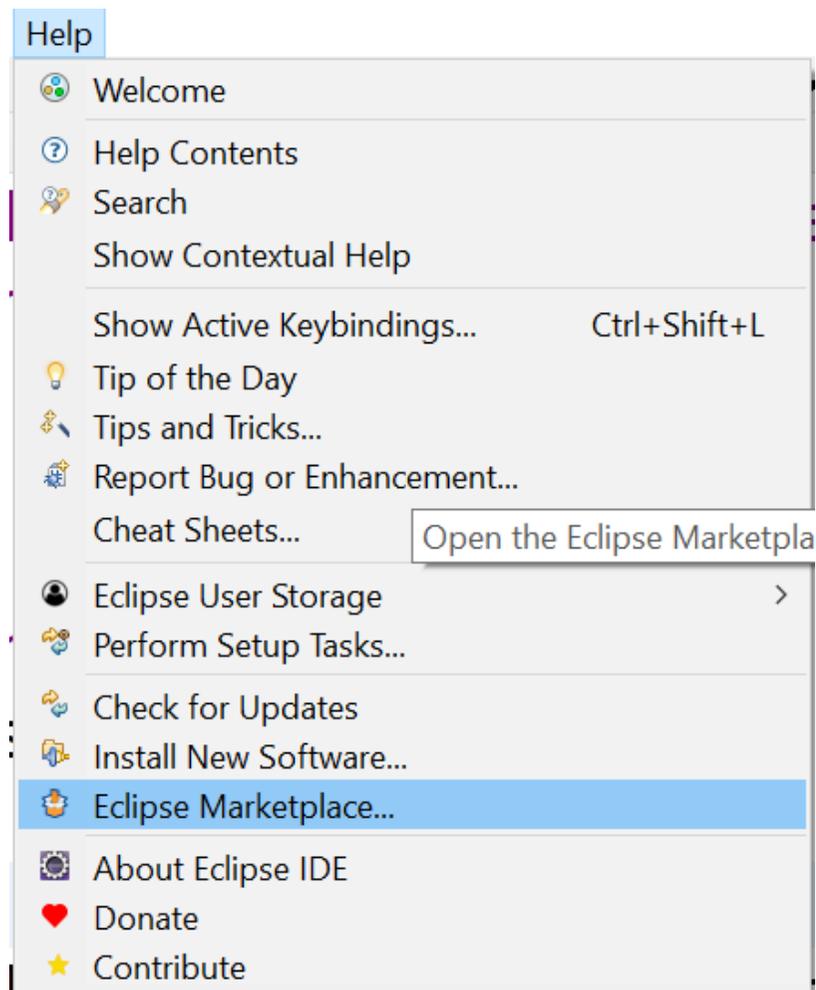
```
Command Prompt
Microsoft Windows [Version 10.0.19041.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Wpark>mvn -v
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: C:\Program Files\Apache Software Foundation\apache-maven-3.6.3\bin\..
Java version: 15.0.1, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-15.0.1
Default locale: en_US, platform encoding: MS949
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

C:\Users\Wpark>
```

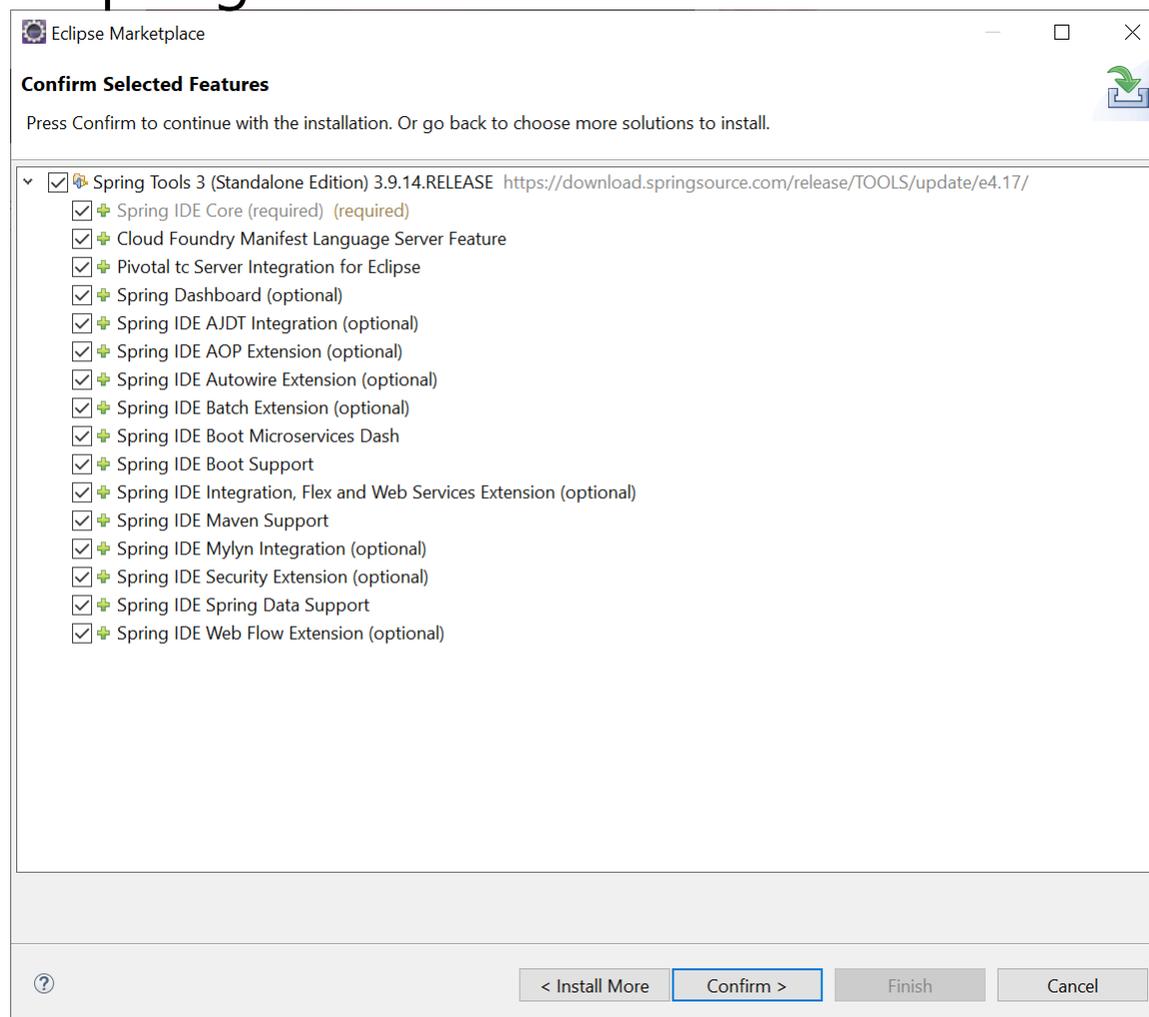
# Spring Tool Suite (STS)

- ❑ Eclipse -> Help -> Eclipse Marketplace -> Find: STS -> Spring Tools 3.9.14.RELEASE->Install



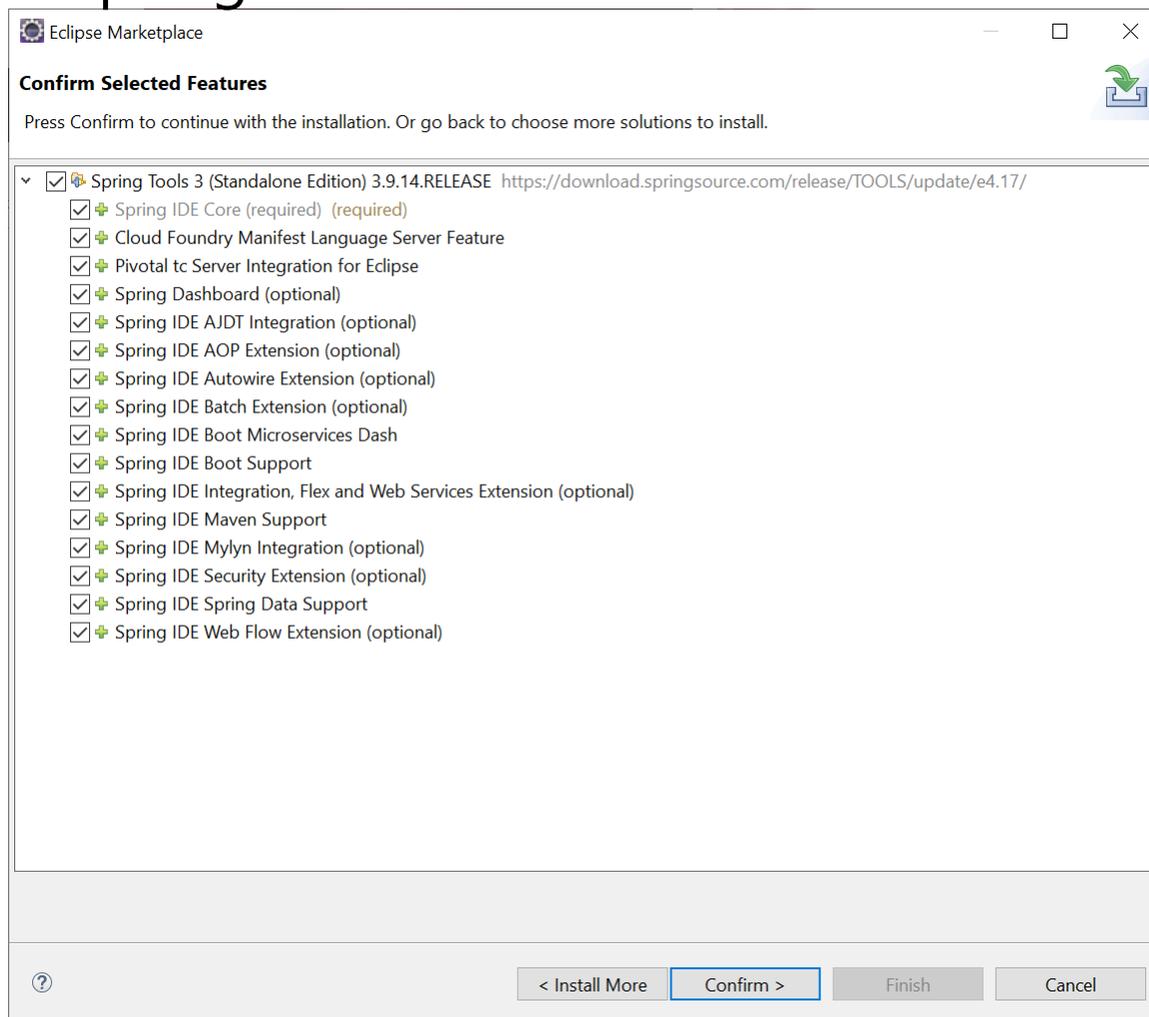
# Spring Tool Suite (STS)

- ❑ Eclipse -> Help -> Eclipse Marketplace -> Find: STS -> Spring Tools 3.9.14.RELEASE->Install & Restart Eclipse



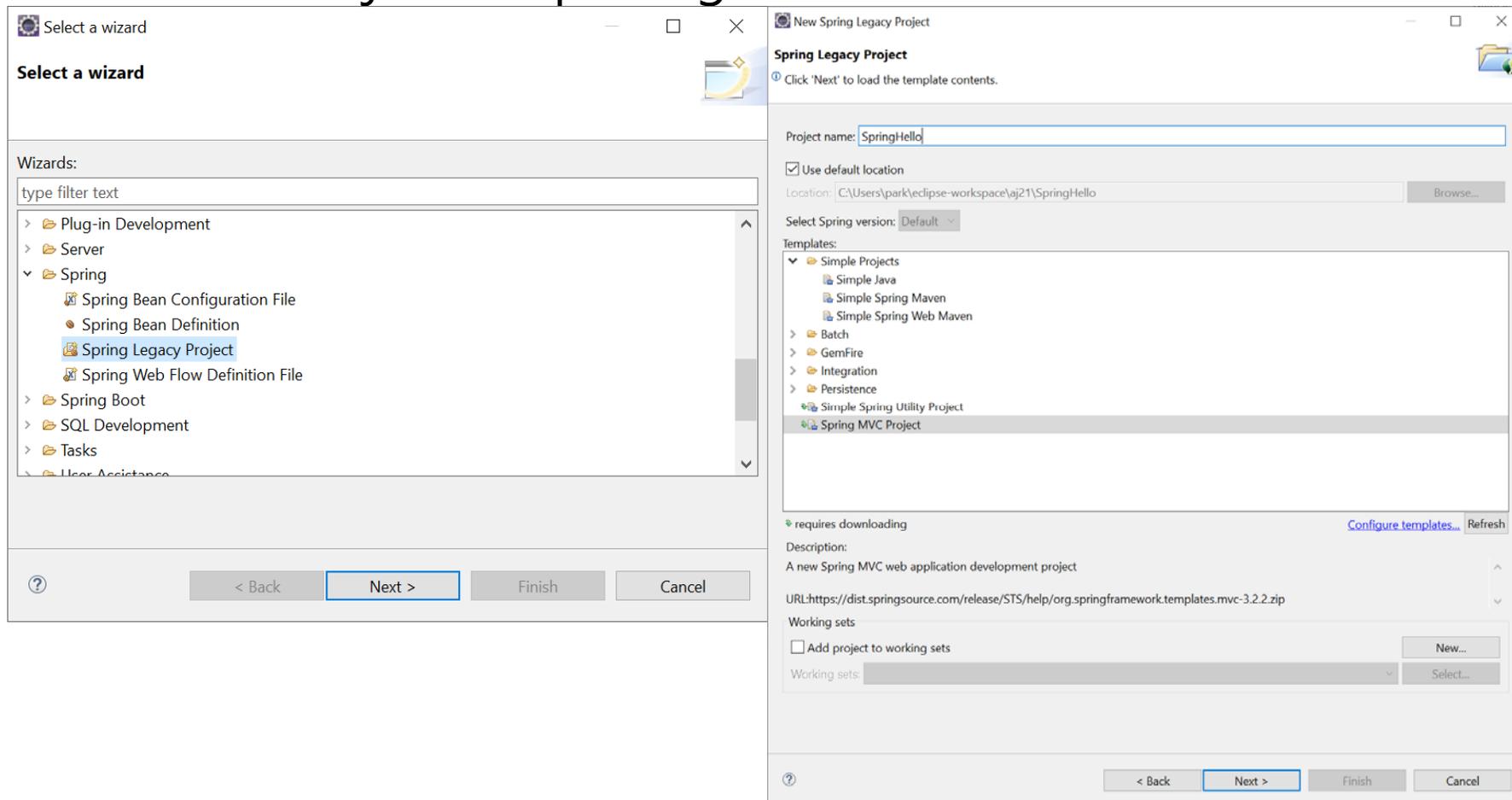
# Spring Tool Suite (STS)

- ❑ Eclipse -> Help -> Eclipse Marketplace -> Find: STS -> Spring Tools 3.9.14.RELEASE->Install & Restart Eclipse



# Eclipse Spring Legacy Project

- ❑ Eclipse -> File -> New -> Other -> Spring -> Spring Legacy Project -> Project name & Templates: Spring MVC Project -> package name -> Finish



# Spring MVC Project

## Spring MVC Project 디렉토리 구조

The image shows a screenshot of a Spring MVC project directory structure on the left and a browser window on the right. The browser window displays "Hello world!" and "The time on the server is 2021? 4? 4? ?? 7? 50? 4? KST." The directory structure is as follows:

- SpringHello
  - Deployment Descriptor: SpringHello
  - Spring Elements
  - JAX-WS Web Services
  - Java Resources
  - Deployed Resources
  - src
    - main
      - java
        - kr
          - ac
            - dankook
              - HomeController.java
    - resources
    - webapp
      - resources
    - WEB-INF
      - classes
      - spring
        - appServlet
          - servlet-context.xml
          - root-context.xml
      - views
        - home.jsp
        - web.xml
    - test
    - target
    - pom.xml

자바 코드 디렉토리

리소스 디렉토리

View JS, CSS

웹을 위한 Spring 셋팅

Spring MVC configuration

View JSP

톰캣 configuration file

메이븐 configuration file

# pom.xml 수정

---

pom.xml

```
<properties>
<java-version>15</java-version>
<org.springframework-version>5.2.8.RELEASE</org.springframework-version>
<org.aspectj-version>1.6.10</org.aspectj-version>
<org.slf4j-version>1.6.6</org.slf4j-version>
</properties>
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.8.1</version>
      <configuration>
        <source>15</source>
        <target>15</target>
        <compilerArgument>-Xlint:all</compilerArgument>
        <showWarnings>true</showWarnings>
        <showDeprecation>true</showDeprecation>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Eclipse Project -> Maven -> Update Project (Update Maven)

# home.jsp 수정

---

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

home.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```

```
<%@ page session="false" %>
```

```
<html>
```

```
<head>
```

```
<title>Home</title>
```

```
</head>
```

```
<body>
```

```
<h1>
```

```
Hello world!
```

```
</h1>
```

```
<P> The time on the server is ${serverTime}. </P>
```

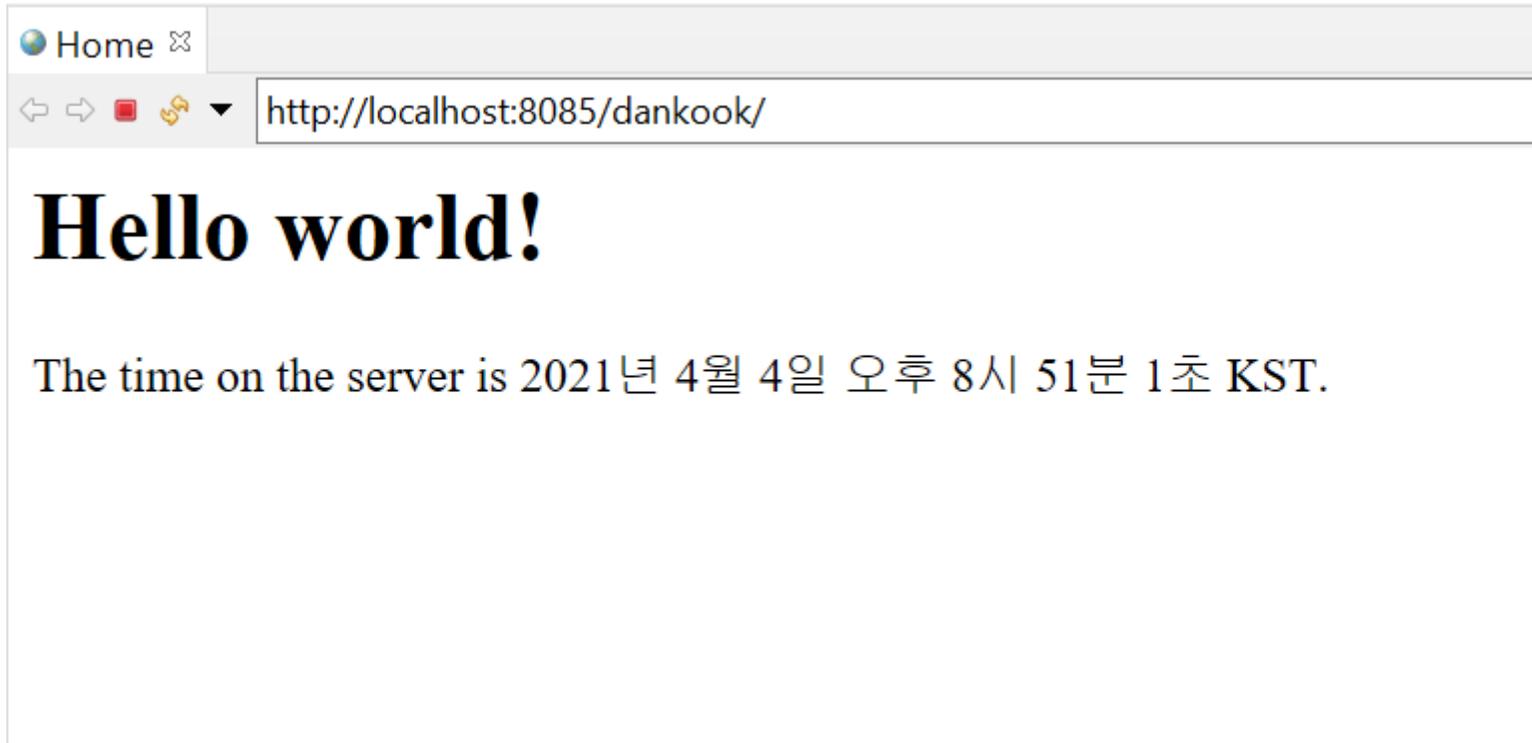
```
</body>
```

```
</html>
```

Eclipse Project -> Run As -> Maven Clean & Maven Install & Run on Server

# Spring MVC Project 실행

---



# HomeController.java 수정

HomeController.java

```
@Controller
public class HomeController {
    private static final Logger logger = LoggerFactory.getLogger(HomeController.class);

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public String home(Locale locale, Model model) {
        logger.info("Welcome home! The client locale is {}.", locale);
        Date date = new Date();
        DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG,
            DateFormat.LONG, locale);
        String formattedDate = dateFormat.format(date);
        model.addAttribute("serverTime", formattedDate );
        // add more attribute
        String greetings = "Greetings, Spring MVC";
        model.addAttribute("message", greetings);
        return "home";
    }
}
```

serverTime과 message 속성이 모델에 추가됨

# home.jsp 수정

---

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

home.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```

```
<%@ page session="false" %>
```

```
<html>
```

```
<head>
```

```
<title>Home</title>
```

```
</head>
```

```
<body>
```

```
<h1>
```

```
Hello world! ${message}
```

```
</h1>
```

```
<P> The time on the server is ${serverTime}. </P>
```

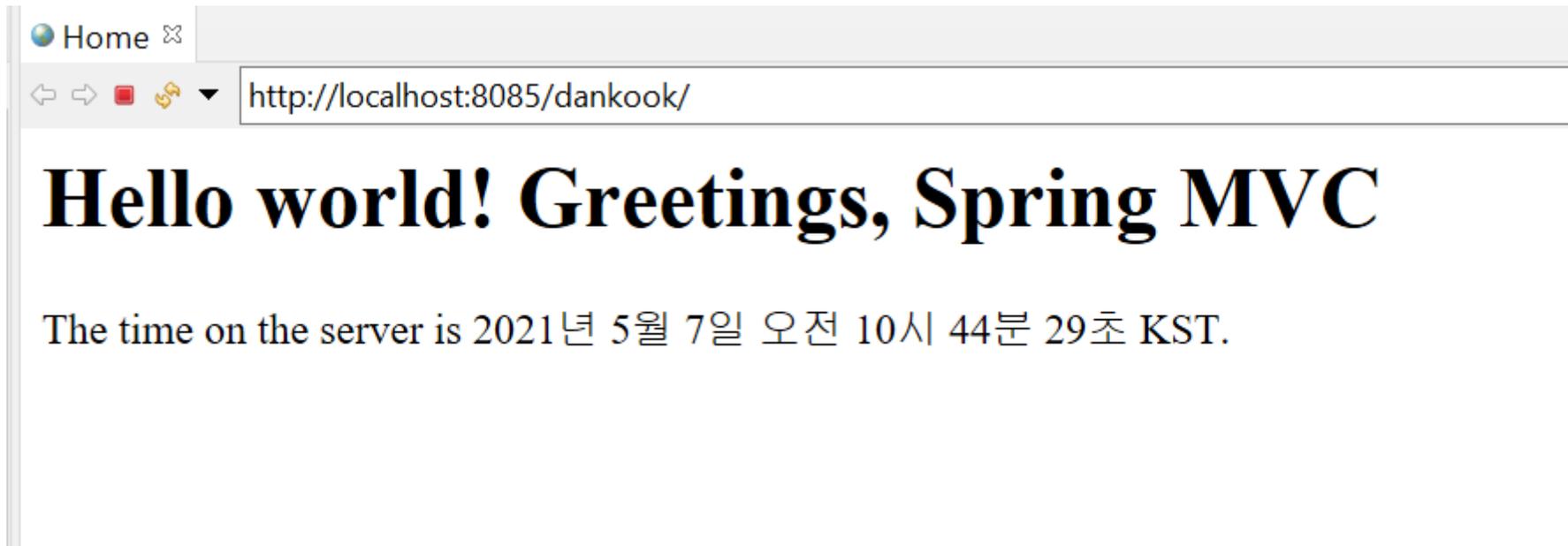
```
</body>
```

```
</html>
```

Eclipse Project -> Run As -> Maven Clean & Maven Install & Run on Server

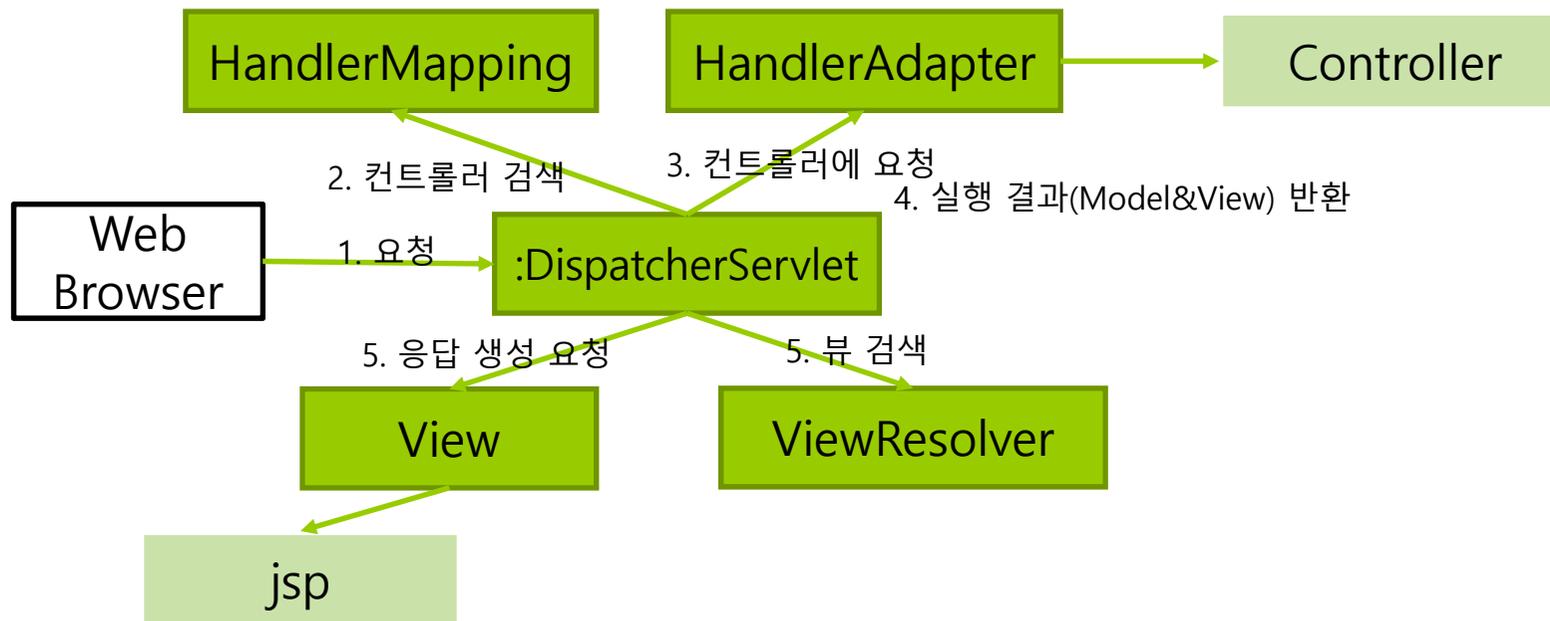
# Spring MVC Project 실행

---



# 기본 흐름

- MVC 모델 특성상 누가 누구를 호출하는지 직접적으로 표현되지 않음
- 전체 구조를 이해할 필요가 있음



# 스프링 MVC 설정 과정

---

- web.xml에 DispatcherServlet 설정
  - 어떤 경로 접근에 어떤 기능(컨트롤러) 할당할 것인지
- web.xml 에 캐릭터 인코딩 필터 설정
- 스프링 MVC 설정
  - HandlerMapping, HandlerAdapter 설정
  - ViewResolver 설정

↓

.jsp 뷰를 생성하는데 사용

가장 심플하게는  
<mvc:annotation-driven>태그를 xml  
설정파일에 추가

java 설정 사용시  
@EnableWebMvc

@Controller 태그  
활성화

# DispatcherServlet 설정

---

- 웹브라우저의 요청(<http://localhost:8080/hello>)  
받아들여서 스프링 컨테이너를 생성
  - 컨테이너 설정 파일로는 /WEB-INF/서블릿이름-servlet.xml이 기본 설정됨
  - 설정 파일 목록을 만들기 위해서는 contextConfigLocation 매개변수를 web.xml에 지정

```
<!-- Processes application requests -->
<servlet>
  <servlet-name>appServlet</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```

web.xml

# DispatcherServlet 설정

## □ servlet-context.xml 확인

servlet-context.xml

```
<!-- DispatcherServlet Context: defines this servlet's request-processing
infrastructure -->
<!-- Enables the Spring MVC @Controller programming model -->
<annotation-driven />
<!-- Handles HTTP GET requests for /resources/** by efficiently serving up static
resources in the ${webappRoot}/resources directory -->
<resources mapping="/resources/**" location="/resources/" />
<!-- Resolves views selected for rendering by @Controllers to .jsp resources in the
/WEB-INF/views directory -->
<beans:bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
<beans:property name="prefix" value="/WEB-INF/views/" />
<beans:property name="suffix" value=".jsp" />
</beans:bean>

<context:component-scan base-package="kr.ac.dankook" />
```

# 캐릭터 인코딩 필터 설정

---

- 한글 문자열의 전송 패러미터(localhost/사용자/) 처리를 위한 인자
- web.xml의 webapp 태그 내에 다음 내용 추가

```
<filter>
  <filter-name>encodingFilter</filter-name>
  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
  <init-param>
    <param-name>encoding</param-name>
    <param-value>UTF-8</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>encodingFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

# MVC 구성요소 설정 (servlet-context.xml)

- HandlerMapping, HandlerAdapter, ViewResolver 설정
- 내장 매퍼와 어댑터 사용은 다음으로 해결
  - @Controller 어노테이티드 클래스를 컨트롤로 사용할 수 있도록 함

```
<!-- Enables the Spring MVC @Controller programming model -->  
<annotation-driven />
```

servlet-context.xml

- 뷰 리졸버는 아래를 통해 구현
  - /web-inf/views/ 아래의 .jsp 파일들을 뷰로 사용

```
<!-- Resolves views selected for rendering by @Controllers to .jsp resources  
in the /WEB-INF/views directory -->  
<beans:bean  
class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
<beans:property name="prefix" value="/WEB-INF/views/" />  
<beans:property name="suffix" value=".jsp" />  
</beans:bean>
```

# 컨트롤러 경로 매핑(라우팅) (web.xml)

---

- 사용자가 요청하는 모든 경로에 대해 appServlet 사용
  - \*.do 와 같이 확장자로 기능을 할당하는 경우
  - /users/\* 와 같이 경로의 이름으로 기능을 할당하는 경우
    - 다음 코드는 전체 경로에 대해 **appServlet**을 할당하고 있음
    - jsp 요청을 제외한 나머지 모든 요청을 appServlet이 받아들이게 됨
    - 스프링 mvc 설정파일에 <mvc:default-servlet-handler />를 추가하여 톰캣의 404오류 페이지도 스프링이 핸들링하도록 할 수 있음

```
<servlet-mapping>  
  <servlet-name>appServlet</servlet-name>  
  <url-pattern>/</url-pattern>  
</servlet-mapping>
```

# 컨트롤러 구현

@Controller 어노테이션

HomeController.java

```
public class HomeController {  
    private static final Logger logger = LoggerFactory.getLogger(HomeController.class);
```

@RequestMapping(value = "/", method = RequestMethod.GET) 경로매핑

```
public String home(Locale locale, Model model) {  
    logger.info("Welcome home! The client locale is {}.", locale);  
    Date date = new Date();  
    DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG,  
        DateFormat.LONG, locale);  
    String formattedDate = dateFormat.format(date);
```

```
    model.addAttribute("serverTime", formattedDate );  
    // add more attribute  
    String greetings = "Greetings, Spring MVC";  
    model.addAttribute("message", greetings);
```

모델에 데이터 추가  
serverTime과 message  
속성이 모델에 추가됨

```
    return "home"; 뷰 이름 리턴
```

```
}
```

```
}
```