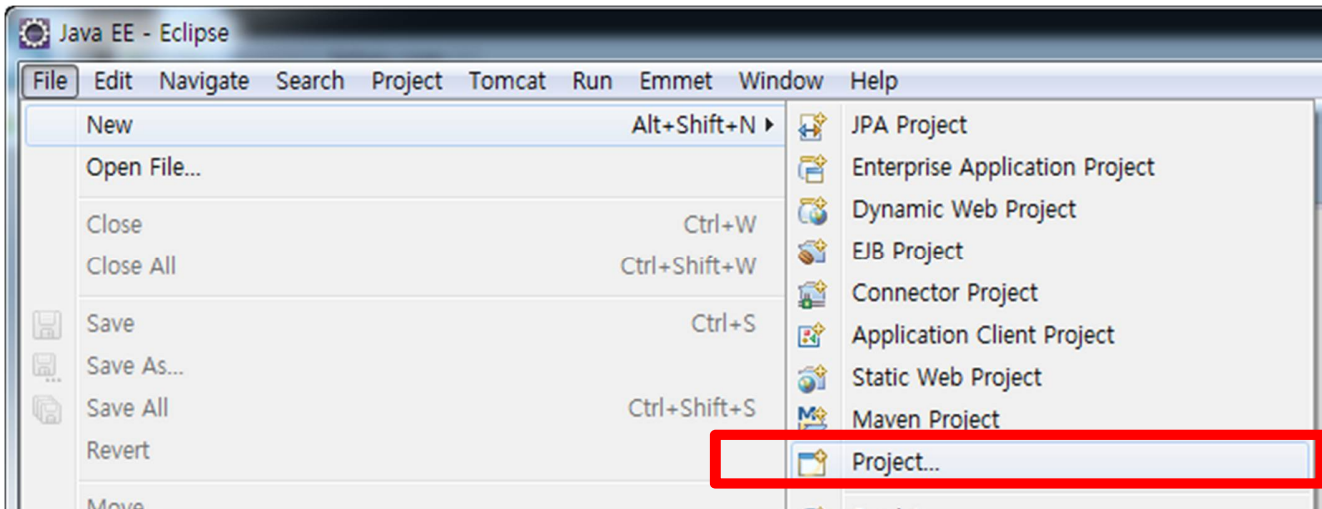


목차

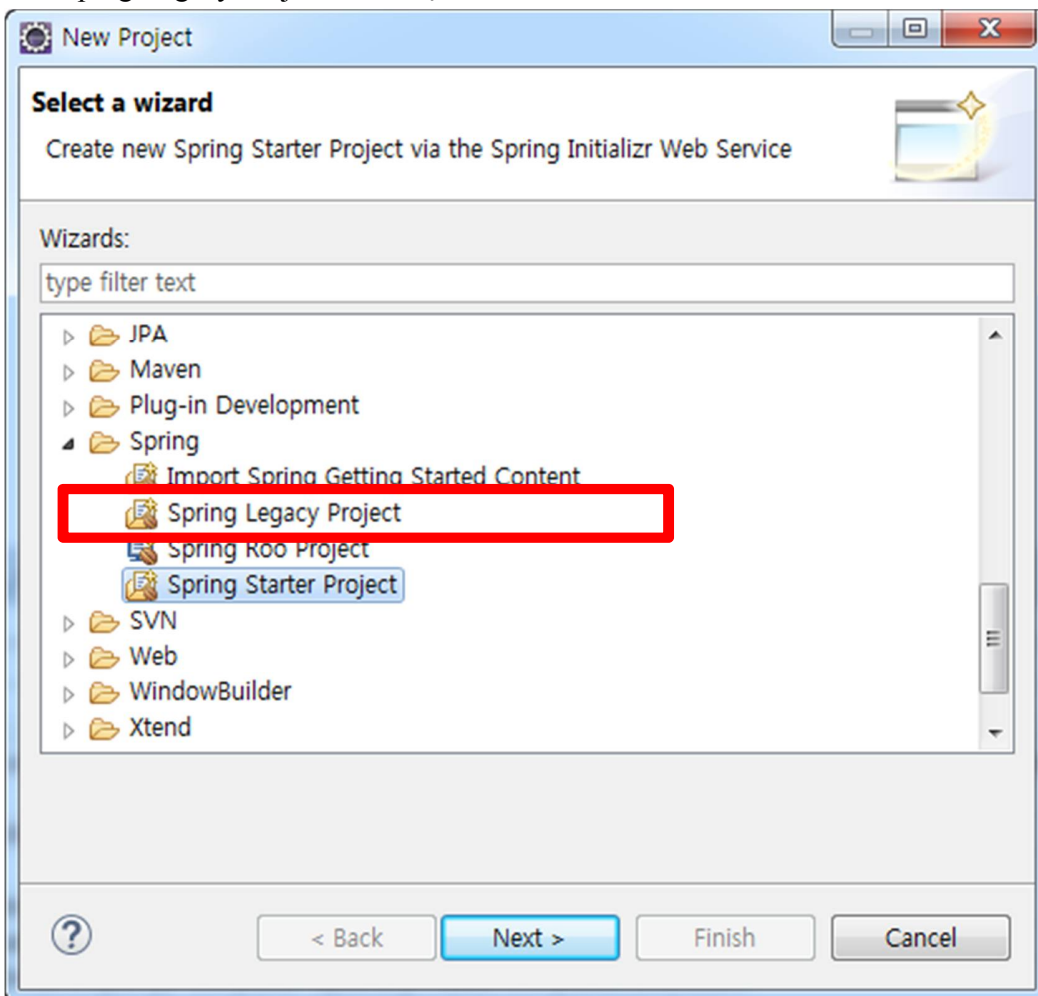
1. 스프링 프로젝트 생성 & 실행	2
2. 스프링 프로젝트의 디렉터리 구조	5
3. Tomcat 서버 설정	6
4. 스프링 설정	7
4.1 web.xml	7
4.2 root-context.xml	8
4.3 servlet-context.xml	9
5. Converting a Maven Project to Gradle project	10
6. build.gradle 생성	11
7. Gradle >> Refresh All	19
8. 프로젝트 실행	20
9. Context root 수정	21
10. 크롬을 기본 브라우저로 설정	22
11. Reference	23

1. 스프링 프로젝트 생성 & 실행

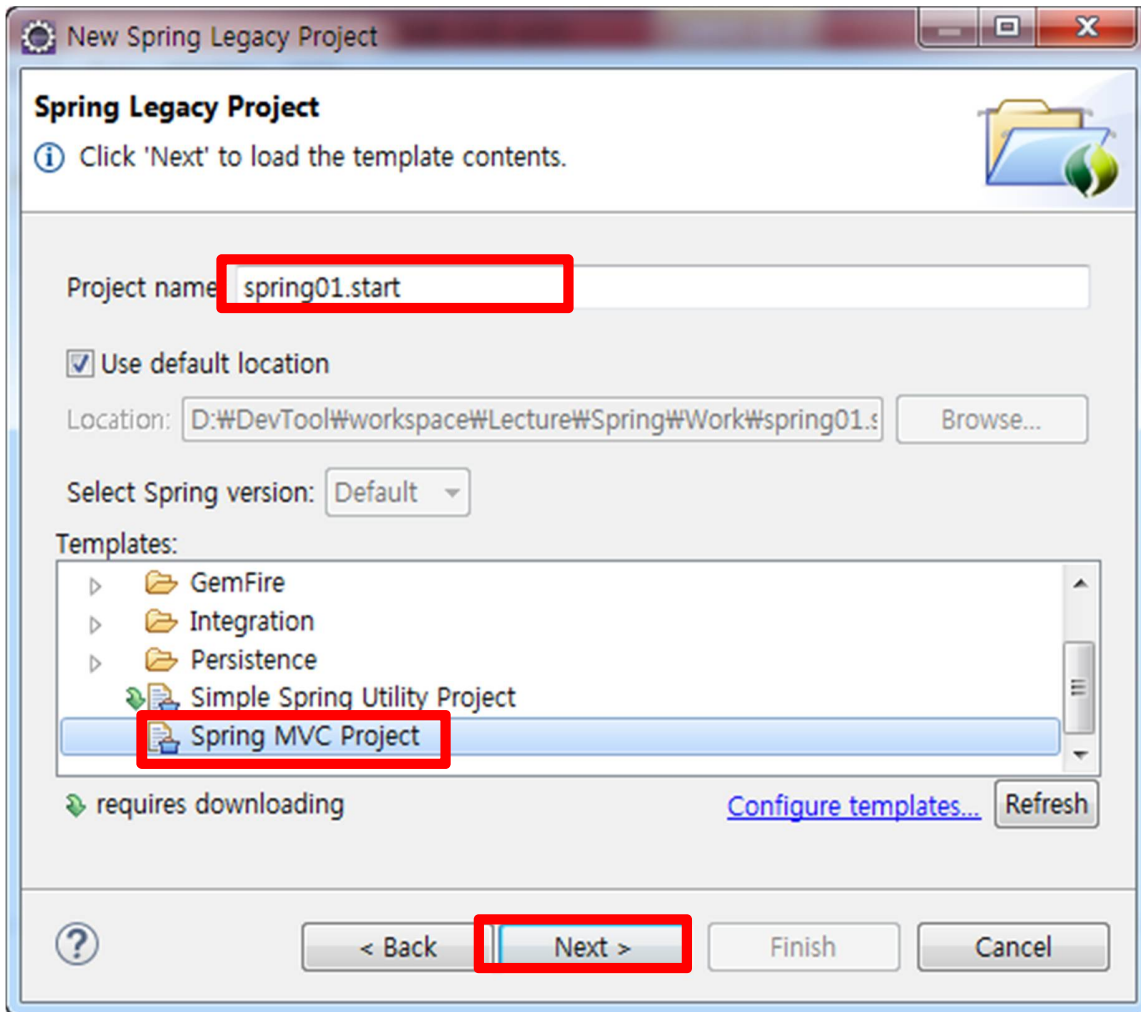
- File > New > Other 를 선택 후 아래와 같이 spring project 를 하나 생성해보자.



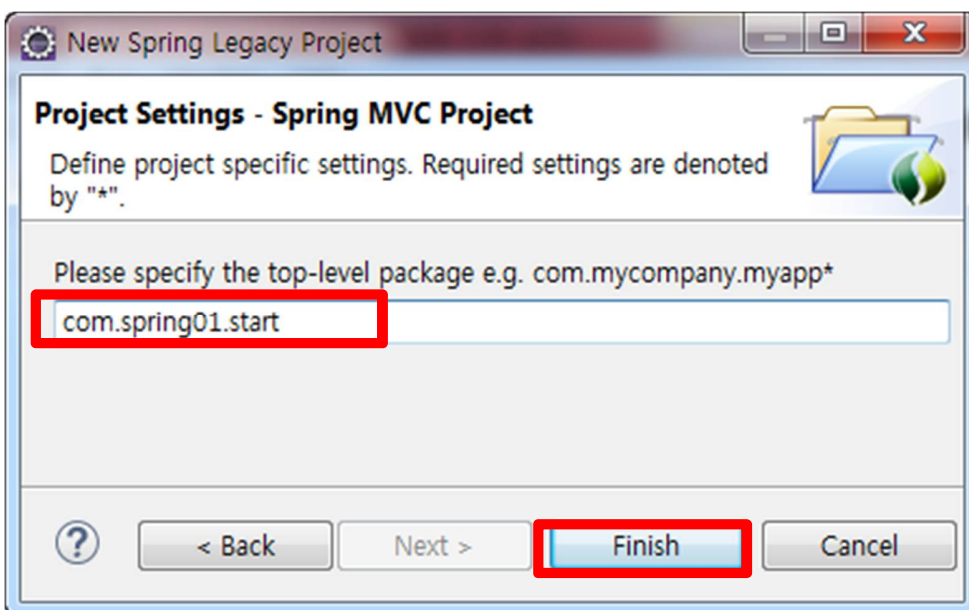
- Sping Legacy Project 를 선택



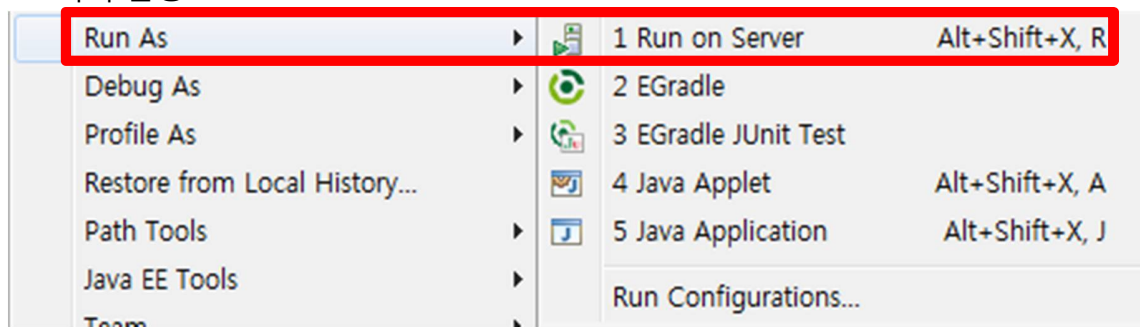
- Spring Project 를 생성 시 아래를 보면..Spring MVC Project 가 있다.



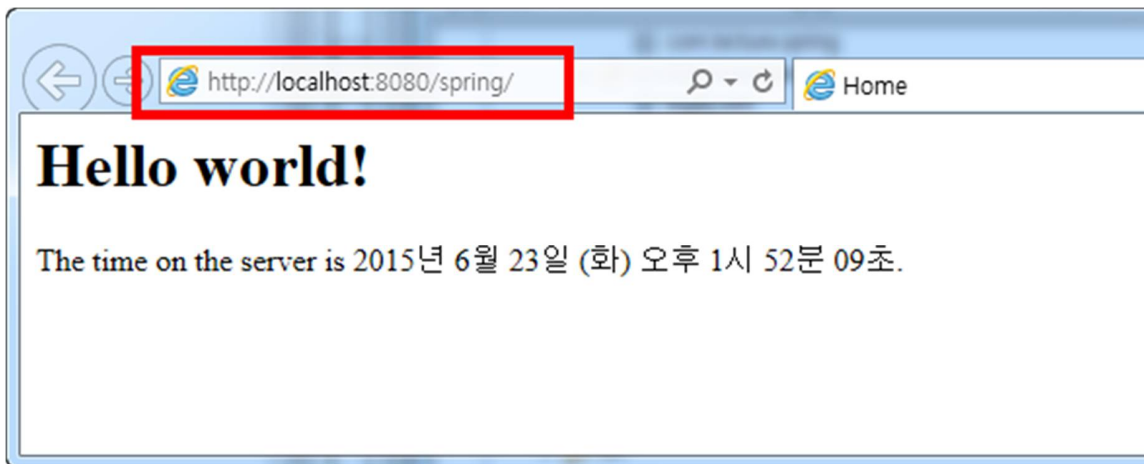
- 패키지 설정



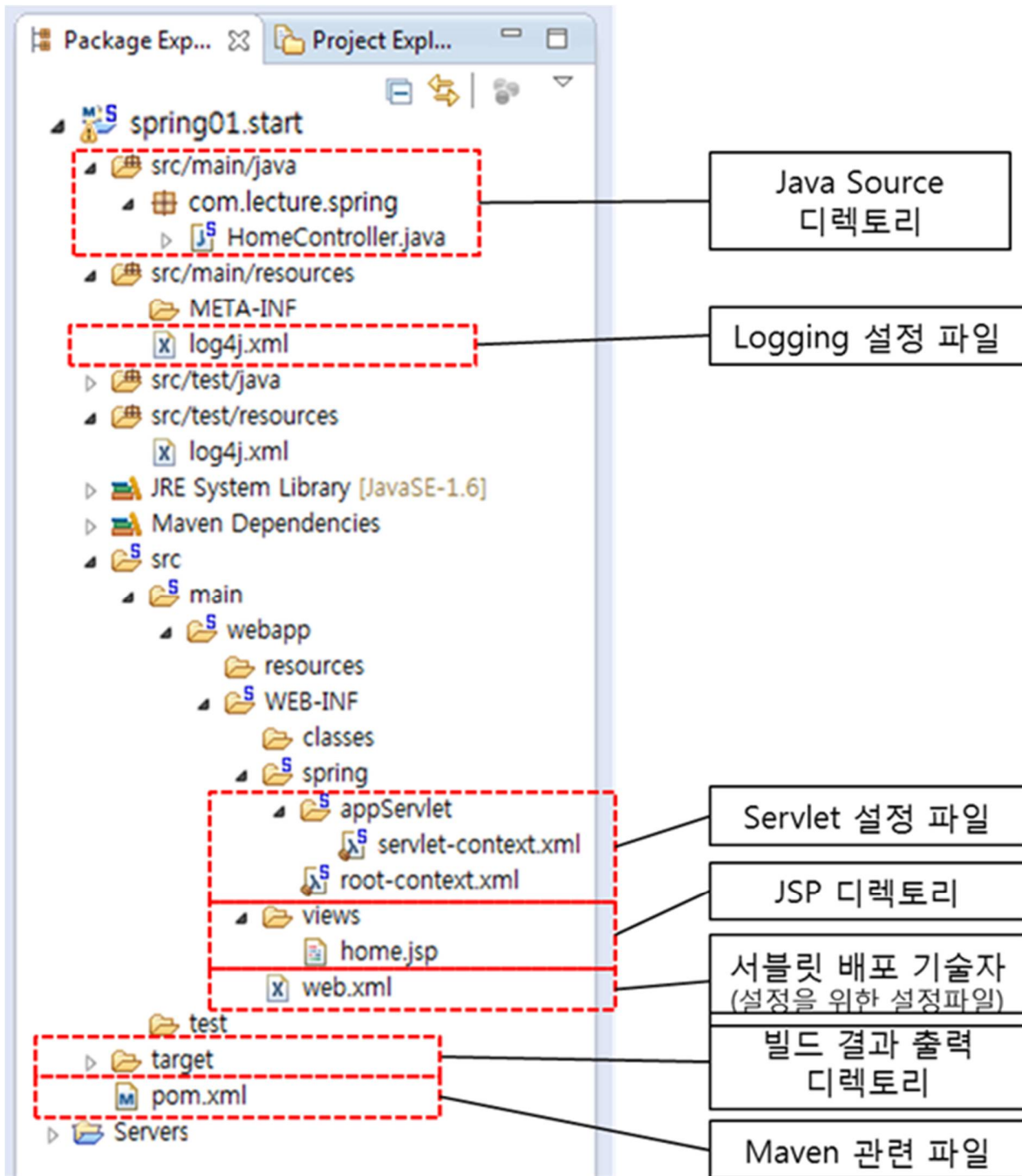
- 서버 실행



- 실행 화면



2. 스프링 프로젝트의 디렉터리 구조



3. Tomcat 서버 설정

- {CATALINA_HOME}/conf/server.xml 수정

```
<Connector port="80" protocol="HTTP/1.1"  
    connectionTimeout="20000"  
    redirectPort="8443"  
    URIEncoding="utf-8" />
```

4. 스프링 설정

4.1 web.xml

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

  <!-- Spring Web 어플리케이션을 위한 메인 설정파일을 등록한다. -->
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
  </context-param>

  <!-- Spring Web 어플리케이션 컨테스트를 로딩한다. -->
  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>

  <!-- Spring Web 어플리케이션의 맨 앞단 Controller(DispatcherServlet) 를 등록한다. -->
  <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>
  <servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

  <!-- 문자 인코딩 처리 필터 설정 -->
  <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>
    <init-param>
      <param-name>forceEncoding</param-name>
      <param-value>true</param-value>
    </init-param>
```

```

</filter>
<filter-mapping>
  <filter-name>encodingFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>

```

```

<!-- -->
<welcome-file-list>
  <welcome-file>index.html</welcome-file>
</welcome-file-list>

```

```

<!-- jsp 파일 utf-8 페이지 인코딩 설정 <%@ page pageEncoding="UTF-8" %> -->
<jsp-config>
  <jsp-property-group>
    <url-pattern>*.jsp</url-pattern>
    <page-encoding>UTF-8</page-encoding>
  </jsp-property-group>
</jsp-config>

```

```

<!-- 에러 페이지 설정 -->
<error-page>
  <error-code>403</error-code>
  <location>/WEB-INF/views/error.jsp</location>
</error-page>

```

```

<error-page>
  <error-code>404</error-code>
  <location>/WEB-INF/views/error.jsp</location>
</error-page>

```

```

<error-page>
  <error-code>500</error-code>
  <location>/WEB-INF/views/error.jsp</location>
</error-page>

```

```

</web-app>

```

4.2 root-context.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

```



```
<!-- Root Context: defines shared resources visible to all other web components -->
```

```
</beans>
```

4.3 servlet-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:beans="http://www.springframework.org/schema/beans"
  xmlns:context="http://www.springframework.org/schema/context"

  xsi:schemaLocation="http://www.springframework.org/schema/mvc
    http://www.springframework.org/schema/mvc/spring-mvc.xsd
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context.xsd">

  <!-- 스프링의 DispatcherServlet 에게 정적인 자원을 알려준다
  정적자원에는 css 파일, js 파일, 이미지등... -->
  <resources mapping="/resources/**" location="/resources/" />

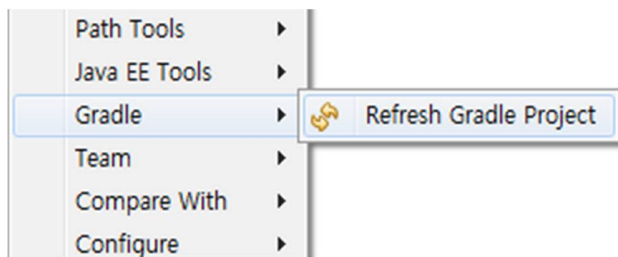
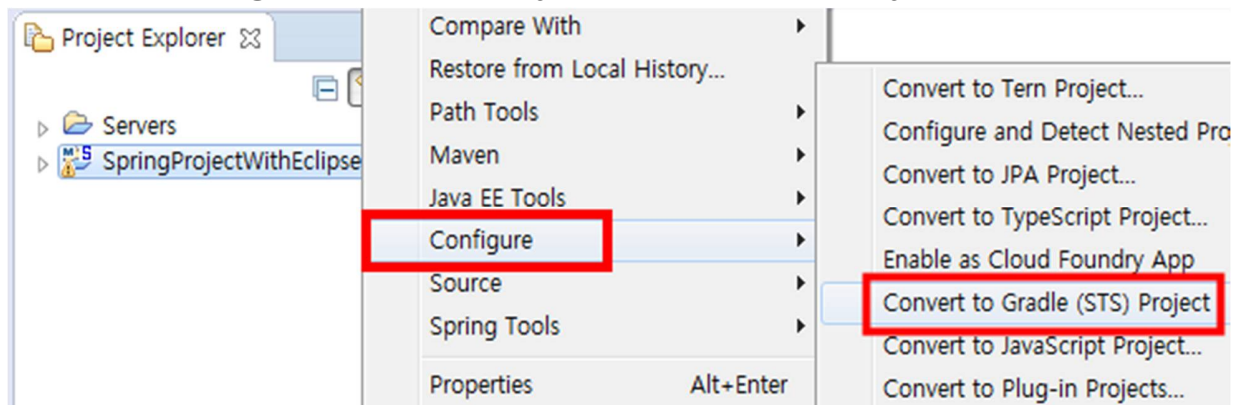
  <!-- Resolves views : jsp 파일 찾는 경로 설정 -->
  <beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <beans:property name="prefix" value="/WEB-INF/views/" />
    <beans:property name="suffix" value=".jsp" />
  </beans:bean>

  <!-- step1. 스프링의 어노테이션을 사용할 수 있도록 하는 설정 -->
  <!-- @RequestMapping , @ExceptionHandler 등과 같은 어노테이션을 사용하는 경우 설정해야 함 -->
  <annotation-driven />
  <context:spring-configured />
  <context:annotation-config />

  <!-- step2. 어노테이션이 지정된 클래스를 컨테이너에 자동으로 등록되게 하는 설정 -->
  <!-- 복수의 패키지를 사용하고 싶은 경우 <context:component-scan> 태그를 여러개 작성 -->
  <context:component-scan base-package="com.lecture.spring" />

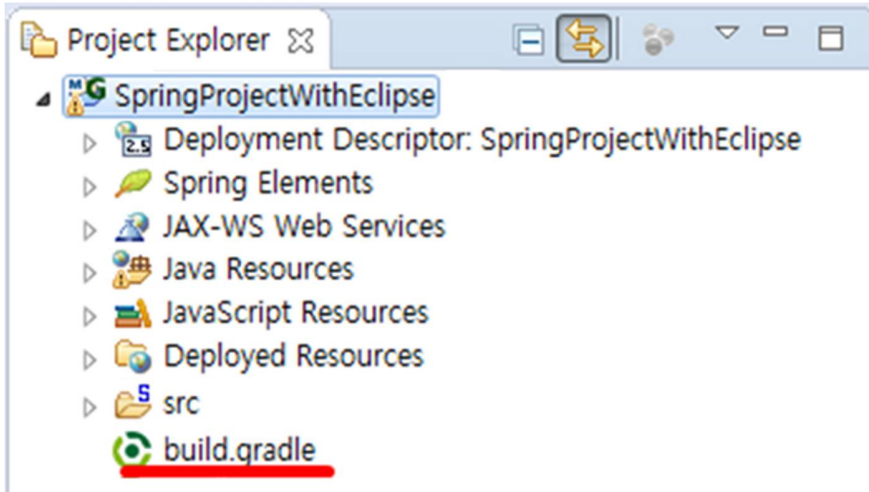
</beans:beans>
```

5. Converting a Maven Project to Gradle project



6. build.gradle 생성

프로젝트에 build.gradle 을 생성하고 아래의 내용을 복사하여 붙여 넣는다.



```

/*
 * Reference Site
 *
 * http://netframework.tistory.com/entry/gradle%EC%9D%84-%EC%9D%B4%EC%9A%A9%ED%95%9C-Spring-MVC-
Web-Application-Spring-Data-JPA-QueryDSL
 * http://huskdoll.tistory.com/8
 * http://hangaebal.blogspot.kr/2014/06/spring-eclipse-gradle.html
 *
 * https://github.com/bmuschko/gradle-tomcat-plugin
 * https://github.com/bmuschko/gradle-cargo-plugin
 *
 * http://stackoverflow.com/questions/4384809/cause-no-such-property-sourcesets-for-class-org-
gradle-api-plugins-convention
 */

// tomcat 과 cargo plugin 에 대한 repository 설정입니다.
buildscript {
    repositories {
        jcenter()
    }

    dependencies {
        classpath 'org.gradle.api.plugins:gradle-tomcat-plugin:1.0'
        classpath 'org.gradle.api.plugins:gradle-cargo-plugin:1.4'
    }
}

// Apply the java plugin to add support for Java
apply plugin: 'java'
apply plugin: 'war'

```

```

apply plugin: 'eclipse'
apply plugin: 'eclipse-wtp' // WTP(Web Tools Platform) -> 웹 프로젝트로 인식하도록
apply plugin: 'groovy'
apply plugin: 'tomcat'
apply plugin: 'cargo'

group = 'com.lecture.spring'
version = '1.0.0'
description = 'controller example'

// JAVA Version 1.8
compileJava {
    sourceCompatibility = 1.8
    targetCompatibility = 1.8
}

compileTestJava {
    sourceCompatibility = 1.8
    targetCompatibility = 1.8
}

// 소스 인코딩 UTF-8 로 지정
[compileJava, compileTestJava]*.options*.encoding = 'UTF-8'

jar {
    manifest {
        attributes 'Implementation-Title': 'Gradle Quickstart', 'Implementation-Version': version
    }

    //exclude 'log4j.properties'
}

// 메이븐 Central 저장소 사용
repositories {
    // Use 'jcenter' for resolving your dependencies.
    // You can declare any Maven/Ivy/file repository here.
    maven { url 'http://repo.spring.io/libs-release' }
    maven { url 'http://repo.spring.io/milestone' }
    maven { url 'https://oss.sonatype.org/content/repositories/releases' }
    mavenCentral()
}

// dependency 버전 정보
ext {
    versions = [
        spring: '4.1.7.RELEASE'
        , security: '3.2.8.RELEASE'
        , junit: '4.11'
        , servletApi: '3.1.0'
    ]
}

```

```

    , jstl: '1.2'
    , slf4j: '1.7.9'
    , mockito: '1.9.0'
    , cglib: '2.2.2'
    , groovy: '2.2.1'
    , jackson: '2.3.1'
    , aspectj: '1.8.6'
    , springSpock: '0.7-groovy-2.0'
    , tiles: '3.0.5'
    , hibernate: '4.2.20.Final'
    , tomcatVersion: '7.0.62'
    , cargoVersion: '1.4.5'
  ]
}

eclipse {
  wtp {
    facet {
      facet name: 'jst.web', version: '2.5' // Servlet Spec Version 지정, 미 지정시 2.4
      facet name: 'jst.java', version: '1.8' // Java Version 지정
    }
  }

  classpath {
    containers.remove('org.eclipse.jdt.launching.JRE_CONTAINER')
    containers
'org.eclipse.jdt.launching.JRE_CONTAINER/org.eclipse.jdt.internal.debug.ui.launcher.StandardVMType
/JavaSE-1.8'
  }
}

configurations {
  providedRuntime
  providedCompile
}

// In this section you declare the dependencies for your production and test code
// 의존성 설정
dependencies {

  compile fileTree(dir: 'lib', include: ['*.jar'])

  // spring 관련 라이브러리 추가
  compile "org.springframework:spring-beans:$versions.spring"
  compile "org.springframework:spring-context:$versions.spring"
  compile "org.springframework:spring-context-support:$versions.spring"
  compile "org.springframework:spring-webmvc:$versions.spring"
  compile "org.springframework:spring-orm:$versions.spring"
  compile "org.springframework:spring-core:$versions.spring"

```

st01.이클립스로 Spring Web 프로젝트 생성

```
compile "org.springframework:spring-expression:${versions.spring}"
compile "org.springframework:spring-aop:${versions.spring}"
compile "org.springframework:spring-jdbc:${versions.spring}"
compile "org.springframework:spring-tx:${versions.spring}"
compile "org.springframework:spring-websocket:${versions.spring}"
compile "org.springframework:spring-messaging:${versions.spring}"
runtime "org.springframework:spring-aspects:${versions.spring}"

compile "org.springframework.security:spring-security-core:${versions.security}"
compile "org.springframework.security:spring-security-web:${versions.security}"
compile "org.springframework.security:spring-security-config:${versions.security}"

compile "org.springframework.webflow:spring-js:2.4.1.RELEASE"
compile "org.springframework.webflow:spring-webflow:2.4.1.RELEASE"

compile "com.fasterxml.jackson.core:jackson-annotations:${versions.jackson}"
compile "com.fasterxml.jackson.core:jackson-databind:${versions.jackson}"

compile "org.apache.tiles:tiles-api:${versions.tiles}"
compile "org.apache.tiles:tiles-core:${versions.tiles}"
compile "org.apache.tiles:tiles-jsp:${versions.tiles}"

// Apache Commons Lang, a package of Java utility classes for the classes that are in
// java.lang hierarchy,
// or are considered to be so standard as to justify existence in java.lang.
compile "org.apache.commons:commons-lang3:3.4"
compile "org.apache.commons:commons-dbcp2:2.0"

compile "commons-fileupload:commons-fileupload:1.2.1"
compile "commons-io:commons-io:2.4"
compile "commons-pool:commons-pool:1.6"
compile "commons-beanutils:commons-beanutils:1.9.2"

//
compile "org.aspectj:aspectjrt:${versions.aspectj}"
compile "org.aspectj:aspectjweaver:${versions.aspectj}"
compile "org.aspectj:aspectjtools:${versions.aspectj}"

// JSR 330 JAR 를 포함하기 위한 라이브러리. @Inject, @Named 어노테이션 사용 가능
compile "javax.inject:javax.inject:1"

// JSP Standard Tag Library 사용을 위한 라이브러리.
compile "jstl:jstl:${versions.jstl}"
compile "javax.servlet.jsp.jstl:jstl-api:${versions.jstl}"
compile "taglibs:standard:1.1.2"

//
compile "cglib:cglib-nodep:${versions.cglib}"
```

```

// log library
compile "org.slf4j:slf4j-api:$versions.slf4j"
runtime "org.slf4j:slf4j-log4j12:$versions.slf4j"
runtime "org.slf4j:jcl-over-slf4j:$versions.slf4j"
runtime "log4j:log4j:1.2.17"

// log4jdbc library
compile "com.googlecode.log4jdbc:log4jdbc:1.2"

// mysql connector
compile "mysql:mysql-connector-java:5.1.34"

// mybatis library
compile "org.mybatis:mybatis-spring:1.2.2"
compile "org.mybatis:mybatis:3.2.8"

// hibernate library
compile "org.hibernate:hibernate-core:$versions.hibernate"
compile "org.hibernate:hibernate-entitymanager:$versions.hibernate"
compile "org.hibernate.javax.persistence:hibernate-jpa-2.0-api:1.0.1.Final"

// @ResponseBody 를 이용해 json 데이터를 반환하기 위한 라이브러리
compile "org.codehaus.jackson:jackson-mapper-asl:1.9.13"

//
testCompile "junit:junit:$versions.junit"
testCompile "org.springframework:spring-test:$versions.spring"
testCompile "org.mockito:mockito-core:$versions.mockito"

// tomcat plugin 설정입니다.
tomcat "org.apache.tomcat.embed:tomcat-embed-core:$versions.tomcatVersion"
tomcat "org.apache.tomcat.embed:tomcat-embed-logging-juli:$versions.tomcatVersion"
tomcat("org.apache.tomcat.embed:tomcat-embed-jasper:$versions.tomcatVersion") {
    exclude group: "org.eclipse.jdt.core.compiler", module: "ecj"
}

providedCompile "javax.servlet:javax.servlet-api:$versions.servletApi"
providedCompile "javax.servlet.jsp:javax.servlet.jsp-api:2.3.1"
providedCompile "org.apache.tomcat:tomcat-servlet-api:$versions.tomcatVersion"

// cargo 에 대한 설정입니다.
cargo "org.codehaus.cargo:cargo-core-uberjar:$versions.cargoVersion"
cargo "org.codehaus.cargo:cargo-ant:$versions.cargoVersion"
}

```

```

sourceSets {
    main {
        java.srcDirs = ['src/main/java']
        resources.srcDirs = ['src/main/resources']
    }
}

// TEST 설정
test {
    jvmArgs = ['-ea', '-Xmx256m']
    logging.captureStandardOutput(LogLevel.INFO)
    // testReport = false

    systemProperties 'property': 'value'

    testLogging {
        events 'started', 'passed'
    }
}

task copyTask(type: Copy) {

    /*
    copy {
        println 'Copy from ${libsDir} into D:\\Documents\\JAVA\\tomcat\\x64\\lib'

        from '${libsDir}'
        into 'D:\\Documents\\JAVA\\tomcat\\x64\\lib'
        include '*.jar'
    }
    */
}

war {
    baseName = "ROOT"
    version = "${new Date().format('yyyyMMdd')}"

    /*
    from 'src/rootContent' // adds a file-set to the root of the archive
    webInf { from 'src/additionalWebInf' } // adds a file-set to the WEB-INF dir.
    classpath fileTree('additionalLibs') // adds a file-set to the WEB-INF/lib dir.
    classpath configurations.moreLibs // adds a configuration to the WEB-INF/lib dir.
    webXml = file('src/web.xml') // copies a file to WEB-INF/web.xml
    */
}

task deployToTomcat(dependsOn: 'war') << {

```



```

copy {
    from war.archivePath
    into "/Users/jinsoohan/software/apache-tomcat/webapps"
}

tomcatStop() {
    stopPort = 8005
    stopKey = 'stopKey'
}

// tomcatRun 을 실행시키기 위해서 war 에 대한 dependency 를 주입합니다.
tomcatRun {
    httpPort = 8100
    httpsPort = 8093

    stopPort = 8005
    stopKey = 'stopKey'

    enableSSL = true
    URIEncoding = 'utf-8'
    contextPath = ''
    configFile = file('src/main/resources/META-INF/context.xml') // 기본값 src/main/webapp/META-
INF/context.xml

    dependsOn war
}

tomcatRunWar {
    dependsOn war
}

// cargo 를 이용한 배포를 위해서 war 에 대한 dependency 를 주입합니다.
cargoRedeployRemote {
    dependsOn war
}

cargoDeployRemote {
    dependsOn war
}

cargo {
    containerId = 'tomcat7x'
    port = 8080

    deployable {

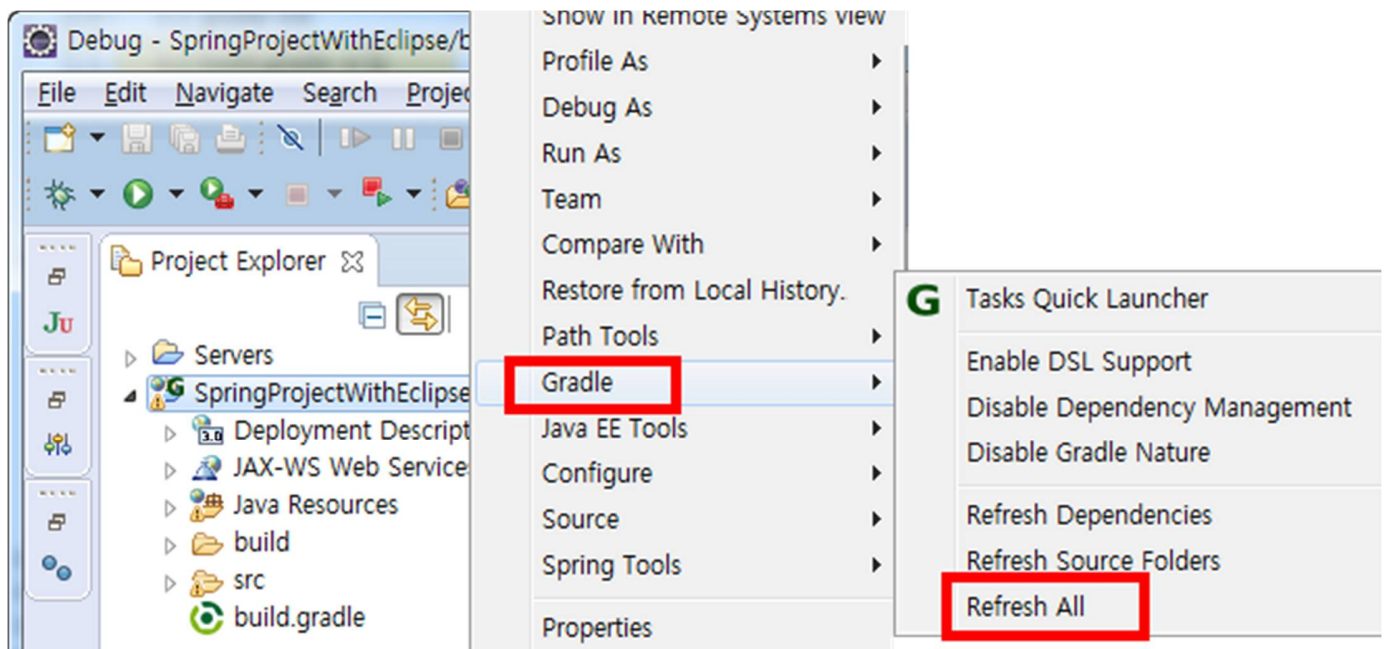
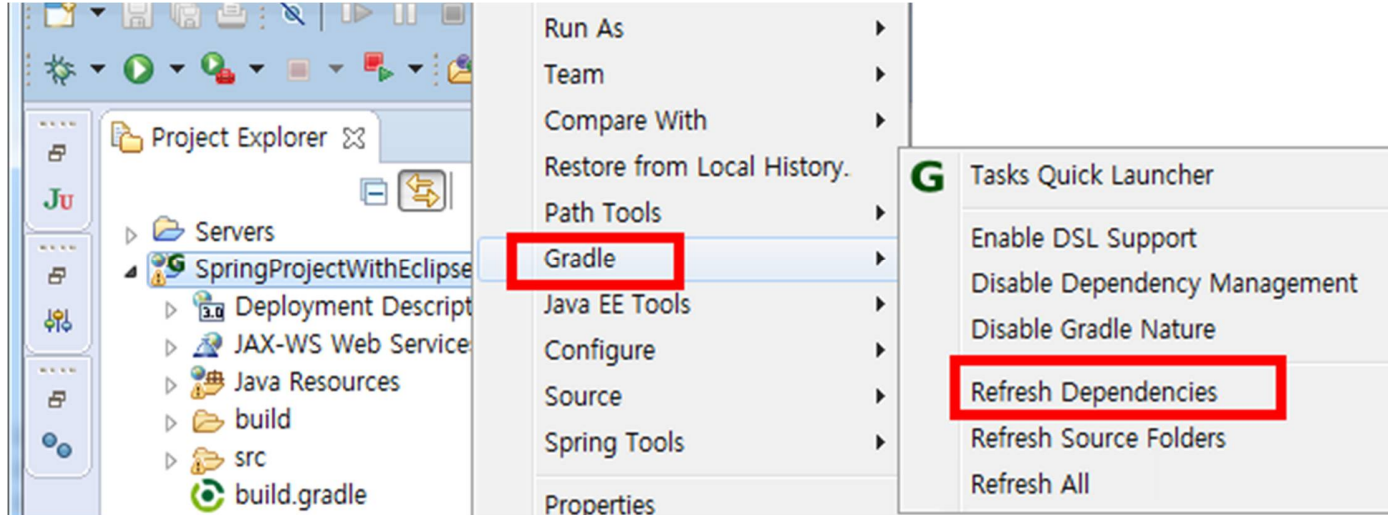
```

st01.이클립스로 Spring Web 프로젝트 생성

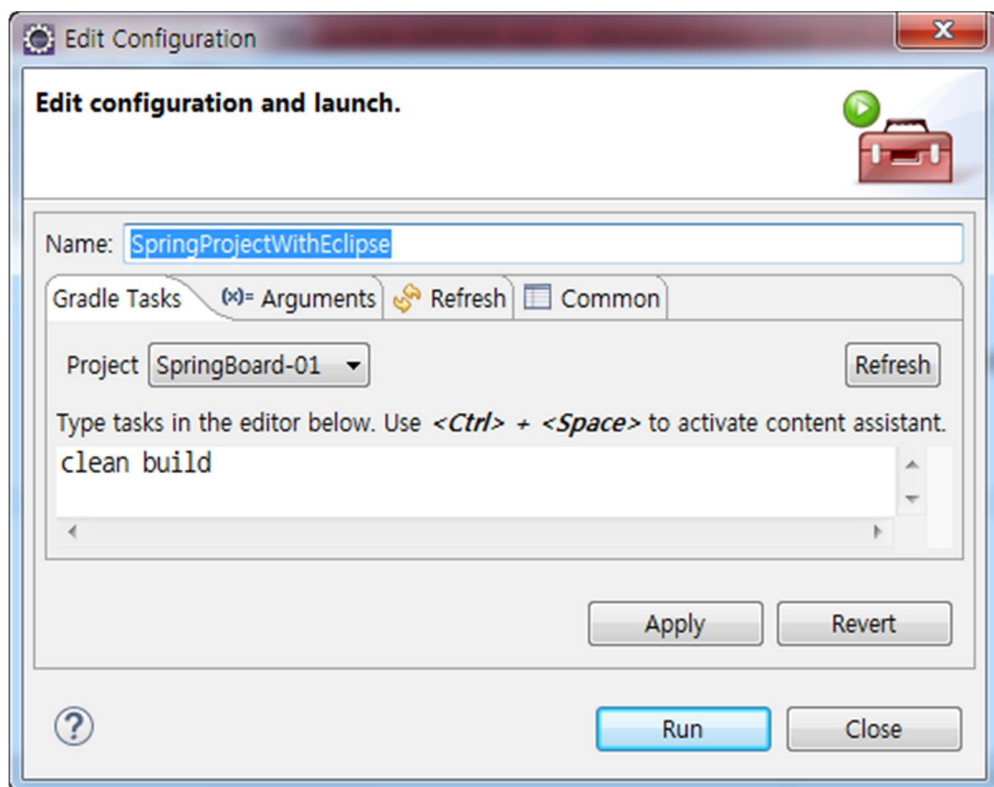
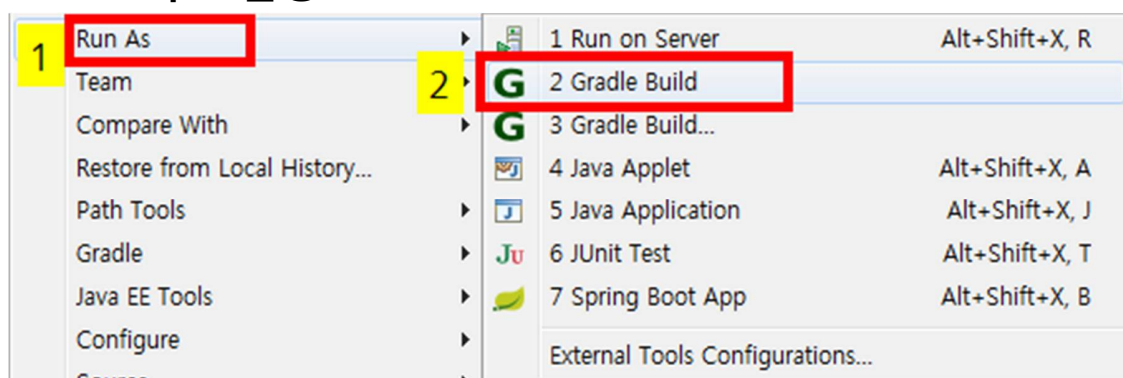
```
    context = '${project.name}'  
}  
  
// remoteDeploy 되는 target 의 tomcat 정보  
remote {  
    hostname = '192.168.13.209'  
    username = 'ykyoon'  
    password = 'qwer12#$'  
}  
}
```

7. Gradle >> Refresh All

build.gradle 이 수정되면 반드시 "Refresh Dependencies" 와 "Refresh All" 과정을 순서대로 해야 한다.



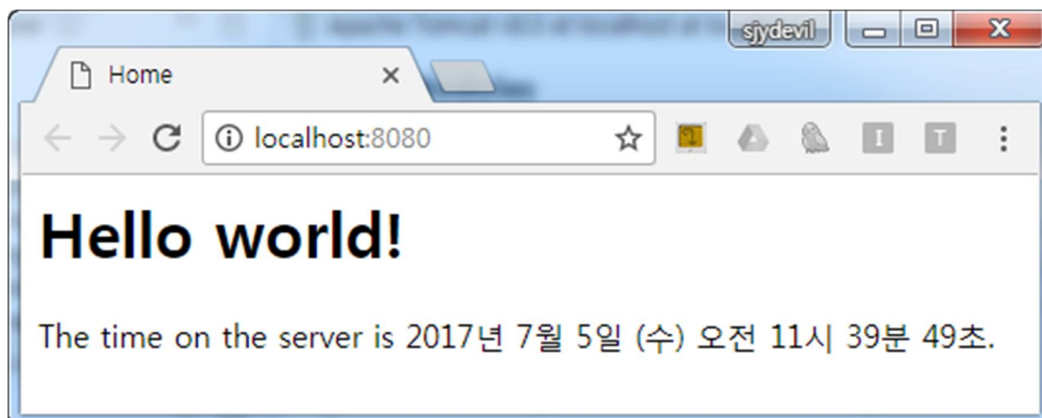
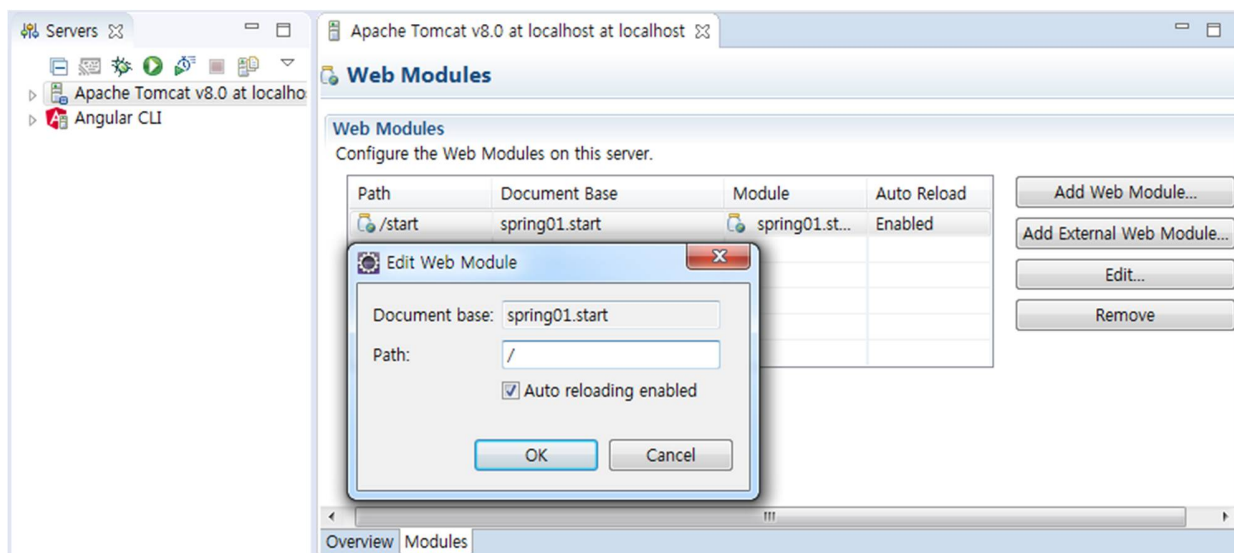
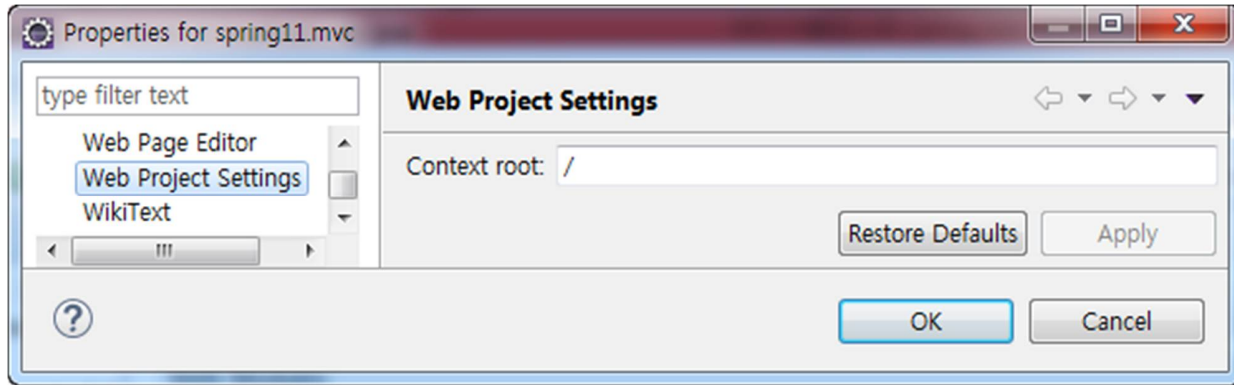
8. 프로젝트 실행



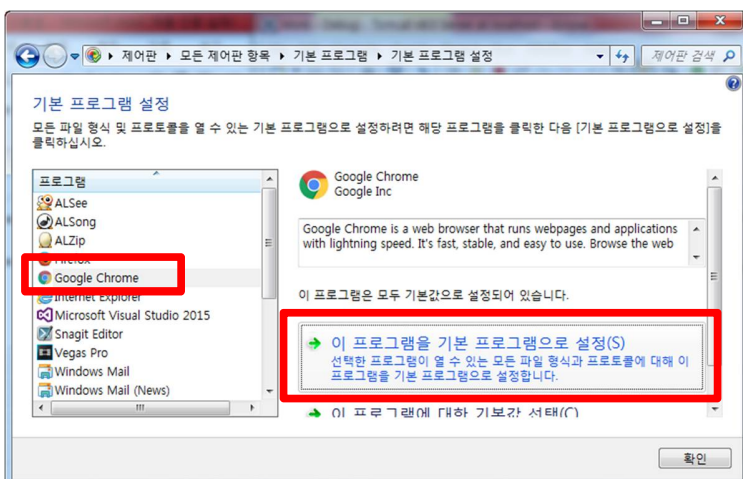
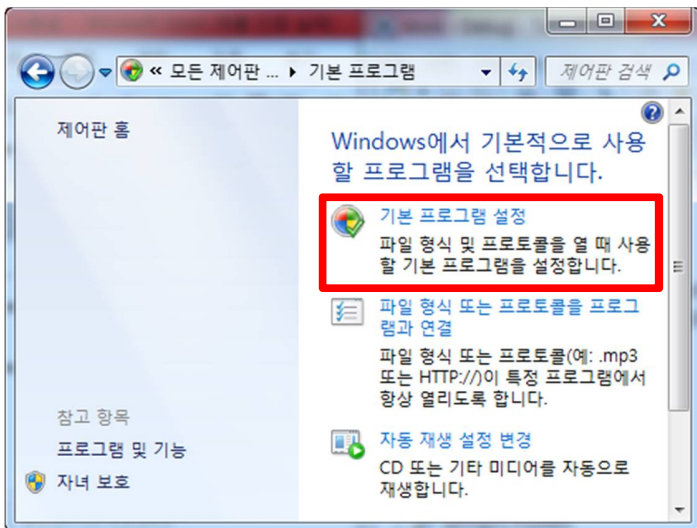
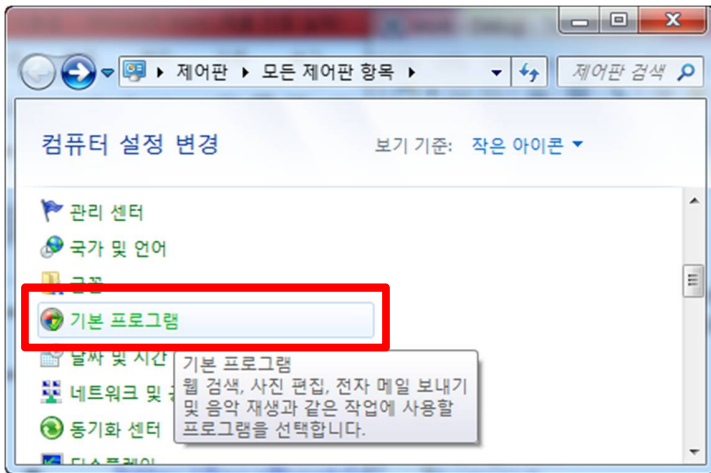
9. Context root 수정

이클립스에서 프로젝트 선택 후 오른쪽 마우스 클릭하여 "Properties" 메뉴를 선택한다.

"Properties" 창에서 "Web Project Settings" 메뉴를 클릭하여 "Context root"를 아래와 같이 바꾼다.



10. 크롬을 기본 브라우저로 설정



11. Reference

<http://www.jayway.com/2013/05/12/getting-started-with-gradle/>

<http://www.slipp.net/wiki/pages/viewpage.action?pageId=12878060>

[https://www.credera.com/blog/custom-application-development/converting-spring-boot-project-maven-gradle-
sts/](https://www.credera.com/blog/custom-application-development/converting-spring-boot-project-maven-gradle-
sts/)

<http://stackoverflow.com/questions/13925724/providedcompile-without-war-plugin>

<http://docs.spring.io/spring/docs/current/spring-framework-reference/html/transaction.html>

<http://hellogk.tistory.com/94>

<http://hellowk1.blogspot.kr/2014/02/spring-framework-transaction-aop.html>

<http://hellowk1.blogspot.kr/2015/03/spring-framework-transaction-with.html>

<http://barunmo.blogspot.kr/2013/06/mybatis.html>

<http://egloos.zum.com/springmvc/v/499291>

<http://blog.outsider.ne.kr/870>