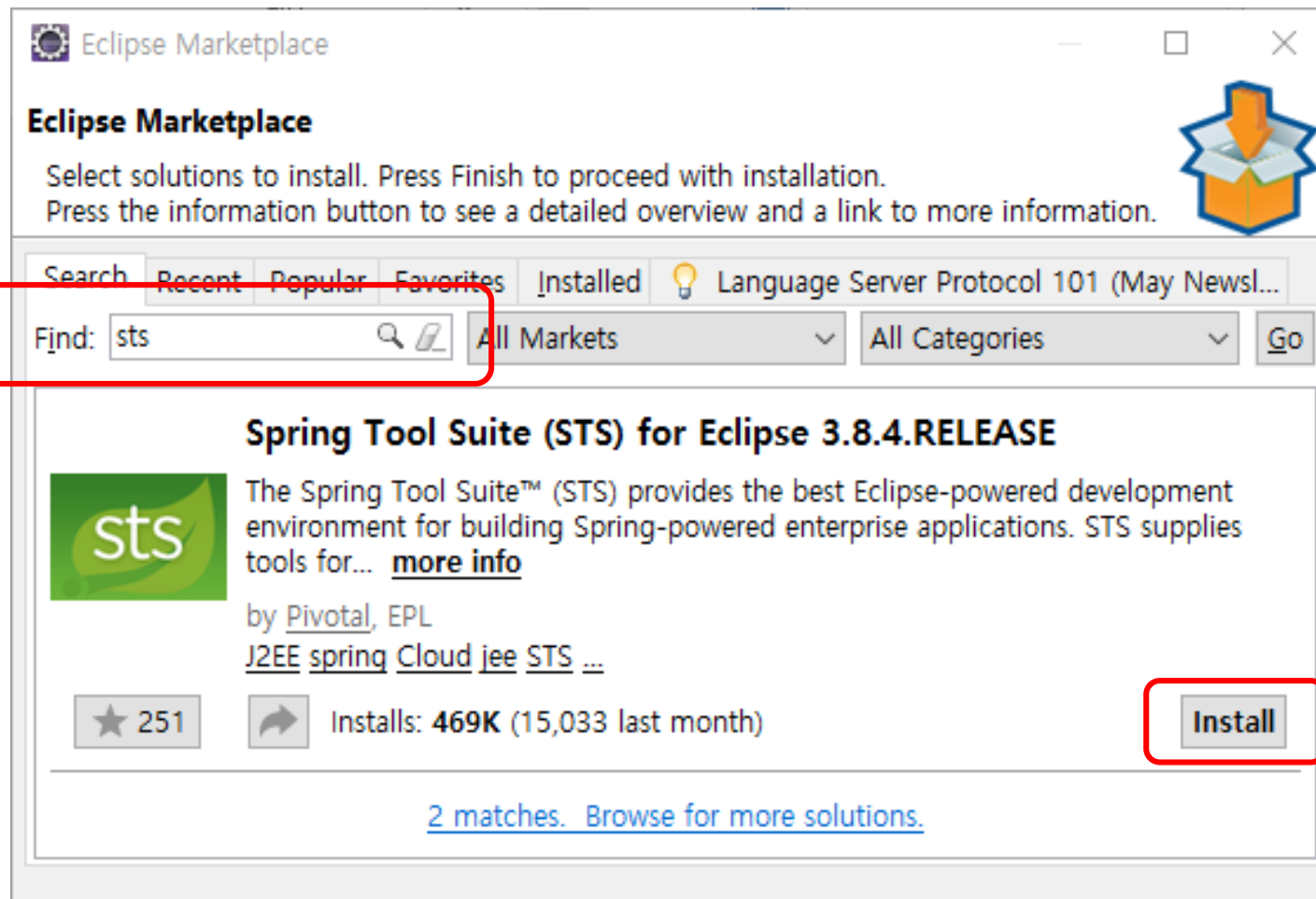


스프링 시작하기

STS(Spring Tool Suite)

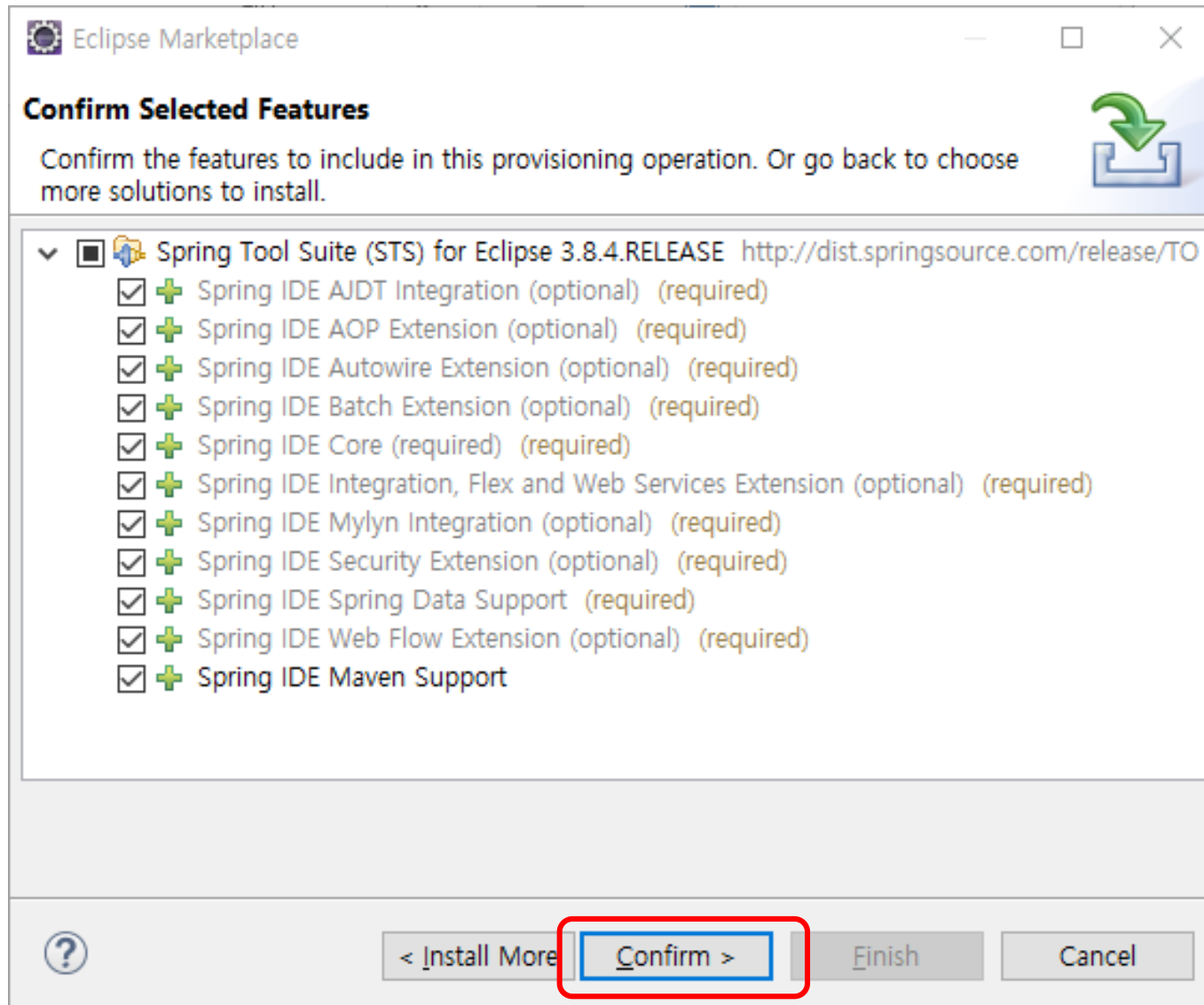
❖ Spring Tool Suite 플러그인 설치

- Help > Eclipse Marketplace
- 검색어 : STS



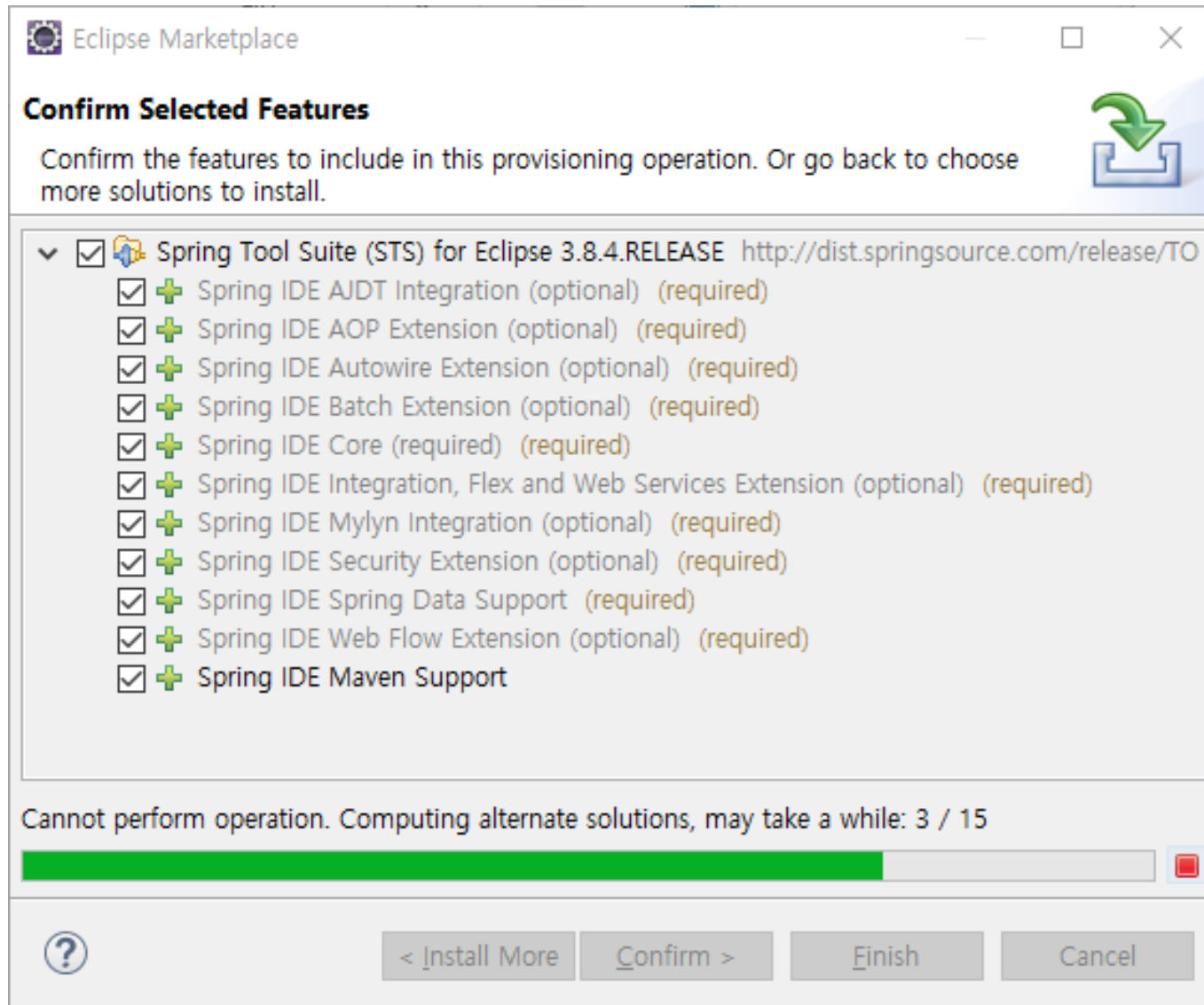
STS(Spring Tool Suite)

❖ Spring Tool Suite 플러그인 설치



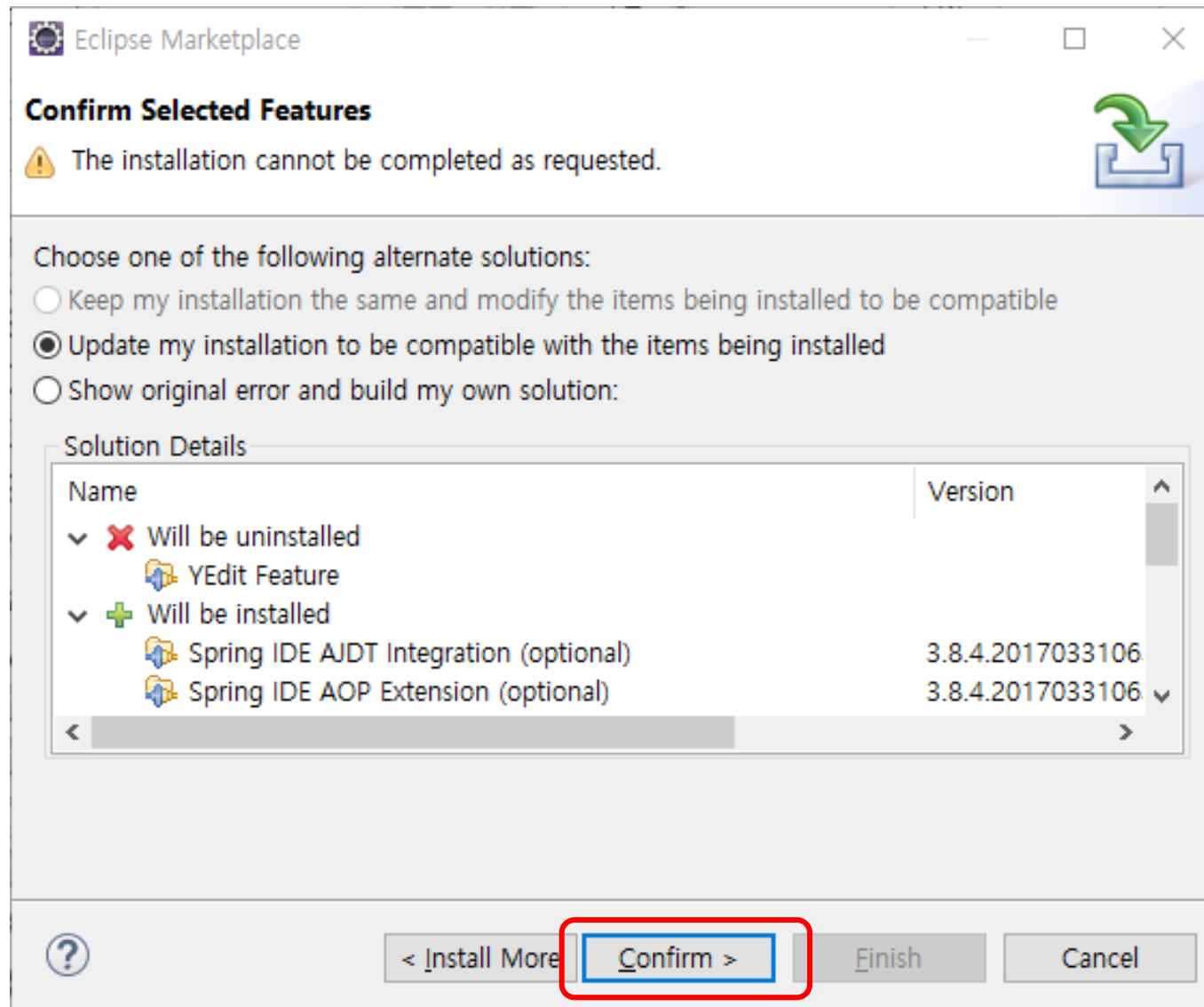
STS(Spring Tool Suite)

❖ Spring Tool Suite 플러그인 설치



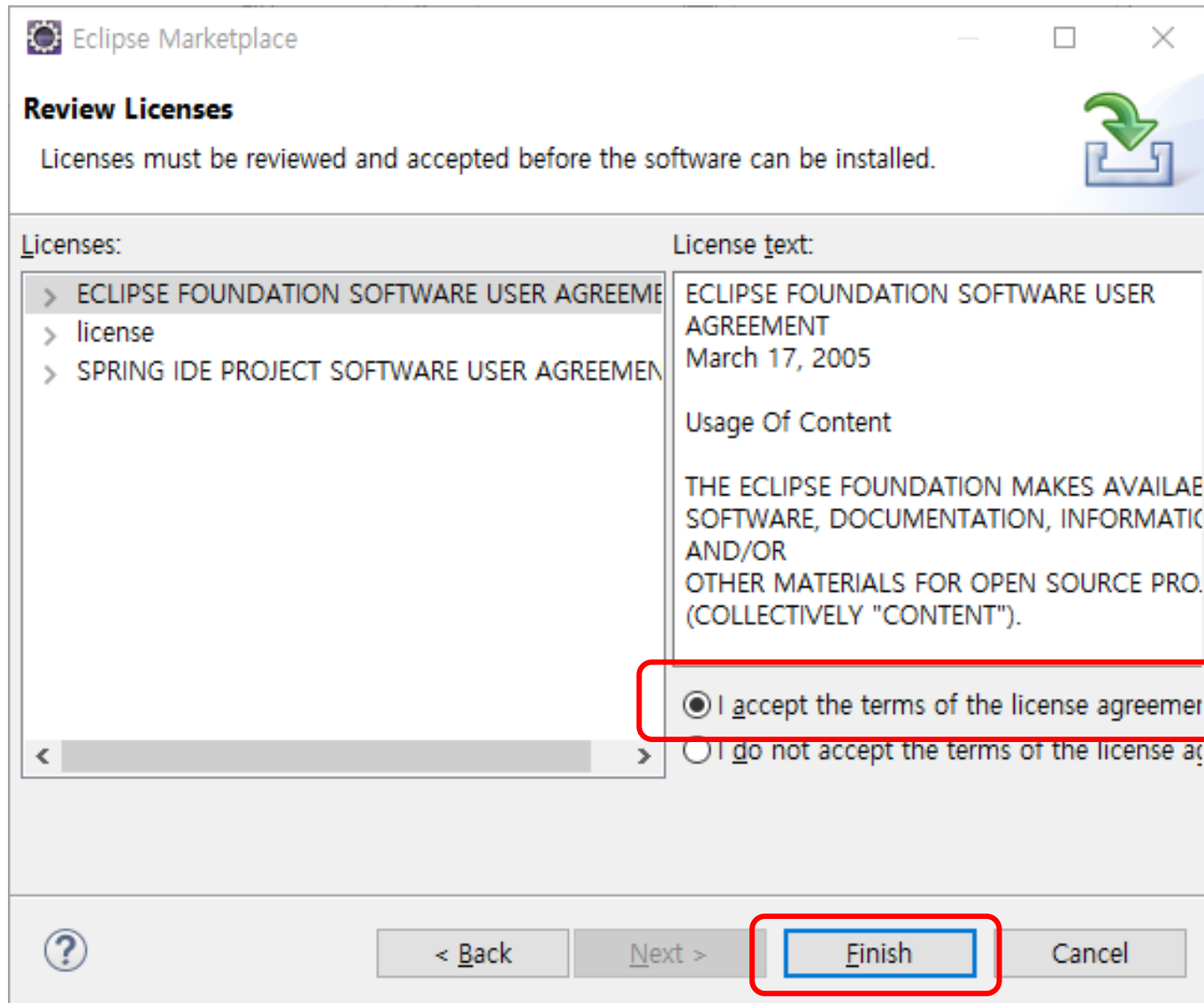
STS(Spring Tool Suite)

❖ Spring Tool Suite 플러그인 설치



STS(Spring Tool Suite)

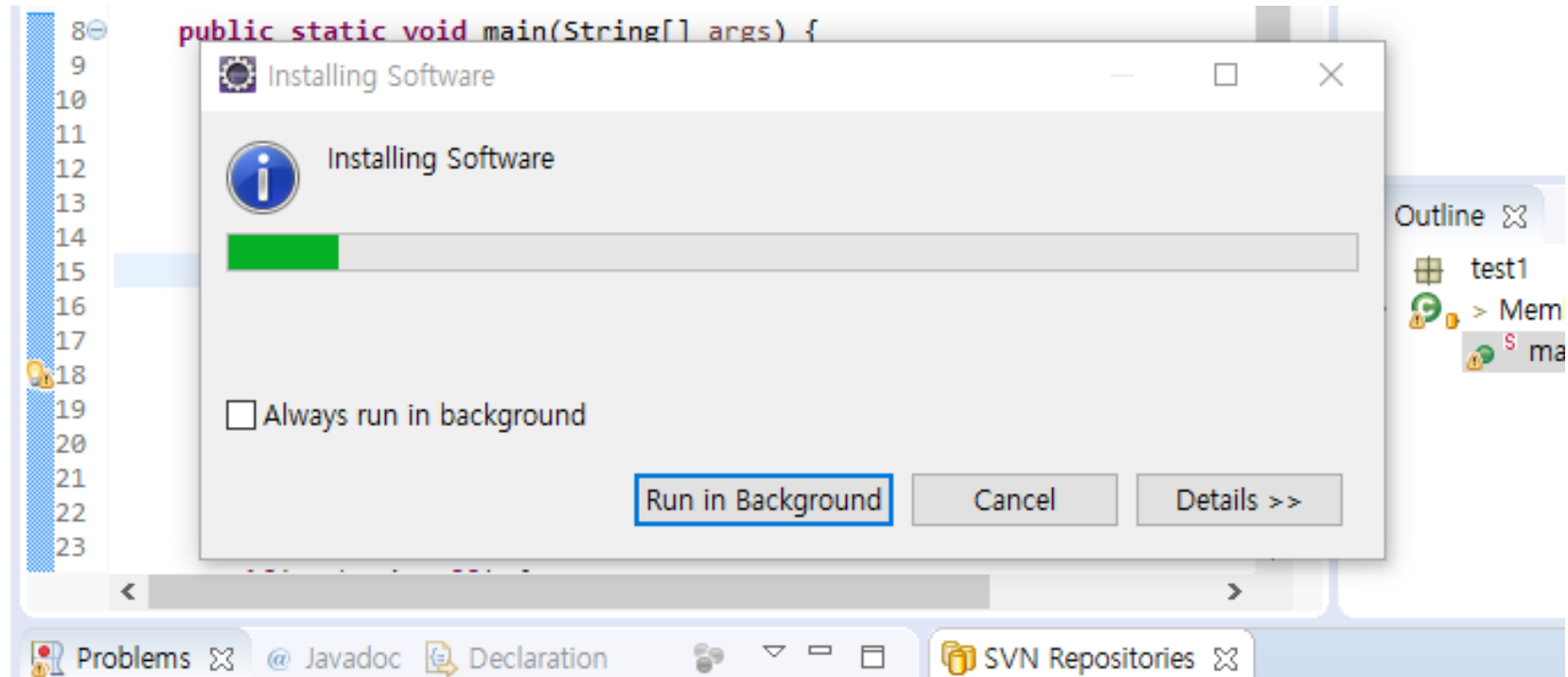
❖ Spring Tool Suite 플러그인 설치



STS(Spring Tool Suite)

❖ Spring Tool Suite 플러그인 설치

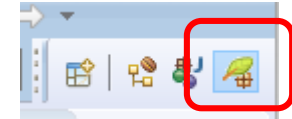
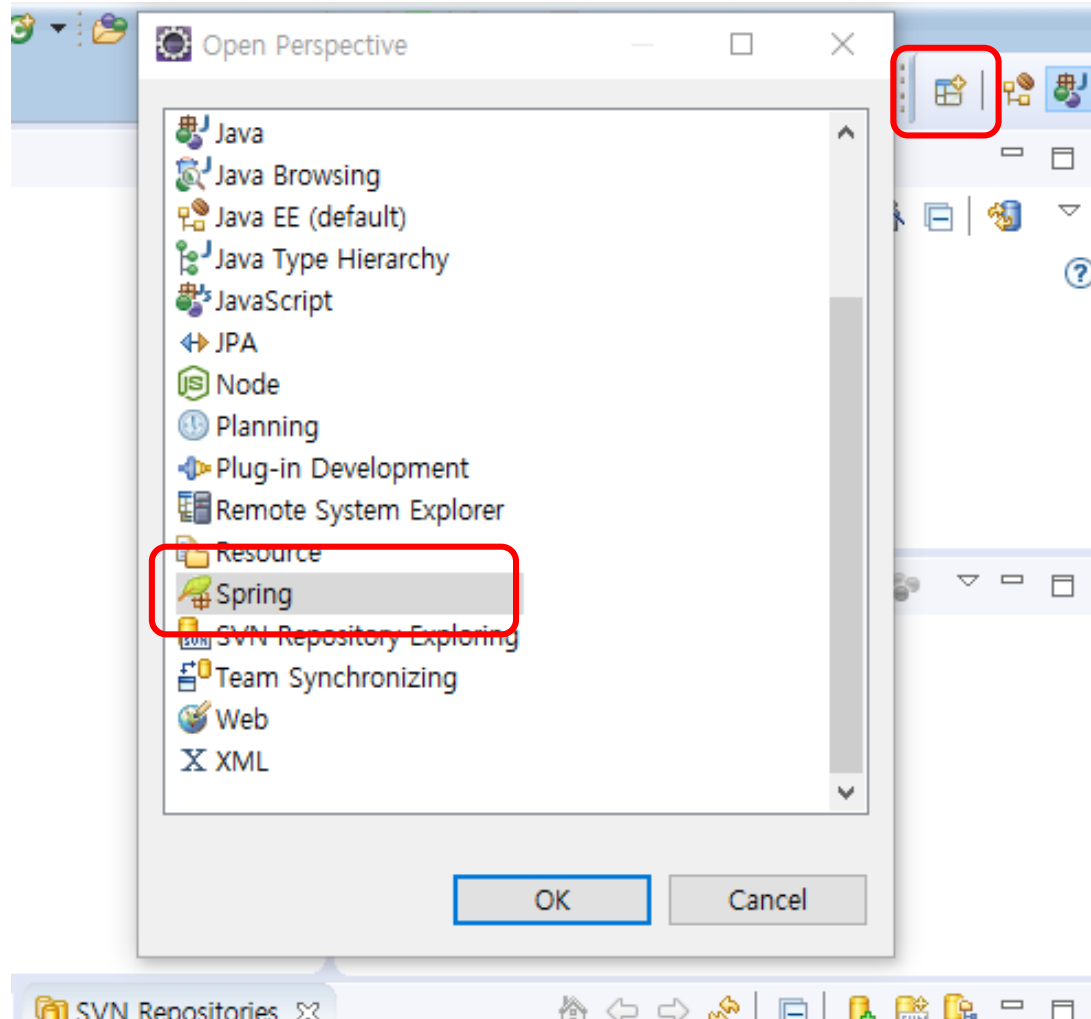
- 설치 후 이클립스 재기동



스프링 프로젝트

❖ 스프링 프로젝트 생성


- Spring perspective로 변경



스프링 프로젝트

❖ 스프링 프로젝트 생성

- New > Spring Legacy Project

 New Spring Legacy Project

Spring Legacy Project

Create a Spring project by selecting a template or simple project type.

Project name:


☒ Use default location

Location: [Browse...](#)

Select Spring version:

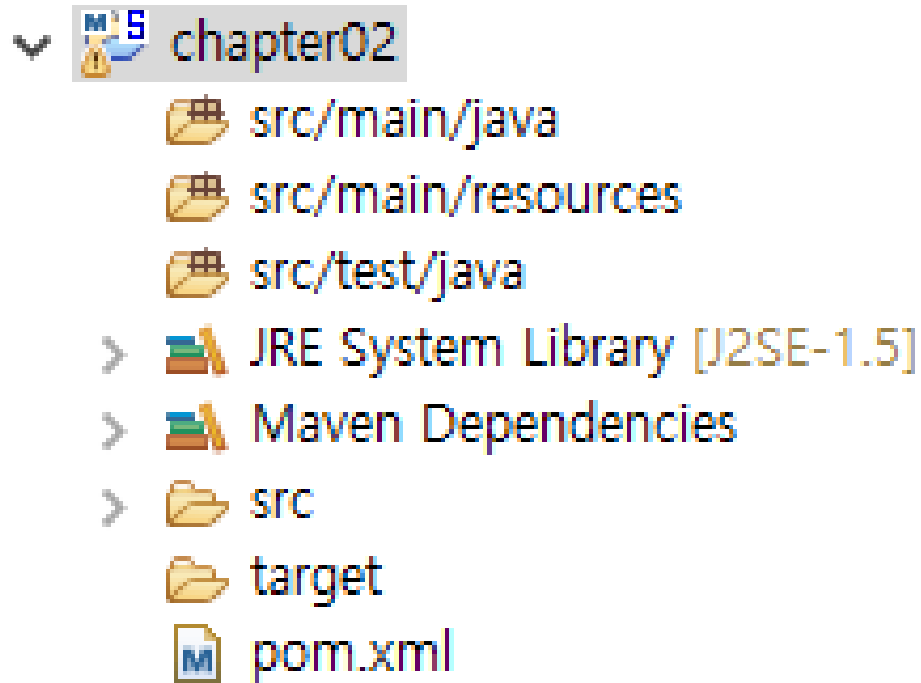
Templates:

- Simple Projects
 - Simple Java
 - Simple Spring Maven**
 - Simple Spring Web Maven
- Batch
- GemFire

 requires downloading [Configure templates...](#) [Refresh](#)

스프링 프로젝트

❖ 스프링 프로젝트 생성



- Maven으로 프로젝트 관리
 - pom.xml에서 의존 라이브러리 관리

❖ 프로젝트 기본 폴더 구조

- src/main/java
 - 자바 파일 배치
- src/main/resource
 - xml 리소스 파일(스프링 설정파일) 배치
- 폴더는 다르지만 classpath는 동일
 - classpath : 자바 클래스로더가 파일을 찾는 경로
 - 운영체제의 PATH 환경변수와 개념적으로 유사

스프링 프로젝트

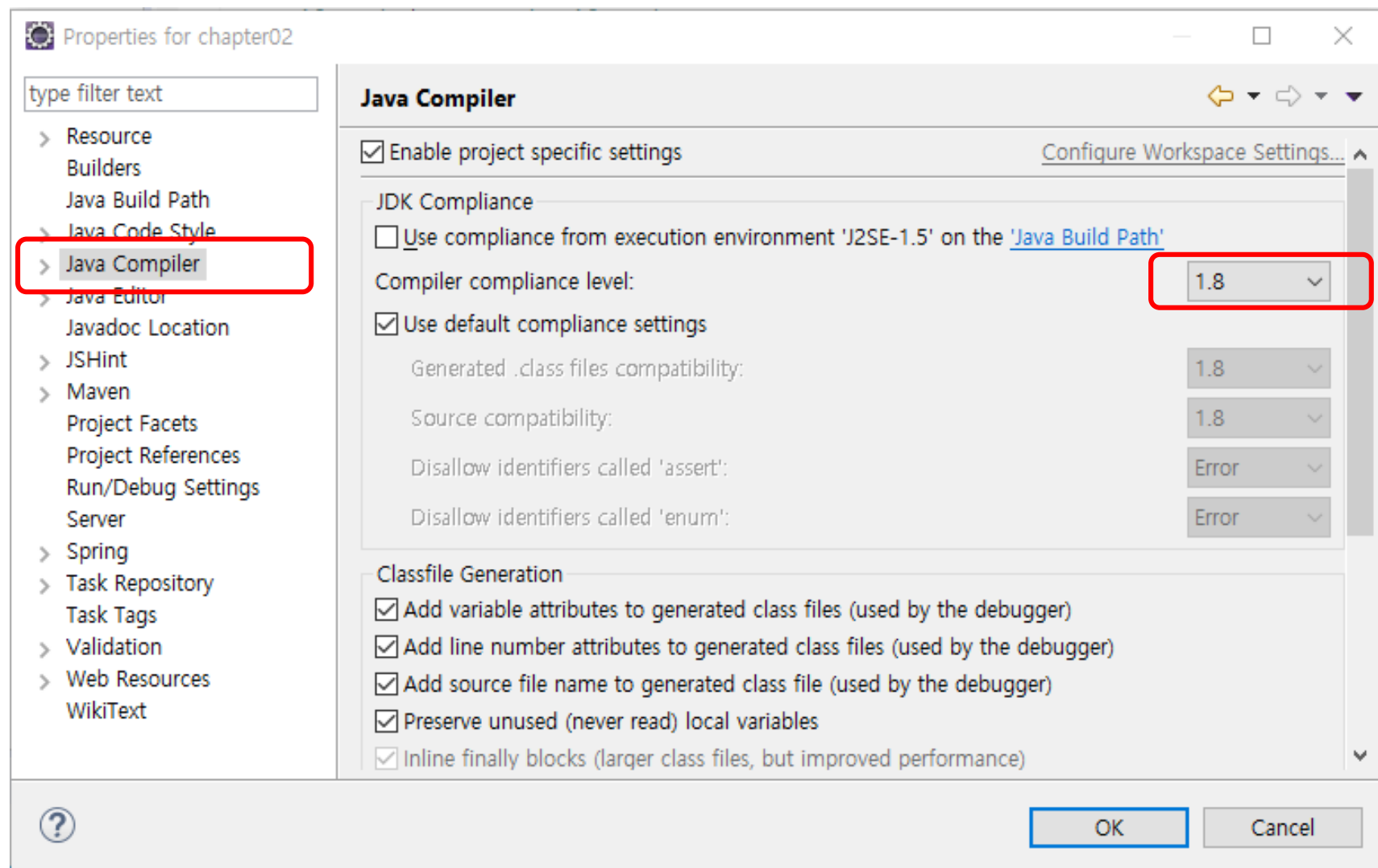
❖ 스프링 프로젝트 속성 변경

- 자바 버전 변경
 - STS는 디폴트로 Java 1.6 버전을 사용
 - Java 1.8로 변경

스프링 프로젝트

❖ 스프링 프로젝트 속성 변경

- 프로젝트 >> Properties



❖ 메이븐 의존 설정

- pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.springframework.samples</groupId>
  <artifactId>chapter02</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <properties>
    <!-- Generic properties -->
    <java.version>1.8</java.version>
    :

    <!-- Spring -->
    <spring-framework.version>4.1.7.RELEASE</spring-framework.version>
    :

  </properties>
```

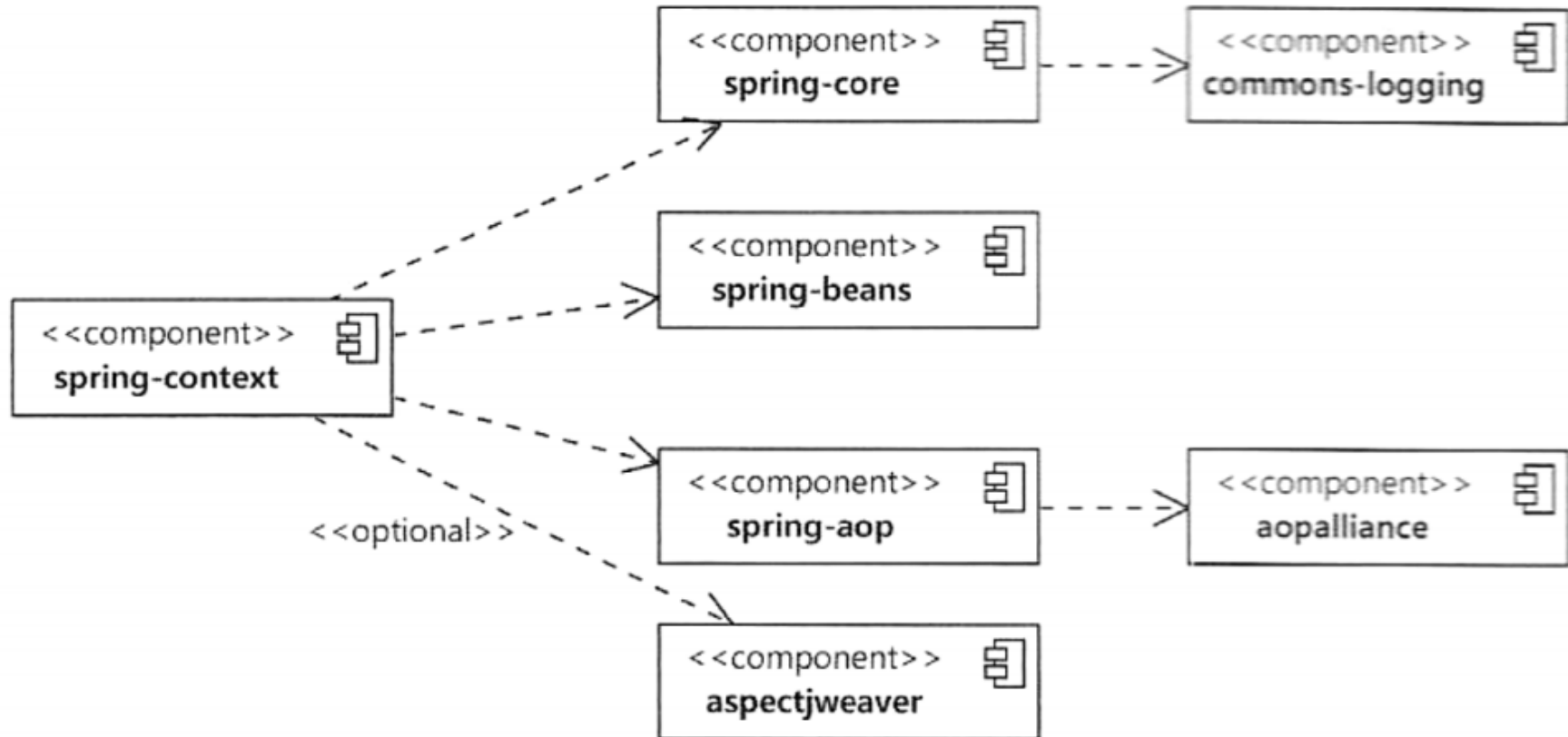
❖ 메이븐 의존 설정

○ pom.xml

```
<dependencies>
  <!-- Spring and Transactions -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${spring-framework.version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-tx</artifactId>
    <version>${spring-framework.version}</version>
  </dependency>
  :
</dependencies>
:
```

❖ 스프링 라이브러리 의존성



❖ 메이븐 레파지토리

- 중앙 레파지토리
 - 의존 라이브러리 저장소
- 로컬 레파지토리
 - 지정한 의존 라이브러리를 다운로드 받아 저장하는 곳
 - 동일 라이브러리에 대해서는 중앙 라이브러리를 사용하지 않고 로컬 레파지토리 사용
 - C:\Users\kim\.m2\repository

간단한 스프링 예제

❖ 예제

- Greeter.java
 - 콘솔에 간단한 문자열을 출력하는 클래스
- applicationContext.xml
 - 스프링 설정 파일
- Main.java, Main2.java
 - main() 메서드를 통해 스프링과 Greeter를 실행하는 자바 클래스

간단한 스프링 예제

❖ Greeter.java

```
package chap02;

public class Greeter {

    private String format;

    public String greet(String guest) {
        return String.format(format, guest);
    }

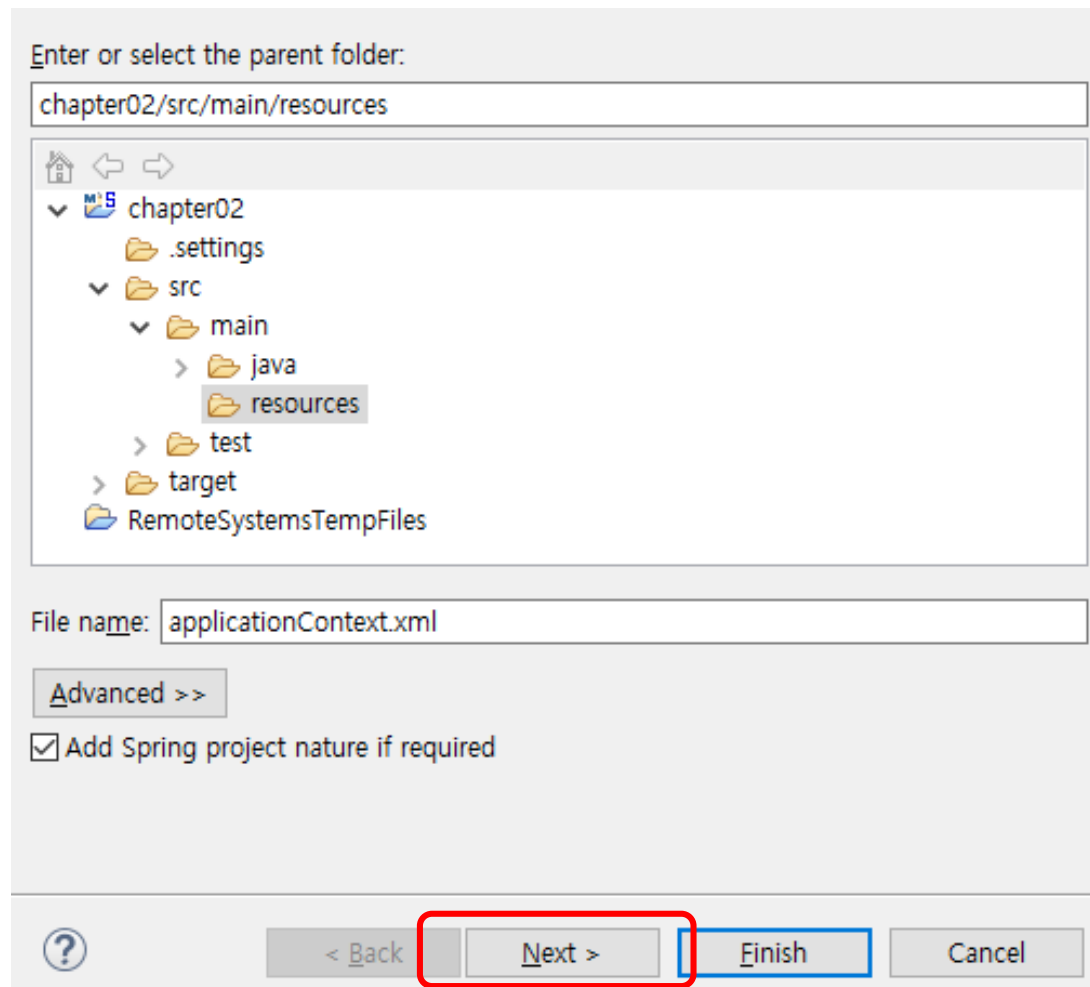
    public void setFormat(String format) {
        this.format = format;
    }

}
```

간단한 스프링 예제

❖ 스프링 설정 파일 생성







- resource >> New > Spring Bean Configuration File
 - File name : applicationContext.xml




간단한 스프링 예제

❖ 스프링 설정 파일 생성

Select desired XSD namespace declarations:

- ☐  cache - http://www.springframework.org/schema/cache
- ☐  context - http://www.springframework.org/schema/context
- ☐  jee - http://www.springframework.org/schema/jee
- ☐  lang - http://www.springframework.org/schema/lang
- ☐  task - http://www.springframework.org/schema/task
- ☐  tx - http://www.springframework.org/schema/tx

Select desired XSD (if none is selected the default will be used):

 < Back Next > **Finish** Cancel

간단한 스프링 예제

❖ resources/applicationContext.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">

</beans>
```

간단한 스프링 예제

❖ resources/applicationContext.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">

    <bean id="greeter" class="chap02.Greeter">
        <property name="format" value="%s, 안녕하세요!" />
    </bean>

</beans>
```

- 빈(bean) 객체
 - 스프링이 생성하는 객체
 - id : 빈 식별자
 - class : 빈 객체로 생성할 전체 클래스명

간단한 스프링 예제

❖ 빈 객체의 프로퍼티

- 객체의 getter/setter 메서드로 접근할 수 있는 객체의 필드
- 프로퍼티명은 getter/setter 메서드 명에서 get/set을 제외한 명칭
 - setFormat() → format
- <bean>태그의 하위 태그 <property> 태그로 정의
 - name 속성 : 프로퍼티명
 - value 속성 : 프로퍼티 값(setter 메서드의 인자로 전달)

```
<bean id="greeter"  
      class="chap02.Greeter">  
  <property  
    name="format"  
    value="%s, 안녕하세요!" />  
</bean>
```



```
Greeter greeter = new Greeter();  
greeter.setFormat("%s, 안녕하세요!");
```


간단한 스프링 예제

❖ Main.java

```
package chap02;

import org.springframework.context.support.GenericXmlApplicationContext;

public class Main {

    public static void main(String[] args) {
        GenericXmlApplicationContext ctx = new GenericXmlApplicationContext(
            "classpath:applicationContext.xml");
        Greeter g = ctx.getBean("greeter", Greeter.class);

        String msg = g.greet("스프링");
        System.out.println(msg);
        ctx.close();
    }
}
```

```
6월 08, 2017 10:23:40 오전 org.springframework.beans.factory.xml.XmlBeanDefinitionRead
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
6월 08, 2017 10:23:40 오전 org.springframework.context.support.GenericXmlApplicationCc
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@17!
스프링, 안녕하세요!
6월 08, 2017 10:23:40 오전 org.springframework.context.support.GenericXmlApplicationCc
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@179d3l
```

스프링은 객체 컨테이너

❖ 스프링 컨테이너

- 인터페이스
 - ApplicationContext
- 구현 클래스
 - GenericXmlApplicationContext
 - AnnotationConfigApplicationContext 등
- 역할
 - 빈(bean) 객체의 생성 및 초기화

스프링은 객체 컨테이너

❖ 스프링 컨테이너

- 스프링 컨테이너 초기화

```
GenericXmlApplicationContext ctx = new GenericXmlApplicationContext(  
    "classpath:applicationContext.xml");
```

- 스프링 컨테이너에서 빈 객체 추출

```
Greeter g = ctx.getBean("greeter", Greeter.class);
```

스프링은 객체 컨테이너

❖ Main2.java

```
package chap02;

import org.springframework.context.support.GenericXmlApplicationContext;

public class Main2 {

    public static void main(String[] args) {
        GenericXmlApplicationContext ctx =
            new GenericXmlApplicationContext(
                "classpath:applicationContext.xml");
        Greeter g1 = ctx.getBean("greeter", Greeter.class);
        Greeter g2 = ctx.getBean("greeter", Greeter.class);

        System.out.println("(g1 == g2) = " + (g1 == g2));
        ctx.close();
    }
}
```

6월 08, 2017 10:25:45 오전 org.springframework.beans.factory.xml.XmlBeanDefinitionRead
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
6월 08, 2017 10:25:46 오전 org.springframework.context.support.GenericXmlApplicationContext
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@17!
(g1 == g2) = true
6월 08, 2017 10:25:46 오전 org.springframework.context.support.GenericXmlApplicationContext
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@179d3l

스프링은 객체 컨테이너

❖ 싱글톤 객체

- 스프링의 빈 객체는 싱글톤 객체로 운영
- 동일 클래스지만 빈 객체 정의가 다르면 각자 싱글톤 객체로 운영

```
<?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">

  <bean id="greeter" class="chapter02.Greeter">
    <property name="format" value="%s, 안녕하세요!" />
  </bean>

  <bean id="greeter2" class="chapter02.Greeter">
    <property name="format" value="%s, 안녕하세요!" />
  </bean>
</beans>
```

스프링은 객체 컨테이너

❖ Main2.java

```
package chap02;

import org.springframework.context.support.GenericXmlApplicationContext;

public class Main2 {

    public static void main(String[] args) {
        GenericXmlApplicationContext ctx =
            new GenericXmlApplicationContext(
                "classpath:applicationContext.xml");
        Greeter g1 = ctx.getBean("greeter", Greeter.class);
        Greeter g2 = ctx.getBean("greeter1", Greeter.class);

        System.out.println("(g1 == g2) = " + (g1 == g2));
        ctx.close();
    }
}
```

6월 08, 2017 10:34:18 오전 org.springframework.beans.factory.xml.XmlBeanDefinitionRead
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
6월 08, 2017 10:34:18 오전 org.springframework.context.support.GenericXmlApplicationCc
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@179d3f1
(g1 == g2) = false
6월 08, 2017 10:34:18 오전 org.springframework.context.support.GenericXmlApplicationCc
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@179d3f1