

01강 스프링 시작하기

스프링이란?

- 스프링에 대해 가장 잘 알려진 정의는 다음과 같다
 - 자바 엔터프라이즈 개발을 편하게 해주는 오픈 소스 경량급 애플리케이션 프레임워크
- 일반적으로 프레임워크란 전체 프로그램의 특정 부분을 손쉽게 만들거나 한가지 기능에 대해 간단한 설정만으로 해당 기술을 제공하는 일련의 구조체를 의미한다.
- 그러나 스프링 프레임워크는 프로그램의 특정 부분이나 기술에 국한하지 않고 프로그램의 전 영역을 포괄하는 범용 프레임워크이다.

스프링이란?

- 스프링은 하나의 기술에 국한하지 않는 수많은 프로젝트의 집합체이다.
- 따라서 모든 것을 배우는 것은 시간상 불가능하며 프레임워크를 관통하는 핵심 기술과 웹 MVC 의 기본에 대해서만 살펴 보기로 한다.

Technical support covers

OpenJDK*	Spring Cloud Zookeeper	Spring Data for Apache Cassandra™	Spring Kerberos
Apache Tomcat	Spring Cloud Kubernetes	Spring Data Commons	Spring REST Docs
VMware tc Server	Spring Cloud CLI	Spring Data KeyValue	Spring Retry
AspectJ	Spring Cloud OpenFeign	Spring Data LDAP	Spring Statemachine
Micrometer	Spring Cloud Gateway	Spring Data JDBC	Spring Security
Reactor	Spring Cloud Function	Spring Data JPA	Spring Security OAuth
Spring AMQP	Spring Cloud Contract	Spring Data MongoDB	Spring Security SAML
Spring Batch	Spring Cloud Task	Spring Data GemFire	Spring Cloud Data Flow
Spring Boot	Spring Cloud Stream	Spring Data for Apache Geode™	Spring Session
Spring Cloud Commons	Spring Cloud Stream Binder for Apache Kafka®	Spring Data Redis	Spring Session for MongoDB
Spring Cloud Config	Spring Cloud Stream Binder for RabbitMQ	Spring Data REST	Spring Session for GemFire/Apache Geode™
Spring Cloud Netflix	Spring Cloud Stream Binder for Kafka Streams	Spring Data Solr	Spring Tool Suite
Spring Cloud Bus	Spring Cloud Stream Binder for AWS Kinesis	Spring Framework	Spring Tools 4
Spring Cloud Cloud Foundry	Spring Cloud Vault	Spring HATEOAS	Spring Vault
Spring Cloud Open Service Broker	Spring CredHub	Spring Integration (Core modules + Kafka + AWS)	Spring Cloud Skipper
Spring Cloud Consul		Spring for Apache Kafka®	Spring Shell
Spring Cloud Security		Spring LDAP	Spring Cloud Stream/Task App Starters
Spring Cloud Sleuth			

스프링이란?

- 스프링의 주요 특징

- DI(의존 주입) 지원
- AOP(관점 지향 프로그래밍) 지원
- IOC(제어의 역전)
- MVC 웹 프레임워크 제공
- JDBC, JPA 연동 기술 및 선언적 트랜잭션 처리 등 DB 연동 기술 지원

- 스프링에는 다양한 프로젝트가 진행되는데 그중 자주 사용되는 프로젝트는 다음과 같다.

- 스프링 데이터 : 적은 양의 코드로 데이터를 연동을 처리할 수 있도록 도와주는 프레임워크
- 스프링 시큐리티 : 인증/인가와 관련된 프레임워크
- 스프링 배치 : 로깅/추적, 작업 통계, 실패처리등 배치 처리에 필요한 기본 기능과 웹 기반 화면을 제공

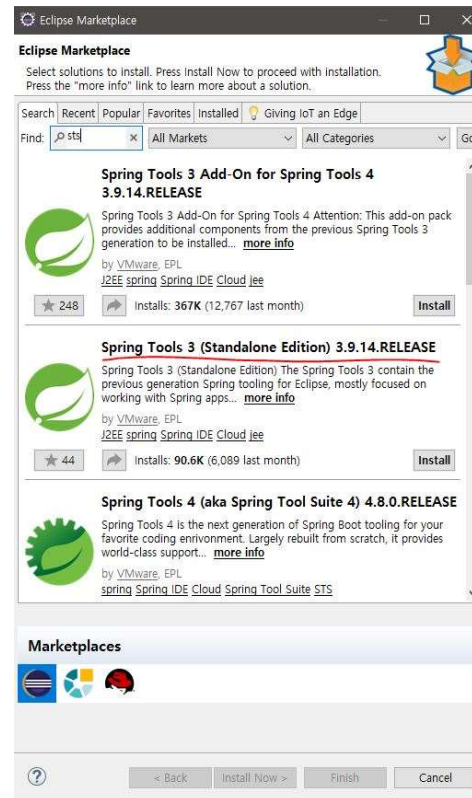
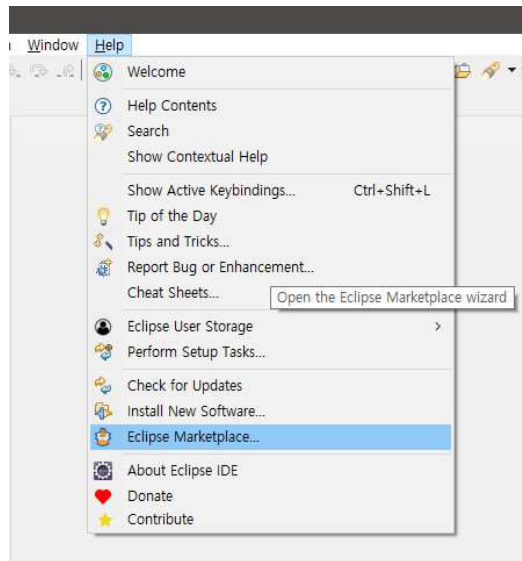
스프링이란?

- 스프링을 학습하기 위해서 다음과 같은 실습환경이 필요하다.
 - JDK
 - maven
 - eclipse
- SPRING 5.X 버전부터는 JDK 1.8이후로 사용가능하다.(우리 수업은 스프링4.X 버전을 사용할 예정이며 학습하는데는 스프링 버전과 관련없이 아무거나 사용가능하다.)
- 메이븐도 필요한 모듈이지만 이클립스 EE 4.3버전 이후로는 이클립스에 플러그인이 내장되어있으므로 4.3.이후 버전을 사용하면 굳이 설치할 필요는 없다.
- 이클립스에 STS(spring tool suite)플러그인을 직접 설치할 수도 있고 스프링 사이트에서 STS가 내장된 이클립스를 직접 다운 받을 수도 있다.

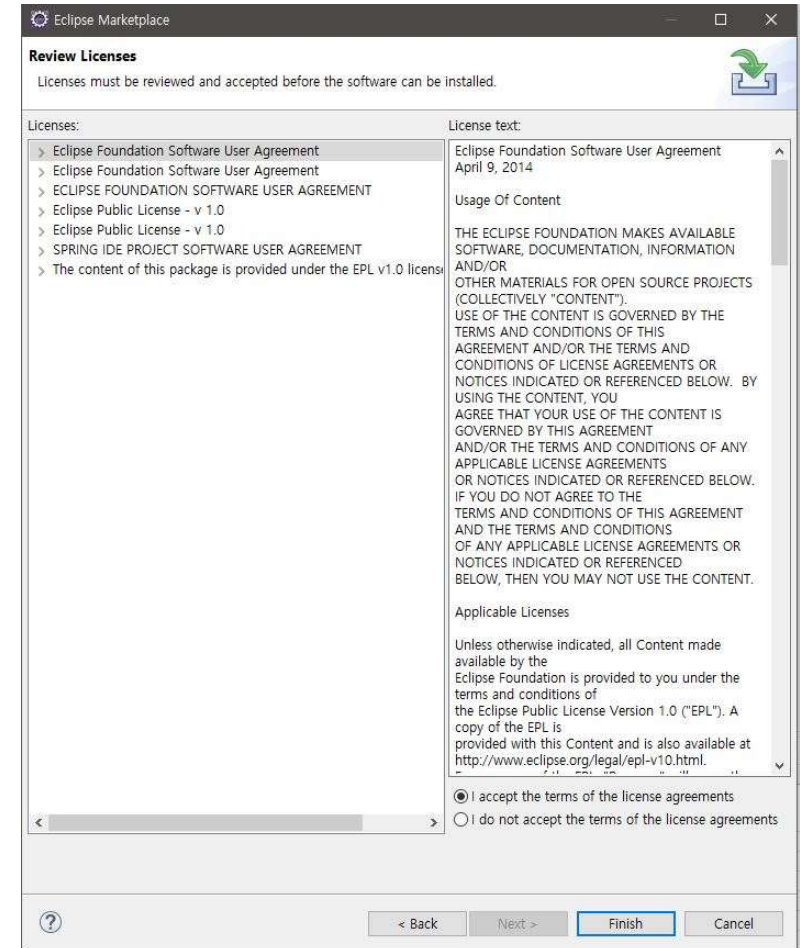
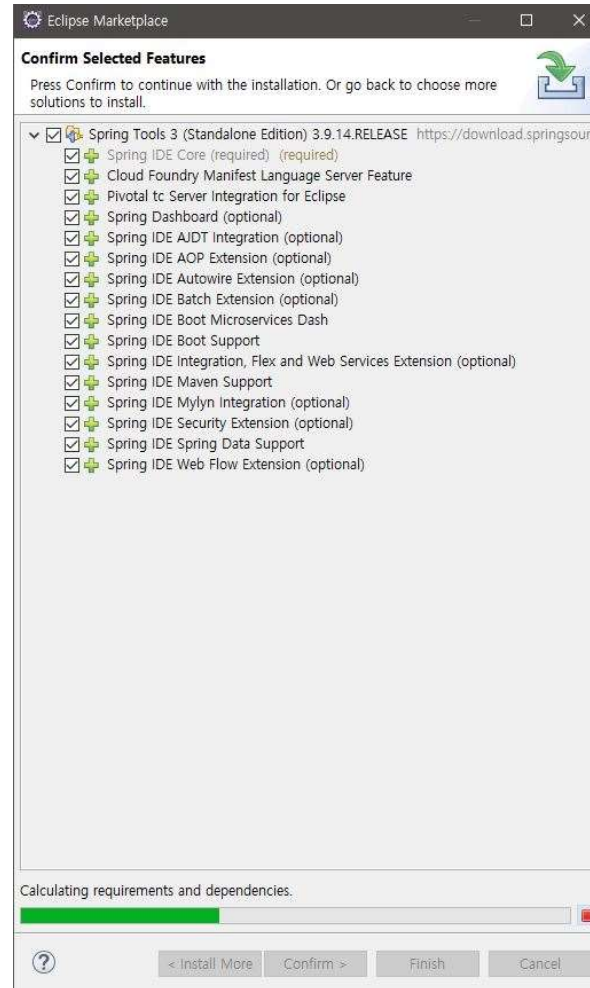
스프링이란?

- 이클립스에 sts플러그인을 설치하는 법을 알아보자
 - help 메뉴에서 Eclipse marketplace를 선택한다.
 - sts를 검색한 후 Spring Tools 3(Standalone Edition)버전으로 설치한다.
 - (sts4는 스프링 부트가 포함된 버전으로 우리 수업에서는 다루지 않는다.)
- 절차에 따라 설치 후 다시 시작을 한다.

스프링이란?

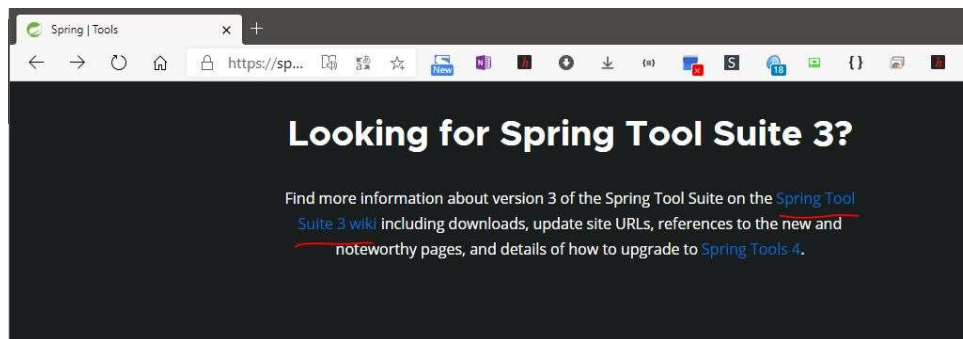


스프링이란?



스프링이란?

- 스프링 공식 홈페이지(<https://spring.io/tools>) 에서 다운 받는다_(추가)



Latest STS3 Downloads

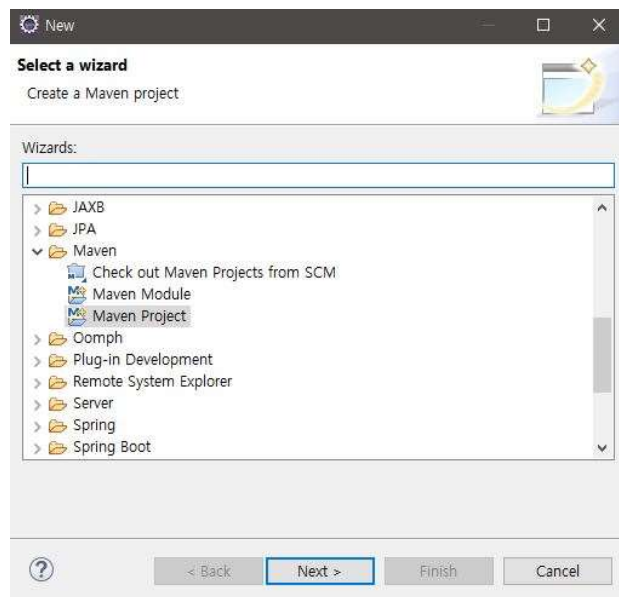
Spring Tool Suite 3.9.14 (New and Noteworthy)

full distribution on Eclipse 4.17

- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-win32-x86_64.zip
- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-macosx-cocoa-x86_64.dmg
- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-linux-gtk-x86_64.tar.gz

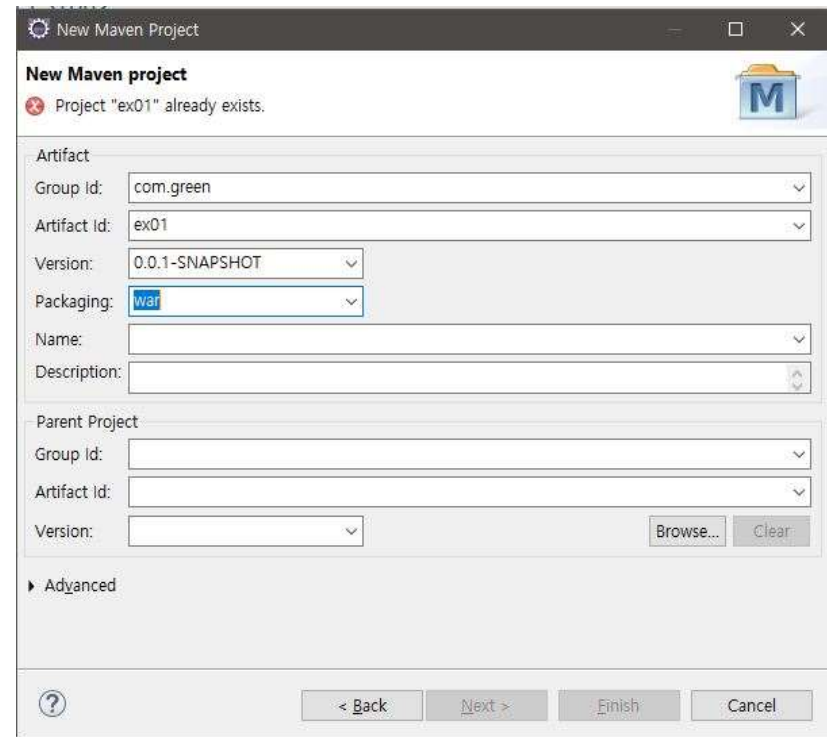
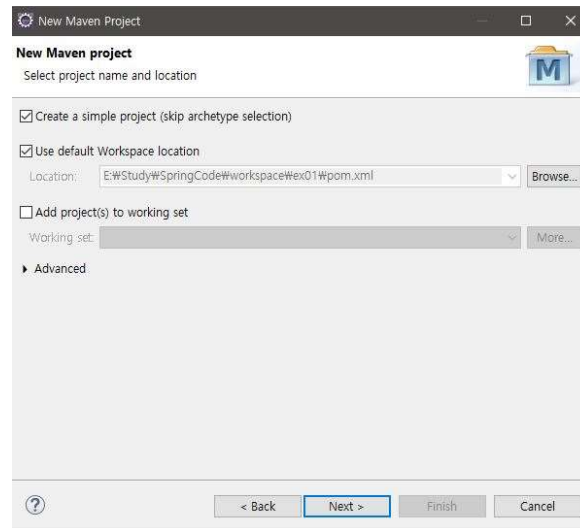
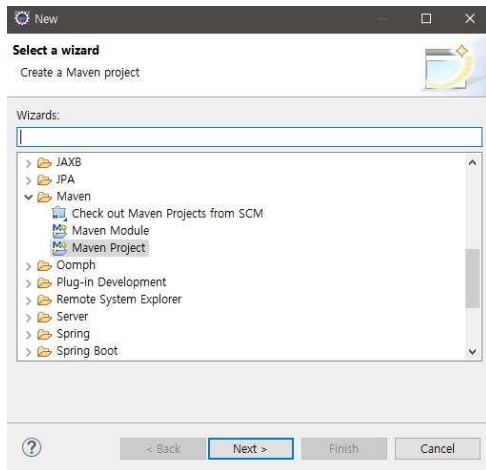
스프링 프로젝트 시작하기

- 앞으로 이클립스에 STS 플러그인 설치 버전으로 학습한다.
- 메이븐 프로젝트를 생성한다.



스프링 프로젝트 시작하기

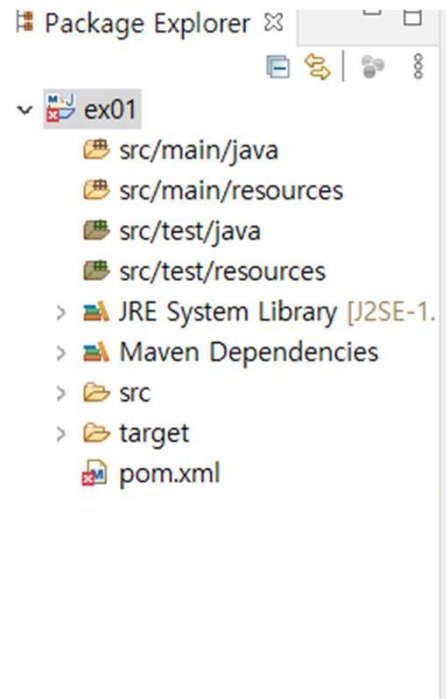
- 앞으로 이클립스에 STS 플러그인 설치 버전으로 학습한다.
- 메이븐 프로젝트를 생성한다.



- Group id : com.green
- Artifact id : ex01

스프링 프로젝트 시작하기

- src/main/java : 자바 소스 위치 폴더
- src/main/resource : 자원 파일 위치 폴더



스프링 프로젝트 시작하기

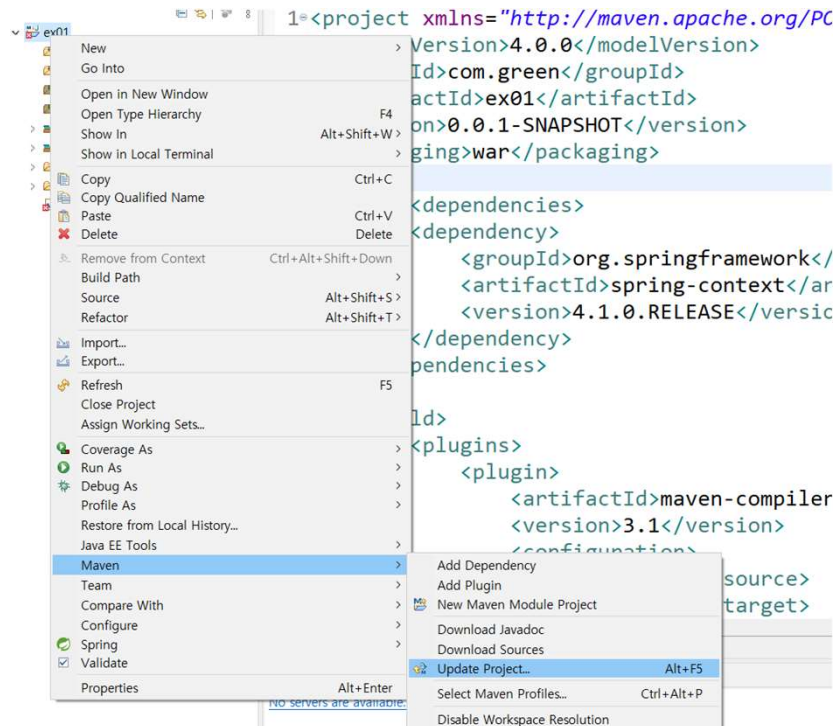
- POM.XML에 다음 내용을 추가한다.

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>4.1.0.RELEASE</version>
  </dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.1</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
        <encoding>utf-8</encoding>
      </configuration>
    </plugin>
  </plugins>
</build>
```

스프링 프로젝트 시작하기

- 프로젝트에 우클릭후 메이븐 업데이트를 진행해서 해당 의존 파일을 다운받는다



스프링 프로젝트 시작하기

- 메이븐 의존 설정(dependencies)
 - 의존을 추가한다는 것은 자바 애플리케이션에 클래스 패스를 추가한다는 것을 의미한다.
 - 각 모듈은 아티팩트 단위로 관리 되는데 이 아티팩트의 완전한 이름은 "아티팩트이름-버전.jar"로 되어있다.
 - 앞서 본 내용에 의하면 spring-context-4.1.0.RELEASE.jar 아티팩트 파일을 추가한다는 것을 의미한다.
- POM.xml에 위 내용을 추가하면 메이븐 로컬 레포지토리를 검색하고 없으면 원격 레포지토리로 부터 해당 의존 파일을 다운받는다.
 - 로컬 위치 : [사용자폴더]\.m2\repository
 - 보통 메이븐 컴파일할때 해당 파일을 다운로드 받는다.
 - 해당 파일을 다운 받은때는 spring-context-4.1.0.RELEASE.pom을 먼저 받는데 여기에는 해당 아티팩트를 실행하기 위해서 필요한 파일들의 목록이 기록되어서 이파일들을 먼저 받는다.(이런 관계를 의존 관계라고 한다.)
- web.xml is missing 에러 발생시 직접 web.xml파일을 작성한다.

스프링 프로젝트 시작하기

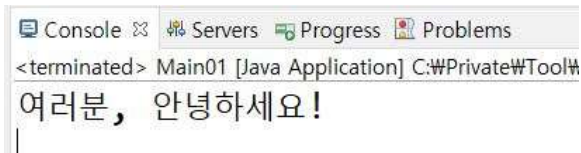
- 기본 패키지에 다음 두개의 파일을 생성한다.
 - Greeter.java
 - Main.java

```
public class Greeter {  
    private String format;  
  
    public String greet(String guest) {  
        return String.format(format, guest);  
    }  
  
    public void setFormat(String format) {  
        this.format = format;  
    }  
}
```

```
public class Main01 {  
  
    public static void main(String[] args) {  
        Greeter gt = new Greeter();  
        gt.setFormat("%s, 안녕하세요!");  
        String msg = gt.greet("여러분");  
  
        System.out.println(msg);  
    }  
}
```


스프링 프로젝트 시작하기

- Main을 실행 해보면 다음과 같이 실행이 된다.(Java 기본)



The screenshot shows an IDE console window with tabs for Console, Servers, Progress, and Problems. The Console tab is active, displaying the output of a Java application named 'Main01 [Java Application]'. The output text is '여러분, 안녕하세요!' (Hello, everyone!).

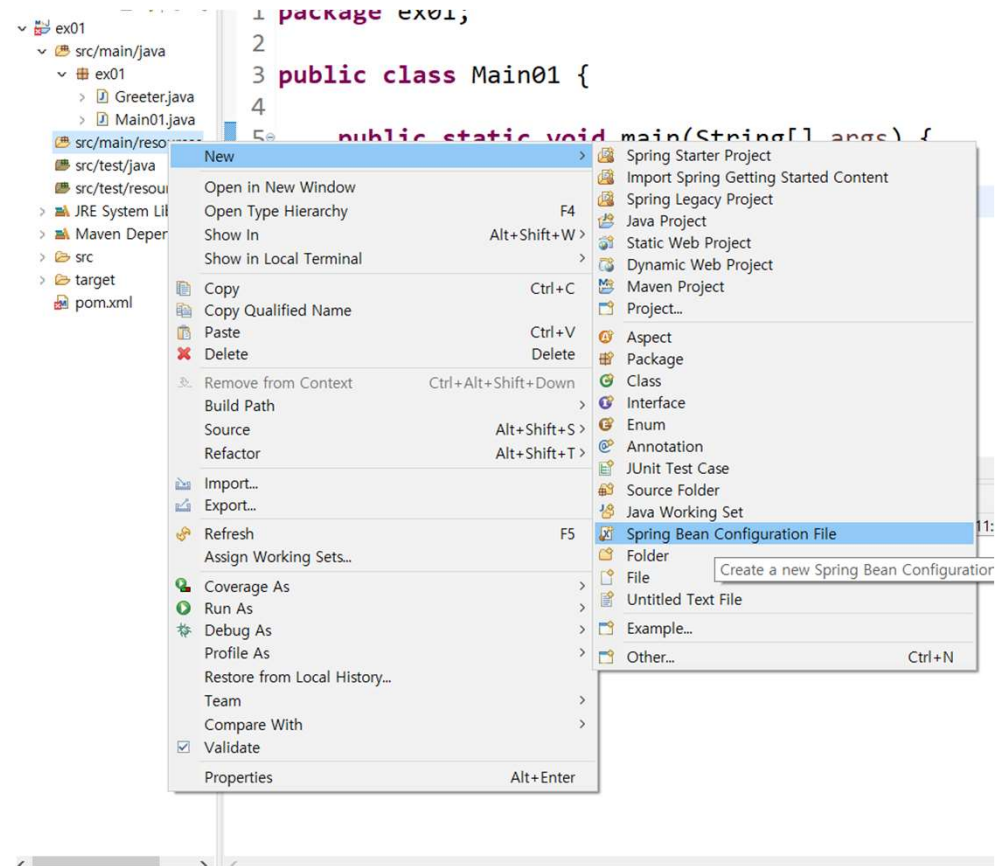
- Java에서는 특정 객체의 기능을 실행하기 위해서는 클래스의 인스턴스를 생성해야 한다.

```
public class Main01 {  
  
    public static void main(String[] args) {  
        Greeter gt = new Greeter();  
        gt.setFormat("%s, 안녕하세요!");  
        String msg = gt.greet("여러분");  
  
        System.out.println(msg);  
    }  
}
```

인스턴스 생성

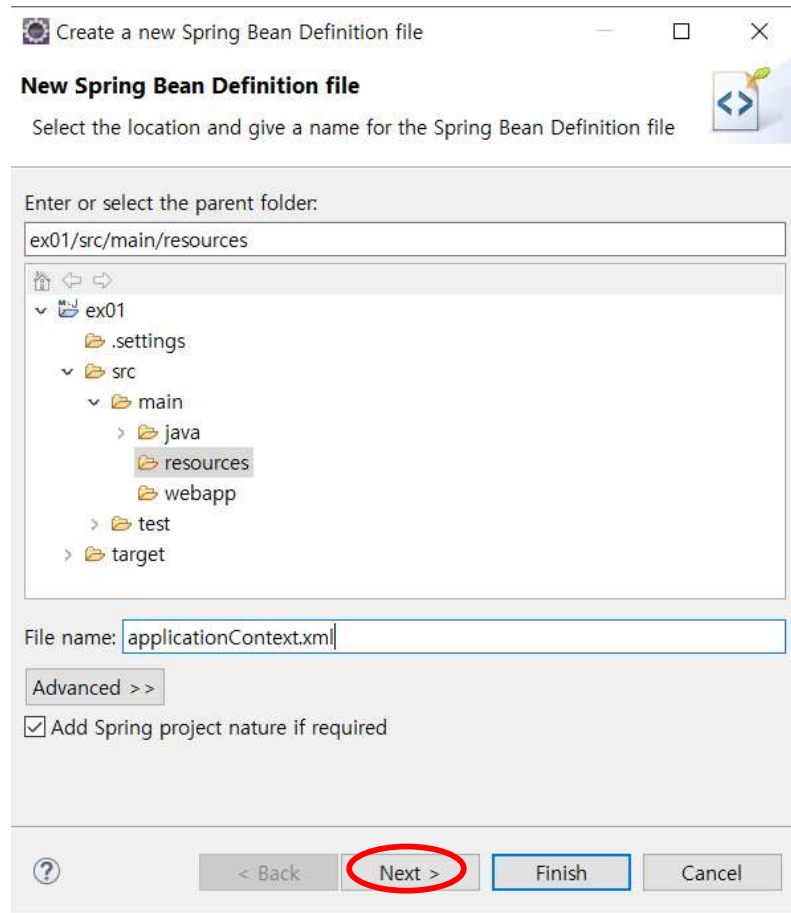
스프링 프로젝트 시작하기

- 기본 자바 프로그램과 다르게 스프링은 스프링 설정파일에서 객체를 만든 다음 만들어진 객체를 불러오는 방법으로 기능을 동작시킨다.
- 스프링 설정 파일을 만들어 보자 –
 - Resources위치에 우클릭후
new -> spring bean configuration 을 선택



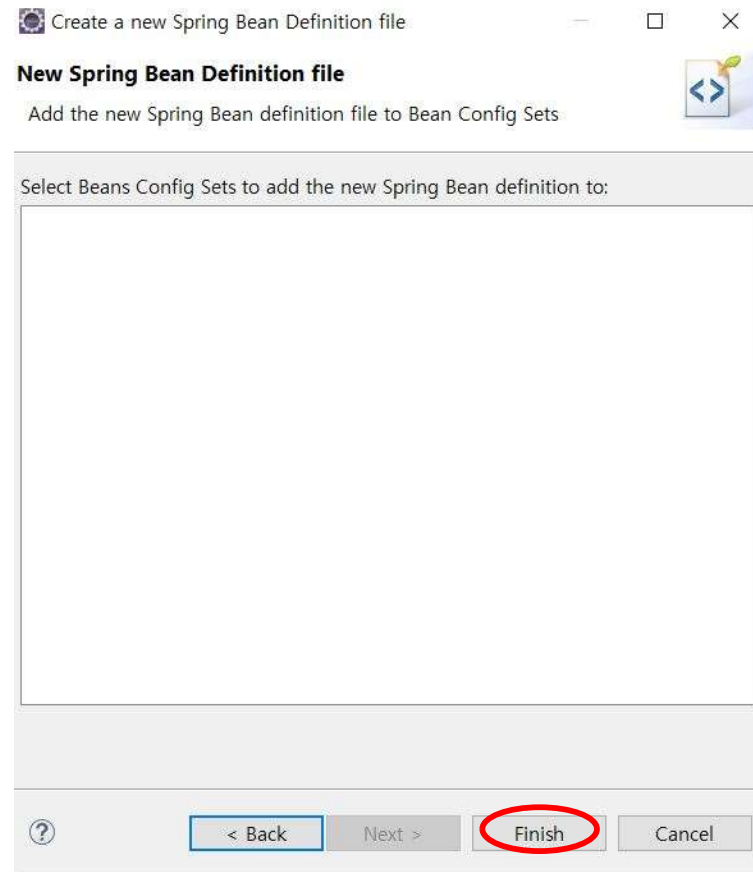
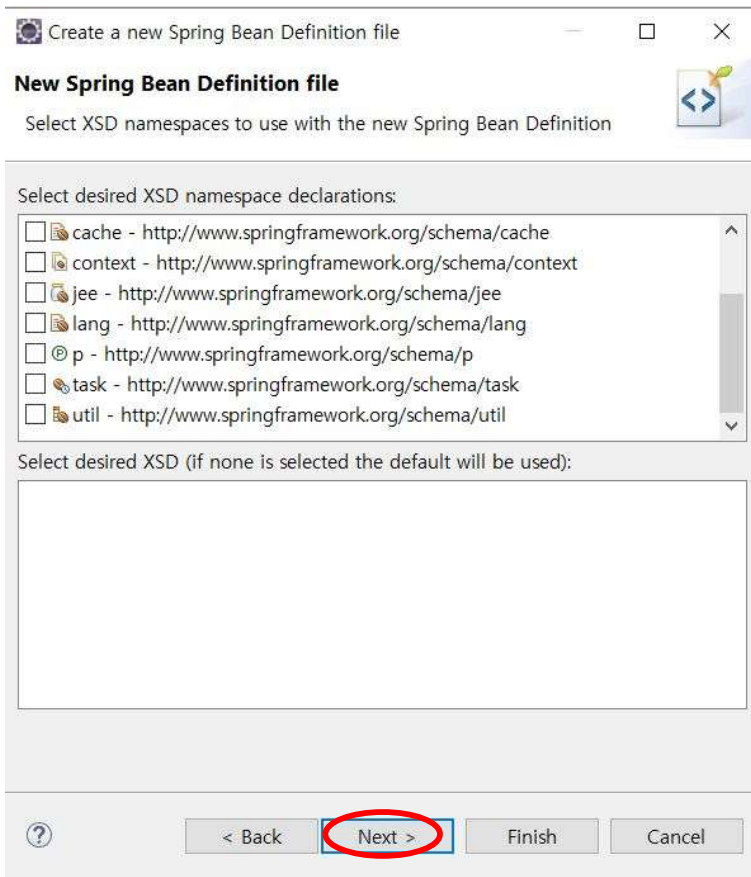
스프링 프로젝트 시작하기

- 설정 파일명은 applicationContext.xml 로 지정한다. => Next



스프링 프로젝트 시작하기

- 현재는 아무런 기능을 추가할 예정이 없으므로 체크 없이 Next, 그리고 Finish로 파일을 생성한다.



스프링 프로젝트 시작하기

- applicationContext.xml파일내부에 객체를 등록한다
 - id : 빈 객체를 구분할 때 사용할 이름
 - class : 빈 객체를 생성할 때 사용할 클래스
- property태그는 set메서드를 호출한다.
 - name 속성은 set메서드가 호출할 클래스의 필드 이름이다.
 - value 속성은 해당 필드에 넣을 값이다.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xsi:schemaLocation="http://www.springframework.org/schema
5
6       <bean id="greeter" class="ex01.Greeter">
7           <property name="format" value="%s, 안녕하세요!" />
8       </bean>
9
10 </beans>
```

스프링은 객체 컨테이너

- 스프링의 핵심기능은 객체를 생성하고 초기화 하는 것이다.
- 이와 관련된 기능은 ApplicationContext라는 인터페이스에 정의 되어있으며 GenericXmlApplicationContext클래스는 이 인터페이스를 구현한 객체이다.
- 위 클래스를 이용해서 스프링 설정에 등록된 빈 객체를 불러오는 Main02 클래스를 생성해보자

```
public class Main02 {  
  
    public static void main(String[] args) {  
        GenericXmlApplicationContext ctx =  
            new GenericXmlApplicationContext("classpath:applicationContext.xml");  
        Greeter gt = ctx.getBean("greeter", Greeter.class);  
  
        String msg = gt.greet("여러분");  
  
        System.out.println(msg);  
    }  
}
```

스프링은 객체 컨테이너

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

⇒ 설정 정보를 이용해서 빈 객체를 생성한다.

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

=> 자바빈 객체를 제공한다.

스프링은 객체 컨테이너

- 스프링은 별도의 설정이 없는 경우 한 개의 빈 객체만 생성한다 => 싱글톤

- Main03

```
import org.springframework.context.support.GenericXmlApplicationContext;

public class Main03 {

    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();
    }
}
```

(g1 == g2) = true