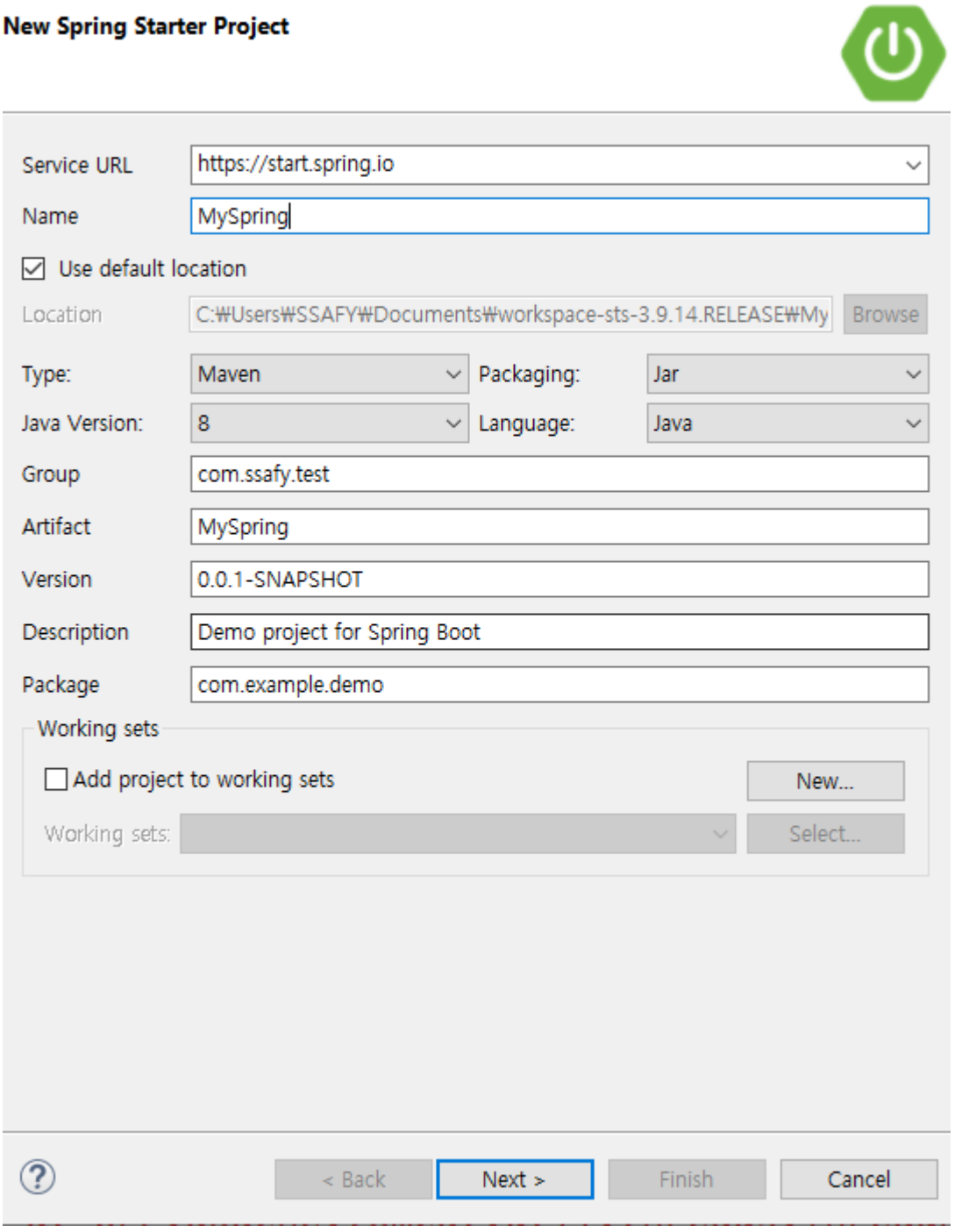
Spring Boot

Spring Boot란?

> Spring 프레임워크를 보다 쉽게 사용할 수 있도록 간소화한 프레임워크

- 기본적인 설정을 미리 제공한다
  - 기본적인 Spring Bean을 미리 등록해두고있음
  - web.xml, root-context.xml, servlet-context.xml이 필요없다
  - application.properties 하나로 간소화된 설정을 등록가능
  - `@Configuration` , `@SpringBootConfiguration`이 적용된 클래스를 정의해서 세밀한 설정도 가능
- Spring-boot 버전에 최적화된 라이브러리(의존성)를 이미 등록하여 제공하고 있다
- tomcat을 내장서버로 가지고 있다

Spring Boot 프로젝트 시작하기



- **Version : 2.7.17**
- **Dependencies**
  - **Spring Boot Devltools**
  - **Spring Web**
  - **MySQL Driver**
  - **MyBatis Framework**

각종 설정들

- resources/spring
  - application 실행시 기본적인 설정은 자동으로 진행됨
  - application.properties에서 추가 설정을 간편하게 등록
- View Resolver
  - servlet-context.xml

```
<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>
```

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

- JDBC Driver
  - root-context.xml

```
<bean id="dataSource"
    class="org.apache.commons.dbcp2.BasicDataSource">
    <property name="driverClassName"
      value="com.mysql.cj.jdbc.Driver"></property>
    <property name="url"
      value="jdbc:mysql://localhost:3306/ssafydb2?serverTimezone=UTC"></property>
    <property name="username" value="ssafy"></property>
    <property name="password" value="ssafy"></property>
  </bean>
```

```
#datasource
spring.datasource.url=jdbc:mysql://localhost:3306/ssafydb?serverTimezone=UTC
spring.datasource.username=ssafy
spring.datasource.password=ssafy
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.type=org.apache.commons.dbcp2.BasicDataSource
```

- MyBatis Mapper 등록
  - root-context.xml

```
<bean id="sqlSessionFactory"
    class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource" />
    <!-- mapper xml 파일의 경로를 ant 표현식의 형태로 사용한다. -->
    <property name="mapperLocations"
      value="classpath*:mappers/**/*.xml" />
    <!-- mapper에서 사용할 DTO들의 기본 패키지를 등록한다. -->
    <property name="typeAliasesPackage" value="com.ssafy.ws.model.dto"></property>
  </bean>
```

```
# mybatis
mybatis.mapper-locations=mappers/**/*.xml
mybatis.type-aliases-package=com.ssafy.hw.model.dto
```

Java 설정방법(알아만 두자)

- @Configuration Class는 Spring Boot Application이 실행될 때 Component 스캔의 대상이 되며,

@Bean 메소드가 return 하는 객체가 Spring Bean으로 등록됨.

```
@Configuration
public class (ConfigClass) {
  @Bean
  public (ConfigBean) myConfigBean() {
    (ConfigBean) myBean = new ...;
    return myBean;
  }
}
```

Main Application Class

- 항상 프로젝트의 최상단에 위치하는(가장 먼저 호출되는) 클래스
- `@SpringBootApplication` 이 붙어있으며, 여기서부터 설정을 읽어나가기 시작한다.
  - Spring Boot관련 기본 설정을 해주는 어노테이션

    - Spring Bean 생성과 주입이 진행됨
    - Component Scan 등을 포함한다
- main 메소드가 존재하며, 어플리케이션이 실행된다

나머지는 Spring Legacy와 동일

- Controller, Dao, Service, Mapper 등등…
- 실습문제 챕터 7을 참고해서 Spring Legacy 프로젝트를 Spring-Boot 프로젝트로 바꿔보자