

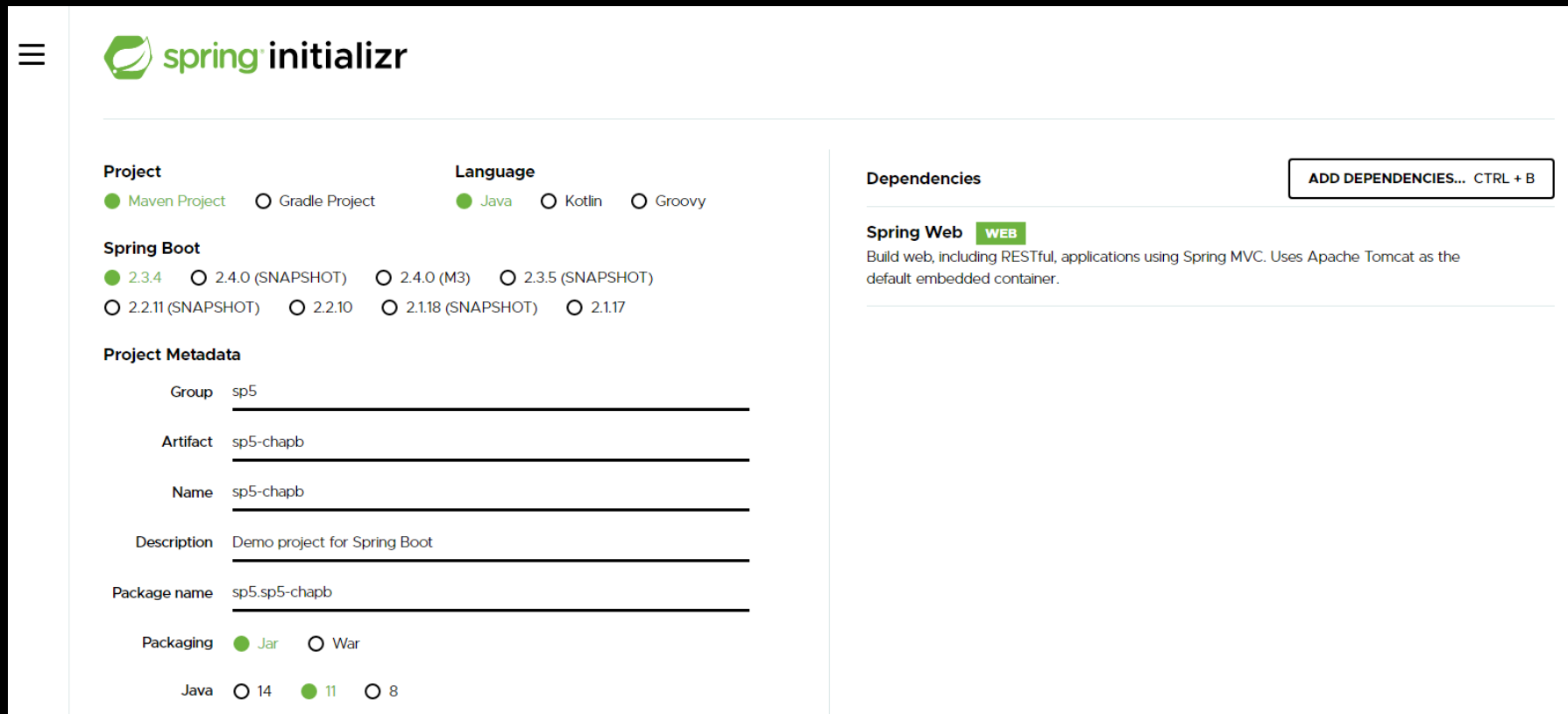
스프링 부트

스프링 부트를 왜 사용할까?

- 스프링 부트는 스프링을 빨리, 쉽게 사용할 수 있도록 대부분의 설정을 미리 세팅해 놓는다.
- Tomcat, Jetty, Undertow 내장형 서버
- 기본설정이 되어있는 starter 컴포넌트를 제공
- 통계, 상태 점검, 외부 설정 등 제공

스프링 부트 시작하기

- 스프링 이니셜라이즈 사이트(<https://start.spring.io/>)에 접속



The image shows the Spring Initializr web form. It is a white form with a green header bar containing the Spring logo and the text 'spring initializr'. The form is divided into several sections: 'Project' with radio buttons for 'Maven Project' (selected) and 'Gradle Project'; 'Language' with radio buttons for 'Java' (selected), 'Kotlin', and 'Groovy'; 'Spring Boot' with radio buttons for versions 2.3.4 (selected), 2.4.0 (SNAPSHOT), 2.4.0 (M3), 2.3.5 (SNAPSHOT), 2.2.11 (SNAPSHOT), 2.2.10, 2.1.18 (SNAPSHOT), and 2.1.17; 'Project Metadata' with text input fields for 'Group' (sp5), 'Artifact' (sp5-chapb), 'Name' (sp5-chapb), 'Description' (Demo project for Spring Boot), and 'Package name' (sp5.sp5-chapb); 'Packaging' with radio buttons for 'Jar' (selected) and 'War'; and 'Java' with radio buttons for versions 14, 11 (selected), and 8. On the right side, there is a 'Dependencies' section with a button 'ADD DEPENDENCIES... CTRL + B' and a 'Spring Web' dependency selected, indicated by a green 'WEB' tag. The description for 'Spring Web' is 'Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.'

spring initializr

Project

☒ Maven Project ☐ Gradle Project

Language

☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

☒ 2.3.4 ☐ 2.4.0 (SNAPSHOT) ☐ 2.4.0 (M3) ☐ 2.3.5 (SNAPSHOT)

☐ 2.2.11 (SNAPSHOT) ☐ 2.2.10 ☐ 2.1.18 (SNAPSHOT) ☐ 2.1.17

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging

☒ Jar ☐ War

Java

☐ 14 ☒ 11 ☐ 8

Dependencies ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

스프링 부트 시작하기

- pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://maven.apache.org/xsi:schemaLocation"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/maven-v4_0_0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>org.springframework.boot</groupId>
7         <artifactId>spring-boot-starter-parent</artifactId>
8         <version>2.3.4.RELEASE</version>
9         <relativePath/> <!-- lookup parent from repository -->
```

starter 모듈이

- 의존 설정 추가
- 기본 설정 추가

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

starter-web 모듈은
spring-webmvc, JSON,
Validator, tomcat 등 웹 개발에
필요한 의존을 설정

starter

- spring-boot-starter-parent
- 스프링 부트에 필요한 의존을 자동으로 추가한다.
- spring-boot-starter-web
- Spring mvc를 사용한 RESTful 서비스를 개발하는데 사용한다.
- Spring-boot-starter-test
- Junit, Hamcrest, Mockito로 어플리케이션 테스트를 가능하게 한다.

@SpringBootApplication

- @EnableAutoConfiguration, @ComponentScan, @Configuration을 하나로 묶어 놓은 것으로 볼 수 있다.
- 스프링 부트를 가동하기 위해 main 메소드가 필요하다. 해당 클래스에 @SpringBootApplication 어노테이션을 붙인다.

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

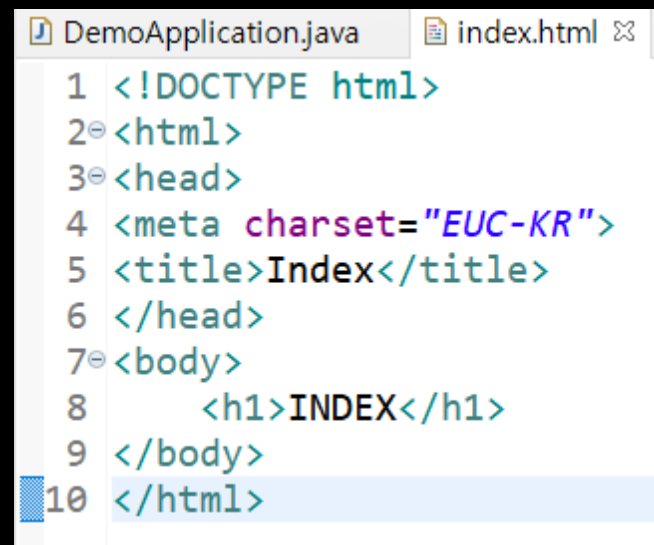
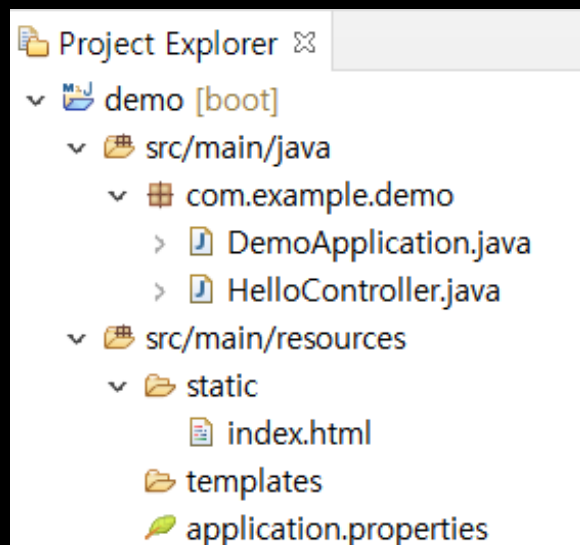
@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

페이지 생성하기

- resources > static에 index.html 파일을 생성한다.
- index.html은 기본 URL로 접속 시 보이는 화면이다.
- static 폴더는 사이트의 정적인 파일을 관리할 때 사용한다. 주로 HTML 문서, 이미지, 영상 등이 담긴다.



컨트롤러 만들기

- HelloController.java
- /hello 경로로 오는 요청에 “Hello, Spring Boot!” 응답을 보낸다.
- @ResponseBody 어노테이션을 이용해 문자열을 응답의 Body로 보낸다.

```
DemoApplication.java  HelloController.java ✖
1 package com.example.demo;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.ResponseBody;
6
7 @Controller
8 public class HelloController {
9
10     @RequestMapping("/hello")
11     public @ResponseBody String hello() {
12         return "Hello, Spring Boot!";
13     }
14 }
```