


서버세팅

1) spring boot 세팅

The screenshot shows a Google search interface with the query 'spring.io'. The search results display the Spring framework's official website. The page title is 'Spring | Home' with the URL 'https://spring.io'. Below the title, there is a tagline: 'Level up your Java code and explore what **Spring** can do for you.' A search bar on the page contains the text 'spring.io 검색결과'. The main content area lists several links: 'Initializr', 'Guides', 'Projects', 'Tools', and 'Spring Boot Project', each followed by a brief description of the resource.

spring.io

전체 이미지 동영상 쇼핑 뉴스 : 더보기 도구

 Spring
https://spring.io

Spring | Home

Level up your Java code and explore what **Spring** can do for you.

spring.io 검색결과

Initializr
Initializr generates spring boot project with just what you need ...

Guides
Consuming a RESTful Web Service. Learn how to retrieve ...

Projects
Spring Web Services. Facilitates the development of contract-first ...

Tools
Spring Tools 4 is the next generation of Spring tooling for ...

Spring Boot Project
Create stand-alone Spring applications · Embed Tomcat ...

spring.io를 검색하면 다음과 같이 spring 공식 사이트가 상단에 노출된다. 기존 spring프레임워크와 달리 spring boot는 spring initializr라는 툴을 통해 아주 손쉽게 프로젝트를 생성할 수 있다.



Project

☒ Gradle - Groovy ☐ Gradle - Kotlin ☐ Maven

Language

☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

☐ 3.4.2 (SNAPSHOT) ☐ 3.4.1 ☐ 3.3.8 (SNAPSHOT) ☒ 3.3.7

Project Metadata

Group

com.highgarden

Artifact

springboot-board

Name

springboot-board

Description

Demo project for Spring Boot

Package name

com.highgarden.springboot-board

Packaging

☒ Jar ☐ War

Java

☐ 23 ☐ 21 ☒ 17

Dependencies

Spring Web WEB

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

Lombok DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

Spring Boot DevTools DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

ADD DEPENDENCIES... CTRL + B

프로젝트 기본 세팅과 의존성은 다음과 같이 설정했다.

-빌드방식: **Gradle-Groovy**

-언어: **Java**

-버전: **Spring Boot 3.3.7**

-패키징: **Jar**

-Java버전: **17**

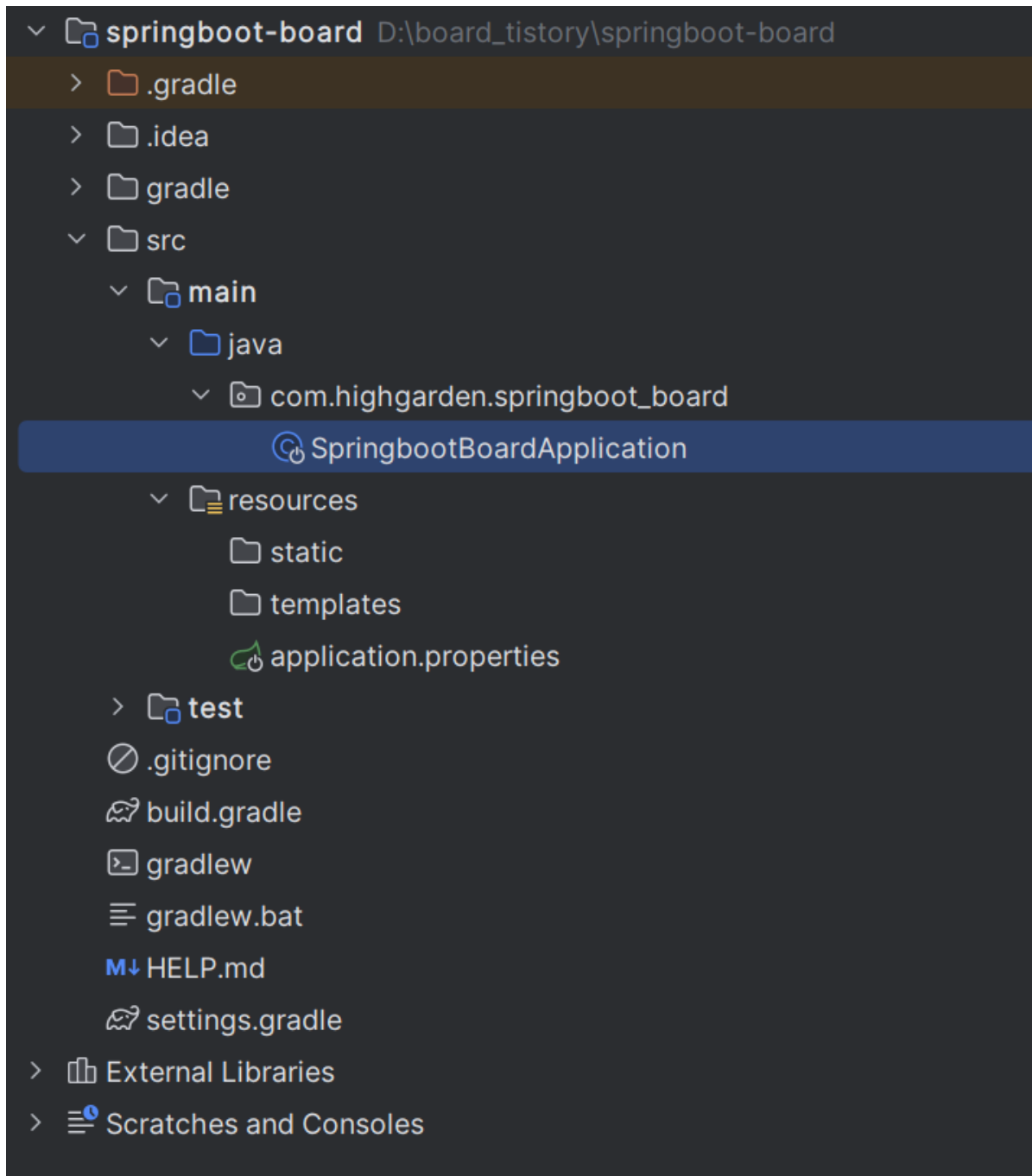
-프로젝트 메타데이터: **com.highgarden.springboot-board**

* 여기서 설정하는대로 프로젝트 디렉토리명이 결정된다.

-의존성: Spring Web, Lombok, Spring Boot DevTools

* 일단 **spring**서버를 띄우기 위한 최소한의 의존성만 추가해 줬다. DB연동이나 템플릿 엔진 관련 의존성은 추후에 직접 추가해 보겠다.

여기까지 마무리되면 **GENERATE** 버튼을 누른다. 그러면 압축파일이 하나 다운로드 되는데 프로젝트를 관리할 적당한 경로에 압축을 풀어준다.



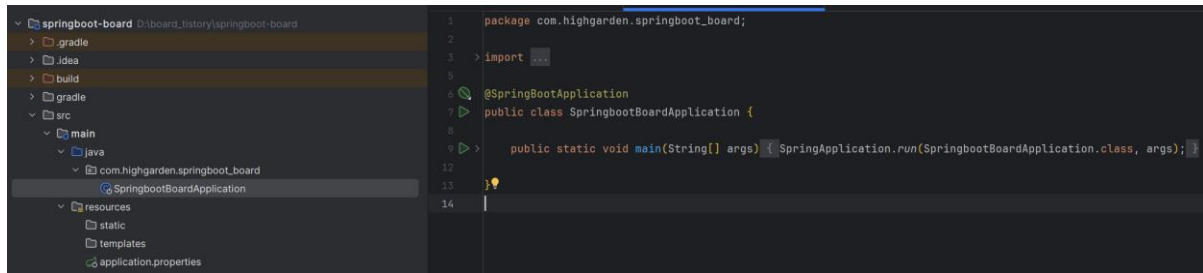
D드라이브에 board_tistory라는 폴더를 하나 만들고 그 안에 프로젝트폴더 압축을 풀어준다. IDE를 사용해 프로젝트를 실행하면 다음과 같은 프로젝트 경로를 확인할 수 있다. 나는 IntelliJ ultimate버전을 사용했다.

```

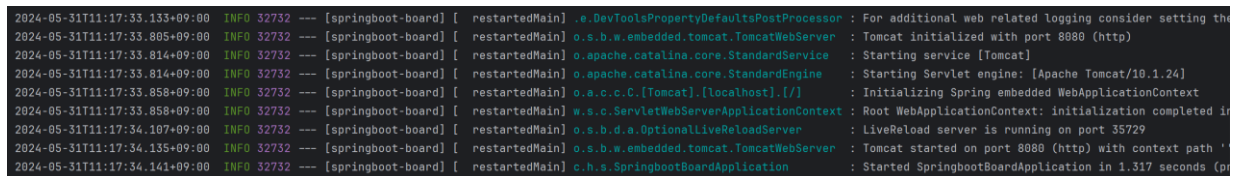
1  ▾ plugins {
2      id 'java'
3      id 'org.springframework.boot' version '3.2.6'
4      id 'io.spring.dependency-management' version '1.1.5'
5  }
6
7  group = 'com.highgarden'
8  version = '0.0.1-SNAPSHOT'
9
10 ▾ java {
11     sourceCompatibility = '17'
12 }
13
14 ▾ configurations {
15     ▾ compileOnly {
16         extendsFrom annotationProcessor
17     }
18 }
19
20 ▾ repositories {
21     mavenCentral()
22 }
23
24 ▾ dependencies { Edit Starters...
25     implementation 'org.springframework.boot:spring-boot-starter-web'
26     compileOnly 'org.projectlombok:lombok'
27     developmentOnly 'org.springframework.boot:spring-boot-devtools'
28     annotationProcessor 'org.projectlombok:lombok'
29     testImplementation 'org.springframework.boot:spring-boot-starter-test'
30     testRuntimeOnly 'org.junit.platform:junit-platform-launcher'
31 }
32
33 ▾ tasks.named('test') { Task it ->
34     useJUnitPlatform()
35 }

```

build.gradle파일을 눌러서 확인해보면 의존성을 확인할 수 있다. 프로젝트를 생성할 때 넣어준 의존성 목록들이 잘 있는지 확인해 보자. spring boot web과 devtools, lombok이 잘 들어와 있다.



이제 프로젝트를 실행해서 서버가 잘 작동하는지 확인해 볼 차례다. spring initializr에서 메타데이터로 설정했던 프로젝트 경로 안에 "프로젝트이름+application"라는 이름의 클래스가 하나 있다. 스프링프로젝트를 실행하는 메인메서드라고 생각하시면 된다. 이 메서드를 실행하면 서버가 구동된다.



로그를 확인해 보니 서버가 잘 작동한다. spring boot는 기본적으로 tomcat이 내장되어 있어 따로 tomcat을 다운로드하고 버전을 맞추는 등의 복잡한 작업을 할 필요가 없다. 로그를 확인해 보면 tomcat이 시작되었고 8080 포트라는 것을 알 수 있다. 기본 디폴트 포트는 8080번인데 나중에 설정을 통해 변경 가능하다.



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri May 31 11:18:31 KST 2024

There was an unexpected error (type=Not Found, status=404).

No static resource .

org.springframework.web.servlet.resource.NoResourceFoundException: No static resource .

실제로 브라우저에서 서버에 접근이 가능한지도 시도해 보자. 현재 도메인이 없기 때문에 서버가 실행 중인 로컬 ip를 통해 접근한다. tomcat 포트번호가 8080이기 때문에 localhost:8080으로 브

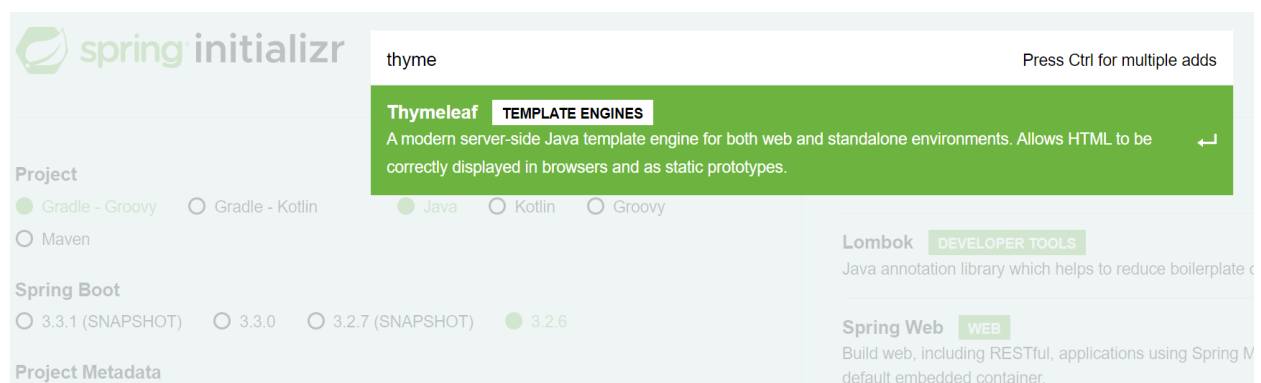
라우저에 검색해 보면 위와 같은 메시지가 나올 것이다. 여기까지 잘 된다면 서버구동 성공이다.

에러메시지를 잘 읽어보면 status=404라고 되어있는데 이는 서버에 요청이 잘 들어갔으나 서버에서 해당하는 페이지를 찾지 못함을 의미한다. 즉, 서버까지 연결은 잘 되었다는 뜻이다.

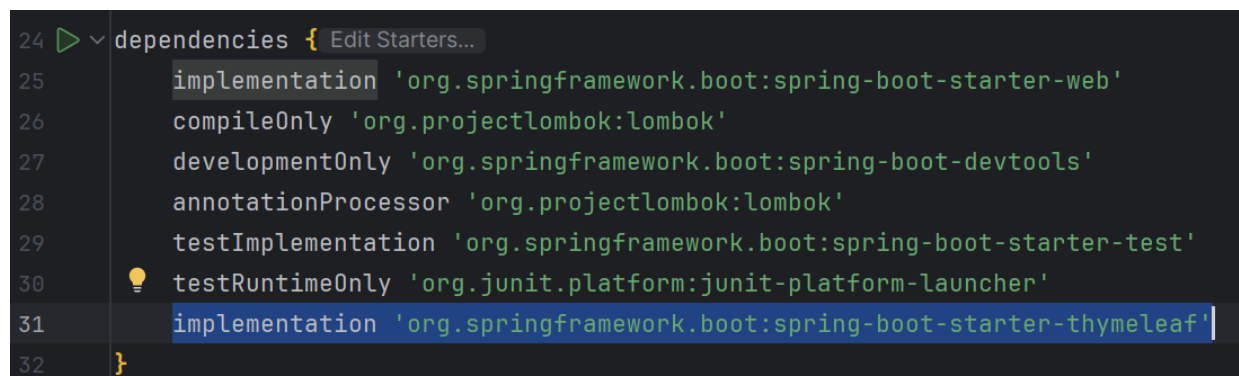
페이지를 찾지 못하는 이유는 아직 index페이지를 생성해 매핑시켜주지 않았기 때문이다.

2) index 페이지 생성

화면구성을 앞서 thymeleaf로 하기로 정한 바 있다. 이를 위해 의존성을 추가해주어야 한다. 의존성 추가는 프로젝트 생성 시 spring initializr에서 해도 되고 아니면 따로 build.gradle 파일에 직접 해주어도 된다.



spring initializr에서 의존성 추가하기



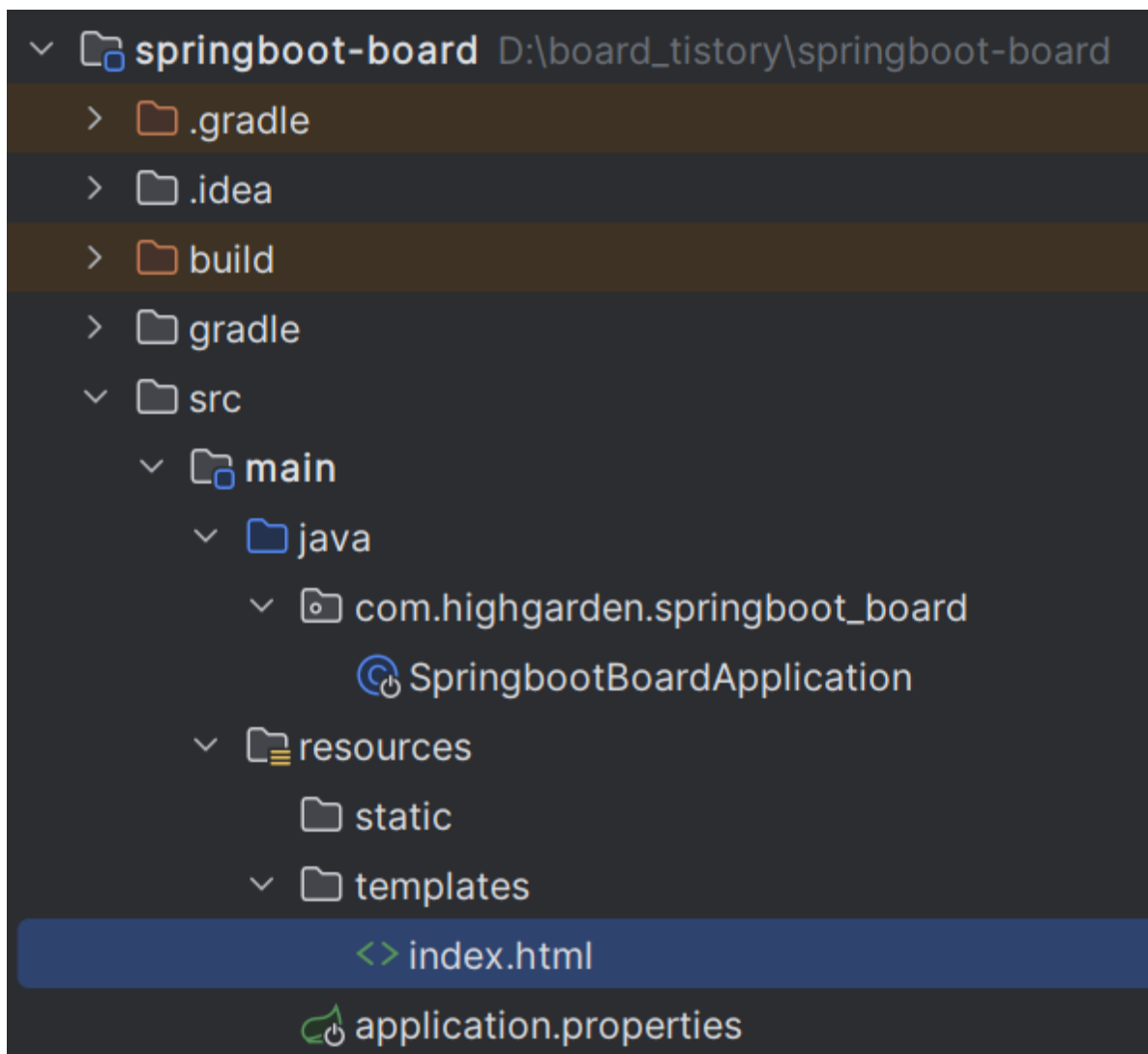
build.gradle파일에 직접 추가하기

직접 치게 되면 오타가 많이 나니 복사 붙여 넣기 하도록 하자.

```
implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
```

이제 템플릿 엔진을 사용할 수 있으니 처음 웹페이지에 들어왔을 때 보일 index 페이지를 만들어 보도록 하자.

일단 src/main/resources/template 경로에 index.html이라는 html 파일을 하나 생성한다.



타임리프 의존성을 추가해 주게 되면 templates 폴더 안에 있는 정적파일을 스프링이 찾을 수 있게 된다.

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <title>Title</title>
```

```
</head>
```

```
<body>
```

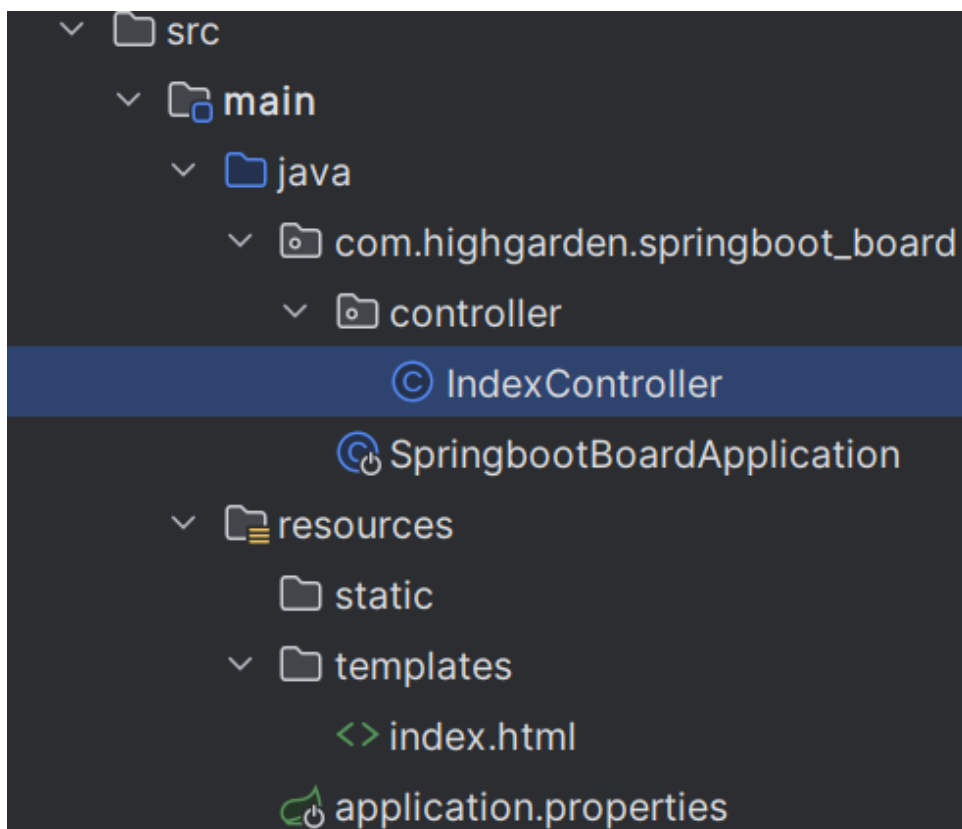
```
    <h2>hello high garden!!!</h2>
```

```
</body>
```

```
</html>
```

기본 html코드를 생성하고 페이지를 확인하기 위한 <h2> 태그를 넣어줬다.

이제 서버로의 요청을 index.html로 연결해 줄 Controller를 생성해야 한다.



위와 같은 경로에 IndexController 클래스를 생성한다.

```
@Controller
```

```
@Slf4j
```

```
public class IndexController {
```

```
    @GetMapping("/")
```

```
    public String index(){
```

```
        log.info("index메서드 call");
```

```
        return "index";
```

```
    }
```

```
}
```

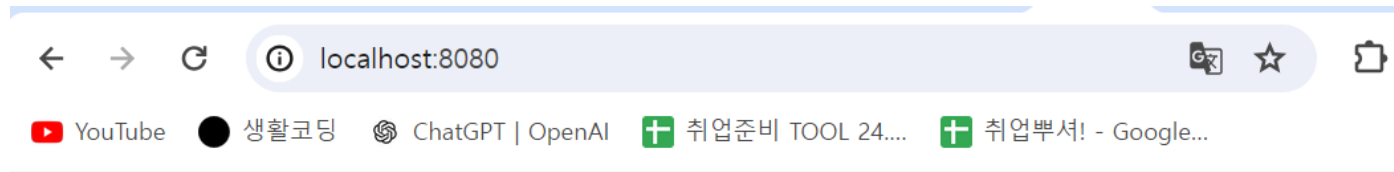
@Controller 어노테이션을 붙이게 되면 스프링이 해당 클래스를 빈에 등록해 두고 클라이언트 요청에 따라 올바른 컨트롤러로 매핑시켜 줄 수 있게 된다.

@Slf4j는 로깅을 손쉽게 사용할 수 있게 해주는 어노테이션으로 메서드가 호출되는지 확인하기 위해 사용했다.

"/" 루트요청을 index.html로 연결해 주기 위해 코드를 작성했다. localhost:8080으로 들어오는 Get 요청을 index.html 페이지로 연결해 준다. return에 "index"라는 String만 넣어도 index.html로 연결될 수 있는 이유는 앞서 잠깐 설명한 것처럼 스프링이 templates라는 폴더에서 자동으로 정적 리소스들을 검색하기 때문이다. 때문에 지정된 경로가 아니라면 스프링이 찾을 수 없다.

***타임리프 의존성이 추가되지 않은 상태라면 스프링은 templates폴더를 탐색하지 않는다. 때문에 404 에러가 발생할 수 있다. 404가 발생한다면 의존성을 확인해 보자.**

이제 다시 localhost:8080으로 요청을 해보자.



hello high garden!!!

컨트롤러와 화면이 잘 연동이 되었다면 index.html에 작성한 글씨가 위처럼 보일 것이다.

```
2024-05-31T12:11:40.934+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialize
2024-05-31T12:11:40.942+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.apache.catalina.core.StandardService : Starting service
2024-05-31T12:11:40.942+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet
2024-05-31T12:11:40.991+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Sprin
2024-05-31T12:11:40.991+09:00 INFO 12880 --- [springboot-board] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicati
2024-05-31T12:11:41.060+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.s.b.a.w.s.WelcomePageHandlerMapping : Adding welcome pa
2024-05-31T12:11:41.246+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server
2024-05-31T12:11:41.268+09:00 INFO 12880 --- [springboot-board] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on
2024-05-31T12:11:41.276+09:00 INFO 12880 --- [springboot-board] [ restartedMain] c.h.s.SpringbootBoardApplication : Started Springboot
2024-05-31T12:11:45.443+09:00 INFO 12880 --- [springboot-board] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Sprin
2024-05-31T12:11:45.444+09:00 INFO 12880 --- [springboot-board] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Serv
2024-05-31T12:11:45.444+09:00 INFO 12880 --- [springboot-board] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initial
2024-05-31T12:11:45.465+09:00 INFO 12880 --- [springboot-board] [nio-8080-exec-1] c.h.s.controller.IndexController : index메서드 call
```

로그를 확인해 보니 index 메서드도 잘 실행되었음을 확인할 수 있다.

여기까지 되었으면 이제 프로젝트 세팅이 절반정도 끝났다. 스프링 부트에서 설정과 관련된 편의 기능을 많이 제공하고 있어서 여기까지는 매우 쉽게 따라 할 수 있을 것이다.

다음시간에는 mysql과 mybatis를 세팅해 보도록 하겠다.