

# 프로젝트 환경 설정

☰ Tags	<a href="#">View</a> <a href="#">라이브러리</a> <a href="#">빌드</a> <a href="#">실행</a> <a href="#">프로젝트생성</a>
📅 학습일	@Jan 17, 2021

## 1️⃣사전 설치

1. Java 11 설치
2. IntelliJ

## 2️⃣프로젝트 생성

### 프로젝트 파일 탐색

1. build.gradle ★
2. src-main

### 프로젝트 실행

추가) Gradle이 아닌, 자바를 통한 직접적인 Gradle project 실행 설정

## 3️⃣라이브러리 살펴보기

1. 스프링 부트 라이브러리
2. 테스트 라이브러리

## 4️⃣View 환경설정

### 1. Welcome Page 만들기

- 스프링 부트가 제공하는 Welcome Page 기능을 사용한다.
- thymeleat 템플릿 엔진
- thymeleat 템플릿 엔진 동작 확인

## 5️⃣빌드하고 실행하기

1. git bash 창에서 확인하기
- 추가) 윈도우에서 Git bash 터미널 사용하기

## 1️⃣사전 설치

### 1. Java 11 설치


버전 호환을 위해 정확히 Java 11을 설치하는 것이 권장된다. [Oracle](#)에 로그인 후, Java SE 11 (LTS) 버전의 Oracle JDK 파일을 다운로드 하였다.


## Java SE 11 (LTS)

Java SE 11.0.9 the latest release for the Java SE 11 Platform

- Documentation
- Installation Instructions
- Release Notes
- Oracle License
  - Binary License
  - Documentation License
- Java SE Licensing Information User Manual
  - Includes Third Party Licenses
- Certified System Configurations
- Readme

### Oracle JDK

 JDK Download

 Documentation Download

- 기존 버전

```
PS C:\Users\sdn11> java -version
java version "1.8.0_251"
Java(TM) SE Runtime Environment (build 1.8.0_251-b08)
Java HotSpot(TM) Client VM (build 25.251-b08, mixed mode)
```

- Java 11 설치 후

```
C:\Users\sdn11> java -version
java version "11.0.8" 2020-07-14 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.8+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.8+10-LTS, mixed mode)
```

- 시행 착오

Java 11 설치 진행 중, 환경변수 Path를 지우는 바람에 Windows Terminal 실행에 애를 먹었다. 결과적으로는 해결 되었다. 자세한 오류 해결과정을 여기에.

## 2. IntelliJ

이미 설치가 되어있었다! Eclipse 보다 IntelliJ가 더 편리하다고한다. 실무에서도 요즘 대세!

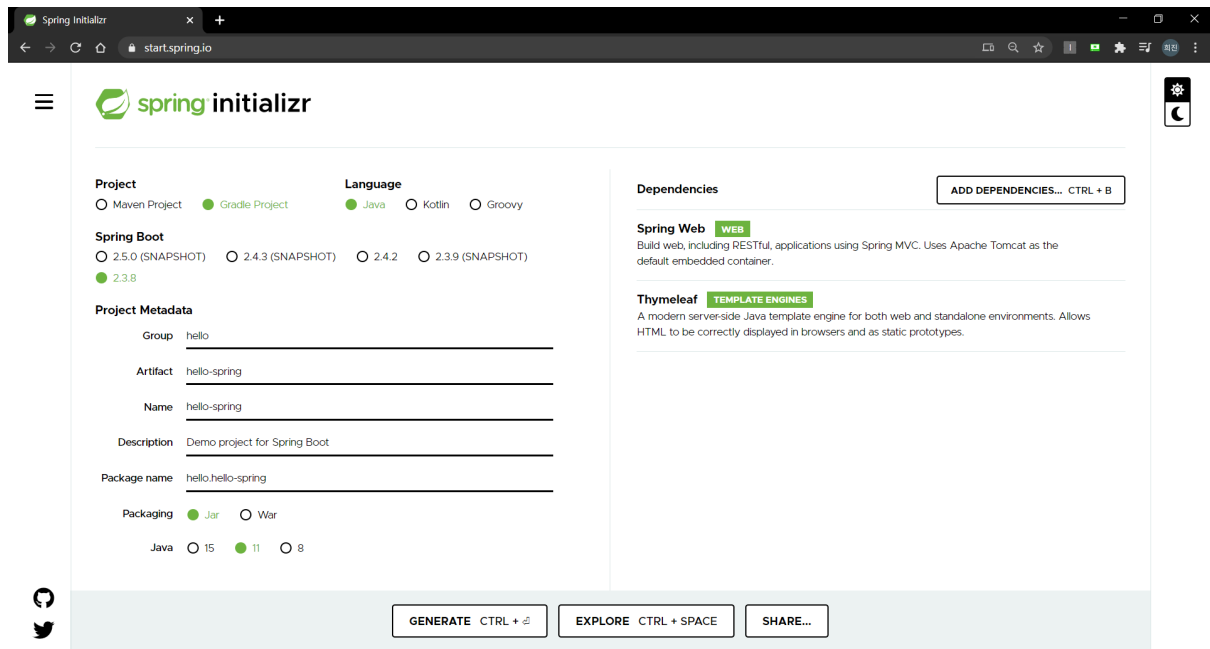
## 2. 프로젝트 생성

요즘은 모두 Spring Boot를 기준으로 프로젝트를 만든다.

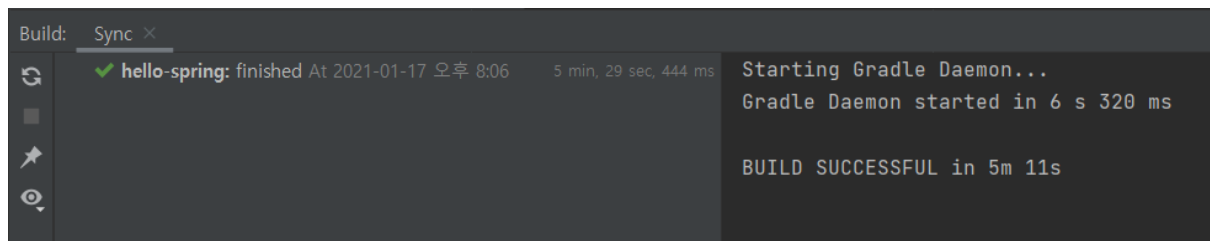
<https://start.spring.io/>

스프링 부트를 기반으로 **스프링 관련 프로젝트**를 만들어주는 사이트

- Project: Gradle Project
  - Gradle, Maven : 필요한 라이브러리를 땡겨서 오고, 빌드한 라이브러리 사이트 까지 관리해주는 사이트
  - 과거에는 Maven을 많이 썼지만, 요즘 추세는 **Gradle**로 넘어오는 추세
- Spring Boot: 2.3.x
  - SNAPSHOT : 아직 만들고 있는 버전
  - M1 : 정식 릴리즈 된 버전이 아니다.
- Project Metadata
  - Group: 보통 기업의 도메인을 주로 쓰는데, 우리는 상관 없으니 그냥 'hello'로 작성
  - Artifact: 빌드되어 나오는 결과물 'hello-spring'으로 작성
  - Name: 프로젝트 네임
- Language: Java
- Packaging: Jar
- Java: 11
- Dependencies ★
  - 어떤 라이브러리를 땡겨올 것인가?
  - Spring Web: 우리는 웹 프로젝트이니까
  - Thymeleaf: HTML을 만들어주는 템플릿 엔진



## 프로젝트 파일 탐색



Gradle Daemon을 처음 시작할 때 시간이 조금 오래 걸린다.

### 1. build.gradle ★

가장 중요한 파일. 스프링 부트를 통해 생성함으로서, 직접 치지 않아도 된다는 장점이 있다.

- group: 아까 설정한 그룹 이름
- sourceCompatibility: JAVA version
- repositories: 라이브러리를 다운로드 받을 장소. mavenCentral이라는 공개된 사이트에서 다운로드 받으라는 이야기(따로 설정 가능)
- dependencies: 아까 설정했던 dependencies + 이제는 테스트 라이브러리가 자동으로 붙는다.

- gitignore: git에는 소스코드만 올라가도록 지정(빌드된 결과물 등은 제외시켜준다)

## 2. src-main

- main : java/resources

java 파일을 제외한 모든 것들을 resources라고 생각하면 된다.

## 프로젝트 실행

## 1. [src]-[main]-[java]-[hello.hellospring]-[HelloSpringApplication] 실행

```
Task :compileJava
Task :processResources
Task :classes

Task :HelloSpringApplication.main()

      ____ _
     / ___ \_ __| |_) |__| |
    ( |___) | '_ \| | | | | |
   / ___ \| | | | | | | | | |
  /_/   \_\_| |_| |_| |_| |_|
=====|_|=====|_|/_/_/_/_/

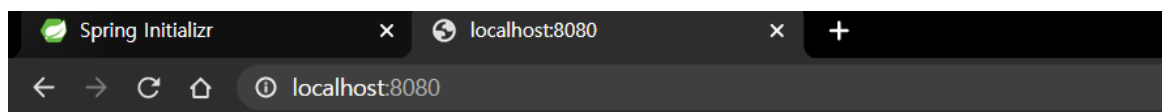
: Spring Boot ::                (v2.3.8.RELEASE)


[21-01-17 20:12:04.234 INFO 21256 --- [main] h.hellospring.HelloSpringApplication : Starting HelloSpringApplication on DESKTOP-0
[21-01-17 20:12:04.246 INFO 21256 --- [main] h.hellospring.HelloSpringApplication : No active profile set, falling back to default
[21-01-17 20:12:10.343 INFO 21256 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
[21-01-17 20:12:10.374 INFO 21256 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
[21-01-17 20:12:10.374 INFO 21256 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.
[21-01-17 20:12:10.602 INFO 21256 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
[21-01-17 20:12:10.603 INFO 21256 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization c
[21-01-17 20:12:11.010 INFO 21256 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTas
[21-01-17 20:12:13.145 INFO 21256 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with
[21-01-17 20:12:13.203 INFO 21256 --- [main] h.hellospring.HelloSpringApplication : Started HelloSpringApplication in 10.4 seconds
```

Tomcat initialized with port(s): 8080 (http)

8080 포트에 열렸다는 뜻

2. localhost: 8080 접속했을 때, 아래와 같은 에러 페이지가 뜨면 성공



# Whitelabel Error Page

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

Sun Jan 17 20:19:07 KST 2021

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

3. 왼쪽의 중지버튼을 통해 중지시킬 수 있다.

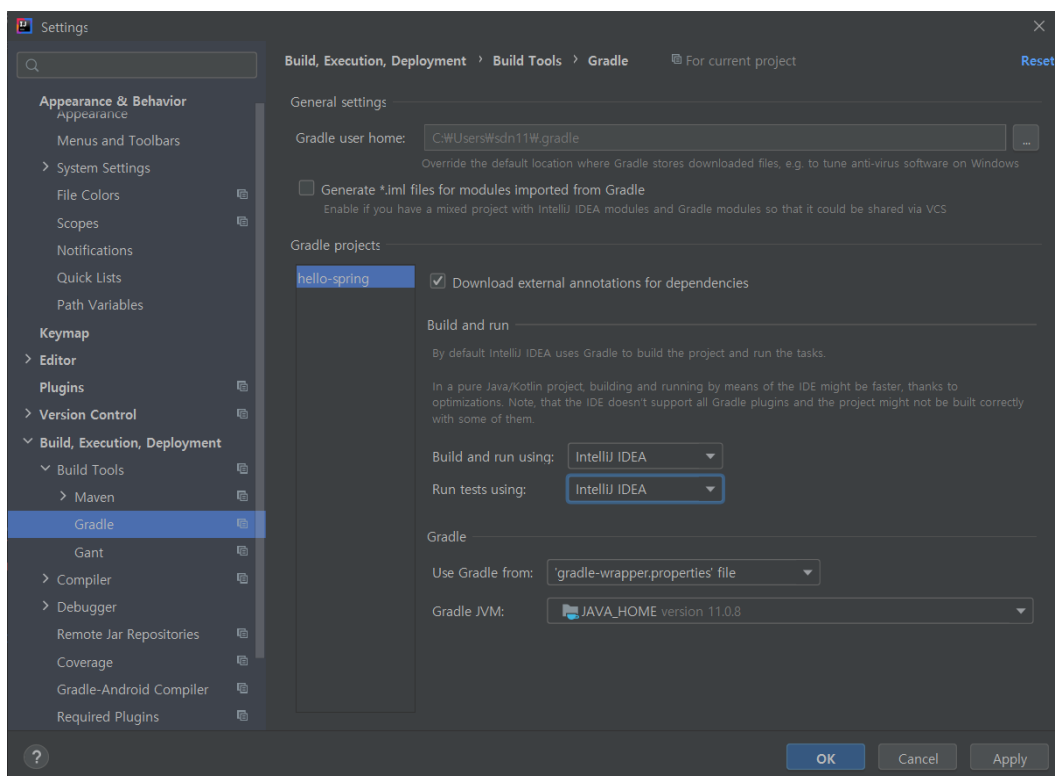
## 추가) Gradle이 아닌, 자바를 통한 직접적인 Gradle project 실행 설정

+ ) IntelliJ를 쓰면 빌드할 때, 자바를 직접 실행하는 게 아니라, 그래들을 통해 실행될 때가 있다. 빌드가 오래걸리는 원인이 될 수 있음

→ 설정값 바꾸기

[Ctrl]+[Alt]+[s] 로 Setting에 접속

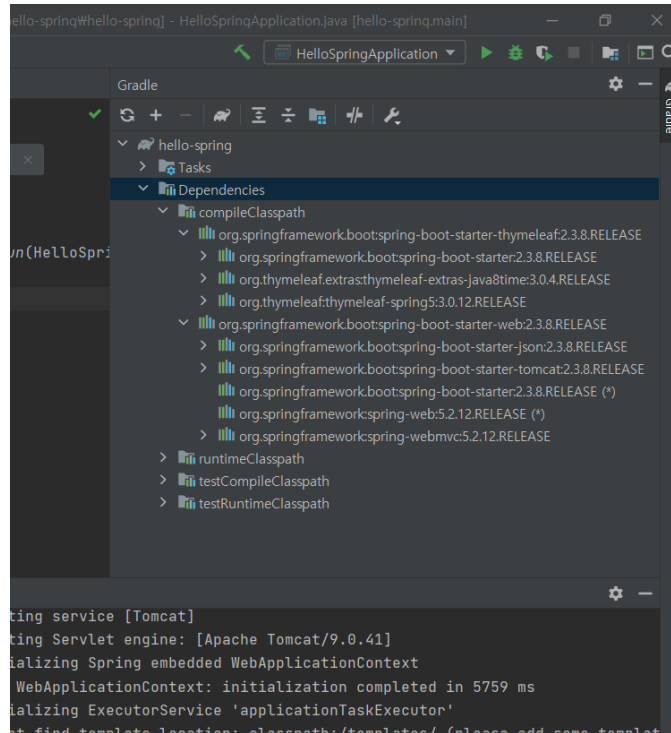
[Setting]-[Build, Execution, Deployment]-[Build Tools]-[Gradle]에서 Build and run using, Run test using 값을 Gradle이 아닌, IntelliJ로 바꾼다.



## 3 라이브러리 살펴보기

External Libraries를 보면, 선택한 dependencies를 제외하고도 무수히 많은 라이브러리를 확인할 수 있다.

→ Gradle은 **의존 관계가 있는 라이브러리**를 함께 다운로드하기 때문



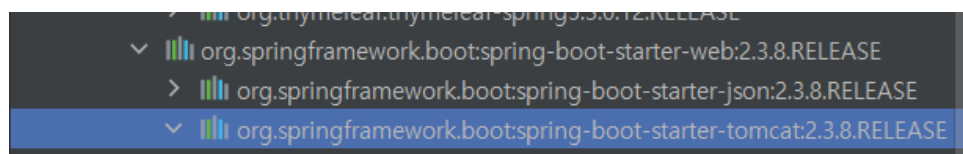
맨 우측에 [Gradle]을 선택하면,, 선택한 두 개의 dependencies와, 그와 의존관계에 있는 여러 라이브러리들을 확인할 수 있다.

## 1. 스프링 부트 라이브러리

### • spring-boot-starter-web

#### ▼ spring-boot-starter-tomcat: 톰캣 (웹서버)

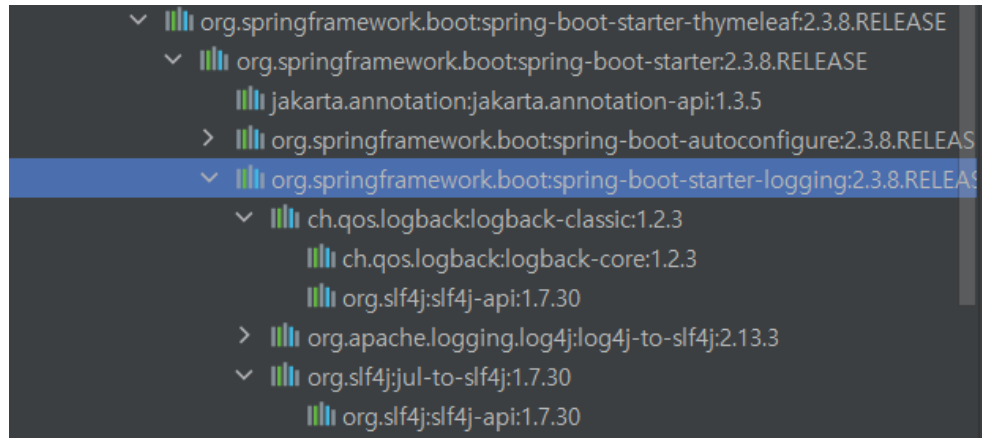
- tomcat 서버: 예전에는 tomcat 서버를 깔고 연결해주었지만, 이제는 Main method 하나만 실행해도 그냥 바로 웹이 실행된다. embedded 되었다!



- spring-webmvc: 스프링 웹 MVC
- **spring-boot-starter-thymeleaf** : 타임리프 템플릿 엔진(View)
- **spring-boot-starter**(공통) : 스프링 부트 + 스프링 코어 + 로깅
  - spring-boot
    - spring-core
  - spring-boot-starter-logging

### ▼ logback, slf4j

- logging: logback / slf4j (인터페이스) : 로그를 어떤 구현체로 출력할지에 대해서 선택할 수 있는데, 요즘은 logback을 많이 사용한다. 성능도 빠르고 지원하는 기능도 많다. (현업에서는 로그데이터를 단순히 system.out.println으로 출력해서는 안된다. 로그로 남겨야 나중에 분석이 가능!)

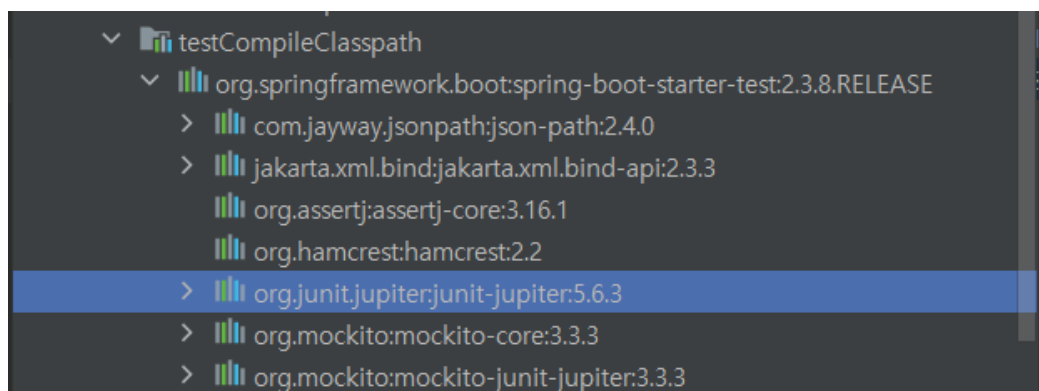


## 2. 테스트 라이브러리

### • spring-boot-starter-test

#### ▼ junit: 테스트 프레임워크

- junit: 자바 관련 프로그램 테스트를 할 때에는 junit 라이브러리를 보통 사용한다.



- mockito: 목 라이브러리(참고만)
- assertj: 테스트 코드를 좀 더 편하게 작성하게 도와주는 라이브러리(참고만)
- spring-test: 스프링 통합 테스트 지원



## 4 View 환경설정

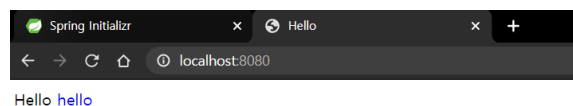
### 1. Welcome Page 만들기

- 스프링 부트가 제공하는 Welcome Page 기능을 사용한다.

- static/index.html 을 올려두면 Welcome page 기능을 제공한다.
- 그냥 파일을 static하게 던져준 것
- 관련된 기능 찾는 법
  - spring.io에 접속해서, [Projects]-[Spring Boot]-[Learn]-[Reference Doc.]에서 원하는 기능을 찾는다.
  - Welcome Page에 대한 기능 설명

index.html 파일을 추가한다.

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Hello</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
Hello
<a href="/hello">hello</a>
</body>
</html>
```



새롭게 빌드된 화면

- thymeleat 템플릿 엔진

- 파일을 그냥 던져주는 방식이 아니라, 원하는대로 바꿀 수 있다.

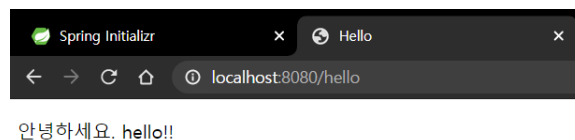
```
package hello.hellospring.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class HelloController {
    @GetMapping("hello")
    public String hello(Model model) {
        model.addAttribute("data", "hello!!");
        return "hello";
    }
}
```

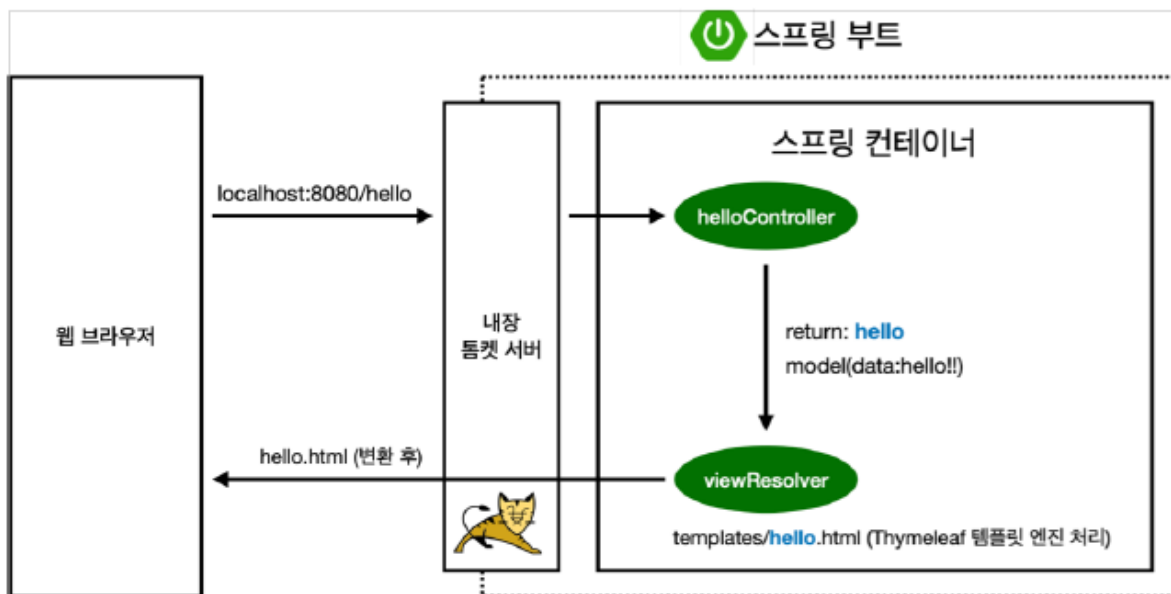
```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Hello</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<p th:text="'안녕하세요. ' + ${data}" >안녕하세요. 손님</p>
</body>
</html>
```

- <http://localhost:8080/hello> 에서 확인할 수 있다.



## - thymeleat 템플릿 엔진 동작 확인

## 동작 환경 그림



- GetMapping : GET 방식으로 웹에서 요청
- 컨트롤러에서 리턴 값으로 문자를 반환하면 뷰 리졸버( `viewResolver` )가 해당 이름의 화면을 찾아서 처리한다.
  - 스프링 부트 템플릿엔진 기본 viewName 매핑
  - `resources:templates/` + {ViewName}+ `.html`

참고: `spring-boot-devtools` 라이브러리를 추가하면, html 파일을 컴파일만 해주면 서버 재시작 없이 View 파일 변경이 가능하다.

- 인텔리J 컴파일 방법: 메뉴 build Recompile

## 5 빌드하고 실행하기

- IntelliJ를 통한 실행 말고, 서버에서 이를 빌드하고 실행해보자!
  - 과거에는 tomcat 다 설치하고, 특정 폴더에 집어 넣고.. 아주 복잡한 방법이지만, 이제는 jar 파일 실행하면 하면 된다.
- 참고) 둘 중 하나만 실행해야 한다! 그렇지 않은 경우에는 오류가 뜬다.

# 1. git bash 창에서 확인하기

1. ./gradlew build (./gradlew.bat build 명령어도 잘 먹는다.)

```
sdn11@DESKTOP-CCA07AI MINGW64 ~/OneDrive - 연세 대학교 (Yonsei University)/**회진**
/연세대/ybigta/20-겨울스터디/스프링/hello-spring/hello-spring
$ ./gradlew build

Welcome to Gradle 6.7.1!

Here are the highlights of this release:
- File system watching is ready for production use
- Declare the version of Java your build requires
- Java 15 support

For more details see https://docs.gradle.org/6.7.1/release-notes.html

> Task :test
2021-01-17 22:42:06.348 INFO 20612 --- [extShutdownHook] o.s.s.concurrent.ThreadPoolTaskExecutor : Shutting down ExecutorService 'applicationTaskExecutor'

BUILD SUCCESSFUL in 36s
5 actionable tasks: 5 executed

sdn11@DESKTOP-CCA07AI MINGW64 ~/OneDrive - 연세 대학교 (Yonsei University)/**회진**/연세대/ybigta/20-겨울스터디/스프링/hello-spring/hello-spring
$ ls
build/ build.gradle gradle/ gradlew* gradlew.bat HELP.md out/ settings.gradle src/
```

2. cd build/libs

build를 수행하면, 자동적으로 build 폴더가 생성되고, 폴더 안에 jar 파일로 존재한다.

3. java -jar hello-spring-0.0.1-SNAPSHOT.jar

```
sdn11@DESKTOP-CCA07AI MINGW64 ~/OneDrive - 연세 대학교 (Yonsei University)/학회진 ☆/연세대 /ybigta/20-겨울
스터디 /스프링 /hello-spring/hello-spring
$ cd build/libs

sdn11@DESKTOP-CCA07AI MINGW64 ~/OneDrive - 연세 대학교 (Yonsei University)/학회진 ☆/연세대 /ybigta/20-겨울
스터디 /스프링 /hello-spring/hello-spring/build/libs
$ ls -ar1th
total 18M
drwxr-xr-x 1 sdn11 197609 0 1월 17 22:41 ./
-rw-r--r-- 1 sdn11 197609 18M 1월 17 22:41 hello-spring-0.0.1-SNAPSHOT.jar
drwxr-xr-x 1 sdn11 197609 0 1월 17 22:41 ../

sdn11@DESKTOP-CCA07AI MINGW64 ~/OneDrive - 연세 대학교 (Yonsei University)/학회진 ☆/연세대 /ybigta/20-겨울
스터디 /스프링 /hello-spring/hello-spring/build/libs
$ java -jar hello-spring-0.0.1-SNAPSHOT.jar

:: Spring Boot :: (v2.3.8.RELEASE)

2021-01-17 22:43:26.115 INFO 17492 --- [main] h.hellospring.HelloSpringApplication : St
arting HelloSpringApplication on DESKTOP-CCA07AI with PID 17492 (C:\Users\sdn11\OneDrive - 연세 대학교 (
Yonsei University)/학회진 ☆/연세대 /ybigta/20-겨울 스터디 \스프링 \hello-spring\hello-spring\build\libs\hell
o-spring-0.0.1-SNAPSHOT.jar started by sdn11 in C:\Users\sdn11\OneDrive - 연세 대학교 (Yonsei University
)\학회진 ☆/연세대 /ybigta/20-겨울 스터디 \스프링 \hello-spring\hello-spring\build\libs)
2021-01-17 22:43:26.126 INFO 17492 --- [main] h.hellospring.HelloSpringApplication : No
active profile set, falling back to default profiles: default
2021-01-17 22:43:30.609 INFO 17492 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : To
mcat initialized with port(s): 8080 (http)
2021-01-17 22:43:30.644 INFO 17492 --- [main] o.apache.catalina.core.StandardService : St
arting service [Tomcat]
2021-01-17 22:43:30.645 INFO 17492 --- [main] org.apache.catalina.core.StandardEngine : St
arting Servlet engine: [Apache Tomcat/9.0.41]
2021-01-17 22:43:30.874 INFO 17492 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : In
itializing Spring embedded WebApplicationContext
2021-01-17 22:43:30.874 INFO 17492 --- [main] w.s.c.ServletWebServerApplicationContext : Ro
ot WebApplicationContext: initialization completed in 4439 ms
2021-01-17 22:43:31.540 INFO 17492 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : In
itializing ExecutorService 'applicationTaskExecutor'
2021-01-17 22:43:31.737 INFO 17492 --- [main] o.s.b.a.w.s.WelcomePageHandlerMapping : Ad
ding welcome page: class path resource [static/index.html]
2021-01-17 22:43:32.120 INFO 17492 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : To
mcat started on port(s): 8080 (http) with context path ''
2021-01-17 22:43:32.152 INFO 17492 --- [main] h.hellospring.HelloSpringApplication : St
arted HelloSpringApplication in 7.664 seconds (JVM running for 8.85)
2021-01-17 22:43:39.451 INFO 17492 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : In
itializing Spring DispatcherServlet 'dispatcherServlet'
2021-01-17 22:43:39.451 INFO 17492 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : In
itializing Servlet 'dispatcherServlet'
```

4. 만약 제대로 실행되지 않는다면, ./gradlew clean build 를 수행해보면 될 것이다.  
clean은 새로 지우고 다시 실행하도록 하는 명령어.

## 추가) 윈도우에서 Git bash 터미널 사용하기

Windows 환경에서 Terminal 사용 시, 리눅스 명령어 사용이 가능하도록 설정하기 위해,  
인텔리J의 기본 터미널을 cmd가 아닌, Git Bash로 연동하였다. 강사님이 사용하셨던 `ls`  
`-ar1th` 명령어( `-ar1th`는 모든 파일, 폴더를 시간 역순으로 출력) 도 수행 가능

[Settings]-[Terminal]-[Shell Path]

"C:\Program Files\Git\bin\sh.exe(shell이 설치된 경로)" -login -i 로 변경.