Spring Boot Part2

스프링 부트의 소개
스프링 프로젝트 생성
스프링 프로젝트의 구조
스프링 프로젝트 초기 설정

Spring Framework

스프링 프레임워크는 자바 기반 오픈 소스 프레임워크로, 자바 언어를 베이스로 한 웹 애플리케이션을 쉽고 빠르게 구축할 수 있도록 도와준다.

스프링 프레임워크의 핵심은 객체의 생성과 소멸을 관리하고, 생성된 객체에 추가적인 기능을 제공하는 스프링 컨테이너라는 컴포넌트이다.

스프링은 WAS로 톰캣을 내장하고 있다. Spring Framework를 줄여서 통상적으로 Spring이라 부른다.

Spring (Framework) vs Spring Boot

Spring Boot는 기존 Spring에서 초기설정 구성에 많은 시간을 할애한다는 단점과 초보자 기준 초기설정을 구성하는 진입장벽이 높다는 단점을 보완하여 기본적인 설정을 개발자가 직접 작성하지 않아도 간단히 프로젝트를 실행할 수 있도록 설계된 프레임워크이다.

각자의 장단점이 있기 때문에 실무에서는 어떤 회사에 입사하냐에 따라 사용하는 프레임워크가 달라진다.

프레임워크 vs 라이브러리

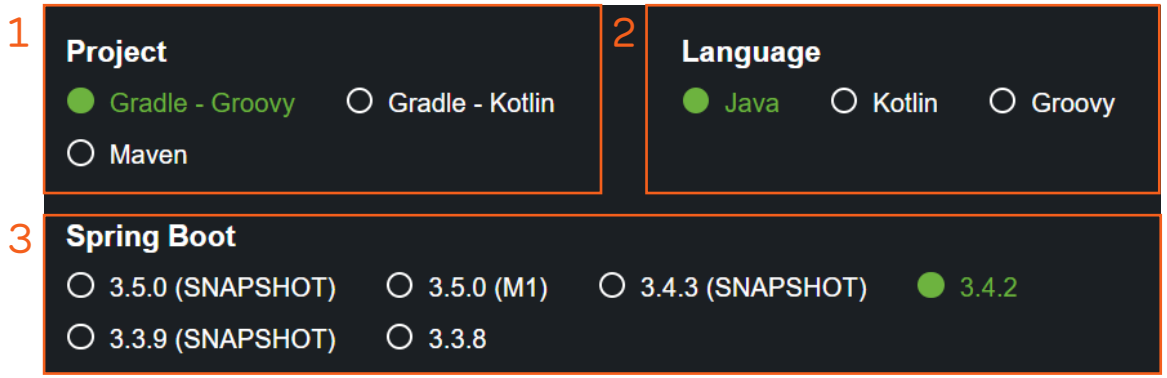
프레임워크와 라이브러리는 현재 뚜렷히 구분하여 사용하기보단 혼용으로 사용한다. 하지만 둘의 차이점은 알아두면 좋다. (종종 면접에서 질문함)

라이브러리는 다른 개발자가 만들어놓은 유용한 기능의 모음집이라 생각하면 된다. 우리는 다른 개발자가 만들어놓은 여러 기능을 우리 입맛에 맞게 사용하고, 사용에 있어 제한이 없다. 프레임워크에서 프레임은 틀이고 워크는 작업, 일 등을 의미한다. 직역하면 틀 위에서 작업(코딩)한다는 의미이다. 프레임워크는 라이브러리처럼 많은 유용한 기능을 제공한다. 여기에 추가하여 유용한 기능을 우리가 마음대로 사용하지 않고, 정해진 방식 혹은 틀 안에서 사용하길 권장하고 있다. 이러한 제한을 주는 이유는, 개발자가 마음대로 코드를 작성하는 것보다 정해진 틀 안에서 코드를 구성하는 것이 더 좋은 코딩 방식을 지킬 수 있기 때문이다. 결국, 라이브러리와 프레임워크는 공통적으로 여러 유용한 기능을 제공한다는 점이 있지만, 코드의 작성 방식을 결정하는 주체가 누구냐에 차이점이 있다.

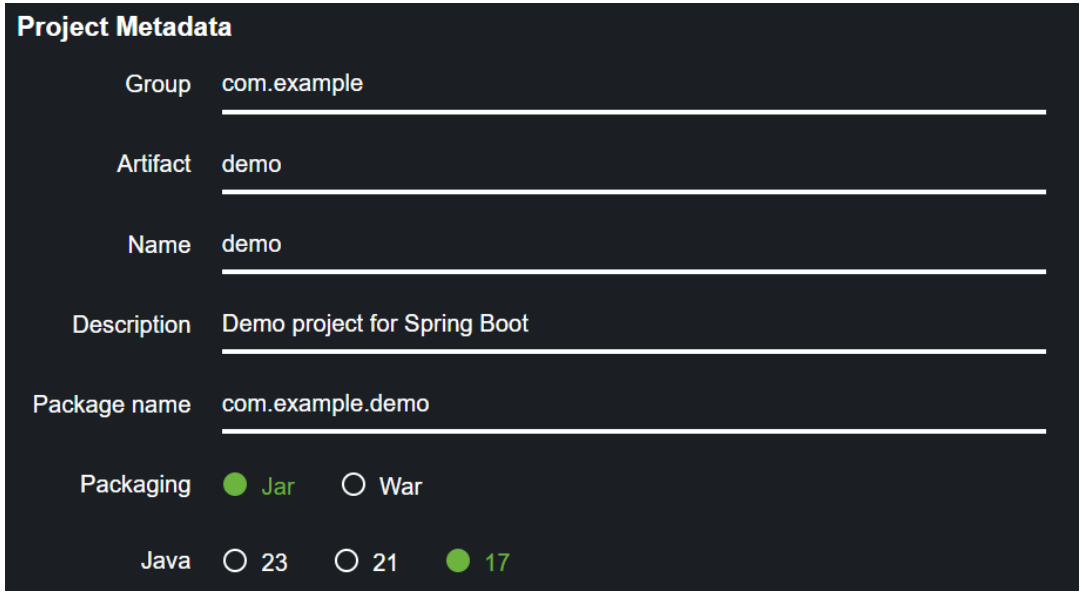
IntelliJ 무료버전에서는 Spring 프로젝트를 직접 생성할 수 없기 때문에 spring.io 싸이트에서 프로젝트를 생성 후 IntelliJ에서 열어줘야 한다.

먼저 웹 브라이저의 URL에 spring.io를 입력하여 스프링 싸이트에 접속한다.

상단 메뉴 중 Projects 메뉴에서 Spring Initializr 메뉴를 클릭하면 스프링 프로젝트를 생성하는 페이지로 이동할 수 있다.



1. 빌드 관리 도구를 Gradle(그레이들), Maven(메이븐) 중 선택하는 메뉴이다. 빌드관리도구란 여러 설정파일을 인식하고, 우리가 만든 코드를 실행할 수 있는 앱으로 만들어주며, 외부 라이브러리를 다운로드하고 관리해주는 기능을 가진 소프트웨어다. 결국 Gradle, Maven 중 무엇을 선택했냐에 따라 각종 설절파일의 형식이 달라지고, 작성한 코드를 앱으로 만들어주는 방식이 다르며, 외부 라이브러리를 다운로드하고 관리해주는 방식이 달라진다는 말이다. 요즘은 Gradle을 많이 사용하기 때문에 강의도 Gradle을 사용한다.
2. Spring 프로젝트에서 사용하는 언어를 선택하는 메뉴이다. 우리는 당연히 java를 선택한다.
3. Spring Boot 프로젝트의 버전을 선택하는 메뉴이다. 우리는 default 버전을 사용한다.



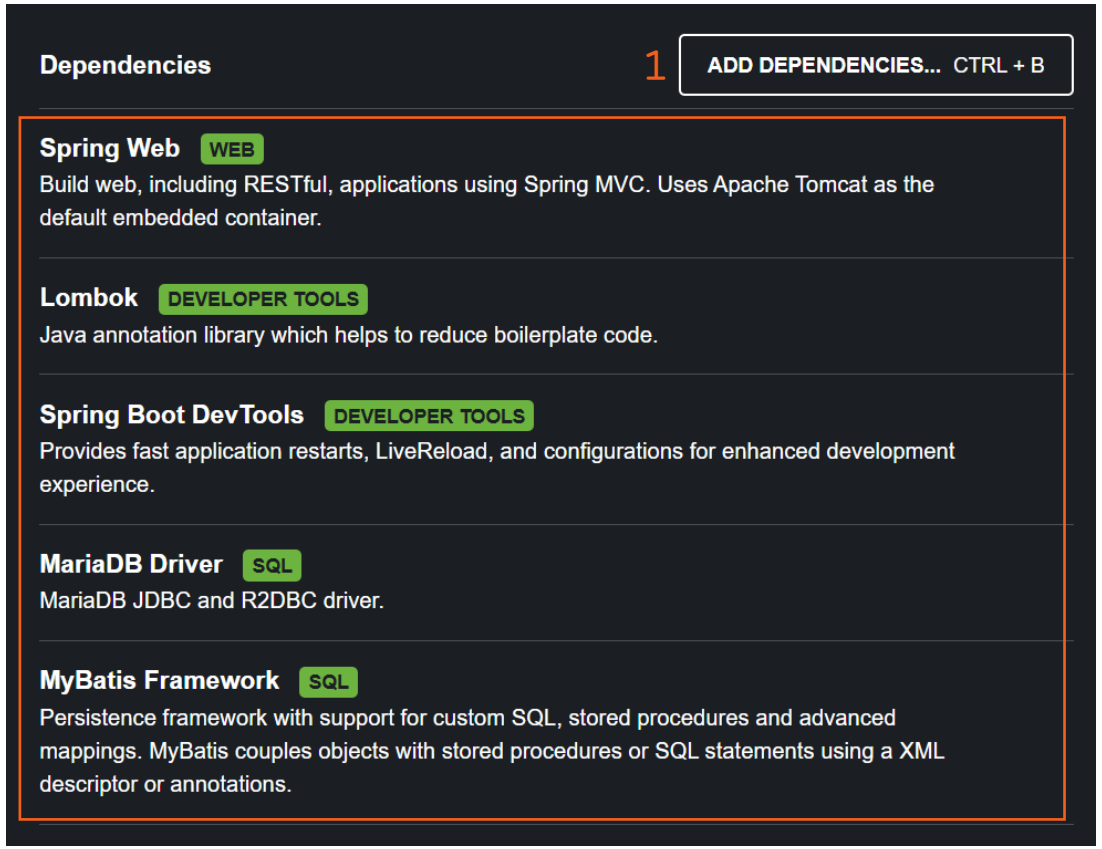
1. Group은 프로젝트를 생성하는 기업의 도메인을 작성한다.

   ex> 삼성 : com.samsung, 엘지 : com.lg

   Group을 변경하면 6번의 Package name이 자동 변경된다.
2. Artifact는 프로젝트가 완성되어 최종 실행 프로그램으로 배포할 때의 완성본의 이름을 작성한다. 통상적으로는 3번의 Name도 동일하게 작성한다.

   Artifact를 변경하면 3번의 Name도 자동 변경된다.
3. Name은 프로젝트의 이름이다. 프로젝트의 이름과 프로젝트 완성본의 이름이 다를수 있지만, 강의에서는 똑같은 이름을 부여한다.
4. 프로젝트의 설명을 간단하게 작성하는 부분이다.
5. Package는 프로젝트 생성 시 기본으로 만들어지는 패키지명을 입력한다. Group과 Artifact를 작성하면 Package가 자동으로 변경된다.

   그렇기 때문에 강의에서는 자동으로 만들어지는 package name을 사용할 것이며 건들지 않는다.
6. Packaging이란 코드 작성 완료 후 실행 파일을 만들어 배포용 설치파일을 만드는 절차를 의미한다. 쉽게 말해 우리가 프로젝트의 소스를 다 완성하면 해당 프로젝트를 실행할 수 있는 파일로 만들어줘야 한다. 이렇게 실행파일로 만들면 파일 확장자가 .jar와 .war로 만들어지는데, 둘 중 어떤 확장자로 만들것인지 선택하는 메뉴이다. 우리는 기본값인 jar를 선택한다.
7. 스프링 프로젝트에서 사용하는 자바 버전을 선택하는 메뉴이다. 우리는 jdk 17버전을 설치했다.



- 1번의 ADD DEPENDENCIES 메뉴를 클릭하여 프로젝트의 의존성을 추가하는 메뉴이다. 추가한 의존성은 2번처럼 목록으로 나타난다.
- 의존성 추가란 간단히 말해 해당 프로젝트에서 사용할 라이브러리를 추가하는 것이라 생각하면 된다. 리액트에서 axios 라이브러리를 설치하는 절차와 같은 것이라 생각하면 된다.

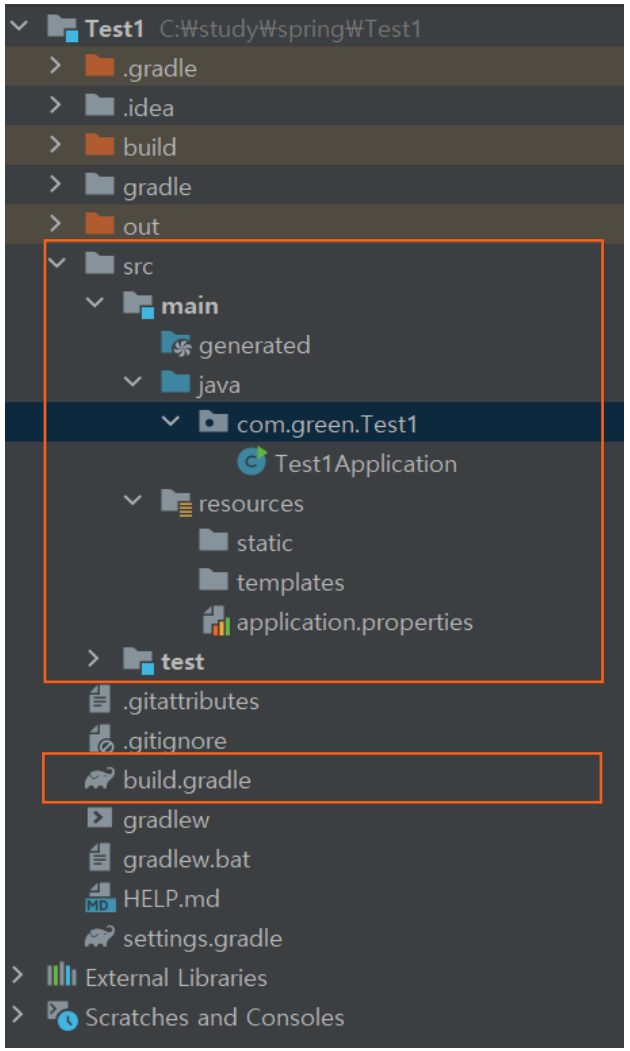
다음은 우리가 학습을 하며 많이 추가할 의존성이다.

- Spring Web : 프로젝트에 Web 개발에 필요한 여러 기능을 추가한다.
- Lombok : 생성자, getter, setter, toString 등 많이 사용하는 자바 코드를 어노테이션 사용으로 생성해준다.
- Spring Boot DevTools : 프로젝트 개발 시에 유용한 도구로, 자동 재시작, 변경된 설정파일 자동 반영 등의 기능을 제공한다.
- MariaDB Driver : 스프링 프로젝트에 MariaDB를 연동할 수 있는 기능을 추가한다.
- Mybatis Framework : 연동한 DB를 자바코드로 구현할 때 사용하는 Framework를 추가한다.

이렇게 기본적인 설정을 다 끝낸 후, 페이지 하단의 GENERATE 버튼을 클릭하면 압축파일로 프로젝트를 내려받는다.

압축파일은 반드시 '여기에 풀기'로 압축을 풀고, 압축을 푼 폴더를 IntelliJ에서 열어서 작업을 시작하면 된다.

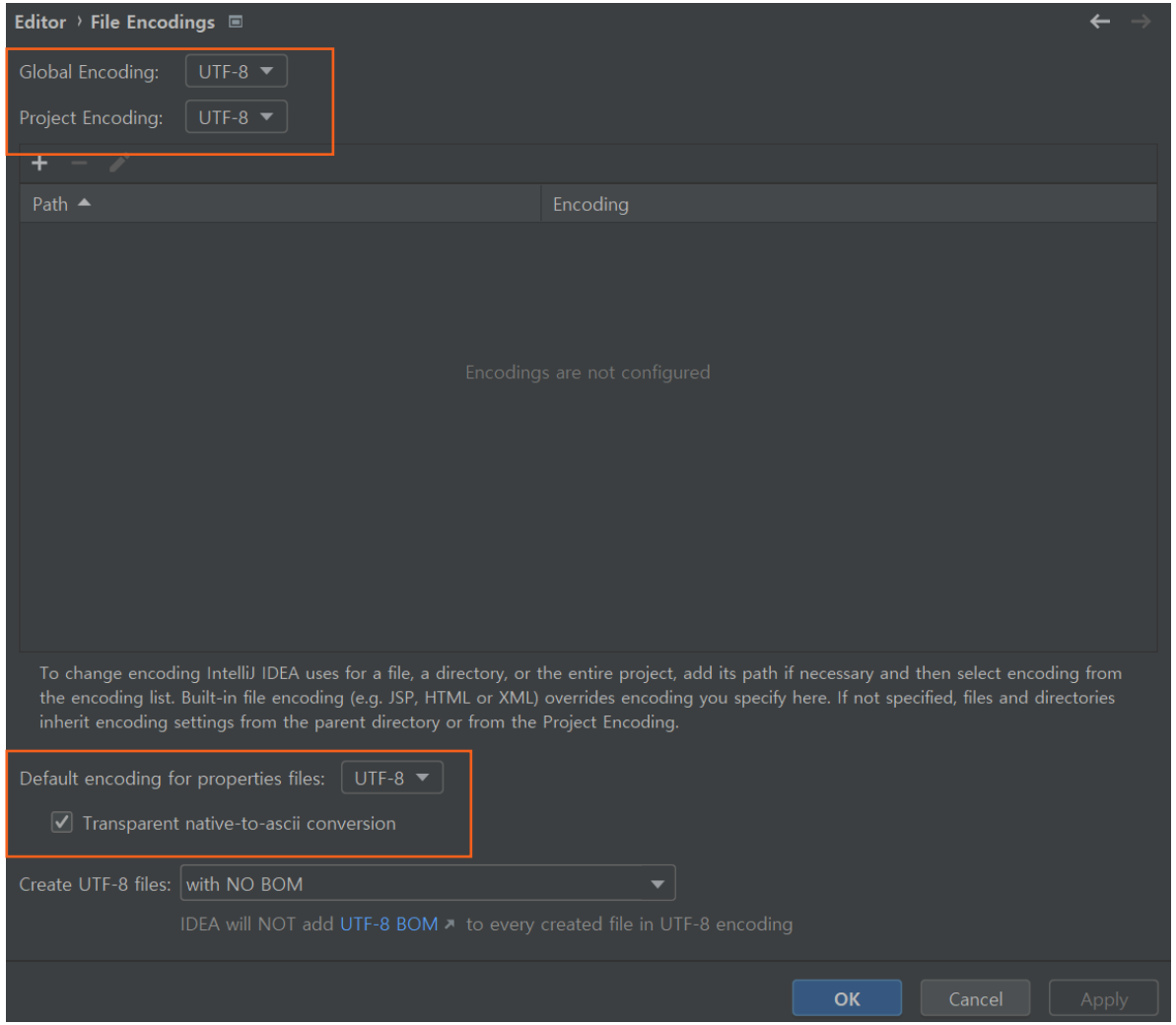
생성된 프로젝트의 구조는 다음과 같다. 프로젝트에서 크게 봐야할 부분은 src와 build.gradle 부분이다.



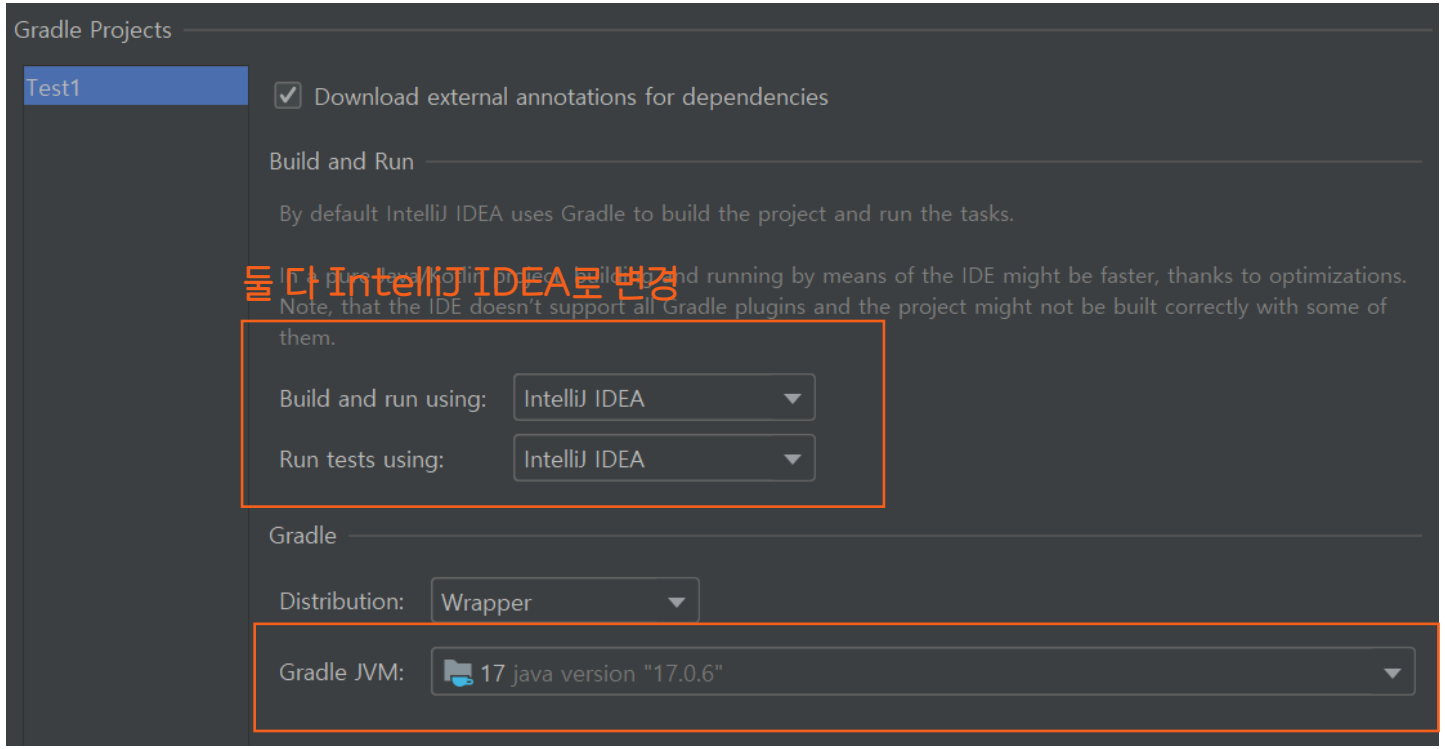
- src에는 우리가 작성하는 코드가 들어간다.
- src는 크게 main과 test로 나뉘는데 test에는 테스트에 필요한 코드를 작성하며, main에서 실제 코드를 작성한다. 우리는 당분간 main에서만 코드를 작성한다.
- main은 또 다시 java와 resources 영역으로 나뉜다. resources 영역에는 정적 데이터(html, css, javascript, 이미지 등)이 들어간다. 우리는 react를 사용하기 때문에 resources영역에는 웹 개발에 필요한 이미지 파일만 추가된다.
- main 안의 java 폴더에 실제 우리가 작성하는 클래스파일을 추가한다. 이때 반드시 모든 클래스파일은 default 패키지 안에 작성한다. (여기서 default 패키지란, 프로젝트 생성 시 자동으로 만들어진 패키지를 의미한다. 이유는 나중에 설명! 아주 중요함!!!)
- default 패키지 안에는 main 메서드가 있는 클래스 파일이 하나 존재한다. 이 파일의 이름은 프로젝트 생성 시 작성한 Artifact + application으로 자동으로 만들어진다. 이 클래스의 메인메서드를 실행하면 스프링 프로젝트가 실행된다.
- build.gradle 파일에는 프로젝트의 여러 정보가 작성되어 있다. 특히, dependencies 부분에는 우리가 프로젝트 생성 시 추가한 의존성 정보가 들어가 있는 것을 확인할 수 있다.

프로젝트를 생성하면 기본과 같이 초기 설정을 해준다.

File -> Settings -> Editor -> File Encodings에 들어가서 인코딩을 utf-8로 변경. 마지막 체크박스 체크. -> Apply 클릭



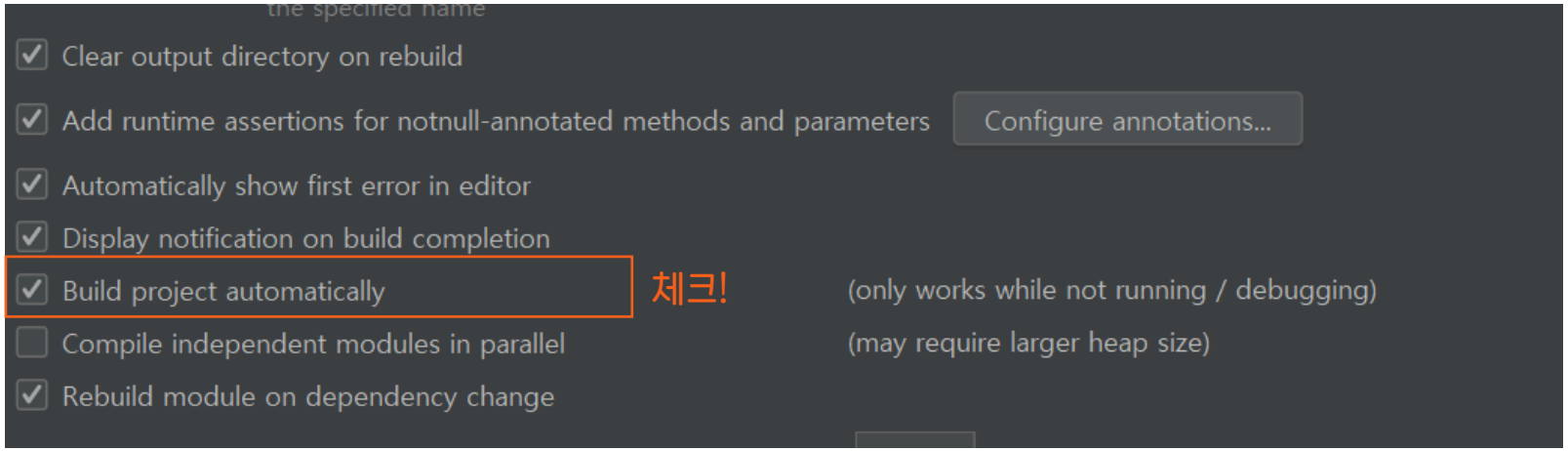
File -> Settings -> Build, Execution, Deployment -> Build Tools -> Gradle에 아래와 같이 변경 후 Apple 버튼 클릭



둘 다 IntelliJ IDEA로 변경

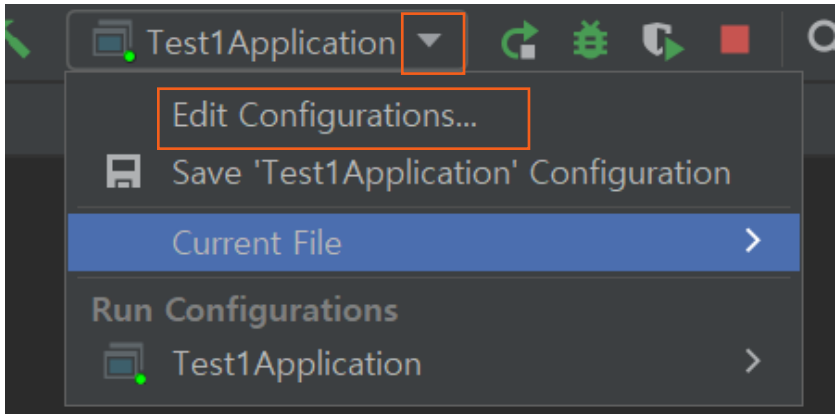
현재 설치된 JDK로 변경

File -> Settings -> Build, Execution, Deployment -> Compiler 클릭. 아래와 같이 체크박스 체크 후 Apply 클릭, OK 클릭!

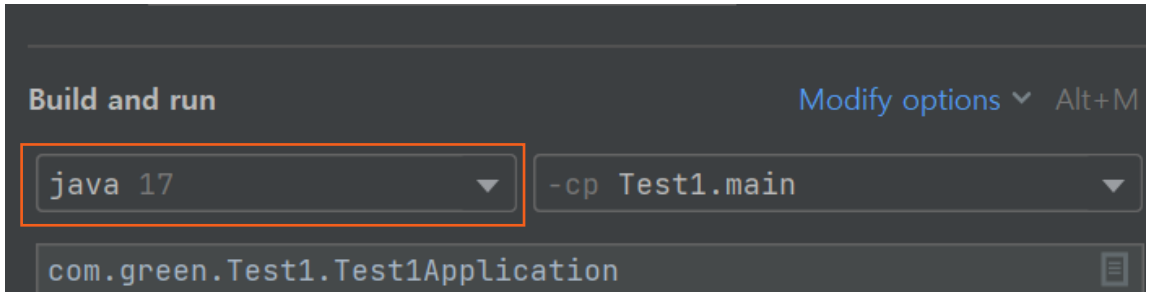


프로젝트 최초 실행(프로젝트 생성 시 만들어지는 클래스 메인 메서드 왼쪽의 초록색 화살표 클릭).

실행하면 인텔리제이 상단 우측에 아래와 같이 실행되고 있는 프로젝트의 설정을 변경할 수 있는 Edit Configuration 메뉴가 나타남. 클릭!



팝업창에서 Build and run 메뉴를 pc에 설치된 jdk 버전으로 변경



마지막으로 lombok 라이브러리를 설치한다. (이건 프로젝트마다 할 필요 없음! 한 번만 하면 됨)

spring.io 싸이트에서 프로젝트를 생성할 때 의존성으로 lombok(롬북)을 추가했지만, 별도로 플로그인을 설치해줘야 한다.

File -> Settings -> Plugins에 들어와서 검색창에 lombok 입력 후 install 클릭!