

02. 스프링 부트 시작하기

- - 환경 준비
 - 프로젝트 생성
 - hello 컨트롤러
 - Hello API 테스트
 - HTTPie
 - HTTP 요청과 응답

환경 준비

```
#java version
java -version

# sdk
curl -s "https://get.sdkman.io" | bash

sdk list java
sdk install java 11.0.18-zulu

#      version
sdk use java 11.0.18-zulu

# springboot cli
sdk install springboot 2.7.6
```

프로젝트 생성

1. Spring Initializr 웹 사용
2. IDE에 있는 initializer 사용
3. spring boot CLI 사용

```
a. # spring initializer
#
spring init

#
spring shell

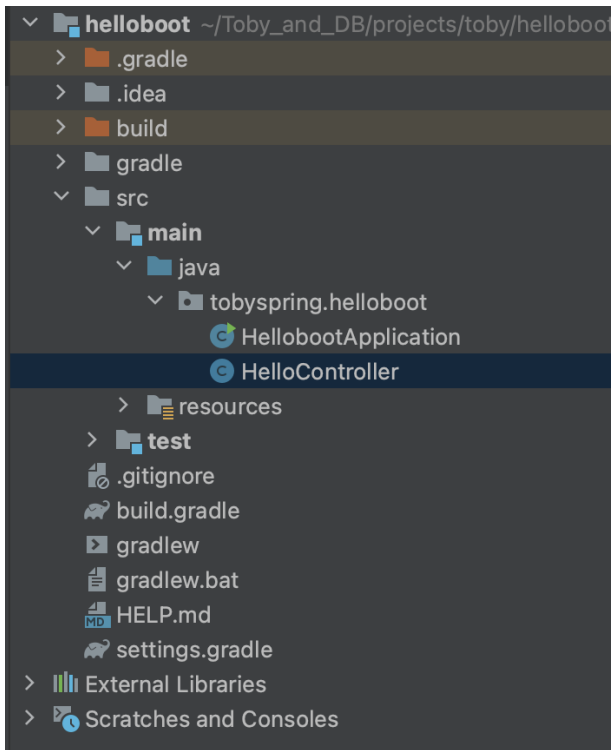
#init
$init

# init -b = boot version, -d = dependency, -g == group, -j == java version, -n == project name, -x
==
inti -b 2.7.6 -d web -g tobyspring -j 11 -n helloboot -x helloboot

# application
./gradlew bootRun
```

hello 컨트롤러

파일 구조



코드:

```
package tobyspring.helloboot;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("")
public class HelloController {
    @GetMapping("/hello")
    public String hello(String name){
        return "Hello " + name;
    }
}
```

Hello API 테스트

웹 브라우저를 사용해 테스트하는게 맞을까?

세밀하게 테스트 해야되는 경우에는?

다양한 방법들

- chrome 개발자 도구
 - header 내용 확인 가능
 - status code
 - Content-Type: json인지 html인지 확인 가능
 - 기대했던 응답과 맞는지 확인
- curl
 - http 요청을 만들고 응답 볼 수 있음
- HTTPie
- IntelliJ IDEA Ultimate - http request
- JUnit test
- 등등

HTTPIe

선호 이유: 간단한 명령어로 http 요청과 응답을 받아볼 수 있음

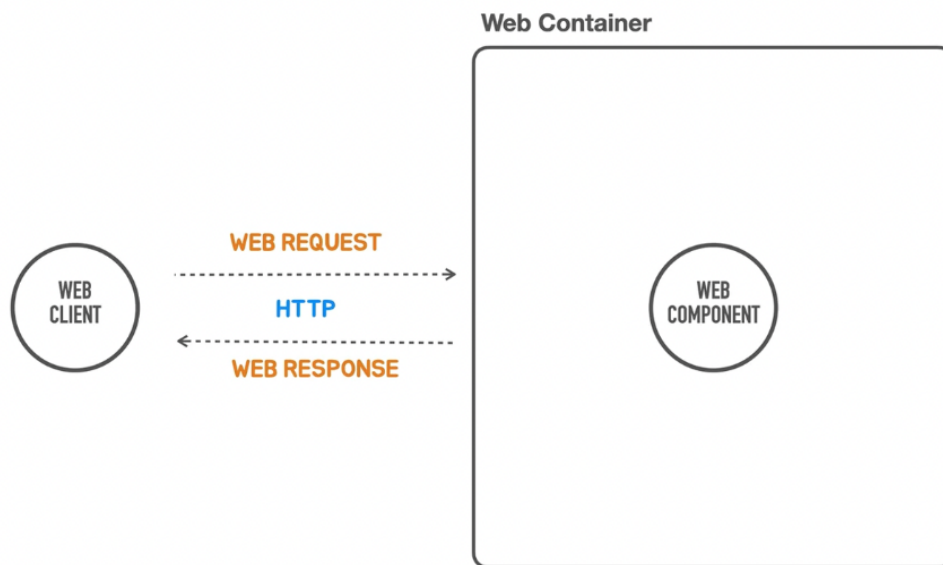
ex :

```
> http -v ":8080/hello?name=Spring"
GET /hello?name=Spring HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8080
User-Agent: HTTPIe/3.2.1

HTTP/1.1 200
Connection: keep-alive
Content-Length: 12
Content-Type: text/plain;charset=UTF-8
Date: Thu, 01 Dec 2022 11:11:26 GMT
Keep-Alive: timeout=60
```

http 요청과 응답을 확인하는 것을 모두 마쳐야 API에 대한 최종 테스트가 마무리된 것!

HTTP 요청과 응답



웹 request와 response의 기본 구조를 이해하고 내용을 확인할 수 있어야 한다.

Request

- request line:
 - method (GET 등)
 - path (ex. /hello)
 - http version (ex. HTTP/1.1)
 - Accept: 수용하는 content-type
- Headers
 - header에 따라 요청하는 방식이 달라지거나
 - 응답의 타입이 바뀌기도 함

- Message body
 - header에 설정된 content-type에 맞는 message body

Response

- Status Line
 - HTTP version
 - Status Code
 - Status Text
- Headers
- Message body