

2025년 상반기 K-디지털 트레이닝

# 스프링의 특징과 의존성 주입

---

[KB] IT's Your Life

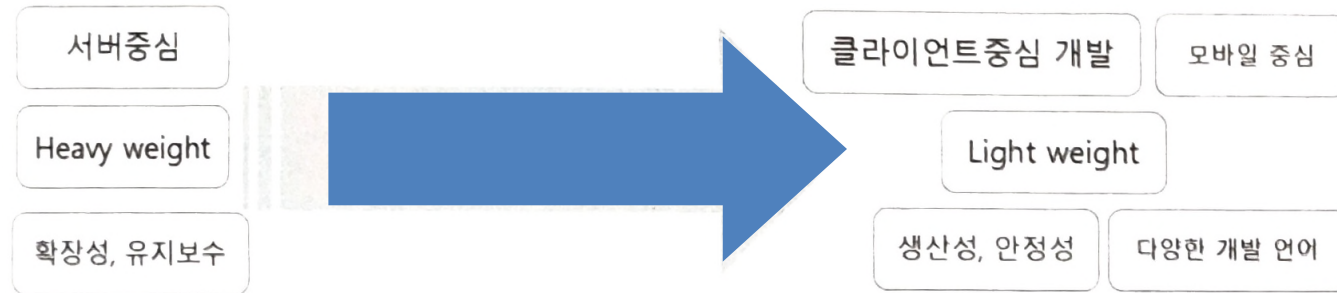
예전엔 java ee 상속 인터페이스 ... 구현으로 만들었음. -- 웹앱인데 너무 무거워짐.  
상속, 인터페이스 말고 단순 객체POJO만으로도 만들수있어 == 가벼워짐!

스프링특징

## 1 스프링 프레임워크의 간략한 역사

\* KB국민은행

### ✓ 경량 프레임워크(light-weight Framework) ✓



# 1 스프링 프레임워크의 간략한 역사

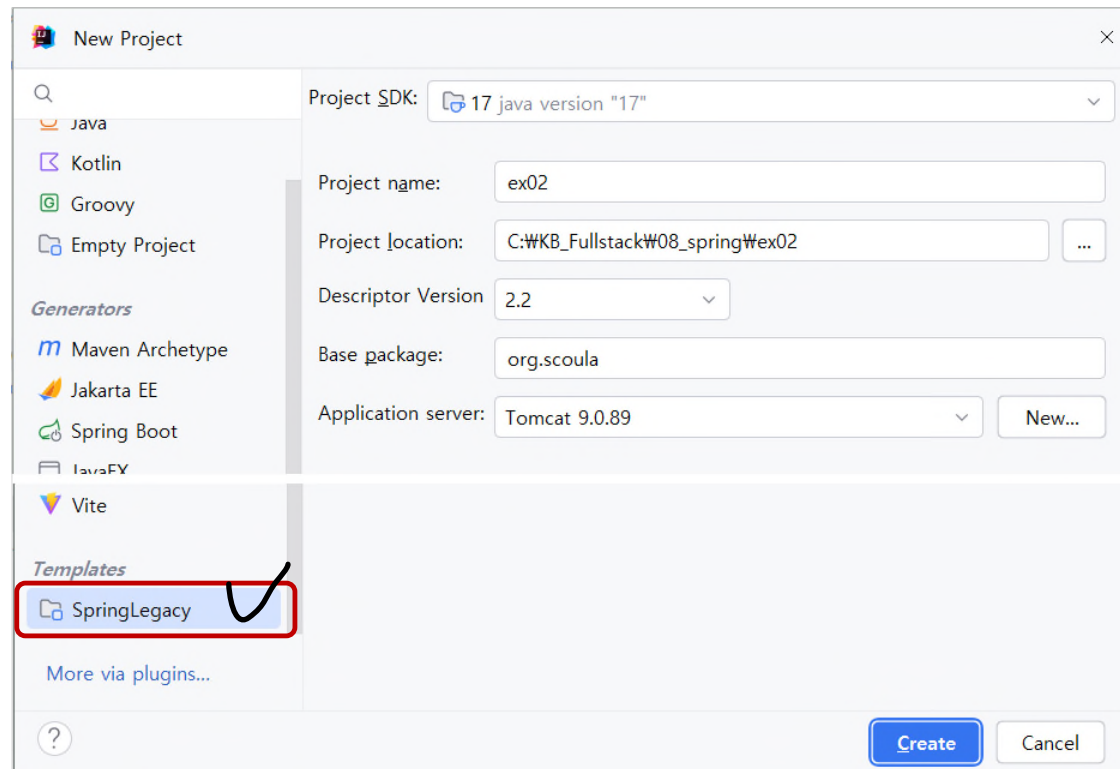
## ✓ 스프링의 주요 특징

- POJO(Plain Old Java Object) 기반의 구성 ✓
- \* ○ 의존성 주입(Dependency Injection) ✓
- AOP(Aspect Oriented Programming) 지원
- 편리한 MVC 구조 ✓
- WAS에 종속적이지 않은 개발 환경 ✓

## 2 의존성 주입 테스트

### ✓ 프로젝트 만들기

- File > New > Project...
  - Templates: Spring Legacy



## 2 의존성 주입 테스트

### settings.gradle

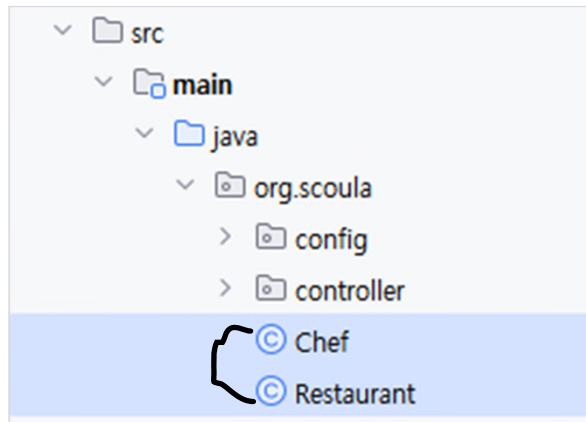
```
rootProject.name = "ex02" ✓
```

gradle sync!

## 2 의존성 주입 테스트

### ✓ 예제 클래스 생성

- org.scoula
  - Chef.java
  - Restaurant.java



레스토랑엔 반드시 셰프가 있어야한다는 전제

## 2 의존성 주입 테스트

### Chef.java

```
package org.scoula;
```

```
import org.springframework.stereotype.Component;
```

```
✓ @Component
```

```
public class Chef {  
}
```

## 2 의존성 주입 테스트

### Restaurant.java

```
package org.scoula;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import lombok.Data;
import lombok.Setter;

@Component
@Data
public class Restaurant {
    @Autowired
    private Chef chef;
}
```

권장하지 않는다.  
가독성도 떨어지고  
final 권장사항 못지키고잉.

생성자에 에다가 Autowired해보자.  
롬복도 활용해보자

생성자가 하나밖에 없다면  
묵시적으로  
생성자에 autowired된다.



## 2 의존성 주입 테스트

### RootConfig.java

```
package org.scoula.config;
```

```
import org.springframework.context.annotation.ComponentScan;  
import org.springframework.context.annotation.Configuration;
```

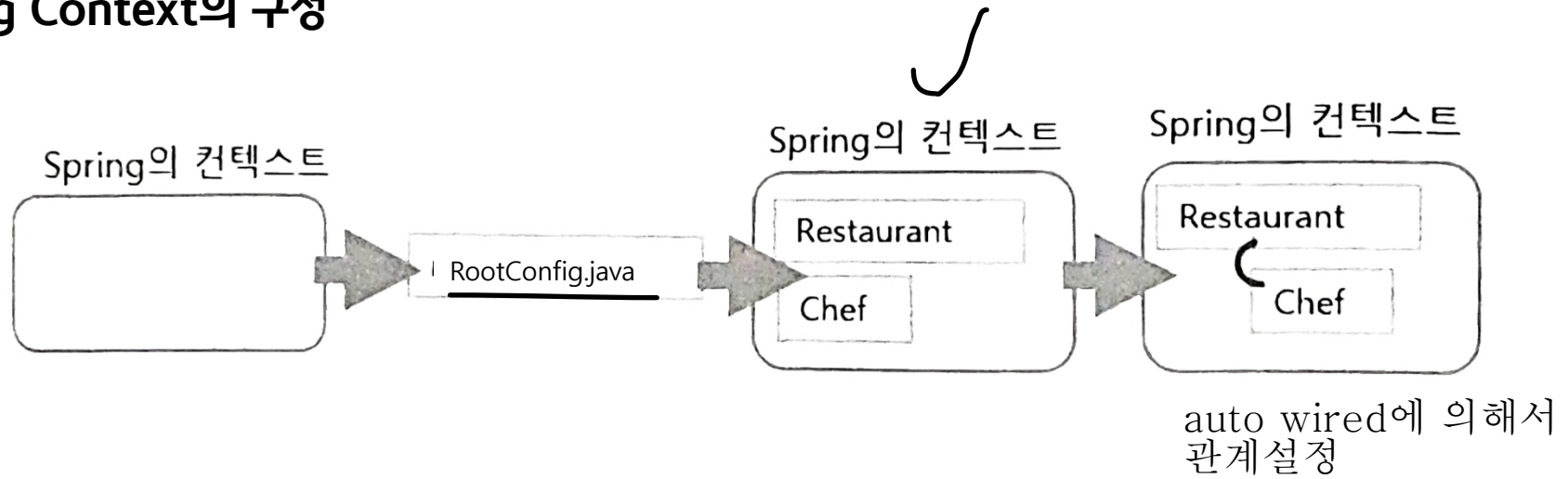
```
@Configuration
```

```
@ @ComponentScan(basePackages = {"org.scoula"}) ✓  
public class RootConfig {
```

```
}
```

### 3 스프링이 동작하면서 생기는 일

#### ✓ Spring Context의 구성

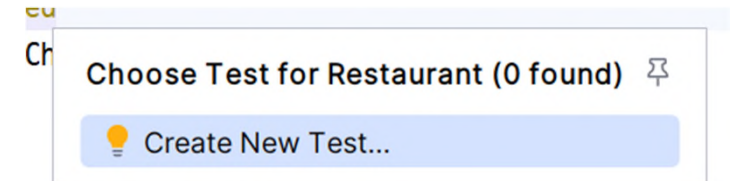
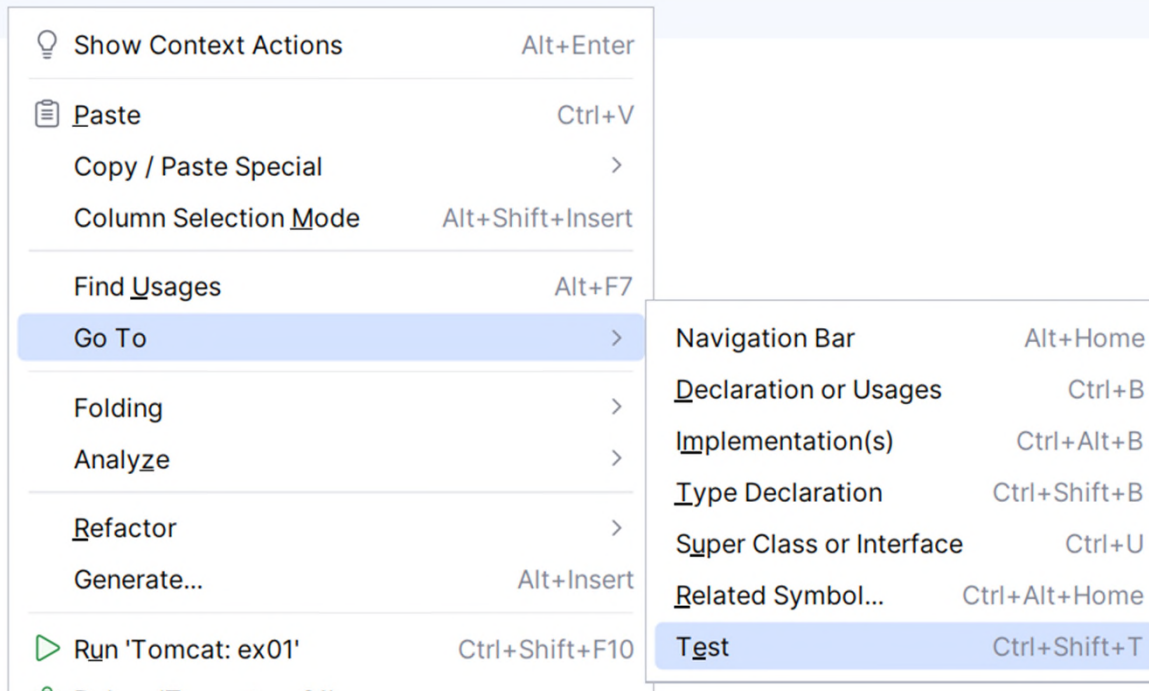


### 3 의존성 주입

#### ✓ 테스트 코드를 통한 확인 ✓

##### ○ 테스트 클래스 만들기

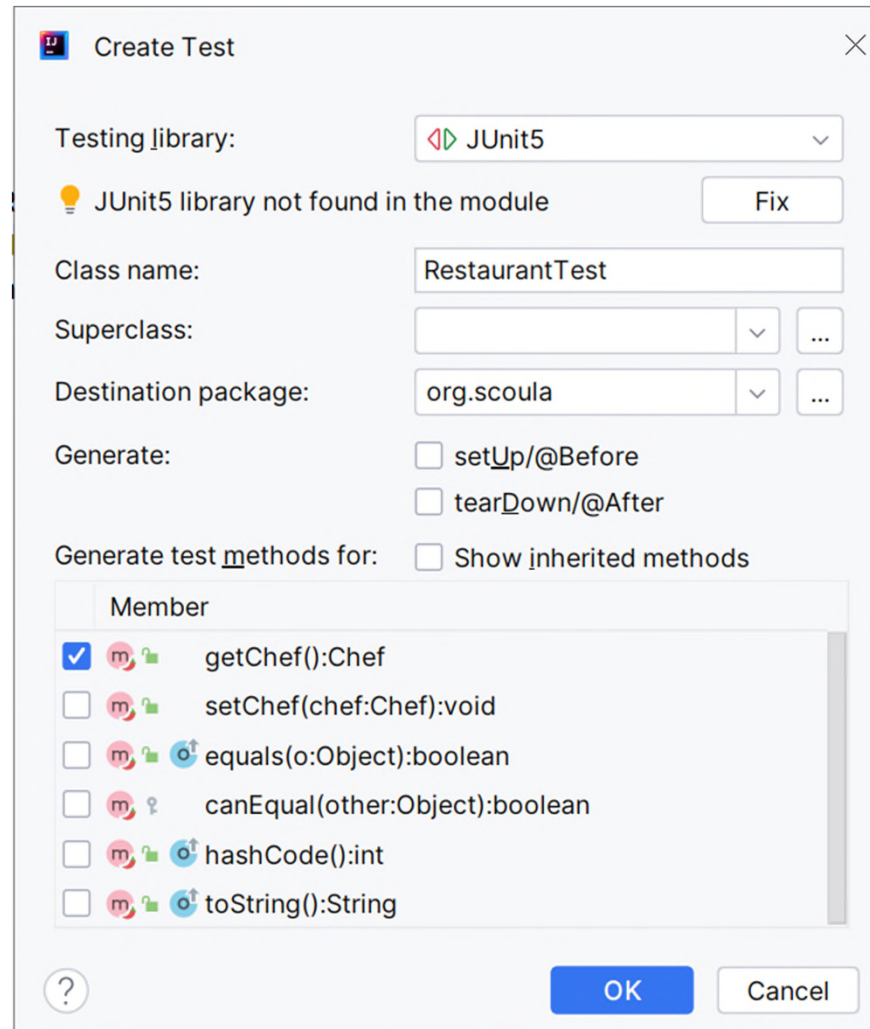
- Restaurant.java 편집창 >> Go To > Test ✓



### 3 의존성 주입

#### ✓ 테스트 코드를 통한 확인

##### ○ 테스트 클래스 만들기



### 3 의존성 주입

#### test :: RestaurantTest.java

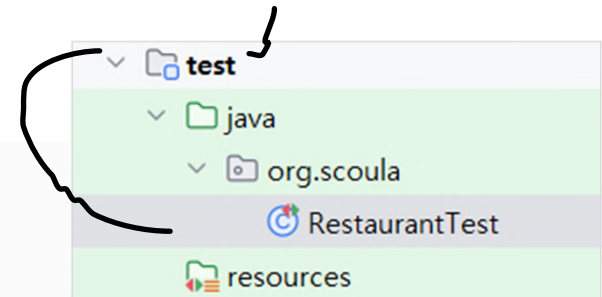
```
package org.scoula;
```

```
import org.junit.jupiter.api.Test;
```

```
import static org.junit.jupiter.api.Assertions.*;
```

```
class RestaurantTest {
```

```
    @Test  
    void getChef() {  
    }  
}
```



## 3 의존성 주입

## test :: RestaurantTest.java

```

...
import static org.junit.jupiter.api.Assertions.assertNotNull;

@ExtendWith(SpringExtension.class)
@ContextConfiguration(classes = {RootConfig.class})
@Log4j2
class RestaurantTest {
    @Autowired
    private Restaurant restaurant;

    @Test
    void getChef() {
        assertNotNull(restaurant);
        log.info(restaurant);
        log.info("-----");
        log.info(restaurant.getChef());
    }
}

```

log라는 멤버 자동 추가

널이면 예외발생



```

INFO : org.scoula.RestaurantTest - Restaurant(chef=org.scoula.Chef@25230246)
INFO : org.scoula.RestaurantTest - -----
INFO : org.scoula.RestaurantTest - org.scoula.Chef@25230246

```

### 3 의존성 주입

#### ✓ @Log4j2

- Lombok을 이용해서 로그를 기록하는 Logger 변수 생성
- 로그 설정 파일

- src/main/resources/log4j2.xml
  - 실제 운영을 위한 로거 설정
- src/test/resources/log4j2.xml
  - 테스트를 위한 로거 설정
  - 없으면 운영을 위한 로거 설정이 적용

