

2024년 상반기 K-디지털 트레이닝

JUnit5

[KB] IT's Your Life

✓ JUnit

- 단위 테스트 라이브러리
- 빌드 시스템을 gradle로 지정한 경우 JUnit5가 디폴트 단위 테스트 라이브러리로 설정됨
- test 폴더 자동 생성됨
 - 실제 코드와 테스트 코드를 분리해서 관리

✓ JUnit

- `@DisplayName("테스트이름")`
 - 테스트 구분 제목이 됨
- `@Test`
 - 테스트 메서드를 지정

✓ Assertions

- 여러가지 단정문 static 메서드를 제공
- `assertEquals(실제값, 기대값)`
 - 실제값과 기대값이 다르면 테스트 실패 → 예외 발생

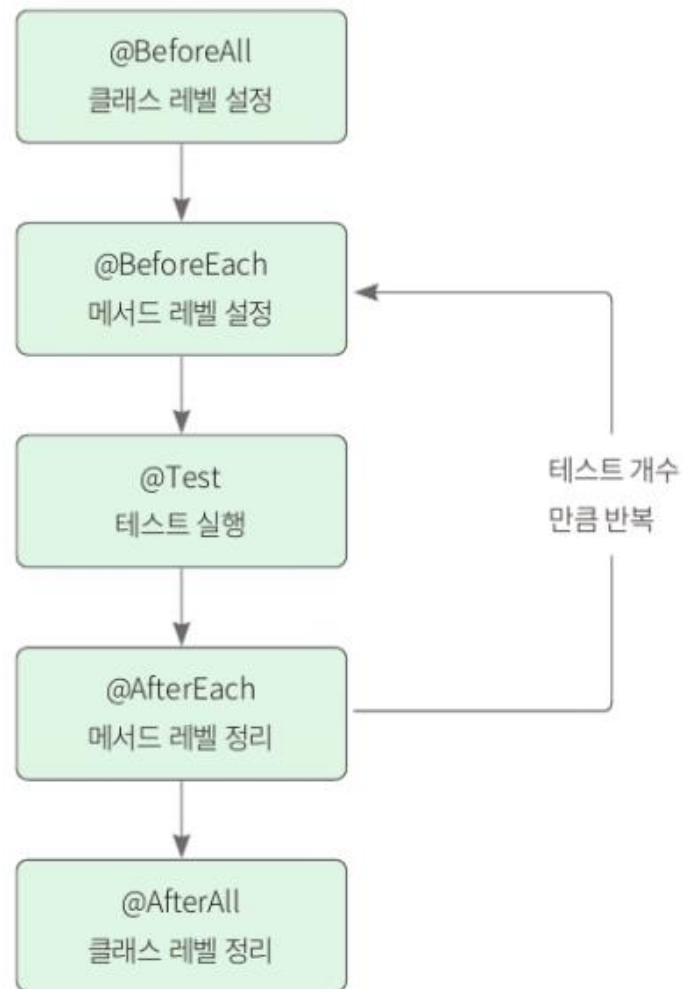
✓ JUnit으로 테스트 클래스 만들기

- src → test → java/JUnitTest.java

```
public class JUnitTest {  
  
    @DisplayName("1+2는 3이다")  
    @Test  
    public void junitTest() {  
        int a = 1;  
        int b = 2;  
        int sum = 3;  
  
        Assertions.assertEquals(a+b, sum);  
    }  
}
```

✔ JUnit 주요 어노테이션

- **@BeforeAll**
 - 전체 테스트 실행 전 1회 호출
 - static 메서드로 정의
- **@BeforeEach**
 - 각 테스트 케이스마다 실행 전 호출
- **@AfterEach**
 - 각 테스트 케이스마다 실행 후 호출
- **@AfterAll**
 - 전체 테스트 실행 후 1회 호출
 - static 메서드로 정의



✓ JUnitCycleTest .java

```
public class JUnitCycleTest {  
    @BeforeAll // 전체 테스트 시작전 1회 실행, static 선언  
    static void beforeAll() {  
        System.out.println("@BeforeAll");  
    }  
  
    @BeforeEach // 테스트 케이스를 시작하기 전마다 실행  
    public void beforeEach() {  
        System.out.println("@BeforeEach");  
    }  
  
    @Test  
    public void test1() {  
        System.out.println("test1");  
    }  
  
    @Test  
    public void test2() {  
        System.out.println("test2");  
    }  
  
    @Test  
    public void test3() {  
        System.out.println("test3");  
    }  
}
```

✓ JUnitCycleTest .java

```
@AfterEach // 테스트 케이스를 종료하기 전마다 실행
public void afterEach() {
    System.out.println("@AfterEach");
}
```

```
@AfterAll // 전체 테스트를 마치고 종료하기 전 1회. static 선언
static void afterAll() {
    System.out.println("@AfterAll");
}
```

```
}
```

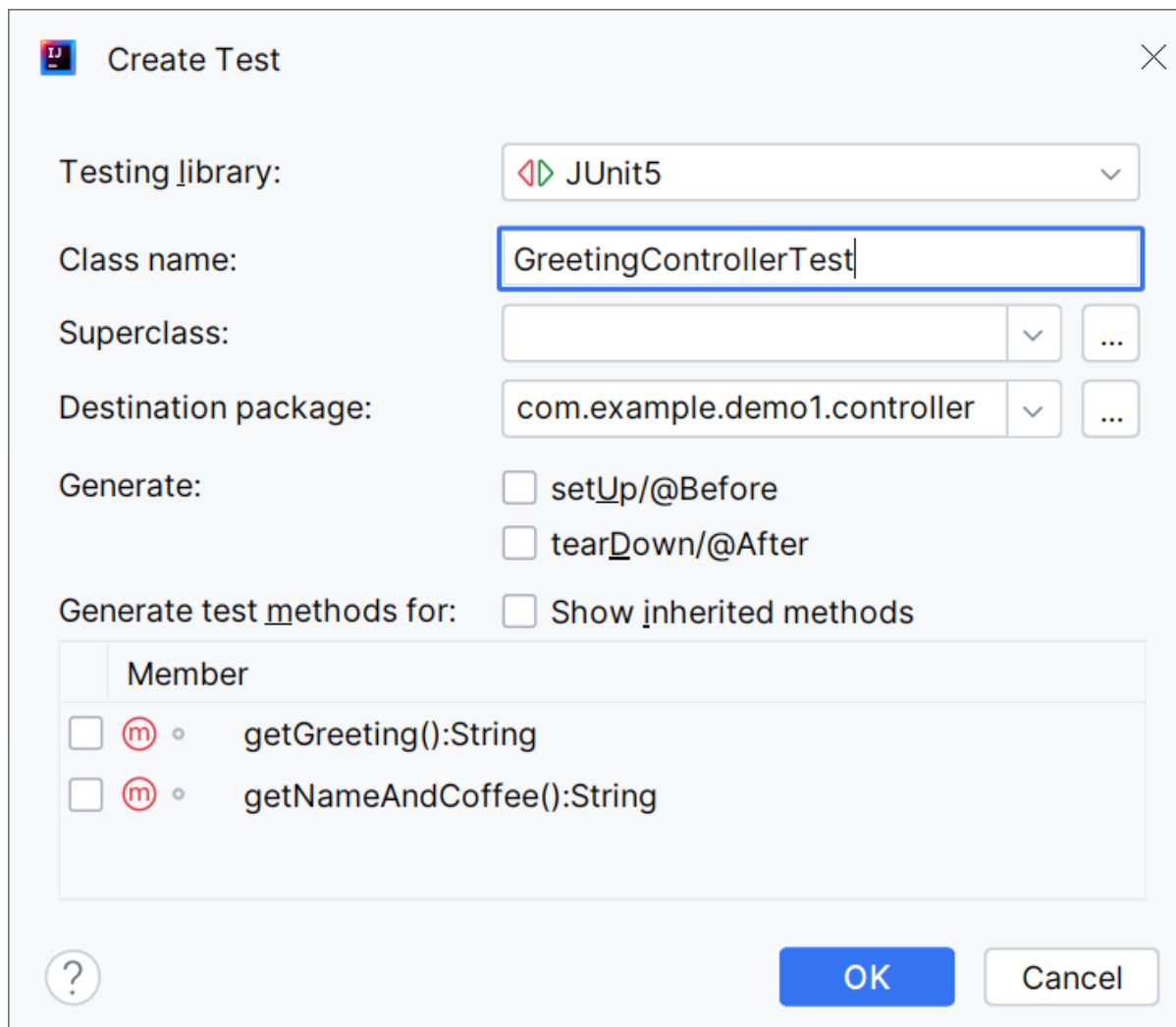
```
@BeforeAll
@BeforeEach
test1
@AfterEach
@BeforeEach
test2
@AfterEach
@BeforeEach
test3
@AfterEach
@AfterAll
```

✔ AssertJ

메서드 이름	설명
isEqualTo(A)	A 값과 같은지 검증
isNotEqualTo(A)	A 값과 다른지 검증
contains(A)	A 값을 포함하는지 검증
doesNotContain(A)	A 값을 포함하지 않는지 검증
startsWith(A)	접두사가 A인지 검증
endsWith(A)	접미사가 A인지 검증
isEmpty()	비어있는 값인지 검증
isNotEmpty()	비어있지 않은 값인지 검증
isPositive()	양수인지 검증
isNegative()	음수인지 검증
isGreaterThan(1)	1보다 큰 값인지 검증
isLessThan(1)	1보다 작은 값인지 검증

✓ 테스트 클래스 만들기

- 테스트 대상 클래스 >> Generate... > Test...



The image shows the 'Create Test' dialog box in an IDE. It is titled 'Create Test' with a JUnit icon and a close button. The dialog contains several fields and checkboxes for configuring a new test class.

Testing library: JUnit5

Class name: GreetingControllerTest

Superclass: (empty) ...

Destination package: com.example.demo1.controller ...

Generate:

- ☐ setUp/@Before
- ☐ tearDown/@After

Generate test methods for: ☐ Show inherited methods

Member	
<input type="checkbox"/> (m)	getGreeting():String
<input type="checkbox"/> (m)	getNameAndCoffee():String

Buttons: ? OK Cancel