7/13 ~ 7/17 2주차 - 6

Junit을 이용한 단위 테스트

- Junit은 자바용 단위 테스트 작성을 위한 산업 표준 프레임워크다.

```
public class Calculator {
  pulbic double sum(double a, double b) {
    return a + b;
  }
}
```

```
package com.y2kpooh.junitTest;

import org.junit.Test;
import static org.junit.Assert.*;

public class CaluratorTest {                    ← ①
   @Test                                        ← ②
   public void testSum(){
      Calcurator c = new Calcurator();
      double result = c.sum(10, 50);            ← ③
      assertEquals(60, result, 0);              ← ④
   }
}
```

1. 테스트 클래스는 반드시 public으로 선언해야한다 테스트 클래스명 + Test
2. @Test를 이용한다. junit이 애너테이션이 선언된 메서드을 알아서 실행을 해준다.
3. Calculraor 클래스의 인스턴스를 선언하여 sum 메서드에 10, 50인자 값을 세팅하여 result 변수에 결과값을 리턴 받는다.
4. Junit 프레임워크에서의 assert클래스의 정적 메서드 assertEquals를 이용하여 결과값을 확인한다. (예상값, 실제값, 허용오차)

Junit assert 주요 메서드 및 사용예시

참고 링크

Assert (JUnit API)

java.lang.Object org.junit.Assert public classAssert extends java.lang.Object A set of assertion methods useful for writing tests. Only failed assertions are recorded. These methods can be used directly: Assert.assertEquals(...), however, they read better if they are referenced through static import: import static

http://junit.sourceforge.net/javadoc/org/junit/Assert.html

assert 메서드	설명
assertArrayEquals(a, b);	배열 A와 B가 일치함을 확인한다.
assertEquals(a, b);	객체 A와 B가 일치함을 확인한다.
assertSame(a, b);	객체 A와 B가 같은 객임을 확인한다. assertEquals 메서드는 두 객체의 값이 같은가를 검사는데 반해 assertSame메서드는 두 객체가 동일한가 즉 하나의 객인 가를 확인한다.(== 연산자)
assertTrue(a);	조건 A가 참인가를 확인한다.
assertNotNull(a);	객체 A가 null이 아님을 확인한다.

테스트 예시

```
@Autowired
RestServiceController controller;

MockMvc mock;

@BeforeEach
public void setup() {
    mock = MockMvcBuilders.standaloneSetup(controller).build();
}

@Test
public void selectAllTest() throws Exception {
    mock.perform(get("/api/countries").param("per", "5").param("page", "1"))
                .andExpect(status().isOk())
                // 전달된 content type 검증
                .andExpect(header().string("content-type", "application/json"))
                // 배열의 개수 검증
                .andExpect(jsonPath("$.data.length()", equalTo(5)))
                // 데이터 검증
                .andExpect(jsonPath("$.data[0].code", equalTo("ABW")))
                .andDo(print());
}
```

JSON 형태의 데이터를 전달하기 위해 **contentType에 MediaType.APPLICATION_JSON**으로 설정되었으며 **content로 json 문자열**을 넘겨준다.

```
@Test
@Transactional
public void insertTest() throws Exception {
    Country country = new Country();
    country.setCode("TTT");
    country.setName("TestCountry");
    country.setContinent("Asia");
    country.setRegion("Eastern Asia");
    country.setSurfaceArea(99434D);
    country.setPopulation(46844000L);
    country.setLocalName("Test Local Name");
    country.setGovernmentForm("Republic");
    country.setCode2("TT");

    // Jackson Databind 사용(객체 → JSON 문자열 변환)
    ObjectMapper mapper = new ObjectMapper();
    String jsonStr = mapper.writeValueAsString(country);

    mock.perform(post("/api/countries").contentType(MediaType.APPLICATION_JSON).content(jsonStr))
            .andExpect(status().isOk())
            .andExpect(jsonPath("$.success", is(true)))
            .andExpect(jsonPath("$.data", is(1)))
            .andDo(print());
}
```