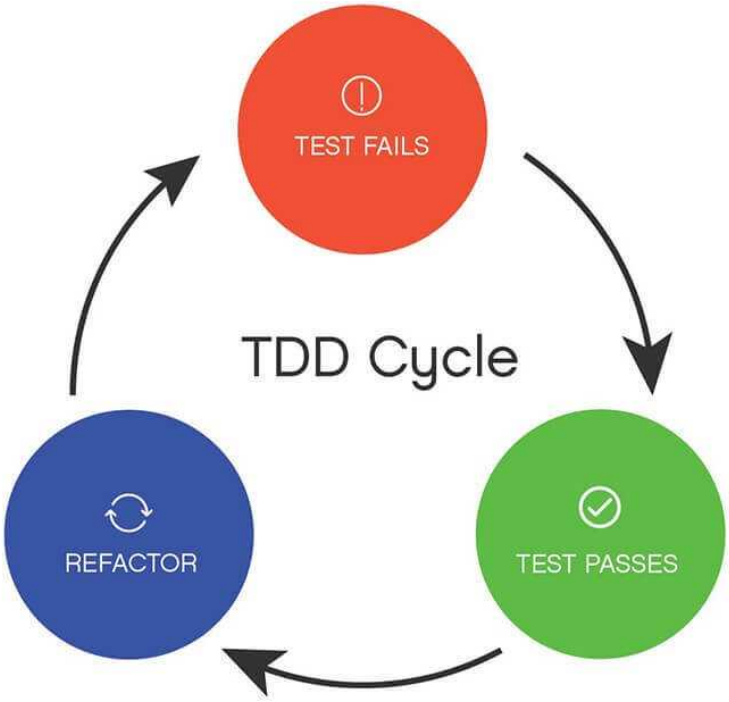
7/13 ~ 7/17 2주차 - 8

테스트 코드에 대해서

먼저 TDD와 단위 테스트는 다른 이야기다.

TDD는 테스트가 주도하는 개발을 이야기하고, 테스트 코드를 먼저 작성하는 것부터 시작을 한다.



- 항상 실패하는테스틀르 먼저 작성한다. (Red) 🔴
- 테스트가 통과하는 프로덕션 코드를 작성한다(Green) 🍏
- 테스트가 통과하면 프로덕션 코드를 리팩토링합니다. (Refactor) 🔵

반면에, 단위 테스트 즉, Unit Test는 TDD의 첫 번째 단계인 기능 단위의 테스트 코드를 작성하는 것을 이야기한다.

TDD와 달리 순수하게 테스트 코드만을 작성하는 것을 이야기 한다.!

단위 테스트 코드를 배운후 TDD에 대해서 배우자.

https://repo.yona.io/doortts/blog/issue/1

위키피다이에서 본 것을 바탕으로 테스트 코드를 작성함으로 얻는 이점을 적습니다.

단위 테스트 코드 작성의 이점

- 단위 테스트는 개발단계 초기에 문제를 발견하게 도와준다.
- 단위 테스트는 개발자가 나중에 코드를 리팩토링하거나 라이브러리 업그레이드 등에서 기존 기능이 올바르게 작동하는 지 확인할 수 있다.(예시 회귀 테스트)
- 단위 테스트는 기능에 대한 불확실성을 감소 시킬 수 있다.
- 단위테스트는 시스템에 대한 실제 문서를 제공한다. 즉, 단위 테스트 자체가 문서로 사용할 수 있습니다.

흔히 하는 테스트 방식...

1. 코드를 작성한다.
2. 프로그램을 실행한 뒤
3. Postman과 같은 테스트 도구로 api콜을 보내고

4. 요청결과를 눈으로 보고 개발 툴에 로그를 찍어 확인한다.
5. 결과가 다르면 실행 중지하고 프로그램을 수정한다.

여기서 2 ~ 5 과정을 매번 코드를 수정할때 마다 반복한다. ....

또 단위 테스트의 이점, 테스트 코드의 이점을 한 예로 들 수 있다.

개발자가 만든 기능을 안전하게 보호해 준다.

> 예를 들어 B라느 기능이 추가되어 테스트를 한다. B 기능이 잘되어 오픈 했더니 기존에 잘 되던 A기능에 문제가 생긴 것을 발견한다. 하나의 기능을 추가할때 마다 서비스의 모든 기능을 테스트할 수는 없기 때문에, 새로운 기능이 추가 될 때 기존 기능이 잘 작동되는 것을 보장하는 것이 테스트 코드이다.

> A를 비롯해 여러 경우를 테스트 코드로 구현해 놓았다면 테스트 코드를 수행만 하면 문제를 빨리 찾을 수 있다.

테스트 코드 프레임워크

Junit - Java

DBUnir - DB

CppUnit - C++

NUnit - .net

Junit 버전 5까지 나왔는데 현업에선 4를 많이 사용하고 있다고 한다.

이건 물어보고 진행하고 지금 프로젝트는 5.3.1를 dependency에 추가해놨다.

다음장에서 실제 코드를 작성해본다.