프론트 엔드 개발자가 알아야 하는 컴퓨터 공학 지식

소프트웨어 공학(SW Engineering)

소프트웨어 공학 | 프론드 엔드 개발자가 알아야 하는 CS 지식

강사 나동빈

프론트 엔드 개발자가 알아야 하는 컴퓨터 공학 지식

소프트웨어 공학(SW Engineering)

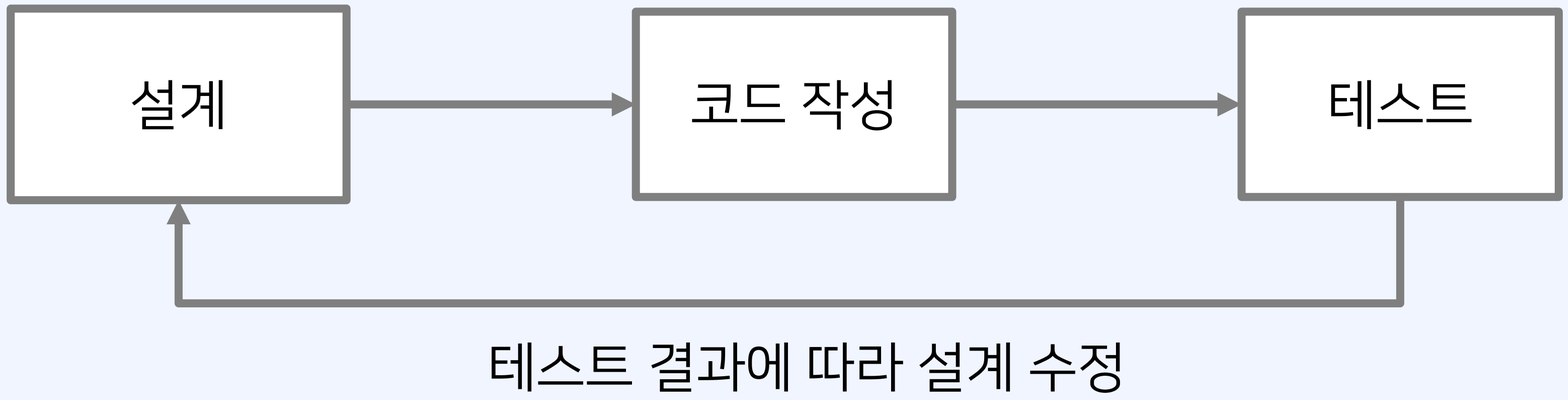
TDD (Test Driven Development): 테스트 주도 개발

- 소프트웨어 개발 방법론 중 하나로, 작은 단위의 테스트 케이스를 활용한다.
- ① 먼저 테스트 케이스를 작성한 뒤에 ② 이를 통과하는 코드를 작성하는 방식으로 개발한다.
- 짧은 개발 주기를 반복하는 개발 과정에 적합하다.

→ **애자일(Agile)** 개발 방법론 중 eXtream Programming(XP)의 "Test-First"과 맥락을 같이 한다.

일반적인 개발 과정

- 일반적인 개발 과정은 다음과 같다.



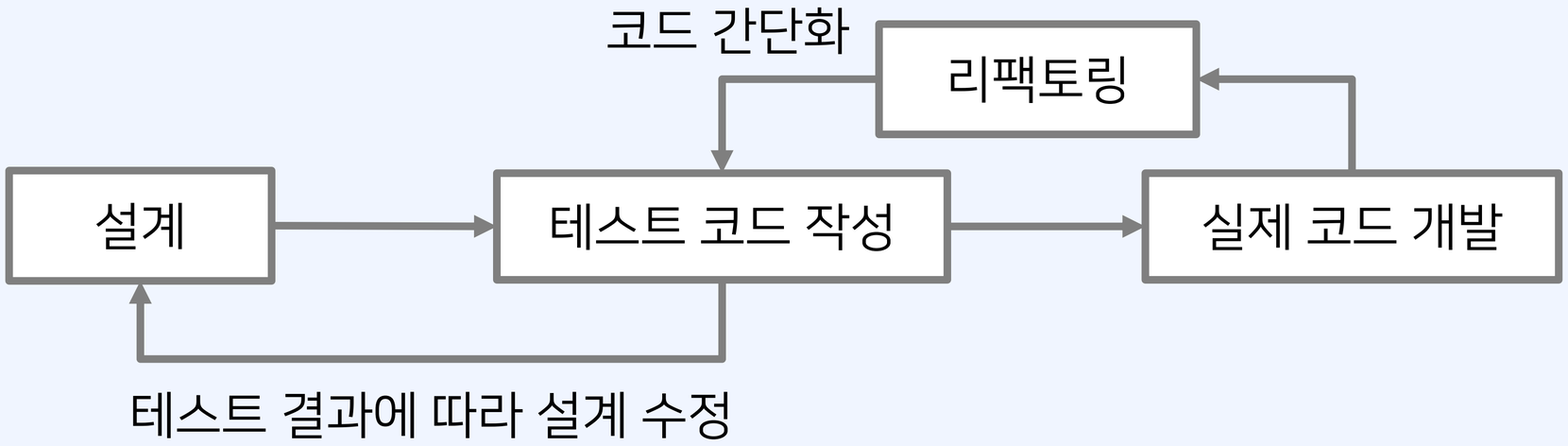
일반적인 개발 과정의 가지는 위험 요소

- 처음부터 완벽하게 설계하고, 철저히 단계를 따라가며 개발하기 어렵다.
- 사용자의 요구사항이 처음부터 명확하지 않고, 변경될 수 있다.
- 소스 코드의 품질이 떨어지거나, 버그를 찾는 것이 어려울 수 있다.

TDD 방식의 개발 과정

- 테스트 코드를 작성한 뒤에 실제 코드를 작성한다. (**테스트 케이스** 필요)
- 아래의 과정을 반복하며, 코드가 점진적으로 개선된다.
- **단위 테스트**에 기반하여, 종속성 및 의존성이 낮은 모듈화를 기대할 수 있다.

→ 코드의 모듈화 효과 및 높은 재사용성을 기대할 수 있다.

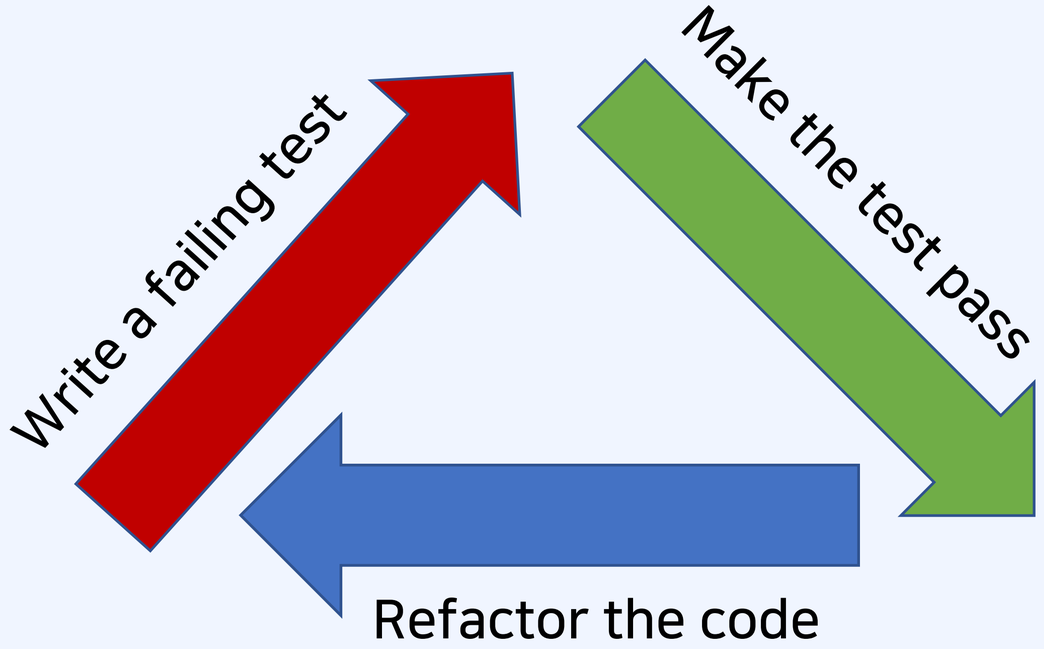


단위 테스트(Unit Test)

- 일반적으로, **하나의 단위 클래스(class)에 대한 기능 테스트**를 의미한다.
- 자바의 경우 JUnit과 같은 라이브러리를 손쉽게 사용할 수 있다.
- 단위 테스트를 통해, 어떤 부분에 오류가 있는지 알 수 있어 오류가 존재하는 클래스에서부터 시작하여 오류를 해결해 나가기에 용이하다.

TDD 개발 주기 살펴보기

- 테스트 케이스가 필요하다. (문제와 정답을 준비하듯이)
- [Red] 실패하는 테스트 코드를 간단히 먼저 작성한다.
- [Green] 테스트를 통과하기 위한 실제 코드를 작성한다.
- [Blue] 코드 중복 제거, 일반화 등 리팩토링 과정을 거친다.



TDD 개발 방식의 특징

- 일반적으로 언급되는 단점으로는 생산성 저하(개발 속도 저하)가 있다.

→ 계속해서 테스트 과정을 거치며 코드를 고쳐 나간다.

- 해당 과정에 익숙하지 않은 개발자에게 TDD 방법을 체득하기 위한 시간이 필요하다.