TDD 리뷰 및 iOS 적용

지난 시간 Android TDD 개발방법론, MVP 패턴, model(junit), view(espresso), presenter(mockito)관련 Live 코딩, 로그인 기능에 적용

Test Driven Development : 테스트 주도개발(테스트가 개발을 이끌어 간다는 이념)

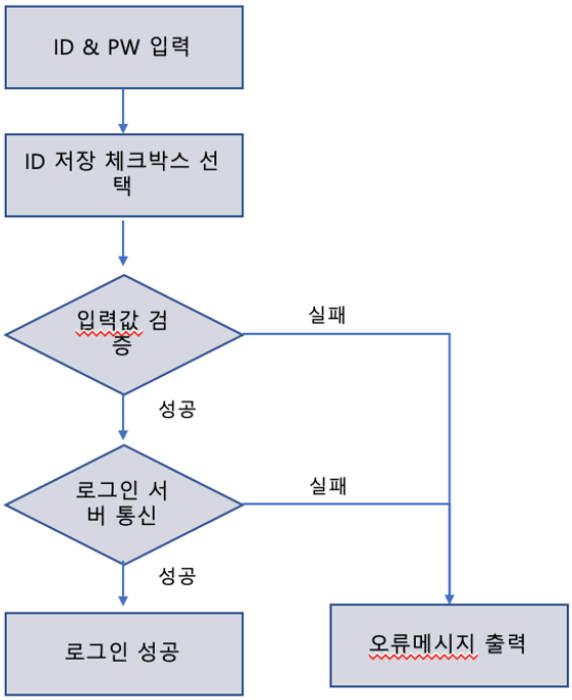
- 업무코드를 짜기전에 테스트코드를 먼저 만드는것, 테스트 코드는 업무코드에 그대로 적용

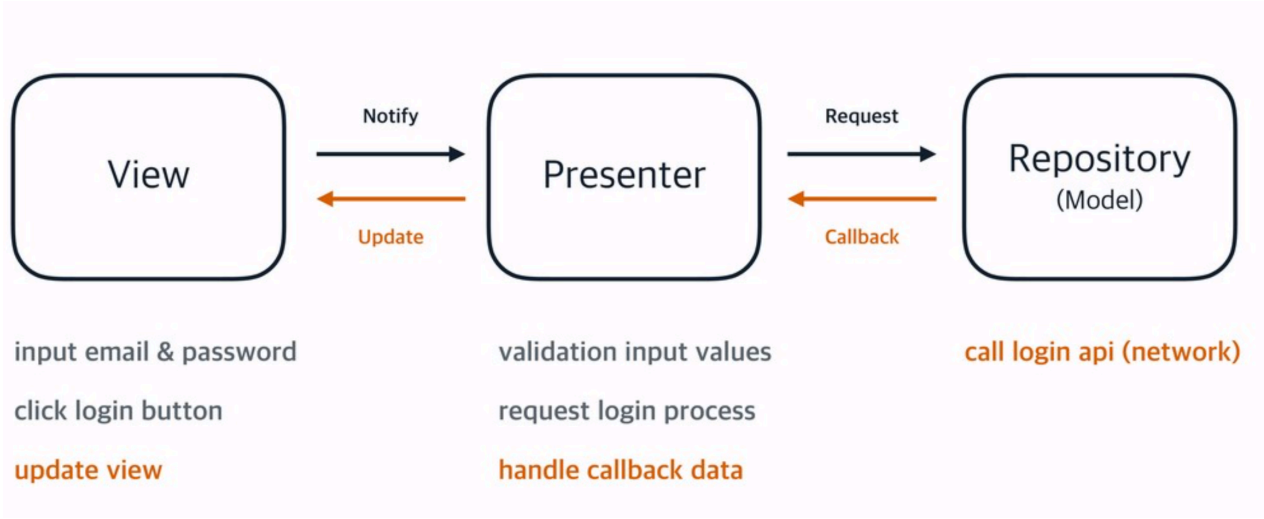
일반적으로 TDD에서 말하는 단위테스트는 메소드 단위의 테스트로서 테스트 케이스를 먼저 만들고 테스트를 통과하기 위한 코딩을 하는 것으로 TDD를 적용하는 것은 높은 수준의 프로그래밍 능력을 요구한다.

실패하는 테스트 케이스를 만들고 그것을 성공하도록(implement) 수정 한다. 설계에 대한 정답은 없지만 새로 구연된 코드는 기존 코드와 합쳐져 리팩토링을 해야한다. 기본적인 디자인패턴 지식과 예쁜? 코드를 만들도록 기술을 익혀야 한다.

1. TDD를 하면 코드 복잡도가 떨어진다.
2. 엔트로피(Entropie)가 낮아진다.
3. 잘 동작하는 깔끔한 코드 가 나온다.
4. 유지보수 비용이 낮아진다.

로그인 기능에 대한 TDD 적용





View

- 에러 팝업띄우기, 서버재설정 팝업 띄우기
- 서버 설정 재시작 시 어플리케이션 재시작
- 입력필드 초기화
- 로그인 완료시 메인(메뉴)화면으로 이동
- ID저장 체크 값 가져오기
- ID/PW/CompanyKey 가져오기, 세팅하기
- 입력 키패드 내리기
- 로딩 다이얼로그 팝업(프로그래스바) 보이기/ 숨기기

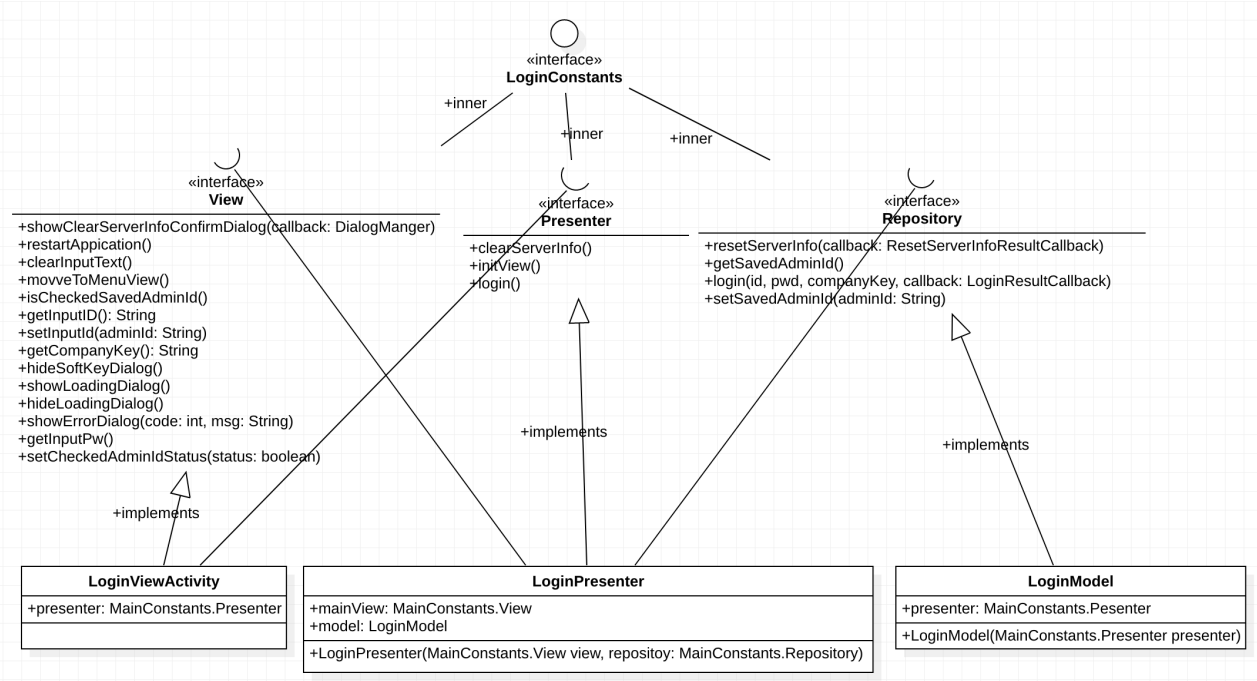
Presenter

- 로그인 로직
- ID 저장 상태 값에 따른 로직
- 서버 재설정 시 로직 처리

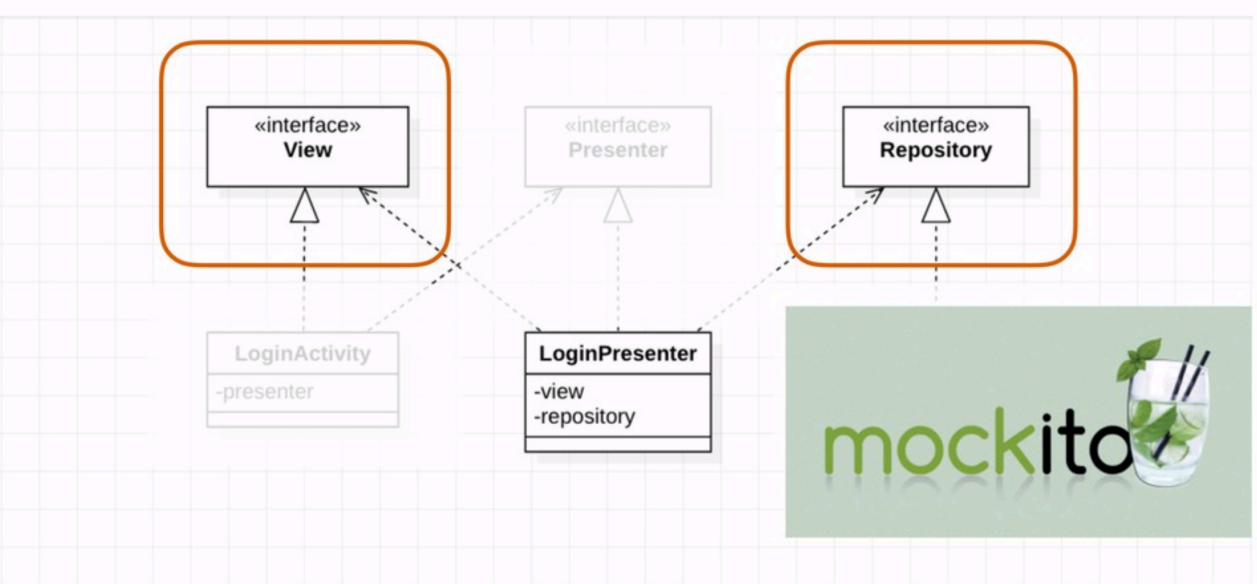
Repository

- 서버 재설정 시 값 초기화
- 어드민 ID 저장/가져오기
- 로그인 서버통신

MVP 로그인 ClassDiagram(완성)



- **Mockito 적용 (view, repository)하여 TDD 실습**



iOS Unit 테스트 - Local Unit Testing

- 관련 라이브러리: XCTest, OCMockito, OCHamcrest
  - 에뮬레이터나 실제 단말로 테스트

OCMockito, OCHamcrest

- OCHamcrest.framework
    - 지정된 객체가 기준과 일치하는지 여부에 대한 규칙을 선언 할 수있는 "매칭"객체 라이브러리
- OCMockito.framework
    - Mock 객체의 생성 및 특정 메서드 호출에 대해 미리 결정된 값을 반환(verify 함수로 검증)하도록 설정된 객체 라이브러리
    - 즉, mock 객체에 대한 원하는 메소드가 특정조건으로 실행되었는지 검증