MVC (Model + View + Controller)

Model : 애플리케이션에서 사용되는 데이터와 그 데이터를 처리하는 부분

View : 사용자에서 보여지는 UI

Controller : 사용자의 입력을 받고 처리하는 부분

동작 :

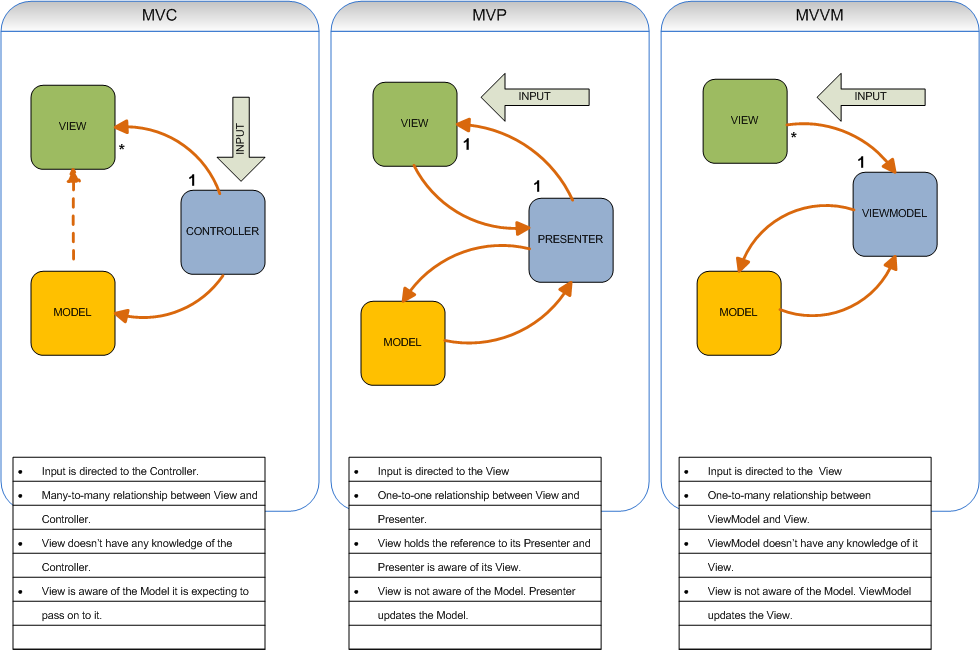
1. 사용자의 액션이 Controller로 들어온다.
2. Controller는 사용자의 액션을 인식하고 Model을 업데이트 한다.
3. Controller는 Model을 나타낼 View를 선택한다.
4. View는 Model을 화면에 나타낸다.

Controller는 여러 개의 View를 선택할 수 있다.

View는 Controller를 알지 못한다.

Controller는 View를 선택만 하고 업데이트 하지 않는다.

단점 : View와 Model 간의 의존성이 높다.



MVC (Model + View + Presenter)

Presenter : View에서 요청한 정보로 Model을 가공하여 View에 전달해 주는 역할, View와Model을 연결해준다.

동작 :

1. 사용자의 액션이 View를 통해 들어온다.
2. View는 Model의 데이터를 Presenter에게 요청한다.
3. Presenter는 Model에게 데이터를 요청한다.
4. Model은 Presenter에서 요청 받은 데이터를 응답한다.
5. Presenter는 Model의 결과를 바인딩하여 View에게 통보한다.
6. View는 Presenter가 응답한 데이터로 화면을 나타낸다.

Presenter와 View는 1:1 관계이다.

View와 Model 간의 의존성을 없앤다. (Presenter를 통해서만 데이터를 전달받기 때문)

Model과 View 는 1:1 관계이다.

View는 Model을 알지 못한다.

단점 : View와 Presenter 간의 의존성이 높다.

MVVM (Model + View + ViewModel)

ViewModel : View를 표현하기 위해 만든 View를 위한 Model로, View를 나타내기 위한 데이터를 처리하는 부분이다.

동작

1. 사용자의 액션은 View를 통해 들어온다.
2. Command 패턴으로 ViewModel에 액션을 전달한다.
3. ViewModel은 Model에게 데이터를 요청한다.
4. Model은 ViewModel에게 요청받은 데이터를 응답한다.
5. ViewModel은 응답받은 데이터를 가공해 저장한다.
6. View는 ViewModel과 DataBinding을 하여 화면에 나타낸다.

MVVM은 Command패턴과 DataBinding을 이용하여 View와 ViewModel 사이의 의존성을 없앴다.

1. Command패턴 : 요청을 객체의 형태로 캡슐화하여 나중에 이용할 수 있도록 정보를 저장, 로깅, 취소할 수 있게 한 패턴이다. (Client, Command, Reciever, Invoker)
2. DataBinding : 데이터를 provider와 consumer간의 bind하고 synchronize한다.

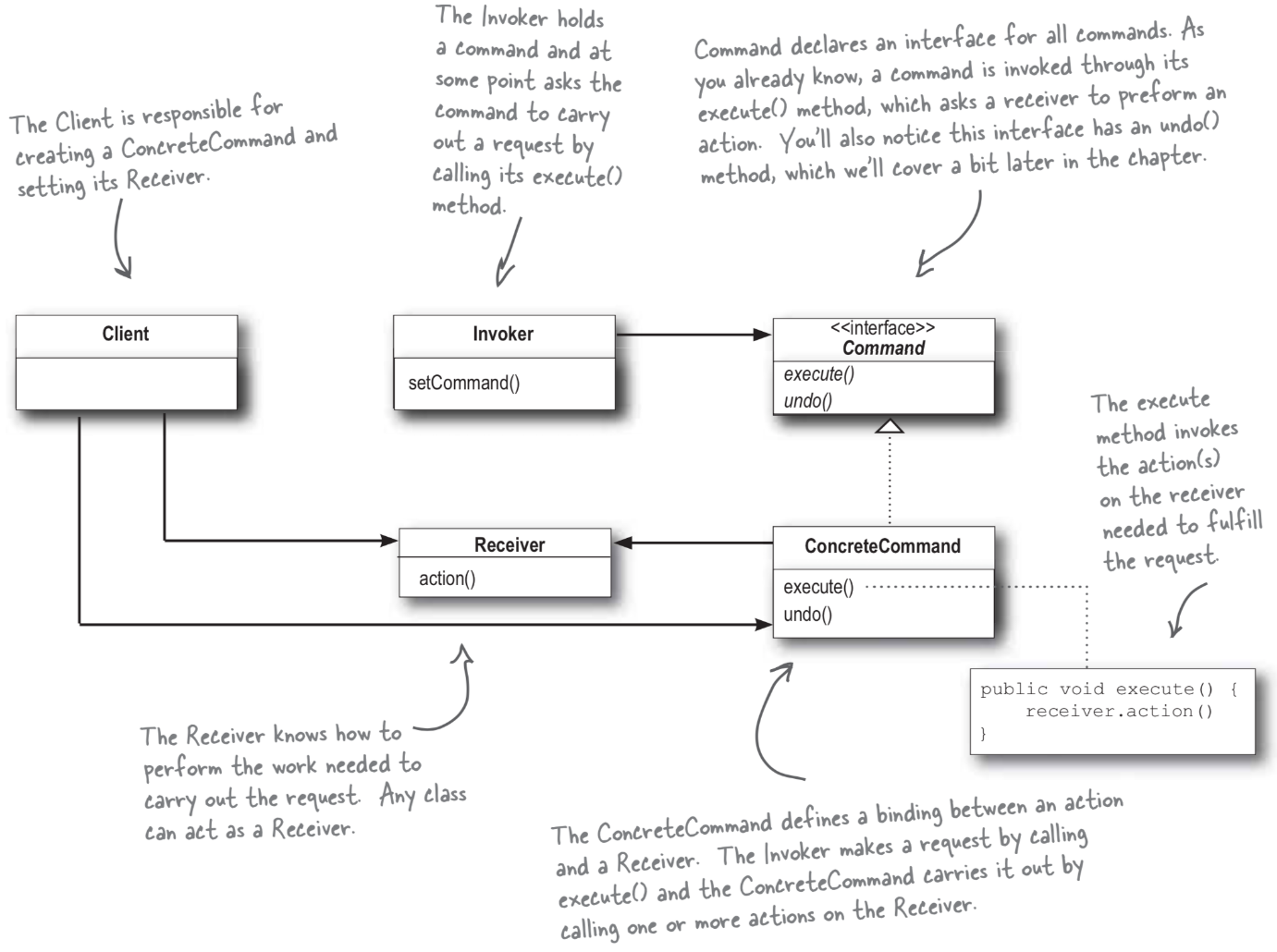
ViewModel과 View는 1:n 관계이다.

UI와 View를 Activity로부터 분리하기 위해 DataBinding을 사용한다.

View와 Model 사이에 의존성이 없다. 독립적이기 때문에 모듈화 개발에 유리

단점 : ViewModel의 설계가 어렵다.

[ Command 패턴 ]



[[디자인패턴] 커맨드 패턴 | Command Pattern | 안드로이드 예제 (tistory.com)](https://codingstorywithme.tistory.com/37)

public void onClick(View v) {

switch(v.getId()) {

case R.id.buttonA:

1 nowCommand = new LightOnCommand(new Light(this));

2 simpleRemoteControl.setCommand(nowCommand);

3 simpleRemoteControl.buttonWasPressed();

usedCommandList.add(nowCommand);

break;

1. Light는 on(), off()메서드를 가진 class
2. LightOnCommand는 클래스를 arg로 받아서 객체화, execute()시 on()을 실행함. setCommand는 인스턴스를 arg로 받아서 객체화한다.
3. buttonWasPressed()는 execute()를 해줌으로서 LightOnCommand의 on()메서드를 실행한다.

ButtonA -> buttonWasPressed -> execute -> lighton

[ DataBinding ]

간단하게 xml 파일에 data를 연결(binding)해서 사용하는 것

Activity에서 따로 View들을 정의하지 않고 Data를 View에 연결시켜두면 Data가 변할 때 따로 세팅하지 않아도 된다.

[[DataBinding] 안드로이드 DataBinding 기본 사용법 - 기본 예제 :: 시계는 와치 (tistory.com)](https://dev-imaec.tistory.com/37)