Architecture pattern.

장혜선

21.10.28

목 차

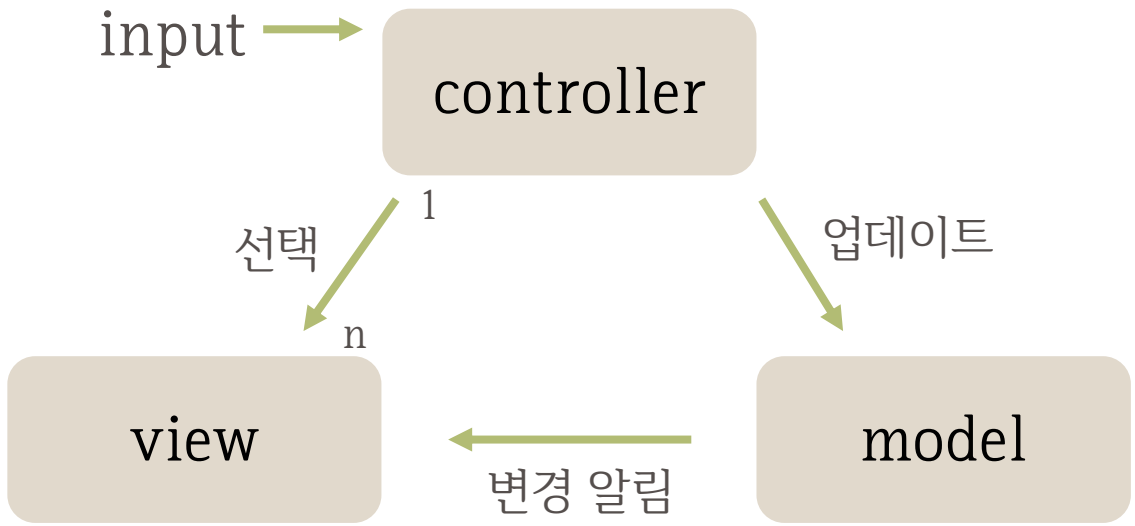
1. 아키텍처 패턴

소프트웨어 아키텍처 :
시스템을 구성하는 구성요소 간의 관계를 관리하는 시스템 구조

아키텍처 패턴 :
일반적으로 발생하는 문제점들을 일반화하고 재사용 가능한 솔루션

- 의존성
- 유지보수
- 협업 향상

2. MVC



model

사용되는 실제 데이터와 그 데이터를 처리

View와 controller에 묶이지 않아 재사용 가능

view

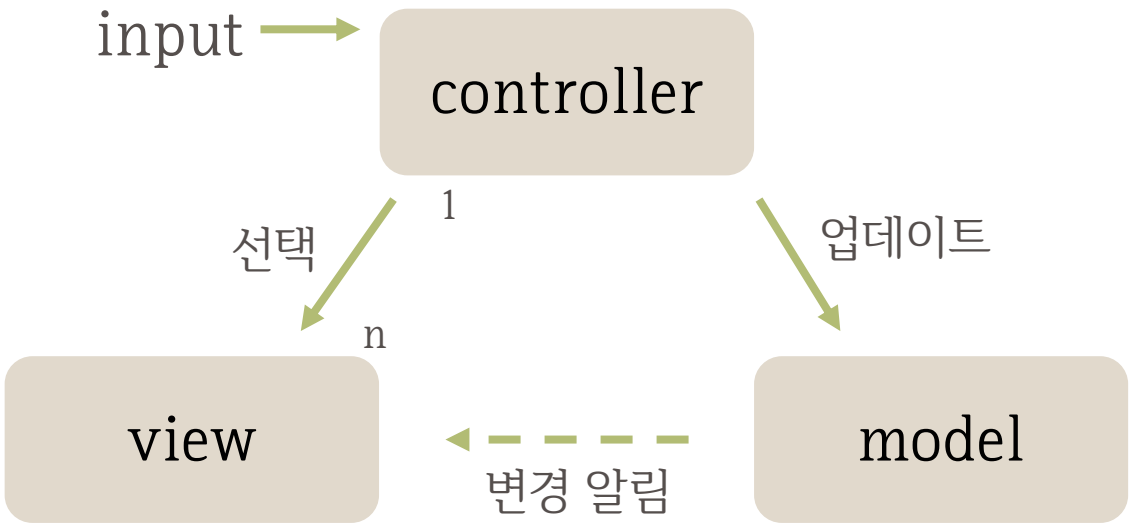
사용자에게 보여지는 화면(UI)

controller

사용자의 입력받고 처리

모델의 데이터 변화에 따라 뷰를 선택

2-1. MVC 동작 과정

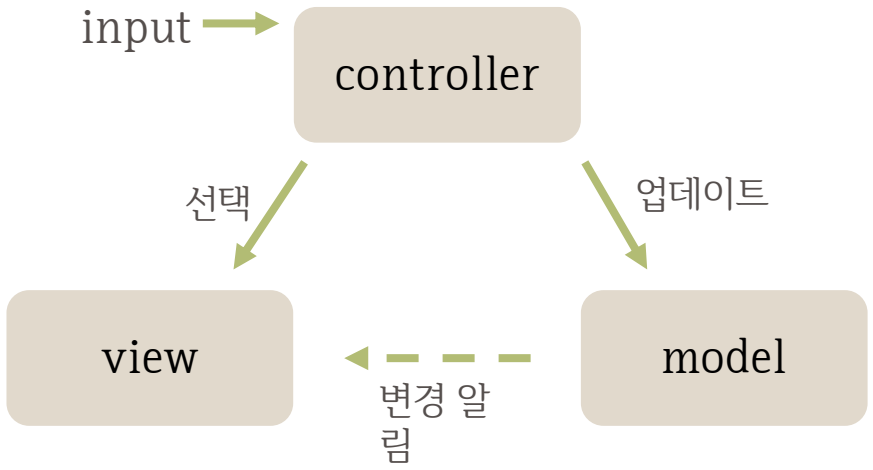


1. 사용자 입력이 컨트롤러에 들어옴
2. 컨트롤러는 입력 확인하고 모델 업데이트
3. 컨트롤러는 데이터를 나타낼 뷰 선택
4. 뷰는 업데이트를 해 화면을 나타냄

2-2. View 업데이트 방법

1	view가 model을 이용해 직접 업데이트
2	model에서 view에게 알려서 업데이트
3	view가 주기적으로 model 변경 감지

2-3. MVC 특징



- controller로 입력 들어옴
- 일대다 관계 - controller(1) : view(n)
- View는 controller를 모름
- View 는 Model 의 변화에 대해 직접적으로 알지 못함

2-4. MVC 장단점

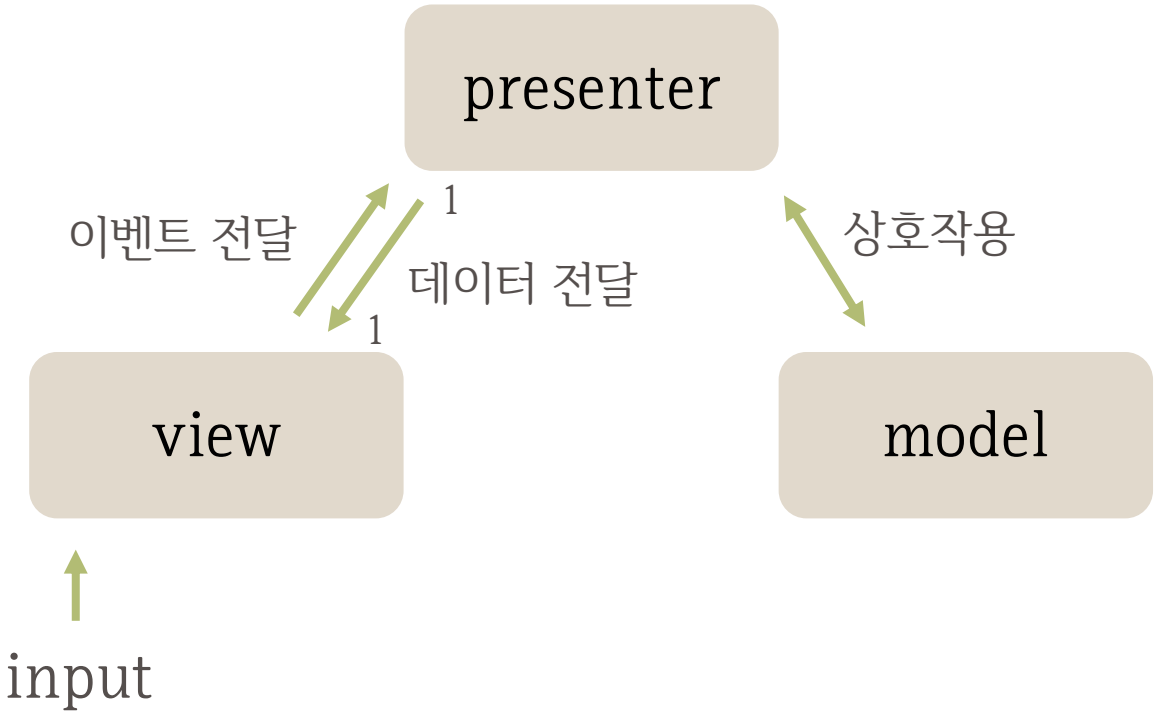
장

- Model과 view 분리
- Model 재사용 가능
- 구현하기 쉽고 단순한 구조

단

- 뷰 업데이트를 위해 모델을 직/간접적으로 참조
  -> 의존성 발생
- 시간이 지날수록 controller 비대화

3. MVP



model

사용되는 실제 데이터와 그 데이터를 처리

독립적 영역

view

사용자에게 보여지는 화면(UI)

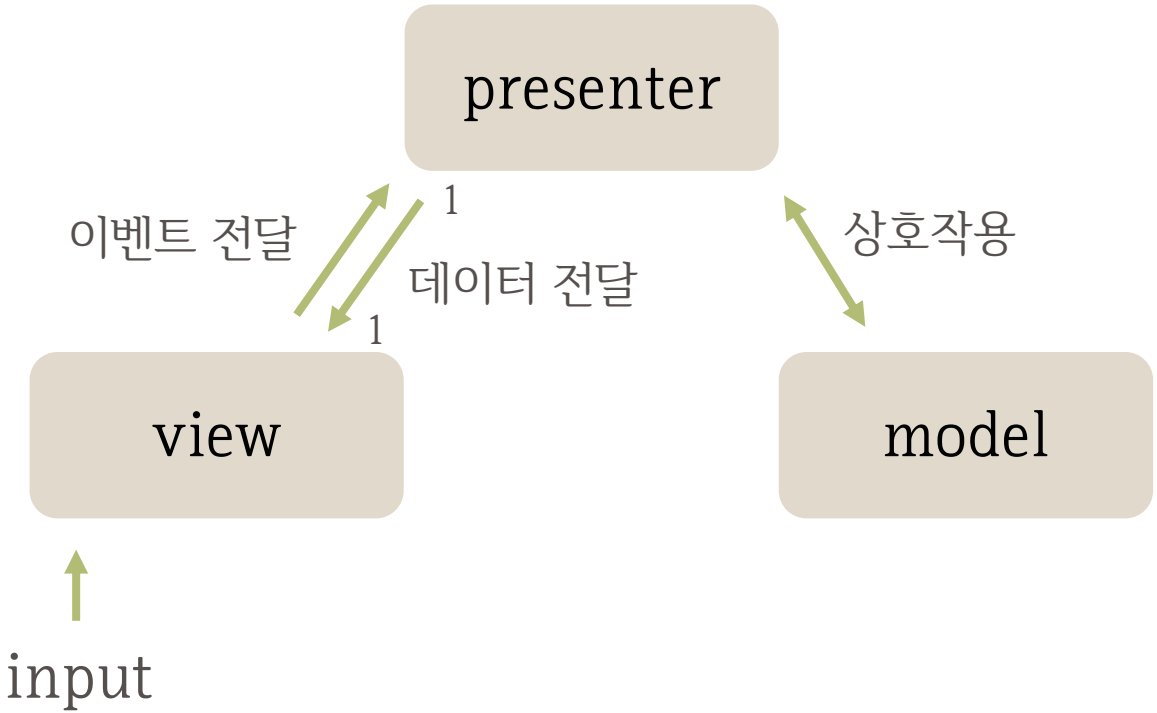
사용자의 입력을 받고 이벤트를 프리젠터로 전달

presenter

view에서 요청한 정보로 Model을 가공해 view에 전달

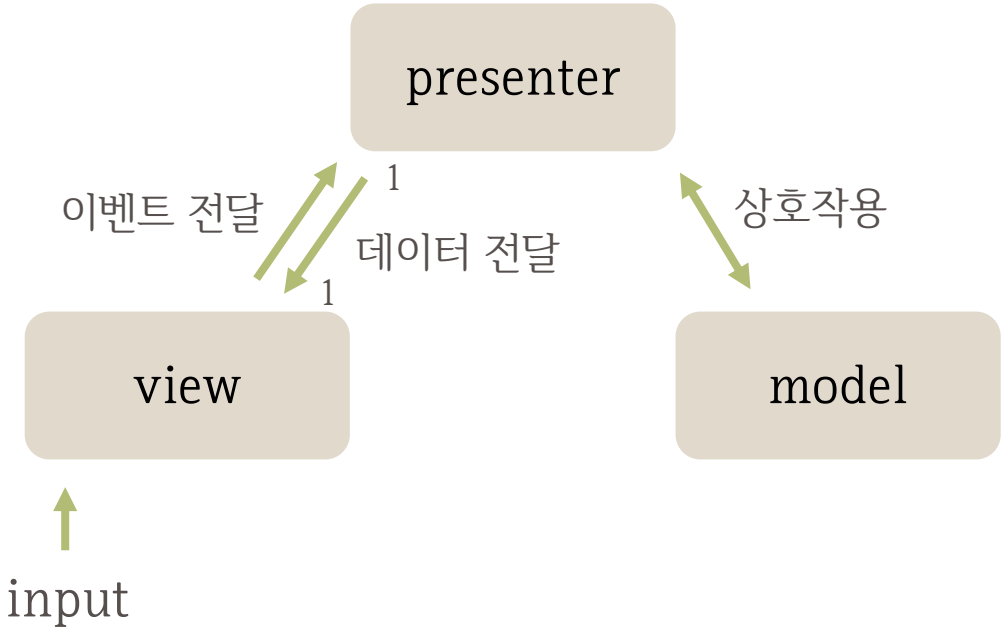
view와 model 중간다리

3-1. MVP 동작 과정



1. 사용자 입력이 뷰에 들어옴
2. 뷰는 프리젠터로 이벤트 전달
3. 프리젠터는 모델에 데이터 요청
4. 모델은 프리젠터에게 응답
5. 프리젠터는 뷰에게 데이터 응답
6. 응답받은 데이터로 뷰는 화면에 표시

3-2. MVP 특징



- view로 입력 들어옴
- 일대일 관계 - presenter(1) : view(1)
- view와 presenter의 의존성이 강함
- presenter는 view에게 표시할 내용만 전달
- view와 model은 서로 존재를 모름

3-3. MVP 장단점

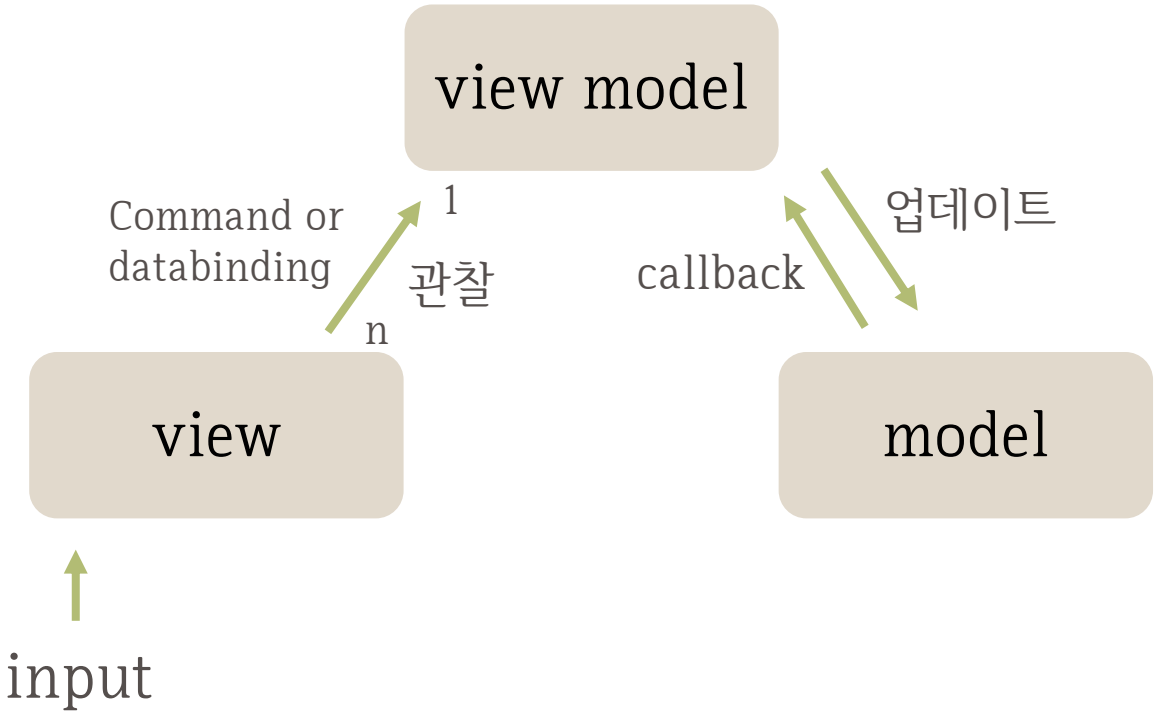
장

- model과 view 의존성 X

단

- presenter-view 의존성 높음
- 시간이 지날수록 presenter 비대화

4. MVVM



model

사용되는 실제 데이터와 그 데이터를 처리

view

사용자에게 보여지는 화면(UI)

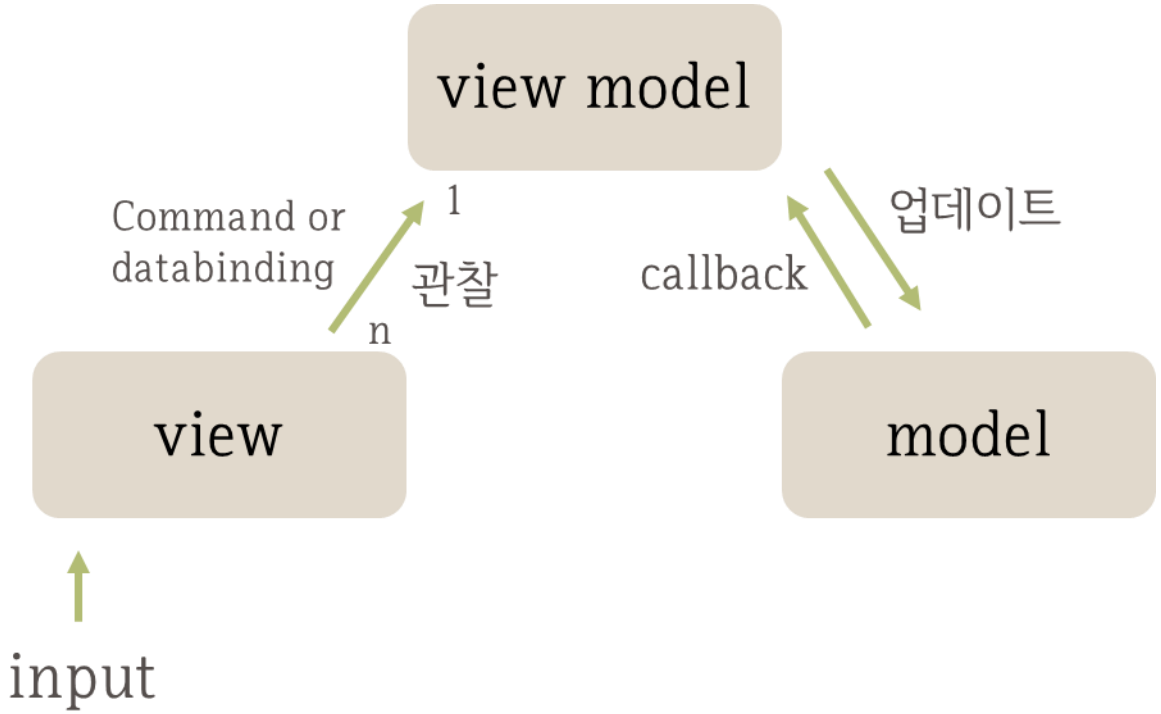
view model을 관찰

view model

view를 표현하기 위해 만듦

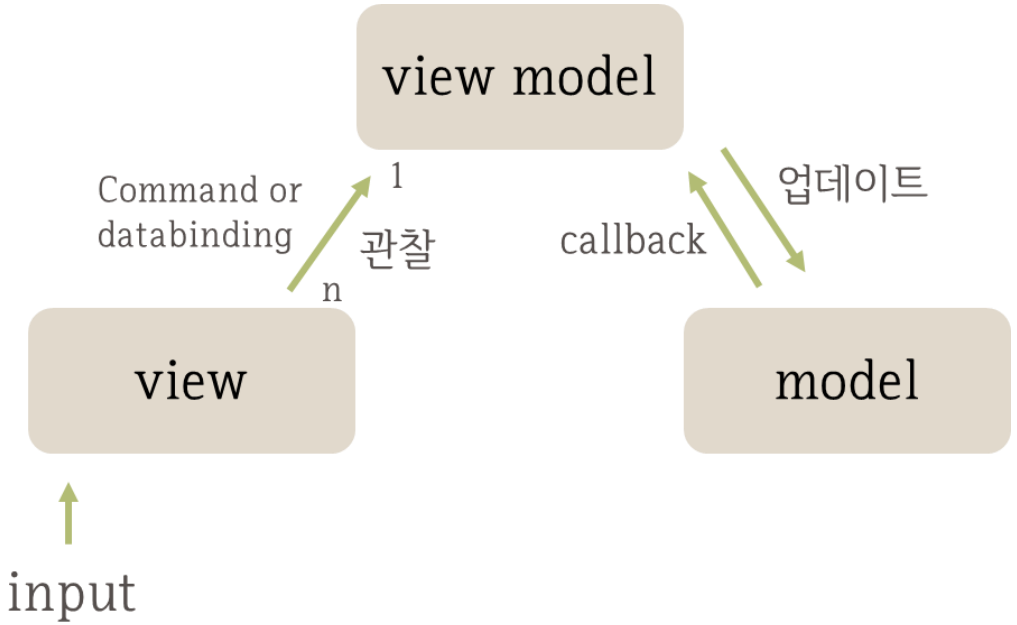
UI 관련 데이터 보관, 관리

4-1. MVVM 동작 과정



1. 사용자 입력이 뷰에 들어옴
2. 뷰는 뷰모델에 명령을 내림
3. 뷰모델은 필요한 데이터 모델에 요청
4. 모델은 필요한 데이터 응답
5. 뷰모델은 응답 받은 데이터 가공해 저장
6. 뷰는 데이터 바인딩으로 인해 자동 업데이트

4-2. MVVM 특징



- view로 입력 들어옴
- 일대다 관계 - viewmodel(1) : view(n)
- command와 databinding 사용
- viewmodel-view 의존성 X
- view-model 의존성 X

4-3. MVVM 장단점

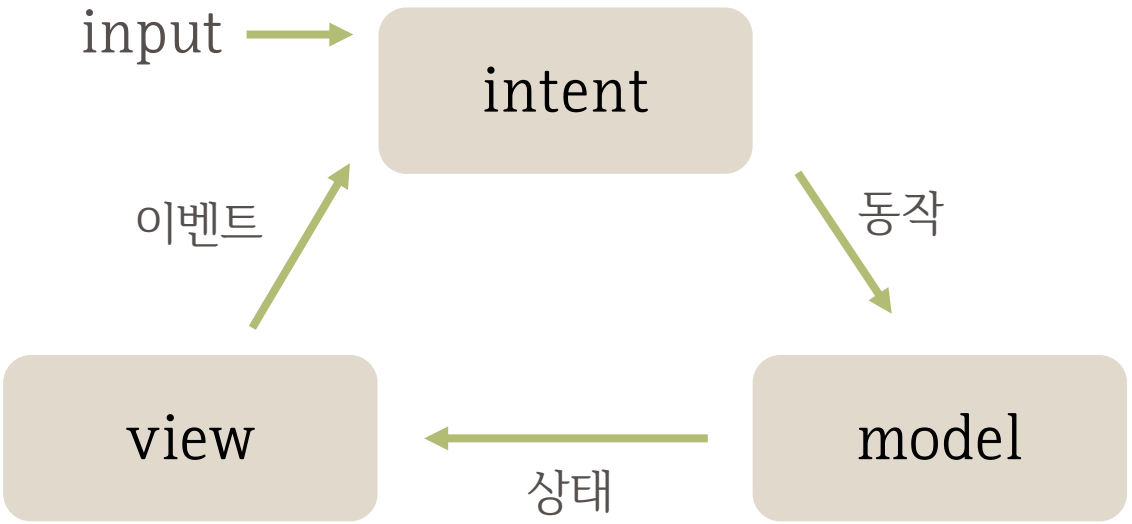
장

- model과 view 의존성 X
- view-viewmodel 의존성 X
- 중복 코드 모듈화 가능

단

- 설계가 어려움

4. MVI



model

데이터를 가짐, UI 상태를 나타냄

데이터 흐름 단방향 -> 불변성 필요

view

새로운 상태를 받아 화면에 표시

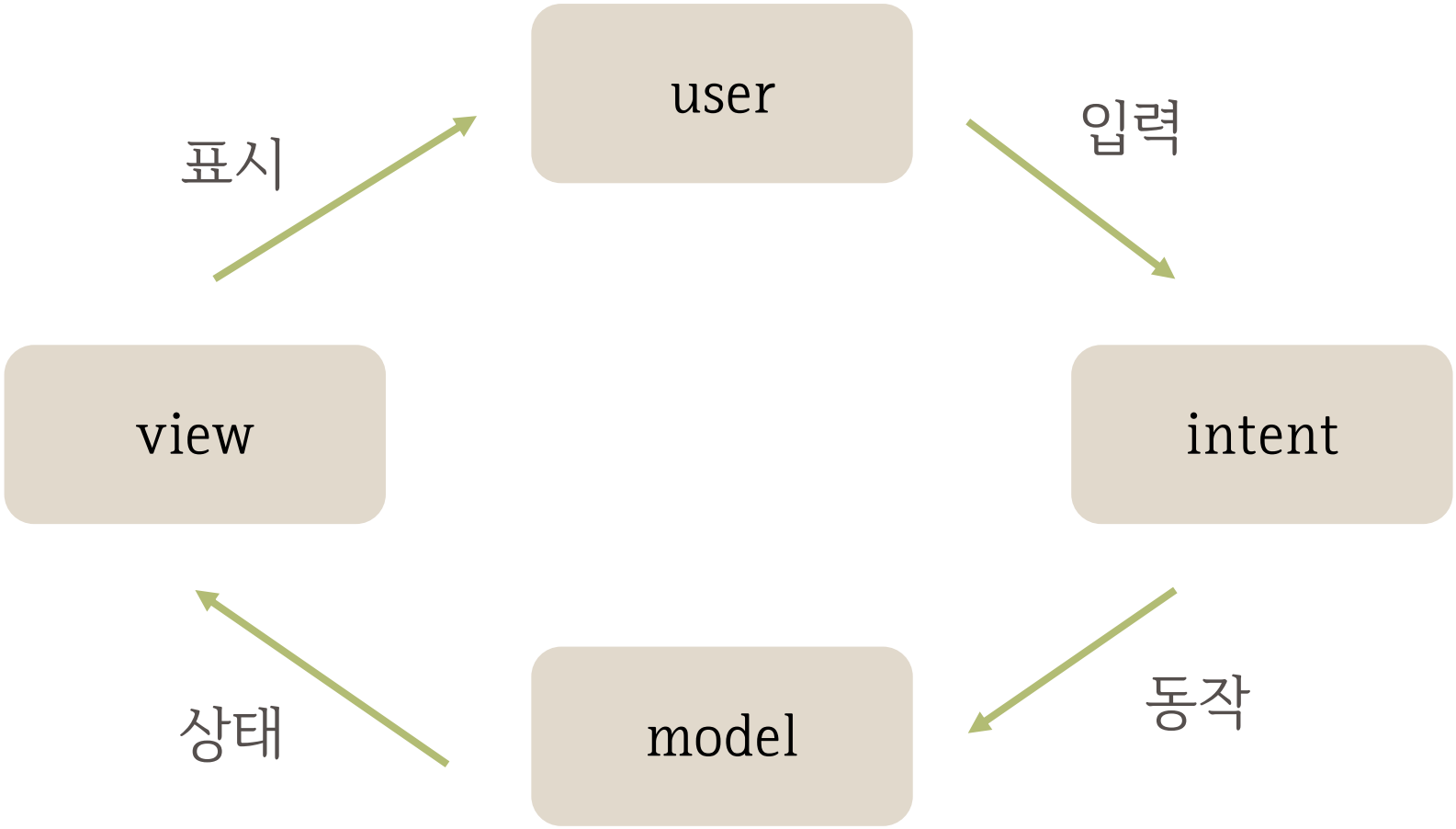
상호작용에 대한 결과 전달

intent

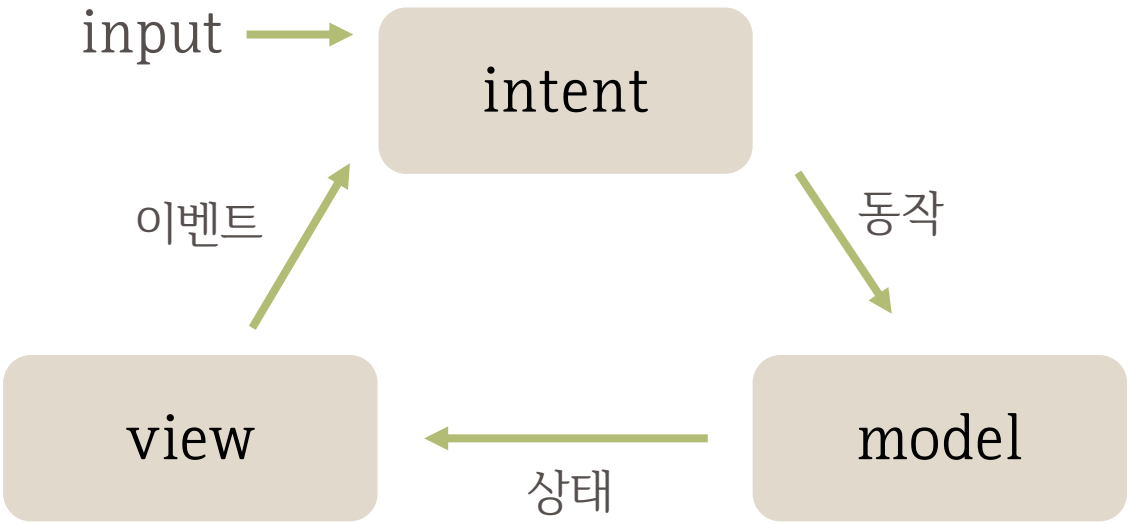
의도를 의미

UI 변화

4-1. MVI 동작 과정



4-2. MVI 특징



- intent로 입력 들어옴
- 단방향 흐름
- 사이클
- 의도와 상태가 추가됨
- 모델이 데이터만 다루지 않음

4-3. MVI 장단점

장

- 예측 가능한 상태
- 유지 보수 용이
- 서로 간 의존성 X

단

- 외부 라이브러리 사용
- 많은 상용구 코드
- 학습 곡선이 높음

감사합니다.