iOS Seminar 3

한상현

오늘 배울 내용

- 과제1, 과제2 피드백
- UICollectionView : 행 열 형태의 표 형태의 View
- RxSwift (RxDataSource)
- RxSwift를 효과적으로 MVVM에 적용하는 방법론
- UITabbarController

과제1, 2 피드백

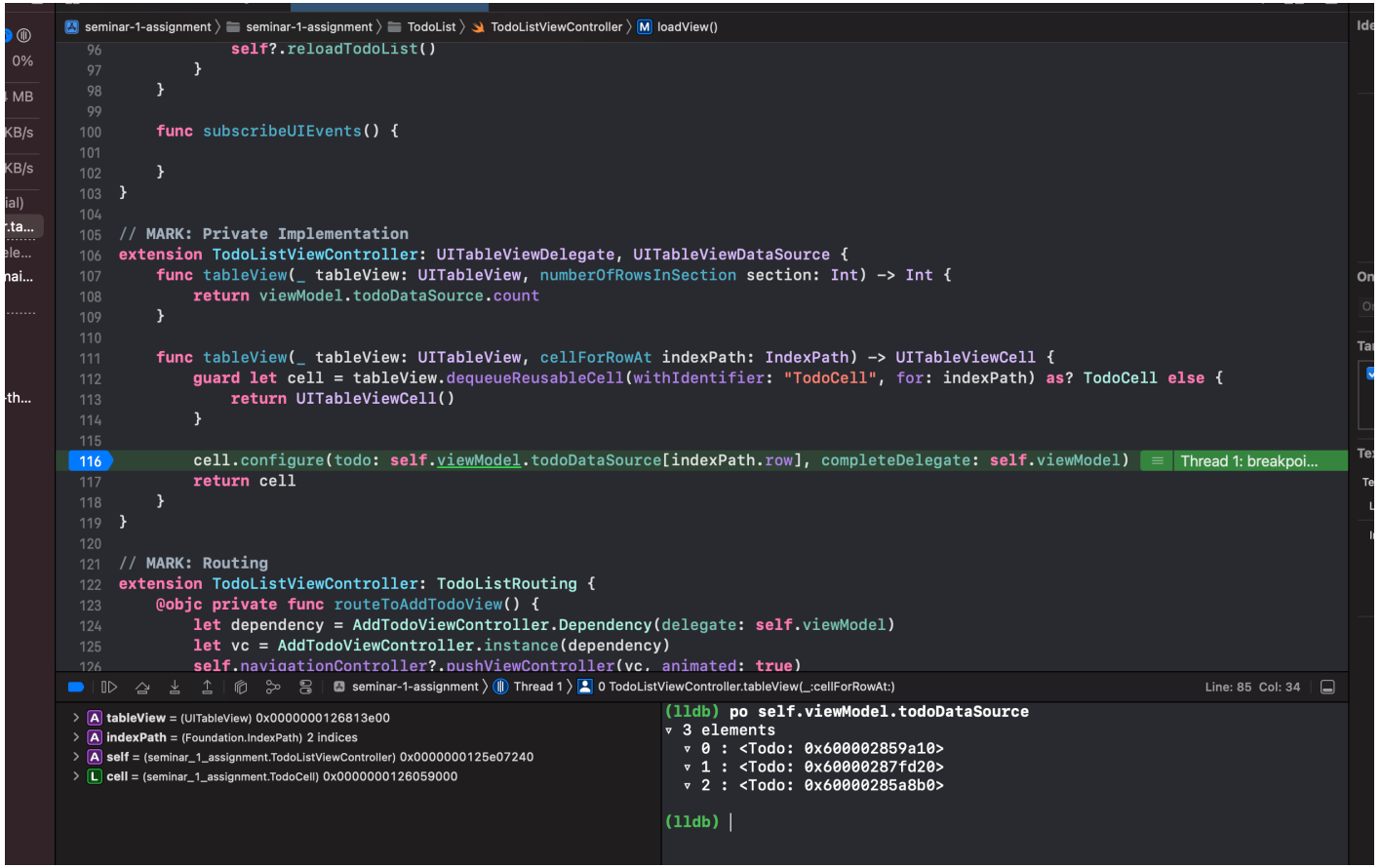
- 타이밍 이슈
- 디버깅의 중요성
- 스펙에 대한 개발자의 마인드셋

타이밍 이슈

- 내 데이터가 어떤 플로우로 넘어가는지를 완벽히 이해한 채 구현해야 함
- 이 방식에 가이드를 주는게 아키텍쳐 (MVVM, Domain, Repository, Usecase, Manager 등등의 개념)

- 뷰를 갱신하기 전에 갱신에 필요한 데이터가 다 갖춰질 수 있는 환경인지 항상 확인하자!

디버깅


seminar-1-assignment 〉 seminar-1-assignment 〉 TodoList 〉 TodoListViewController 〉 M loadView()
96 self?.reloadTodoList()
97 }
98 }
99
100 func subscribeUIEvents() {
101
102 }
103 }
104
105 // MARK: Private Implementation
106 extension TodoListViewController: UITableViewDelegate, UITableViewDataSource {
107 func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
108 return viewModel.todoDataSource.count
109 }
110
111 func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
112 guard let cell = tableView.dequeueReusableCell(withIdentifier: "TodoCell", for: indexPath) as? TodoCell else {
113 return UITableViewCell()
114 }
115
116 cell.configure(todo: self.viewModel.todoDataSource[indexPath.row], completeDelegate: self.viewModel)
Thread 1: breakpoi...
117 return cell
118 }
119 }
120
121 // MARK: Routing
122 extension TodoListViewController: TodoListRouting {
123 @objc private func routeToAddTodoView() {
124 let dependency = AddTodoViewController.Dependency(delegate: self.viewModel)
125 let vc = AddTodoViewController.instance(dependency)
126 self.navigationController?.pushViewController(vc, animated: true)
seminar-1-assignment 〉 Thread 1 〉 0 TodoListViewController.tableView(_:cellForRowAt:)
Line: 85 Col: 34
A tableView = (UITableView) 0x0000000126813e00
A indexPath = (Foundation.IndexPath) 2 indices
A self = (seminar_1_assignment.TodoListViewController) 0x0000000125e07240
L cell = (seminar_1_assignment.TodoCell) 0x0000000126059000
(lldb) po self.viewModel.todoDataSource
3 elements
0 : <Todo: 0x600002859a10>
1 : <Todo: 0x60000287fd20>
2 : <Todo: 0x60000285a8b0>
(lldb)

디버깅

- lldb 명령어를 몇가지 알고 있으면 수월

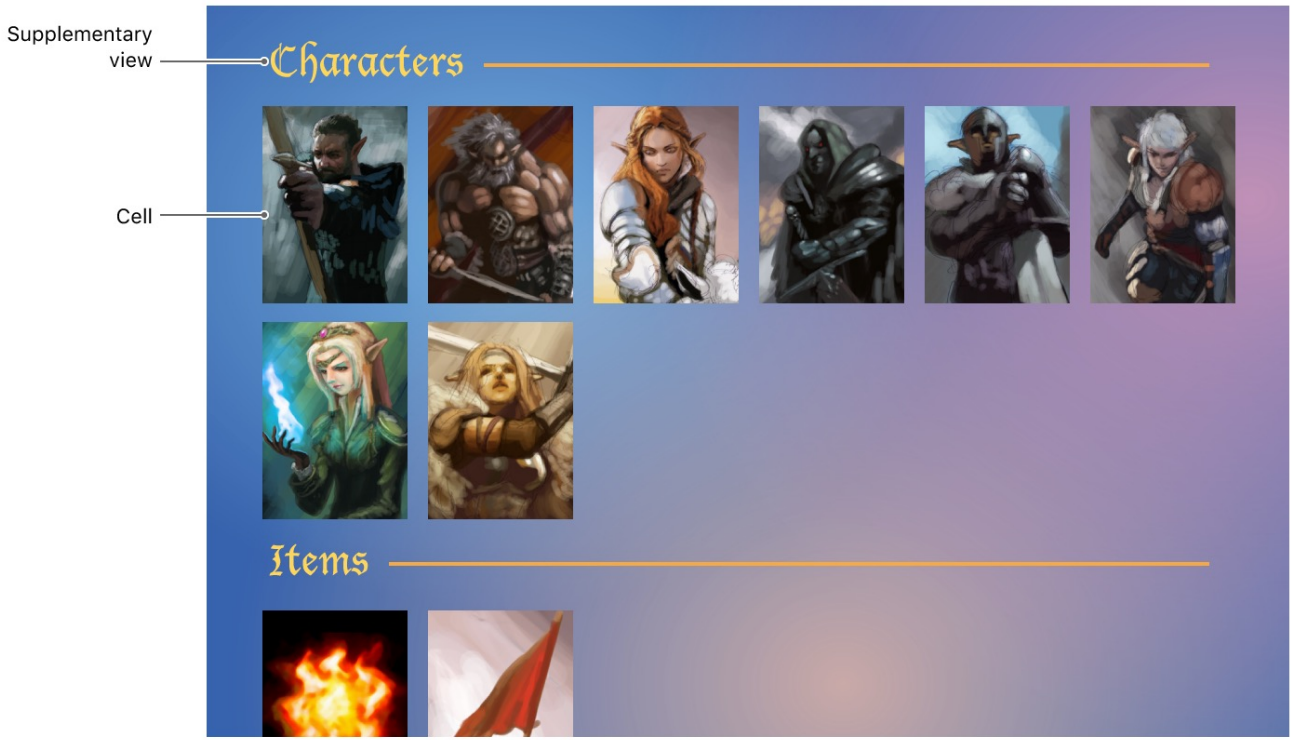
  : 사실 po 만 알아도 큰 문제는 없다

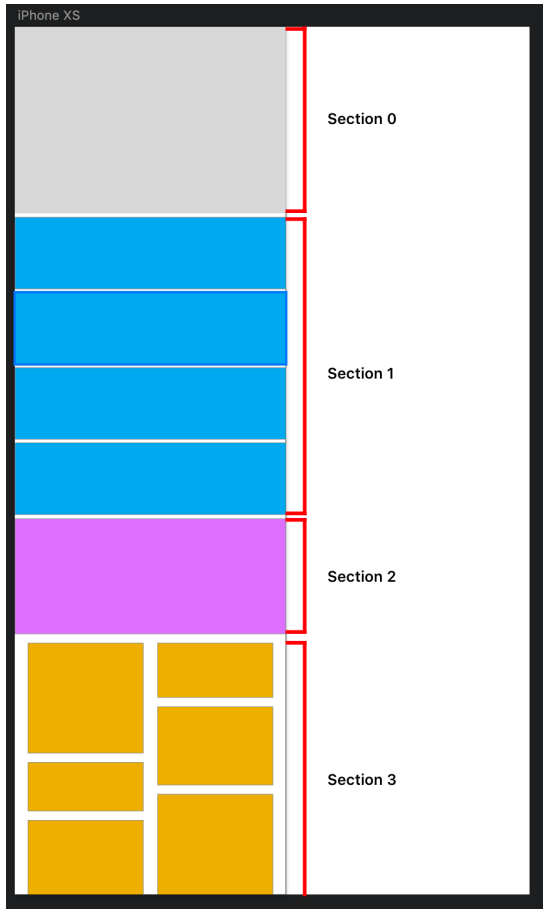
스펙

- **스펙이란건 절대적이지 않다**
  : 어차피 사람이 정하는 결정사항에 불과하고, 유저의 반응, 시장의 상황, 코드 설계 및 구현의 난이도에 따라 언제든 바뀔 수 있다.
- 협업의 관점에서 스펙
  : 회사에 따라 다르지만 개발자가 직접 기획에 참여하는 경우 / 기획자가 기획 명세를 작성해 요청하는 경우에 따라 적절한 대처가 필요
  : 혼자 다하는 경우는 상관없지만, 누군가 그 스펙을 작성해서 전달했다면 이를 수정하기 위해서는 이에 합당한 근거가 필요하다. 이런 부분에서 돈 받고 코딩하는 사람과 취미로 코딩하는 사람의 차이가 생기지 않나 생각

UICollectionView

Figure 1 A collection view using the flow layout





UICollectionViewCell

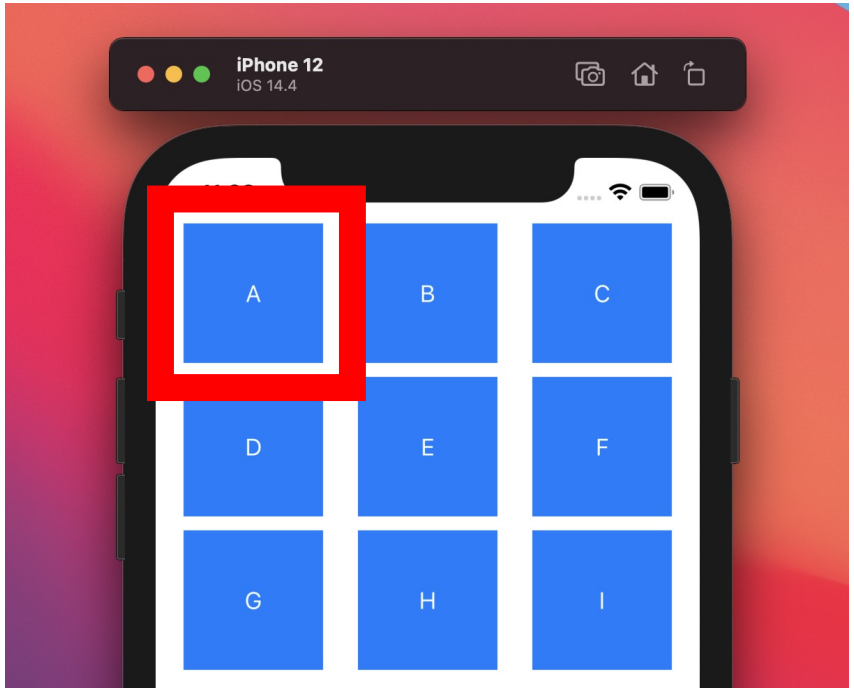
Class

UICollectionViewCell

A single data item when that item is within the collection view's visible bounds.

Declaration

```
@MainActor class UICollectionViewCell : UICollectionReusableView
```



UICollectionViewFlowLayout

- 각 셀의 크기와 배치를 정해주는 일종의 Decorator 객체
- UICollectionViewDelegate를 활용해도 비슷한 기능 구현 가능
  (이게 먼저 나왔음)

UICollectionViewFlowLayout 예시

```swift
class MonthCalendarLayout: UICollectionViewFlowLayout {
    override init() {
        super.init()

        self.minimumLineSpacing = Constants.verticalSpacing
        self.minimumInteritemSpacing = Constants.horizontalSpacing
        self.scrollDirection = .vertical
        self.sectionInset = .init(top: Constants.sectionSpacing, left: 0, bottom: 0, right: 0)
        let width = (UIScreen.main.bounds.width
                     - Constants.horizontalInset * 2
                     - Constants.horizontalSpacing * 6
                     - Constants.calendarBorderWidth * 2) / 7
        let height = Constants.cellHeight
        self.estimatedItemSize = CGSize(width: width, height: height)
    }

    required init?(coder: NSCoder) {
        fatalError("init(coder:) has not been implemented")
    }
}
```

```swift
private let collectionView = UICollectionView(frame: .zero, collectionViewLayout: MonthCalendarLayout())
```

TableView 작업하셨던 거 기억하시죠?

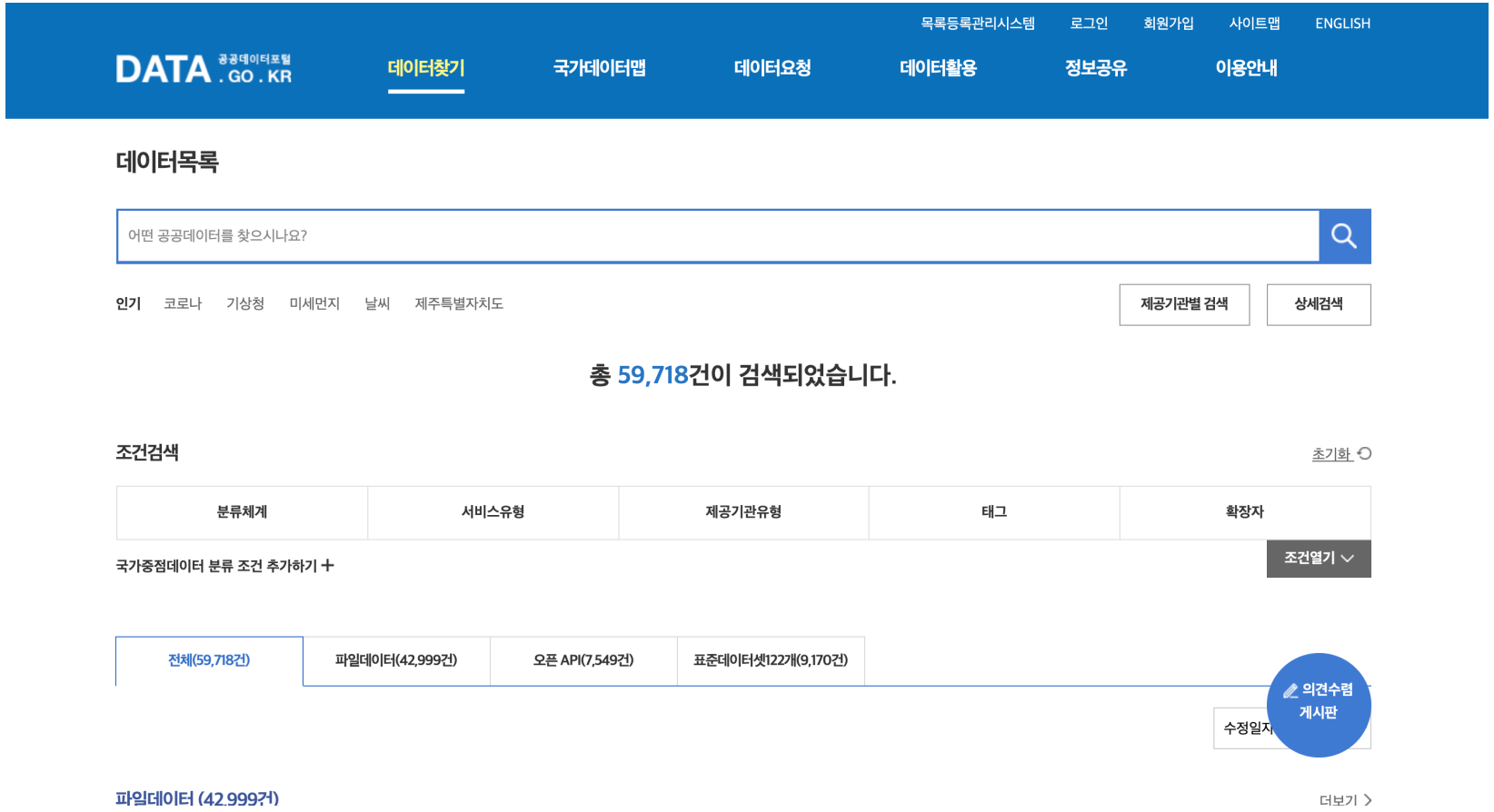
- Delegate 메소드만 다르지 원리는 기존과 똑같습니다~~

1. 몇 행 몇 열에 어떤 내용을 넣을지?
2. 특정 셀이 선택되었을 때 어떤 작업을 할지?

그 외에도 여러가지 기능들이 delegate 메소드로 구성되어 있어요

-> 자세한 건 공식문서를 참조하세요!

Open API란?


목록등록관리시스템
로그인
회원가입
사이트맵
ENGLISH
DATA 공공데이터포털 .GO.KR
데이터찾기
국가데이터맵
데이터요청
데이터활용
정보공유
이용안내
데이터목록
어떤 공공데이터를 찾으시나요?
인기 코로나 기상청 미세먼지 날씨 제주특별자치도
제공기관별 검색
상세검색
총 59,718건이 검색되었습니다.
조건검색
초기화
분류체계
서비스유형
제공기관유형
태그
확장자
국가중점데이터 분류 조건 추가하기 +
조건열기
전체(59,718건)
파일데이터(42,999건)
오픈 API(7,549건)
표준데이터셋122개(9,170건)
의견수렴 게시판
수정일자
파일데이터 (42,999건)
더보기

Cocoapods

- 지난 시간에 Alamofire를 쓰셨다면 써보셨죠??
- 이번엔 RxSwift를 쓸겁니다

우리가 이번 과제에서 사용할 pod들

- RxSwift
- RxDataSource

- Alamofire (옵션) : URLSession 사용하셔도 무관

RxSwift

- 비동기 / 반응형 프로그래밍을 위한 Third Party Library

: iOS 13부터는 first party로 애플이 Combine을 제공함

RxSwift를 이용한 UICollectionView 구현

- 핵심 : 내가 가진 데이터를 DataSource로 mapping하고 이를 collectionView에 binding 한다

-> 우리가 데이터 내용을 수정하면 별도로 reloadData를 하지 않아도 알아서 자동으로 collectionView에 반영된다

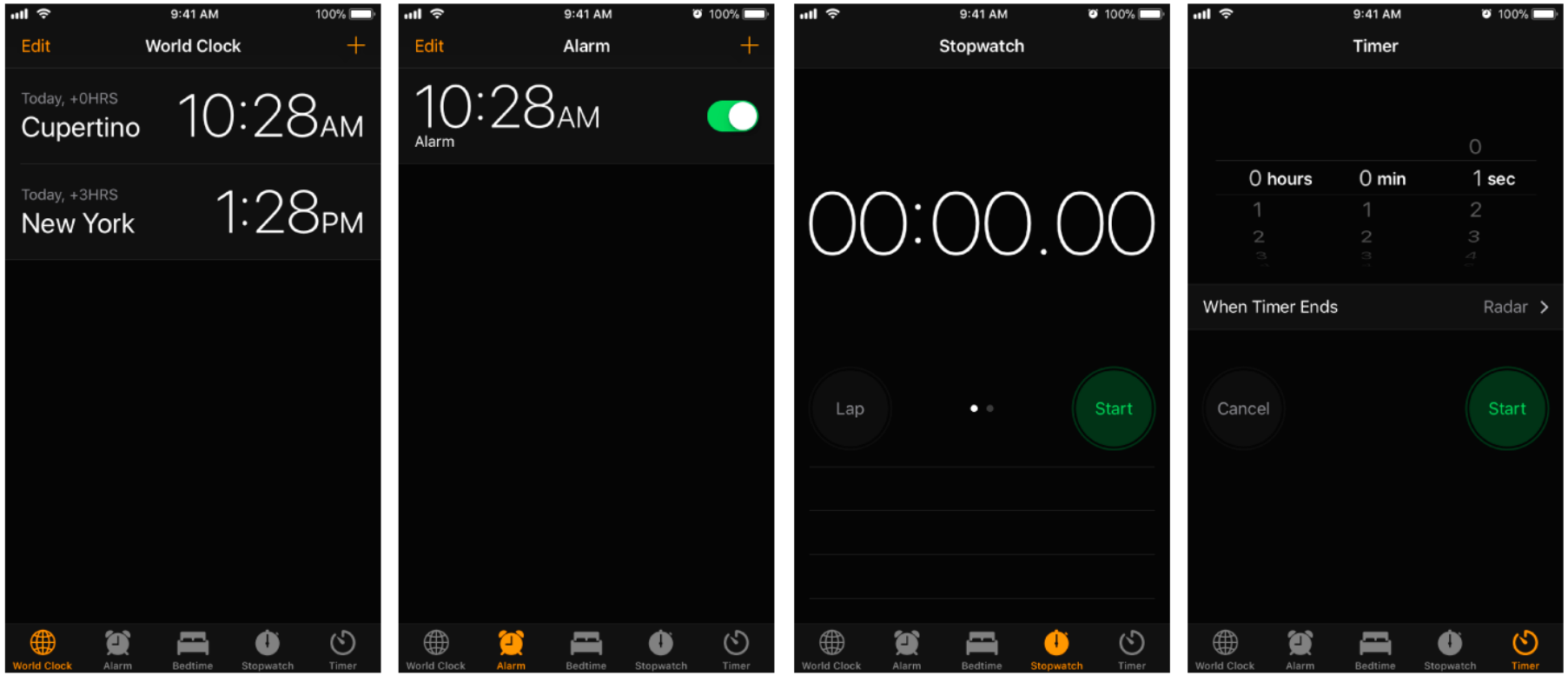
-> TableView와 완전히 같은 방식으로 동작하지만 row / column이 있다는 점이 다름

TableView/CollectionView with Rx

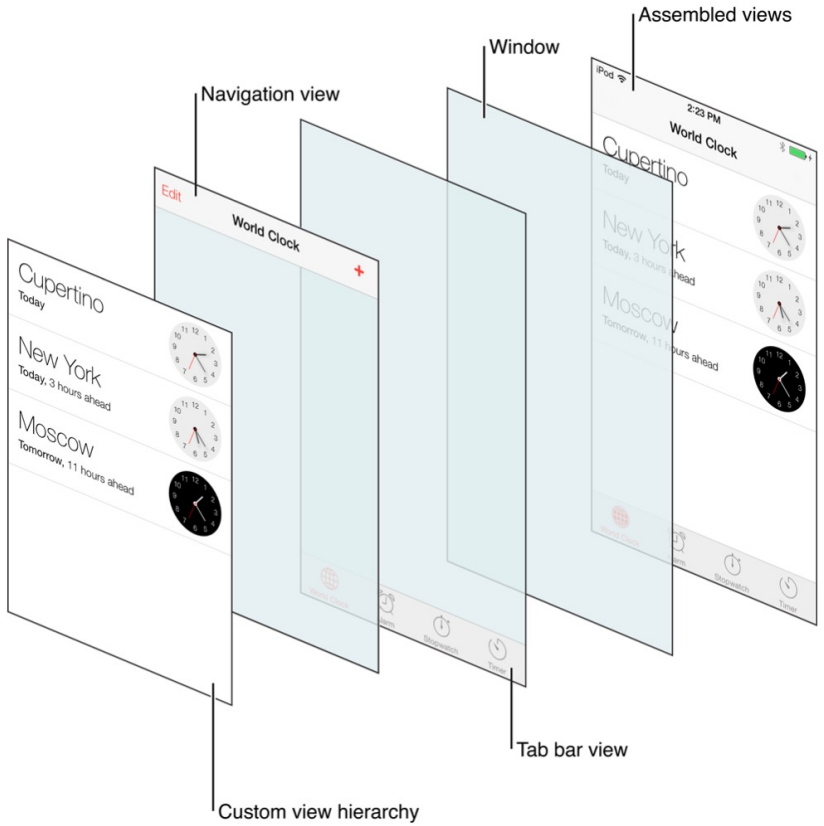
- https://eunjin3786.tistory.com/29 -> 여기에 다 소개되어 있음
- 공식 문서 : https://github.com/RxSwiftCommunity/RxDataSources

- 똑같이 생긴 Cell을 반복해서 사용할때는 SectionModel이 불필요하지만 다양한 형태의 Cell을 쓸거라면 꼭 정의해주어야 함

UITabbarController



UITabbarController



UISegmentControl

- 우리가 흔히 본 내부 탭 나누는 용도의 View Component
- 다른 탭을 탭하면 UI가 전환되면서 현재 선택되는 탭 index를 방출함
- 이를 인식하여 childViewController를 바꿔치기 해주면 된다!

Rx + MVVM

- ViewController
  1) Component들의 디자인
  2) Component들의 constraint 세팅 -> 오토레이아웃
  3) VM에 유저 액션, View Lifecycle에 맞춰 작업 요청
  4) VM으로부터 데이터를 구독 or 수신하여 유저에게 노출

Rx + MVVM

- ViewModel

  1) VC에 작업 요청을 받는 input 메소드

  2) VC에 결과를 전달하는 output variable

  3) 라우팅 로직, 결과값 수신을 위한 router와 listener, interator, presenter 등을 들고 있을 수 있음

Rx + MVVM

- Domain : Usecase, Manager

  1) VM에서 요청하는 네트워크 데이터 요청, 캐시 데이터 요청, 전역적으로 사용되는 데이터 등을 관리하고 있음

  2) 상황에 맞춰 VM에서 데이터 요청 메소드를 콜하면 적정 수준까지 wrapping하여 전달

Rx를 이용하여 MVVM 구현해보기

- 과제1 코드를 같이 바꿔보면서 감을 잡아봅시다!

Assignment 3 : 영화 소개 앱 만들기

- 메인 화면에는 두 개의 탭이 존재

1. Movie 탭

1) 헤더의 형태로 타이틀과 정렬 타입 선택이 가능한 버튼 존재 (Navigation Bar는 Hidden인 상태)

2) CollectionView는 2열로 구성되어 있으며, API 기준 1페이지에 해당하는 영화의 정보가 화면에 표시됨 (포스터 + 제목 + 평점)

3) 스크롤을 가장 아래까지 내리면 다음 페이지 데이터를 불러오는 API를 호출하여 데이터를 추가로 로드하고, CollectionView를 갱신함

4) 영화를 선택하면 Custom Animation이 적용된 NavigationController의 push로 세부 정보 뷰가 뜸 (포스터 + 제목 + 평점 + overview) (Custom Animation 구현 및 적용은 추가 과제 : 필수 아님)

5) 세부 정보 오른쪽 위에는 별 버튼이 있어 favorite 여부 체크할 수 있도록 (Favorite 체크된 영화들은 로컬에 저장하여 재실행시에도 목록이 저장될 수 있도록 구현)

Assignment 3 : 영화 소개 앱 만들기

2. Favorite 탭

1) 상단엔 타이틀 + 즐겨찾기 등록된 영화의 개수 (HeaderView)

2) CollectionView의 형태로 1탭과 똑같이 영화 목록이 뜨도록 (이 데이터는 로컬에 저장된 Favorite 영화들의 데이터

3) 영화 선택 시 세부정보 뷰가 뜨고, 별을 눌러 Favorite 해제 가능

4) 만약 해제 했을 경우 뒤로 돌아갔을 때 목록에서 그 영화가 사라져야 함

Assignment 3 - 주의사항

- 모든 뷰는 View - ViewModel 구조를 가지고 있어야 하고, 주요 데이터 처리 로직은 ViewModel - Domain 단에서 처리해야 합니다.
- 과제3부터는 스펙 체크용 테스트 케이스를 체크리스트 형태로 제공할 예정입니다. 어떠한 이유로든 모든 체크리스트가 클리어 되지 않는 경우 미통과입니다.