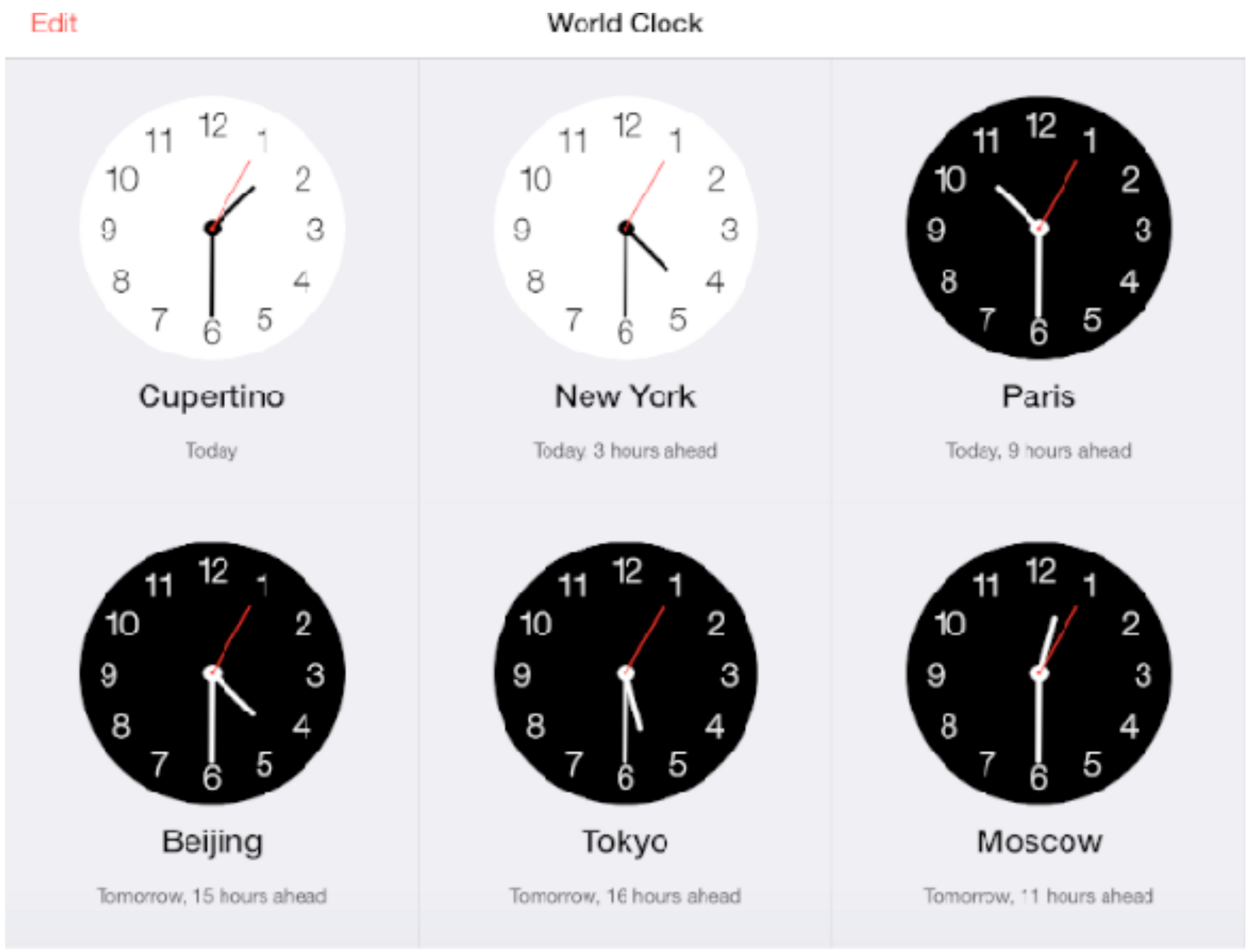
UICollection

CollectionView

- Cell
- Decoration view
- Supplementary view

UICollectionView

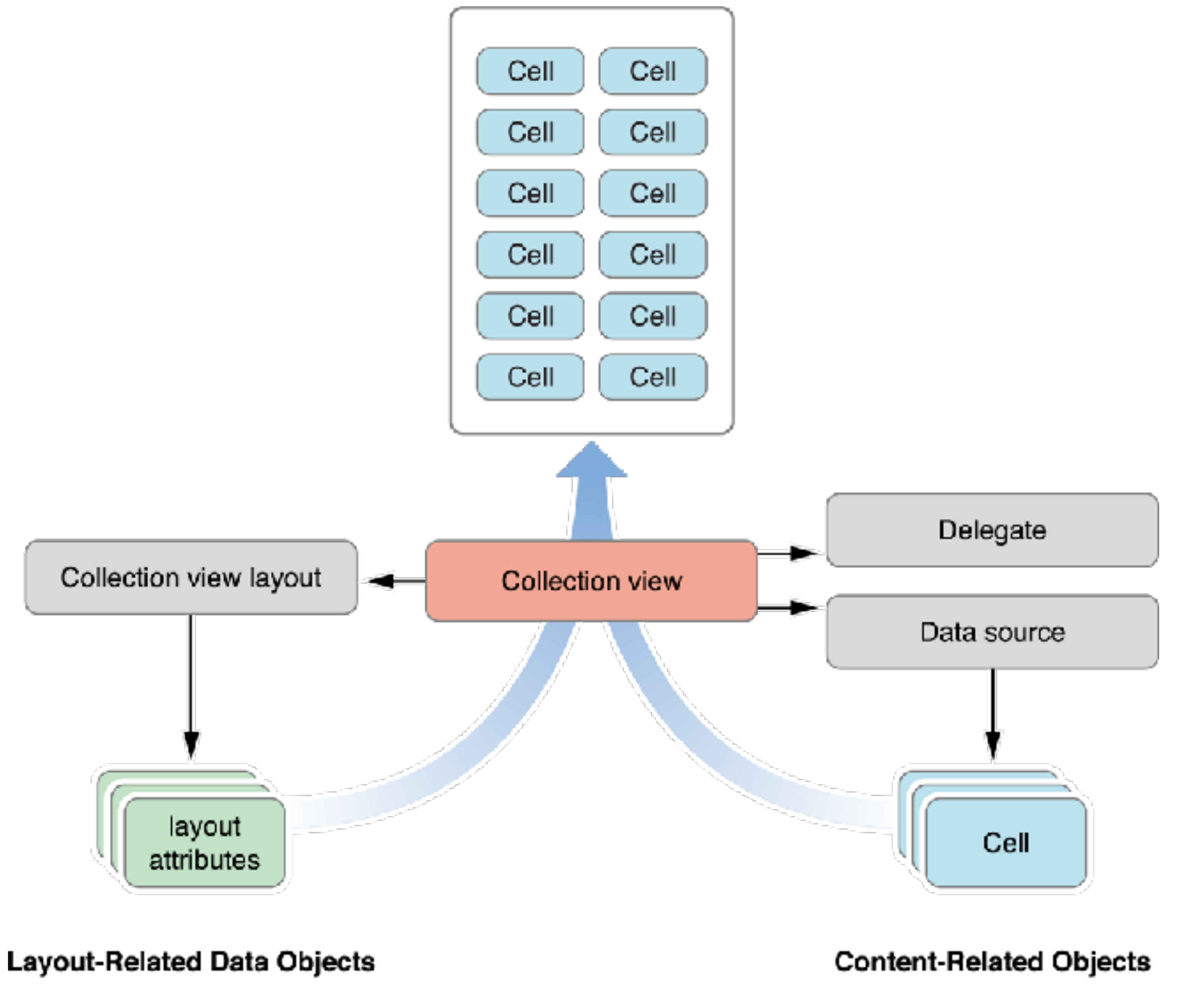
- 데이터 항목의 정렬 된 컬렉션을 관리하고 사용자 정의 레이아웃을 사용하여 데이터 항목을 제공하는 객체입니다



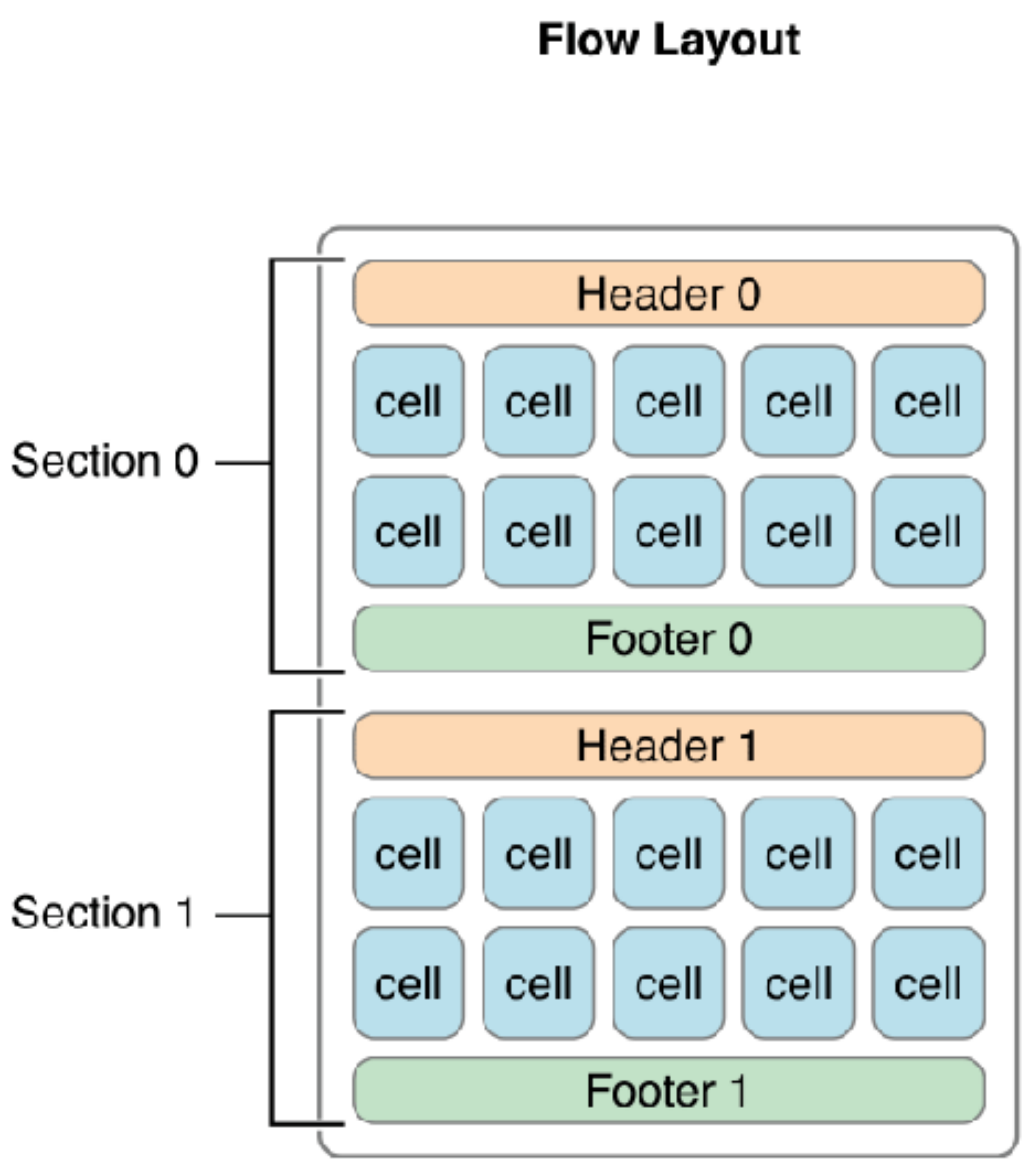
UICollectionView

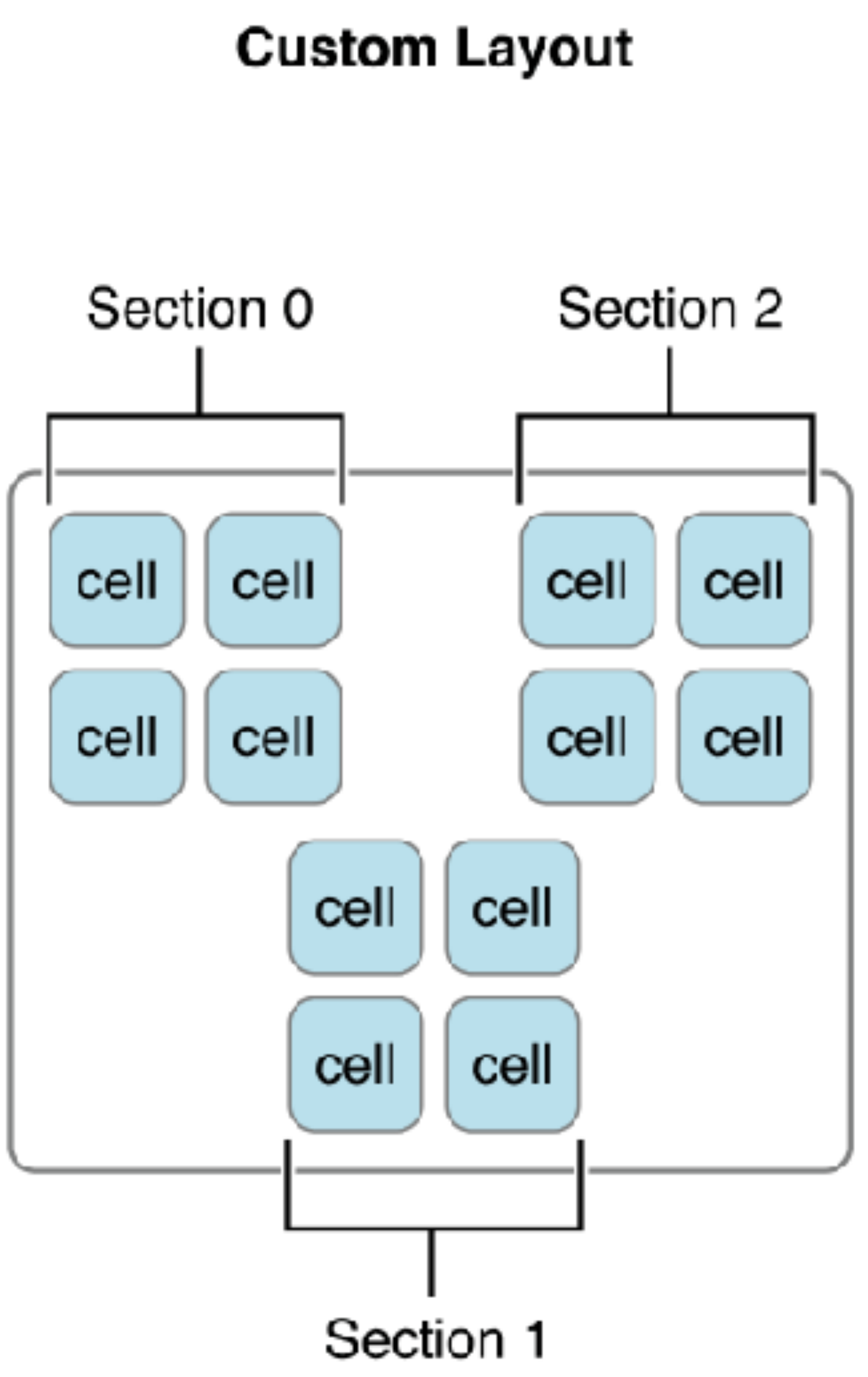
- The classes and protocols for implementing collection views
  - Manager
    - UICollectionView
    - UICollectionViewController
  - Protocol
    - UICollectionViewDataSource
    - UICollectionViewDelegate
  - Presentation
    - UICollectionReusableView
    - UICollectionViewCell
  - Layout
    - UICollectionViewLayout
    - UICollectionViewLayoutAttributes
    - UICollectionViewUpdateItem
  - Flow Layout
    - UICollectionViewFlowLayout
    - UICollectionViewDelegateFlowLayout

UICollectionView

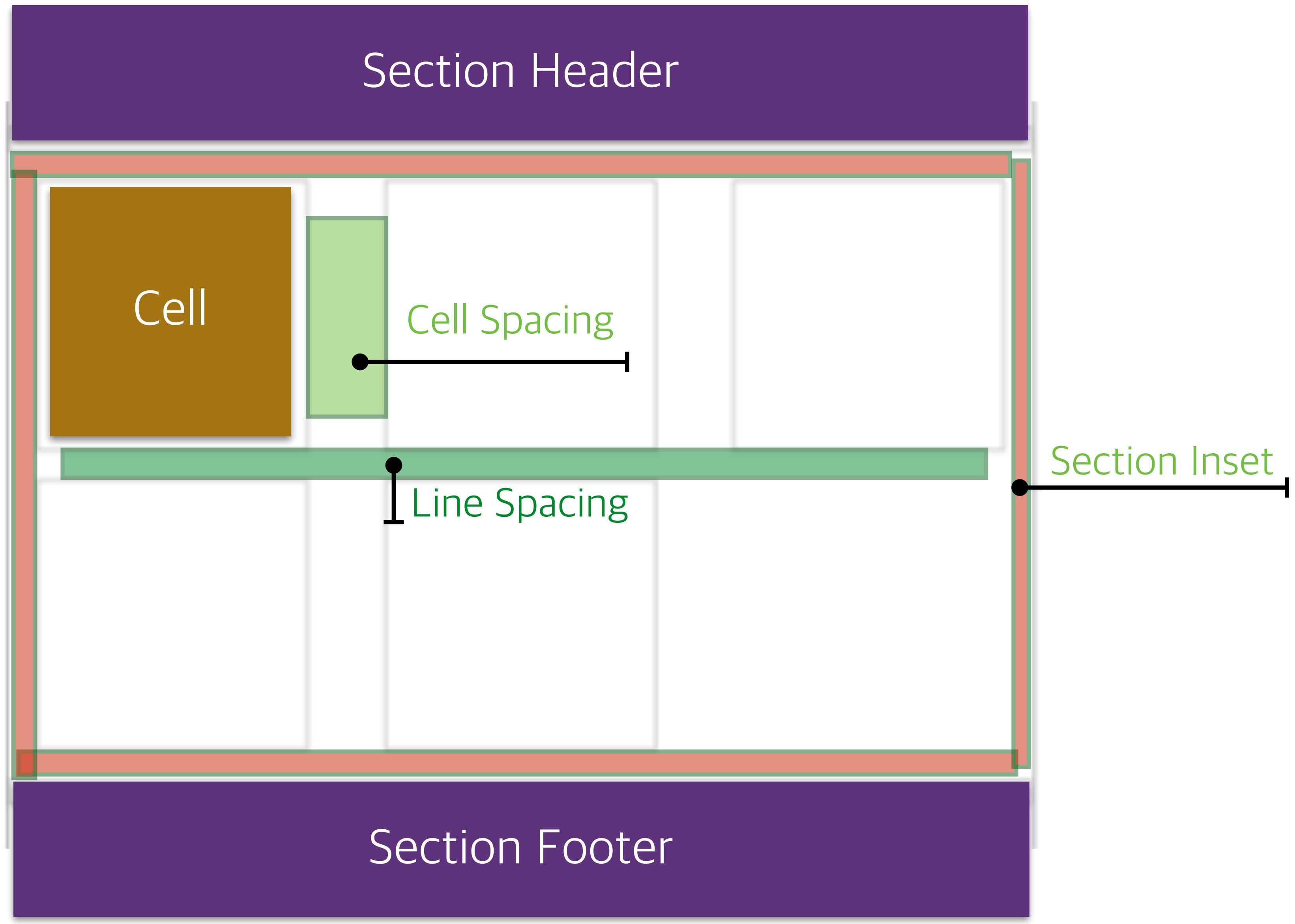


Layout


Flow Layout
Section 0
Header 0
cell
Footer 0
Section 1
Header 1
cell
Footer 1

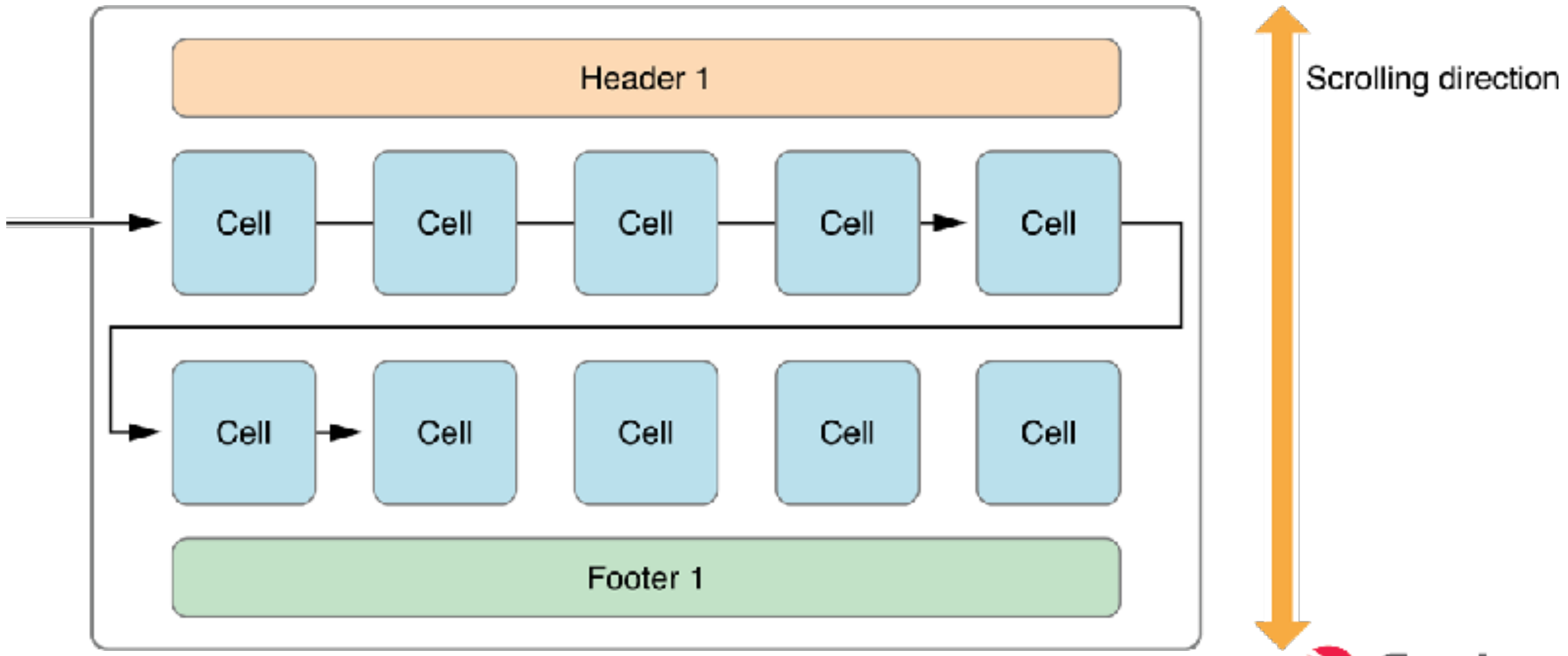

Custom Layout
Section 0
Section 2
cell
Section 1

Section Layout



FlowLayout

- UICollectionViewFlowLayout Class에 정의
- Scroll Direction
    1. Vertical
    2. Horizontal

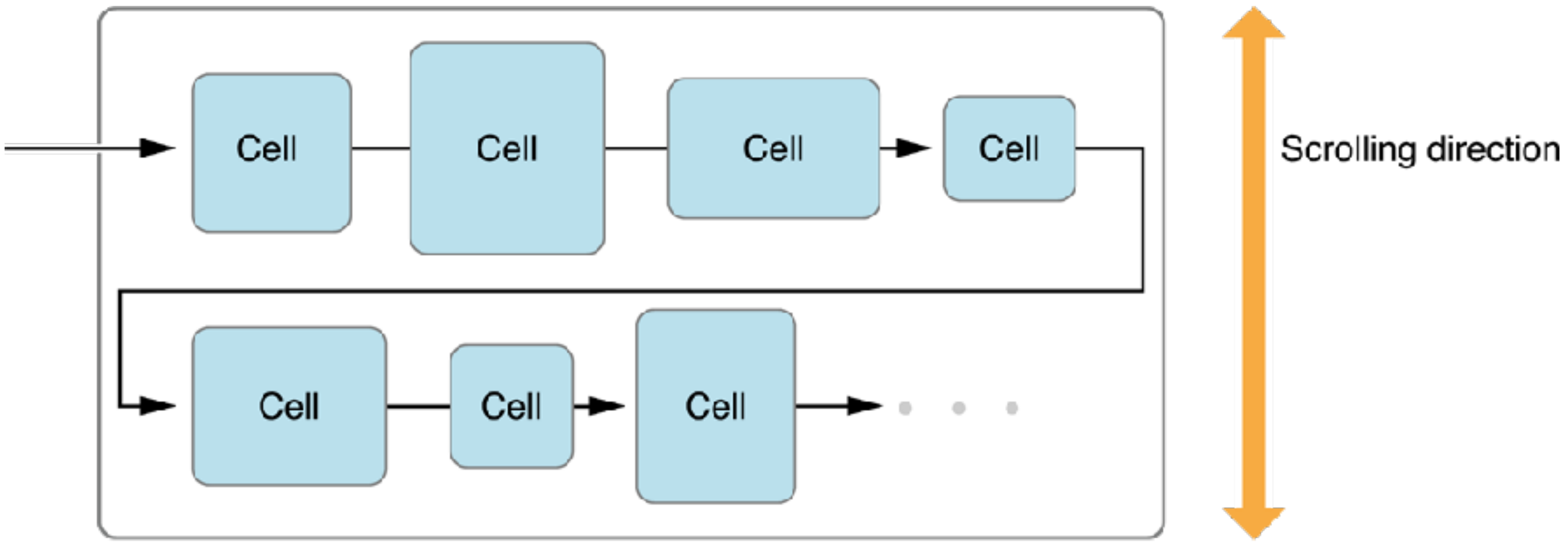


Flow Layout Attributes 정의

- UICollectionViewDelegateFlowLayout protocol 이용
- UICollectionViewDelegate가 정의된 인스턴스에서 작성.

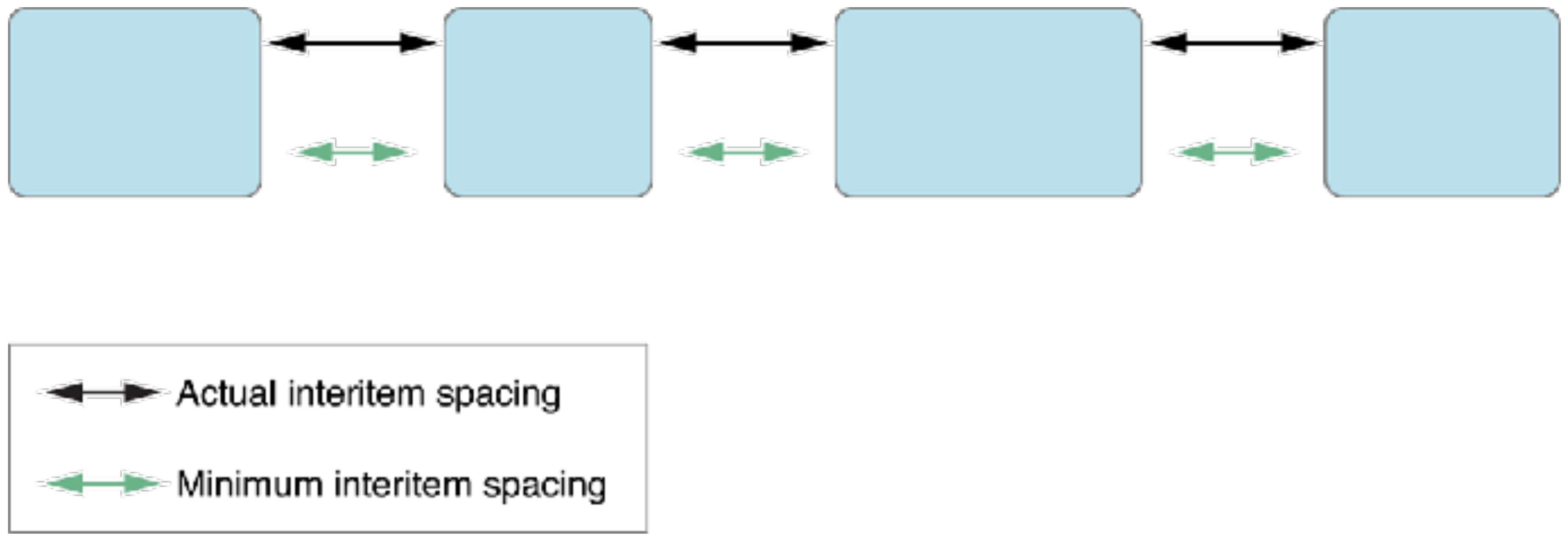
```
func collectionView(_ collectionView: UICollectionView, layout
collectionViewLayout: UICollectionViewLayout, sizeForItemAt indexPath:
IndexPath) -> CGSize

func collectionView(_ collectionView: UICollectionView, layout
collectionViewLayout: UICollectionViewLayout,
minimumInteritemSpacingForSectionAt section: Int) -> CGFloat

func collectionView(_ collectionView: UICollectionView, layout
collectionViewLayout: UICollectionViewLayout,
minimumLineSpacingForSectionAt section: Int) -> CGFloat

func collectionView(_ collectionView: UICollectionView, layout
collectionViewLayout: UICollectionViewLayout, insetForSectionAt
section: Int) -> UIEdgeInsets
```

Item Size



```
func collectionView(_ collectionView: UICollectionView,
          layout collectionViewLayout: UICollectionViewLayout,
             sizeForItemAt indexPath: IndexPath) -> CGSize
```

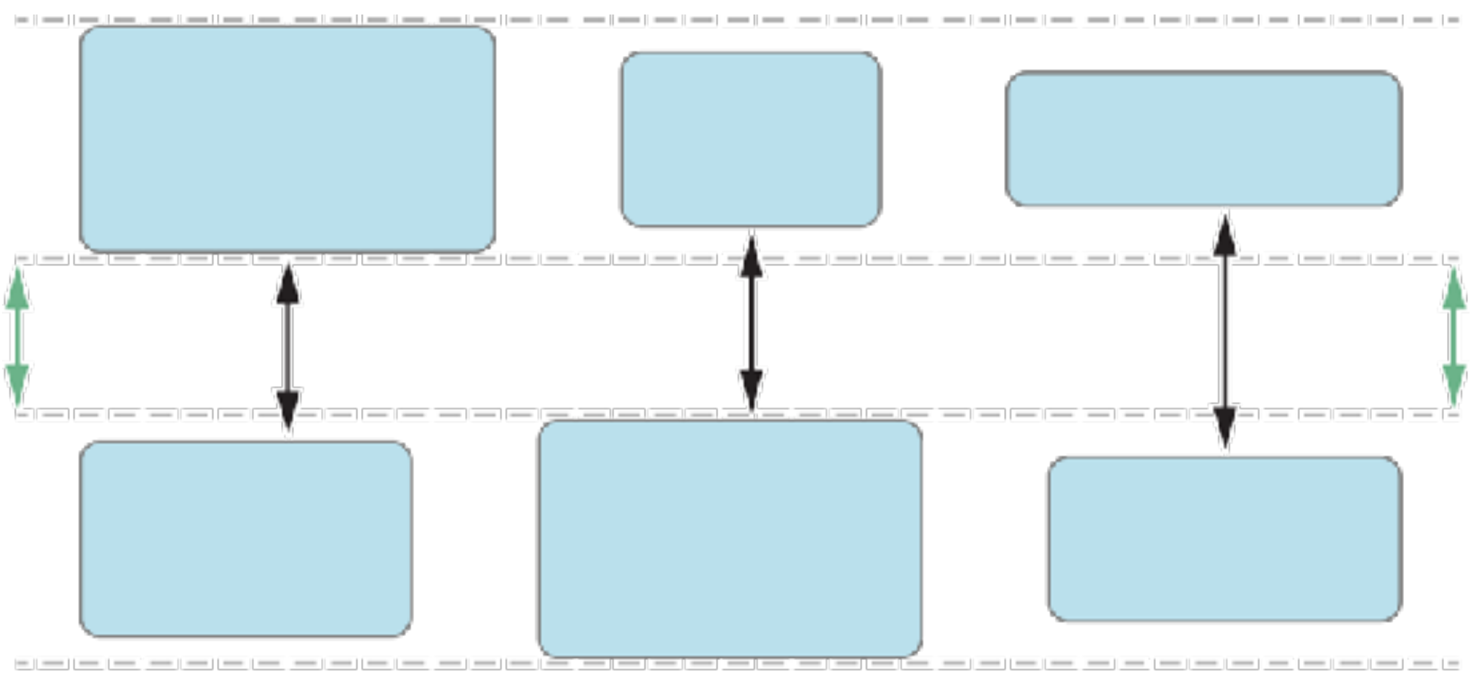
Item Spacing

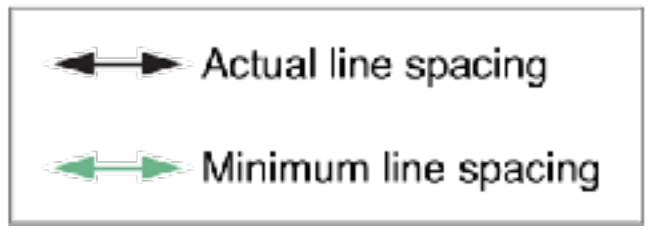


```
func collectionView(_ collectionView: UICollectionView,
          layout collectionViewLayout: UICollectionViewLayout,
            minimumInteritemSpacingForSectionAt section: Int) -> CGFloat
```

Fast campus

Line Spacing

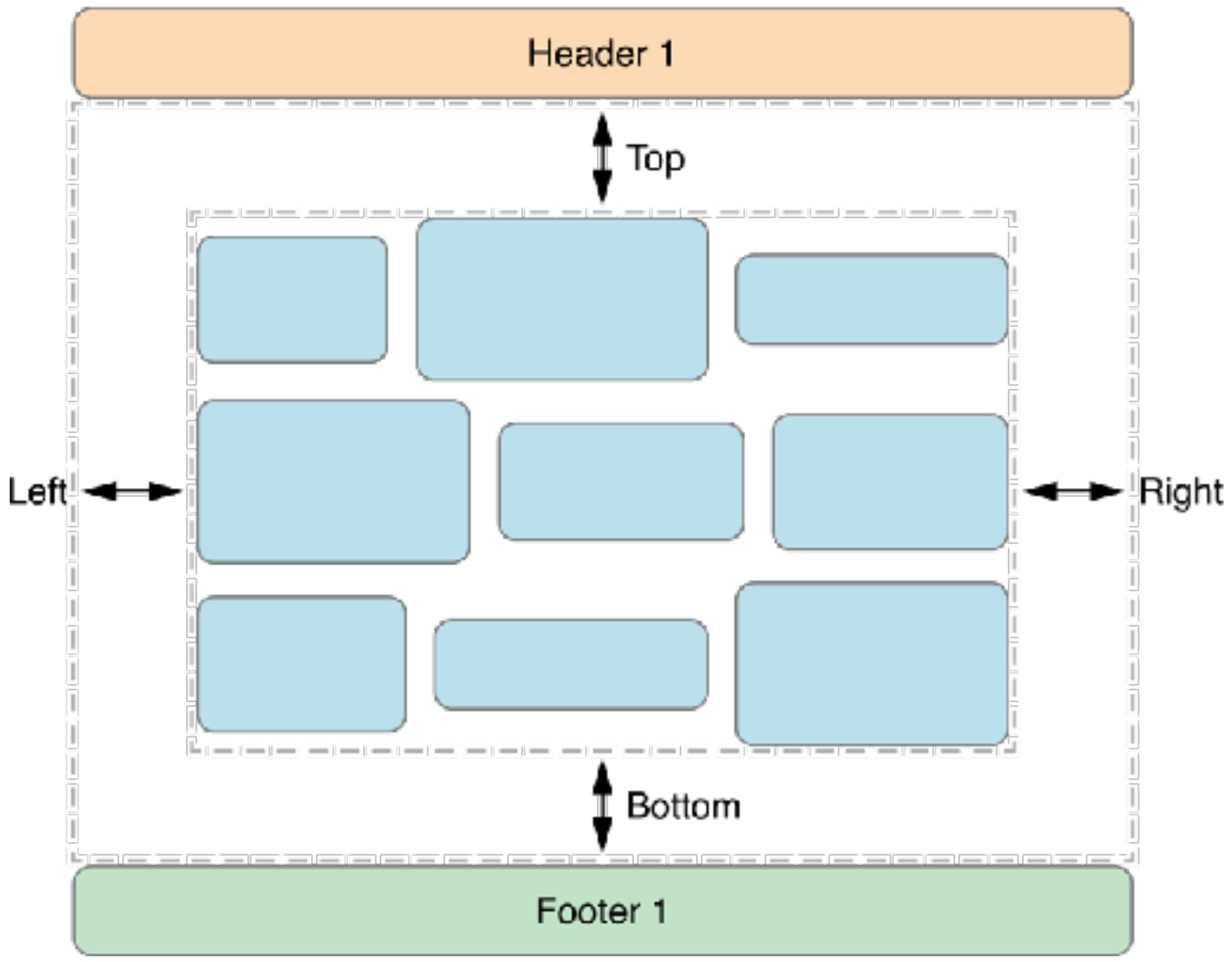




```
func collectionView(_ collectionView: UICollectionView,
          layout collectionViewLayout: UICollectionViewLayout,
           minimumLineSpacingForSectionAt section: Int) -> CGFloat
```

Section Inset

```
inset = UIEdgeInsetsMake(top, left, bottom, right)
```



```swift
func collectionView(_ collectionView: UICollectionView,
                layout collectionViewLayout: UICollectionViewLayout,
                  insetForSectionAt section: Int) -> UIEdgeInsets
```

Protocol

- UICollectionViewDataSource
- UICollectionViewDelegate
- UICollectionViewDataSourcePrefetching

UICollectionViewDataSource

```
public func collectionView(_ collectionView: UICollectionView,
numberOfItemsInSection section: Int) -> Int

public func collectionView(_ collectionView: UICollectionView,
cellForItemAt indexPath: IndexPath) -> UICollectionViewCell

optional public func numberOfSections(in collectionView: UICollectionView)
-> Int
```

UICollectionViewDelegate

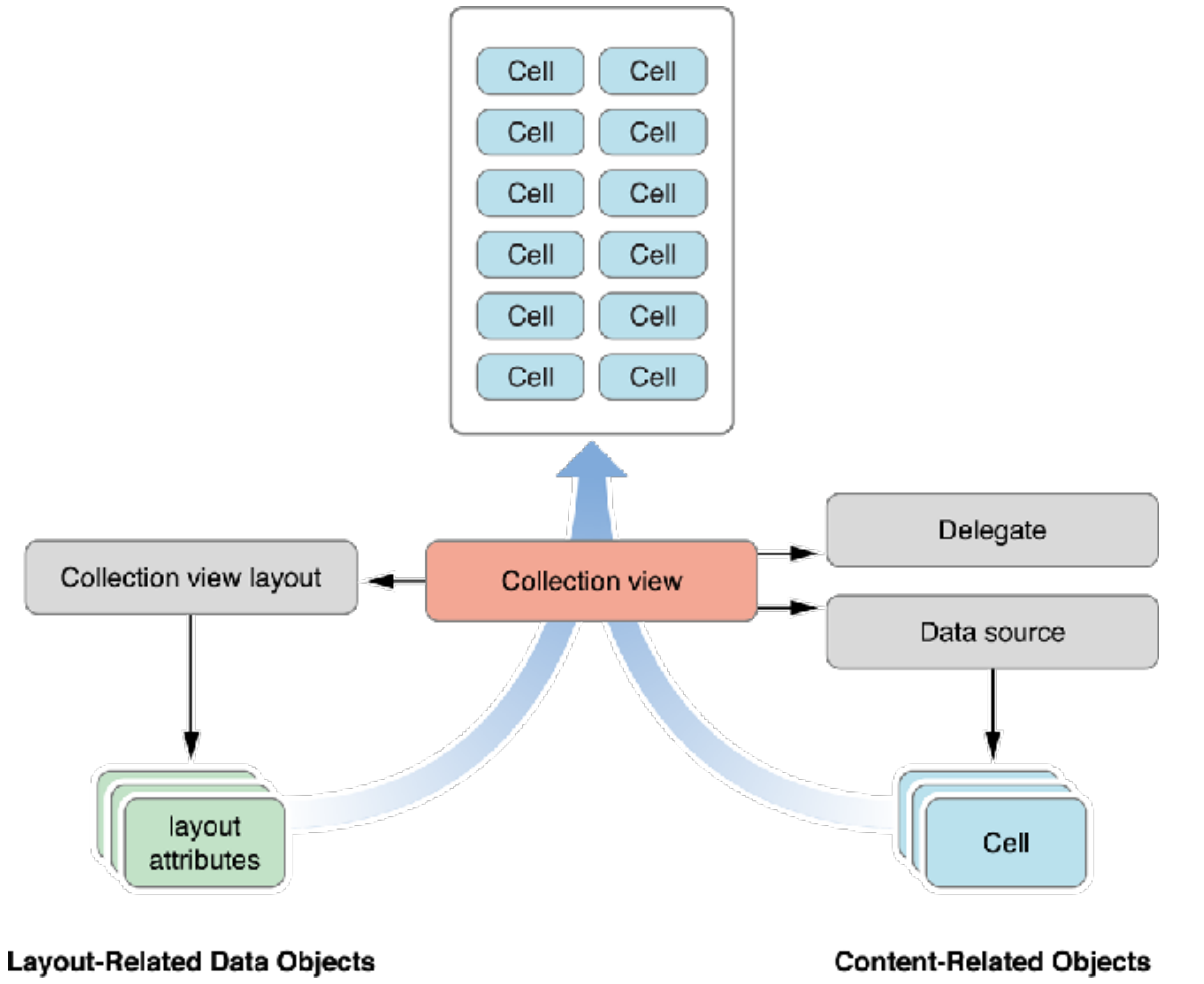
```
optional public func collectionView(_ collectionView: UICollectionView,
shouldHighlightItemAt indexPath: IndexPath) -> Bool

optional public func collectionView(_ collectionView: UICollectionView,
didHighlightItemAt indexPath: IndexPath)

optional public func collectionView(_ collectionView: UICollectionView,
didUnhighlightItemAt indexPath: IndexPath)

optional public func collectionView(_ collectionView: UICollectionView,
shouldSelectItemAt indexPath: IndexPath) -> Bool

optional public func collectionView(_ collectionView: UICollectionView,
shouldDeselectItemAt indexPath: IndexPath) -> Bool // called when the user
taps on an already-selected item in multi-select mode

optional public func collectionView(_ collectionView: UICollectionView,
didSelectItemAt indexPath: IndexPath)

optional public func collectionView(_ collectionView: UICollectionView,
didDeselectItemAt indexPath: IndexPath)
```

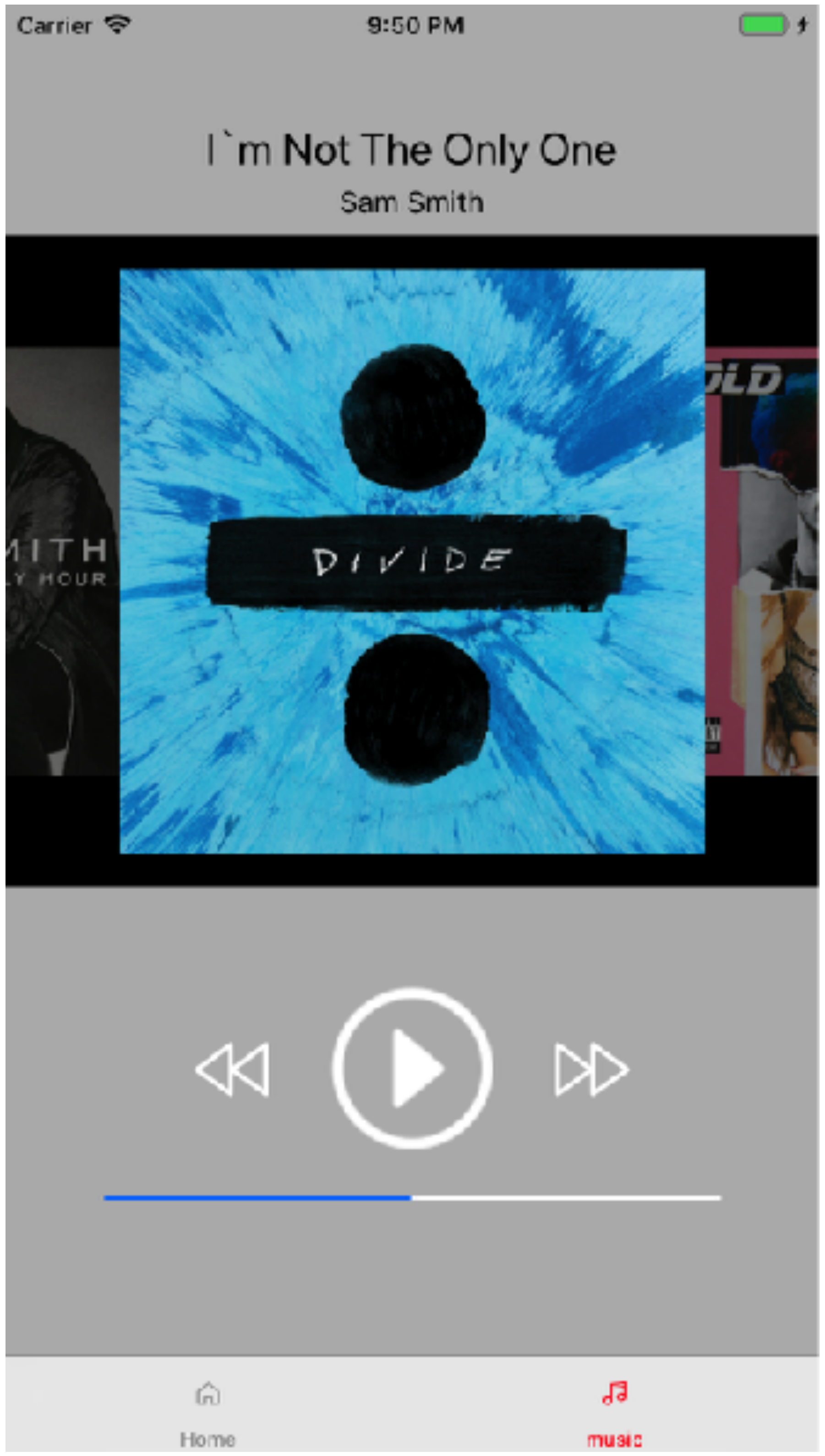
컬렉션 뷰 만들기

- 같이 해봐요

UICollection-Layout

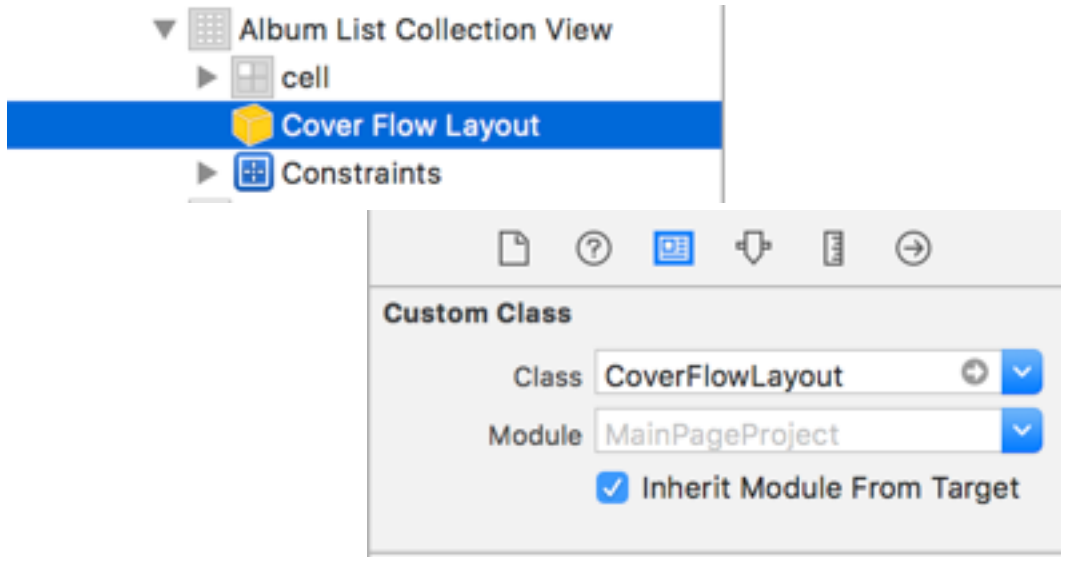
UICollectionView





CoverFlow만들기

1. UICollectionViewFlowLayout 상속 받은 커스텀 layout 만들기
2. Layout CustomClass에 적용

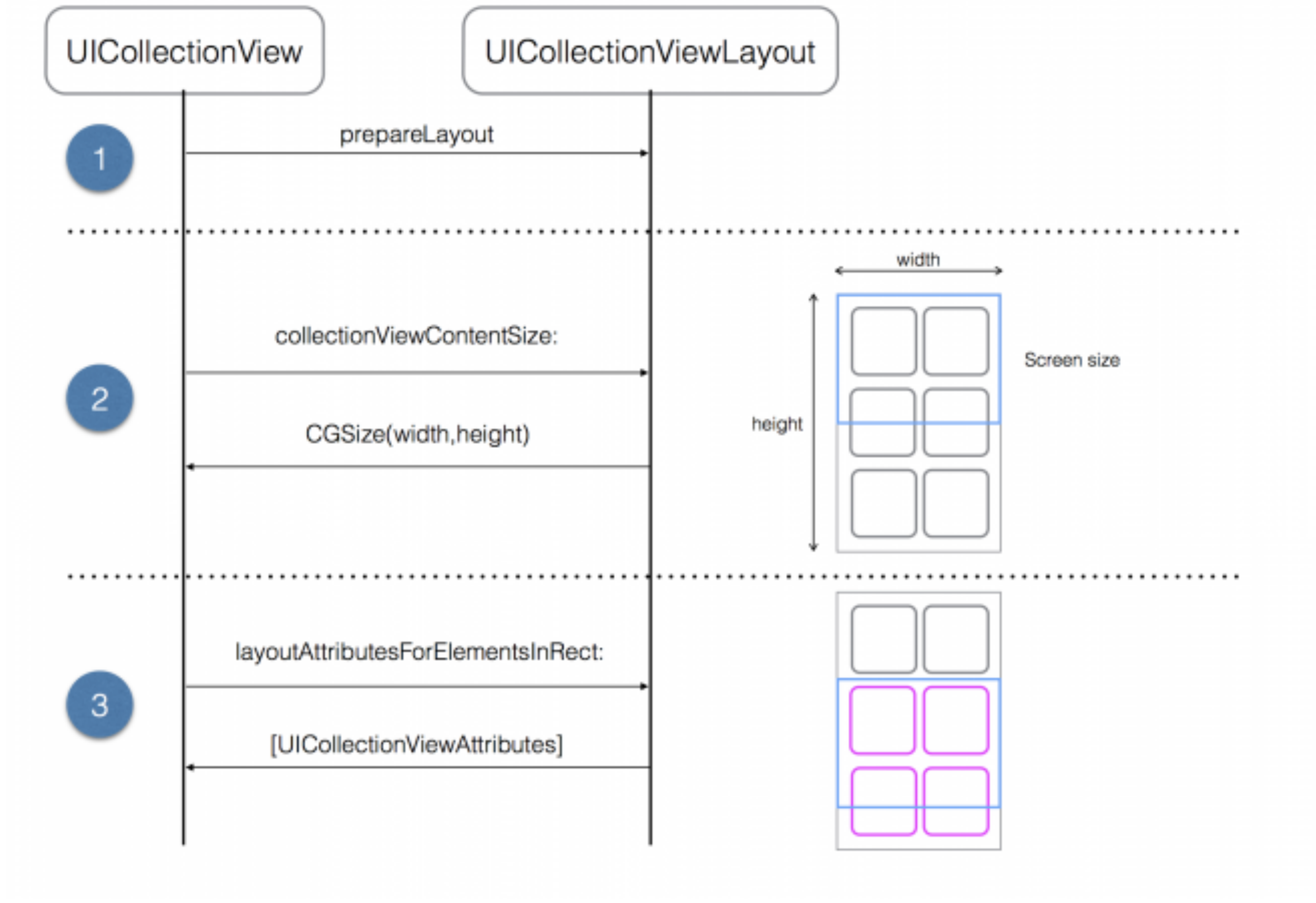


Class CoverFlowLayout

Module MainPageProject

✓ Inherit Module From Target

Layout Process



CoverFlow

```
override func layoutAttributesForElements(in rect: CGRect) ->
[UICollectionViewLayoutAttributes]? {

    guard  let attributes = super.layoutAttributesForElements(in: rect) else
{return nil}

    var layoutAttribute:[UICollectionViewLayoutAttributes] = []
    for attribute in attributes
    {
        //change
        changeLayoutAttribute(attribute: attribute)
        //add
        layoutAttribute.append(attribute)
    }
    return layoutAttribute
}

//레이아웃 정보들 다시 불러오도록 허락함
override func shouldInvalidateLayout(forBoundsChange newBounds: CGRect) ->
Bool {
     return true
}
```

CoverFlow

```
//실제 연산내용
func changeLayoutAttribute(attribute:UICollectionViewLayoutAttributes)
{
    //센터 컬렉션 뷰의 반
    let collectionViewCenter = (self.collectionView?.frame.size.width)! / 2.0
    //현재 아이템의 offsetX + 센터
    let offSet = (self.collectionView?.contentOffset.x)! + collectionViewCenter
    //변경가능한 최대 거리
    let maxDistance = self.itemSize.width + self.minimumLineSpacing
    //최대거리 이상은 변경 않함
    let distance = min(fabs(offSet - attribute.center.x), maxDistance)
    //비율
    let ratio = (maxDistance - distance) / maxDistance
    //비율에 따라 스케일과 투명도 변경
    let scale = ratio * (1 - self.itemScale) + 1.0
    let alpha = ratio * (1 - self.itemAlpha) + self.itemAlpha;

    attribute.alpha = alpha;
    attribute.transform3D = CATransform3DScale(CATransform3DIdentity, scale,
scale, 1);
    //alpha값에 따른 z 좌표 변경
    attribute.zIndex = NSInteger(alpha * 10.0)
}
```

CoverFlow

```
override func targetContentOffset(forProposedContentOffset proposedContentOffset:
CGPoint, withScrollingVelocity velocity: CGPoint) -> CGPoint {
    print(proposedContentOffset.x)
    //0.준비
    guard let collectionView = self.collectionView  else {
        return proposedContentOffset
    }
    //현재 컬렉션 뷰의 [UICollectionViewLayoutAttributes] 가져오기
    guard let attributeList =  self.layoutAttributesForElements(in:
collectionView.bounds) else {
        return proposedContentOffset
    }
    //1.센터 위치
    let xCenter = collectionView.frame.size.width / 2
    //2. 아이템 정렬
    let sortedAttributeList = attributeList.sorted { (attribute1, attribute2) -> Bool
in
        attribute1.center.x > attribute2.center.x
    }
    //3. 중앙이랑 가장 가까운 아이템 중앙값
    let xCenterOfMinimumAttributes = sortedAttributeList.first?.center.x
    //4. 중앙으로 이동
    let targetContentOffset = CGPoint(x:xCenterOfMinimumAttributes! - xCenter, y:
proposedContentOffset.y)

    return targetContentOffset
}
```

추가 학습

- 한번 해보세요

*핀터레스트 만들기

https://www.raywenderlich.com/164608/uicollectionview-custom-layout-tutorial-pinterest-2

*단축 주소

https://goo.gl/NtLmRy

UIGestureRecognizer

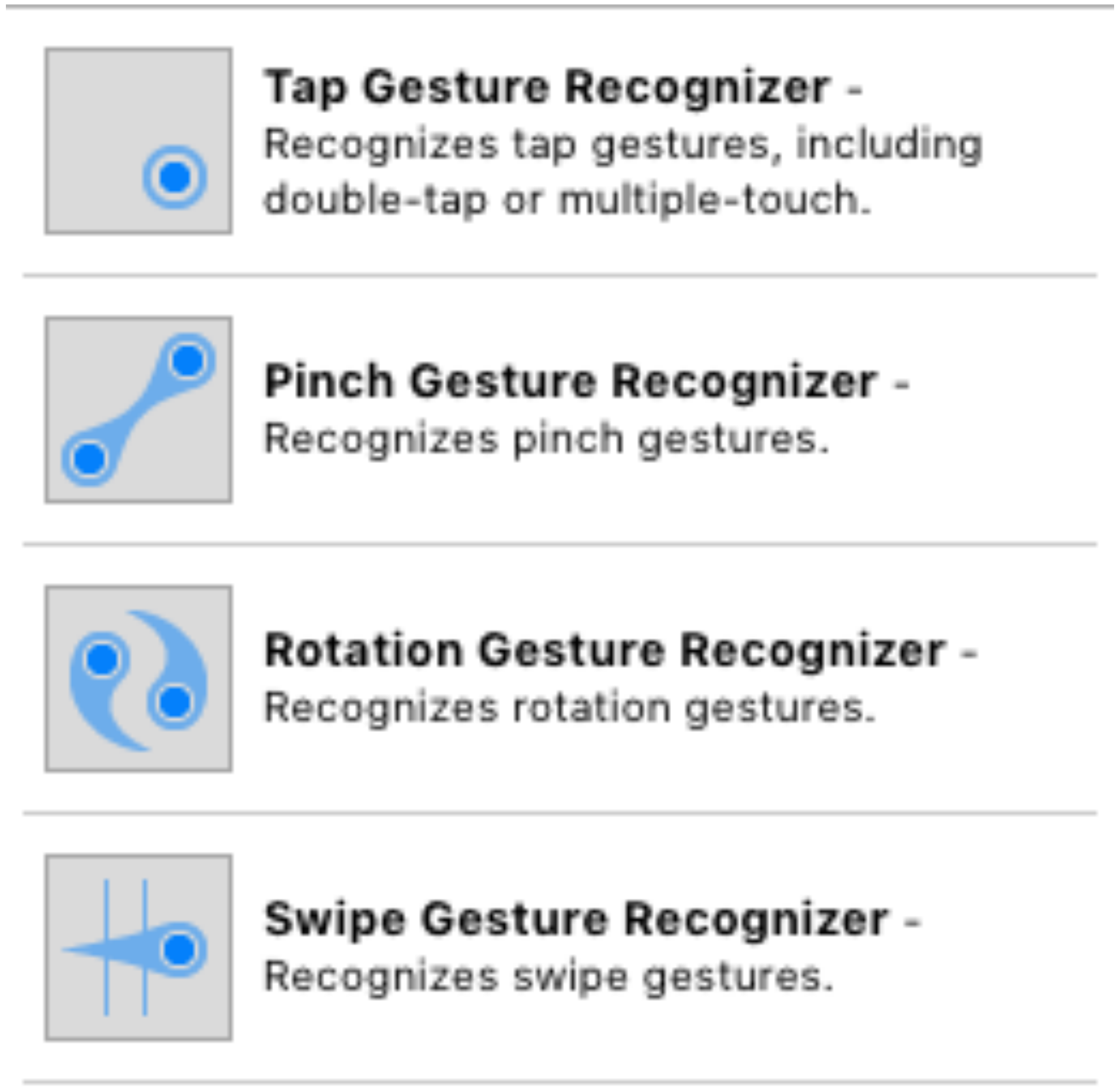
강사 주영민

Fast campus

UIGestureRecognizer

- 사용자의 입력을 전달받을 수 있는 방법을 제공
- Tap, Pinch, Rotation, Swipe, Pan(drag), Edge Pan, Long Press 등을 인지하는 각각의 서브클래스 존재
- View 위에 얹어 액션을 핸들링

UIGestureRecognizer 종류

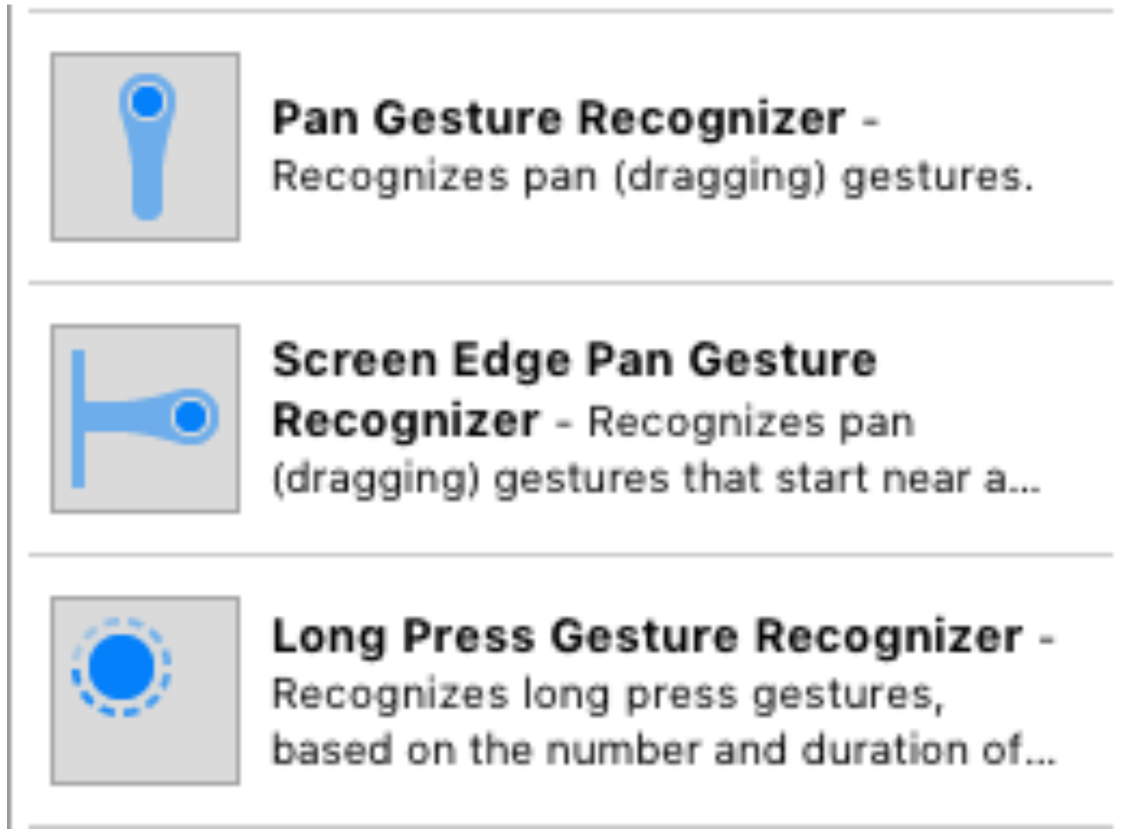


Tap Gesture Recognizer - Recognizes tap gestures, including double-tap or multiple-touch.

Pinch Gesture Recognizer - Recognizes pinch gestures.

Rotation Gesture Recognizer - Recognizes rotation gestures.

Swipe Gesture Recognizer - Recognizes swipe gestures.



Pan Gesture Recognizer - Recognizes pan (dragging) gestures.

Screen Edge Pan Gesture Recognizer - Recognizes pan (dragging) gestures that start near a...

Long Press Gesture Recognizer - Recognizes long press gestures, based on the number and duration of...

Step 1. header file 보기

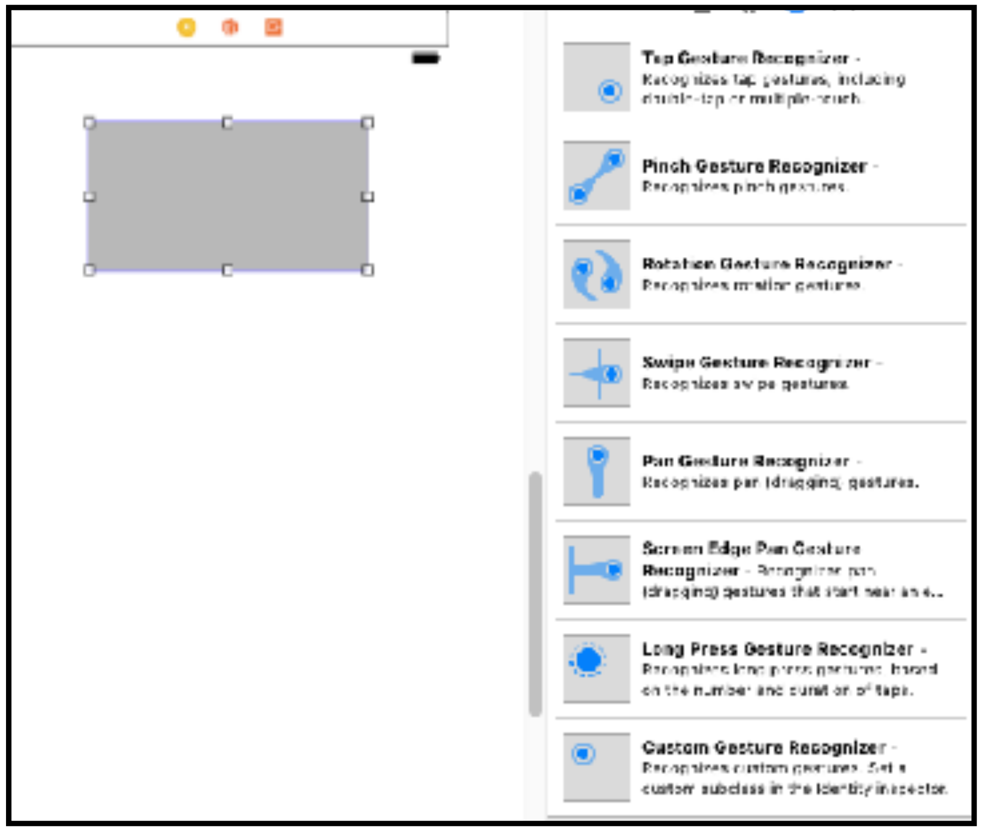
- UIGestureRecognizer Header file 보기
- UIGestureRecognizerDelegate

Step 2. Sample Code

```
let tapGesture = UITapGestureRecognizer(target: self,
                                action: #selector(ViewController.tapAction(_:)))

self.view.addGestureRecognizer(tapGesture)

//ViewController내 존재 하는 함수
@objc func tapAction(_ sender:UITapGestureRecognizer)
{

}
```

Fast campus

Step 2. Using Storyboard



Tap Gesture Recognizer - Recognizes tap gestures, including double-tap or multiple-touch.

Pinch Gesture Recognizer - Recognizes pinch gestures.

Rotation Gesture Recognizer - Recognizes rotation gestures.

Swipe Gesture Recognizer - Recognizes swipe gestures.

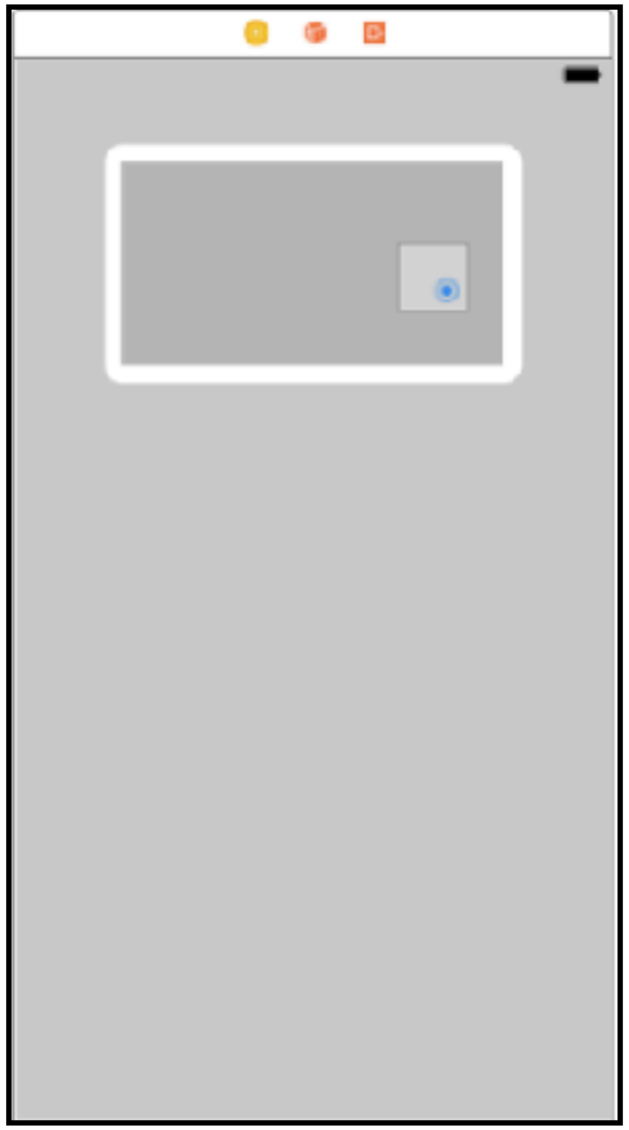
Pan Gesture Recognizer - Recognizes pan (dragging) gestures.

Screen Edge Pan Gesture Recognizer - Recognizes pan (dragging) gestures that start near an e...

Long Press Gesture Recognizer - Recognizes long press gestures, based on the number and duration of taps.

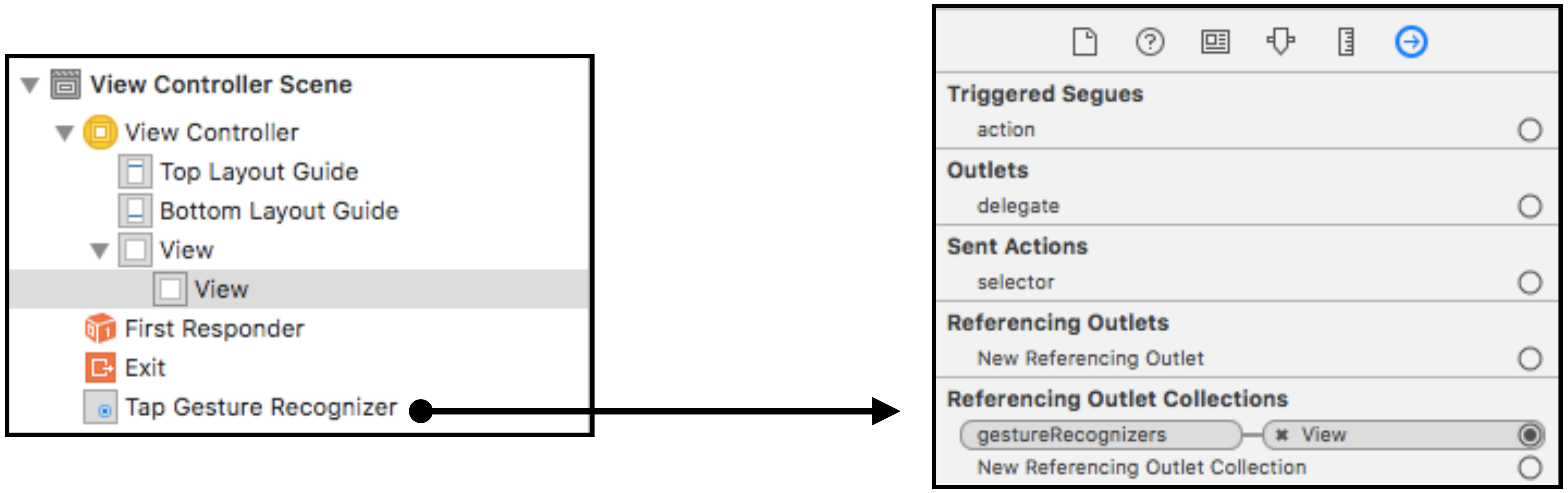
Custom Gesture Recognizer - Recognizes custom gestures. Set a custom subclass in the Identity inspector.

Drag and Drop



Step 2. Using Storyboard

- 선택된 View 에 GestureRecognizer가 설정됨



View Controller Scene
View Controller
Top Layout Guide
Bottom Layout Guide
View
View
First Responder
Exit
Tap Gesture Recognizer

Triggered Segues
action
Outlets
delegate
Sent Actions
selector
Referencing Outlets
New Referencing Outlet
Referencing Outlet Collections
gestureRecognizers — View
New Referencing Outlet Collection

Gesture Delegate

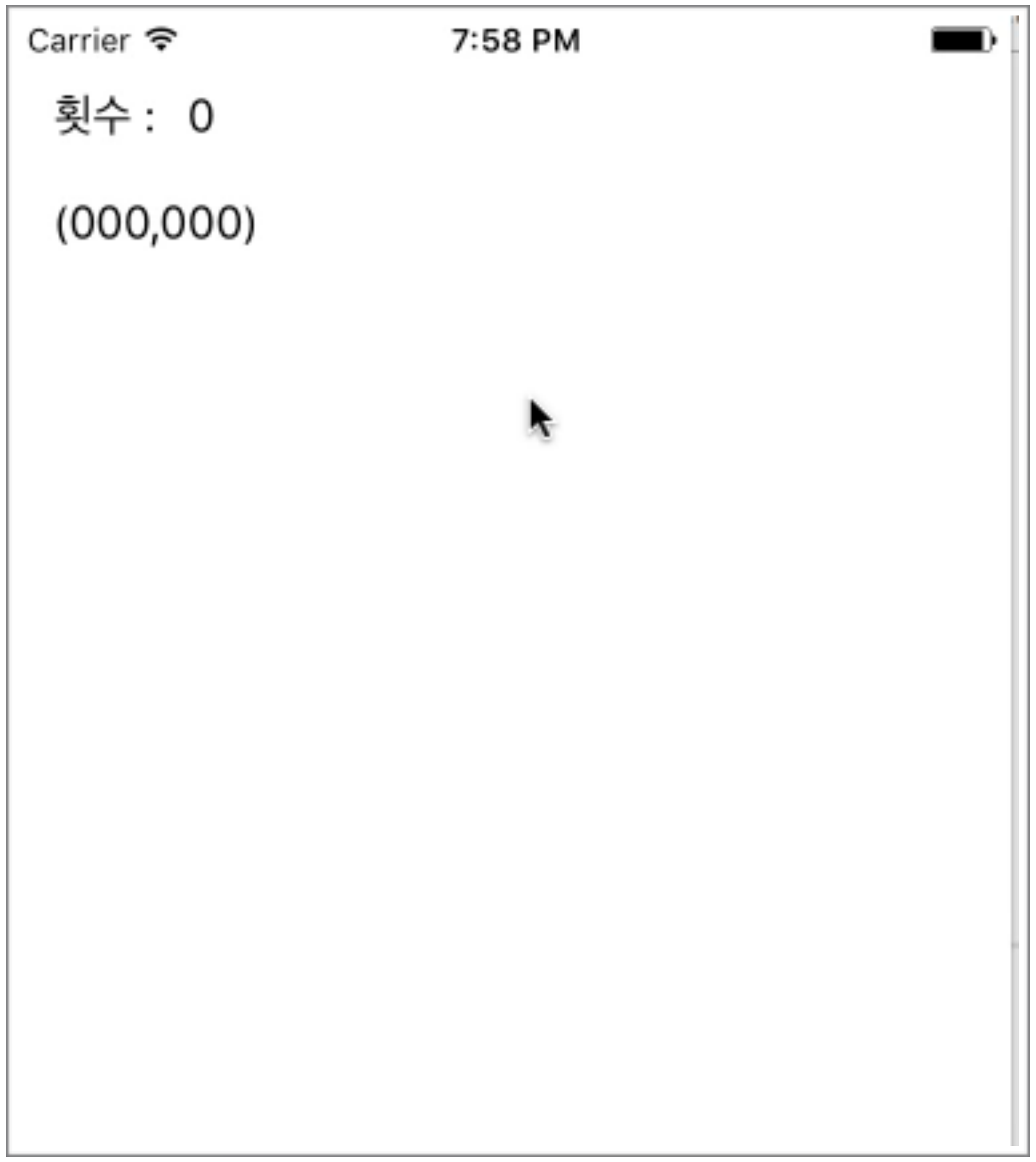
***가장 많이 사용하는 Delegate메소드**

```
func gestureRecognizer(_ gestureRecognizer:
UIGestureRecognizer, shouldReceive touch: UITouch) -> Bool {
    //터치된 포인트가 inView위치에 어느 좌표에 해당되는지 표시
    print("xposition", touch.location(in: touch.view).x)
    //터치가 일어난 시간 반환
    print("touch timeStamp”,touch.timestamp)
    //연속적으로 일어난 터치의 횟수
    print("touch tapCount",touch.tapCount)
    return true
}
```

```
xposition 61.6666564941406
touch timeStamp 188786.85859217
touch tapCount 1
```

Fast campus

Step 3. Exercise



횟수 : 0

(000,000)