

CollectionKit Workshop

Luke Zhao

Luke

- 北美工作的iOS猿
- 开源框架Hero和CollectionKit作者
- 苹果, Pinterest, Snapchat (Bitmoji)
- [https://github.com/HeroTransitions/
Hero](https://github.com/HeroTransitions/Hero)
- [https://github.com/SoySauceLab/
CollectionKit](https://github.com/SoySauceLab/CollectionKit)





Mac

iPad

iPhone

Watch

TV

Music

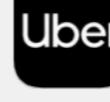
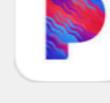
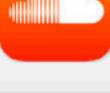
Support



App Store Preview

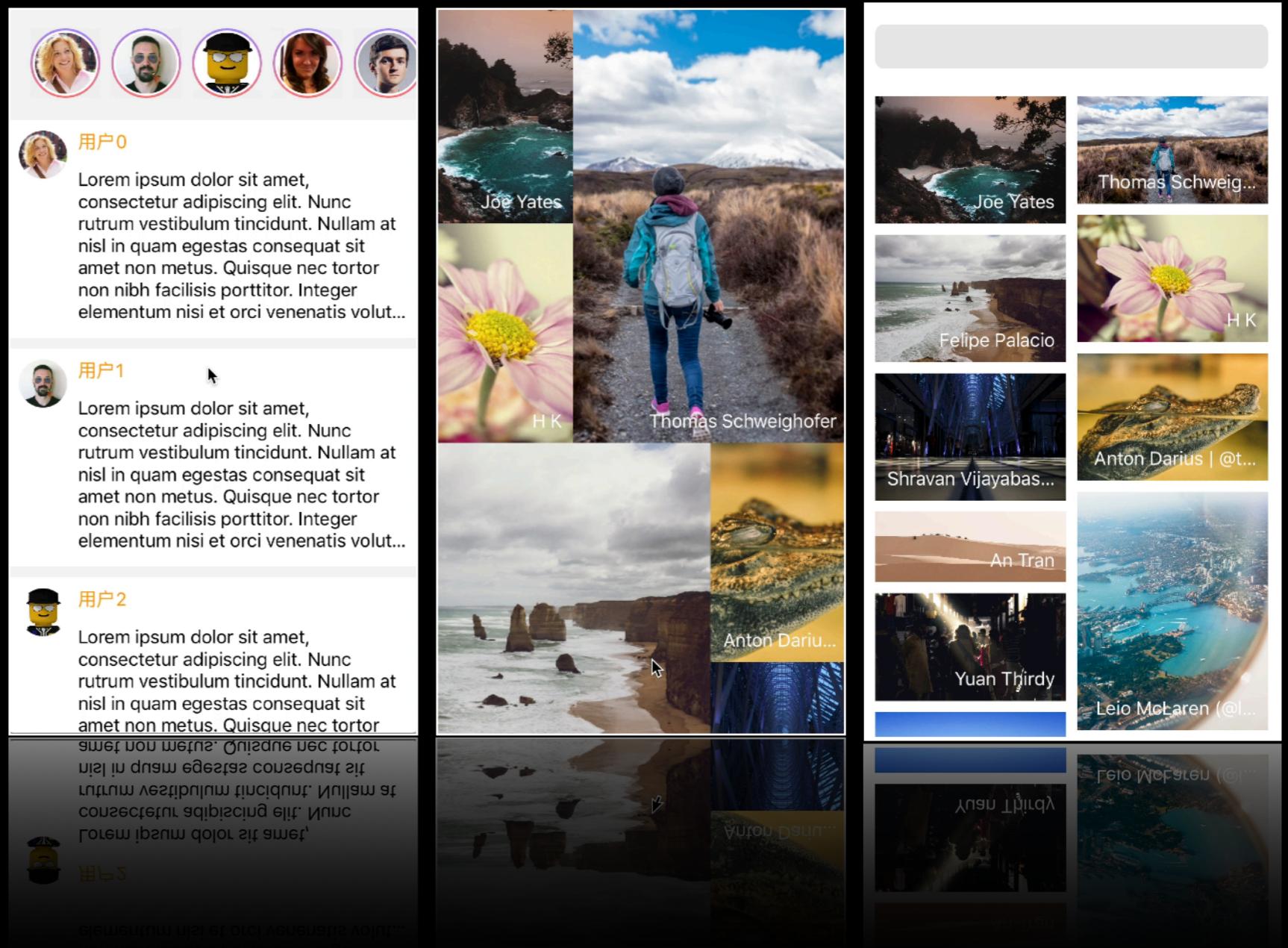
BEST OF 2017

Top Apps Charts

| | | | |
|---|----------------------|---|----------------------|
|  Bitmoji Utilities | VIEW |  Snapchat Photo & Video | VIEW |
|  YouTube: Watch, Listen, Stream Photo & Video | VIEW |  Messenger Social Networking | VIEW |
|  Instagram Photo & Video | VIEW |  Facebook Social Networking | VIEW |
|  Google Maps - Transit & Food Navigation | VIEW |  Netflix Entertainment | VIEW |
|  Spotify Music Music | VIEW |  Uber Travel | VIEW |
|  Gmail - Email by Google Productivity | VIEW |  Pandora Music Music | VIEW |
|  Amazon - Shopping made easy Shopping | VIEW |  WhatsApp Messenger Social Networking | VIEW |
|  Wish - Shopping Made Fun Shopping | VIEW |  Twitter News | VIEW |
|  SoundCloud - Music & Audio Music | VIEW |  Google Chrome Utilities | VIEW |

日程

- CollectionKit简介
- 微博
- 图片库
- 休息 (5分钟)
- 图片搜索
- 内部构造与设计思路



Requirements

- Workshop Project 下载链接：[https://github.com/lkzhao/
CollectionKitWorkshop](https://github.com/lkzhao/CollectionKitWorkshop)
- Xcode 9.4.1
- UICollectionView的一些概念
- 无需Auto Layout, Storyboard
- 一点点几何数学（计算大小, 位置）

CollectionKit

- Reimagining UICollectionView

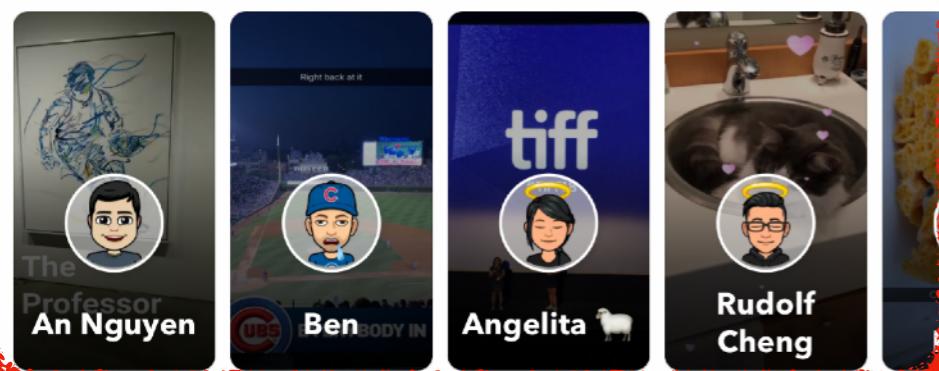
- 代替UICollectionView。显示集合视图
- 数据驱动，更简单更高效
- 提供了布局和动画系统
- 为Swift设计，用Swift实现

21:36 ↗

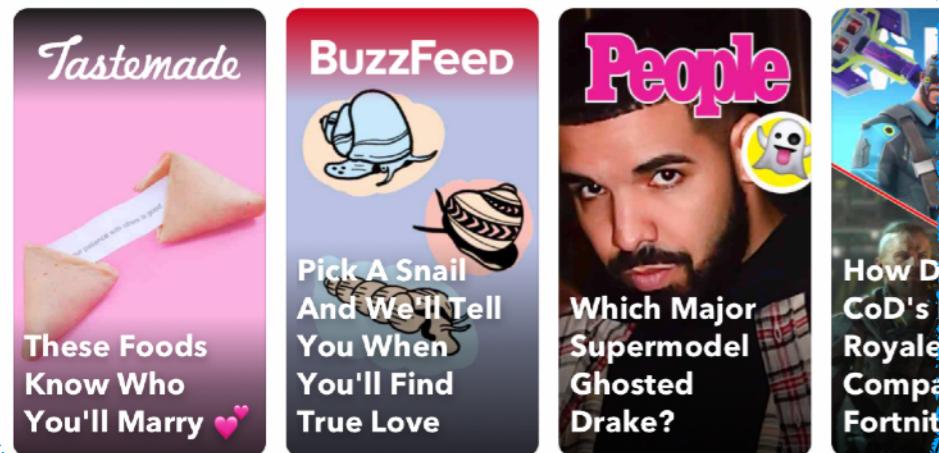


Discover

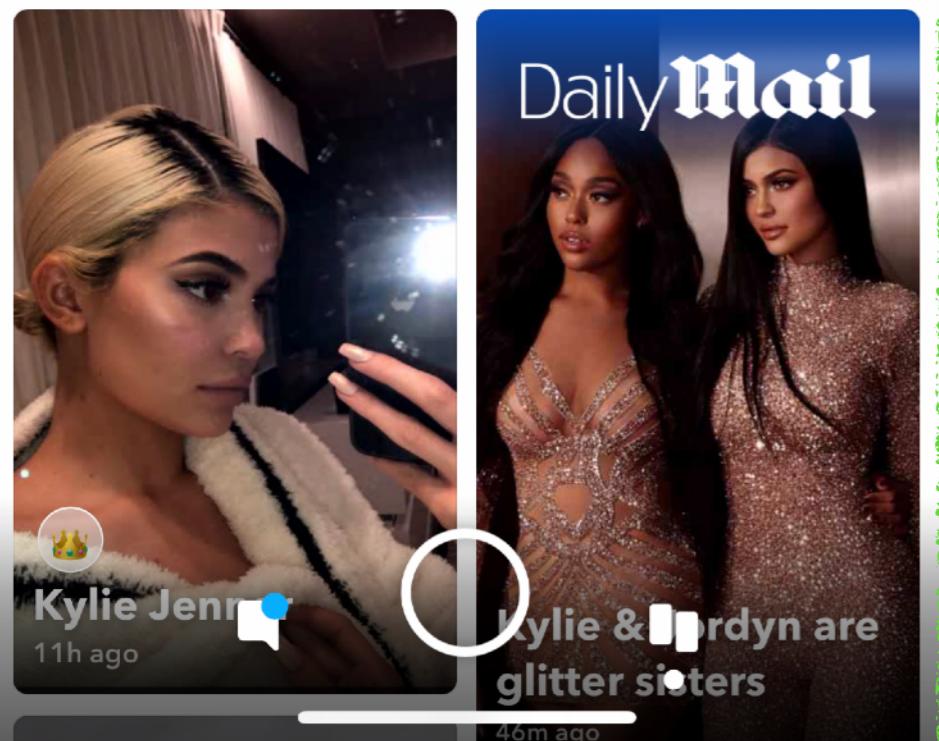
Friends



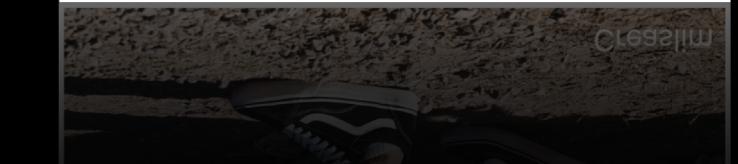
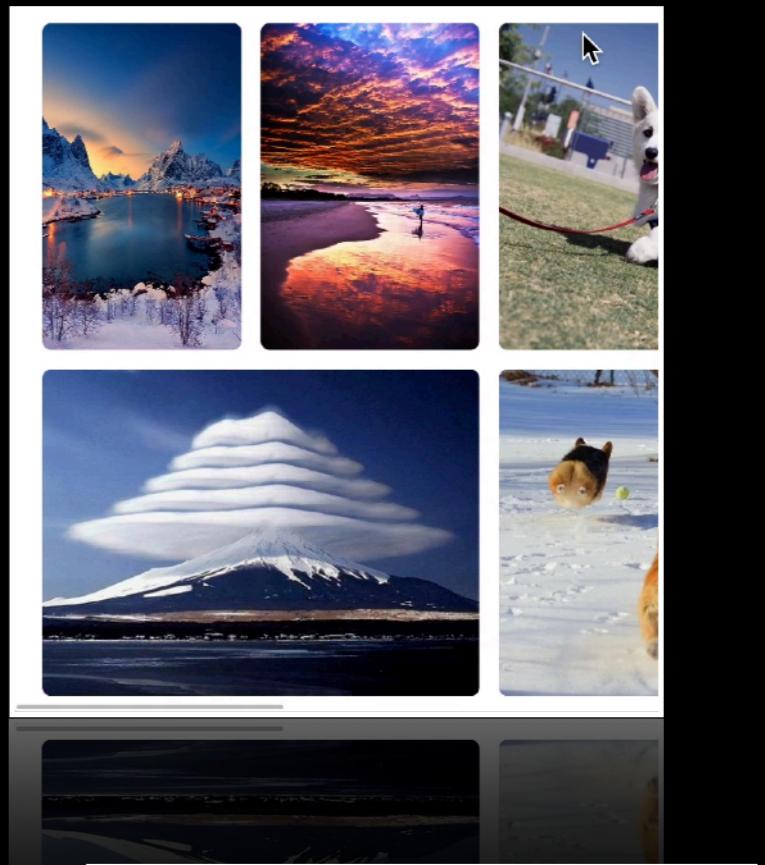
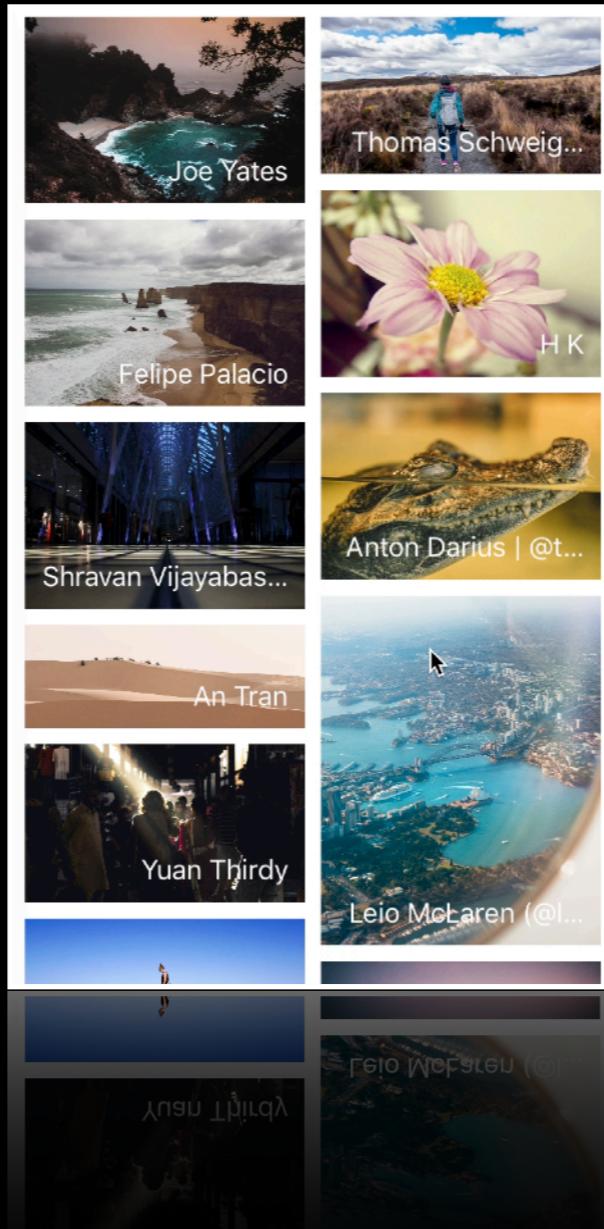
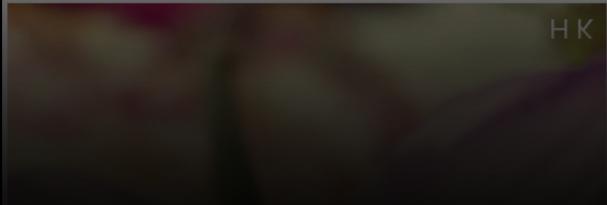
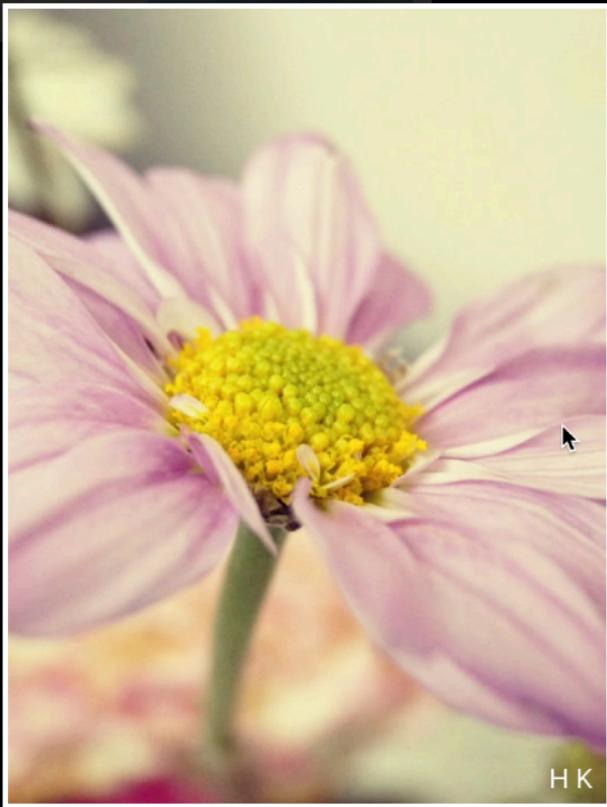
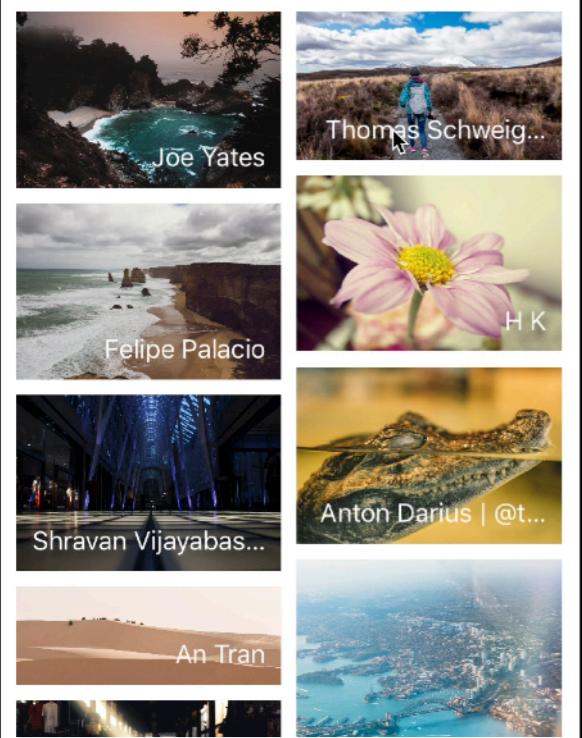
Subscriptions



For You



```
collectionView.provider = ComposedProvider(sections: [  
    FriendSectionProvider(friends: friends),  
    SubscriptionSectionProvider(subscriptions: subscriptions),  
    ForYouSectionProvider(stories: stories),  
])
```



对比UICollectionView

- 简单的管理各个Section的顺序和布局
- Cell无需是UICollectionViewCell的子类
- Type Safe, no more force casting
- 不再是一个黑盒子

对比IGListKit

- 数据管理层面非常相似
- 抛弃了和UICollectionView的连接，变得更加简洁
- 提供了布局和动画支持

Examples of things beyond the scope of IGListKit:

- Advanced/custom UICollectionViewLayouts
- Sizing and layout (e.g. auto layout, estimated sizes)
- Render and display pipelines

CollectionView

.provider

CollectionView

.provider

BasicProvider

ComposedProvider

SimpleViewProvider

CollectionView

.provider

BasicProvider

ComposedProvider

SimpleViewProvider

```
collectionView.provider = BasicProvider( ... )
```

BasicProvider

.dataSource

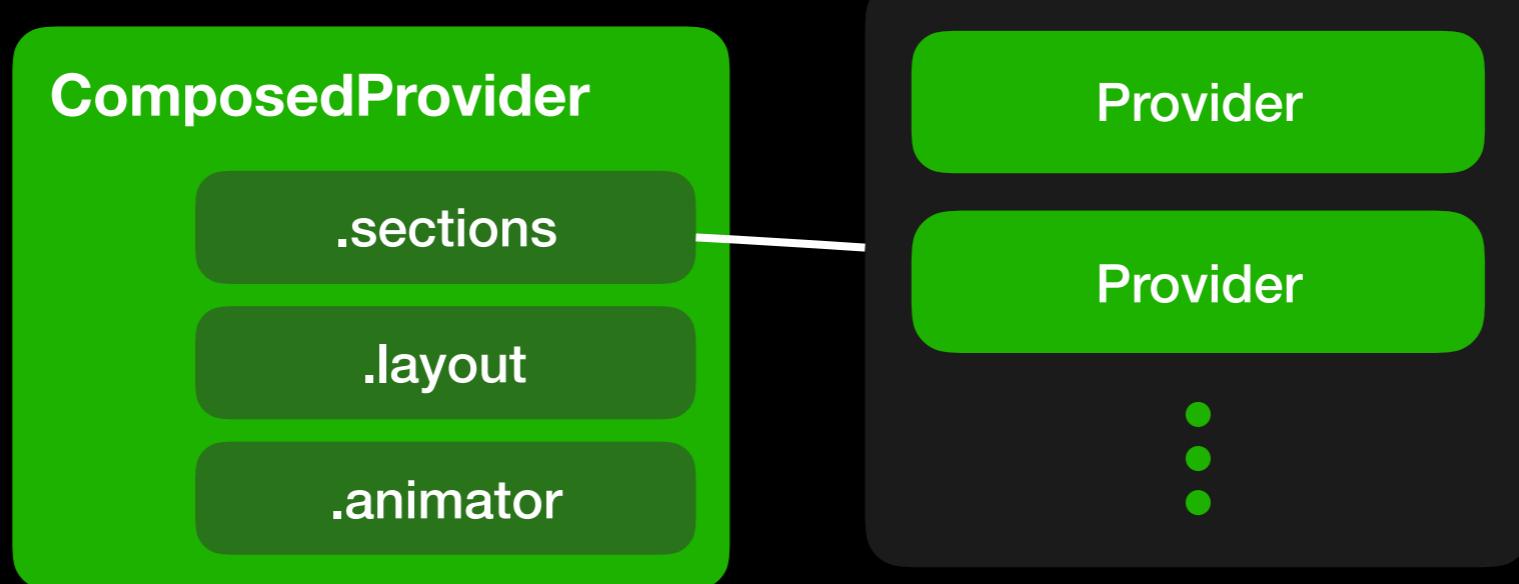
.viewSource

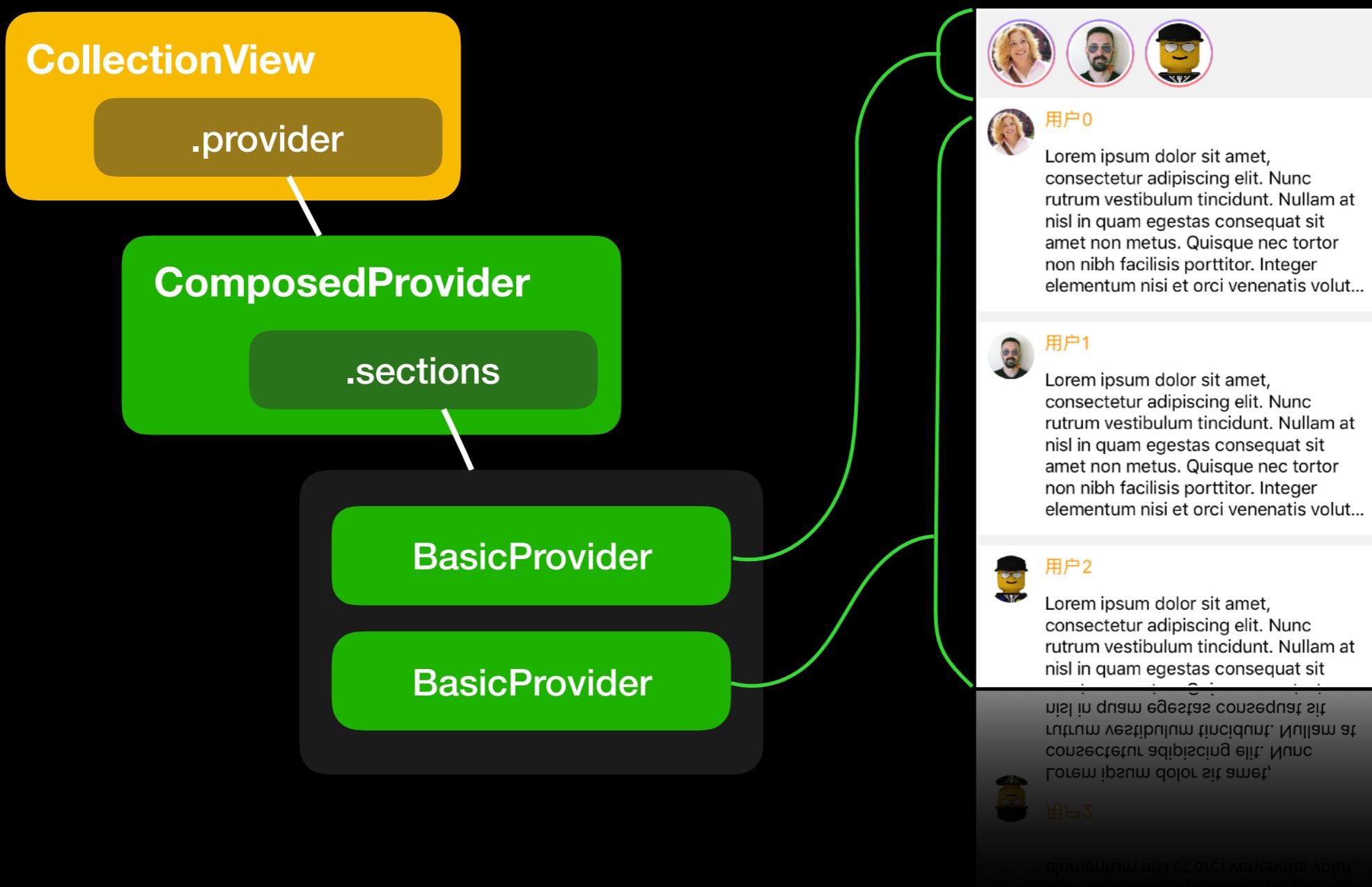
.sizeSource

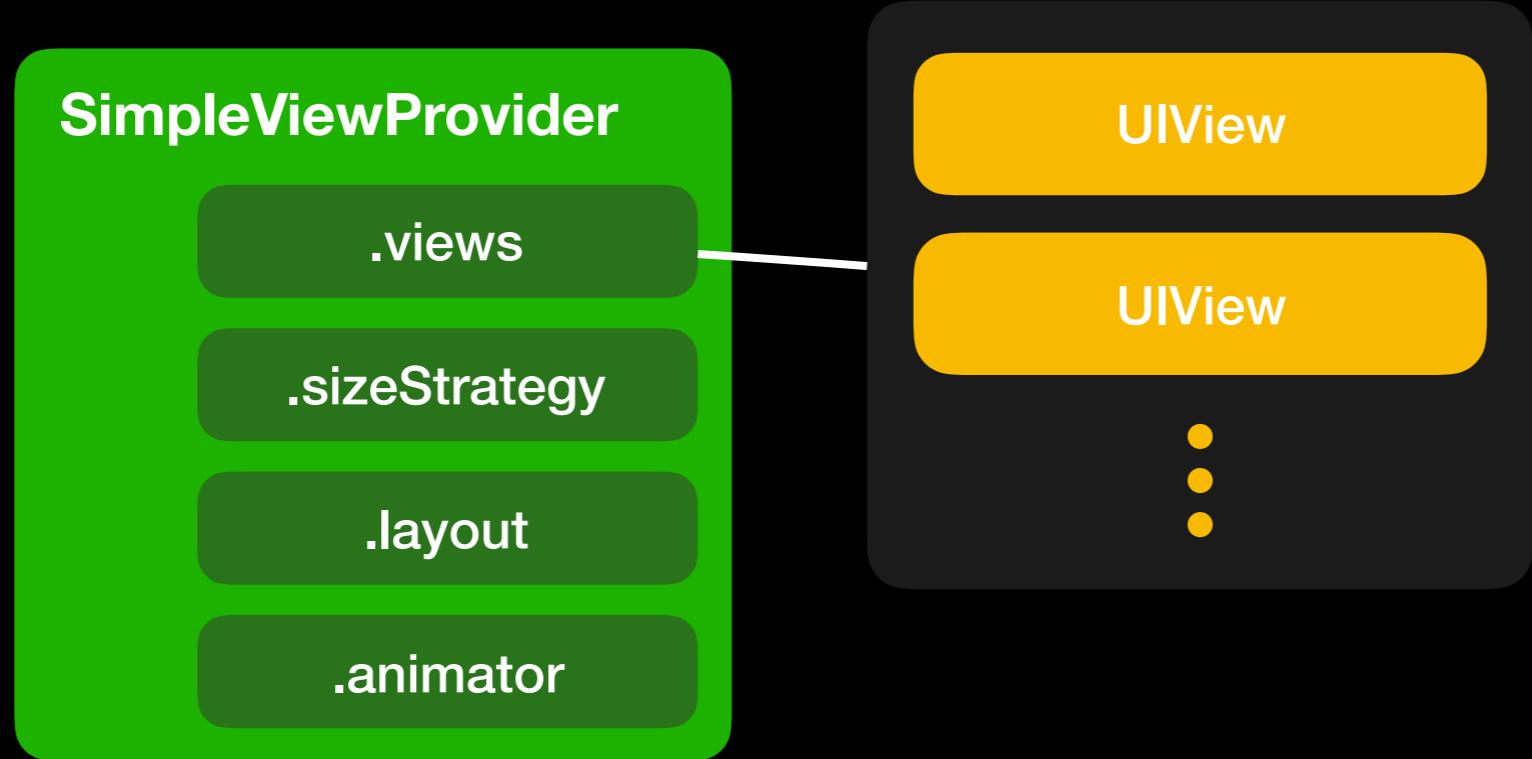
.layout

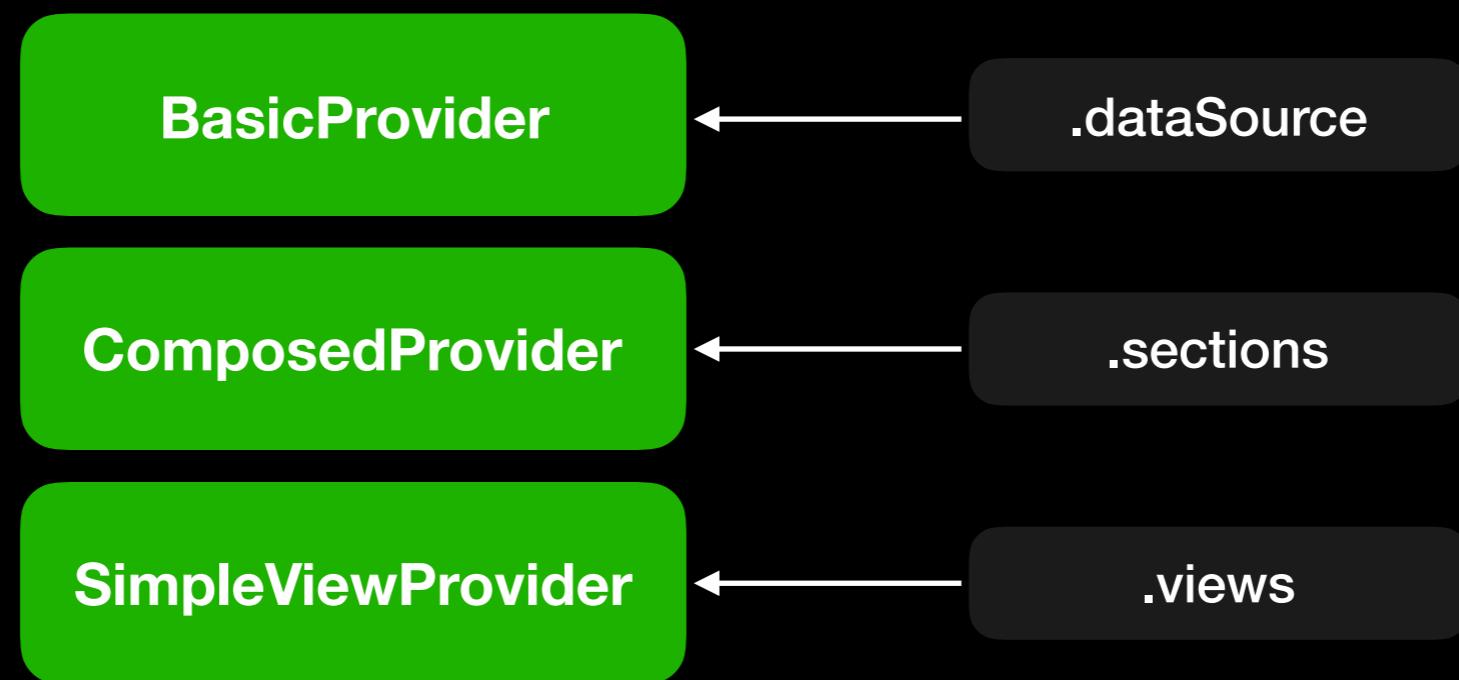
.animator

.tapHandler







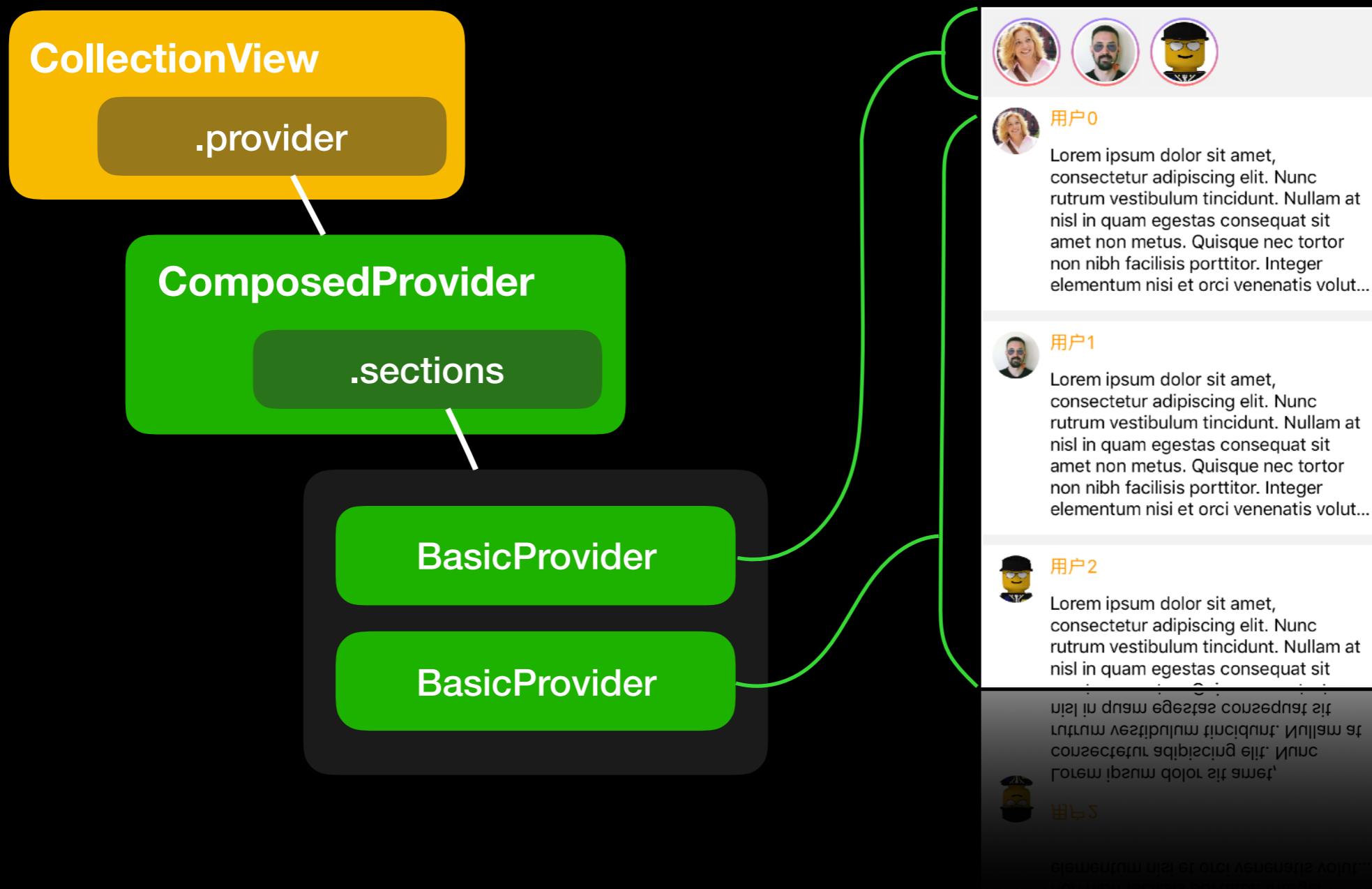


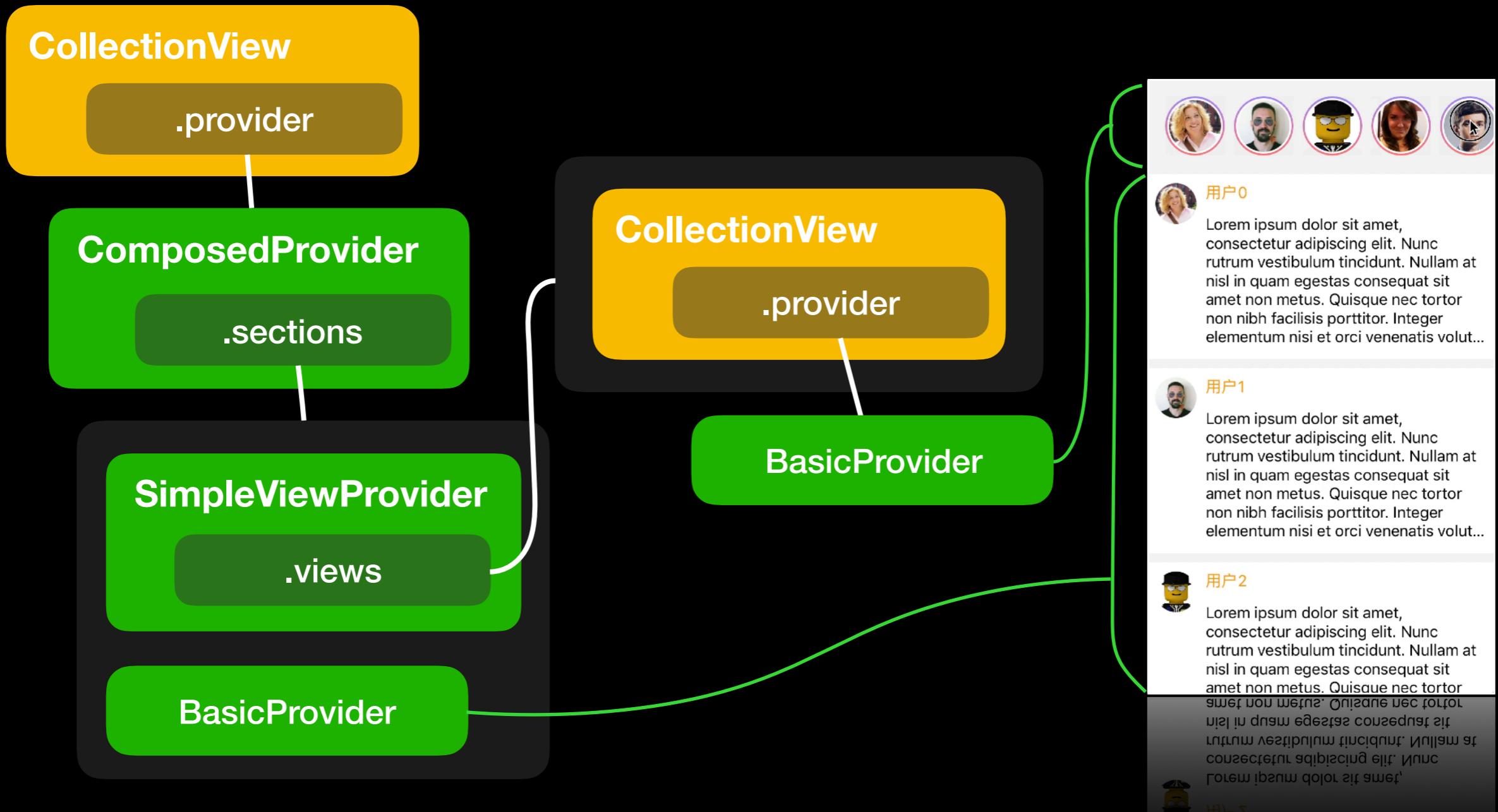
Let's make

The image shows a mobile application interface with a dark background. At the top, there is a row of five circular profile pictures. Below this, the screen is divided into three horizontal sections, each representing a user profile:

- 用户0**: This section includes a small profile picture of a woman, followed by the text "用户0" in orange, and a block of placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc rutrum vestibulum tincidunt. Nullam at nisl in quam egestas consequat sit amet non metus. Quisque nec tortor non nibh facilisis porttitor. Integer elementum nisi et orci venenatis volut...".
- 用户1**: This section includes a small profile picture of a man, followed by the text "用户1" in orange, and a block of placeholder text identical to the one above.
- 用户2**: This section includes a small profile picture of a person wearing a hat and sunglasses, followed by the text "用户2" in orange, and a block of placeholder text identical to the ones above.

At the very bottom of the screen, there is a dark footer bar with some very small, illegible text.





小结

- CollectionView
- BasicProvider, ComposedProvider, SimpleViewProvider
- 省去了
 - Cell Registration
 - Force Casting
 - 管理Section Indexes
- Delegate vs Declarative

Layout

FlowLayout

RowLayout

WaterfallLayout

InsetLayout

TransposeLayout

FlowLayout

FlowLayout

.spacing

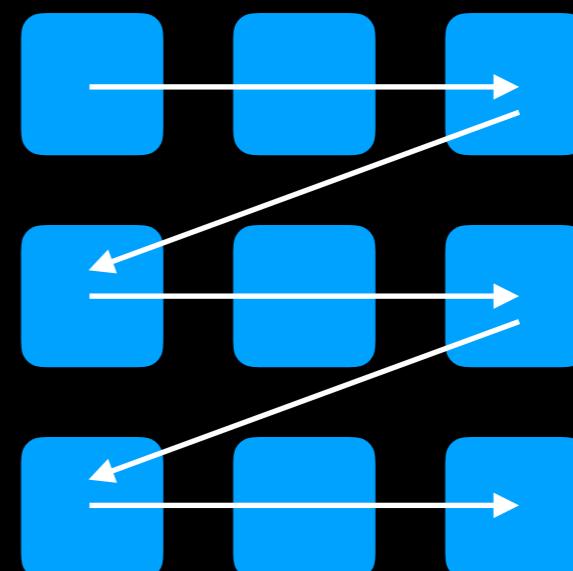
.lineSpacing

.interItemSpacing

.justifyContent

.alignItems

.alignContent



RowLayout

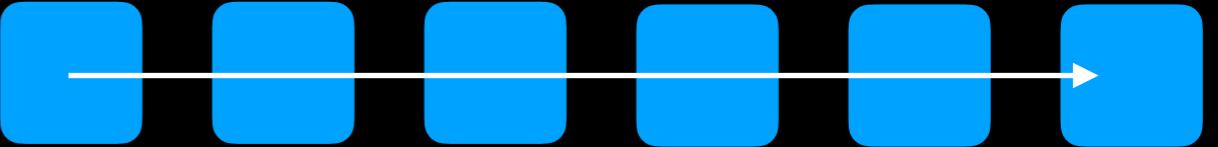
RowLayout

.spacing

.fillIdentifiers

.justifyContent

.alignItems



RowLayout

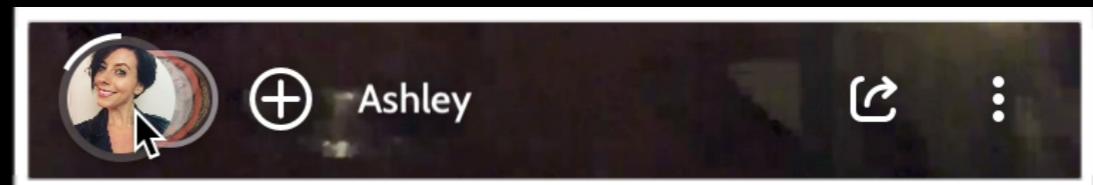
RowLayout

.spacing

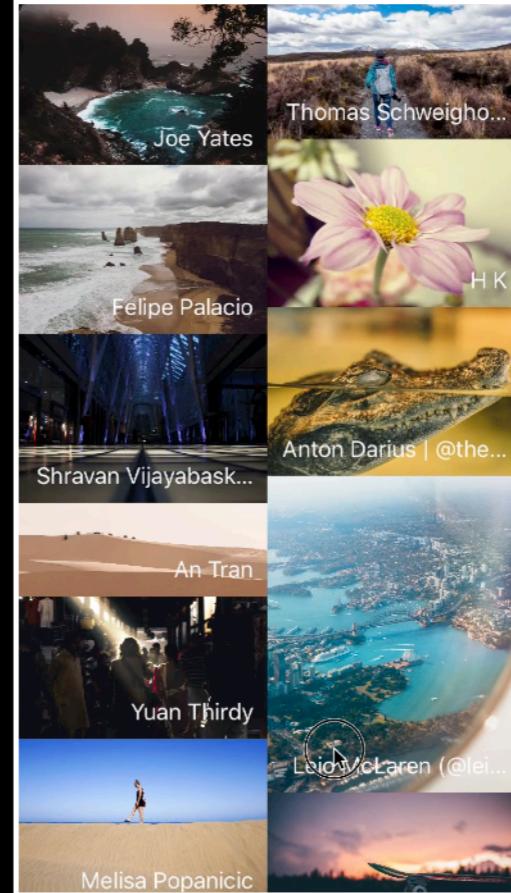
.fillIdentifiers

.justifyContent

.alignItems

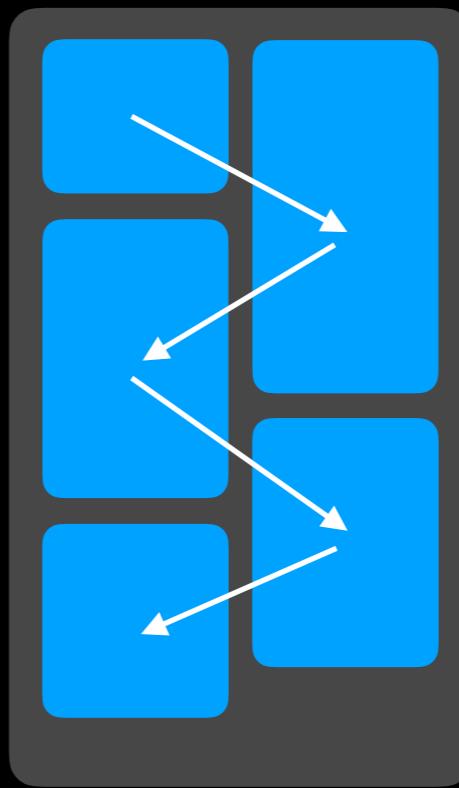
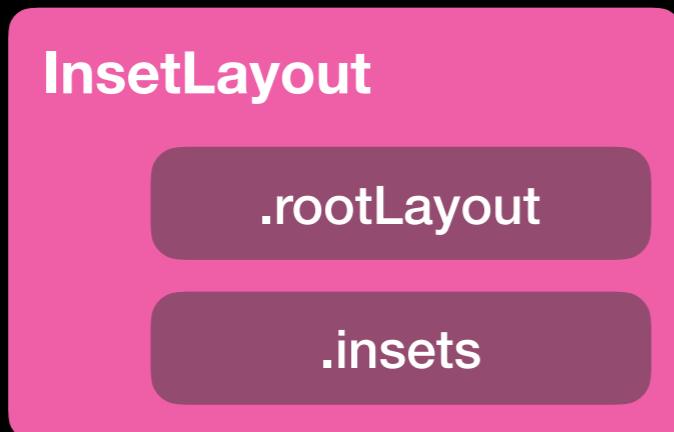


WaterfallLayout



WaterfallLayout(columns: 2, spacing: 10)
WaterfallLayout(columns: 2, spacing: 0)

InsetLayout



`InsetLayout(WaterfallLayout(columns: 2, spacing: 10),
 insets: UIEdgeInsets(top: 10, left: 10, bottom: 10, right: 10))`

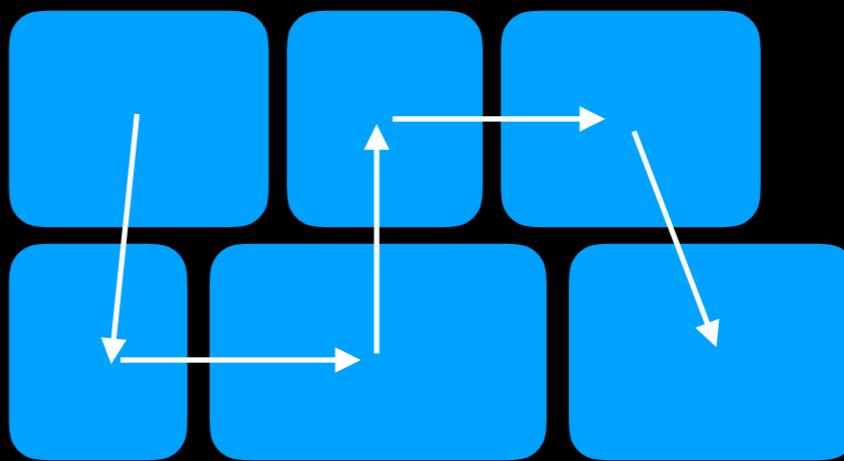
OR

`WaterfallLayout(columns: 2, spacing: 10)
.inset(by: UIEdgeInsets(top: 10, left: 10, bottom: 10, right: 10))`

TransposeLayout

TransposeLayout

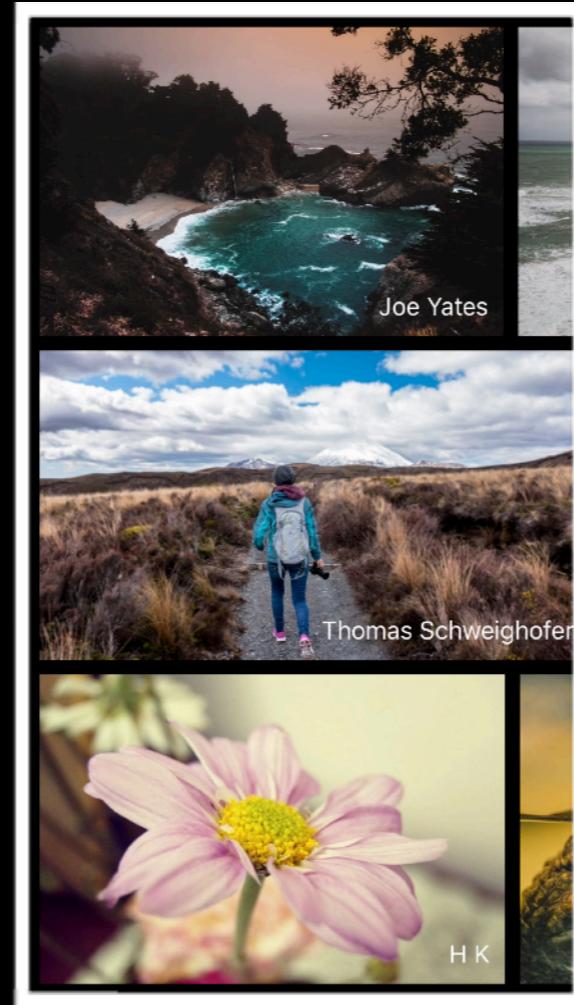
.rootLayout



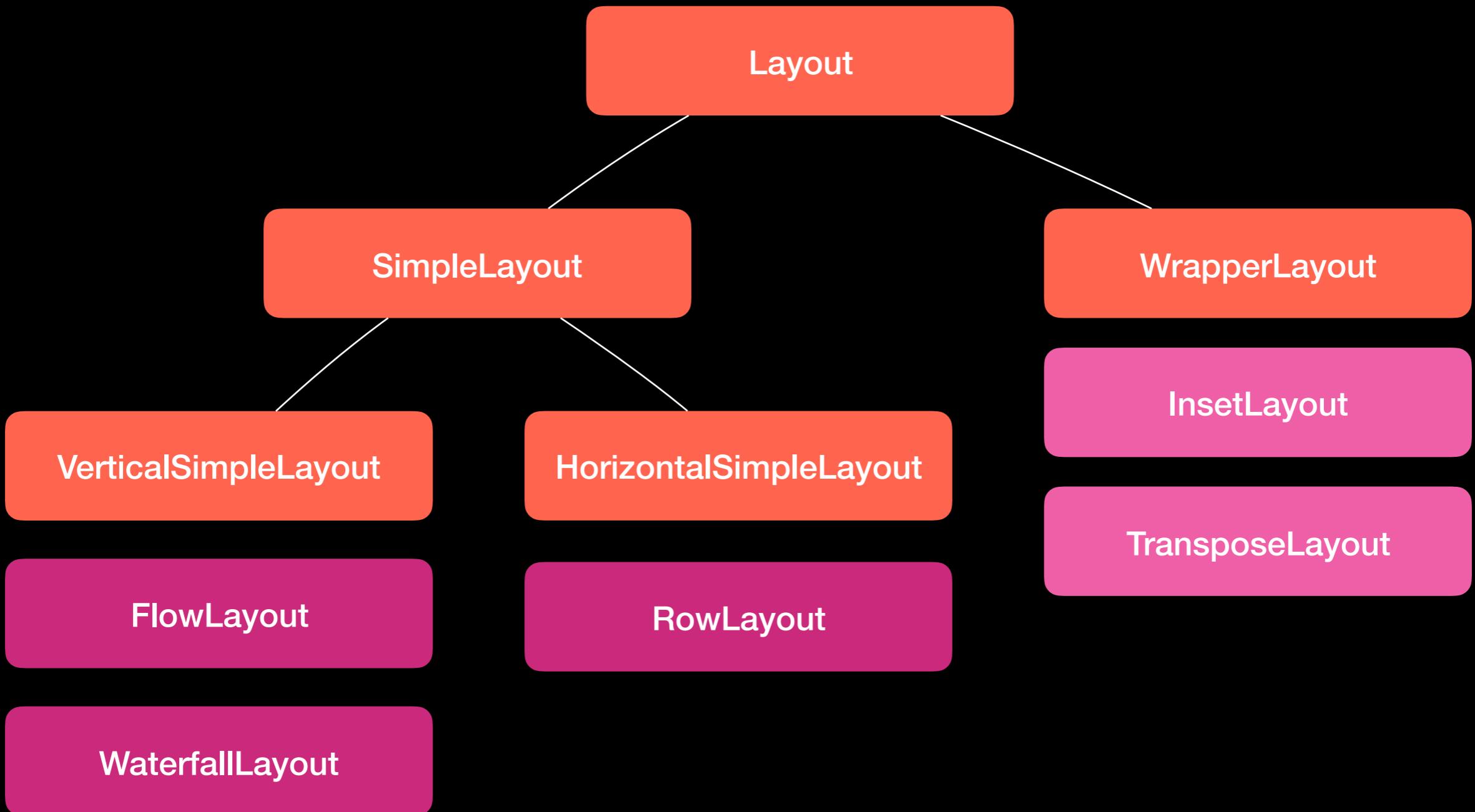
TransposeLayout(WaterfallLayout(columns: 2, spacing: 10))

OR

WaterfallLayout(columns: 2, spacing: 10).transposed()



```
InsetLayout( TransposeLayout( WaterfallLayout(columns: 3, spacing: 10) ) ,  
           insets: UIEdgeInsets(top: 10, left: 10, bottom: 10, right: 10) )
```



Layout

```
// 返回整个布局完成后的大小  
// 同 `collectionViewContentSize` 相似  
var contentSize: CGSize  
  
// 布局计算  
// 同 `prepare()` 相似  
func layout(context: LayoutContext)  
  
// 返回相对应Cell的Frame  
func frame(at: Int) -> CGRect  
  
// 返回在visibleFrame里面可见Cell的indexes  
// 同 `layoutAttributesForElements(in rect: CGRect)` 相似  
func visibleIndexes(visibleFrame: CGRect) -> [Int]
```

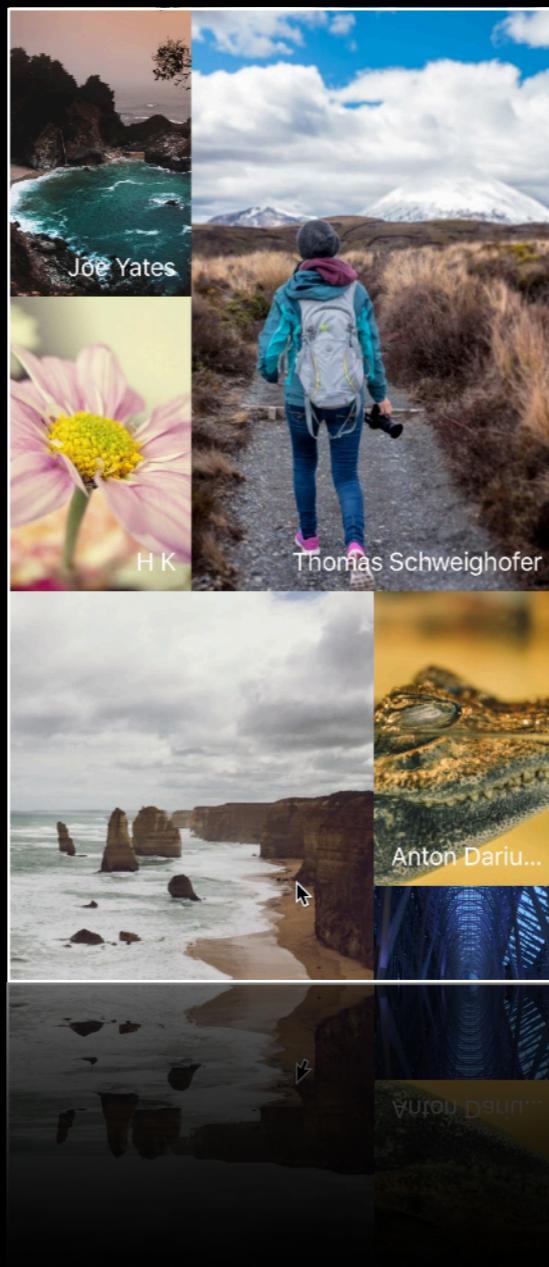
SimpleLayout

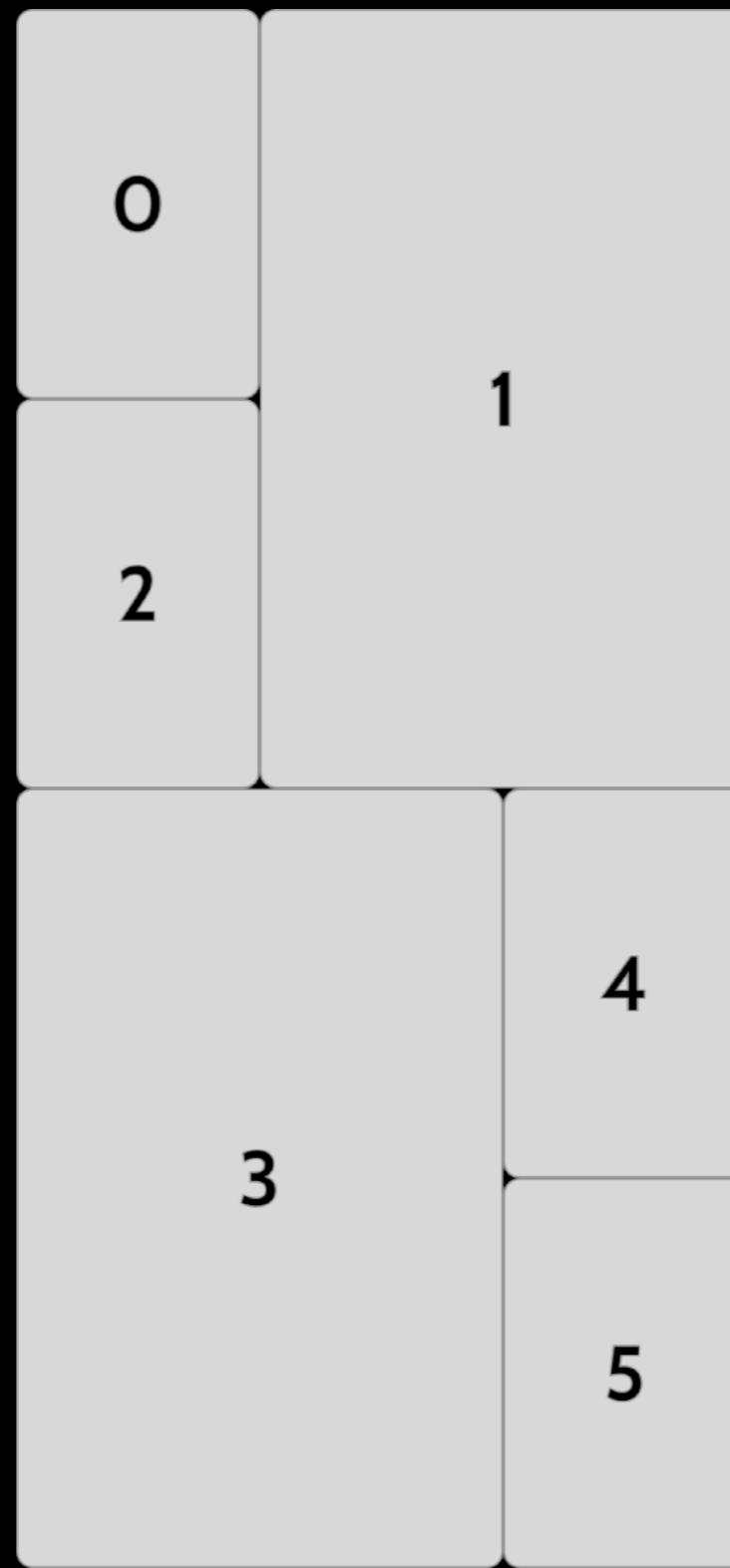
VerticalSimpleLayout

HorizontalSimpleLayout

// 根据`LayoutContext`, 返回所有`Cell`的`Frames`
`func simpleLayout(context: LayoutContext) -> [CGRect]`

Let's make





Layout 小结

- 上手快
- 每个Section都可以有不同的Layout
- 简化了布局实现
- 可以通过Helper Layout来完成一些修改

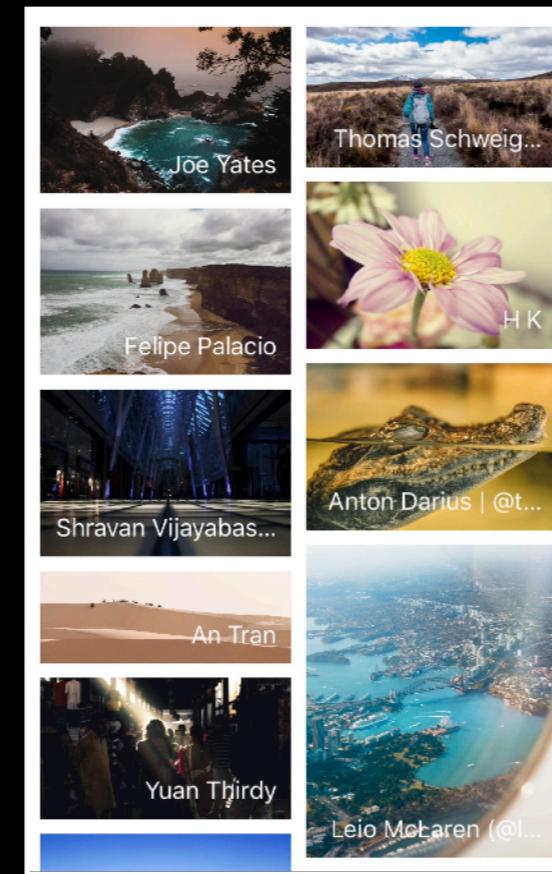
Break

5 min

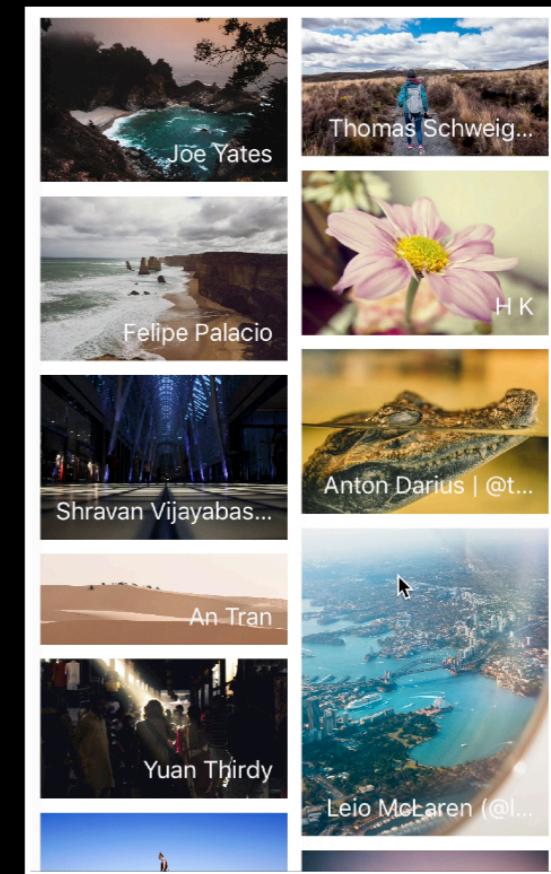
Animator



Insertion / Deletion



Move

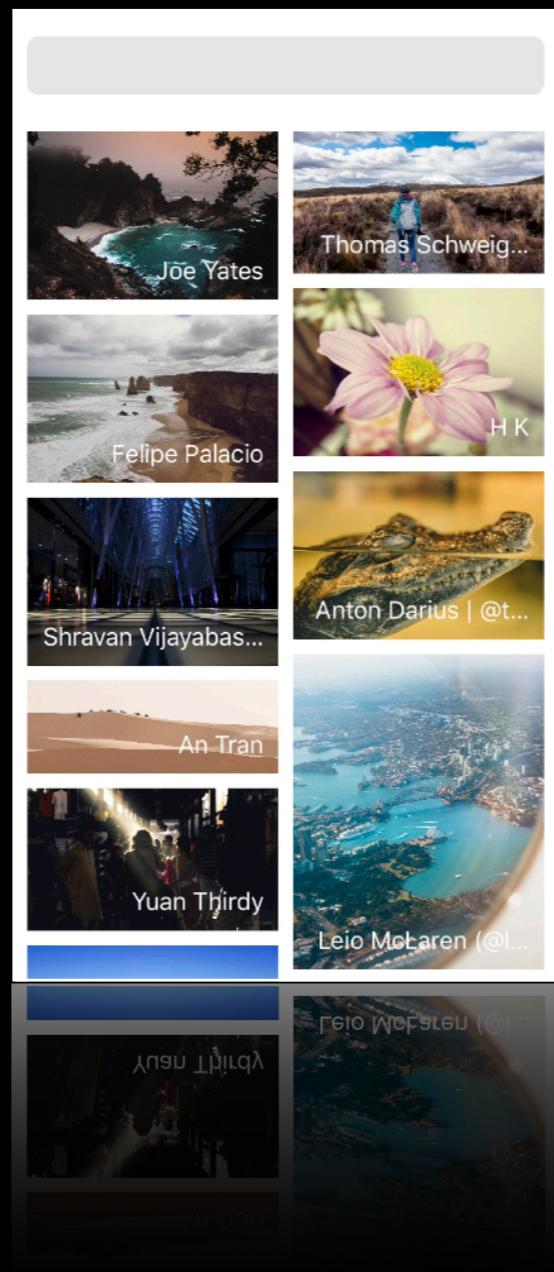


Scroll

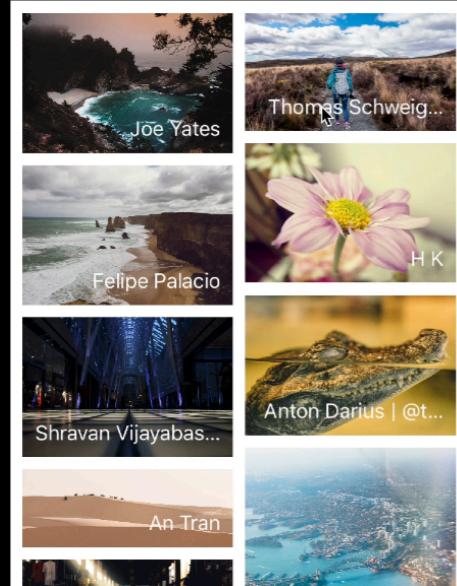
Animator

```
/// Called when CollectionView inserts a view into its subviews.  
///  
/// Perform any insertion animation when needed  
///  
func insert(collectionView: UICollectionView, view: UIView,  
    at: Int, frame: CGRect)  
  
/// Called when CollectionView deletes a view from its subviews.  
///  
/// Perform any deletion animation, then  
/// call `view.recycleForCollectionKitReuse()`  
/// after the animation finishes  
///  
func delete(collectionView: UICollectionView, view: UIView)  
  
/// Called when:  
/// * the view has just been inserted  
/// * the view's frame changed after `reloadData`  
/// * the view's screen position changed when user scrolls  
///  
func update(collectionView: UICollectionView, view: UIView,  
    at: Int, frame: CGRect)
```

Let's make



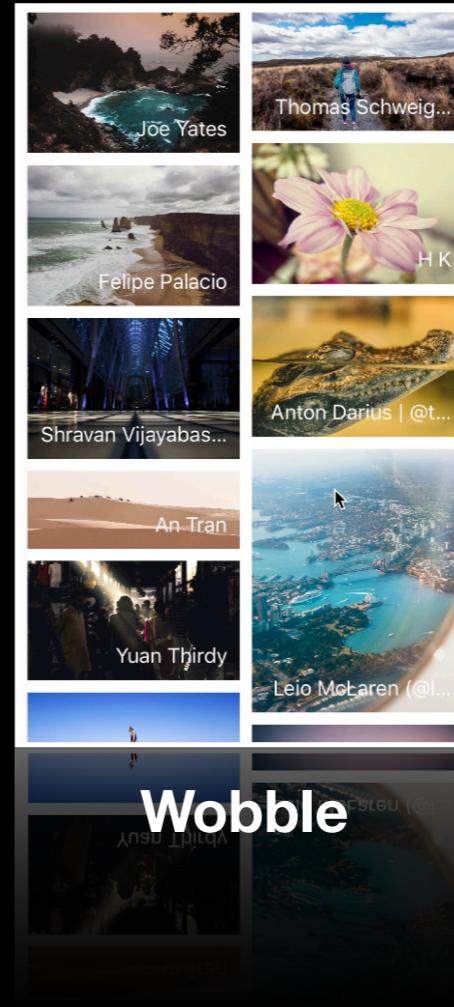
Animator



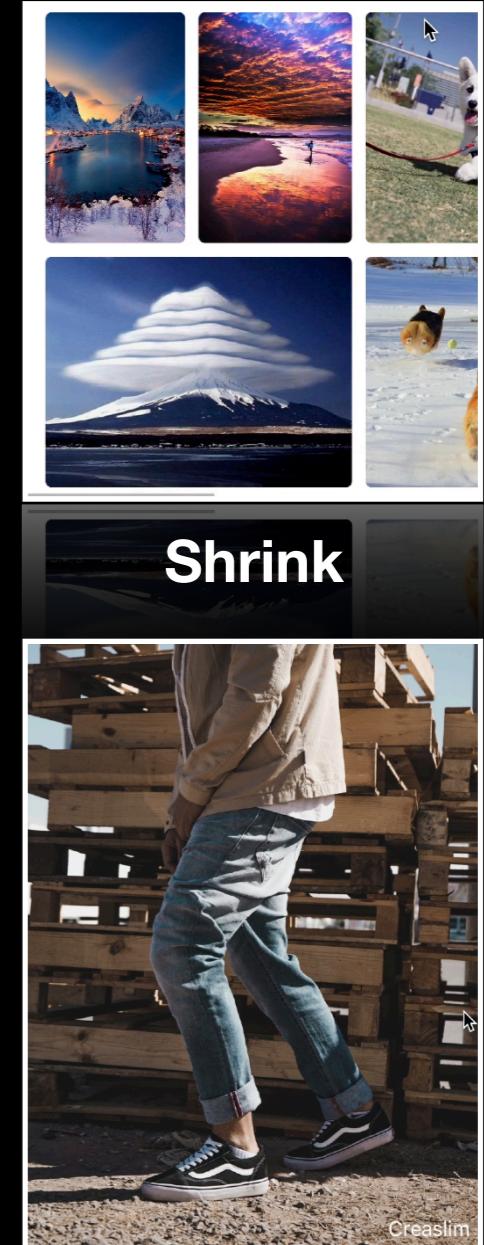
Overscroll



Rotate



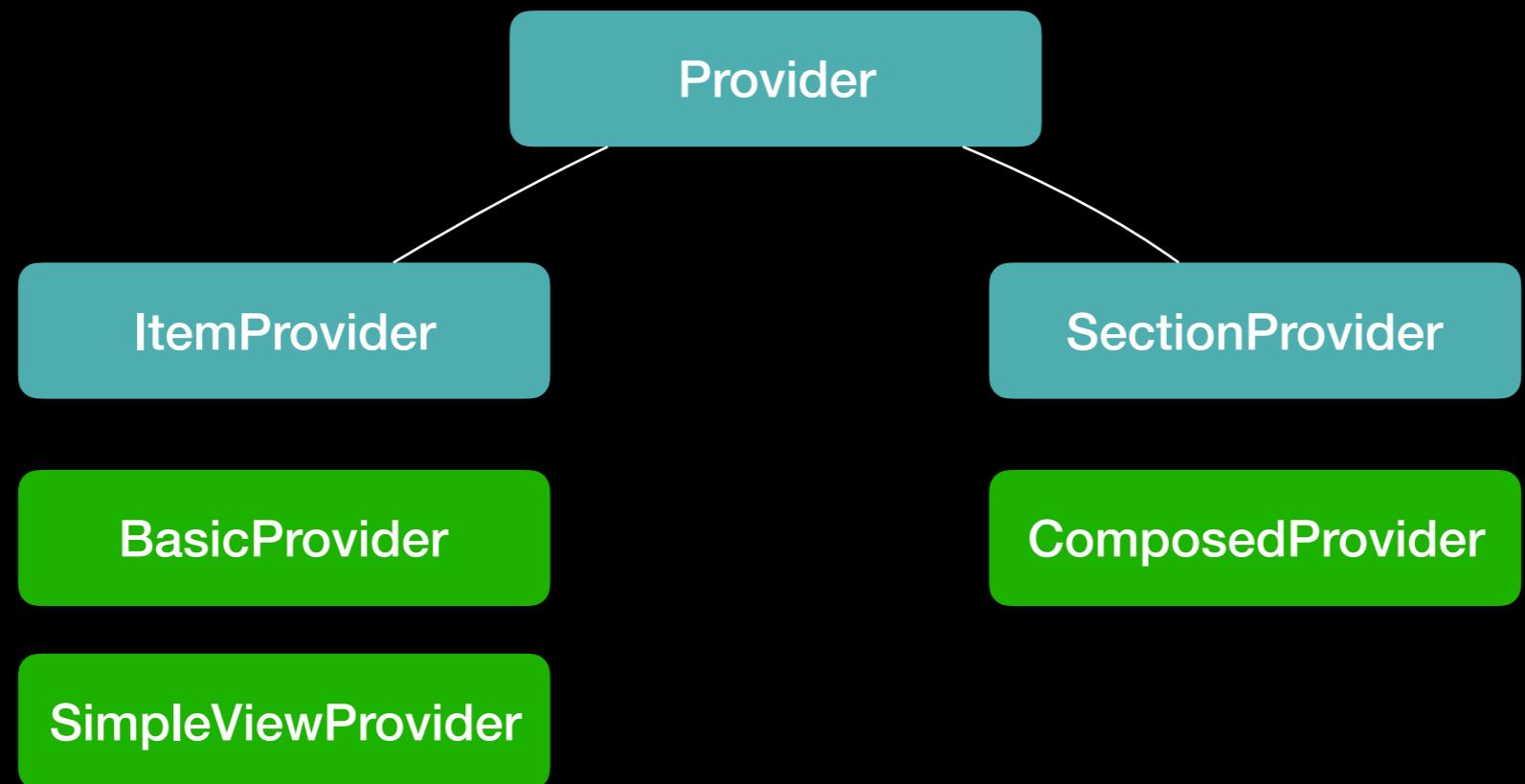
Wobble



Shrink



Shrink



Provider

```
protocol Provider {
    var identifier: String? { get }

    // data
    var numberofItems: Int { get }
    func identifier(at: Int) -> String

    // layout
    func layout(collectionSize: CGSize)
    func visibleIndexes(visibleFrame: CGRect) -> [Int]
    var contentSize: CGSize { get }
    func frame(at: Int) -> CGRect

    // ....
}
```

ItemProvider

```
public protocol ItemProvider: Provider {  
    func view(at: Int) -> UIView  
    func update(view: UIView, at: Int)  
  
    func didTap(view: UIView, at: Int)  
}
```

SectionProvider

```
public protocol SectionProvider: Provider {  
    func section(at: Int) -> Provider?  
}
```

ItemProvider

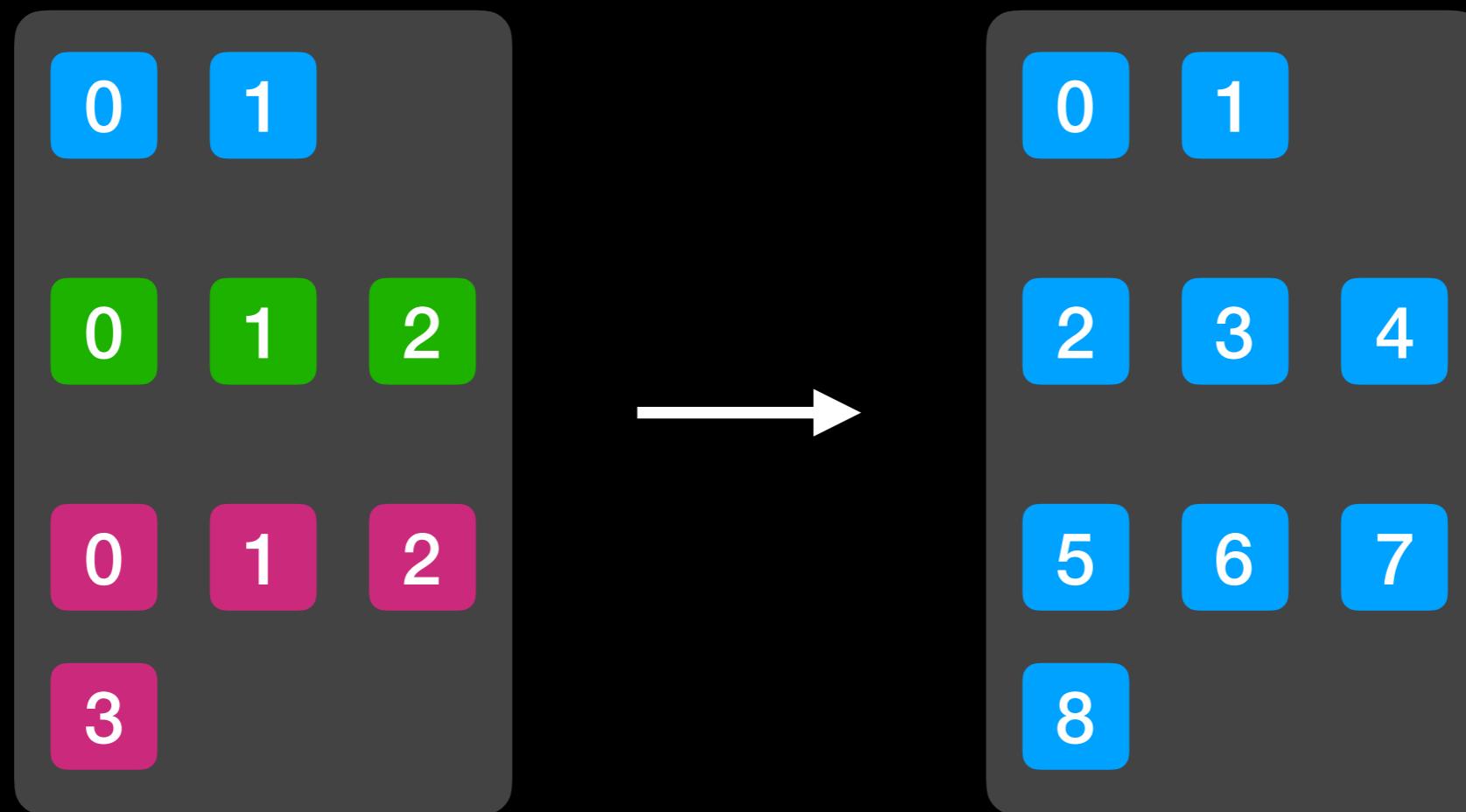
```
protocol Provider {
    func flattenedProvider() -> ItemProvider
    // ....
}

extension ItemProvider {
    public func flattenedProvider() -> ItemProvider {
        return self
    }
}
```

SectionProvider

```
extension SectionProvider {
    public func flattenedProvider() -> ItemProvider {
        return FlattenedProvider(provider: self)
    }
}
```

Flattening



ComposedProvider
(SectionProvider)

FlattenedProvider

ReloadData

- 1. Flatten the provider**
 - `flattenedProvider = provider.flattenedProvider()`
- 2. Ask the provider to layout**
 - `flattenedProvider.layout(collectionSize:)`
- 3. Retrieve a list of visible indexes**
 - `flattenedProvider.visibleIndexes(visibleFrame:)`
- 4. Calculate a diff between old visible cells and new visible cells**
- 5. Remove deleted cells**
- 6. Insert newly added cells**

Scrolling

When `contentOffset` changed

1. Flatten the provider

- `flattenedProvider = provider.flattenedProvider()`

2. Ask the provider to layout

- `flattenedProvider.layout(collectionSize:)`

1. Retrieve a list of visible indexes

- `flattenedProvider.visibleIndexes(visibleFrame:)`

2. Calculate a diff between old visible cells and new visible cells

3. Remove deleted cells

4. Insert newly added cells

CollectionKit的前身

[Ikzhao / MCollectionView](#)

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. Edit Add topics

153 commits 5 branches 0 releases 3 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

| File | Commit Message | Date |
|-----------------------------|-------------------------------|---------------------------------------|
| .gitignore | Add Pods/ in gitignore | 2 years ago |
| LICENSE | Initial Commit | 3 years ago |
| Cartfile | support carth | a year ago |
| Cartfile.resolved | update yaal to 1.1.0 | a year ago |
| Sources | bump version | a year ago |
| MCollectionView.xcworkspace | reorganize project structure | 2 years ago |
| MCollectionViewTests | new visible index calculation | 2 years ago |
| MCollectionView.xcodeproj | support carth | a year ago |
| Examples | update yaal to 1.1.0 | a year ago |
| Carthage/Checkouts | update yaal to 1.1.0 | a year ago |
| Ikzhao bump version | | Latest commit b20165a on Jul 11, 2017 |

Carrier 8:19 PM Examples Chat

Vivamus et fermentum diam. Suspendisse vitae tempor lectus.

Etiam a leo nibh. Fusce cursus metus viverra erat viverra, sed efficitur magna consequat. Ut tristique magna et sapien euismod, consequat maximus ipsum varius.

Suspendisse ut turpis velit.

Duis eros eros

You can also drag and drop these message cells to reorder them! 😊

Type here

Delivered

The End

Thanks for joining