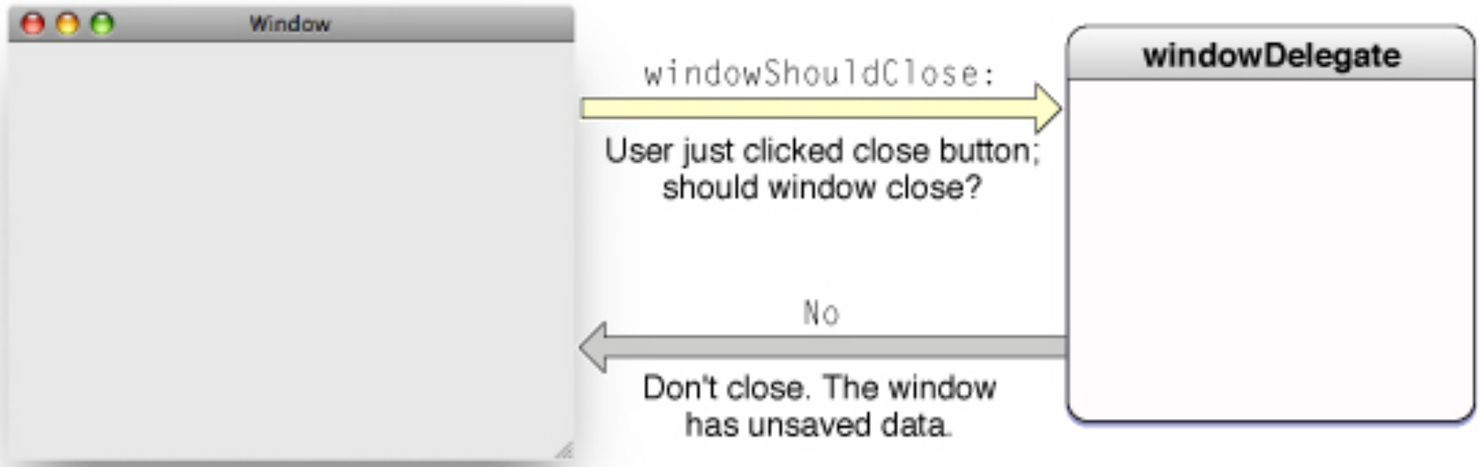
UITableView

&

UICollectionView

Delegates e Data Sources

- Pattern para um objeto avisar (ou perguntar) para outro sobre algum evento



Por que?

- Classes de interface podem ser genéricas

"Favor composition over inheritance"

weak

- Propriedades que fazem o papel de delegate geralmente devem ser `weak`
- Evitar referência cíclica

```
unowned(unsafe) var delegate: UITableViewDelegate?
```

Infelizmente, o UIKit não usa `weak`, mas sim `unowned(unsafe)` (equivalente ao `assign` em Objective-C)

UITableView

- Lista vertical de itens em uma coluna (células)
- Células são `UITableViewCell`
- Herda de `UIScrollView`
- Índices representados por `NSIndexPath`

```
let indexPath = NSIndexPath(forRow: 2, inSection: 0)
indexPath.row     // 2
indexPath.section // 0
```

`UITableViewDelegate`

- Qual o tamanho da célula do índice X?
- A célula do índice X deve ser *highlighted*?
- O que deve acontecer quando uma célula for selecionada?
- ... E mais **30 métodos**!
- Felizmente, todos são `optional`

`UITableViewDataSource`

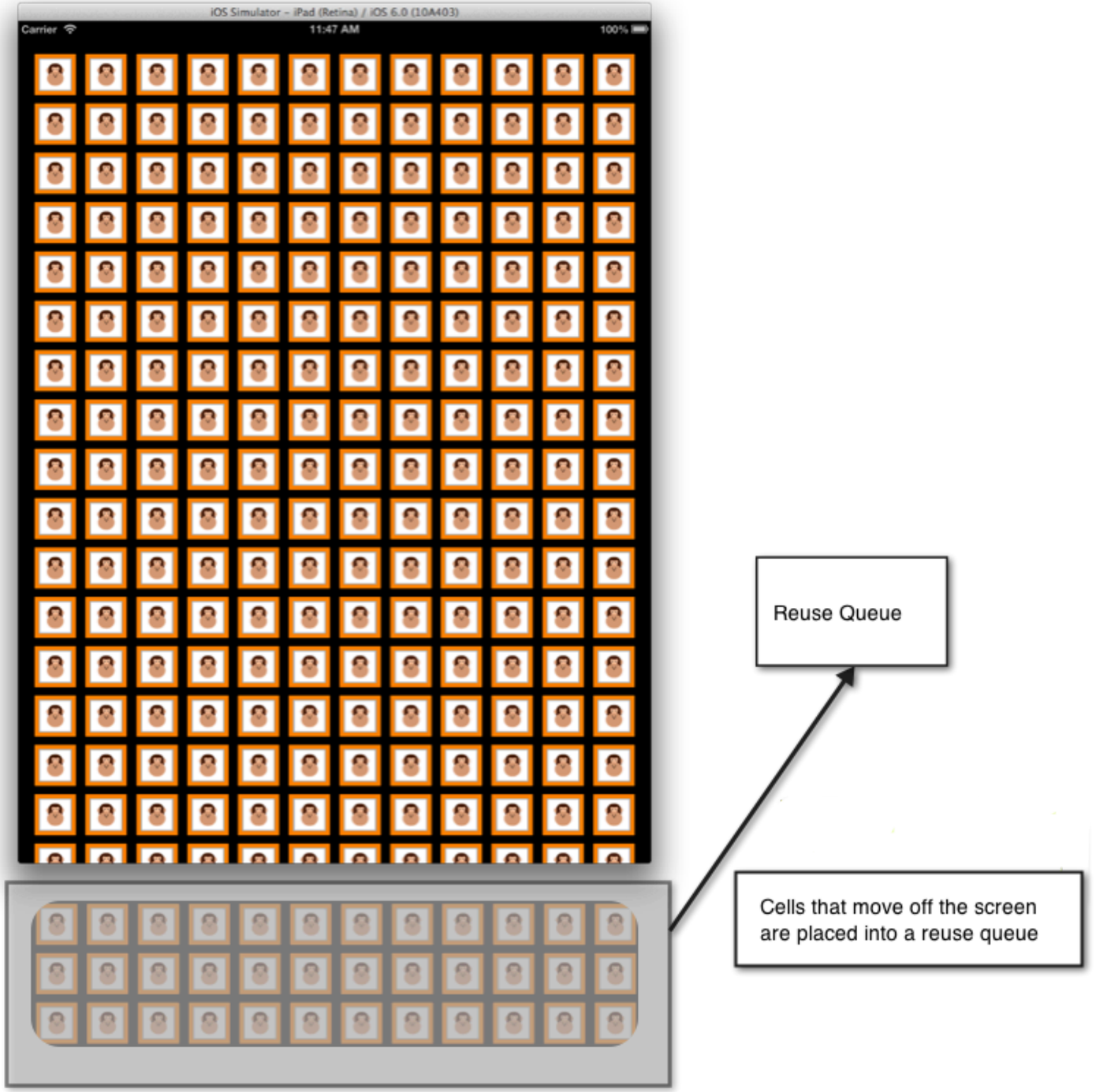
- Quantas seções têm na tabela?
- Quantas linhas tem em uma seção específica?
- Qual é a `UITableViewCell` do índice X?

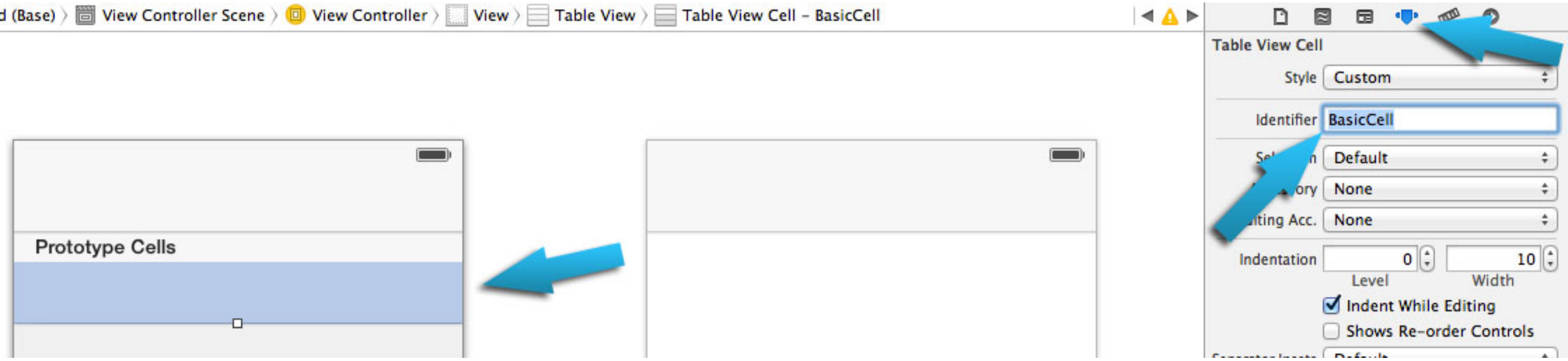
```
override func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return 10
}

override func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCellWithIdentifier("Cell", forIndexPath: indexPath) as! UITableViewCell
    cell.textLabel?.text = "\(indexPath.section) - \(indexPath.row)"

    return cell
}
```

dequeueReusableCellWithIdentifier

Reuso de células

- Conforme as células saem da tela, elas ficam em uma fila de reuso
- Ao invés de criar uma célula nova, uma antiga é desenfileirada e reusada
- Evita travadas, já que criar células novas é caro
- **Maior fonte de bugs para iniciantes!**
- Células (muito) diferentes devem ter *identifiers* diferentes


iOS Simulator – iPad (Retina) / iOS 6.0 (10A403)
Carrier
11:47 AM
100%
Reuse Queue
Cells that move off the screen are placed into a reuse queue


d (Base)
View Controller Scene
View Controller
View
Table View
Table View Cell – BasicCell
Prototype Cells
Table View Cell
Style
Custom
Identifier
BasicCell
Default
None
None
Indentation
0
10
Level
Width
Indent While Editing
Shows Re-order Controls

Identifiers são *String-typed* 😱

Erros apenas em tempo de execução 🔥

Natalie

https://github.com/krzyzanowskim/Natalie

Natalie

- Gerador de código relacionado ao uso de Storyboards
- Evitar o uso de Strings

```
let identifier = Reusable.Cell.identifier!
let cell = tableView.dequeueReusableCellWithIdentifier(identifier,
    forIndexPath: indexPath) as! UITableViewCell
```

- `Storyboards.swift`
- Já foi adicionado pelo `liftoff`! 👏

`brew reinstall natalie --HEAD`

(Não precisa executar, já instalamos para vocês!)

Forçar update

- A table view não consegue saber sozinha quando os dados mudam
- Deve ser avisada para recarregar os dados

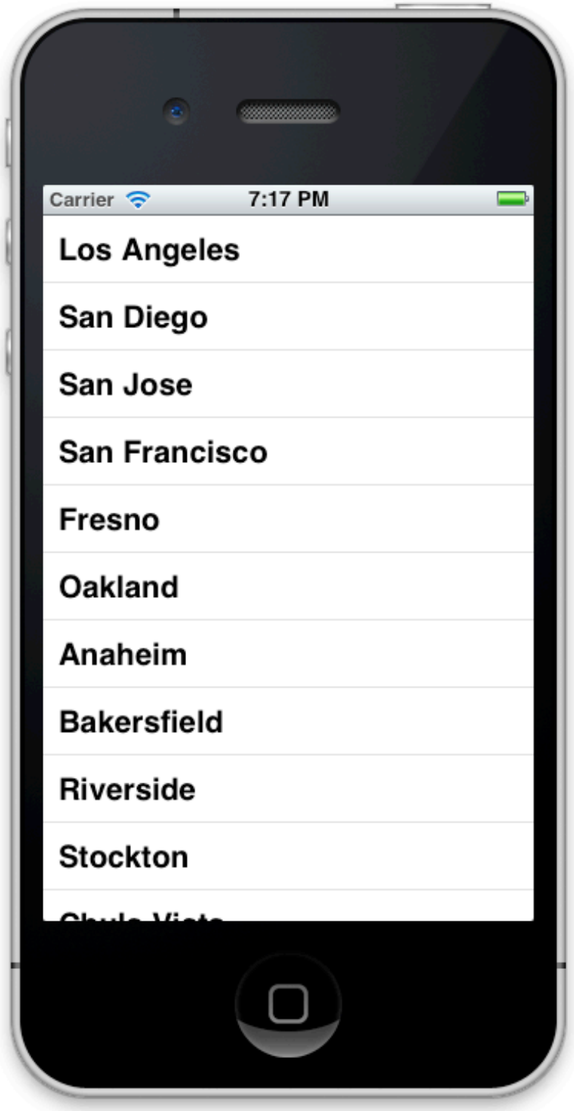
```
tableView.reloadData()

let indexPaths = [NSIndexPath(forRow: 0, inSection: 0)]
tableView.reloadRowsAtIndexPaths(indexPaths, withRowAnimation: .Automatic)

let sections = NSIndexSet(indexesInRange: NSRange(location: 0,length: 2))
tableView.reloadSections(sections, withRowAnimation: .Automatic)
```

UITableViewCell

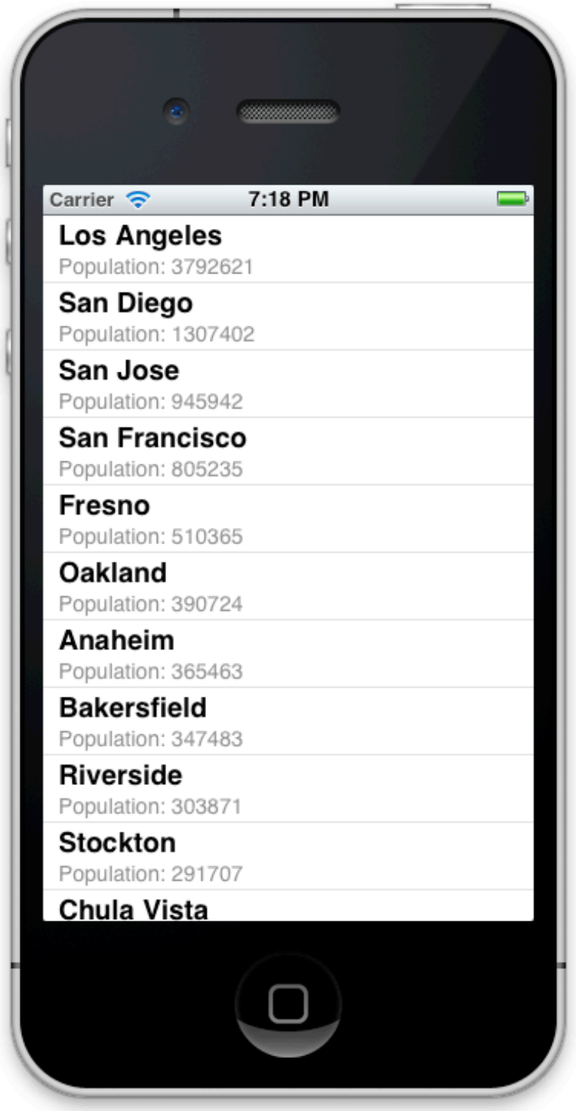
- Geralmente são feitas subclasses

```
override func awakeFromNib() {
    super.awakeFromNib()
}

override func prepareForReuse() {
    super.prepareForReuse()
}
```

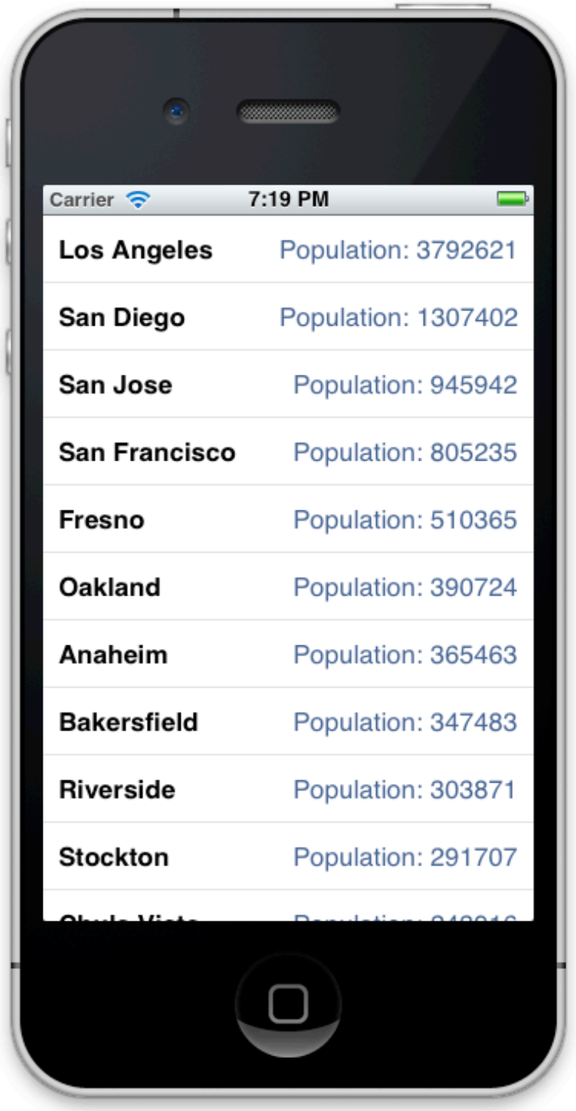
- Outlets privados, com método para "popular"



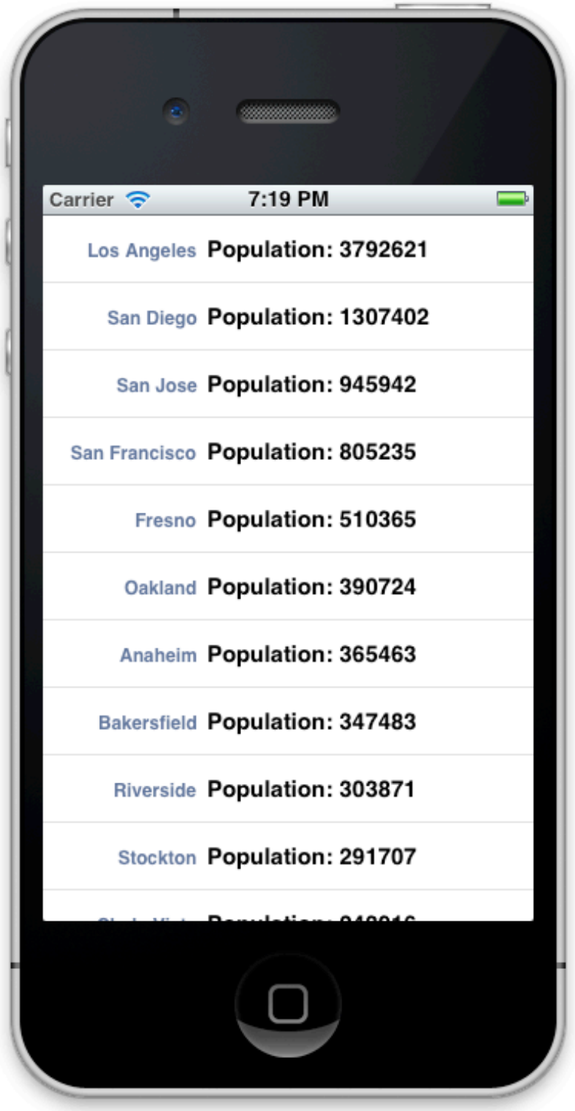
UITableViewCellStyleDefault



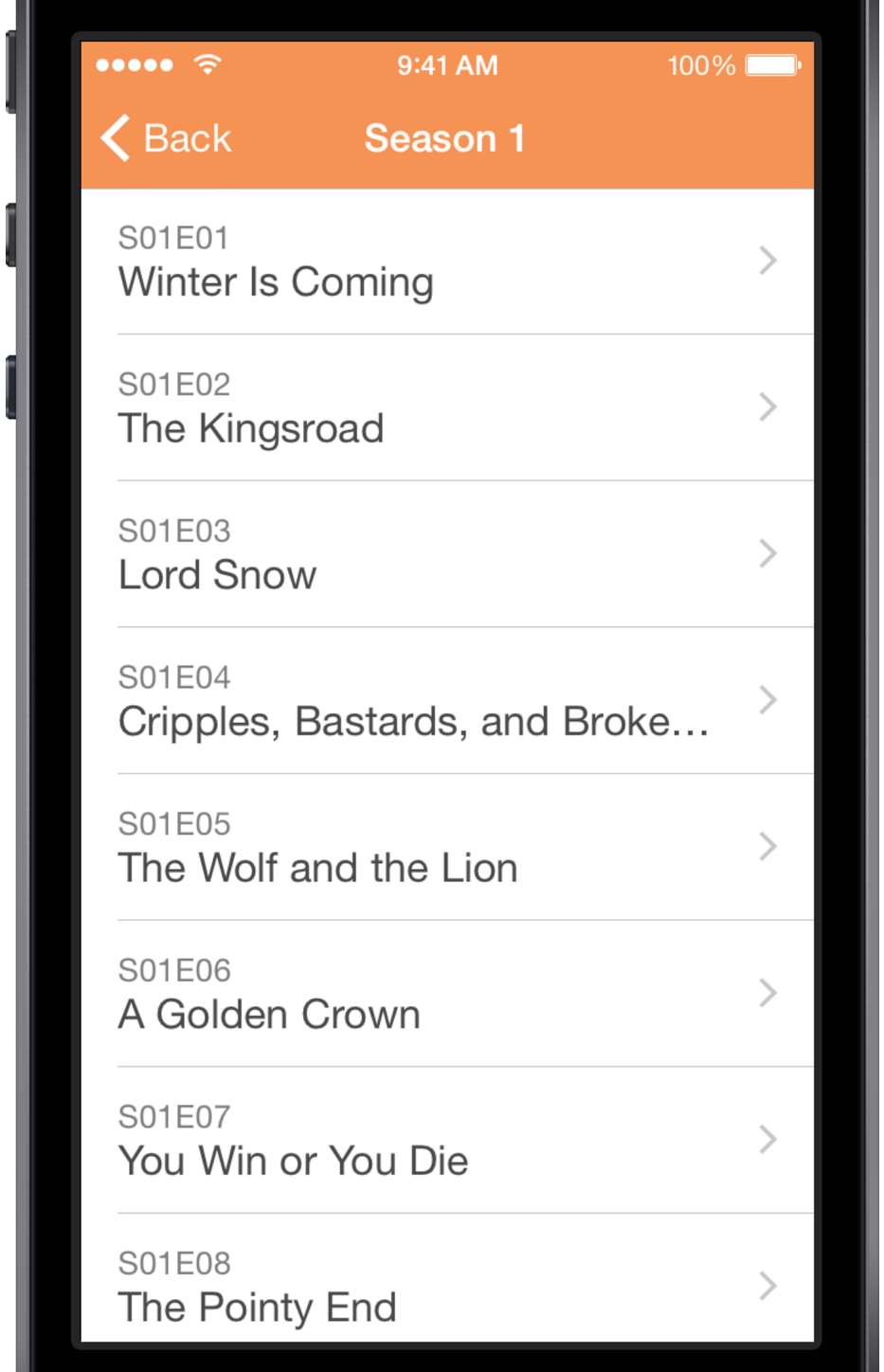
UITableViewCellStyleSubtitle



UITableViewCellStyleValue1



UITableViewCellStyleValue2


9:41 AM
100%
Back
Season 1
S01E01
Winter Is Coming
S01E02
The Kingsroad
S01E03
Lord Snow
S01E04
Cripples, Bastards, and Broke…
S01E05
The Wolf and the Lion
S01E06
A Golden Crown
S01E07
You Win or You Die
S01E08
The Pointy End

`UITableViewController`

- View Controller que já gerencia uma table view
- A `view` do View Controller já é a table view
- O View Controller é o `delegate` e `dataSource` da table view
- Pull to refresh
- **Conveniência**

View Controller – A controller that supports the fundamental view-management model in iPhone OS.

inherits from

Table View Controller – A controller that manages a table view.

implements

UITableViewDelegate - Implementing a delegate allows your class to respond to events occuring in another object. For example, in this case you can respond to a row selection in a table view.

UITableViewDatasource - A datasource is very much like a delegate. However, it is delegated control of data. A table view is driven by an underlying data model and the datasource manages that relationship.

manages & contains

Table View – Displays data in a list of plain, sectioned, or grouped rows.

manages & contains

Table View Cell – Defines the attributes and behavior of cells (rows) in a table view.

UICollectionView

- Generalização da table view
- Mais recente (iOS 6)

https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/UIKitUICatalog/UICollectionView.html

Edit

World Clock

Cupertino

Today

New York

Today, 3 hours ahead

Paris

Today, 9 hours ahead

Beijing

Tomorrow, 15 hours ahead

Tokyo

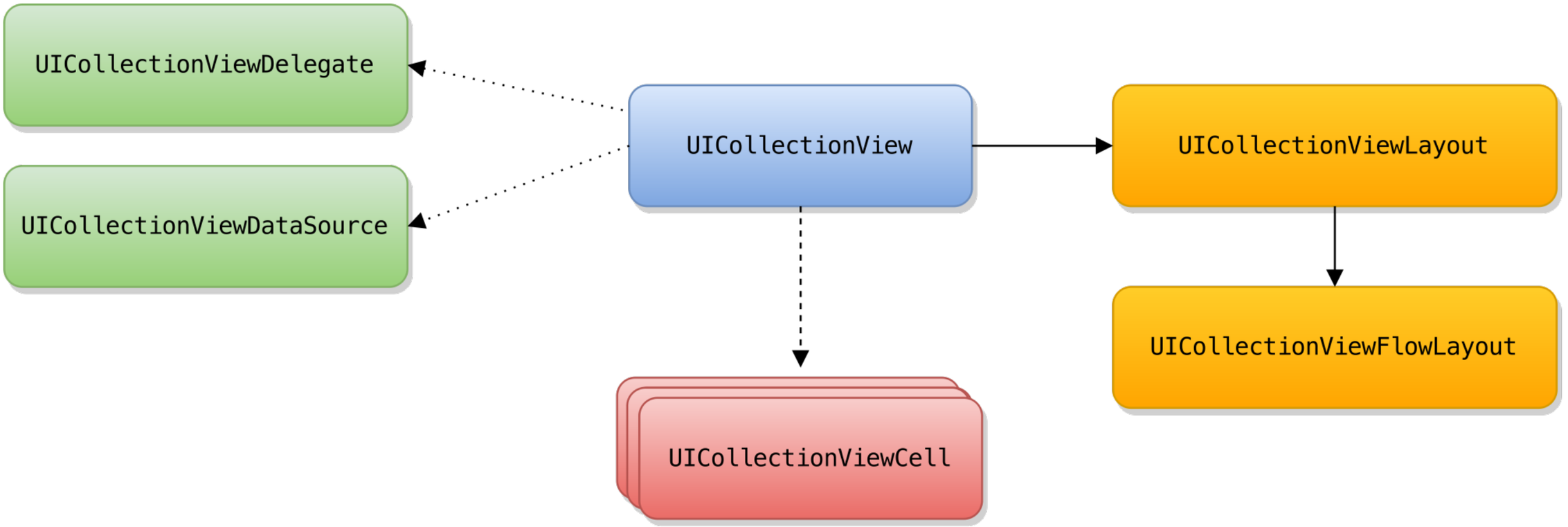
Tomorrow, 16 hours ahead

Moscow

Tomorrow, 11 hours ahead

Arquitetura

- Mesmos princípios da `UITableView`
- Bem mais flexível e customizável
- ... Mas faremos só o básico aqui

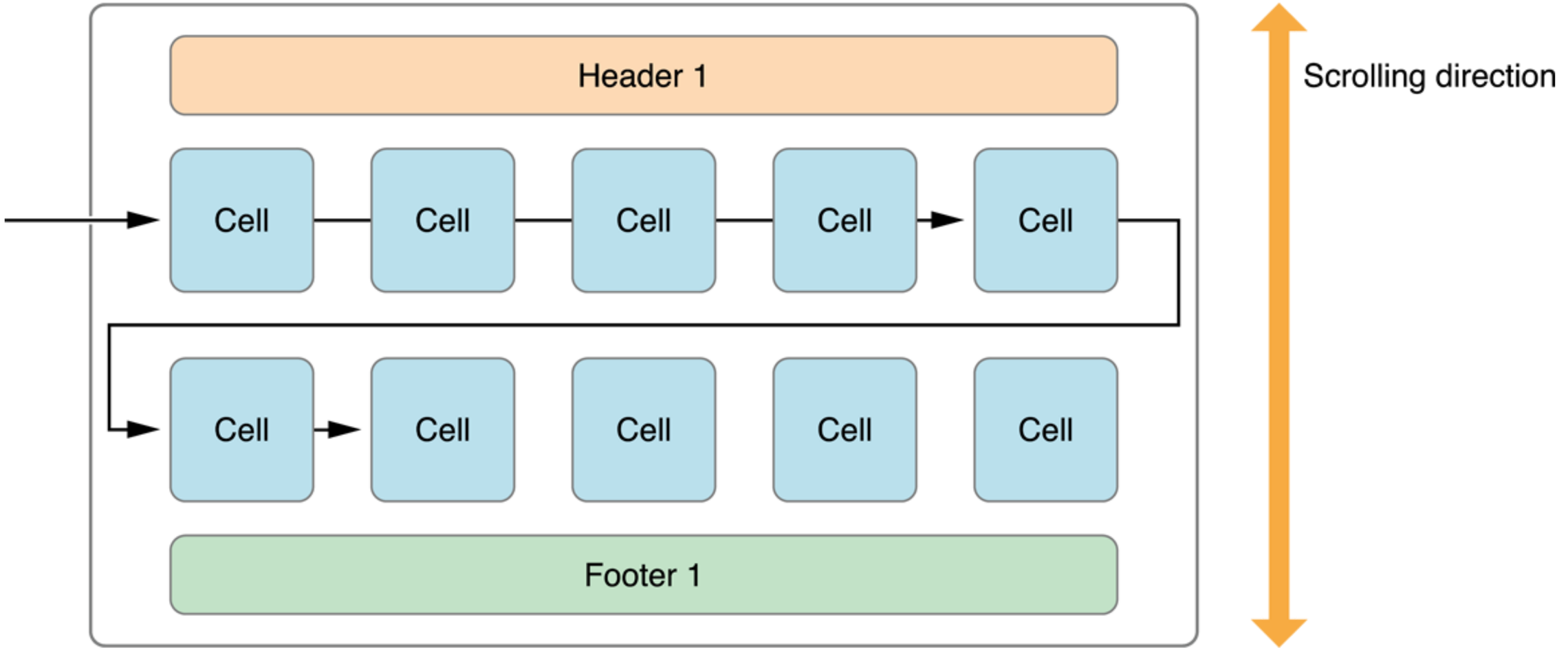

UICollectionViewDelegate
UICollectionViewDataSource
UICollectionView
UICollectionViewLayout
UICollectionViewFlowLayout
UICollectionViewCell

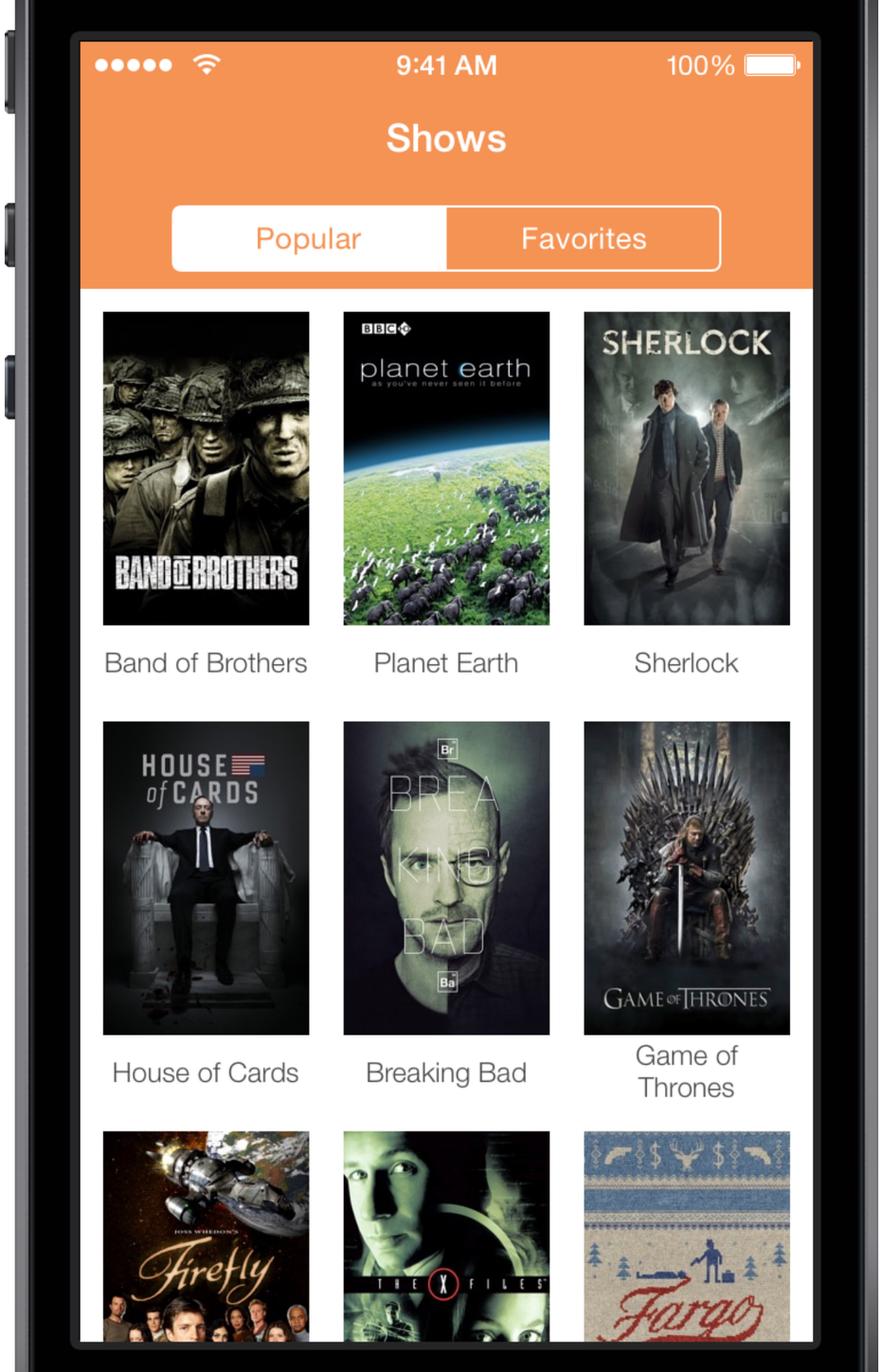
`UICollectionViewLayout`

- Classe abstrata
- Responsável por informar a posição e estado visual de elementos
  - Células
  - Supplementary views
  - Decoration views

`UICollectionViewFlowLayout`

- Implementação concreta de `UICollectionViewLayout`
- Bem mais simples
- Quando possível, herdar dele (ou até mesmo usar diretamente)
- Layout em grid


Header 1
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Footer 1
Scrolling direction


9:41 AM
100%
Shows
Popular
Favorites
BAND OF BROTHERS
Band of Brothers
BBC
planet earth
Planet Earth
SHERLOCK
Sherlock
HOUSE of CARDS
House of Cards
BREAKING BAD
Breaking Bad
GAME OF THRONES
Game of Thrones
Firefly
THE X FILES
Fargo

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

    let flowLayout = collectionViewLayout as! UICollectionViewFlowLayout

    let border = flowLayout.sectionInset.left + flowLayout.sectionInset.right
    let itemSize = flowLayout.itemSize.width + flowLayout.minimumInteritemSpacing
    let maxPerRow = floor((collectionView.bounds.width - border) / itemSize)
    let usedSpace = border + itemSize * maxPerRow

    let space = floor((collectionView.bounds.width - usedSpace) / 2)
    return UIEdgeInsets(top: flowLayout.sectionInset.top, left: space,
        bottom: flowLayout.sectionInset.bottom, right: space)
}
```