Introducing Collection Views

Session 205

Olivier Gutknecht
iOS Application and Framework Engineer

Luke Hiesterman
iOS Application and Framework Engineer

Agenda

What we will cover

Agenda

What we will cover

Collection View

- View architecture
- Data model and interaction

Agenda

What we will cover

Collection View

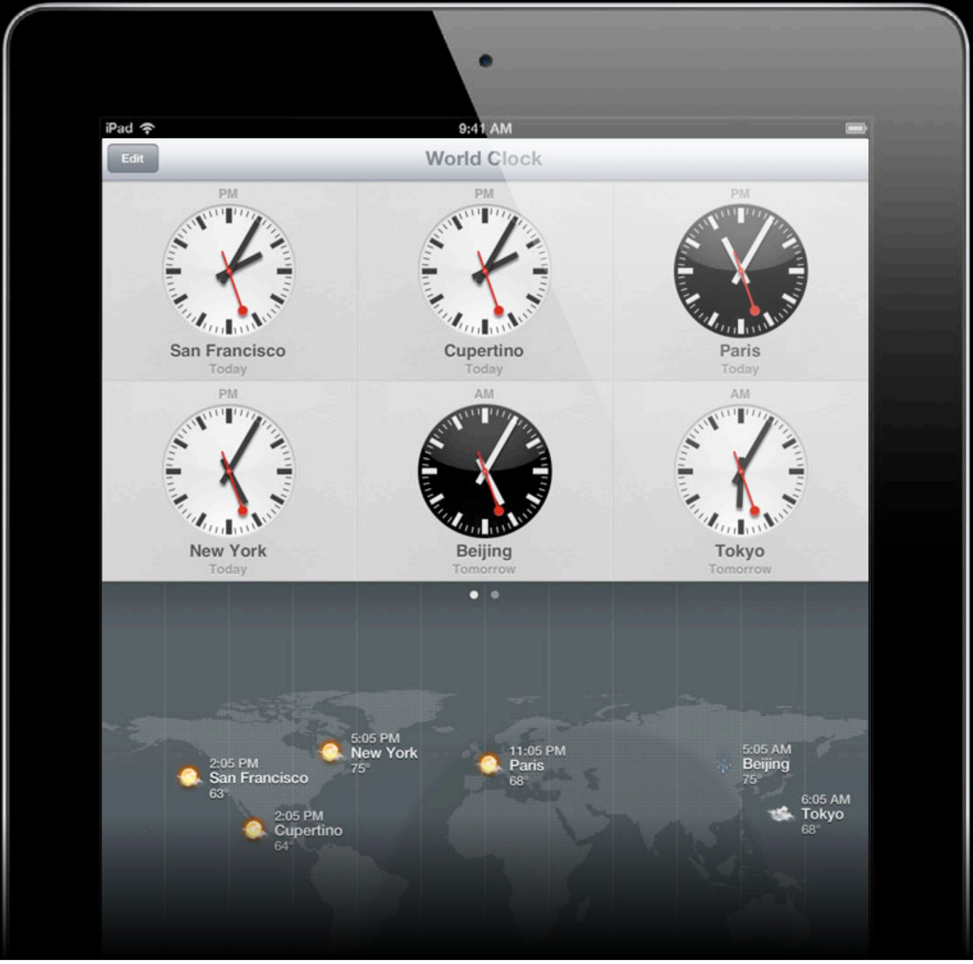
- View architecture
- Data model and interaction

Flow Layout

- Core concepts
- Customizing flow layout

[redacted]

Grids


iPad
9:41 AM
Edit
World Clock
PM
San Francisco
Today
PM
Cupertino
Today
PM
Paris
Today
PM
New York
Today
AM
Beijing
Tomorrow
AM
Tokyo
Tomorrow
2:05 PM
San Francisco
63°
5:05 PM
New York
75°
2:05 PM
Cupertino
64°
11:05 PM
Paris
68°
5:05 AM
Beijing
75°
6:05 AM
Tokyo
68°

Collection Views

A scenic tour

A Family Portrait

UICollectionView, UITableView, NSCollectionView

- Not a direct equivalent of NSCollectionView
- Not a replacement for UITableView—but a close sibling

Why Collection Views?

Why Collection Views?

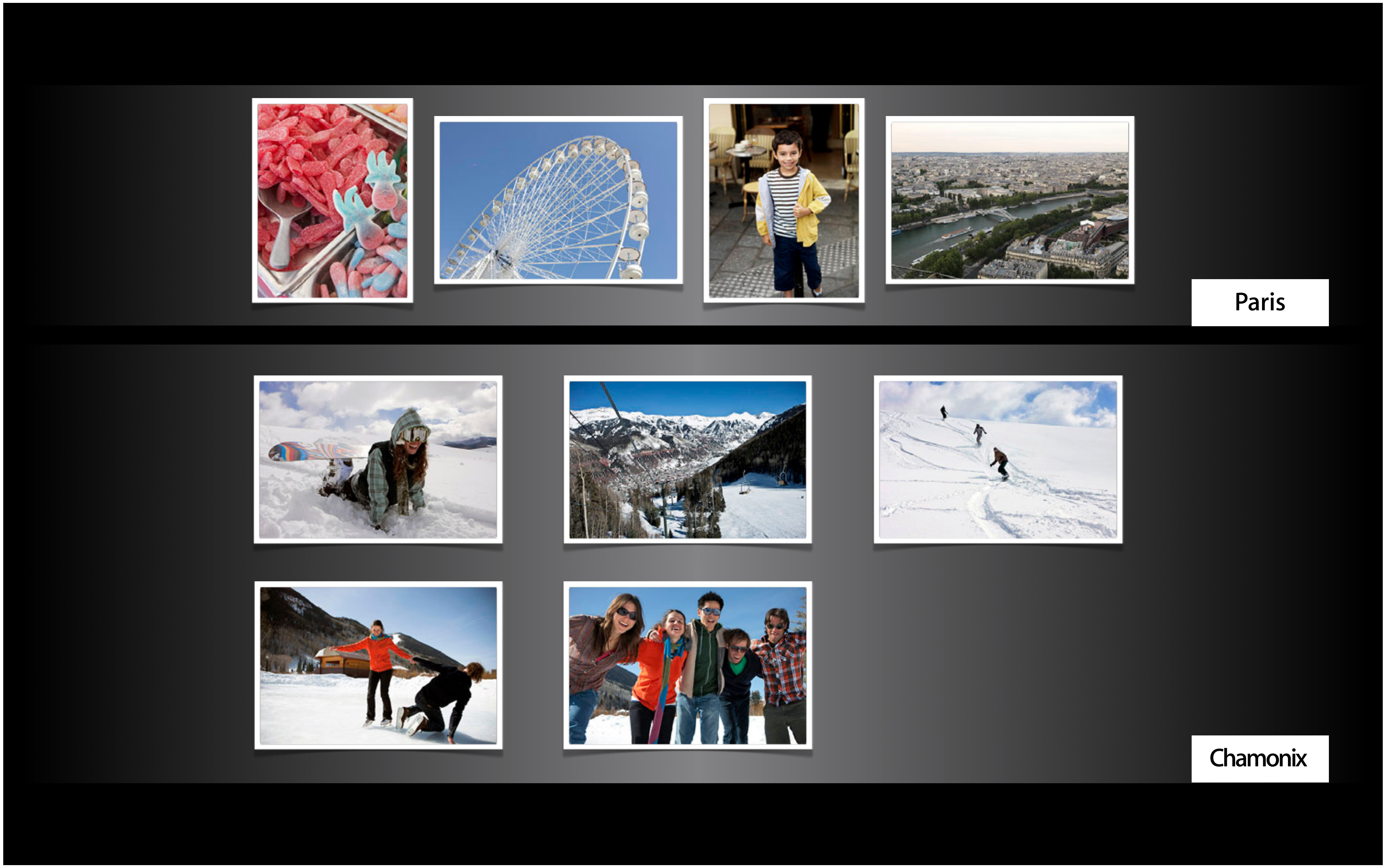
- A highly customizable representation of your content

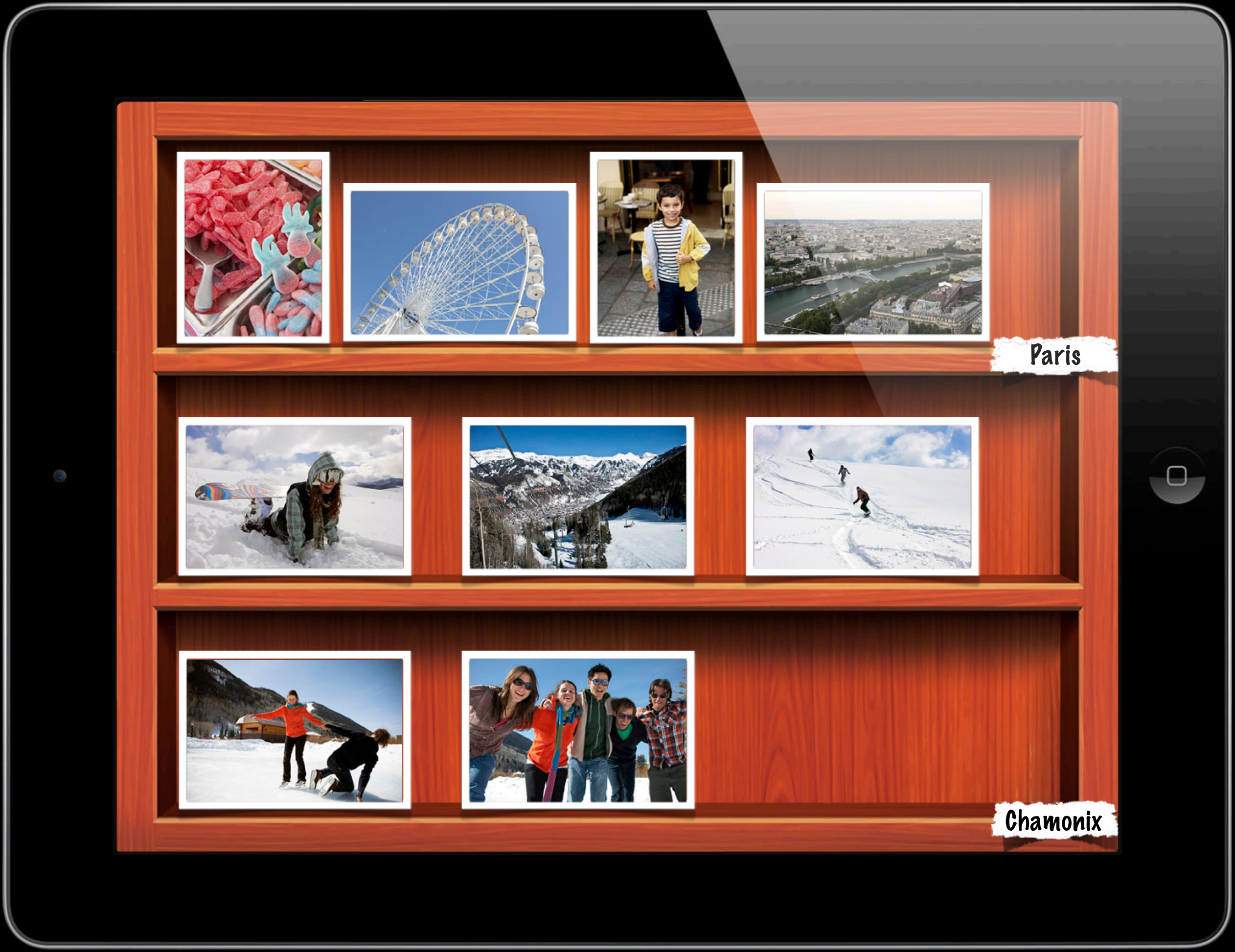
Why Collection Views?

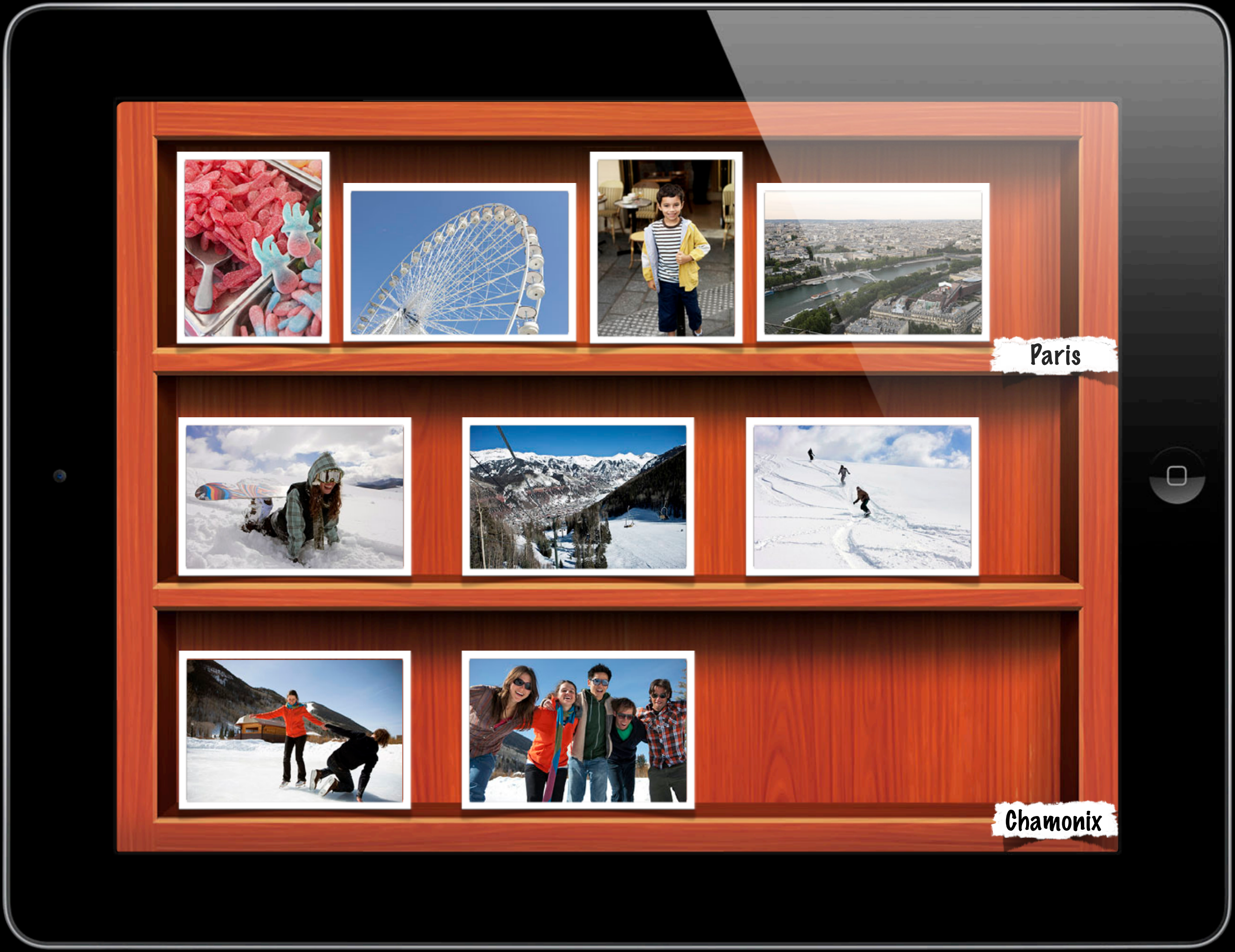
- A highly customizable representation of your content
- Keeps the usual best practices to manage data

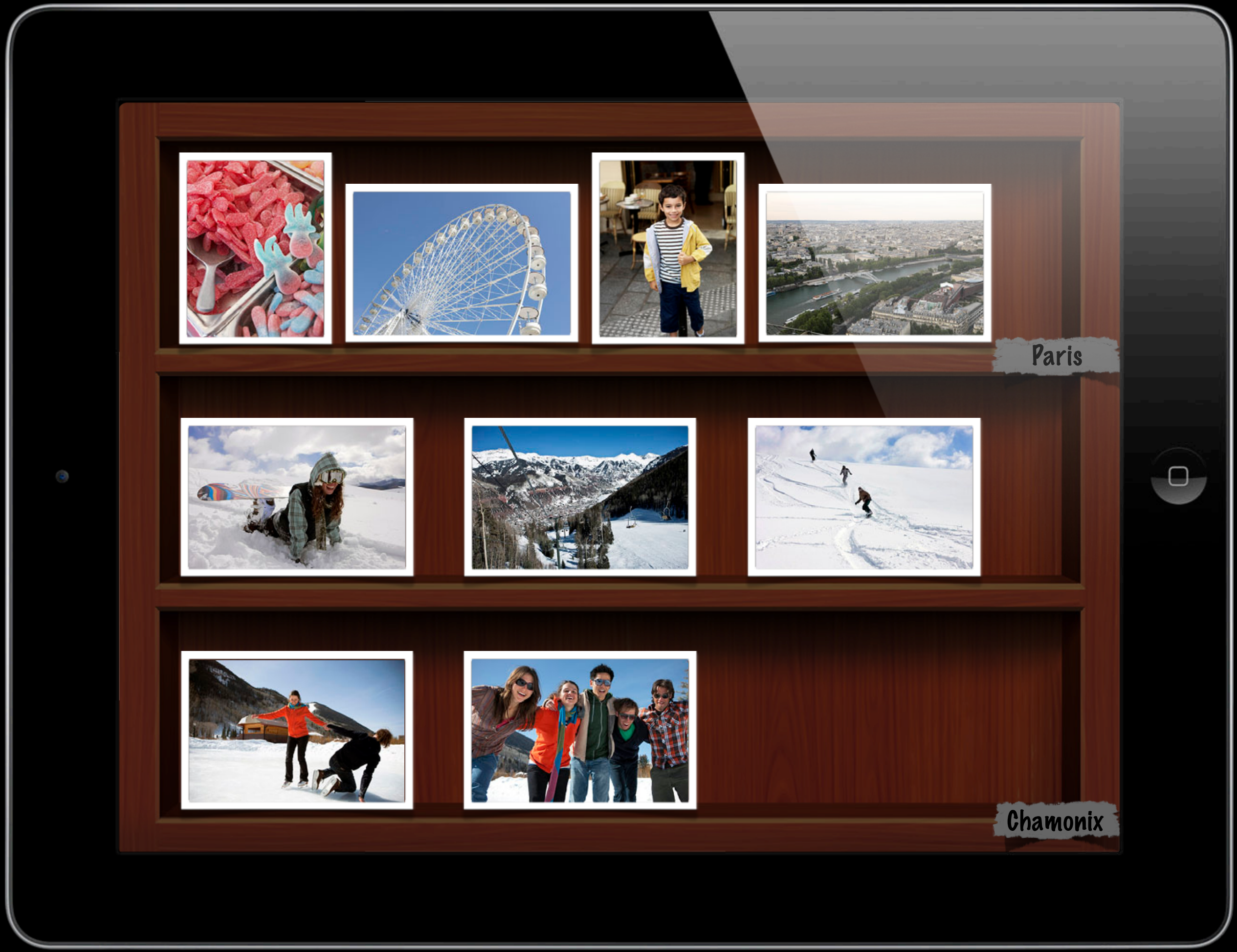
Why Collection Views?

- A highly customizable representation of your content
- Keeps the usual best practices to manage data
- High-performance even with large data sets

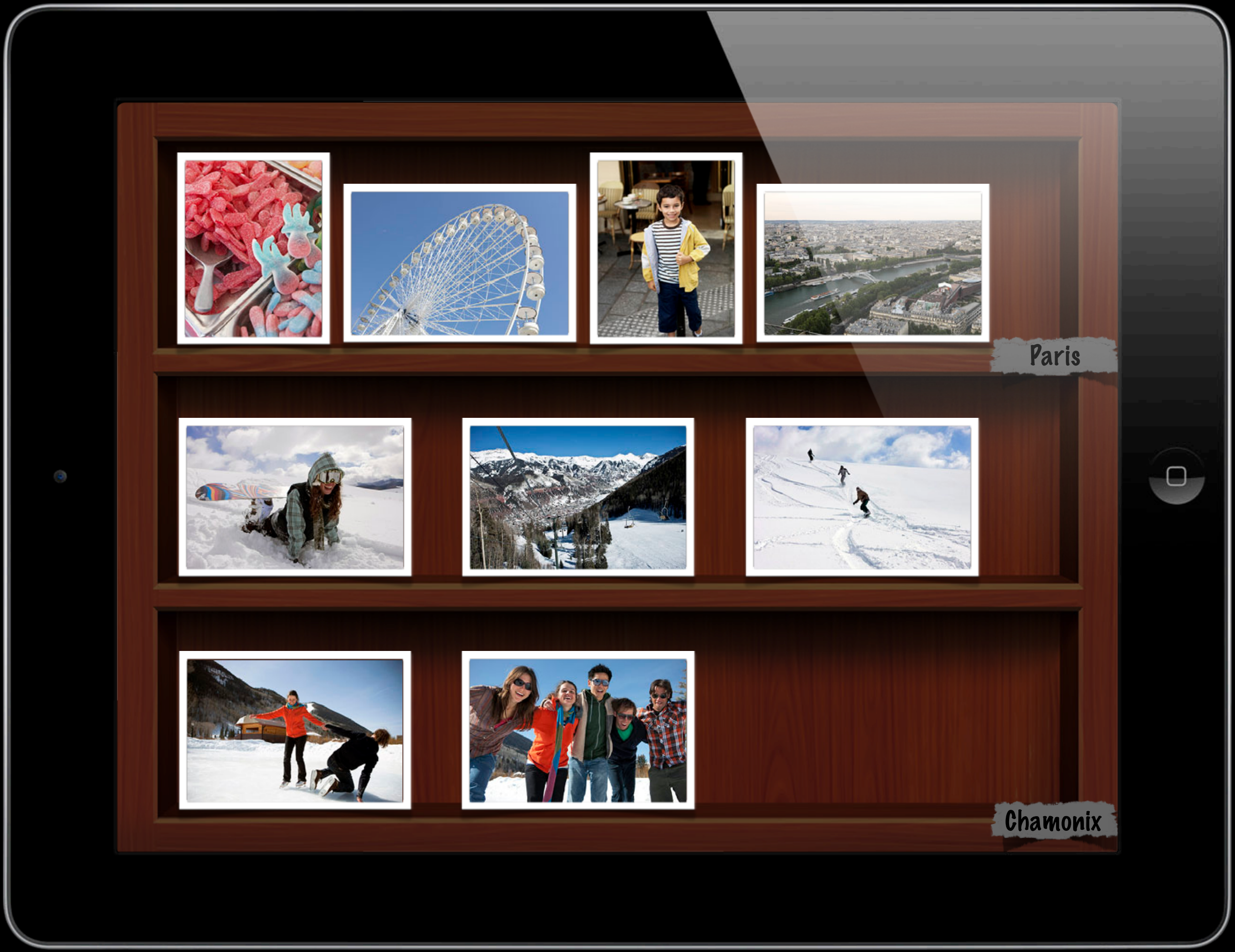

Paris
Chamonix


Paris
Chamonix

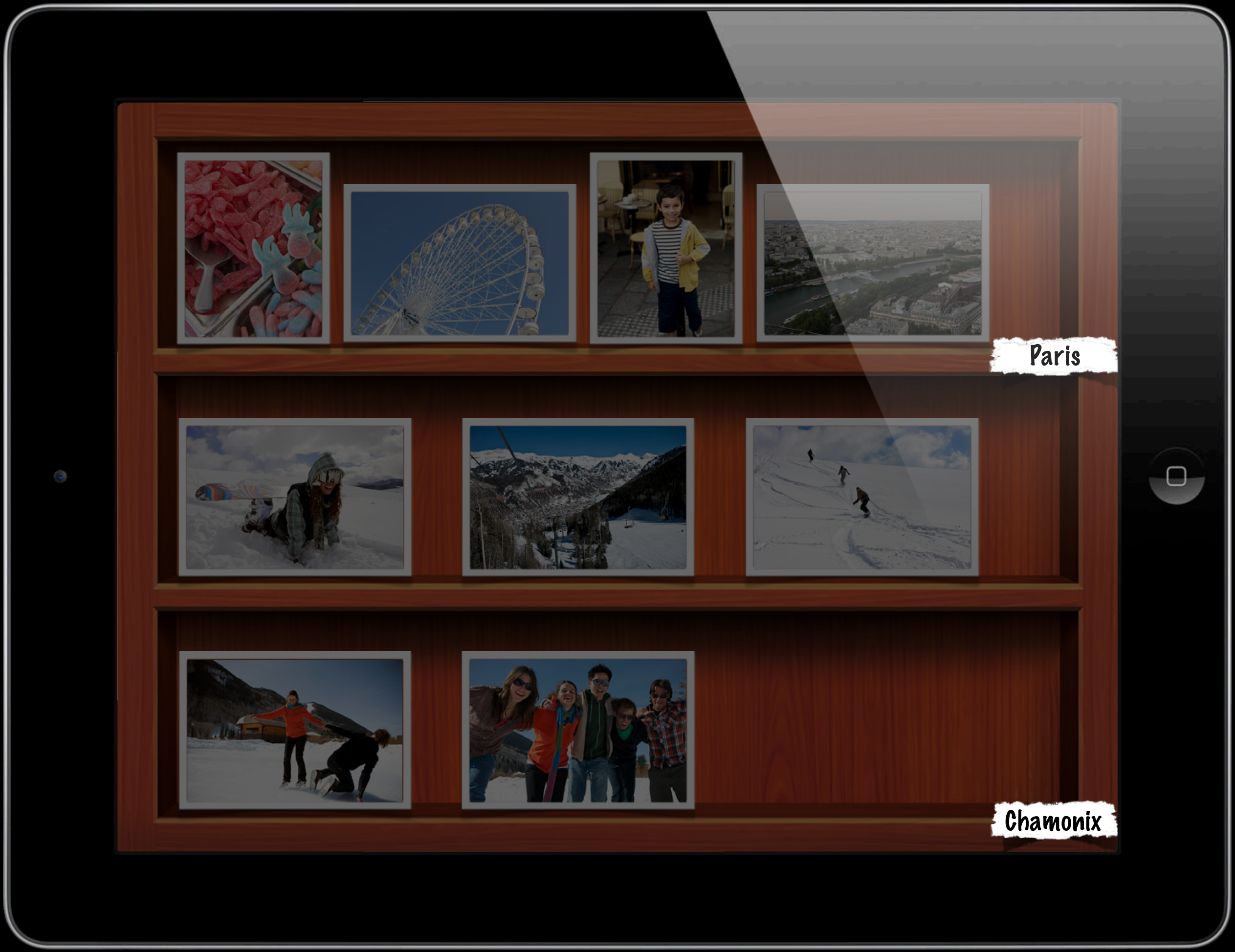

Paris
Chamonix


Paris
Chamonix

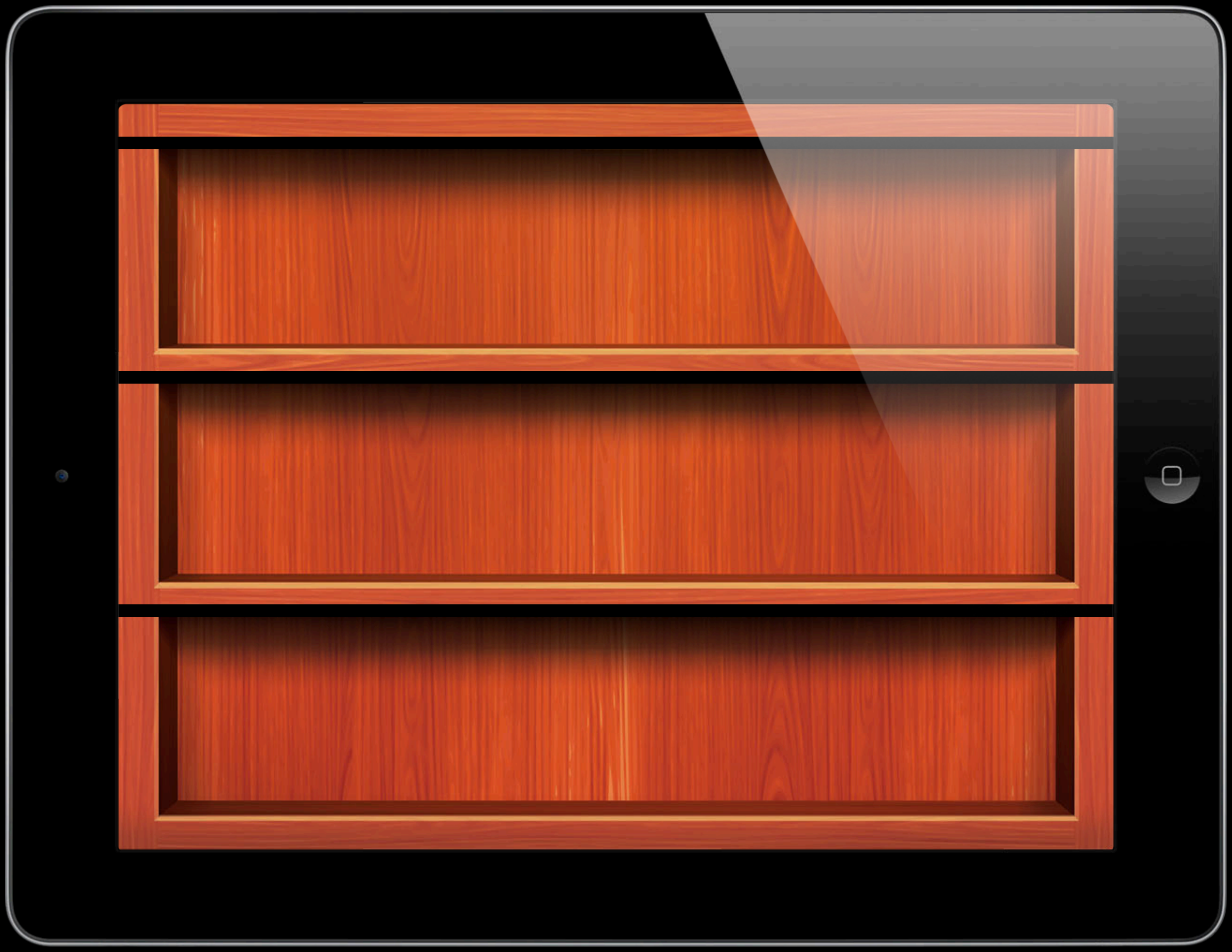
Cells



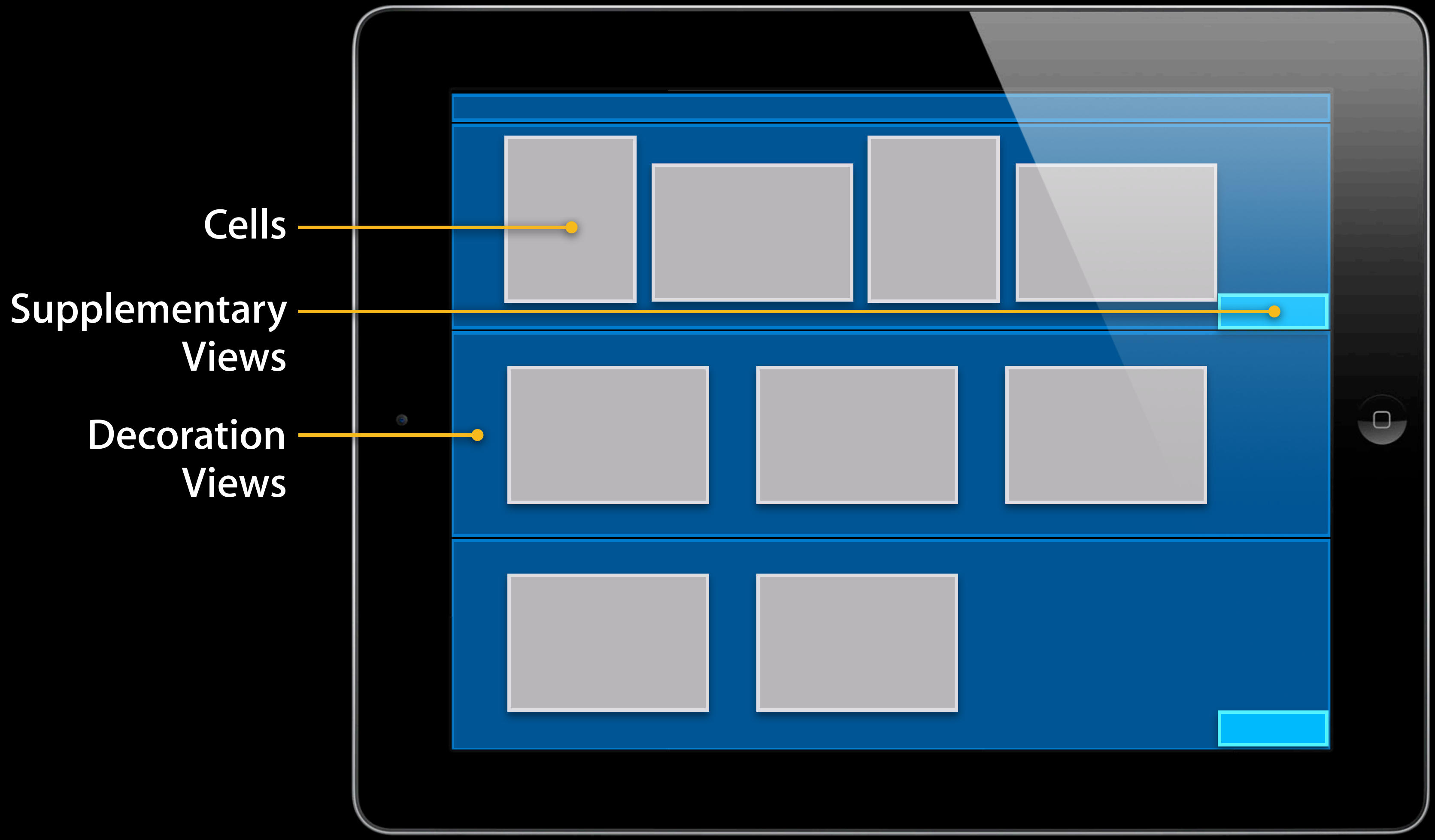
Supplementary Views



Decoration Views



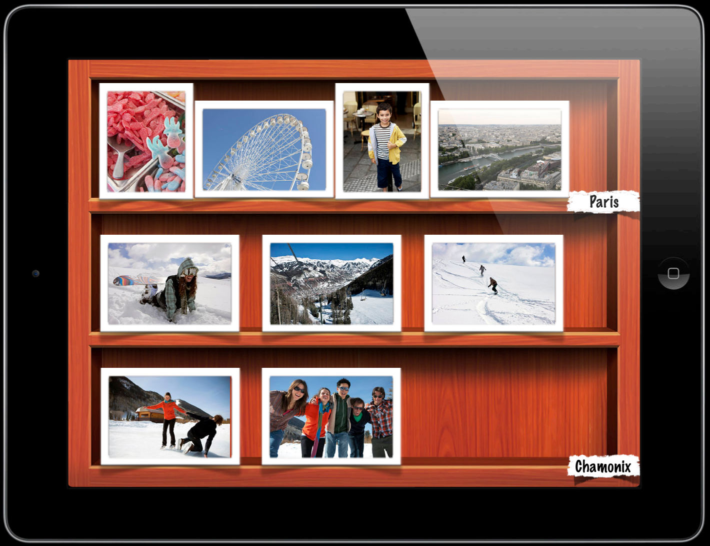
Paris
Chamonix


Cells
Supplementary Views
Decoration Views

Providing Content

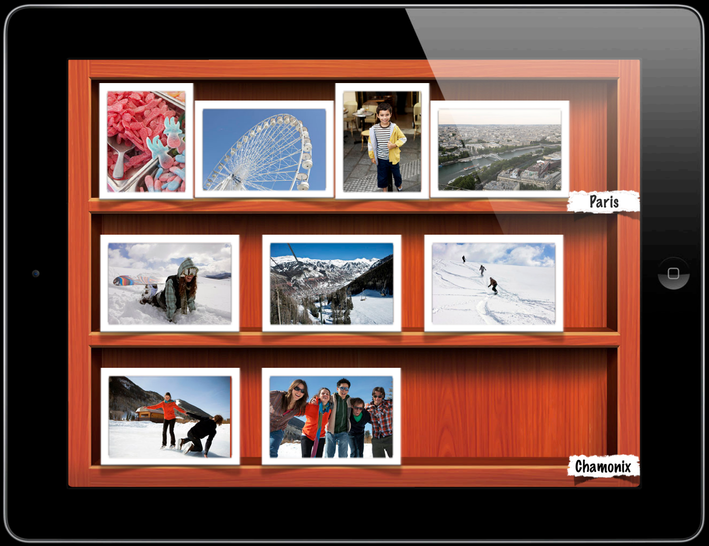
UICollectionViewDataSource

UICollectionViewDataSource



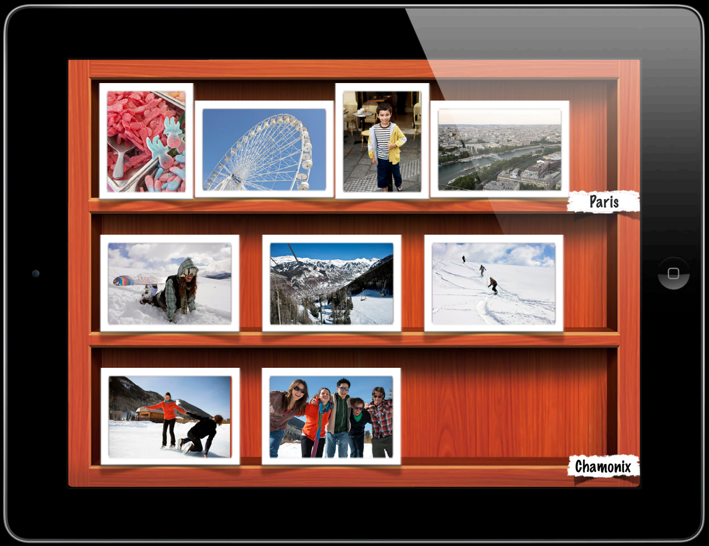
UICollectionViewDataSource

- Number of sections



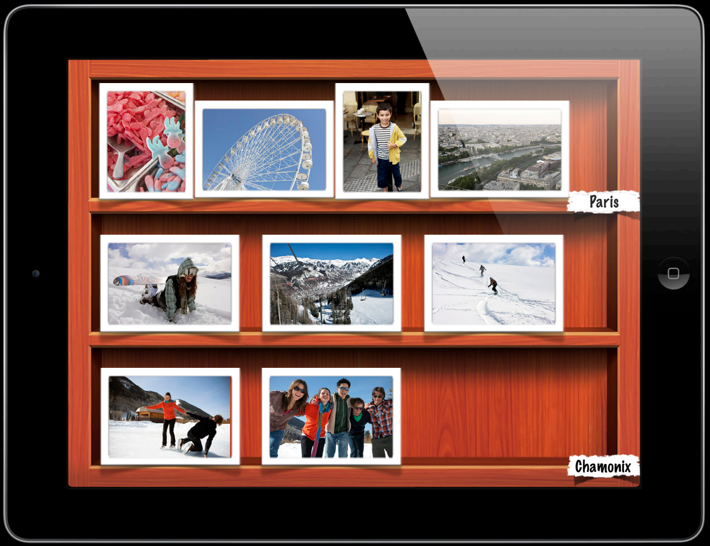
UICollectionViewDataSource

- Number of sections
- Number of items in sections

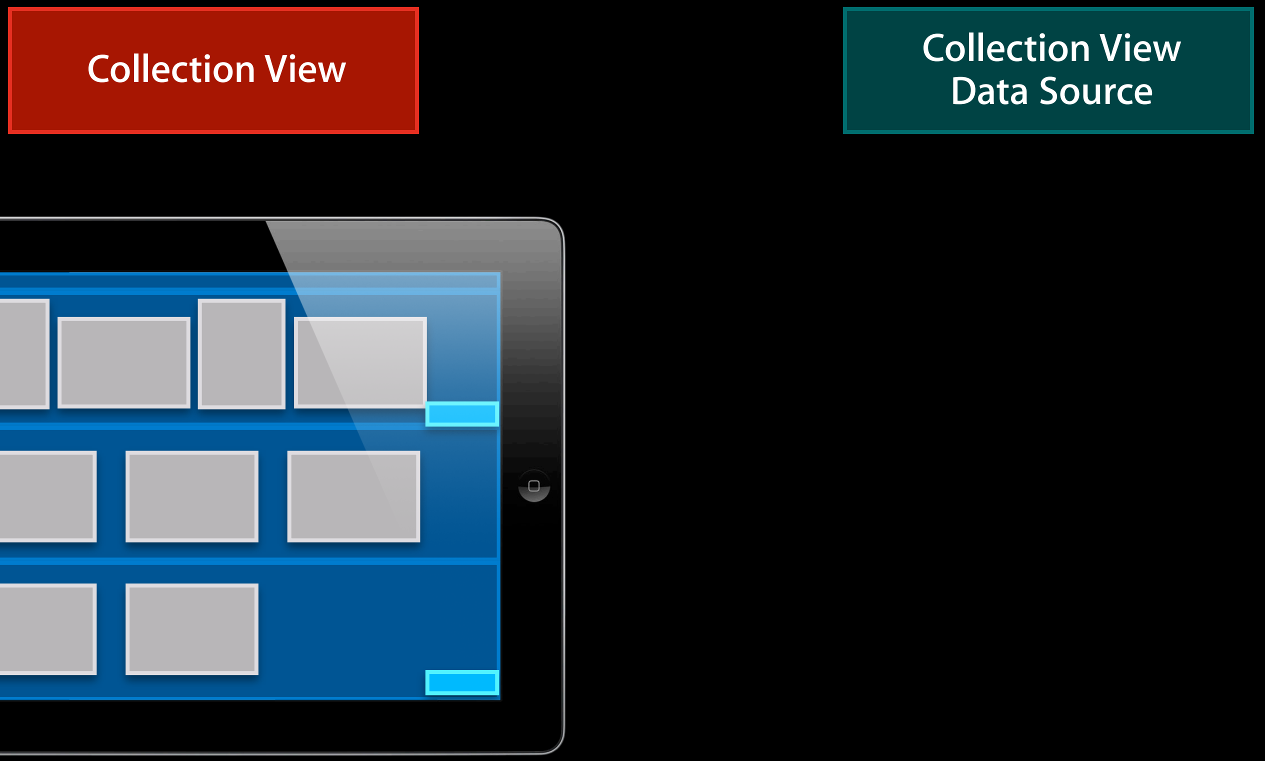


UICollectionViewDataSource

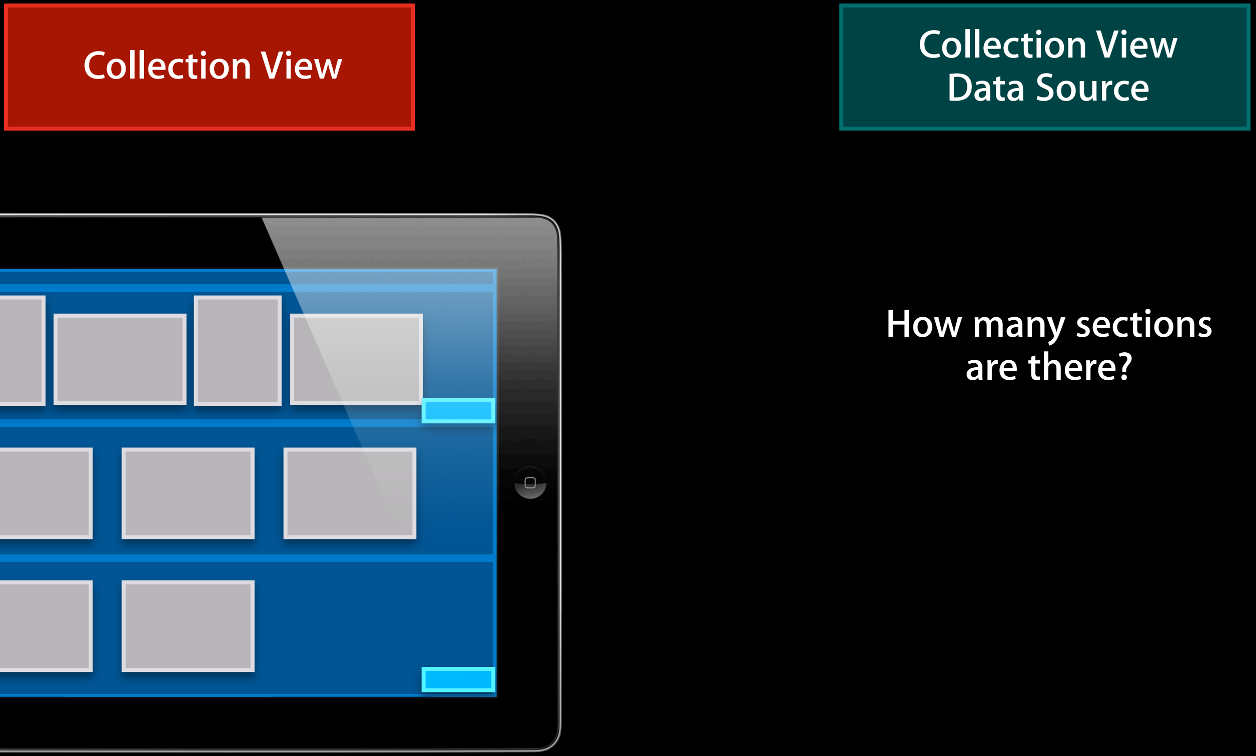
- Number of sections
- Number of items in sections
- Cell and supplementary view setup



`numberOfSectionsInCollectionView:`

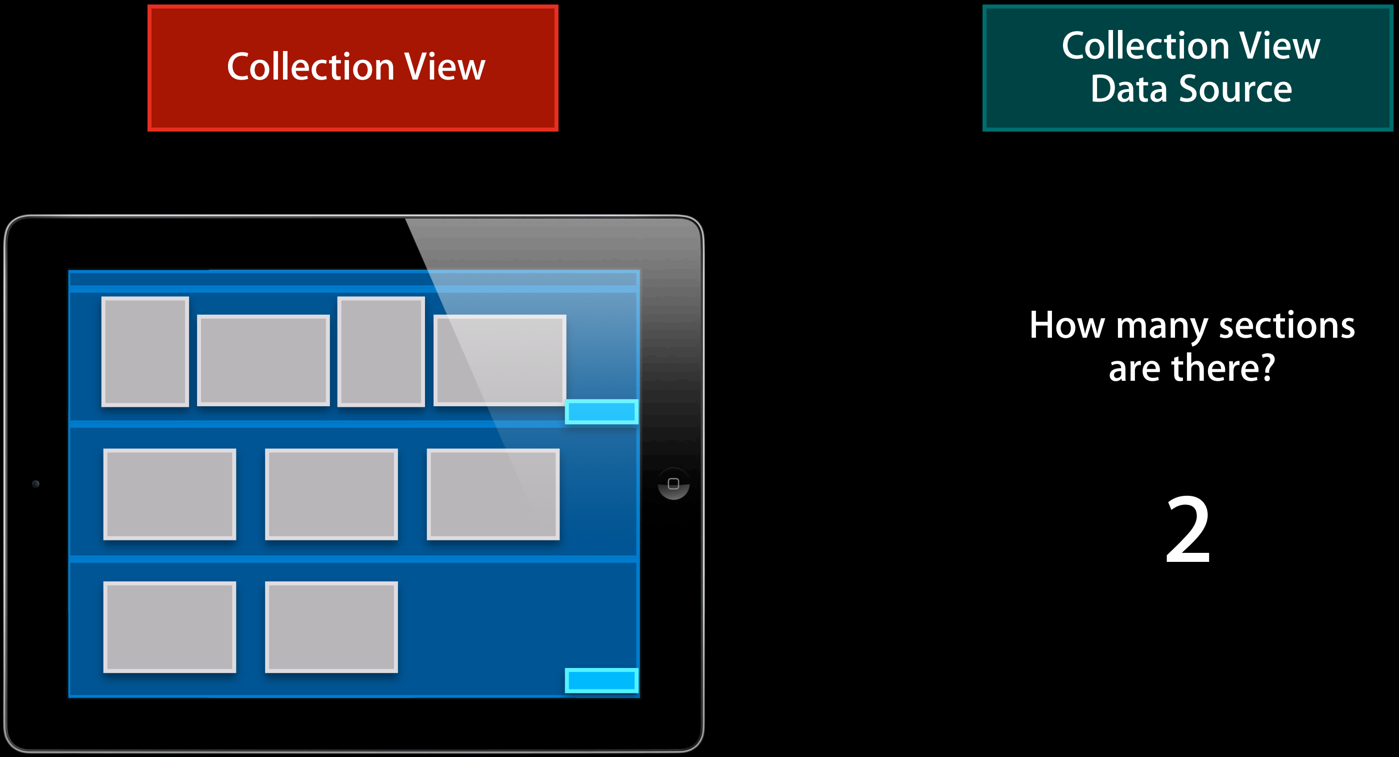


`numberOfSectionsInCollectionView:`



How many sections are there?

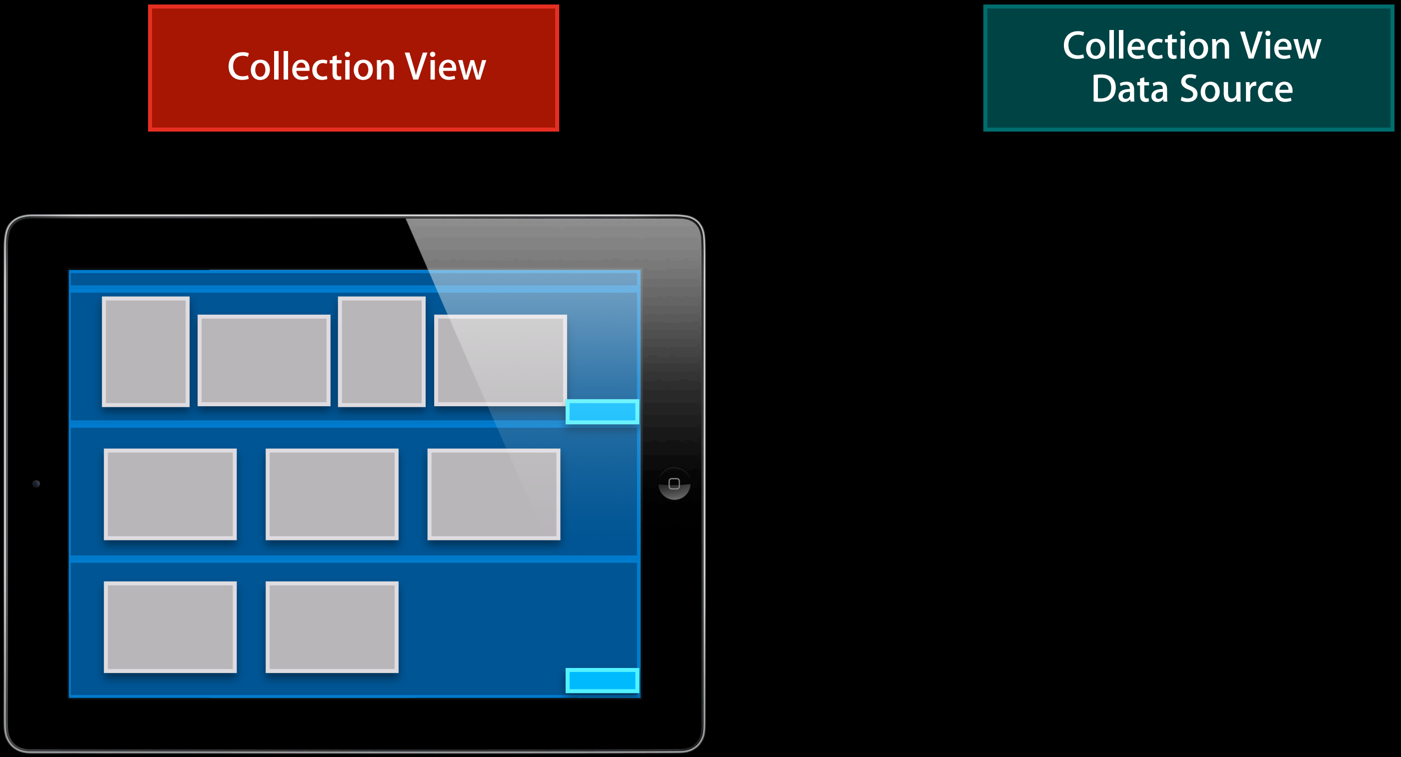
`numberOfSectionsInCollectionView:`



How many sections are there?

2

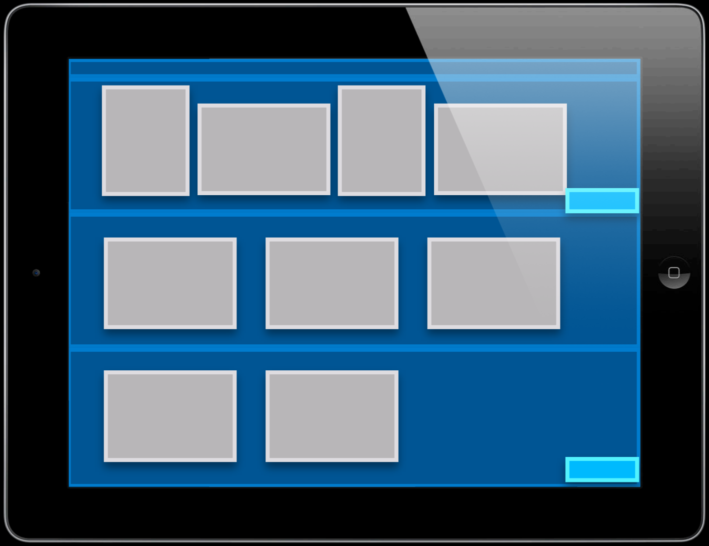
collectionView:numberOfItemsInSection:



`collectionView:numberOfItemsInSection:`

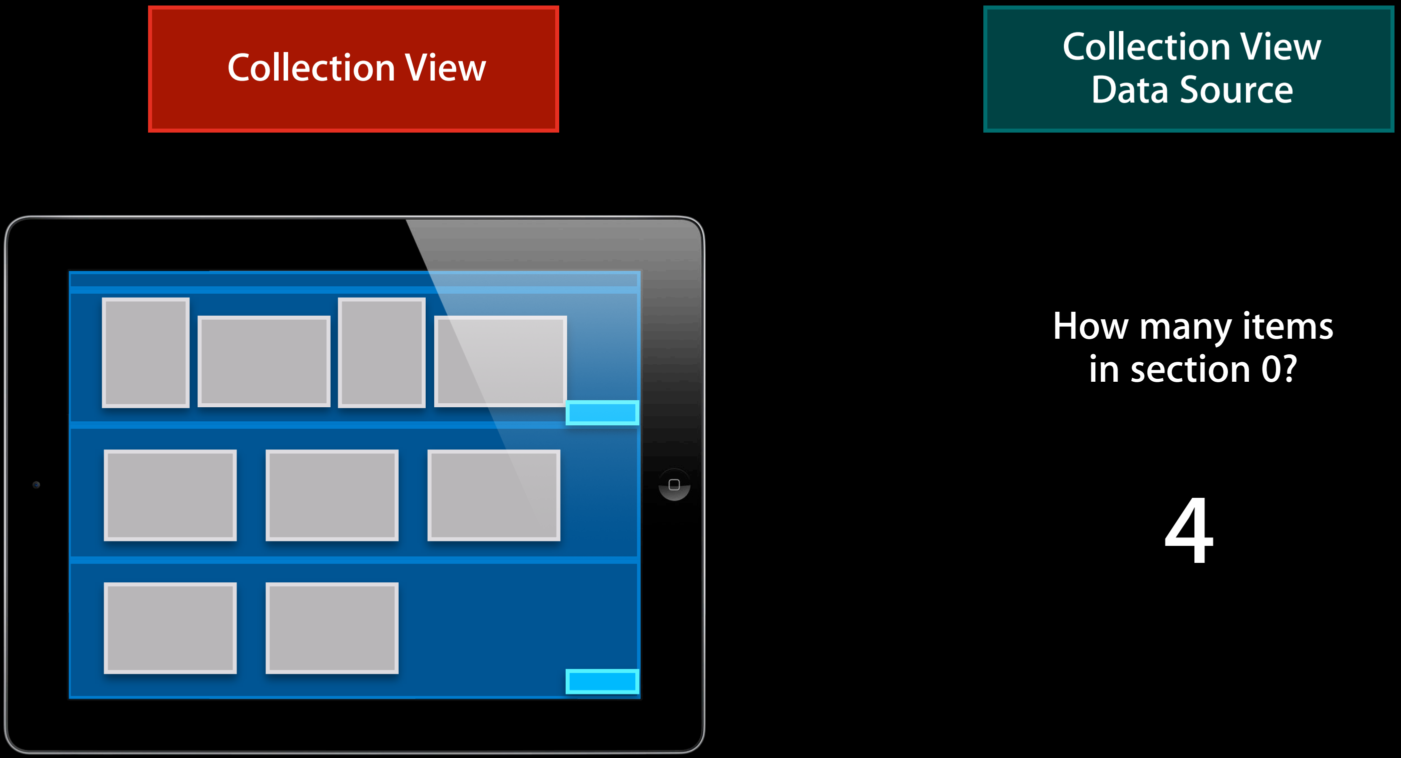
Collection View

Collection View
Data Source



How many items
in section 0?

`collectionView:numberOfItemsInSection:`



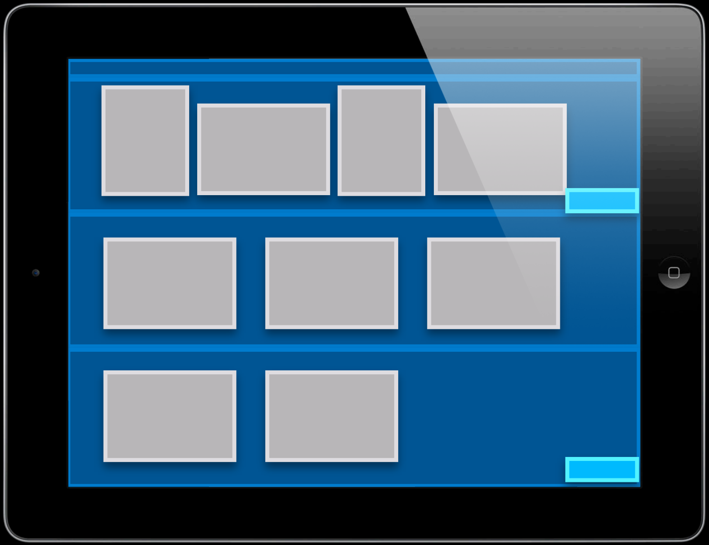
How many items
in section 0?

4

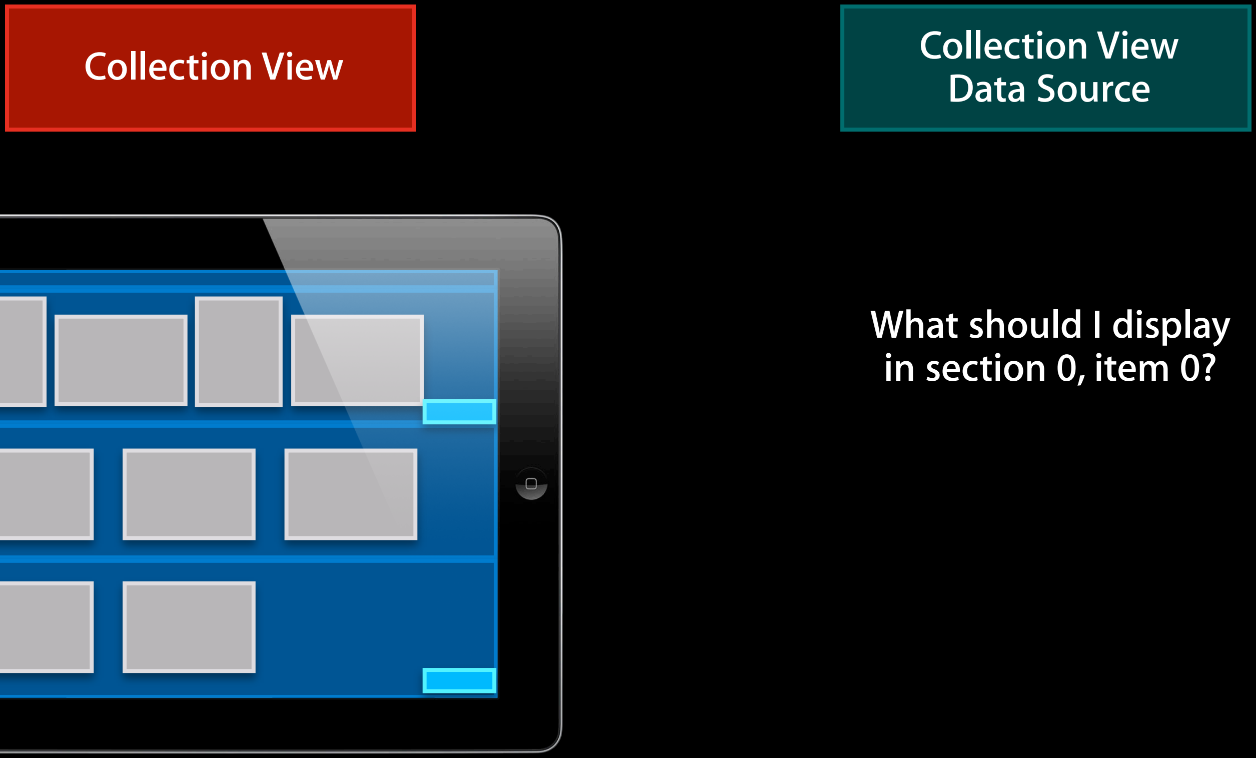
`collectionView:cellForItemAtIndexPath:`

Collection View

Collection View
Data Source



`collectionView:cellForItemAtIndexPath:`

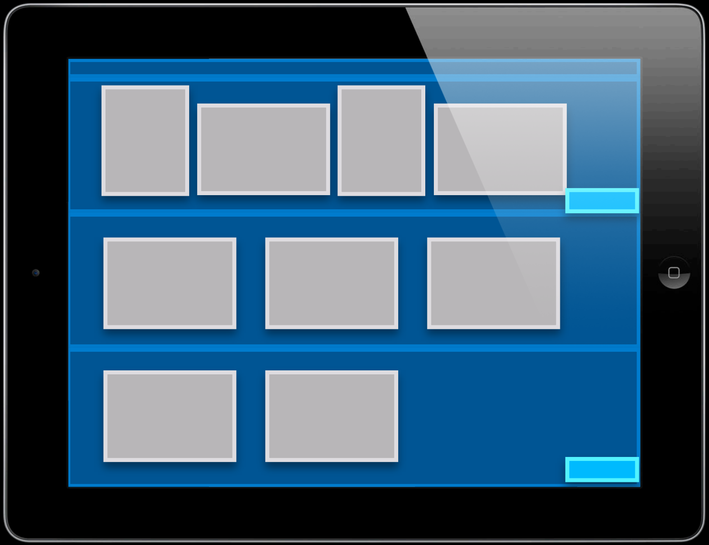


What should I display in section 0, item 0?

`collectionView:cellForItemAtIndexPath:`

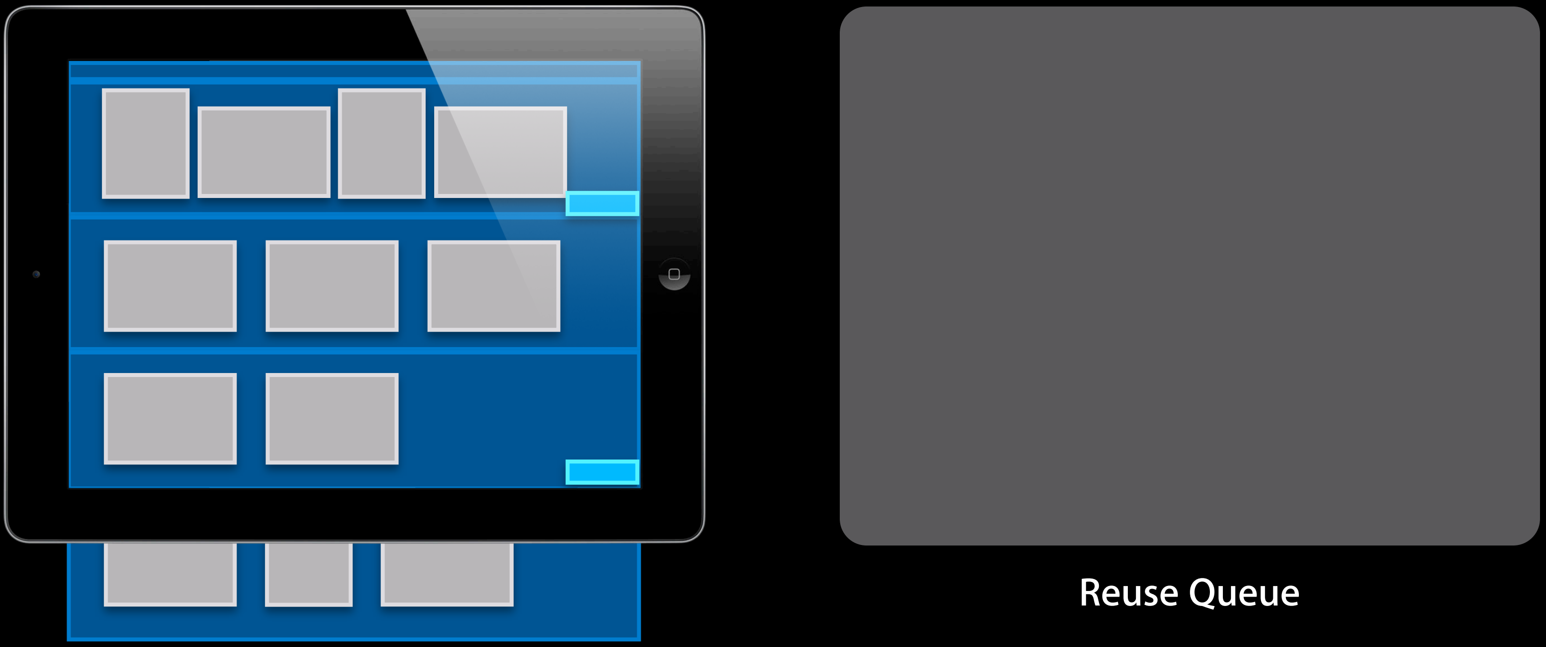
Collection View

Collection View
Data Source

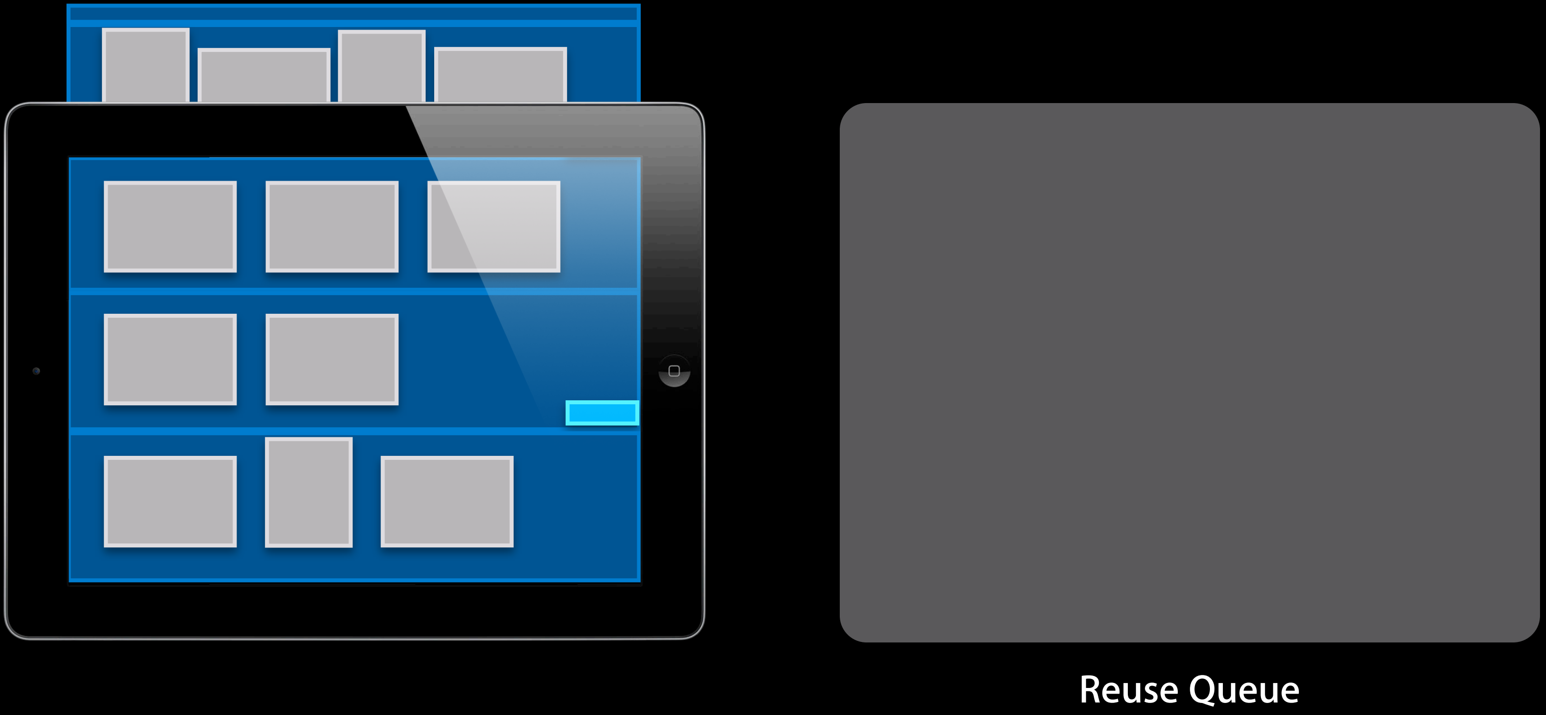


What should I display
in section 0, item 0?

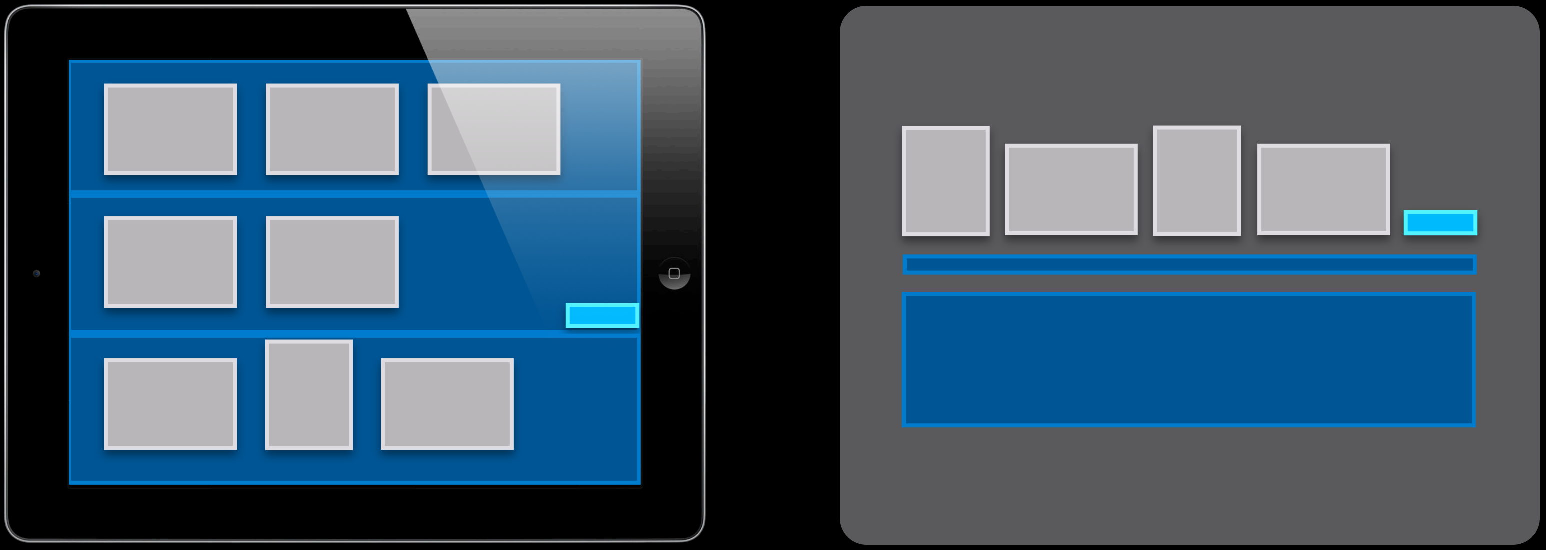
Cell and View Reuse



Cell and View Reuse

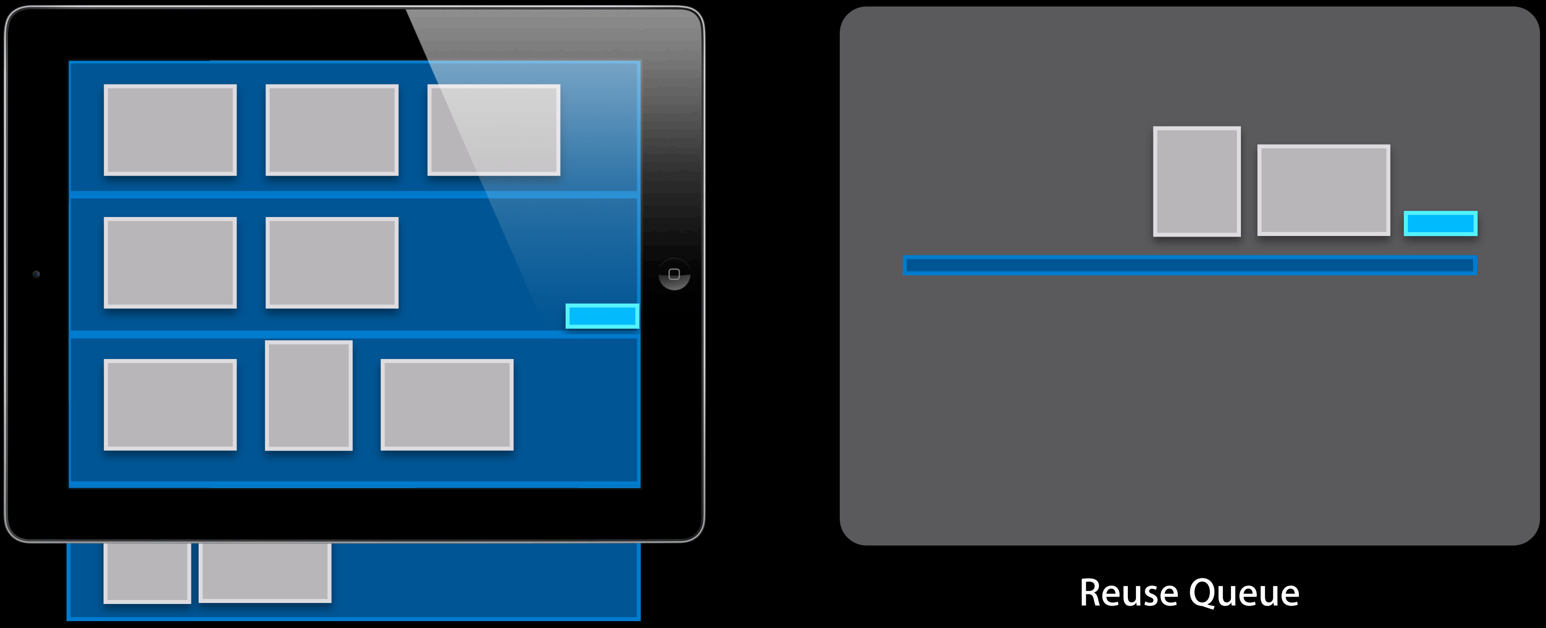


Cell and View Reuse

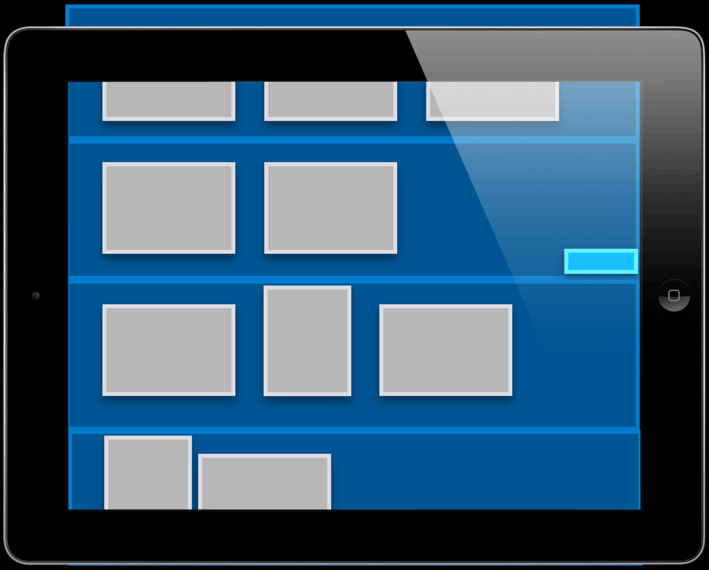


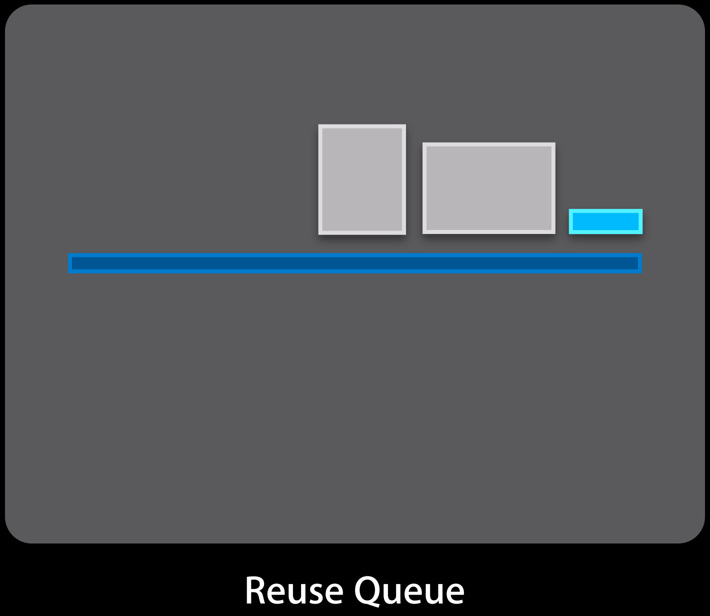
Reuse Queue

Cell and View Reuse



Cell and View Reuse





Cell Reuse, Improved!

Cell Reuse, Improved!

```
// In collection view setup...
[collectionView registerClass:[MyCell class]
                forCellWithReuseIdentifier:@”MY_CELL_ID”]
```

Cell Reuse, Improved!

```
// In collection view setup...
[collectionView registerClass:[MyCell class]
                forCellWithReuseIdentifier:@"MY_CELL_ID"]


- (UICollectionView*)collectionView:(UICollectionView*)cv
          cellForItemAtIndexPath:(NSIndexPath*)indexPath
 {
   MyCell *cell = [cv dequeueReusableCellWithReuseIdentifier:@"MY_CELL_ID"];
   if (!cell) {
...
   }
   // Configure the cell's content
   cell.imageView.image = ...
   return cell;
 }
```

Cell Reuse, Improved!

```
// In collection view setup...
[collectionView registerClass:[MyCell class]
                forCellWithReuseIdentifier:@”MY_CELL_ID”]


- (UICollectionView*)collectionView:(UICollectionView*)cv
         cellForItemAtIndexPath:(NSIndexPath*)indexPath
 {
   MyCell *cell = [cv dequeueReusableCellWithReuseIdentifier:@”MY_CELL_ID”];
   if (!cell) {
     // Well, nothing really. Never again
   }
   // Configure the cell's content
   cell.imageView.image = ...
   return cell;
 }
```

Cell Reuse, Improved!

```
// In collection view setup...
[collectionView registerClass:[MyCell class]
                forCellWithReuseIdentifier:@”MY_CELL_ID”]


- (UICollectionView*)collectionView:(UICollectionView*)cv
         cellForItemAtIndexPath:(NSIndexPath*)indexPath
 {
   MyCell *cell = [cv dequeueReusableCellWithReuseIdentifier:@”MY_CELL_ID”];
   // Configure the cell's content
   cell.imageView.image = ...
   return cell;
 }
```

Cell Reuse, Improved!

Cell Reuse, Improved!

- We always instantiate the cell for you!

Cell Reuse, Improved!

- We always instantiate the cell for you!
- You just have to register

```
– (void)registerClass:forCellWithReuseIdentifier:
– (void)registerClass:forSupplementaryViewOfKind:withReuseIdentifier:
– (void)registerNib:forCellWithReuseIdentifier:
– (void)registerNib:forSupplementaryViewOfKind:withReuseIdentifier:
```

Cell Reuse, Improved!

- We always instantiate the cell for you!
- You just have to register

```
– (void)registerClass:forCellWithReuseIdentifier:
– (void)registerClass:forSupplementaryViewOfKind:withReuseIdentifier:
– (void)registerNib:forCellWithReuseIdentifier:
– (void)registerNib:forSupplementaryViewOfKind:withReuseIdentifier:
```

- …and dequeue a cell

```
– (id)dequeueReusableCellWithReuseIdentifier:forIndexPath:
– (id)dequeueReusableSupplementaryViewOfKind:withReuseIdentifier:forIndexPath:
```

Interacting with Content

UICollectionViewDelegate

UICollectionViewDelegate

UICollectionViewDelegate

- Control cell highlight

UICollectionViewDelegate

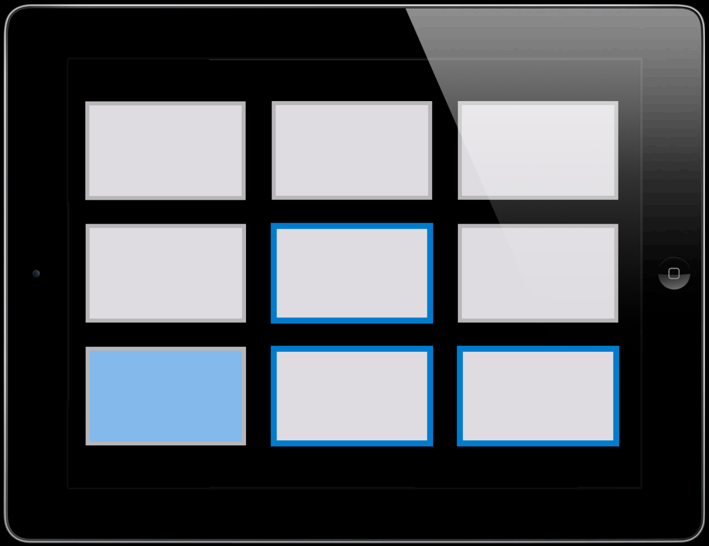
- Control cell highlight
- Control cell selection

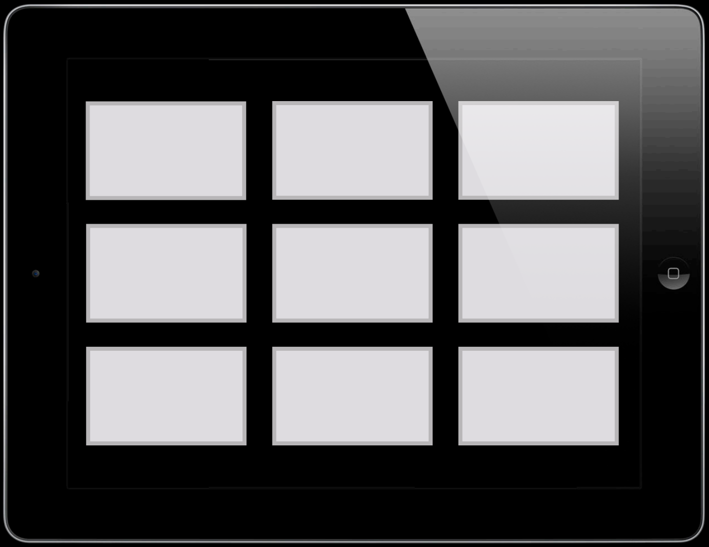
UICollectionViewDelegate

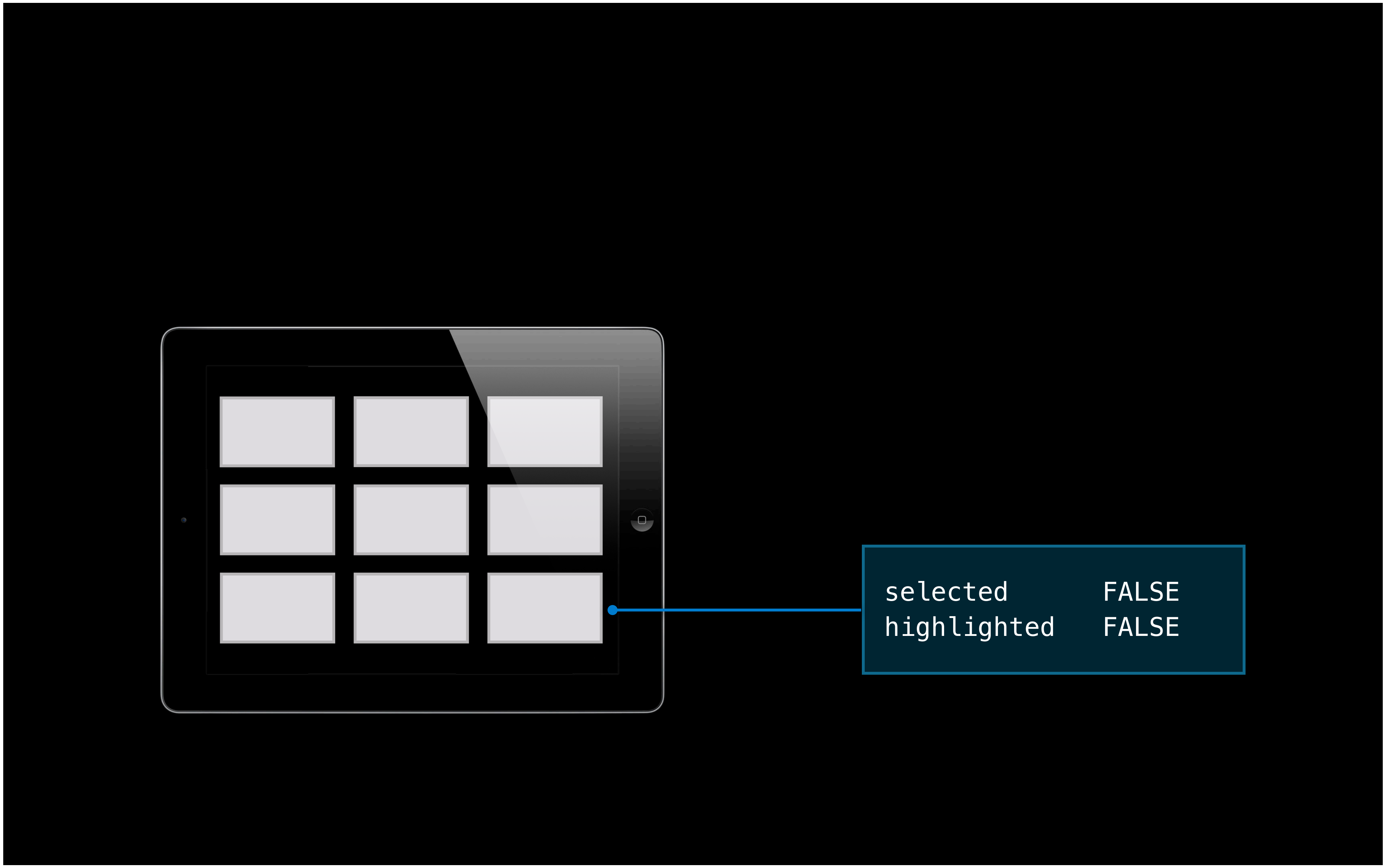
- Control cell highlight
- Control cell selection
- Support for menu actions on cells

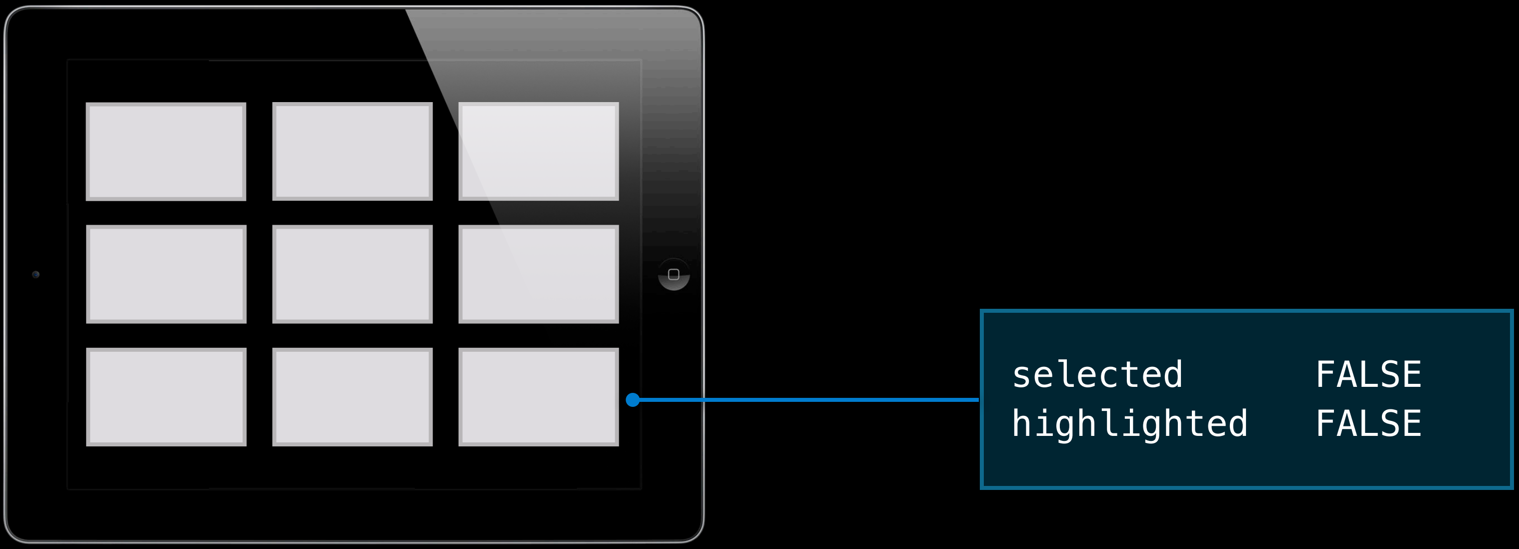
Selection and Highlight, Improved!

Fine control of highlighting and precise flow



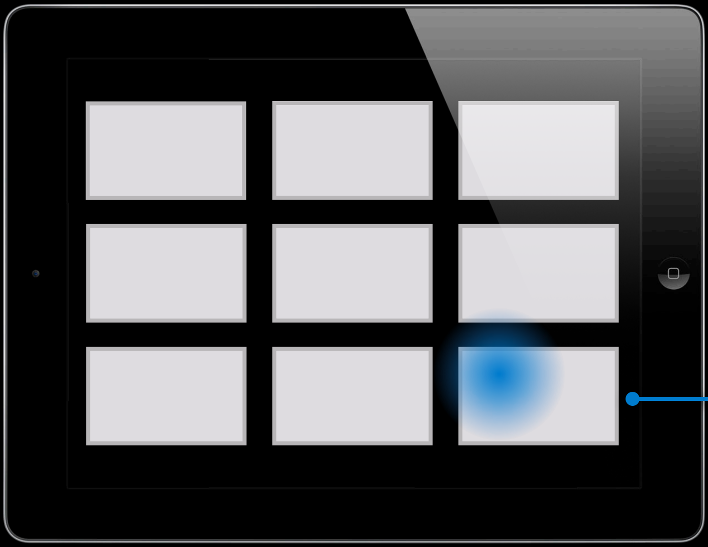



```
selected       FALSE
highlighted    FALSE
```



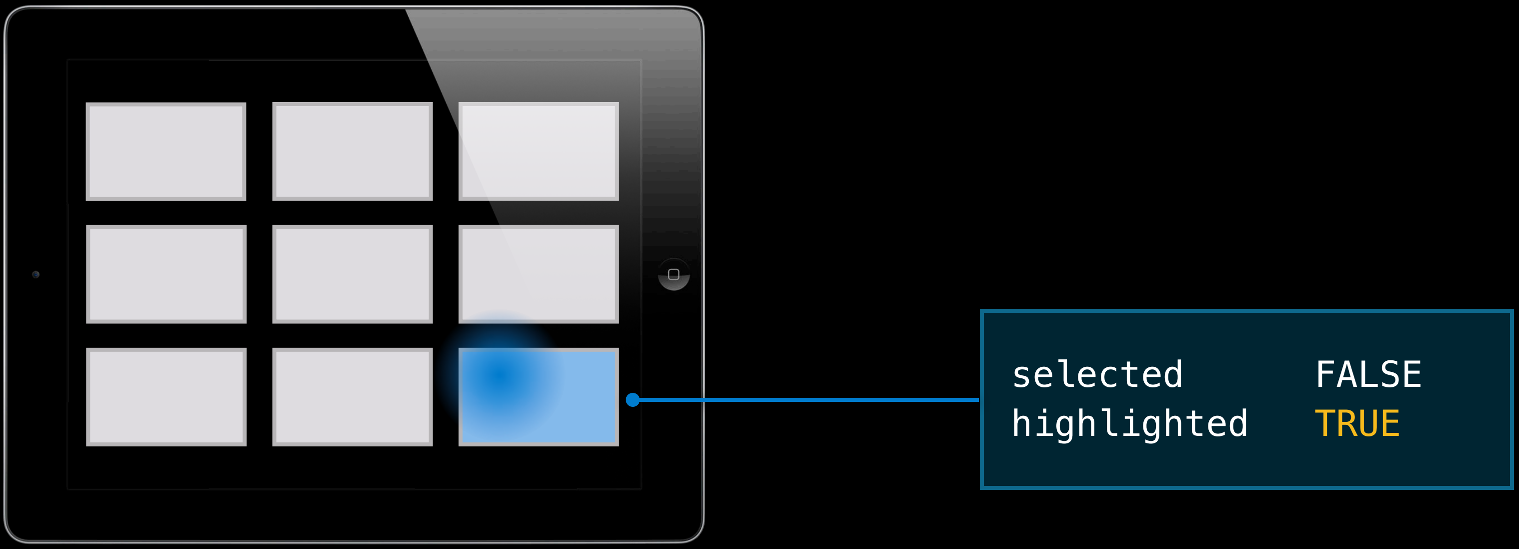
```
selected      FALSE
highlighted   FALSE
```

`collectionView:shouldHighlightItemAtIndexPath:`



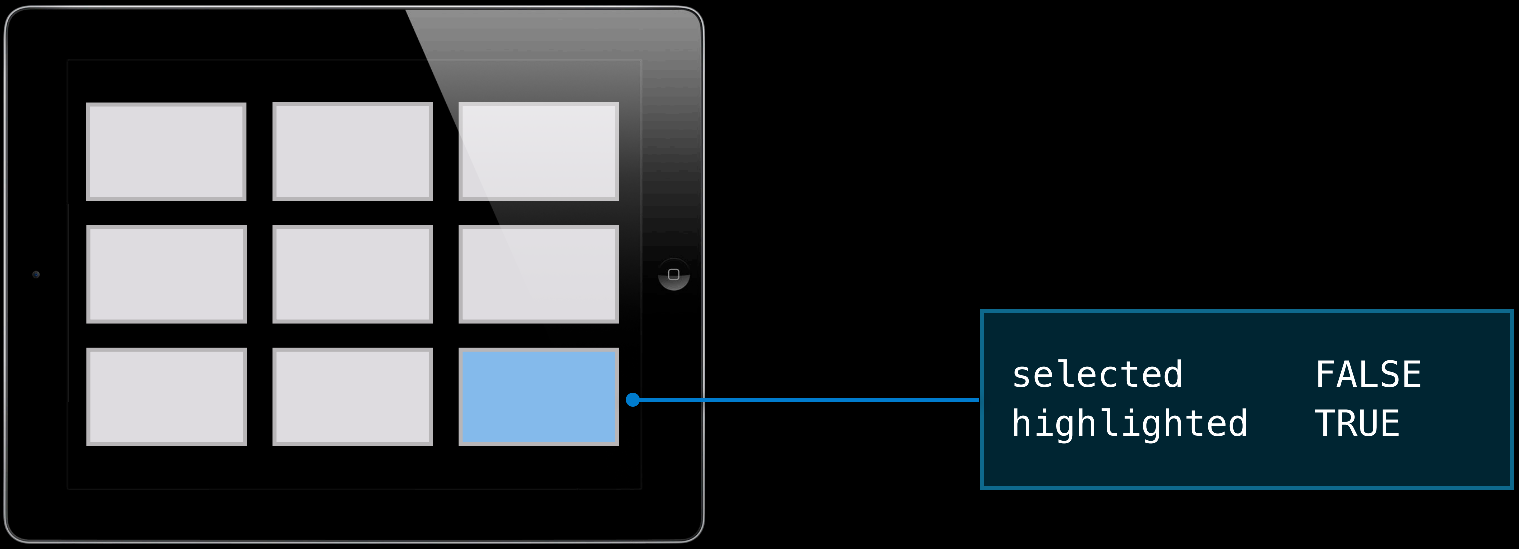
```
selected      FALSE
highlighted   FALSE
```

collectionView:didHighlightItemAtIndexPath:



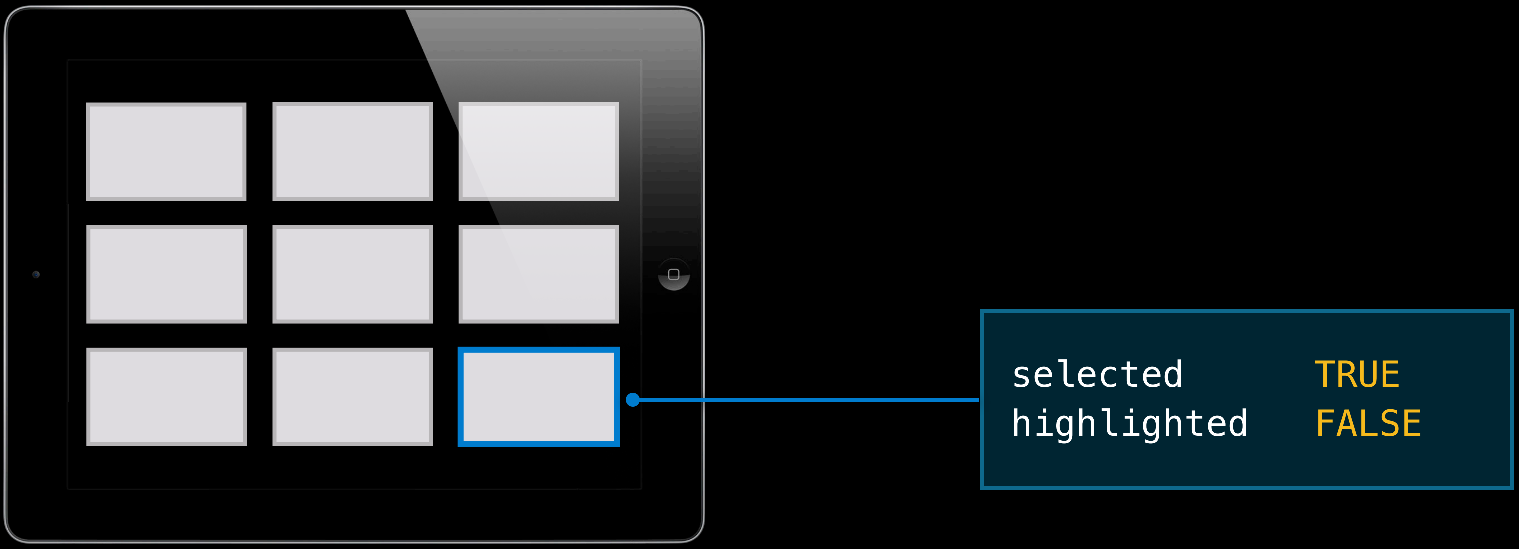
```
selected       FALSE
highlighted    TRUE
```

collectionView:shouldSelectItemAtIndexPath:

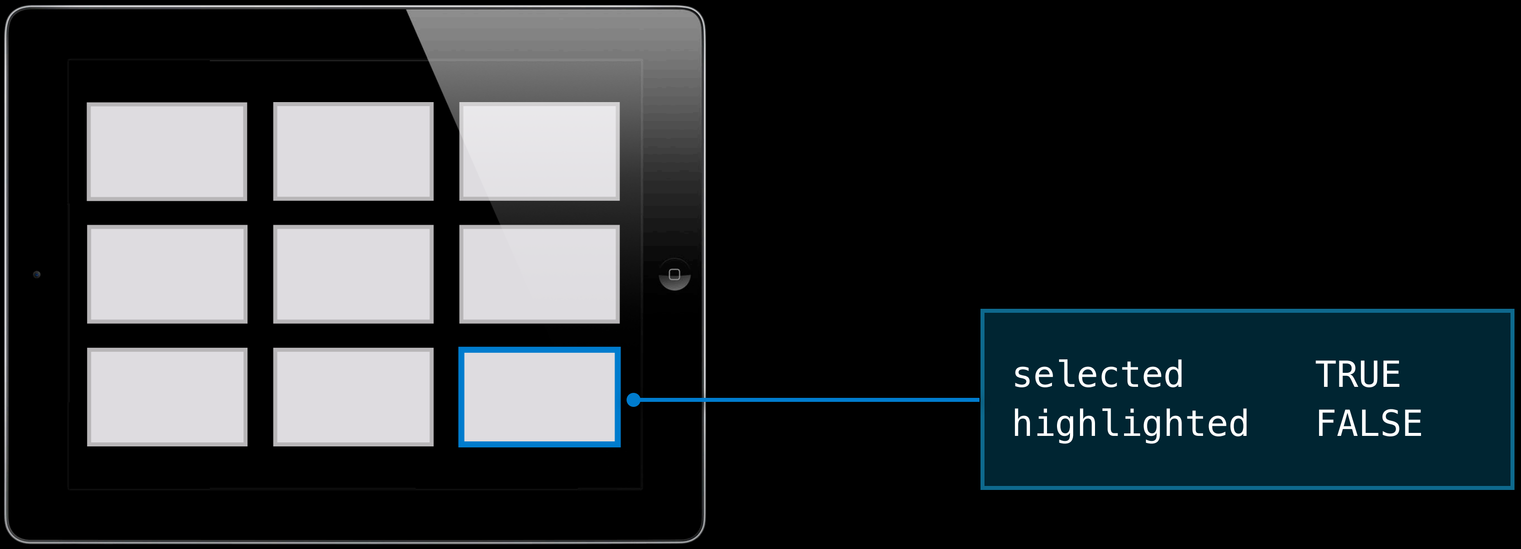


```
selected       FALSE
highlighted    TRUE
```

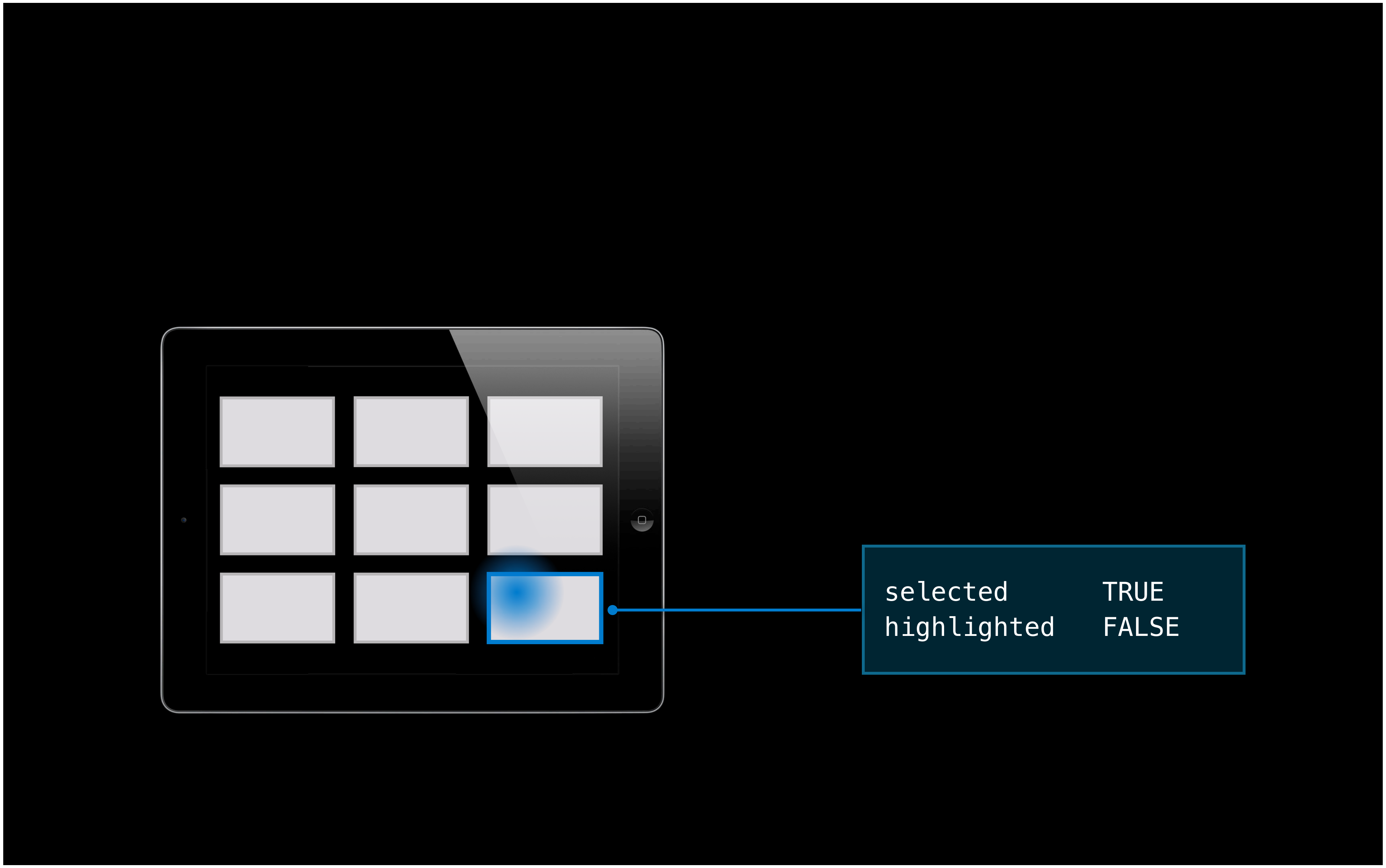
```
collectionView:didUnhighlightItemAtIndexPath:
collectionView:didSelectItemAtIndexPath:
```



```
selected      TRUE
highlighted   FALSE
```

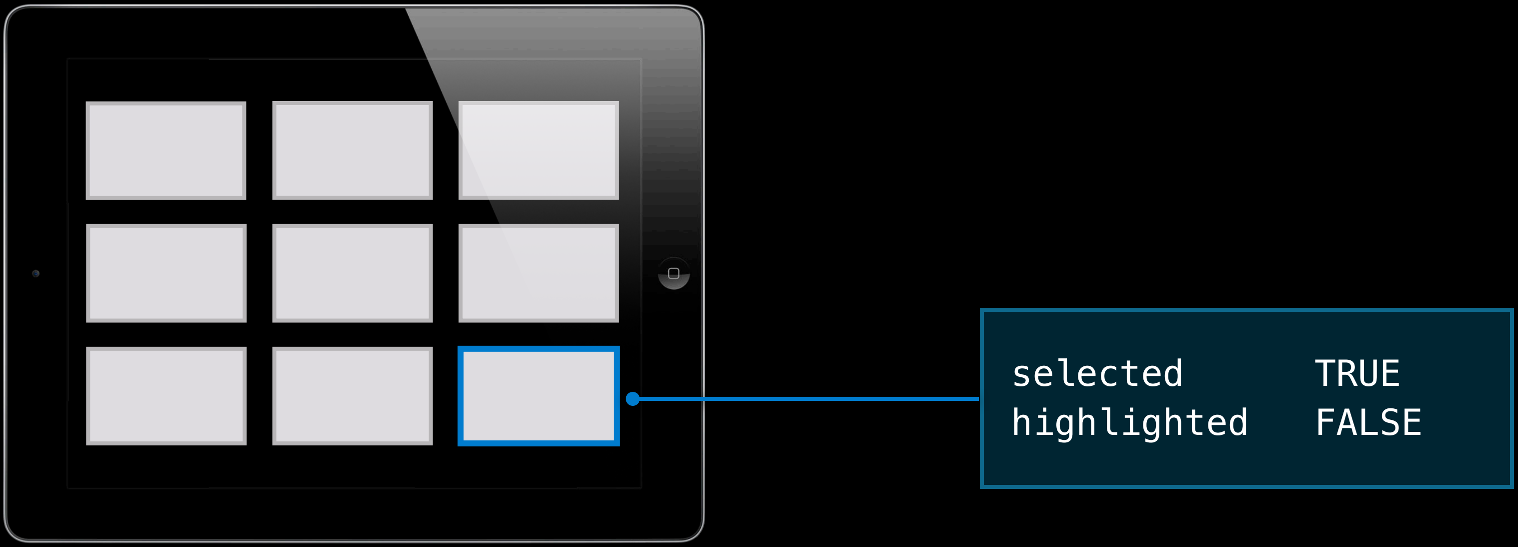


```
selected      TRUE
highlighted   FALSE
```



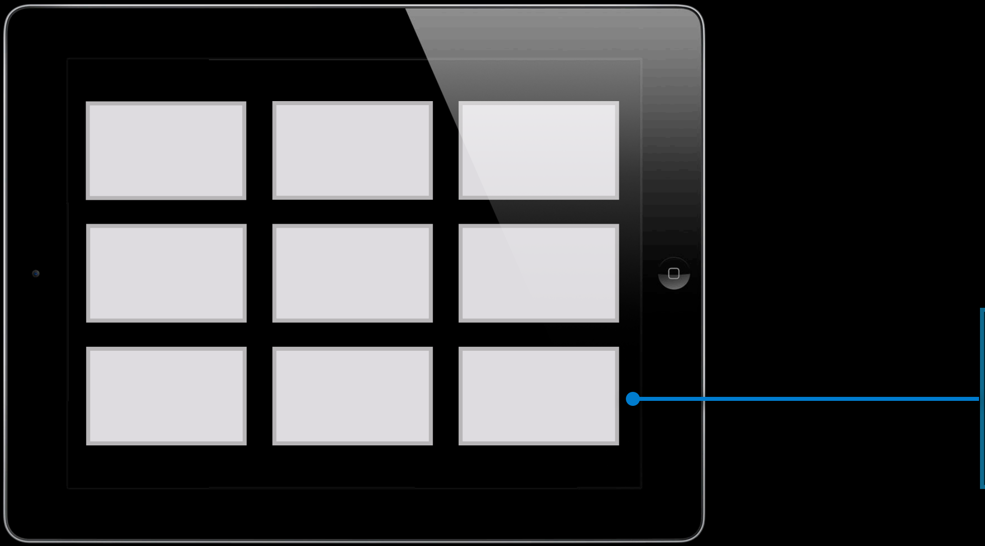
```
selected      TRUE
highlighted   FALSE
```

collectionView:shouldDeselectItemAtIndexPath:



```
selected      TRUE
highlighted   FALSE
```

collectionView:didDeselectItemAtIndexPath:



```
selected      FALSE
highlighted   FALSE
```

Visual Content

Cells and layouts

UICollectionViewCell ~~Styles~~

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)
  - By switching background view and selected background view if configured

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)
  - By switching background view and selected background view if configured

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)
  - By switching background view and selected background view if configured

Background View

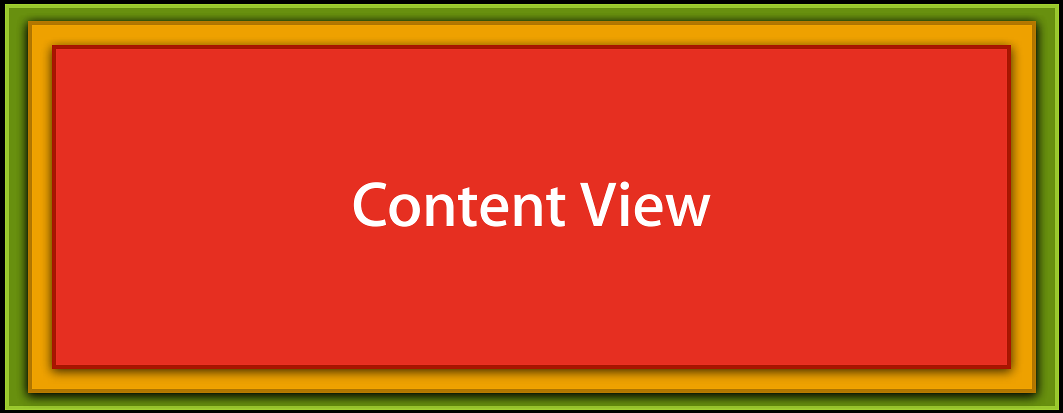
UICollectionViewCell ~~Styles~~

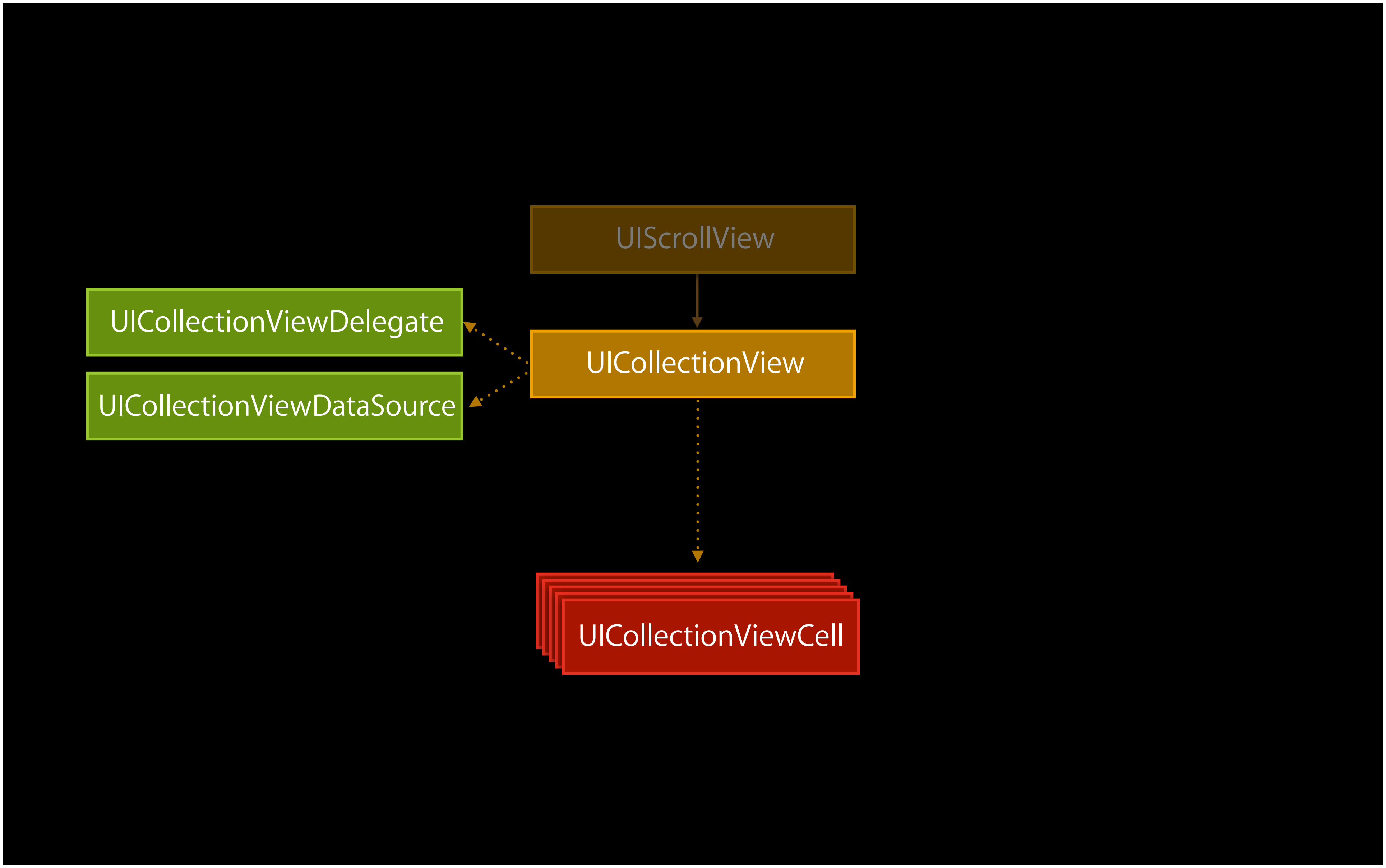
- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)
  - By switching background view and selected background view if configured

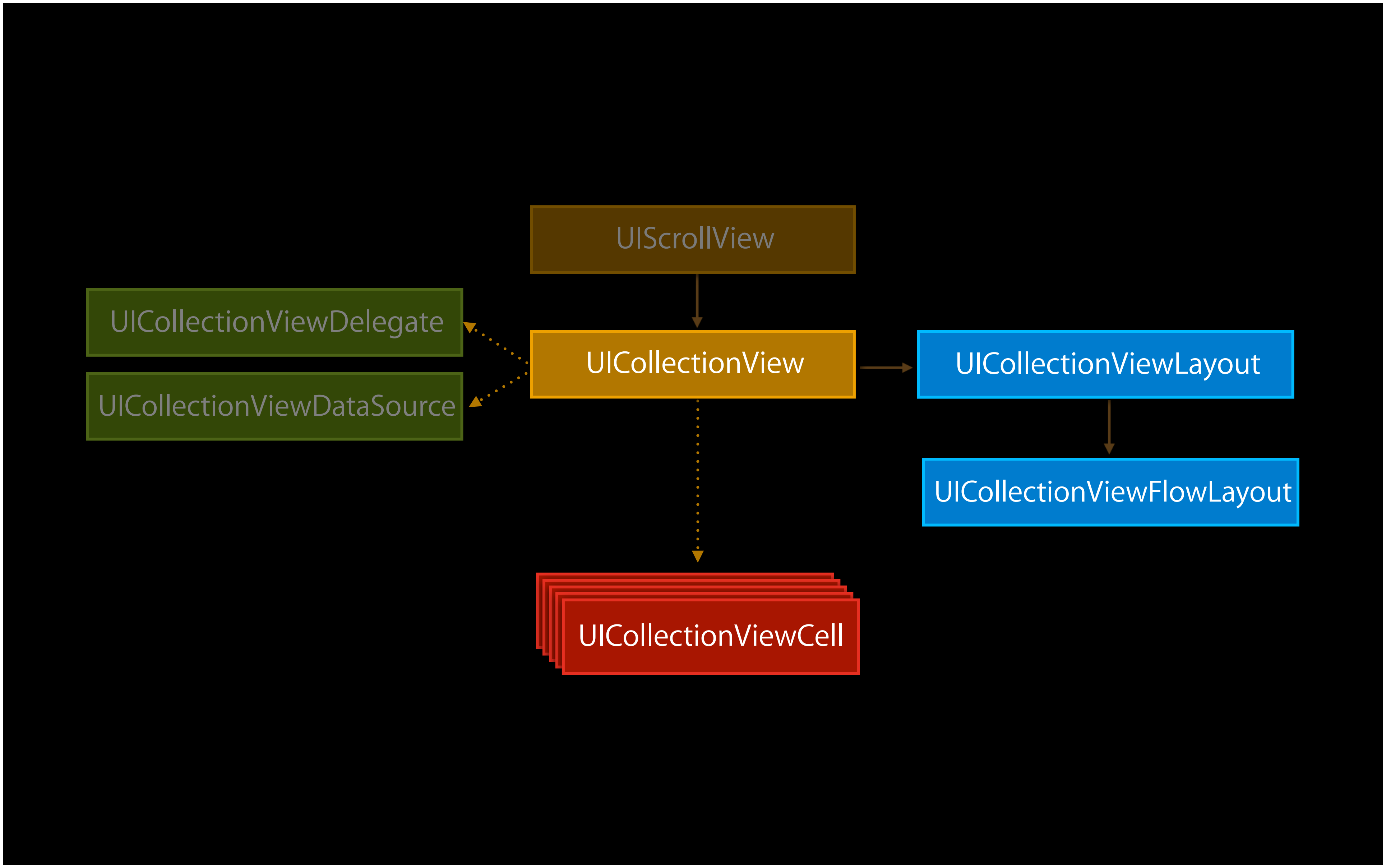
Selected Background View

UICollectionViewCell ~~Styles~~

- We do not provide predefined styles in cells
- Collection view does track selection and highlight
  - By setting the highlight and selection properties (including subviews)
  - By switching background view and selected background view if configured




UIScrollView
UICollectionViewDelegate
UICollectionView
UICollectionViewDataSource
UICollectionViewCell


UIScrollView
UICollectionViewDelegate
UICollectionView
UICollectionViewLayout
UICollectionViewDataSource
UICollectionViewFlowLayout
UICollectionViewCell

BYOL—Bring Your Own Layout

UICollectionViewLayout

- Compute layout attributes as needed for
  - Cells
  - Supplementary views
  - Decoration views

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

- Position

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

- Position
- Size

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

- Position
- Size
- Opacity

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

- Position
- Size
- Opacity
- zIndex

BYOL—Bring Your Own Layout

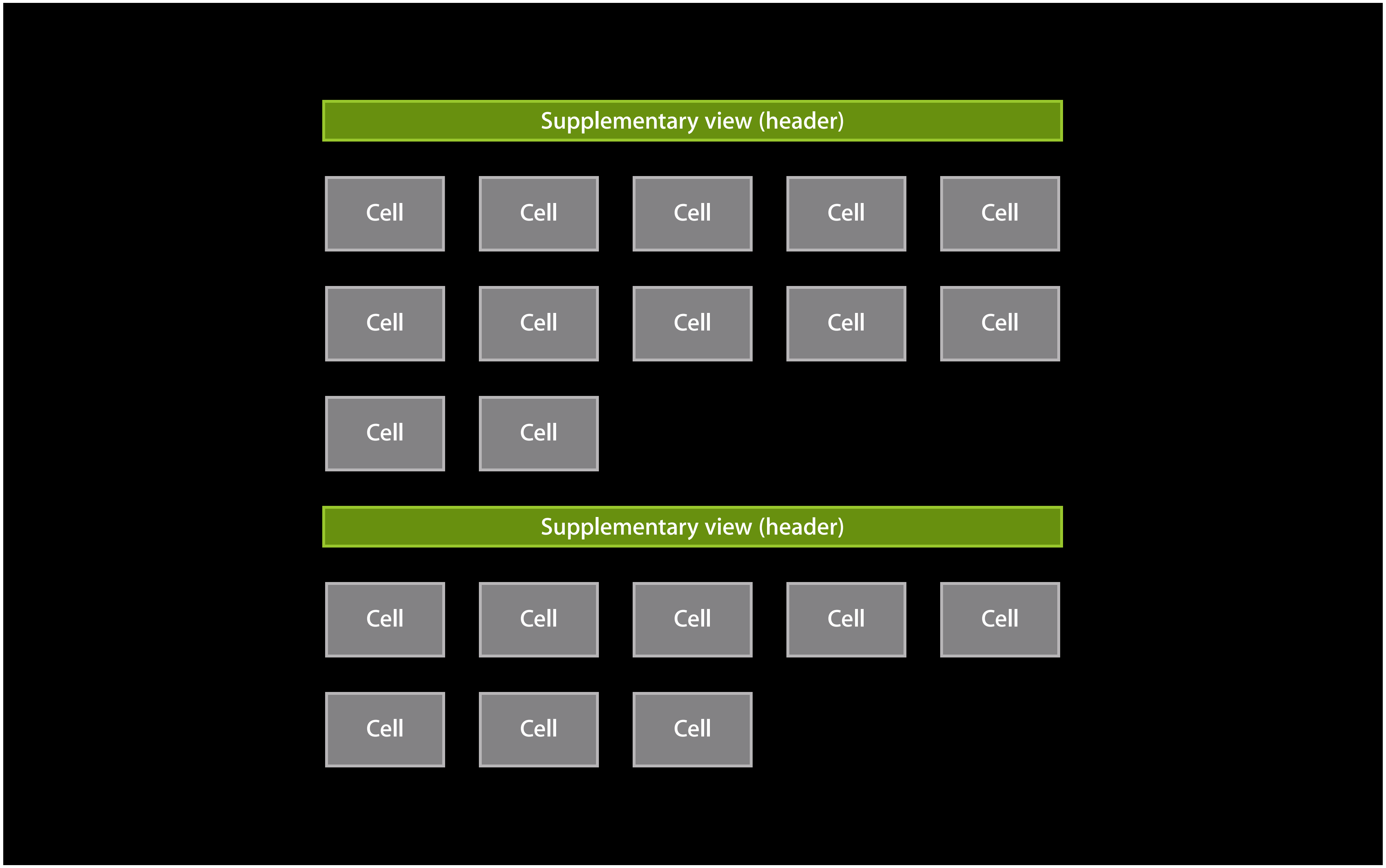
UICollectionViewLayoutAttributes

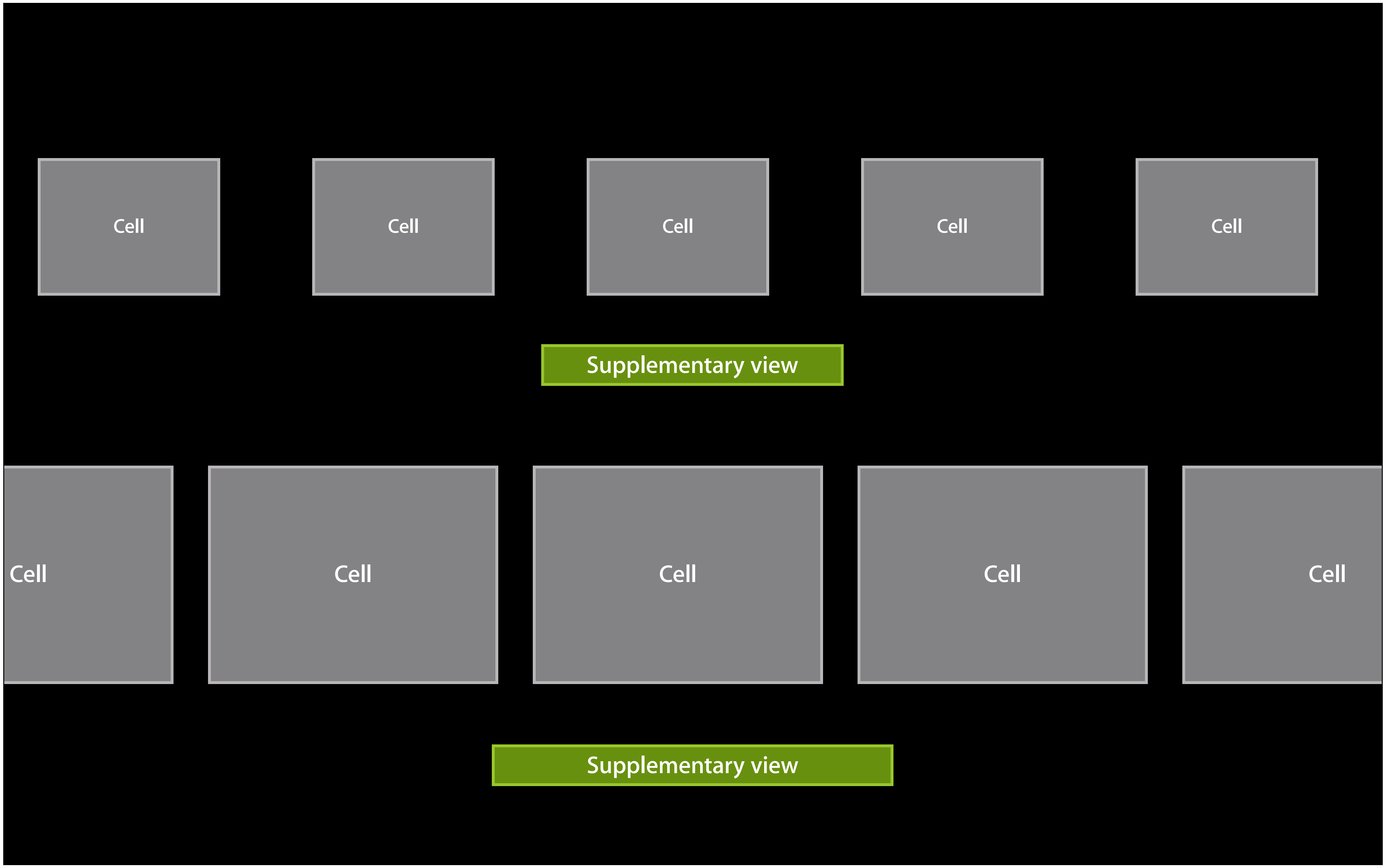
- Position
- Size
- Opacity
- zIndex
- Even transform

BYOL—Bring Your Own Layout

UICollectionViewLayoutAttributes

- Position
- Size
- Opacity
- zIndex
- Even transform
- …


Supplementary view (header)
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Supplementary view (header)
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell

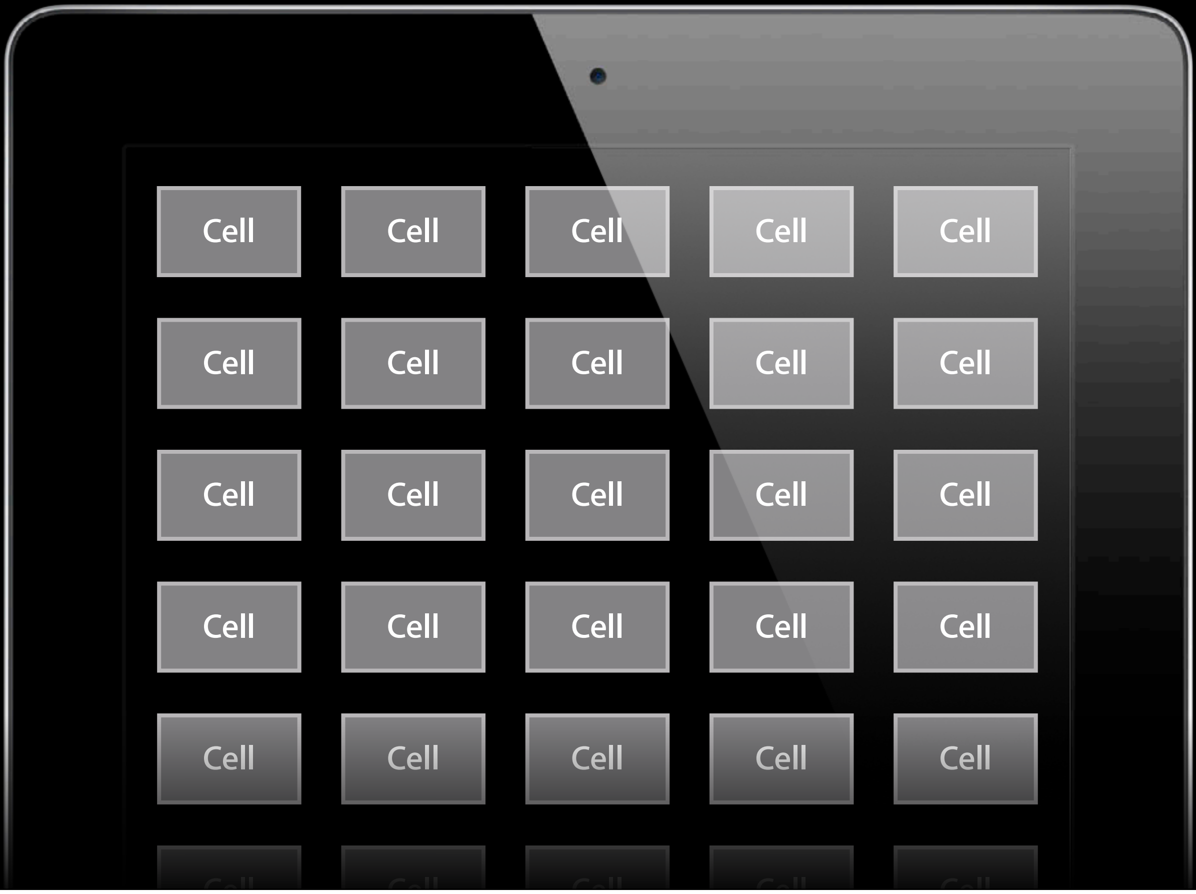

Cell
Cell
Cell
Cell
Cell
Supplementary view
Cell
Cell
Cell
Cell
Cell
Supplementary view

Flow Layout

Mathieu Martin
iOS Application and Framework Engineer

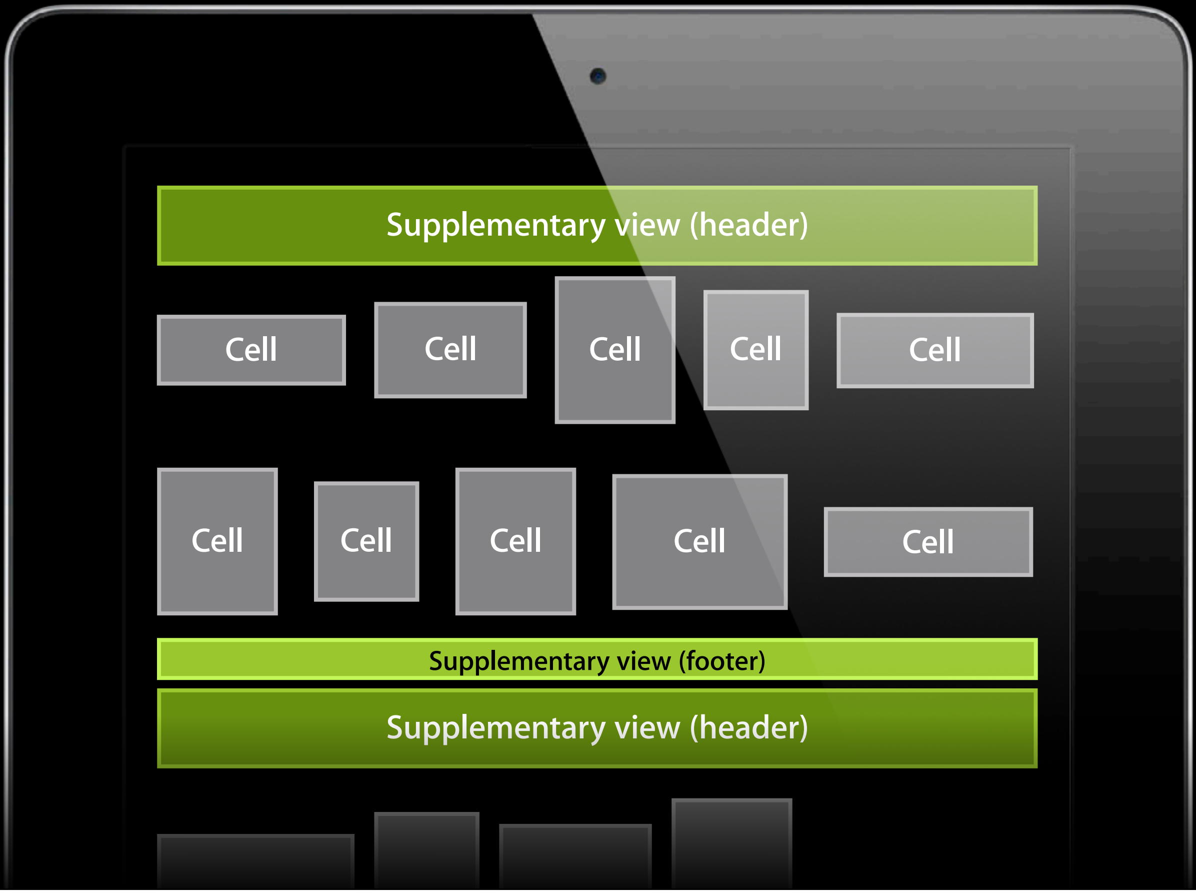
UICollectionViewFlowLayout

From simple...



UICollectionViewFlowLayout

...to complex


Supplementary view (header)
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Cell
Supplementary view (footer)
Supplementary view (header)

Demo

Flow Layout

- A line-oriented layout
- Could be configured as a grid
- …or as a group of lines
- Headers and footers, redux

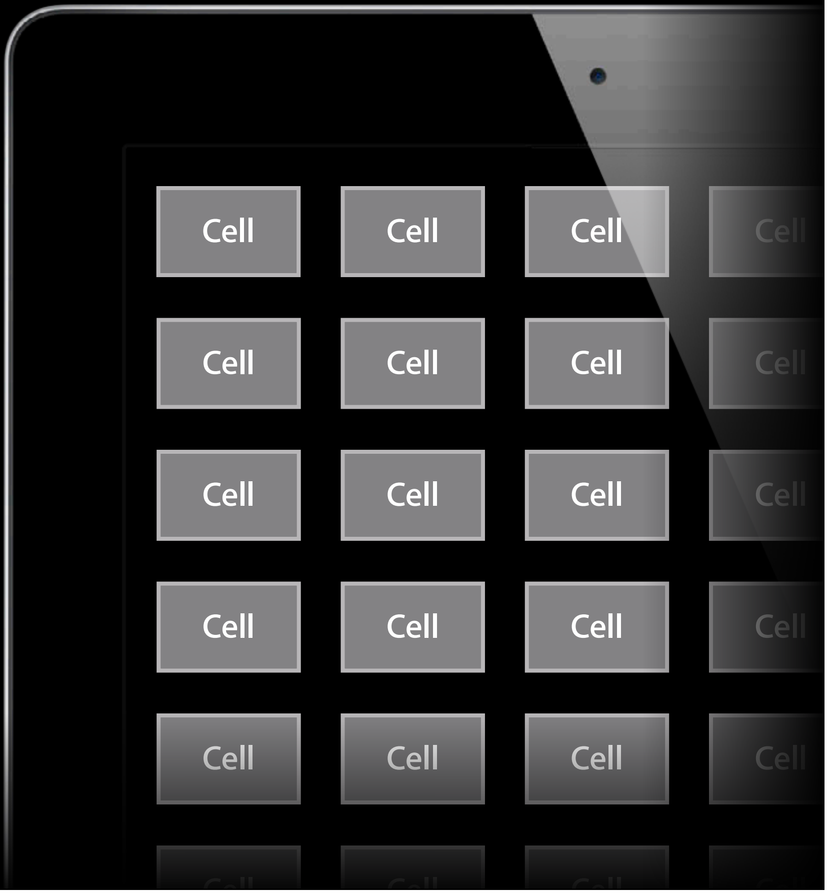
Customization

- Item size
- Line spacing
- Inter cell spacing
- Scrolling direction
- Header and footer size
- Section Inset

Item Size

- Can be configured globally

```
@property (CGSize)itemSize

layout.itemSize = CGSizeMake(30,20);
```



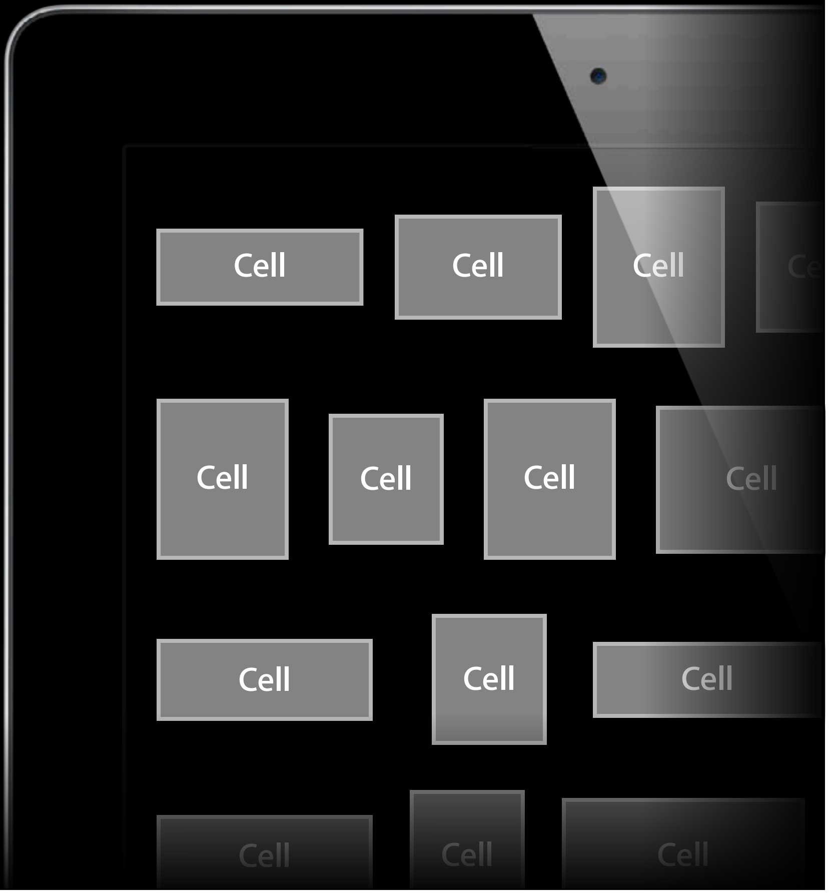
Item Size

- Can be configured globally

```
@property (CGSize)itemSize

layout.itemSize = CGSizeMake(30,20);
```

- Or per item, through delegate

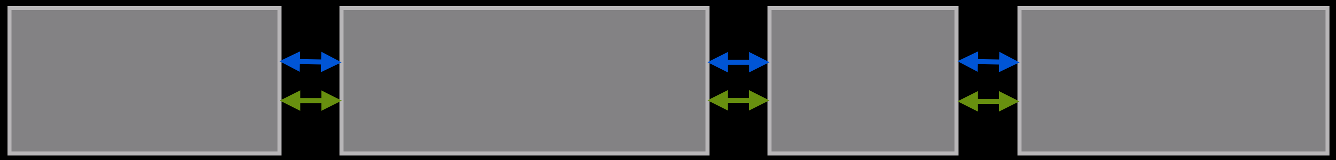
```
collectionView:layout:sizeForItemAt
IndexPath:
{
    return ...;
}
```



Inter-Item Spacing

Minimum Inter-Item Spacing

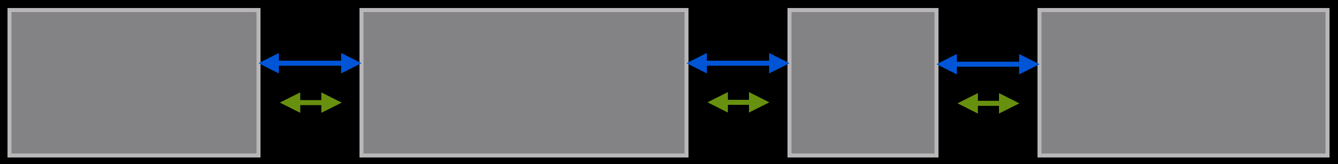
Minimum Inter-Item Spacing



↔ Actual interitem spacing

↔ Minimum interitem spacing

Minimum Inter-Item Spacing



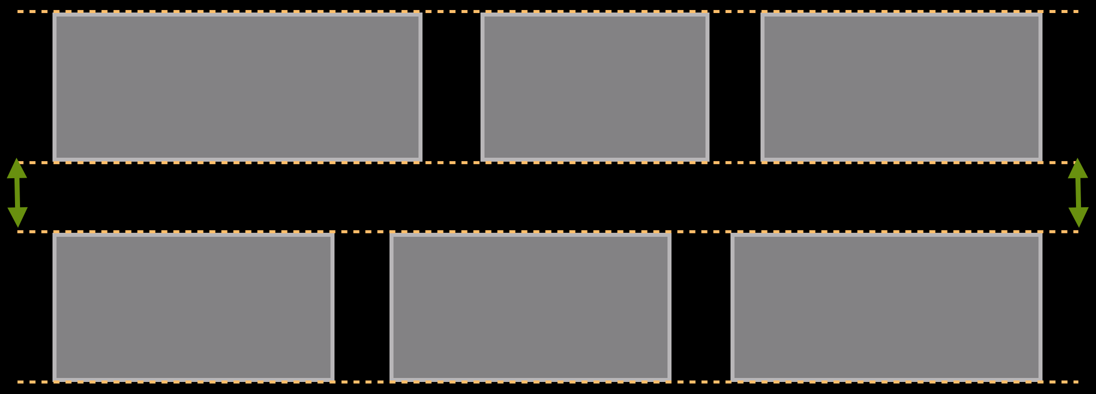
↔ Actual interitem spacing

↔ Minimum interitem spacing

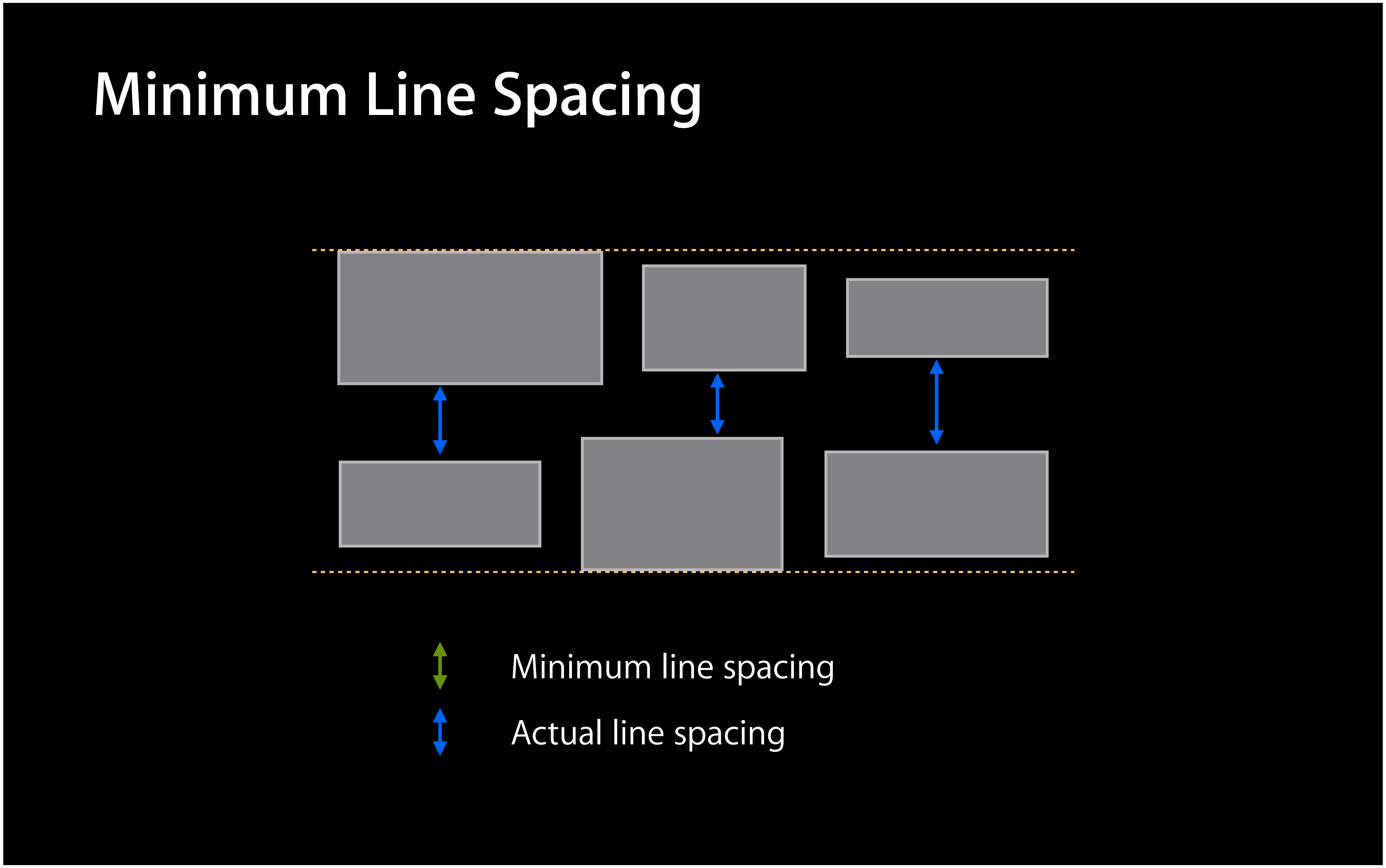
Line Spacing

Minimum Line Spacing

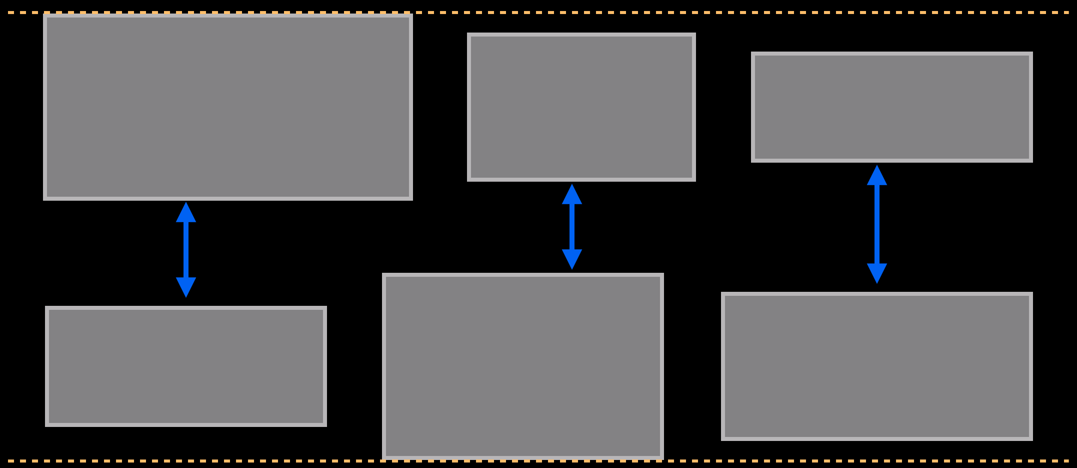
Minimum Line Spacing



↕ Minimum line spacing



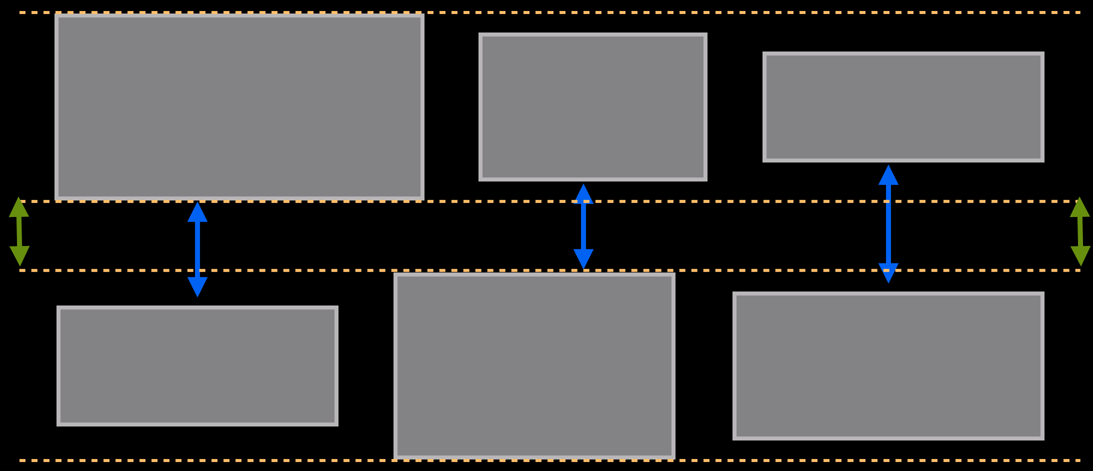
Minimum Line Spacing



↕ Minimum line spacing

↕ Actual line spacing

Minimum Line Spacing



Minimum line spacing

Actual line spacing

Spacing

- Can be configured globally

```
@property (CGFloat) minimumInteritemSpacing

@property (CGFloat) minimumLineSpacing
```

- Or per section, through delegate

```
collectionView:layout:minimumInteritemSpacingForSectionAtIndex:

collectionView:layout:minimumLineSpacingForSectionAtIndex:
```

There might be a pattern here…

Global or Per-Delegate Customization

- For almost all properties on `UICollectionViewFlowLayout`
- The delegate is actually the collection view delegate
- `UICollectionViewDelegateFlowLayout` extends `UICollectionViewDelegate`

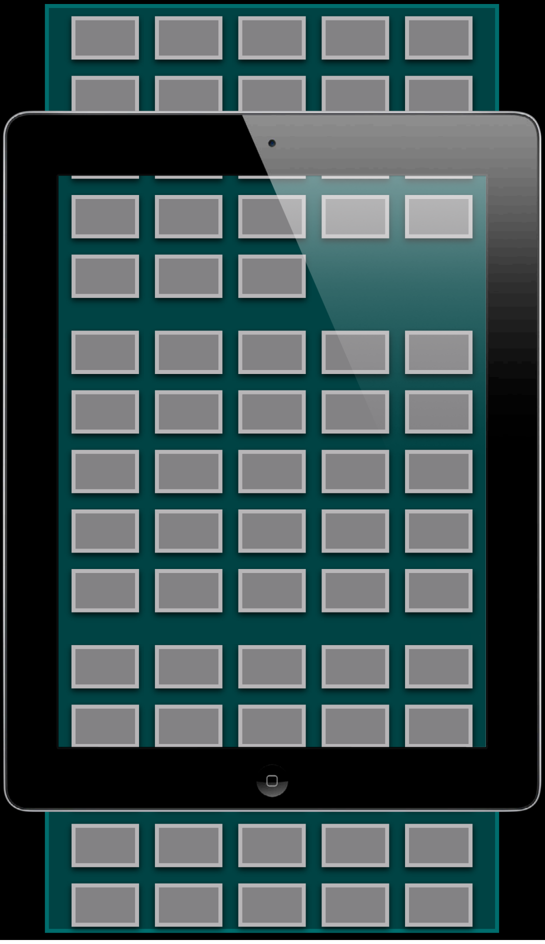
Scroll Direction

scrollDirection property

- Defines the base behavior of Flow Layout
- Controls the main dimension for headers and footers

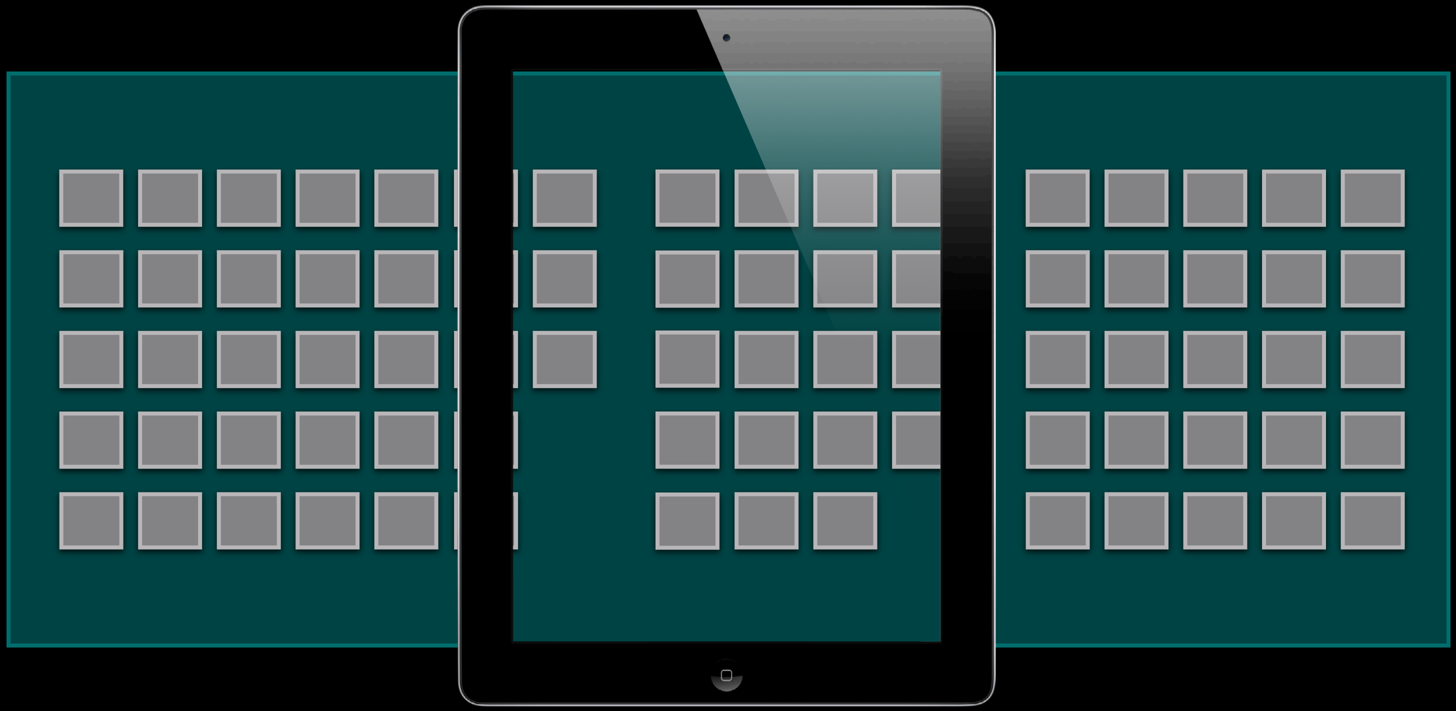
Scroll Direction

UICollectionViewScrollDirectionVertical



Scroll Direction

UICollectionViewScrollDirectionHorizontal



Headers and Footers

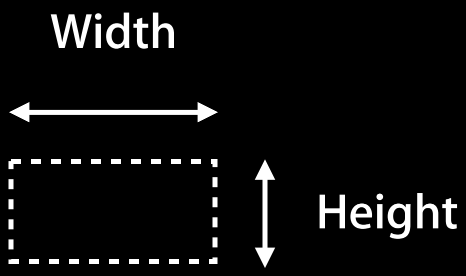
- Can be configured globally

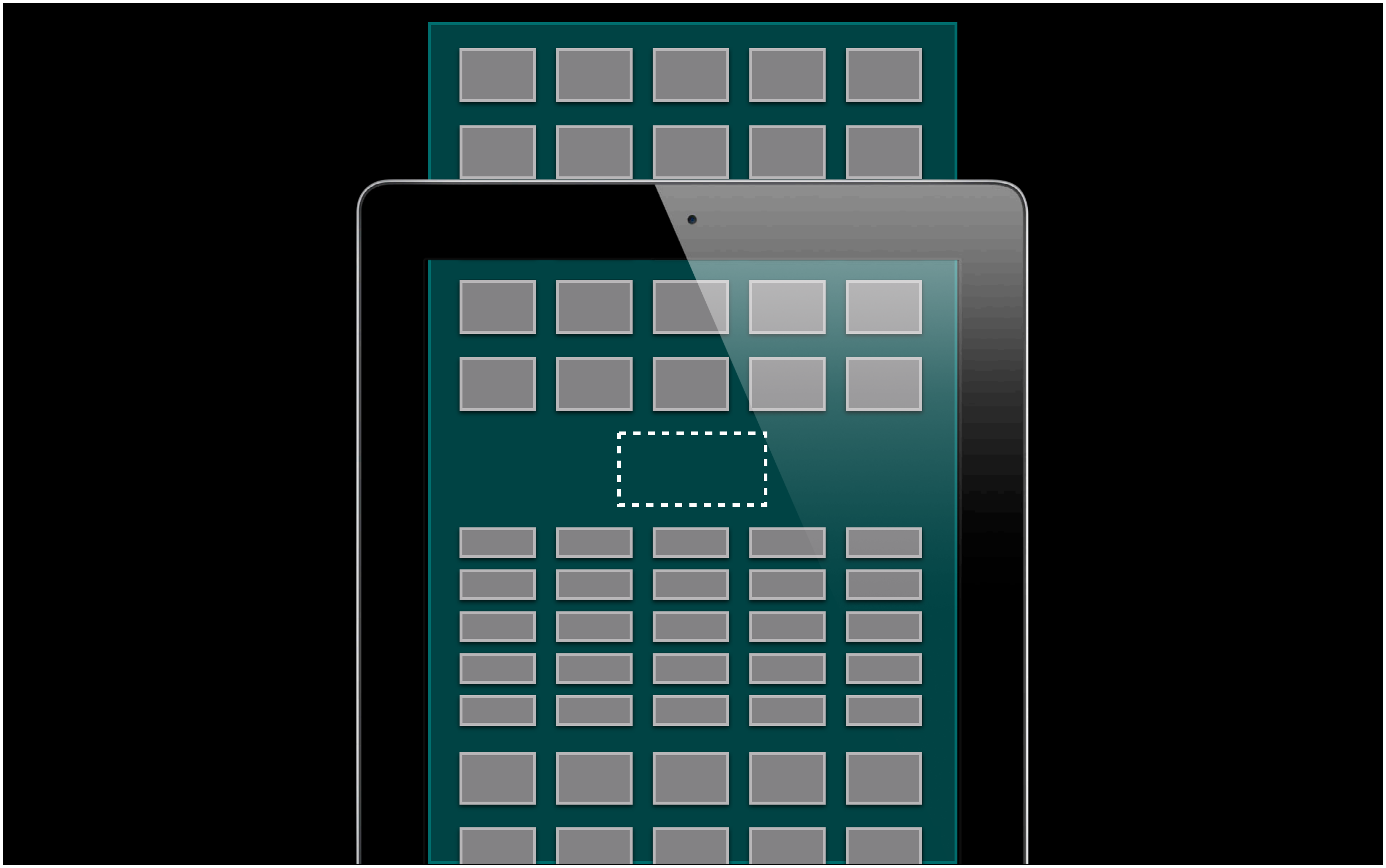
```
@property (CGSize) headerReferenceSize

@property (CGSize) footerReferenceSize
```

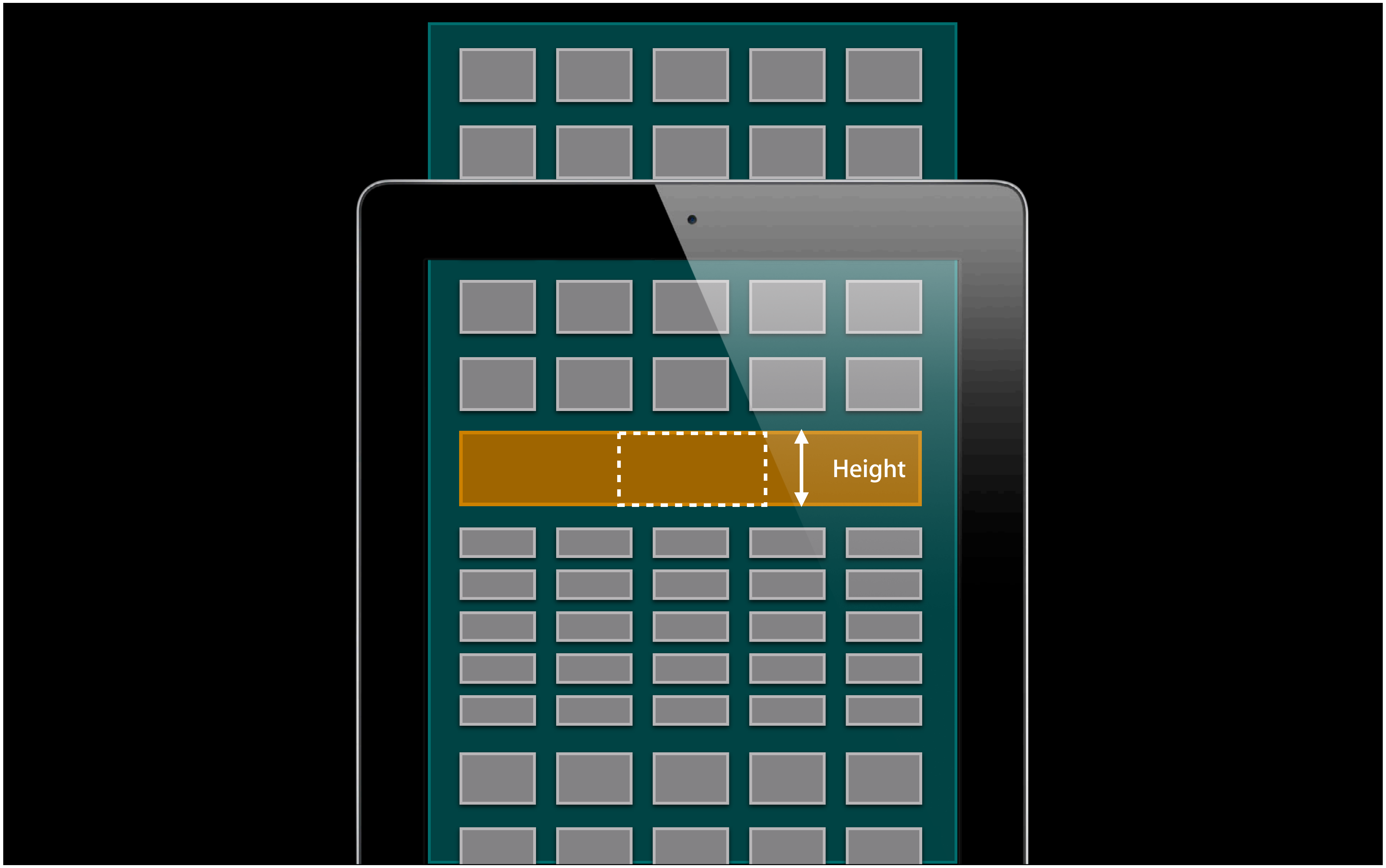
- Or per section, through delegate

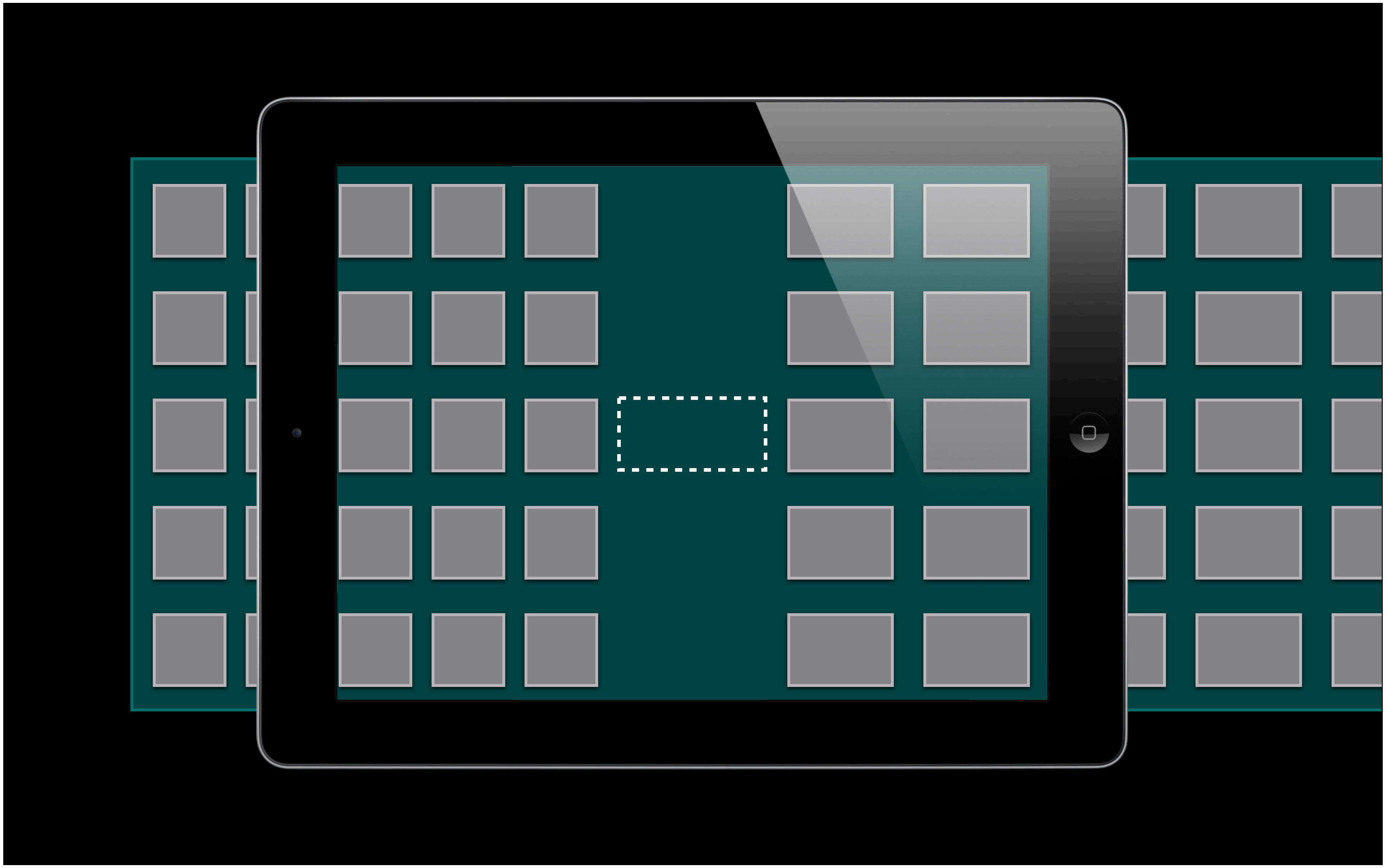
```
collectionView:layout:referenceSizeForHeaderInSection:

collectionView:layout:referenceSizeForFooterInSection:
```

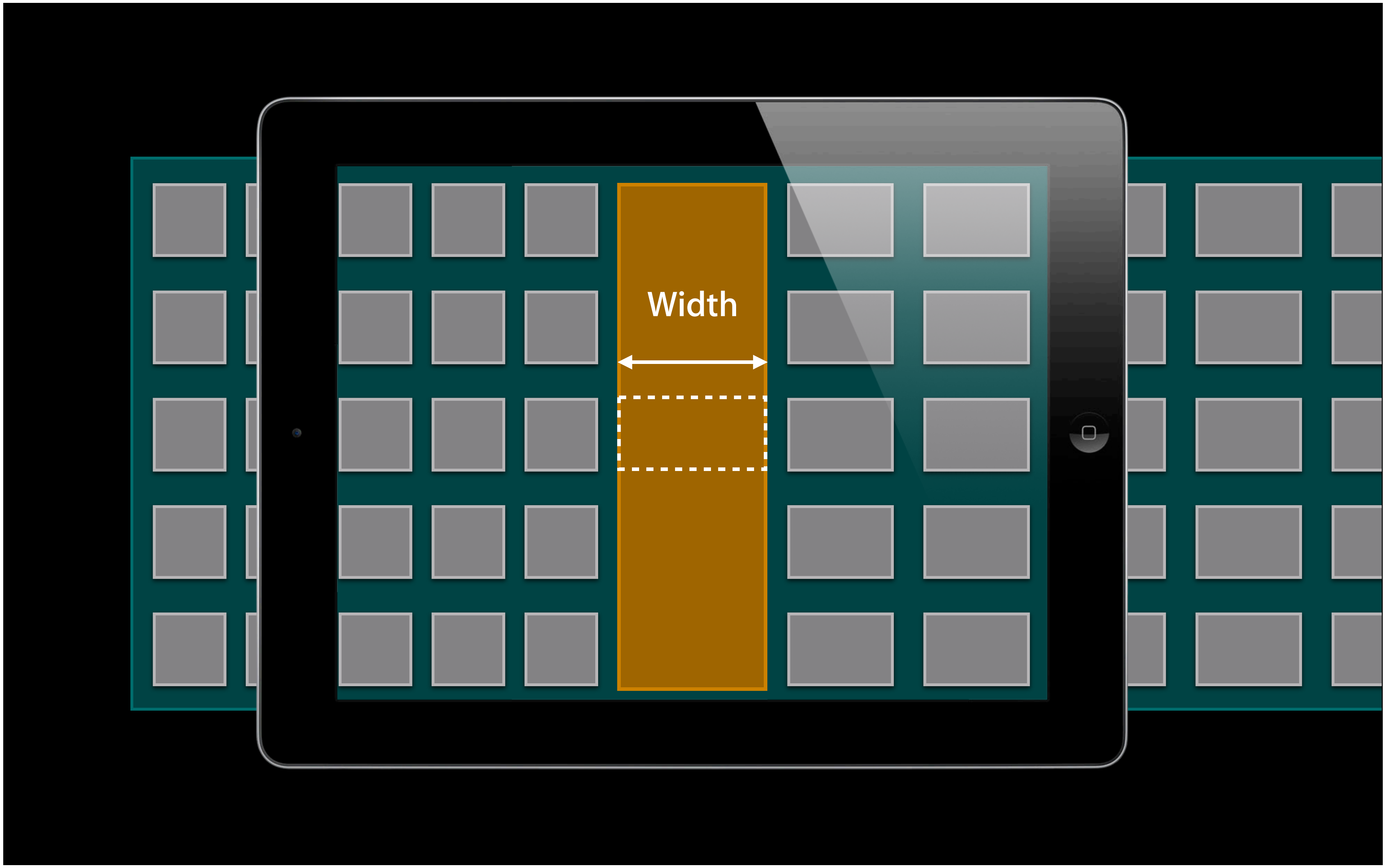
Headers and Footers






Height




Width

Headers and Footers

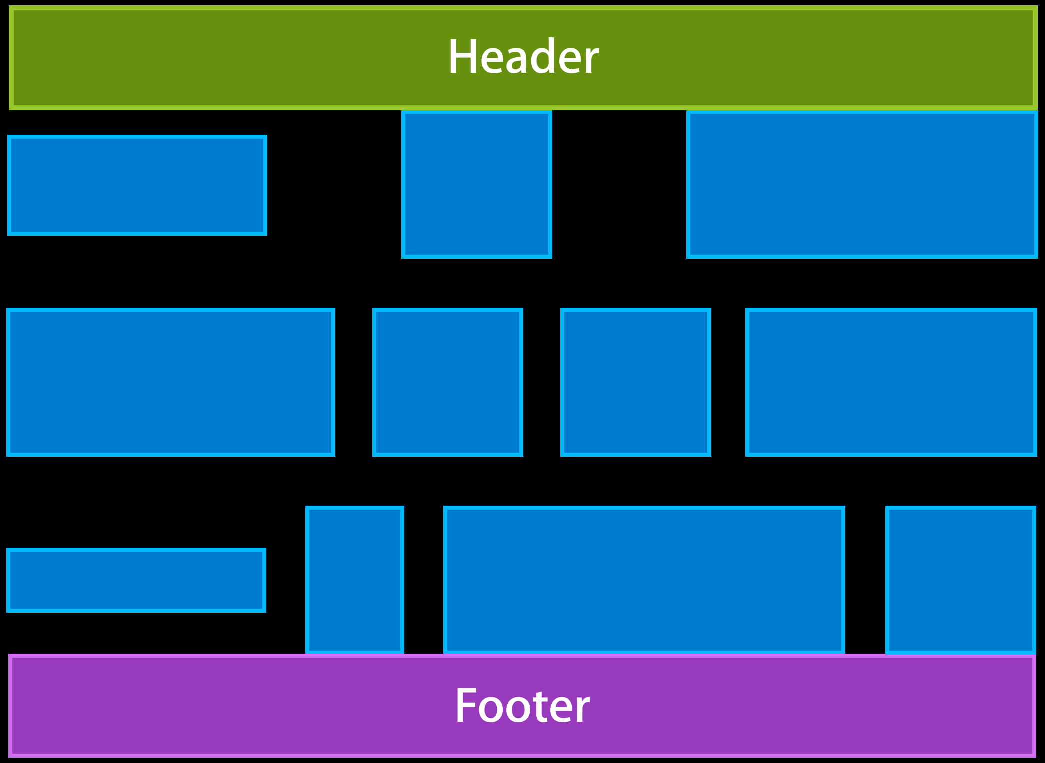
- Just supplementary views!
- Content is provided by the usual data source method
  - `collectionView:viewForSupplementaryElementOfKind:atIndexPath:`
- Two "kind" constants:

```
UICollectionElementKindSectionHeader
UICollectionElementKindSectionFooter
```

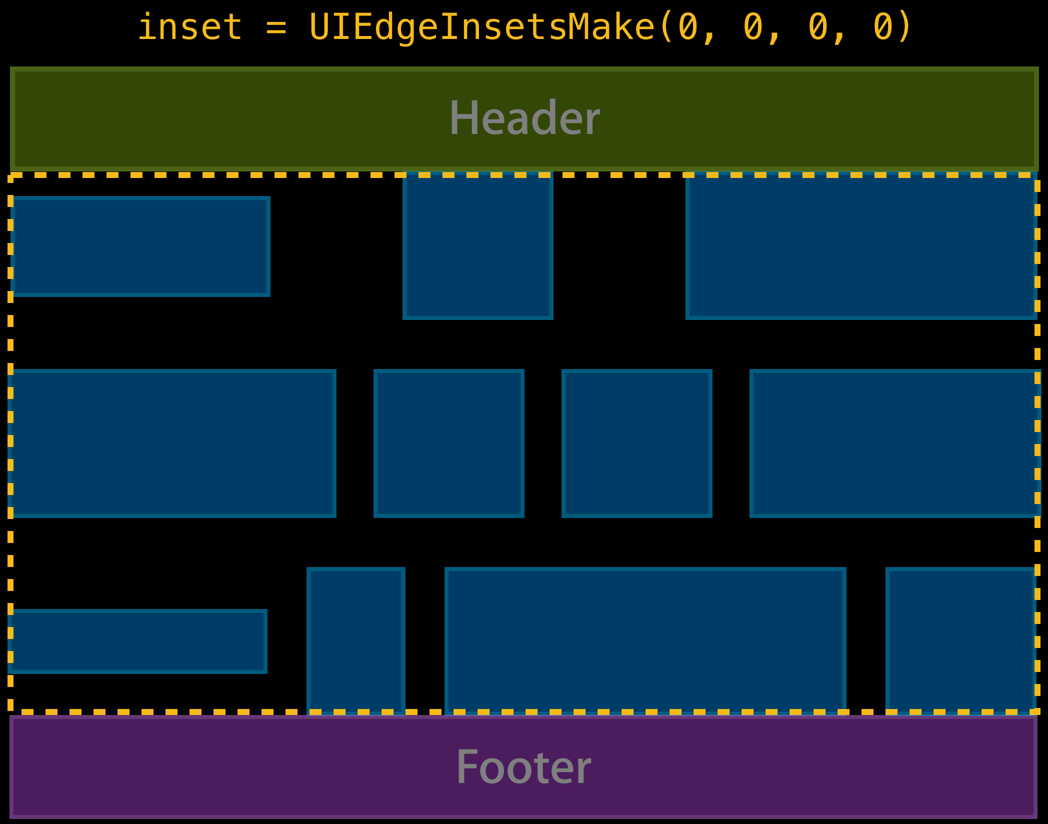
- You still need to register a class and dequeue
  - `registerClass:forSupplementaryViewOfKind:withReuseIdentifier:`
  - `registerNib:forSupplementaryViewOfKind:withReuseIdentifier:`
  - `dequeueReusableSupplementaryViewOfKind:withReuseIdentifier:forIndexPath:`

Sections Insets

Sections Insets



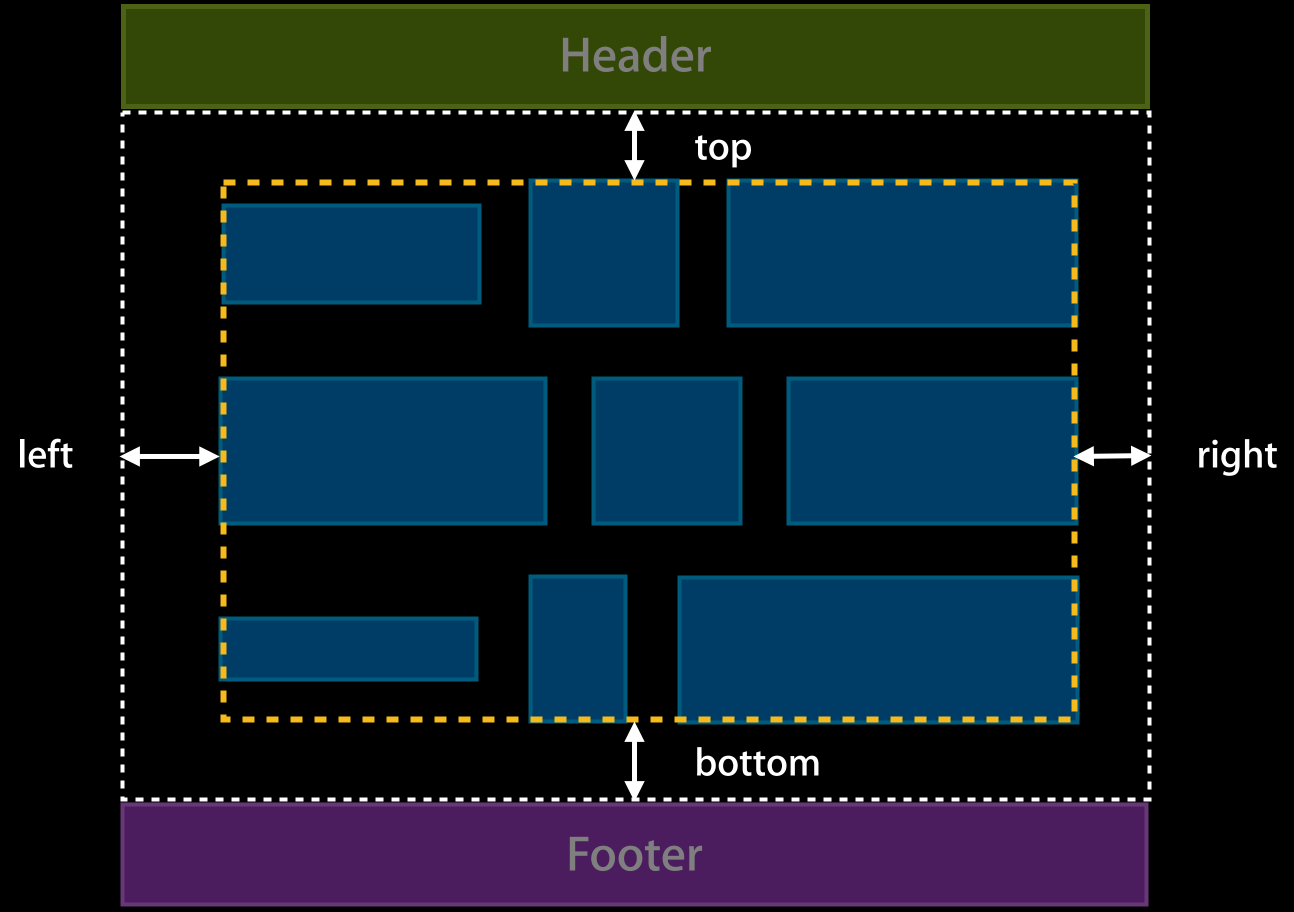
Sections Insets



```
inset = UIEdgeInsetsMake(0, 0, 0, 0)
```

Sections Insets

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



Sections Insets

A bounding box for your cells

- Can be configured globally

```
@property UIEdgeInsets sectionInset;
```

- Or per section, through delegate

```
- (UIEdgeInsets)collectionView:layout:insetForSectionAtIndex:
```

Flow Layout

- Powerful. Really
- A lot of built-in behaviors
- Can be a great time saver compared to writing your own layout
- And much more with subclassing

Putting It All Together

Recap

- A data-source driven view
- Selection and Highlight tracking built in
- Cells, Supplementary, and Decoration Views
- Layout is provided by a separate class

But There’s More

- Fine-grain, block-based updates
- Animations and relayout
- Scroll view integration
- Custom layouts

Collection View to the Max

“It goes to 11”

AMP
INPUT
0
11
BASS
0
11
MIDDLE
0
11
TREBLE
0
11
REVERB
0
11
PRESENCE
0
11
VOLUME
POWER

Demo

Summary

- Collection view is all about your content
- Flow Layout makes things simple
- Opportunities

More Information

Jake Behrens
UI Frameworks Evangelist
behrens@apple.com

Documentation
UIKit Framework Reference
http://developer.apple.com/library/ios

Apple Developer Forums
http://devforums.apple.com

Related Sessions

Advanced Collection Views and Building Custom Layouts	Mission Wednesday 11:30AM

Labs

Collection View on iOS Lab	Essentials Lab A Thursday 2:00PM

WWDC2012