What’s New in Table and Collection Views

Session 226
Luke Hiesterman
UI Kit Engineer

Olivier Gutknecht
iOS Engineering Manager
Agenda
| Table view |   |
Agenda

Table view
• Dynamic Type adoption
Agenda

Table view
- Dynamic Type adoption
- Self-sizing cells
Agenda

Table view
• Dynamic Type adoption
• Self-sizing cells

Collection view
Agenda

Table view
• Dynamic Type adoption
• Self-sizing cells

Collection view
• Self-sizing cells
Agenda

Table view
• Dynamic Type adoption
• Self-sizing cells

Collection view
• Self-sizing cells
• Smart invalidation
Table View

Adopting dynamic type
Dynamic Type

Apps that support Dynamic Type will adjust to your preferred reading size below.

Drag the slider below

Settings

- Airplane Mode
- Wi-Fi
- Bluetooth
- Cellular
- Notifications
- Control Center
- Do Not Disturb
- General
- Wallpaper & Brightness
Dynamic Type

Apps that support Dynamic Type will adjust to your preferred reading size below.

Drag the slider below

Larger sizes are available in Accessibility Settings.
Adopting Dynamic Type
Adopting Dynamic Type

Comes free with built-in labels
Adopting Dynamic Type

Comes free with built-in labels
+
[UIFont preferredFontForTextStyle:] for custom labels
Dynamic Row Heights

Everything is dynamic
Dynamic Row Heights
Everything is dynamic
Dynamic Row Heights

Everything is dynamic
Dynamic Row Heights

Everything is dynamic
Dynamic Row Heights

Everything is dynamic
Dynamic Row Heights

Everything is dynamic
Dynamic Row Heights

Strategies
Dynamic Row Heights

Strategies

Property—rowHeight
Dynamic Row Heights

Strategies

Property—rowHeight
Delegate—tableView:heightForRowAtIndexPath:
Dynamic Row Heights

Strategies

Property—rowHeight
Delegate—tableView:heightForRowAtIndexPath:
Cell—self-sizing cells
Self-sizing Table Cells
Cell Sizing
Cell Sizing

Master
email, yo
I have no signature, home slice.
Luke

11/5/13
Cell Sizing

Master
email, yo
I have no signature, home slice.
Luke

11/5/13
Cell Sizing

Fonts

Master
email, yo
I have no signature, home slice.
Luke

11/5/13
Cell Sizing

Master
email, yo
I have no signature, home slice.
Luke

11/5/13
Cell Sizing

Margins

Master
email, yo
I have no signature, home slice.
Luke

11/5/13
iOS 8 Sizing Flow
iOS 8 Sizing Flow

Row 1

Row 2

Row 3
iOS 8 Sizing Flow

Row 1
Row 2
Row 3
Row 4
iOS 8 Sizing Flow

Estimated height

Row 1
Row 2
Row 3
Row 4
iOS 8 Sizing Flow

Row 1

Row 2

Row 3

Row 4

Estimated height
iOS 8 Sizing Flow

Create cell

Row 1
Row 2
Row 3
Row 4

Estimated height
iOS 8 Sizing Flow

Create cell

Size cell

Row 1

Row 2

Row 3

Row 4

Estimated height
iOS 8 Sizing Flow

Create cell

Size cell

Row 1

Row 2

Row 3

Row 4
iOS 8 Sizing Flow

Create cell → Size cell → Update contentSize
iOS 8 Sizing Flow

Create cell
- Size cell
- Update contentSize
- Display cell
iOS 8 Sizing Flow

Create cell → Size cell → Update contentSize → Display cell
Self-sizing Cells
Two ways to cut the cake
Self-sizing Cells

Two ways to cut the cake

Autolayout sizing
Self-sizing Cells

Two ways to cut the cake

Autolayout sizing

• Add constraints to cell.contentView
Self-sizing Cells

Two ways to cut the cake

Autolayout sizing
• Add constraints to cell.contentView

Manual-sizing code
Self-sizing Cells

Two ways to cut the cake

Autolayout sizing
• Add constraints to cell.contentView

Manual-sizing code
• Override -sizeThatFits:
Demo
Dynamic Type and self-sizing cells
Collection View

Olivier Gutknecht
iOS Engineering Manager
Today
Today

Self-sizing cells
Today

Self-sizing cells
Smart invalidation
Self-sizing Cells
Self-sizing Cells

New major feature
Self-sizing Cells

New major feature
Available in flow layout
Self-sizing Cells

New major feature
Available in flow layout
Ready for custom layouts!
Demo
Self-sizing cells
Cell Sizing Strategies

Classic
All cell sizes are computed by a collection view layout

Self-sizing Cells
Use constraints on the collection view cell content view, or override sizeThatFits:

Full Control
Use preferredLayoutAttributesFittingAttributes: to change other attributes properties or use custom-sizing code
## Cell Sizing Strategies

<table>
<thead>
<tr>
<th>Classic</th>
<th>All cell sizes are computed by a collection view layout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-sizing Cells</td>
<td>Use constraints on the collection view cell content view, or override <code>sizeThatFits:</code></td>
</tr>
</tbody>
</table>
# Cell Sizing Strategies

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classic</td>
<td>All cell sizes are computed by a collection view layout</td>
</tr>
<tr>
<td>Self-sizing Cells</td>
<td>Use constraints on the collection view cell content view, or override <code>sizeThatFits:</code></td>
</tr>
<tr>
<td>Full Control</td>
<td>Use <code>preferredLayoutAttributesFittingAttributes:</code> to change other attributes properties or use custom-sizing code</td>
</tr>
</tbody>
</table>
Cell-sizing Strategies
Cell-sizing Strategies

Layout-driven attributes
Cell-sizing Strategies

Layout-driven attributes

Layout-estimated attributes
Cell-sizing Strategies

- Layout-driven attributes
- Layout-estimated attributes
- Self-sizing cell
Cell-sizing Strategies

- Layout-driven attributes
- Layout-estimated attributes
- Self-sizing cell
- Layout-estimated attributes
Cell-sizing Strategies

- Layout-driven attributes
- Layout-estimated attributes
  - Self-sizing cell
- Layout-estimated attributes
  - Preferred attributes
  - Fitting attributes
Flow Layout Support
Flow Layout Support

@property (nonatomic) CGSize estimatedItemSize;
@property (nonatomic) CGSize estimatedItemSize;
Equivalent to estimated row height in table views
Flow Layout Support

@property (nonatomic) CGSize estimatedItemSize;
Equivalent to estimated row height in table views
Set it to a non-zero CGSize
Self-sizing Cells in Flow Layout

Collection View

Collection View Layout

Collection View Cell
Self-sizing Cells in Flow Layout

Collection View

Collection View Cell

Collection View Layout
Self-sizing Cells in Flow Layout

Collection View → Collection View Layout

Collection View Cell

A layout computes a first approximation using estimated size
Self-sizing Cells in Flow Layout

Collection View

Collection View Layout

Collection View Cell
Self-sizing Cells in Flow Layout

Collection View

Collection View Cell

Cells are created and self-sized

Collection View Layout
Self-sizing Cells in Flow Layout

Collection View

Collection View Cell

Collection View Layout
Self-sizing Cells in Flow Layout

Collection View  →  Collection View Layout

Collection View Cell  →  The layout receives updated attributes
Self-sizing Cells in Flow Layout

Collection View

Collection View Cell

Collection View Layout
Self-sizing Cells in Flow Layout

Collection View

Collection View Layout

Collection View Cell

And returns final attributes
Collection View Invalidation
Collection View Invalidation

-prepareLayout
Collection View Invalidation

- prepareLayout
- collectionViewContentSize
Collection View Invalidation

- `prepareLayout`
- `collectionViewContentSize`
- `layoutAttributesForElementsInRect:`
Collection View Invalidation

- prepareLayout
- collectionViewContentSize
- layoutAttributesForElementsInRect:

All other calls
Collection View Invalidation

- `prepareLayout`
- `collectionViewContentSize`
- `layoutAttributesForElementsInRect`:

All other calls

- `invalidateLayout`
Collection View Invalidation

- prepareLayout
- collectionViewContentSize
- layoutAttributesForElementsInRect:

All other calls
- invalidateLayout
- prepareLayout

...the cycle continues
How to Build High-performance Layouts
How to Build High-performance Layouts

Recompute only what you need!
How to Build High-performance Layouts

Recompute only what you need!
Use collection view invalidation contexts
How to Build High-performance Layouts

Recompute only what you need!

Use collection view invalidation contexts

New in iOS 8 iOS 7!
How to Build High-performance Layouts

Recompute only what you need!
Use collection view invalidation contexts
New in iOS 8 iOS 7!
Provide fine-grain information to your layout in invalidation situations
How to Build High-performance Layouts

Recompute only what you need!
Use collection view invalidation contexts
New in iOS 8 iOS 7!
Provide fine-grain information to your layout in invalidation situations
Flow layout is already using invalidation contexts in rotations
How to Build High-performance Layouts

Recompute only what you need!
Use collection view invalidation contexts
New in iOS 8 iOS 7!
Provide fine-grain information to your layout in invalidation situations
Flow layout is already using invalidation contexts in rotations
Now used for self-sizing cells
Invalidation Contexts
Invalidation Contexts

Define your own invalidation context class
+ (Class)invalidationContextClass
Define your own invalidation context class
+ (Class)invalidationContextClass

Invalidate with contextual information
- (void)invalidateLayoutWithContext(UICollectionViewLayoutInvalidationContext *)context;
Invalidation Contexts

Define your own invalidation context class
+ (Class)invalidationContextClass

Invalidate with contextual information
- (void)invalidateLayoutWithContext(UICollectionViewLayoutInvalidationContext *)context;

Override point for bounds change
- (UICollectionViewLayoutInvalidationContext *)invalidationContextForBoundsChange:(CGRect)newBounds;
Invalidation Contexts
Invalidation Contexts

@property (nonatomic, readonly) BOOL invalidateDataSourceCounts;
Invalidation Contexts

@property (nonatomic, readonly) BOOL invalidateDataSourceCounts;
@property (nonatomic, readonly) BOOL invalidateEverything;
Invalidation Contexts

@property (nonatomic, readonly) BOOL invalidateDataSourceCounts;
@property (nonatomic, readonly) BOOL invalidateEverything;

The collection view set in response to specific types of invalidation
Invalidation Contexts in iOS 8
Invalidation Contexts in iOS 8

Collaboration between your layout and collection view
Collaboration between your layout and collection view
You can pass information to the collection view to enable smart invalidation
Collaboration between your layout and collection view

You can pass information to the collection view to enable smart invalidation

• High-performance floating headers
Collaboration between your layout and collection view
You can pass information to the collection view to enable smart invalidation
• High-performance floating headers
• Avoid visual ‘jumps’ with self-sizing cells
Invalidation Contexts in iOS 8

Fine-grain invalidation
Invalidation Contexts in iOS 8

Fine-grain invalidation
- (void)invalidateItemsAtIndexPaths:(NSArray *)indexPaths;
Invalidation Contexts in iOS 8

Fine-grain invalidation
- (void)invalidateItemsAtIndexPaths:(NSArray *)indexPaths;
- (void)invalidateSupplementaryElementsOfKind:(NSString *)elementKind atIndexPaths:(NSArray *)indexPaths;
Invalidation Contexts in iOS 8

Fine-grain invalidation

- (void)invalidateItemsAtPath:(NSArray *)indexPaths;
- (void)invalidateSupplementaryElementsOfKind:(NSString *)elementKind atIndexPaths:(NSArray *)indexPaths;
- (void)invalidateDecorationElementsOfKind:(NSString *)elementKind atIndexPaths:(NSArray *)indexPaths;
Invalidation Contexts in iOS 8

Fine-grain invalidation

- (void)invalidateItemsAtIndexPaths:(NSArray *)indexPaths;
- (void)invalidateSupplementaryElementsOfKind:(NSString *)elementKind atIndexPaths:(NSArray *)indexPaths;
- (void)invalidateDecorationElementsOfKind:(NSString *)elementKind atIndexPaths:(NSArray *)indexPaths

@property (nonatomic, readonly) NSArray *invalidatedItemIndexPaths;
@property (nonatomic, readonly) NSDictionary *invalidatedSupplementaryIndexPaths;
@property (nonatomic, readonly) NSDictionary *invalidatedDecorationIndexPaths;
Invalidate supplementary view
The collection view will query `layoutAttributesForSupplementaryViewOfKind:atIndexPath:` for this index path only.

```swift
layoutAttributesForSupplementaryViewOfKind:atIndexPath:
```
The collection view will query `layoutAttributes` for this index path only:

```
layoutAttributesForSupplementaryViewOfKind:atIndexPath:
```

And not:

```
layoutAttributesForElementsInRect:
```

Invalidate supplementary view.
Invalidation Contexts in iOS 8
Invalidation Contexts in iOS 8

Inform the collection view of a content-size change
Invalidation Contexts in iOS 8

Inform the collection view of a content-size change

@property (nonatomic) CGSize contentSizeAdjustment;
Invalidation Contexts in iOS 8

Inform the collection view of a content-size change

@property (nonatomic) CGSize contentSizeAdjustment;
@property (nonatomic) CGPoint contentOffsetAdjustment;
Invalidation Contexts in iOS 8
Invalidation Contexts in iOS 8

Layout support for self-sizing cells
Invalidation Contexts in iOS 8

Layout support for self-sizing cells

- (BOOL)shouldInvalidateLayoutForPreferredLayoutAttributes:(UICollectionViewLayoutAttributes *)preferredAttributes withOriginalAttributes:(UICollectionViewLayoutAttributes *)originalAttributes
Invalidation Contexts in iOS 8

Layout support for self-sizing cells

- `(BOOL)shouldInvalidateLayoutForPreferredLayoutAttributes:(UICollectionViewLayoutAttributes *)preferredAttributes withOriginalAttributes:(UICollectionViewLayoutAttributes *)originalAttributes`
- `(UICollectionViewLayoutInvalidationContext *)invalidationContextForPreferredLayoutAttributes:(UICollectionViewLayoutAttributes *)preferredAttributes withOriginalAttributes:(UICollectionViewLayoutAttributes *)originalAttributes`
Self-sizing Cells in a Custom Layout
Self-sizing Cells in a Custom Layout
Self-sizing Cells in a Custom Layout

Generate a first approximation by returning layout attributes in `layoutAttributesForElementsInRect`: 
Self-sizing Cells in a Custom Layout

The collection view will dequeue and query cells
Cells can self-size with auto layout, sizeThatFits: or preferredLayoutAttributesFittingAttributes:
Self-sizing Cells in a Custom Layout

Other layout attributes (position) can then be invalidated:

shouldInvalidateLayoutForPreferredLayoutAttributes:withOriginalAttributes:
invalidationContextForPreferredLayoutAttributes:withOriginalAttributes:
Self-sizing Cells in a Custom Layout

The collection view layout content size must change
contentSizeAdjustment
Self-sizing Cells in a Custom Layout

The collection view layout offset must change
contentOffsetAdjustment
Self-sizing Cells in a Custom Layout

The actual implementation is layout specific
Final Thoughts
Final Thoughts
Final Thoughts

Every app should adopt Dynamic Type
Final Thoughts

Every app should adopt Dynamic Type
Self-sizing cells will help
Final Thoughts

Every app should adopt Dynamic Type
Self-sizing cells will help
Use invalidation contexts for high-performance collection view layouts
## Related Sessions

<table>
<thead>
<tr>
<th>Event</th>
<th>Speaker</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced User Interfaces with Collection Views</td>
<td>Marina</td>
<td>Thursday 3:15PM</td>
</tr>
<tr>
<td>Labs</td>
<td>Frameworks Lab A Thursday 11:30AM</td>
<td>Frameworks Lab A Thursday 2:00PM</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-----------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>- Table View and Collection View Lab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Cocoa Touch Lab</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
More Information

Jake Behrens
iOS Frameworks Evangelist
behrens@apple.com

Table View and Collection View Reference Guides

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