Auto Layout Techniques in Interface Builder

Session 412

Jonathon Mah, Interface Builder Engineer
Jason Yao, Interface Builder Engineer
Changing layout at runtime
Tracking touch
Dynamic type
Safe areas
Proportional positioning
Stack view adaptive layout
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime

![Image of a mobile interface with editable maximum distance setting]
Changing Layout at Runtime
Changing Layout at Runtime
let zeroHeightConstraint = wrappingView.heightAnchor.constraint(equalToConstant: 0)
let zeroHeightConstraint = wrappingView.heightAnchor.constraint(equalToConstant: 0)

//...
Changing Layout at Runtime

```swift
let zeroHeightConstraint = wrappingView.heightAnchor.constraint(equalToConstant: 0)

//...
zeroHeightConstraint.isActive = true
```
Changing Layout at Runtime

let zeroHeightConstraint = wrappingView.heightAnchor.constraint(equalToConstant: 0)
//...
zeroHeightConstraint.isActive = true
edgeConstraint.isActive = false
@IBOutlet var wrappingView: UIView!
@IBOutlet var edgeConstraint: NSLayoutConstraint!
var zeroHeightConstraint: NSLayoutConstraint!
@IBOutlet var wrappingView: UIView!
@IBOutlet var edgeConstraint: NSLayoutConstraint!
var zeroHeightConstraint: NSLayoutConstraint!
@IBOutlet var wrappingView: UIView!
@IBOutlet var edgeConstraint: NSLayoutConstraint!
var zeroHeightConstraint: NSLayoutConstraint!

@IBAction func toggleDistanceControls(_ sender: Any) {
    if zeroHeightConstraint == nil {
        zeroHeightConstraint = wrappingView.heightAnchor
            .constraint(equalToConstant: 0)
    }
}
@IBOutlet var wrappingView: UIView!
@IBOutlet var edgeConstraint: NSLayoutConstraint!
var zeroHeightConstraint: NSLayoutConstraint!

@IBAction func toggleDistanceControls(_ sender: Any) {
    if zeroHeightConstraint == nil {
        zeroHeightConstraint = wrappingView.heightAnchor
            .constraint(equalToConstant: 0)
    }

    let shouldShow = !edgeConstraint.isActive
zeroHeightConstraint = wrappingView.heightAnchor
    .constraint(equalToConstant: 0)
}

let shouldShow = !edgeConstraint.isActive
```swift
zeroHeightConstraint = wrappingView.heightAnchor
    .constraint(equalToConstant: 0)
}

let shouldShow = !edgeConstraint.isActive

// Deactivate constraint first to avoid constraint conflict message
if shouldShow {
    zeroHeightConstraint.isActive = false
    edgeConstraint.isActive = true
} else {
    edgeConstraint.isActive = false
    zeroHeightConstraint.isActive = true
}
```
@IBAction func toggleDistanceControls(_ sender: Any) {
    if zeroHeightConstraint == nil {
        zeroHeightConstraint = wrappingView.heightAnchor.constraint(equalToConstant: 0)
    }
    let shouldShow = !edgeConstraint.isActive
    // Deactivate constraint first to avoid constraint conflict message
    if shouldShow {
        zeroHeightConstraint.isActive = false
        edgeConstraint.isActive = true
    } else {
        edgeConstraint.isActive = false
        zeroHeightConstraint.isActive = true
    }
    UIView.animate(withDuration: 0.25) {
        self.view.layoutIfNeeded()
    }
}

let shouldShow = !edgeConstraint.isActive

// Deactivate constraint first to avoid constraint conflict message
if shouldShow {
    zeroHeightConstraint.isActive = false
    edgeConstraint.isActive = true
} else {
    edgeConstraint.isActive = false
    zeroHeightConstraint.isActive = true
}
Changing Layout at Runtime

[Images of user interface showing layout changes at runtime]
Changing Layout at Runtime
Changing Layout at Runtime
Changing Layout at Runtime

![Example of layout change](image-url)
Tracking Touch
Medjool Dates
Entire Foods, 2 mi away
Tracking Touch
Tracking Touch

Layout engine owns frame
Tracking Touch

Layout engine owns frame

• Value derived from constraints
Tracking Touch

Layout engine owns `frame`

- Value derived from constraints

`transform` property offsets from frame
Tracking Touch

Layout engine owns `frame`

- Value derived from constraints

`transform` property offsets from frame

CGAffineTransform = translation + rotation + scale
Tracking Touch

View position is a result of multiple properties
Tracking Touch

View position is a result of multiple properties

- frame (derived from constraints)
View position is a result of multiple properties

- `frame` (derived from constraints)
- `transform`
Tracking Touch

View position is a result of multiple properties

- **frame** (derived from constraints)
- **transform**

Transform is great for temporary changes
Tracking Touch

View position is a result of multiple properties

- **frame** (derived from constraints)
- **transform**

Transform is great for temporary changes

- Reset to `CGAffineTransform.identity` when done
Supporting Dynamic Type
Dynamic Type
Dynamic Type
Dynamic Type
Dynamic Type
Dynamic Type
Demo
Dynamic Type

- Text: Plain
- Color: Default
- Font: Headline
- Dynamic Type: Automatically Adjusts Font
- Alignment
- Lines: 1
- Behavior: Enabled
- Baseline: Align Baselines
- Line Break: Truncate Tail
Dynamic Type
Dynamic Type

- **Label**
  - Text: Plain
  - Color: Default
  - Font: Headline

- **Accessibility Inspector**
  - Common accessibility options to test in an App:
    - Invert colors
    - Reduce transparency
    - Reduce motion
  - Font size: A, A, A
Dynamic Type

Building Apps with Dynamic Type

Executive Ballroom
Friday 1:50PM
Dynamic Type

Larger Accessibility Sizes

Apps that support Dynamic Type will adjust to your preferred reading size below.
Safe Areas
Safe Area Layout Guide
Finding Dates

Friday 9:00am

75°F
Finding Dates

Friday 9:00am

☀ 75°F
Safe Area Layout Guide

Property on UIView
Safe Area Layout Guide

Property on UIView

Title safe, unobscured content
Safe Area Layout Guide

Property on UIView

Title safe, unobscured content

iOS storyboards
Safe Area Layout Guide

Property on UIView

Title safe, unobscured content

iOS storyboards

• Constraints automatically upgrade
Safe Area Layout Guide

Property on UIView

Title safe, unobscured content

iOS storyboards
  • Constraints automatically upgrade
  • Backwards deployable
Proportional Positioning
Dates
Dates
Dates

70%
Date B  Sweetness
Demo
Proportional Positioning
Proportional Positioning

Use spacer views when needed in Interface Builder
Proportional Positioning

Use spacer views when needed in Interface Builder
• Mark as hidden
Proportional Positioning

Use spacer views when needed in Interface Builder
• Mark as hidden

Use UILayoutGuide programmatically
Stack View
Adaptive layout
Top Dates
To date or not to date, that is the question.

Lorem ipsum dolor sit amet, ligula suspendisse nulla pretium, rhoncus tempor fermentum, enim integer ad vestibulum volutpat. Nisl rhoncus turpis est, vel elit, congue wisi enim nunc ultricies sit, magna tincidunt. Maecenas aliquam maecenas ligula nostra, accumsan taciti.
Demo
Stack View Adaptive Layout
Stack View Adaptive Layout

Use distribution, alignment, and spacing
Stack View Adaptive Layout

Use distribution, alignment, and spacing

Add some constraints
Stack View Adaptive Layout

Use distribution, alignment, and spacing

Add some constraints

Hidden property by size class in Xcode 9
Stack View Adaptive Layout

Use distribution, alignment, and spacing

Add some constraints

Hidden property by size class in Xcode 9

• Works great with Stack View
Stack View Adaptive Layout

Use distribution, alignment, and spacing

Add some constraints

Hidden property by size class in Xcode 9

- Works great with Stack View
- Backwards deployable
Summary

Dynamic content
• Layout changes with outlets
• Animate using transform

Layout
• Safe areas and spacer views
• Adaptive layout with stack views
More Information

## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building Apps with Dynamic Type</td>
<td>Executive Ballroom</td>
<td>Friday 1:50PM</td>
</tr>
<tr>
<td>Updating Your App for iOS 11</td>
<td></td>
<td>WWDC 2017</td>
</tr>
<tr>
<td>What’s New in tvOS</td>
<td></td>
<td>WWDC 2017</td>
</tr>
<tr>
<td>What’s New in Auto Layout</td>
<td></td>
<td>WWDC 2016</td>
</tr>
<tr>
<td>Mysteries of Auto Layout, Part 1</td>
<td></td>
<td>WWDC 2015</td>
</tr>
<tr>
<td>Mysteries of Auto Layout, Part 2</td>
<td></td>
<td>WWDC 2015</td>
</tr>
<tr>
<td>Advanced Interface Builder and Auto Layout Lab</td>
<td>Technology Lab K</td>
<td>Fri 12:00–1:50PM</td>
</tr>
</tbody>
</table>