

iOS
DeCal

lecture 2

AutoLayout

cs198-001 : fall 2017

today's lecture

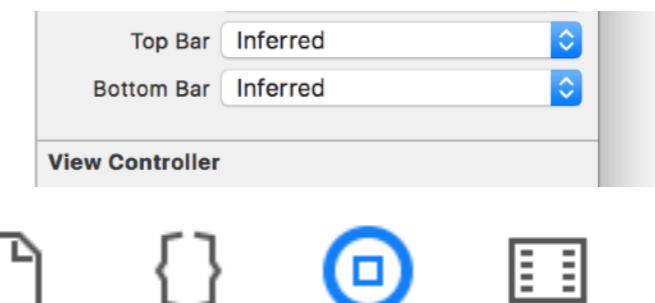
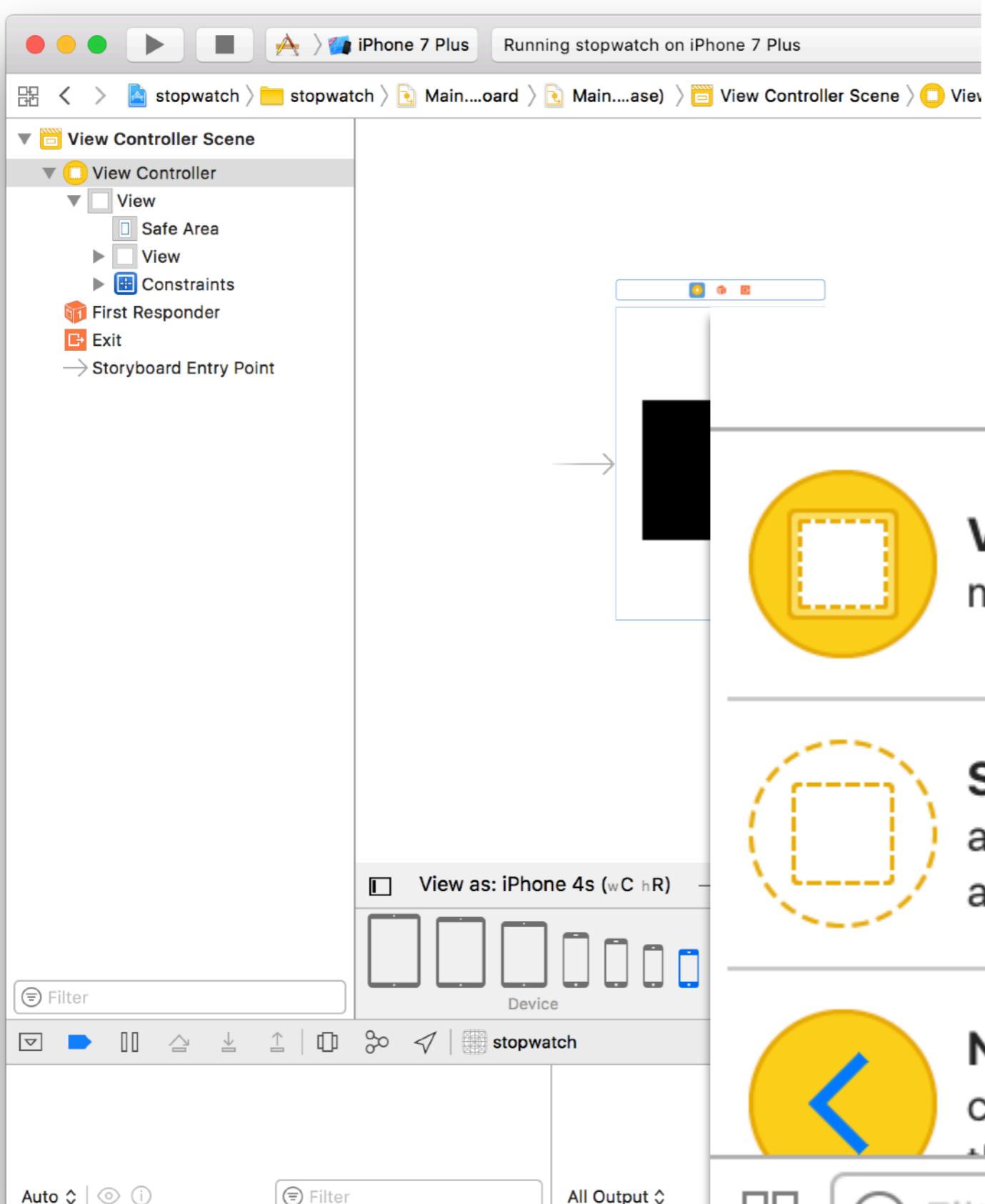
- announcements
- UI toolkit / Object library
- laying out views using Autolayout + Storyboard
- ~special~ check in

announcements

- hw2 (hangman) released????
- due next Monday after lecture
- if you didn't check off for lab last week, you can check off after lecture or at the beginning of next lab

iOS storyboard objects

object library



View Controller - A controller that manages a view.



Storyboard Reference - Provides a placeholder for a view controller in an external storyboard.



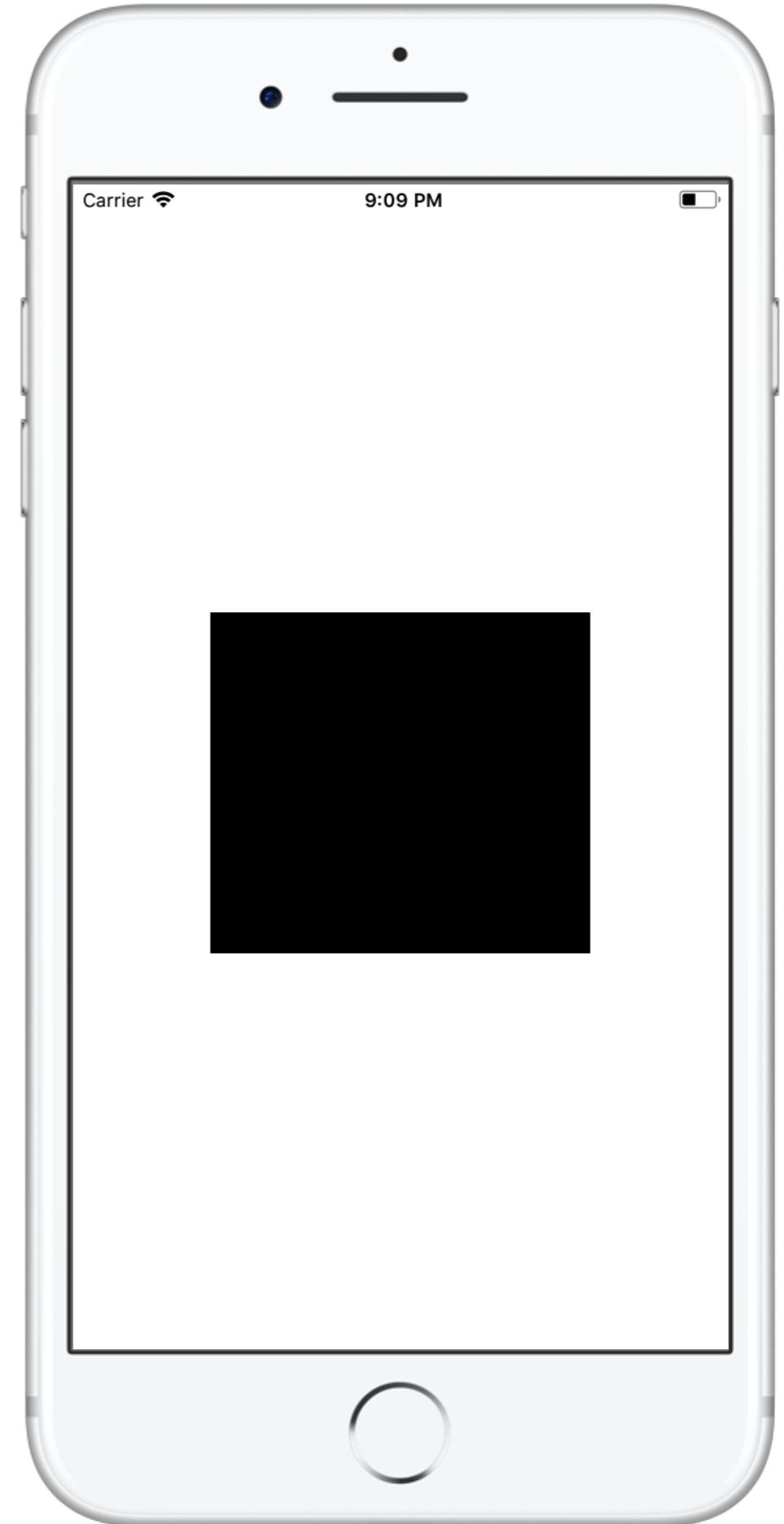
Navigation Controller - A controller that manages navigation

views



View - Represents a rectangular region in which it draws and receives events.

- just a box
- all UI elements subclass `UIView`
- can respond to touches / other gestures

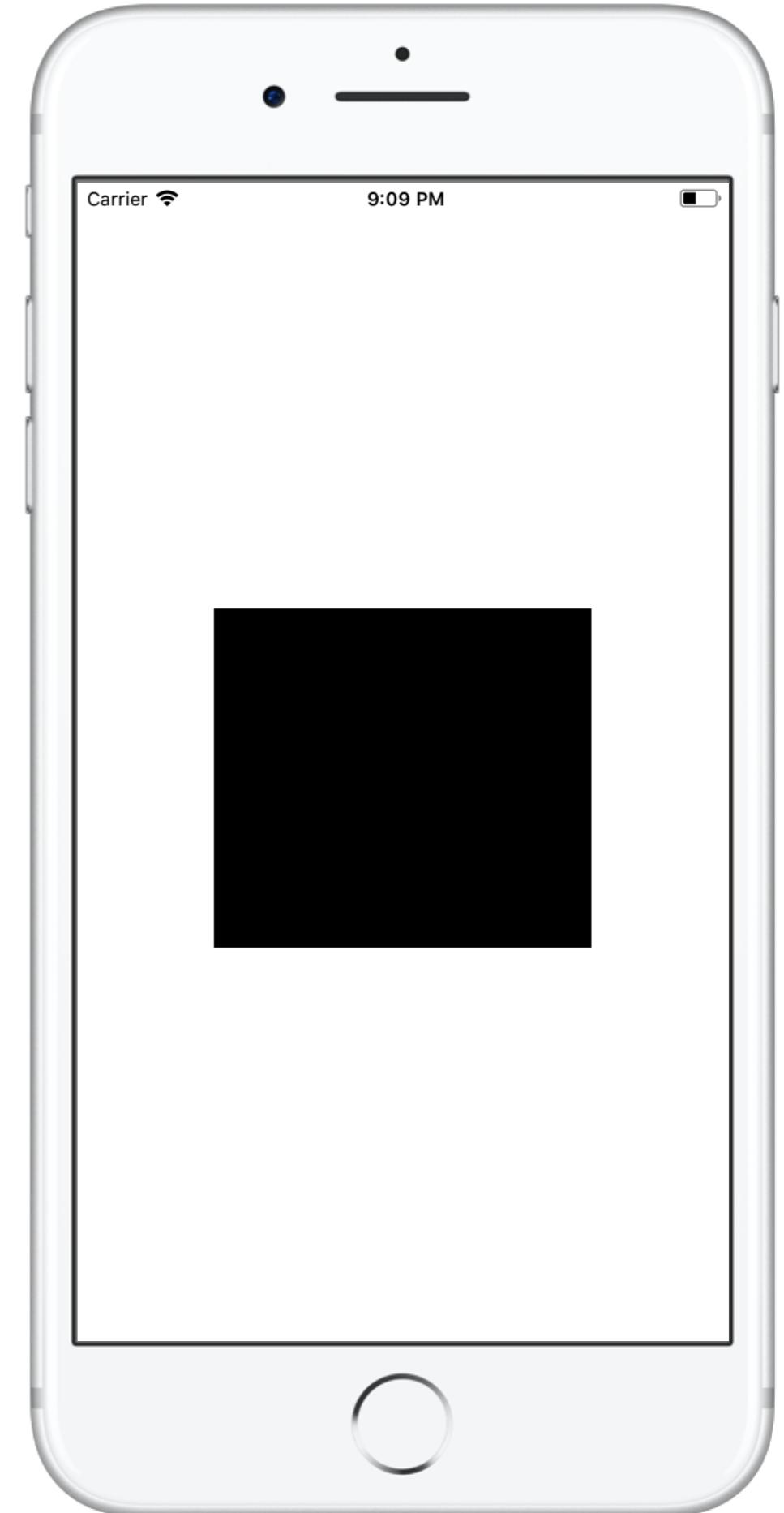


views



View - Represents a rectangular region in which it draws and receives events.

- just a box
- all UI elements subclass **UIView**
- can respond to touches / other gestures
- *how many views are in this image?*

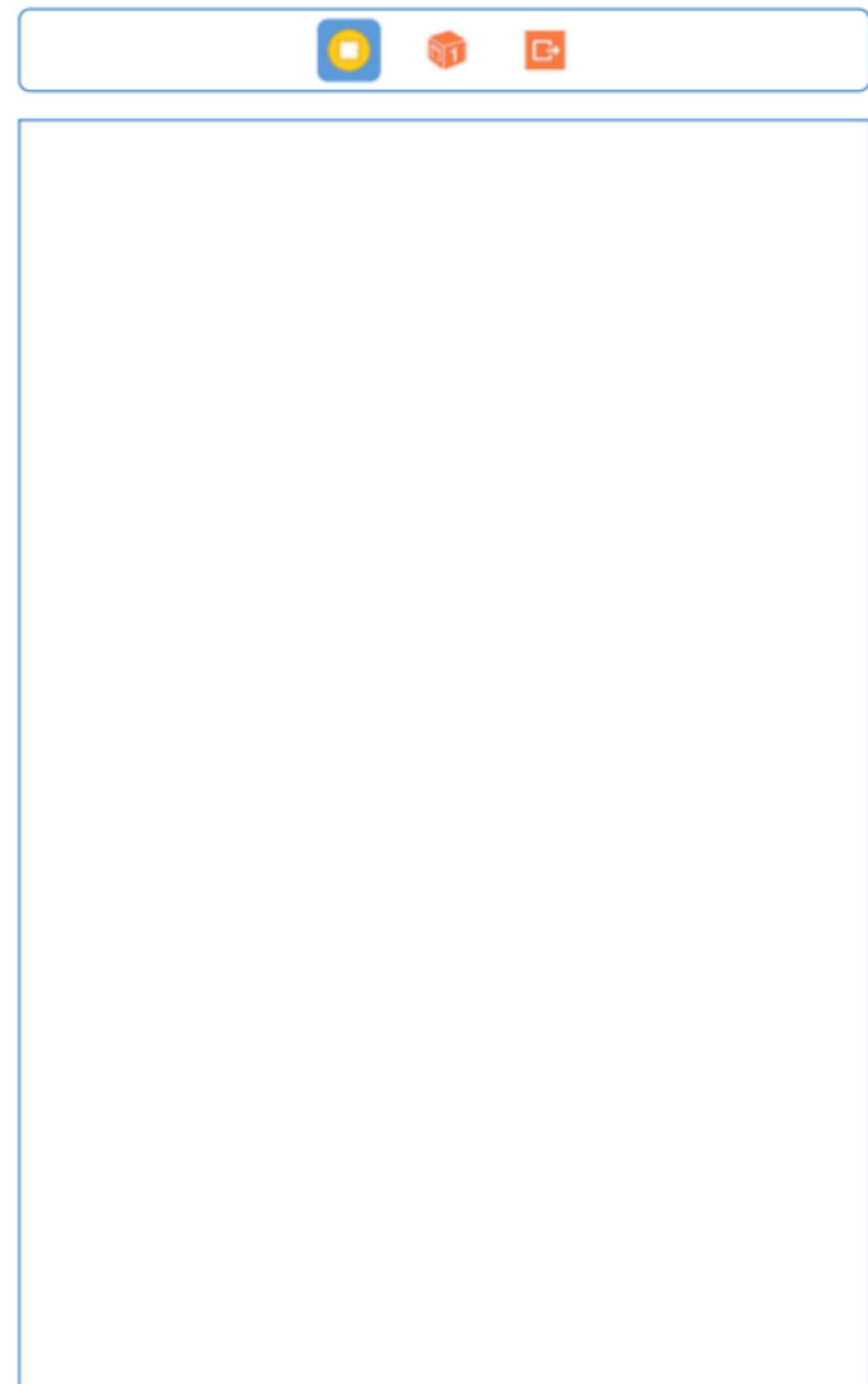


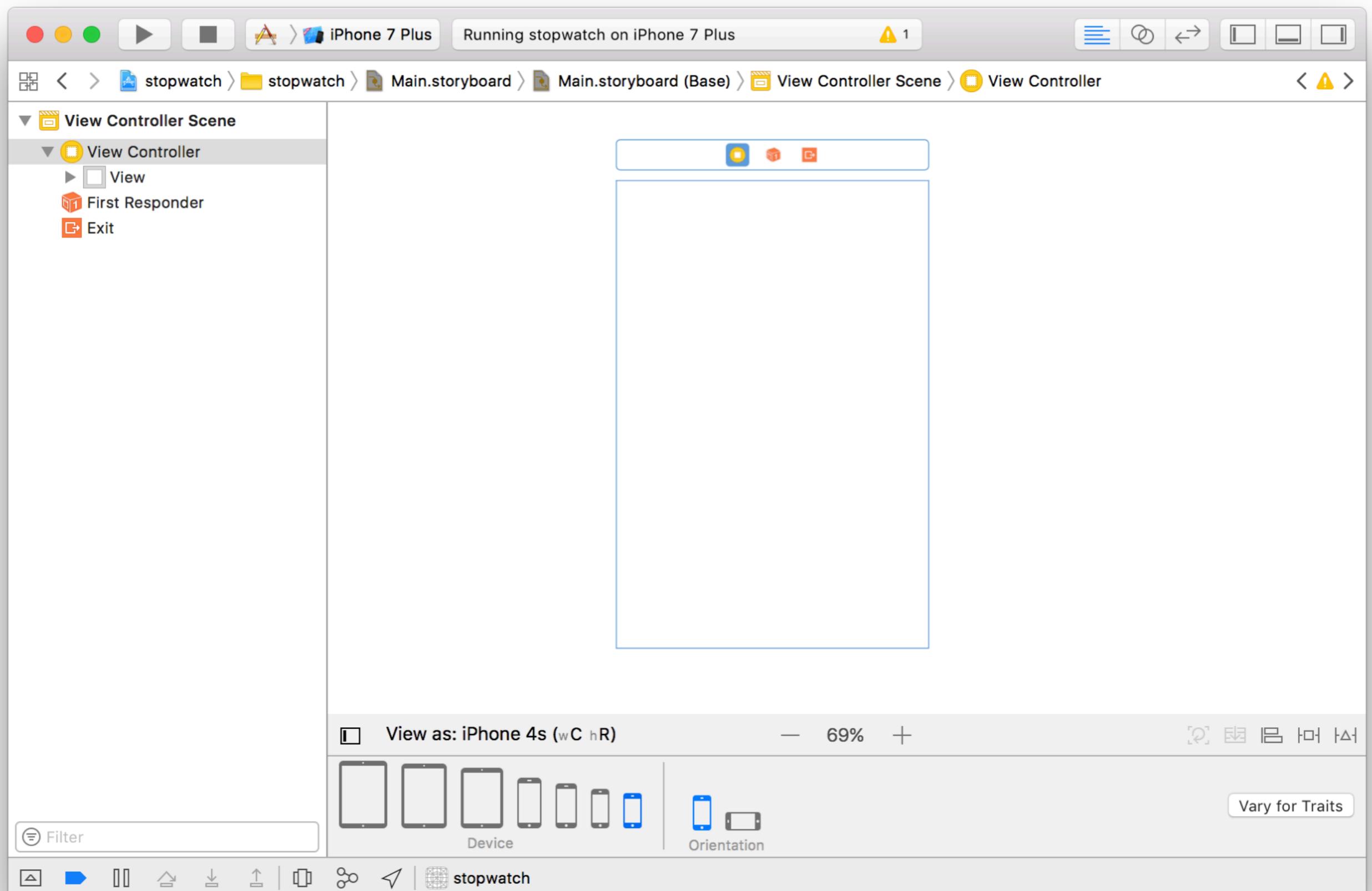
view controllers



View Controller - A controller that manages a view.

- not really a UI object
- adds a new “screen” to your storyboard
- comes with a blank view
- view controller ≠ view



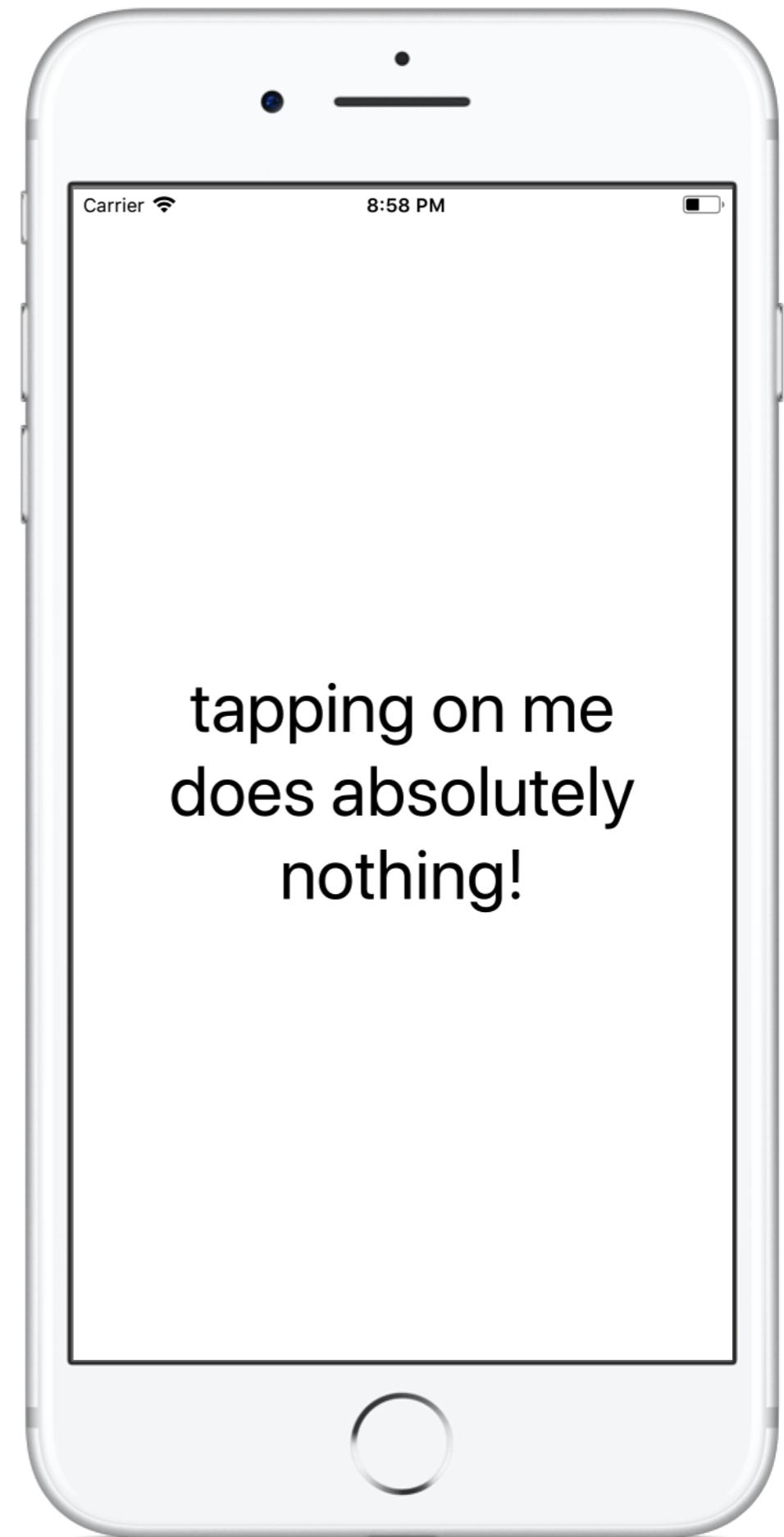


view controller vs. view

labels

Label **Label** - A variably sized amount of static text.

- used to display text
- not editable by user



text fields

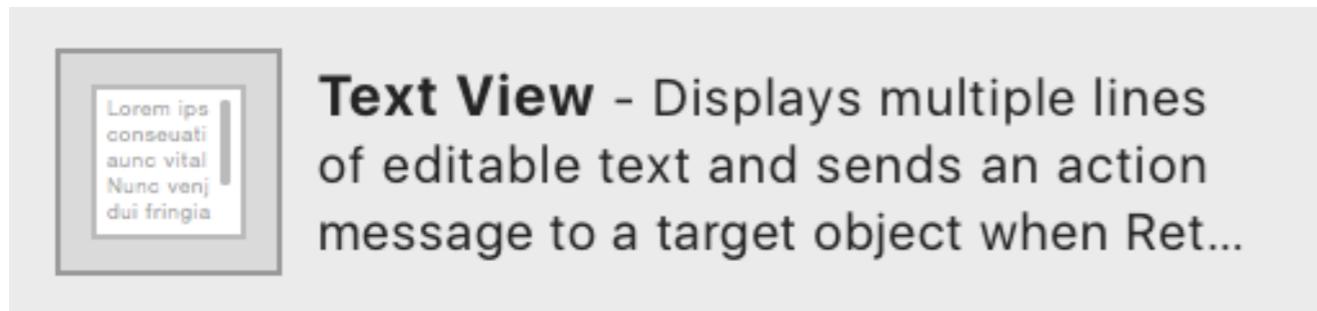
Text

Text Field - Displays editable text and sends an action message to a target object when Return is tapped.

- **editable text**
- **limited to one line**
- **use to get user input**



text views



Text View - Displays multiple lines of editable text and sends an action message to a target object when Ret...

- **editable text**
- **multi-line**
- **use to get lengthy user text input**
- **scrollable**



scroll views

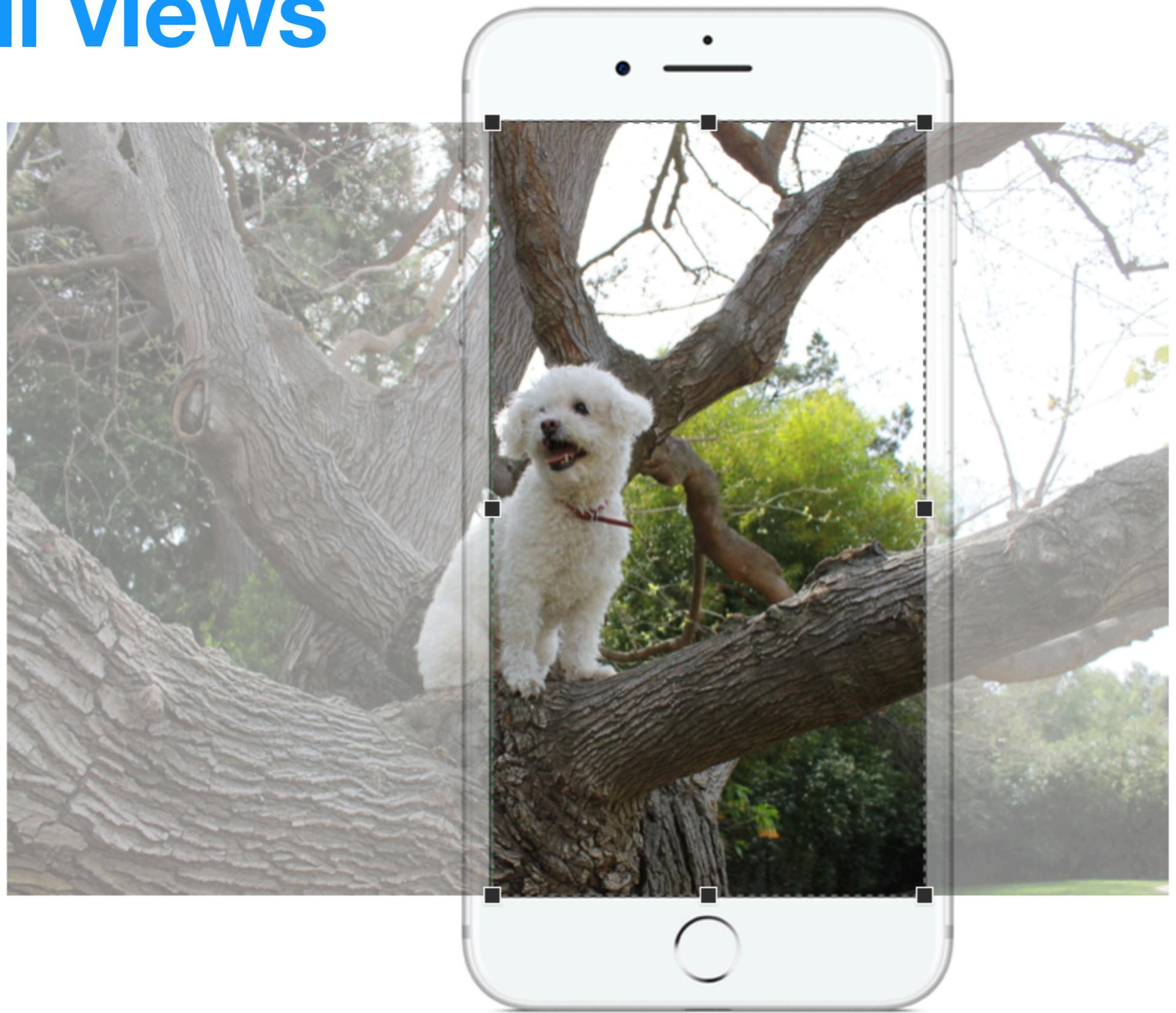


ScrollView - Provides a mechanism to display content that is larger than the size of the application's window.

- accepts user swipes to scroll embedded content
- typically, you'll use a subclass of `UIScrollView`
- examples: `UITextView`, `UICollectionView`, `UITableView`



scroll views



buttons

Button - Intercepts touch events and sends an action message to a target object when it's tapped.

- use to detect touch events
- useless unless you connect to an IBAction or register a selector

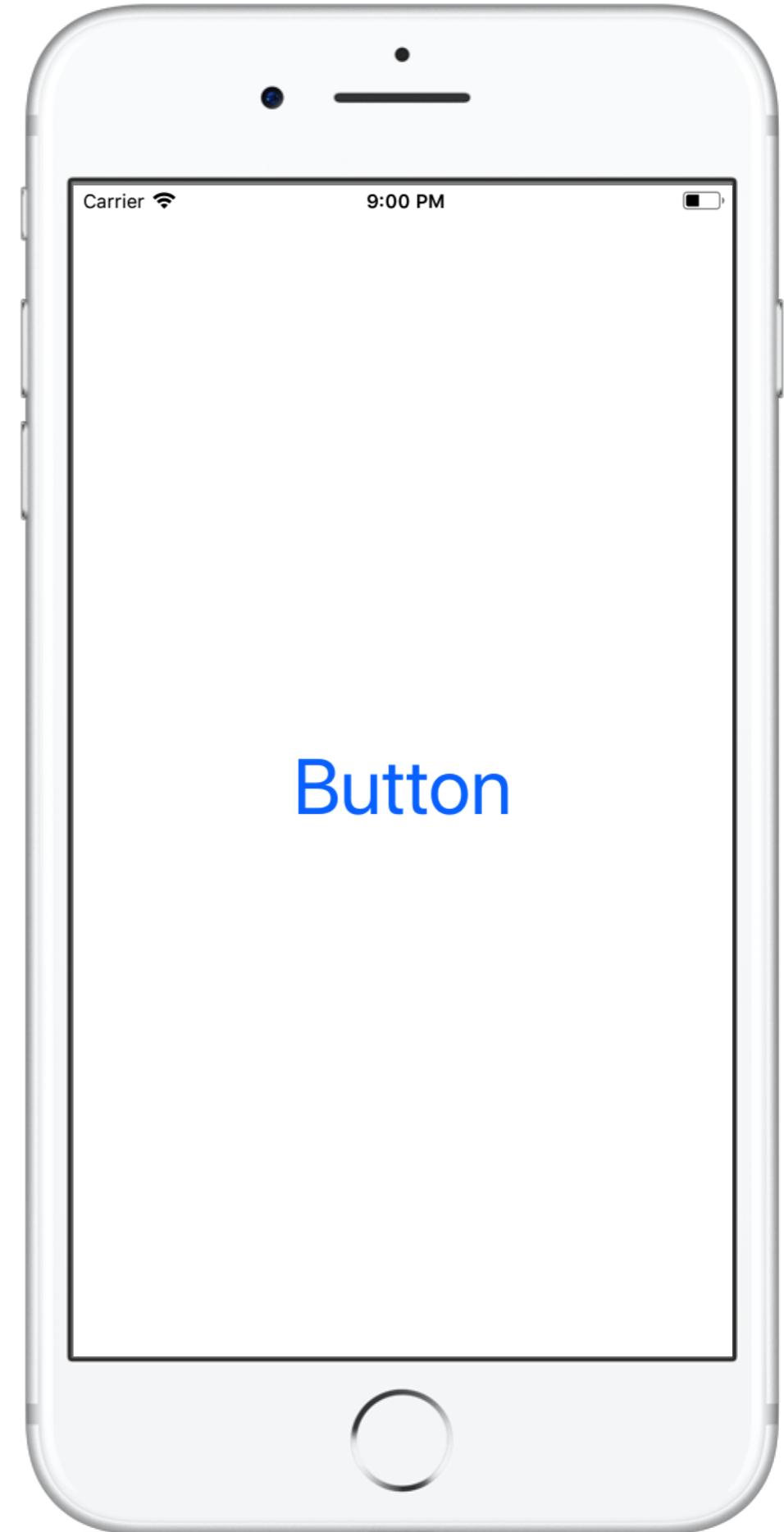
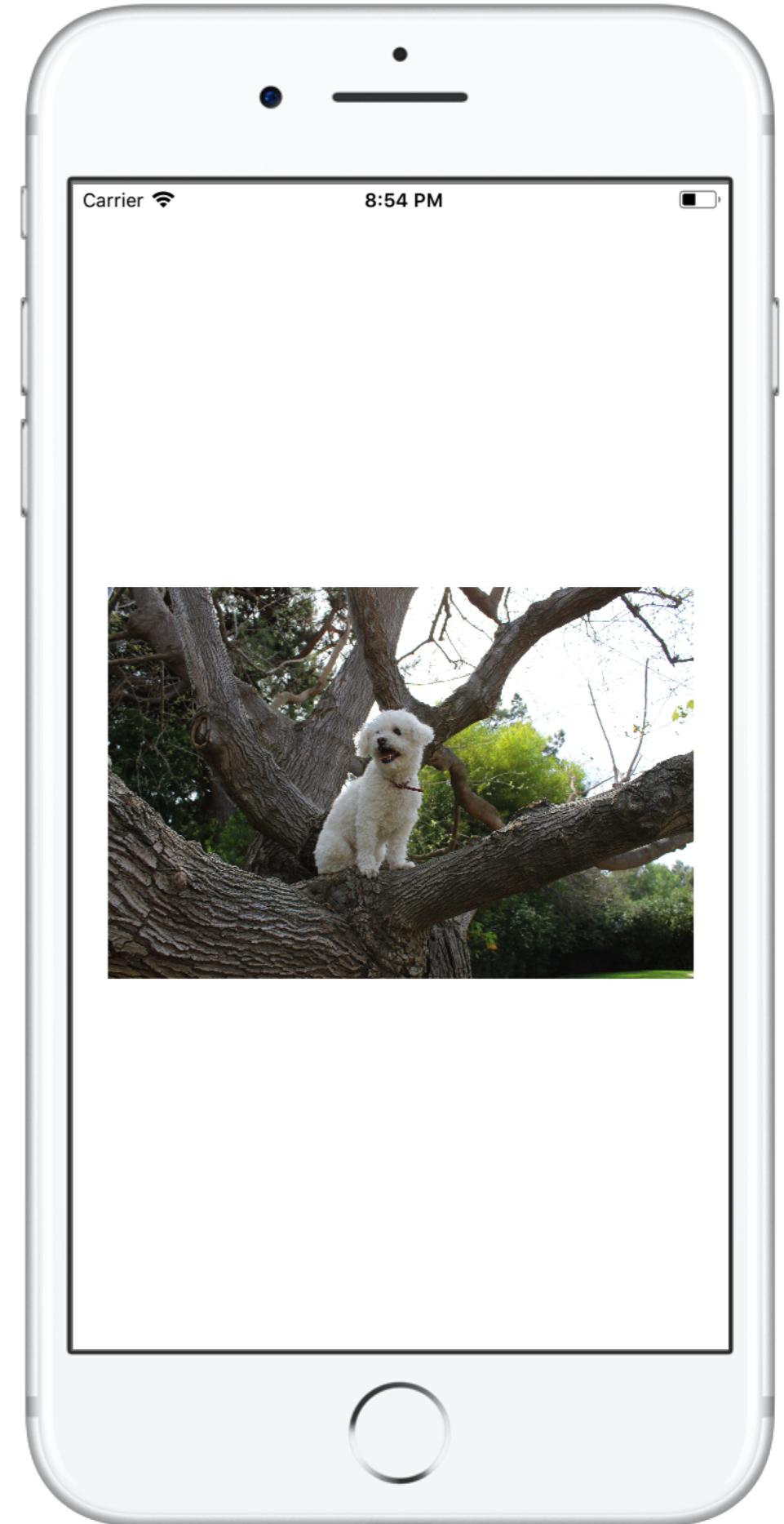


image views



Image View - Displays a single image, or an animation described by an array of images.

- used to display images
- image view ≠ image
- think of it like a picture frame



Storyboard references



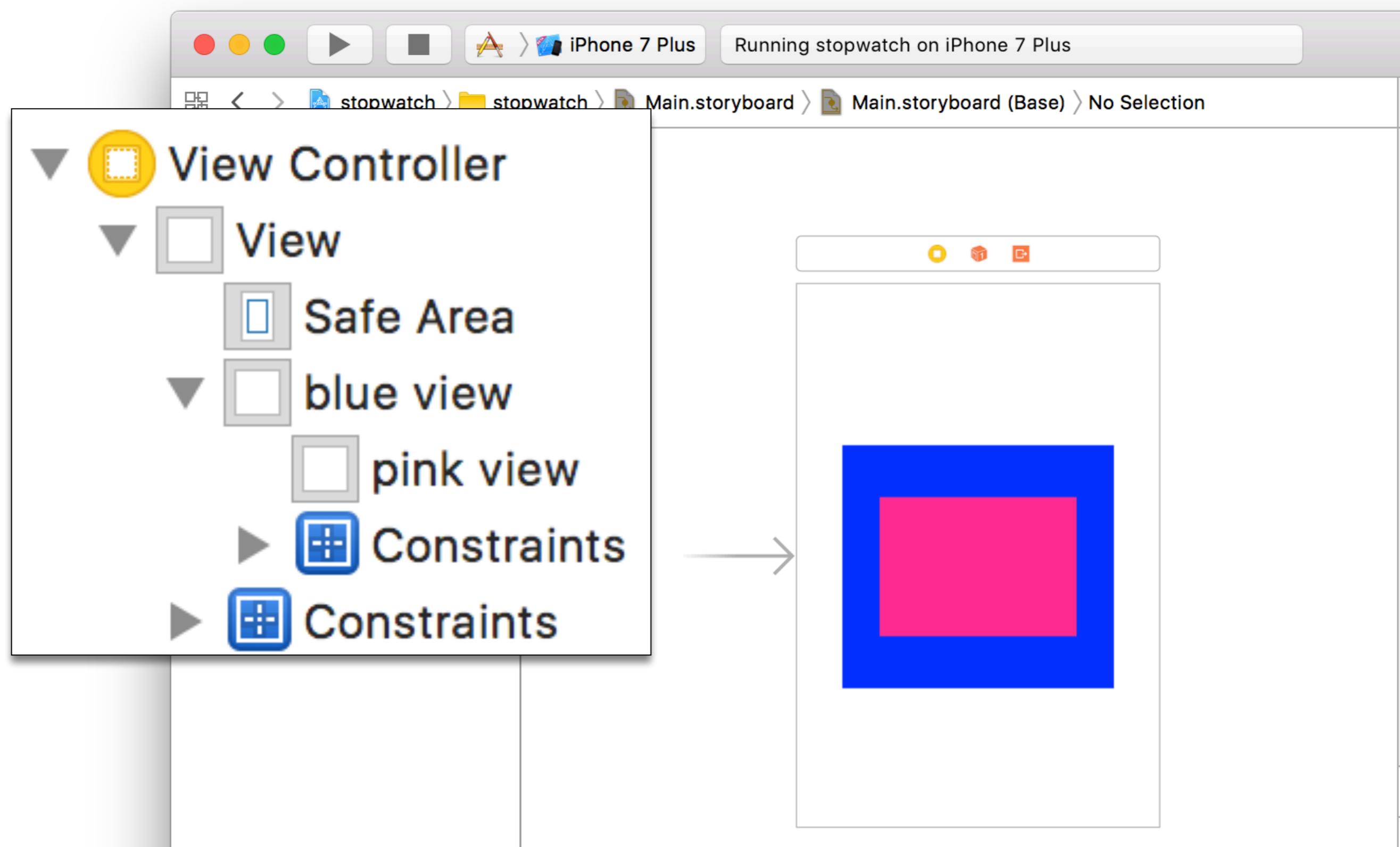
Storyboard Reference - Provides a placeholder for a view controller in an external storyboard.

- more than one storyboard file? no problem!
- *these will make more sense when we cover segues (next lecture)*



Storyboard Reference

superview / subview



stack views

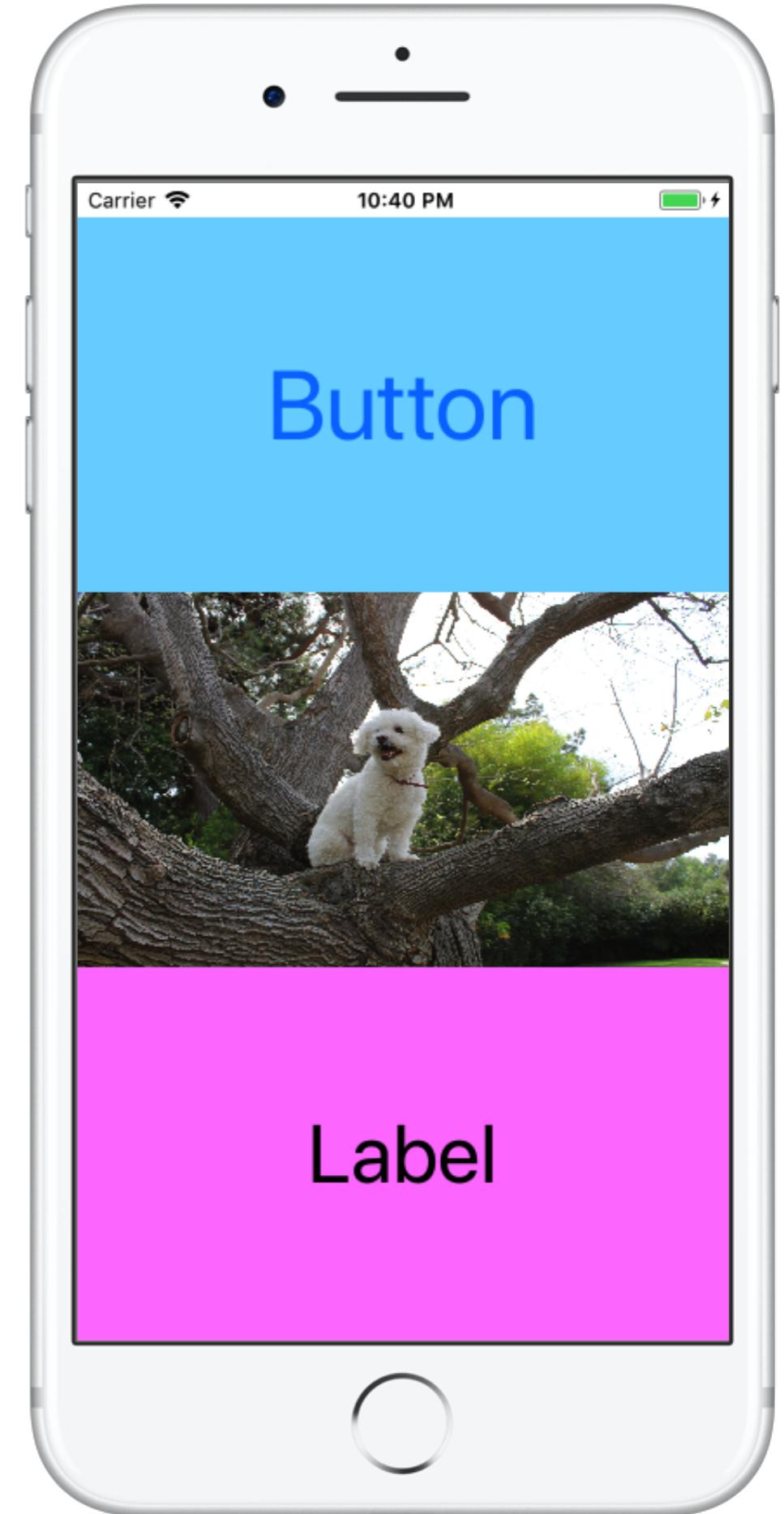


Horizontal Stack View - Arranges views linearly.



Vertical Stack View - Arranges views linearly.

- used to arrange other subviews (static)
- AutoLayout's best friend



stack views



Horizontal Stack View - Arranges views linearly.



Vertical Stack View - Arranges views linearly.

- used to arrange other subviews (static)
- AutoLayout's best friend
- nested stackviews



table views

(future lecture)



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

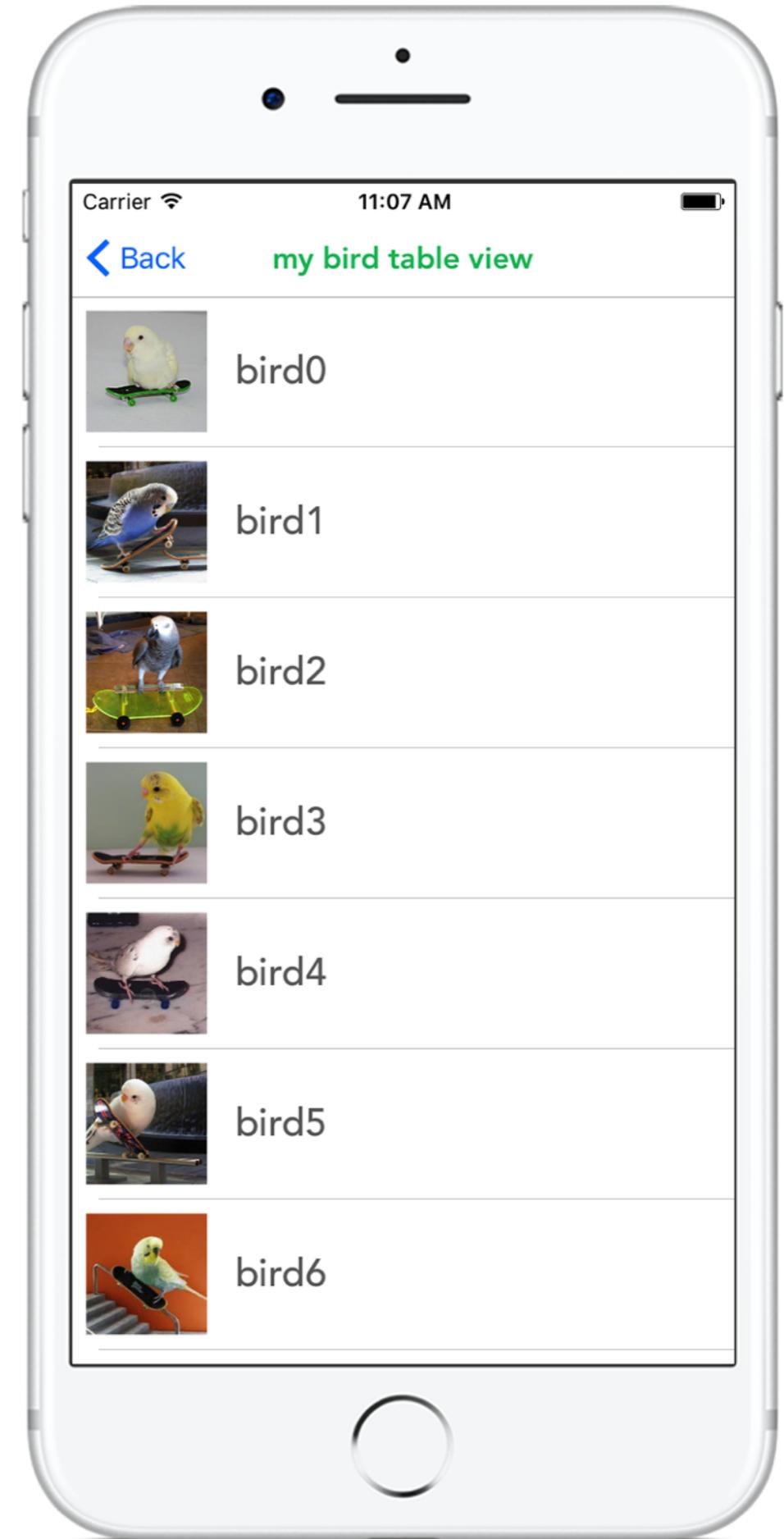


Table View - Displays data in a list of plain, sectioned, or grouped rows.



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

- displays cells of data
- scrollable, static or dynamic, view reuse



collection views

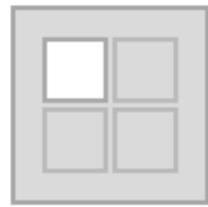
(future lecture)



Collection View Controller - A controller that manages a collection view.

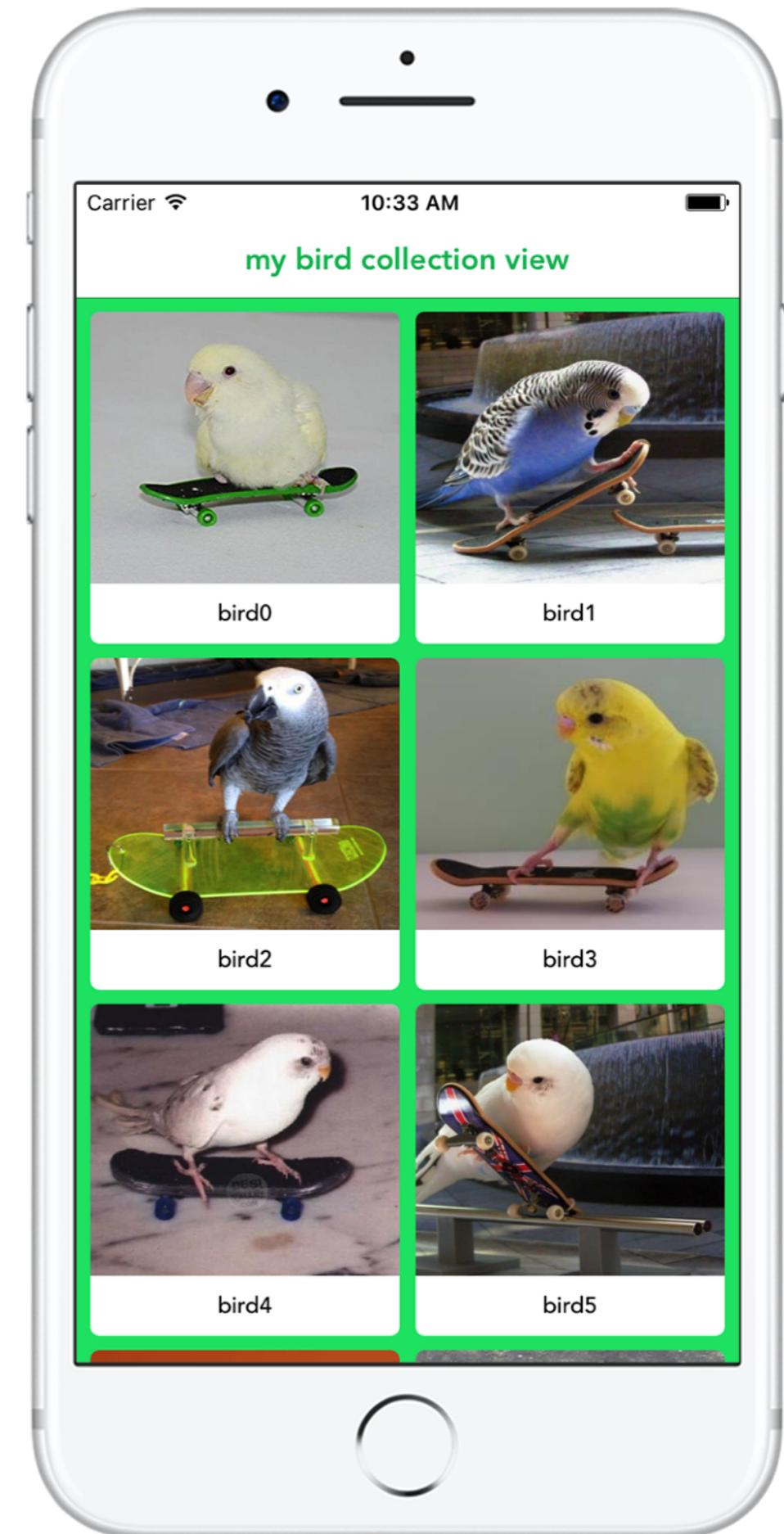


Collection View - Displays data in a collection of cells.



Collection View Cell - Defines the attributes and behavior of cells in a collection view.

- like table views, but displays data in a grid



**laying out your user
interface**

creating your user interface

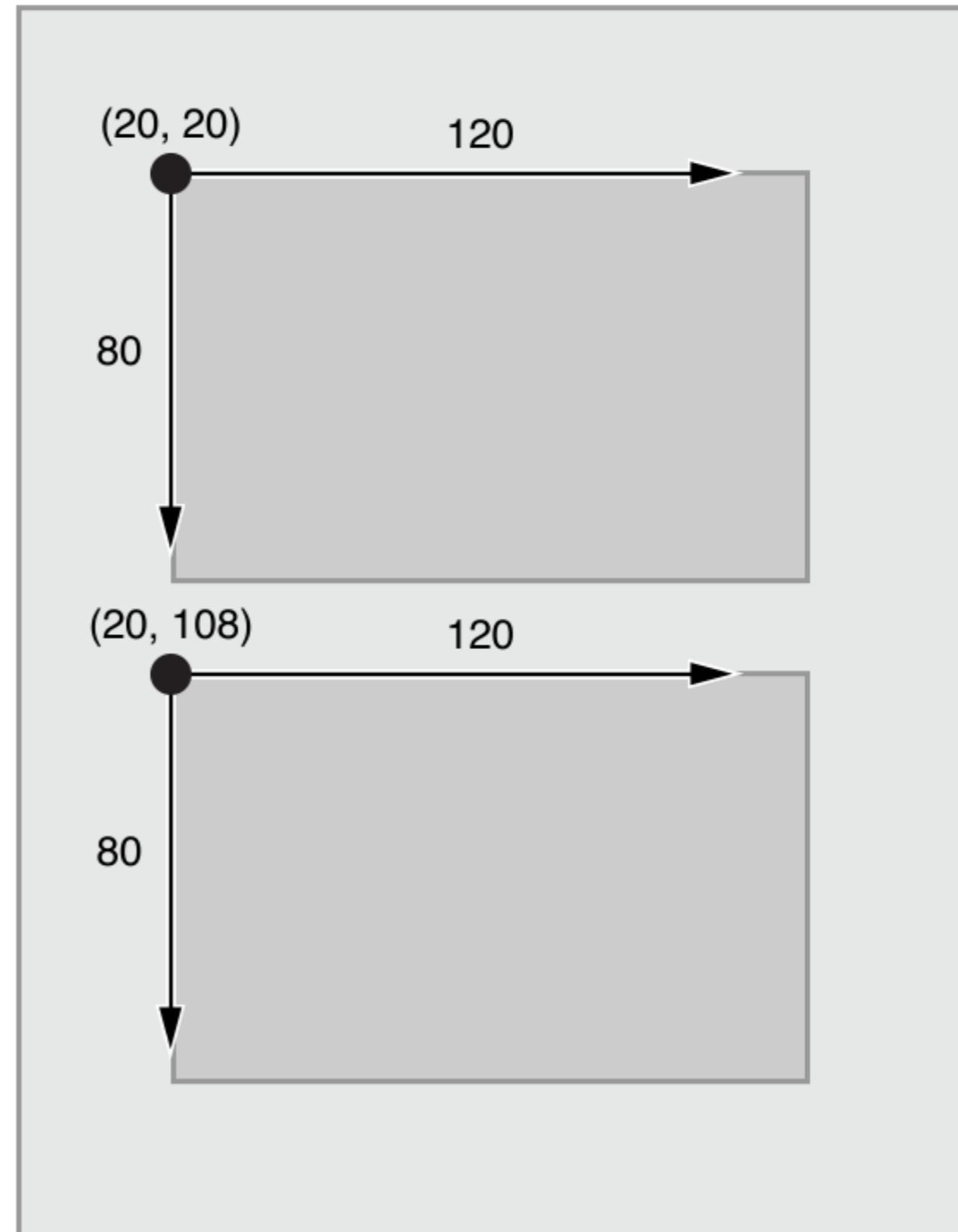
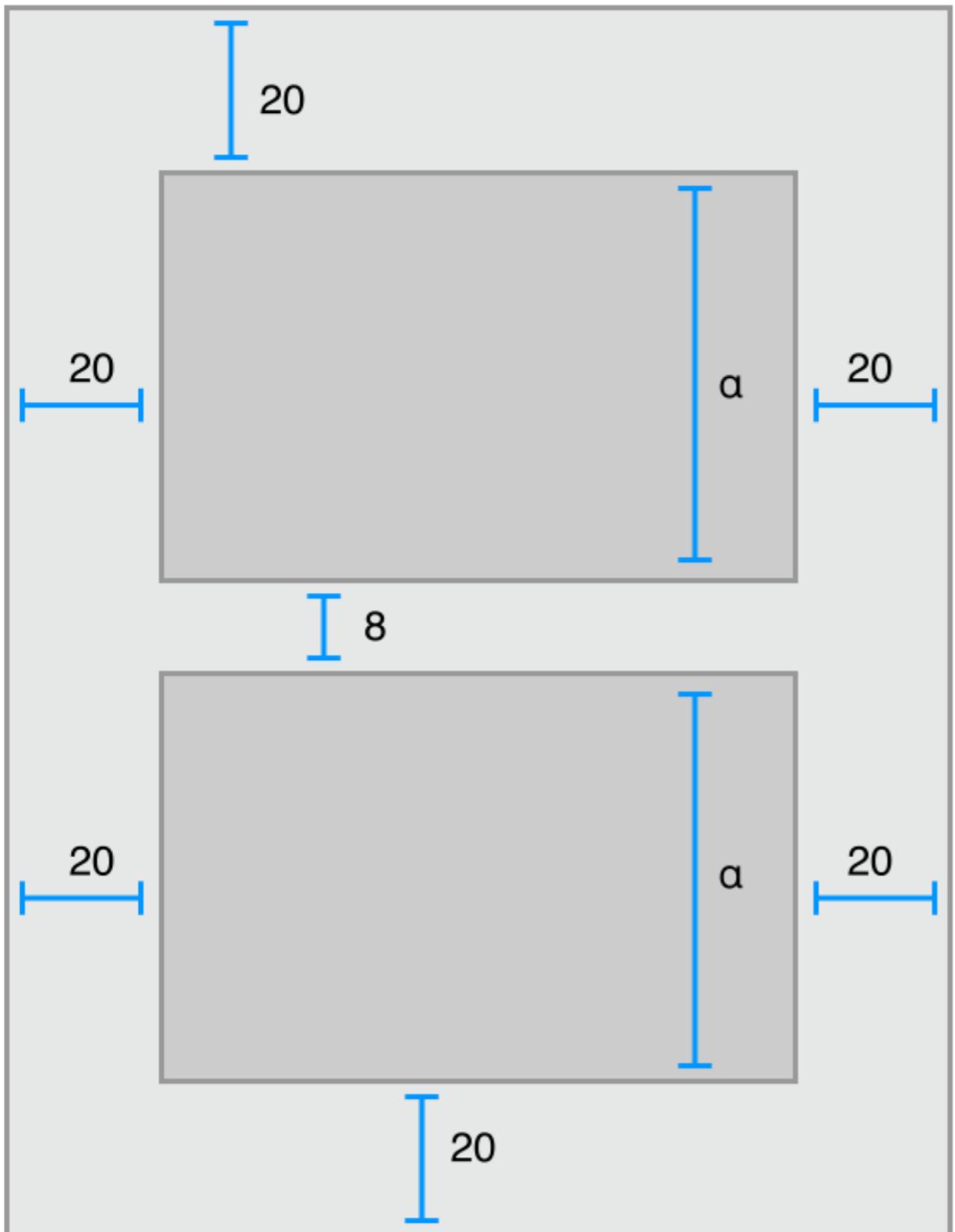
what are our options?

frame-based layout

- define the origin / width / height (frame) for each view
- recalculate frame on layout changes
- implement programmatically

AutoLayout

- define sizing / layout through relational constraints
- implement either in Storyboard or programmatically
- what we'll go over in this class



why autolayout (with storyboard)

- visual (readability pro)
- positioning / frame code is lengthy
- less steep learning curve
- recommended + constantly being updated with new Xcode releases

Auto Layout

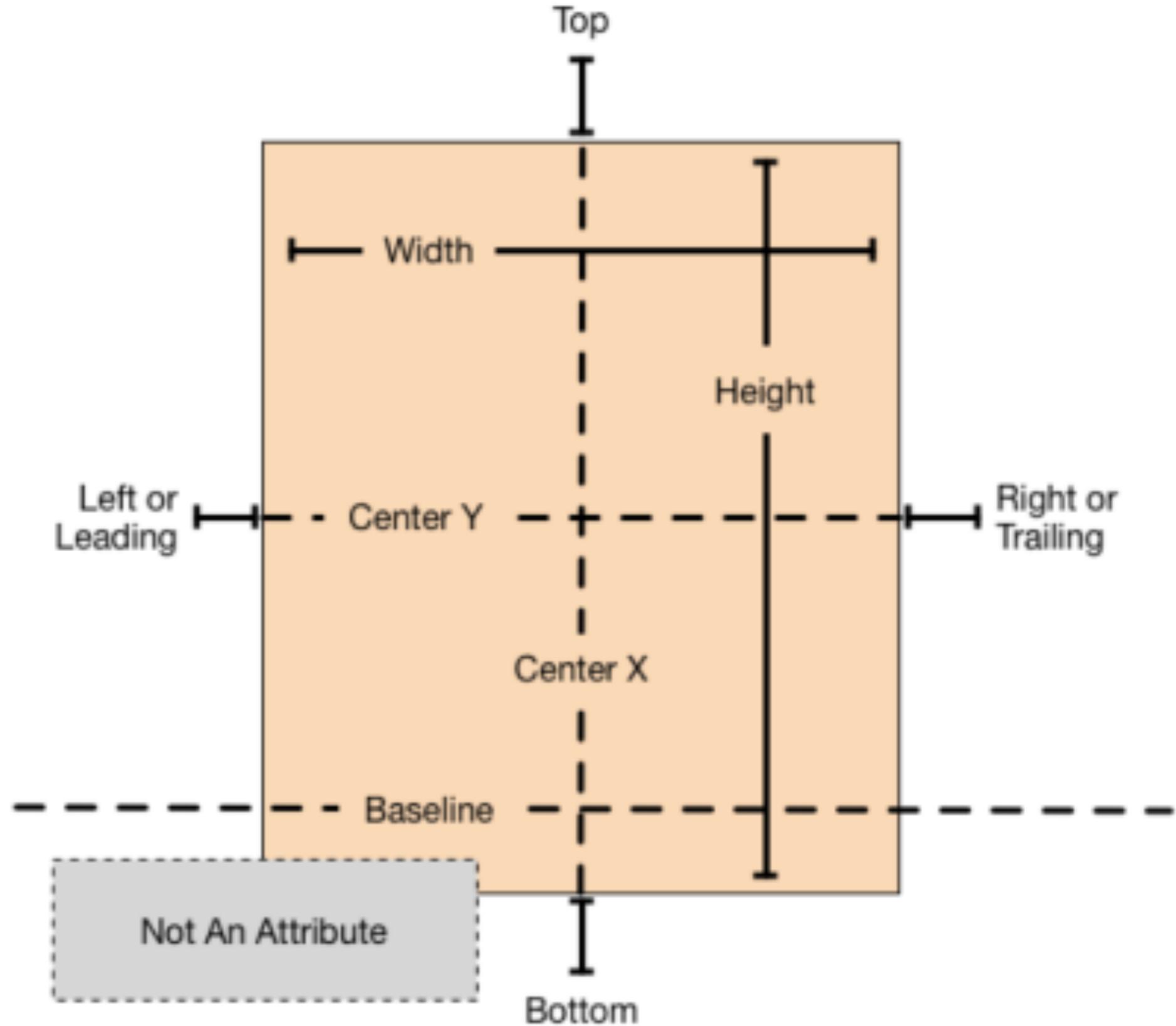
what is Auto Layout?

- constraint based, descriptive layout system
- create adaptive interfaces that responds to changes in screen size and device orientation



layout anchors

Use these properties to create relationships between views



list of constraint types

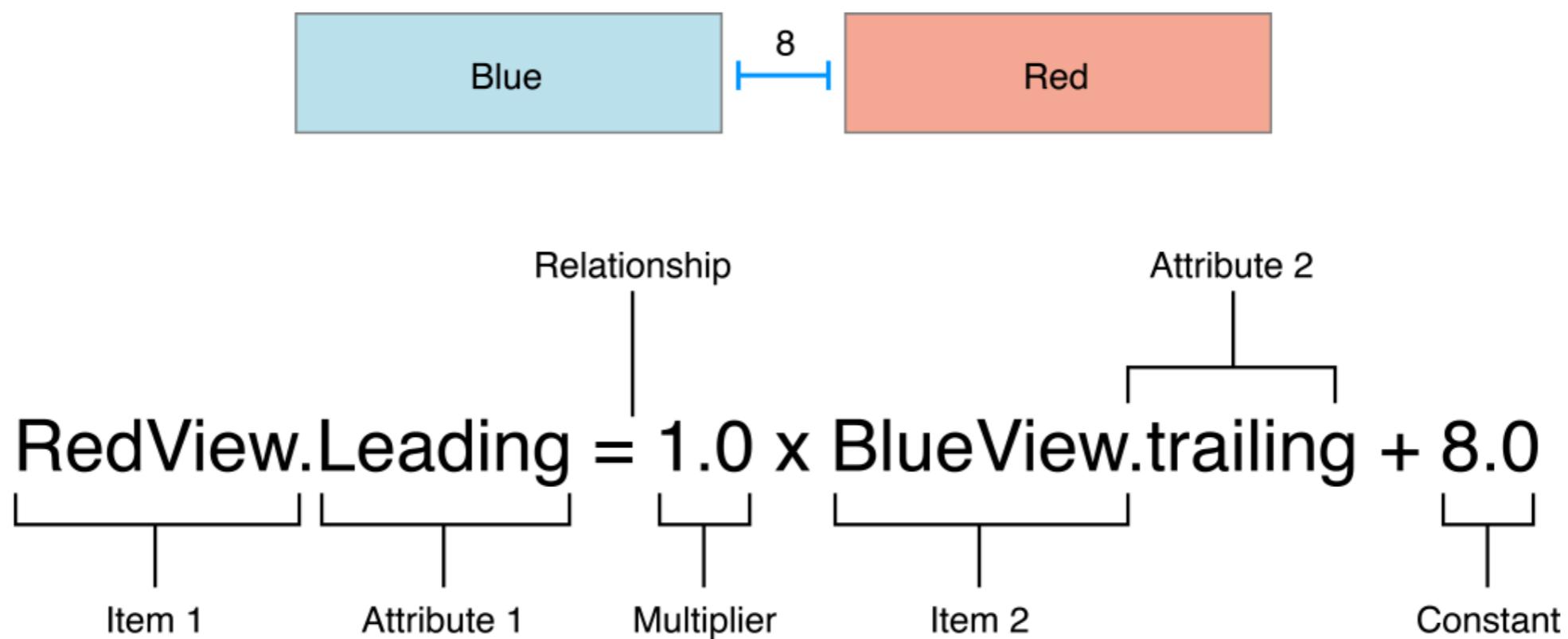
- **height** - height of view
- **width** - width of view
- **top** - vertical spacing to top view
- **bottom** - vertical spacing to bottom view
- **baseline** - align baseline
- **leading** - spacing to left view
- **trailing** - spacing to right view
- **center x** - center align horizontally
- **center y** - center align vertically

a clarification

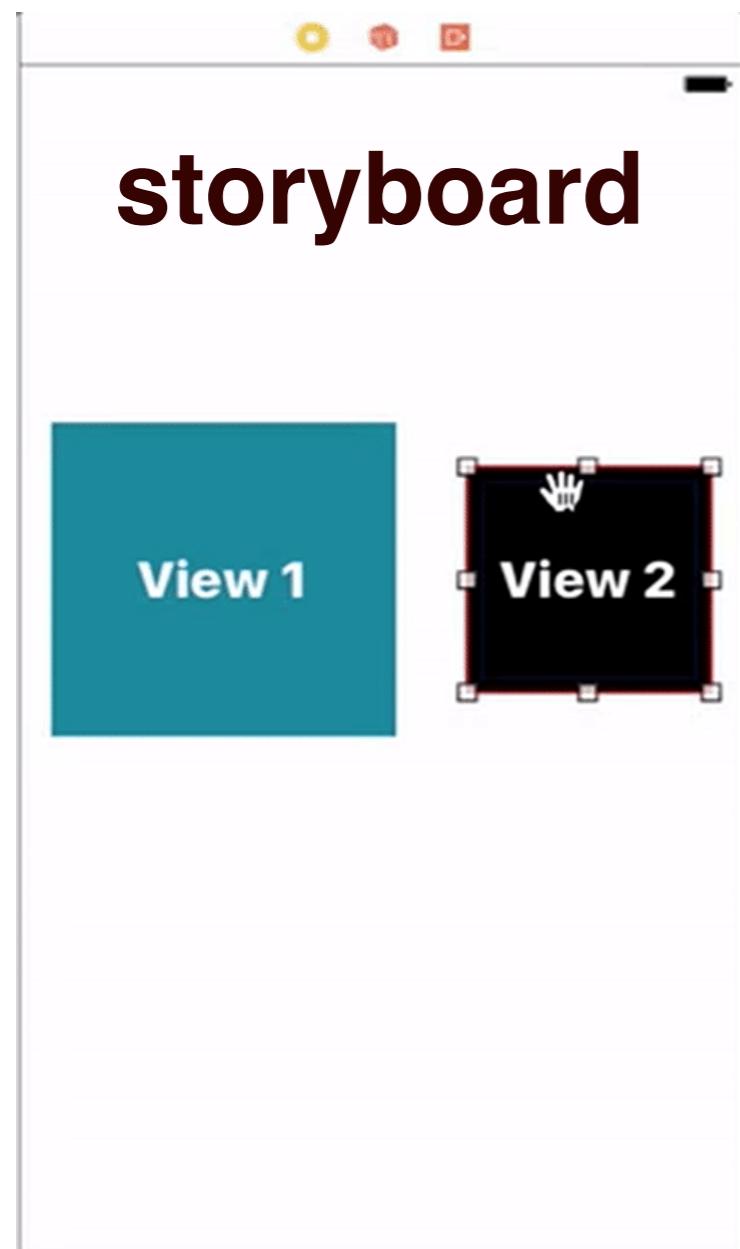
bottom

baseline

constraints (high level)



implementing AutoLayout

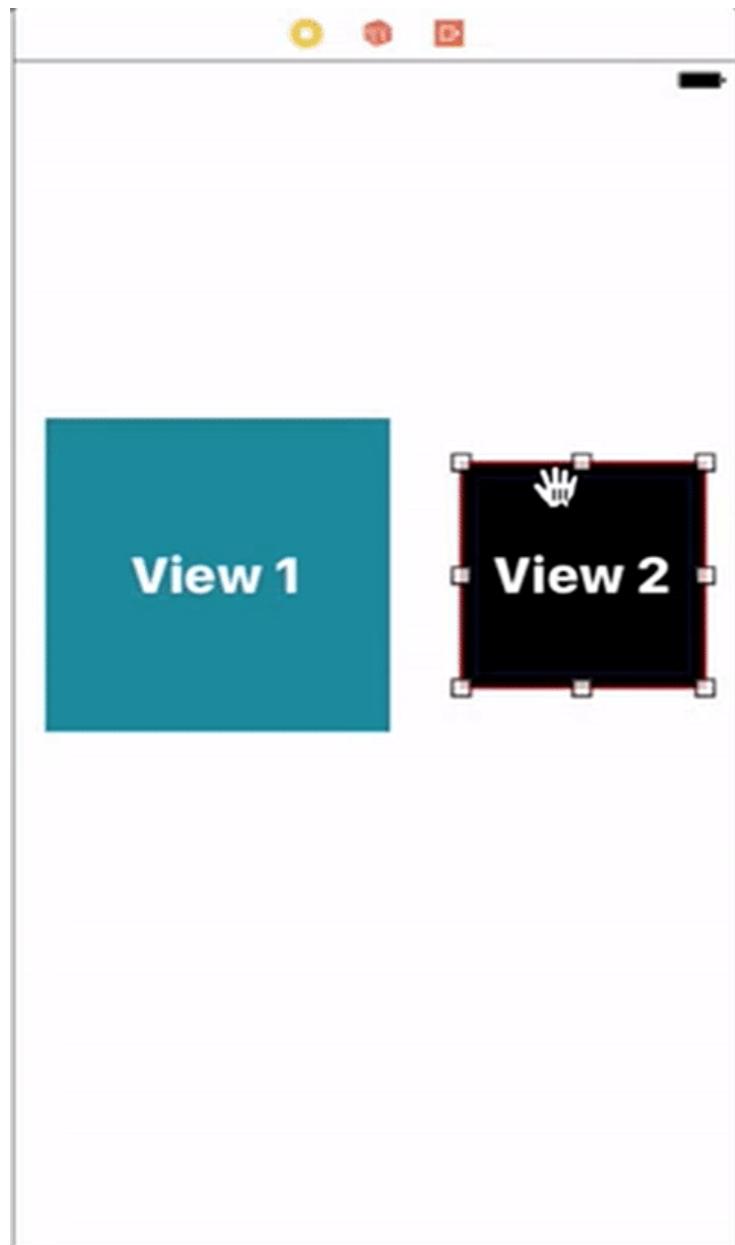


programmatically

```
let constraint =  
    view2.leadingAnchor.constraint(  
        equalTo: view1.trailingAnchor,  
        constant: 8)  
  
constraint.isActive = true
```

in both of these examples, the spacing between view's is set to 8 points

implementing AutoLayout (storyboard)

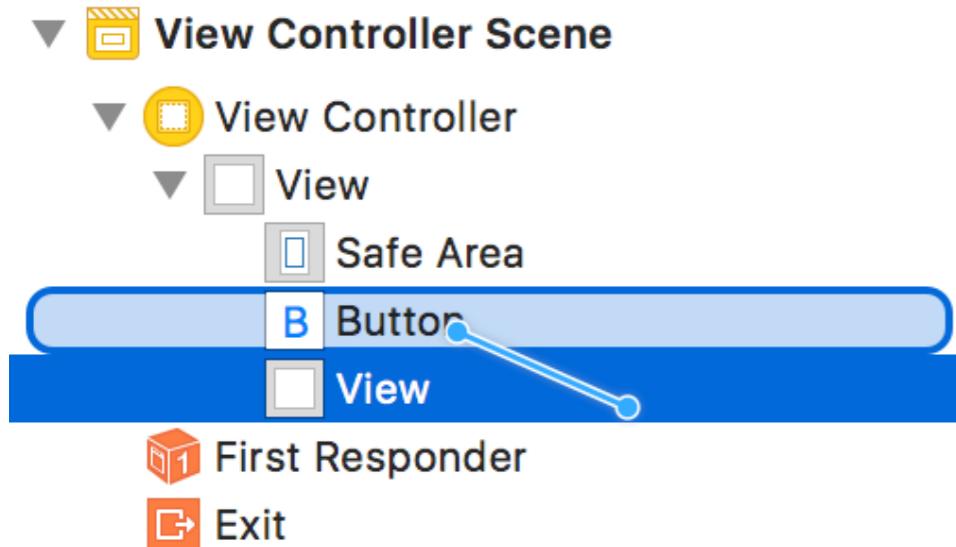


to create a constraint between two views in Storyboard, you can either...

- **control + drag** between the two views
- **control + drag** between view names in the document outline
- use align + add new constraints menu's

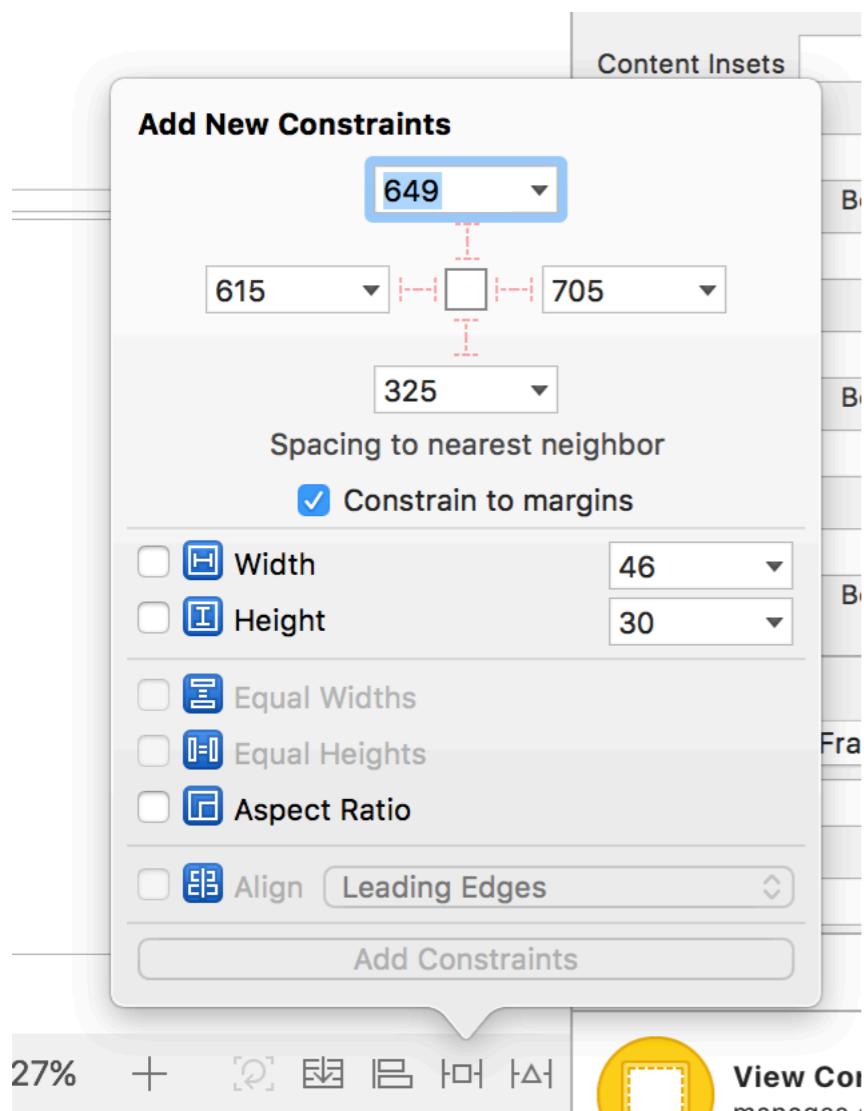
implementing AutoLayout (storyboard)

to create a constraint between two views in Storyboard, you can either...



- **control + drag** between the two views
- **control + drag** between view names in the document outline
- use align + add new constraints menu's

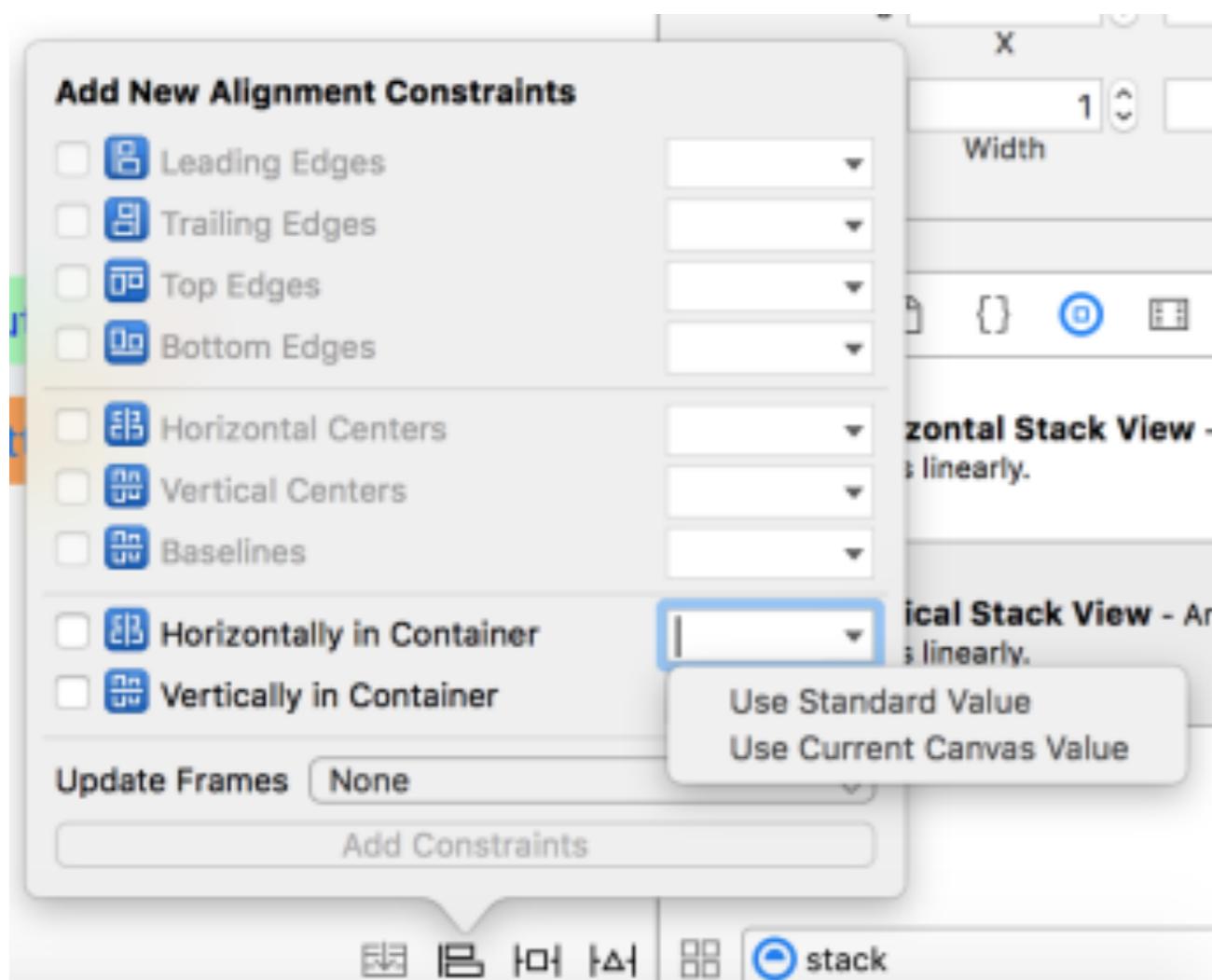
implementing AutoLayout (storyboard)



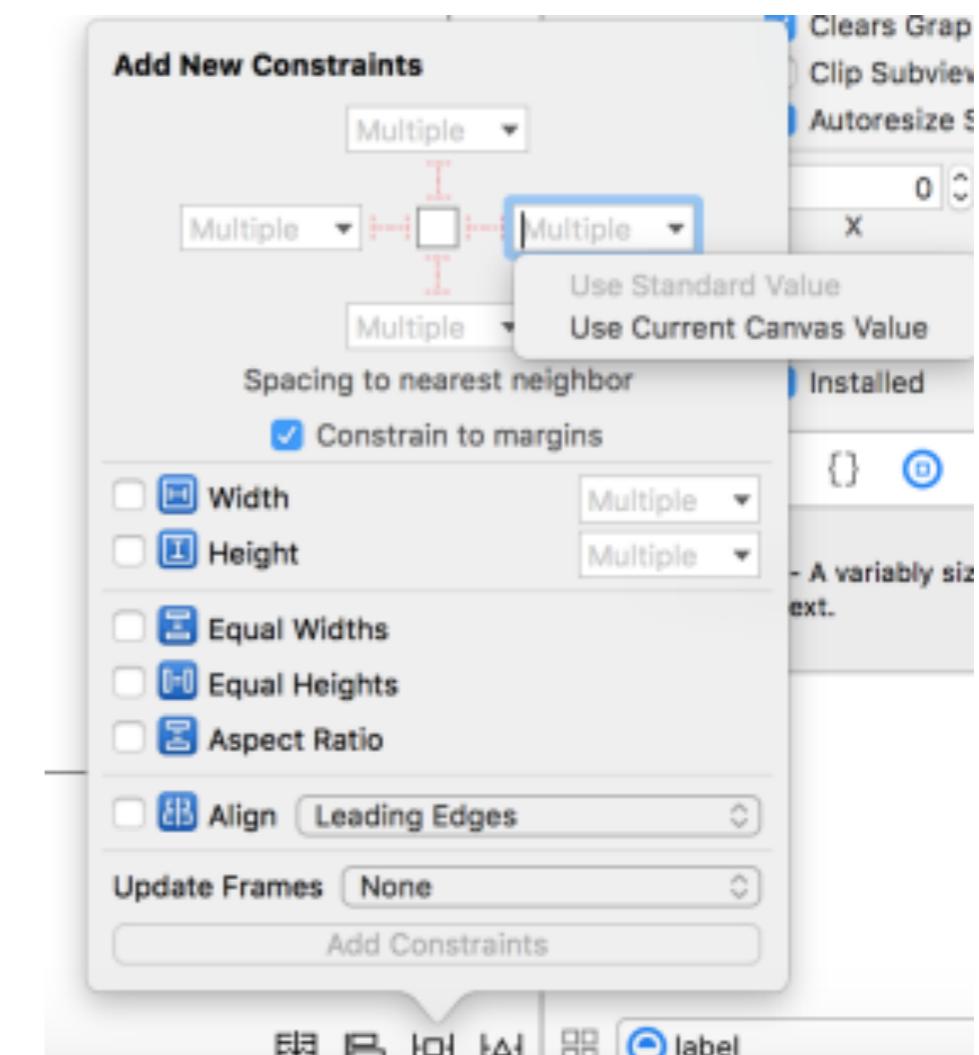
to create a constraint between two views in Storyboard, you can either...

- **control + drag** between the two views
- **control + drag** between view names in the document outline
- use **align + add new constraints** menu's

types of constraints

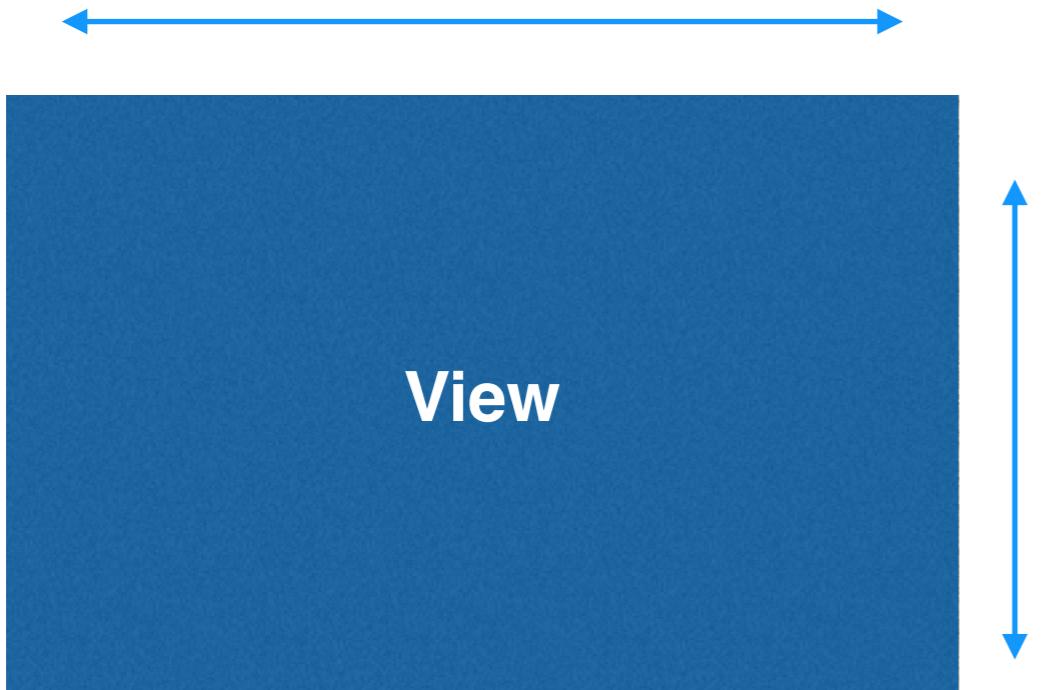


alignment: align objects with each other



pin: adds space to nearest neighbor (can be a superview or itself)

creating constraints - the philosophy



1. x position
2. y position
3. height
4. width

Running iosDecal on iPhone 7 Plus

Structure View Controller

Missing Constraints

- View2
Need constraints for: Y position
- View2
Need constraints for: X position or width

Debug Storyboard Constraint Issues Here!

(Click the red dots for suggested solutions)

Update Constraints you've made in Storyboard here ->

Constraints

- All This Size Class
- Align Center Y to: Superview Edit
- Leading Space to: Superview Edit
- Width Equals: 180 Edit
- Height Equals: 180 Edit

Showing 4 of 4

Content Hugging Priority

- Horizontal 250
- Vertical 250

Content Compression Resistance Priority

View as: iPhone 6s (wC hR)

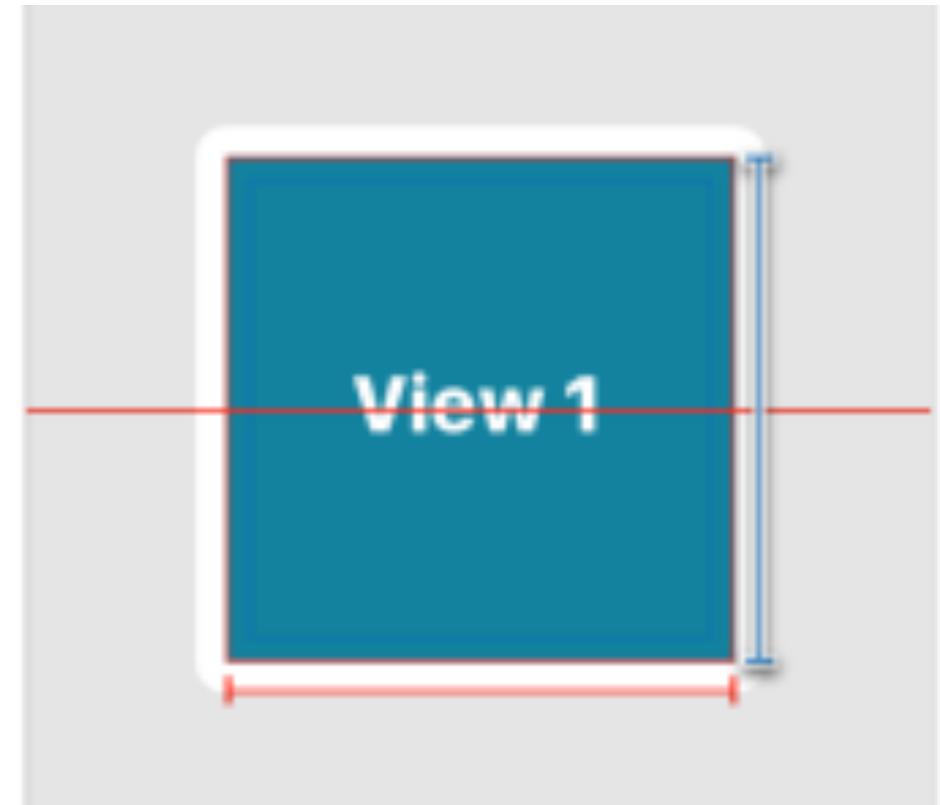
Filter

Auto Layout debugging

red lines - Interface Builder

If you see red lines in interface builder, that means you either have:

- too few constraints
(ambiguous)
- too many constraints
(conflicting)

A screenshot of the Xcode Problems list. The title bar says "View Controller". There are two entries under "Missing Constraints":

- "Image View Need constraints for: X position, width" (with a red dot icon)
- "Image View Need constraints for: height" (with a red dot icon)

Both entries are preceded by a small square icon containing a camera symbol.

conflicts

◀ Structure

View Controller

▼ **Conflicting Constraints**



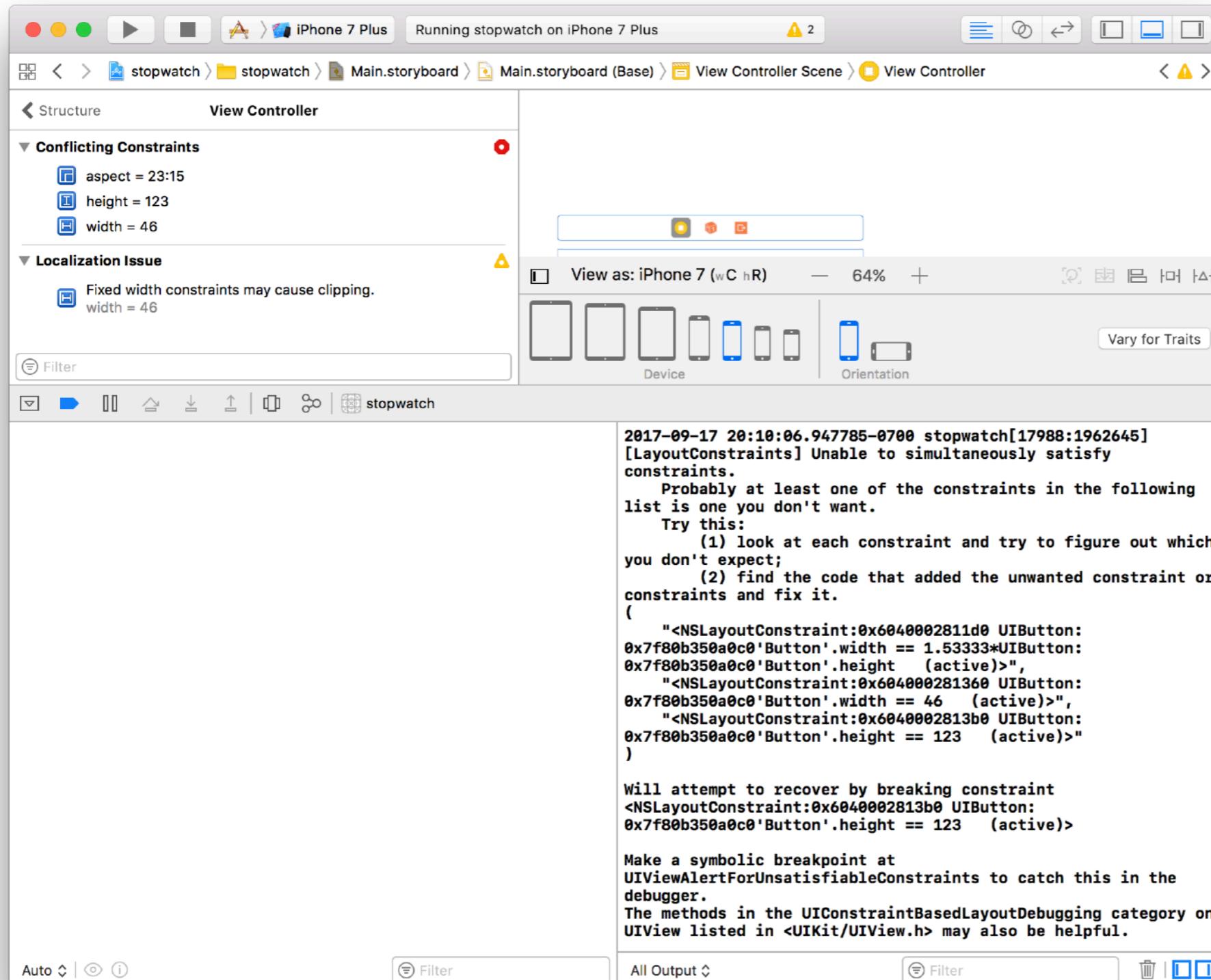
aspect = 23:15

height = 400

width = 46

Xcode will warn you when create multiple conflicting constraints

conflicts



conflicts

The screenshot shows the Xcode interface with a storyboard file open. The top menu bar indicates "Running stopwatch on iPhone 7 Plus". The storyboard navigation bar shows the project structure: stopwatch > stopwatch > Main.storyboard > Main.storyboard (Base) > View Controller Scene > View Controller.

The left sidebar displays "View Controller" with sections for "Conflicting Constraints" and "Localization Issue". Under "Conflicting Constraints", there are three entries: aspect = 23:15, height = 123, and width = 46. Under "Localization Issue", it says "Fixed width constraints may cause clipping." followed by "width = 46".

A large red warning icon is visible in the storyboard canvas area.

The right side of the screen shows a log window with the following text:

```
2017-09-17 20:10:06.947785-0700 stopwatch[17988:1962645]
[LayoutConstraints] Unable to simultaneously satisfy
constraints.

Probably at least one of the constraints in the following
list is one you don't want.

Try this:
(1) look at each constraint and try to figure out which
you don't expect;
(2) find the code that added the unwanted constraint or
constraints and fix it.

(
    "<NSLayoutConstraint:0x6040002811d0 UIButton:
0x7f80b350a0c0'Button'.width == 1.53333*UIButton:
0x7f80b350a0c0'Button'.height (active)>",
    "<NSLayoutConstraint:0x604000281360 UIButton:
0x7f80b350a0c0'Button'.width == 46 (active)>",
    "<NSLayoutConstraint:0x6040002813b0 UIButton:
0x7f80b350a0c0'Button'.height == 123 (active)>"
)

Will attempt to recover by breaking constraint
<NSLayoutConstraint:0x6040002813b0 UIButton:
0x7f80b350a0c0'Button'.height == 123 (active)>

Make a symbolic breakpoint at
UIViewAlertForUnsatisfiableConstraints to catch this in the
debugger.
The methods in the UIConstraintBasedLayoutDebugging category on
UIView listed in <UIKit/UIKit.h> may also be helpful.
```

localization issues

▼ Localization Issue

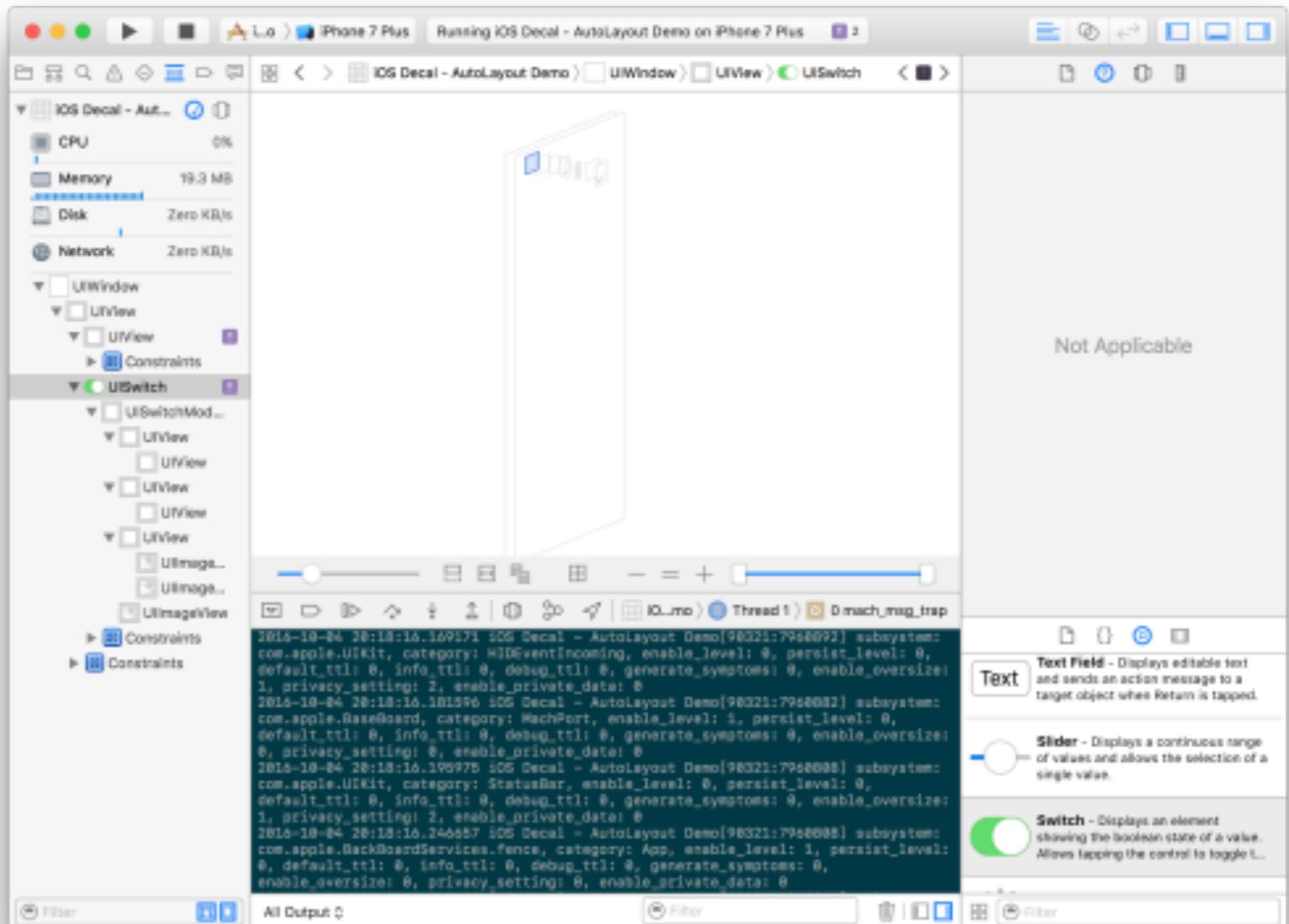
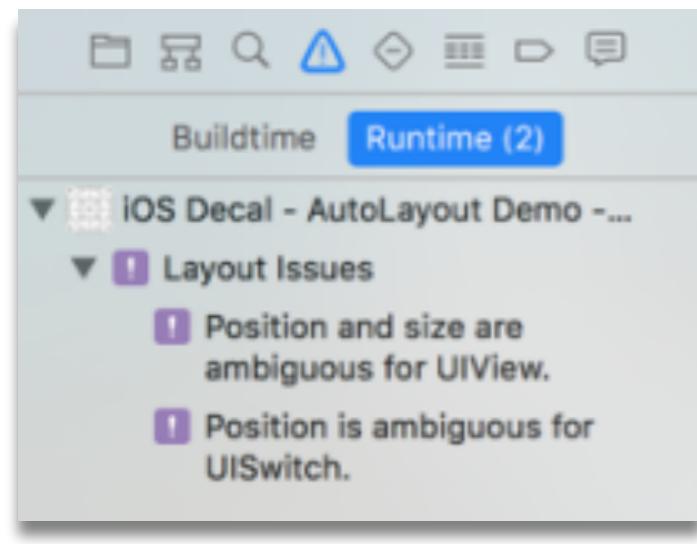


Fixed width constraints may cause clipping.

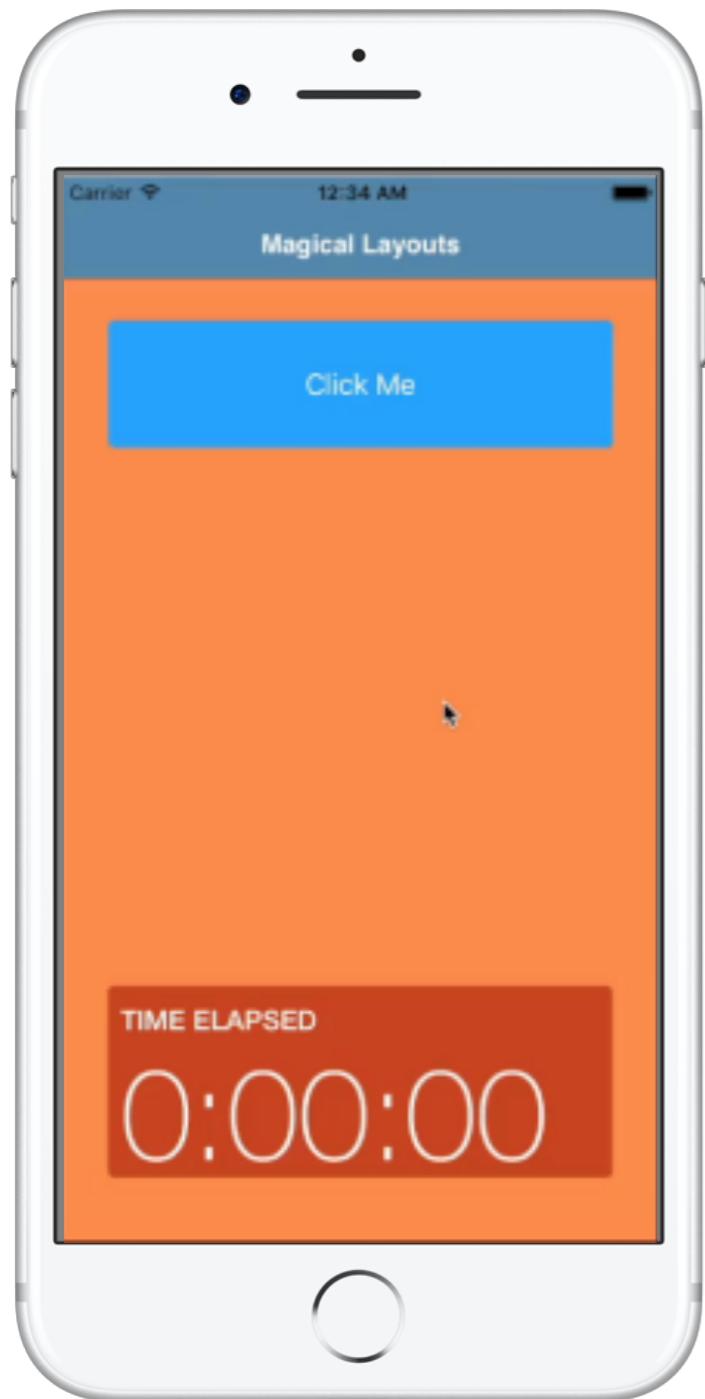
width = 46

fixed width constraints for text elements generate localization warnings

Debug View Hierarchy



Auto Layout with stack views



**use stack views to cut down
on the amount of layout work
you need to do**

side note: setting a stack
subview's hidden property to
true animates beautifully

stack view properties

arrangedSubviews - the views inside the stack

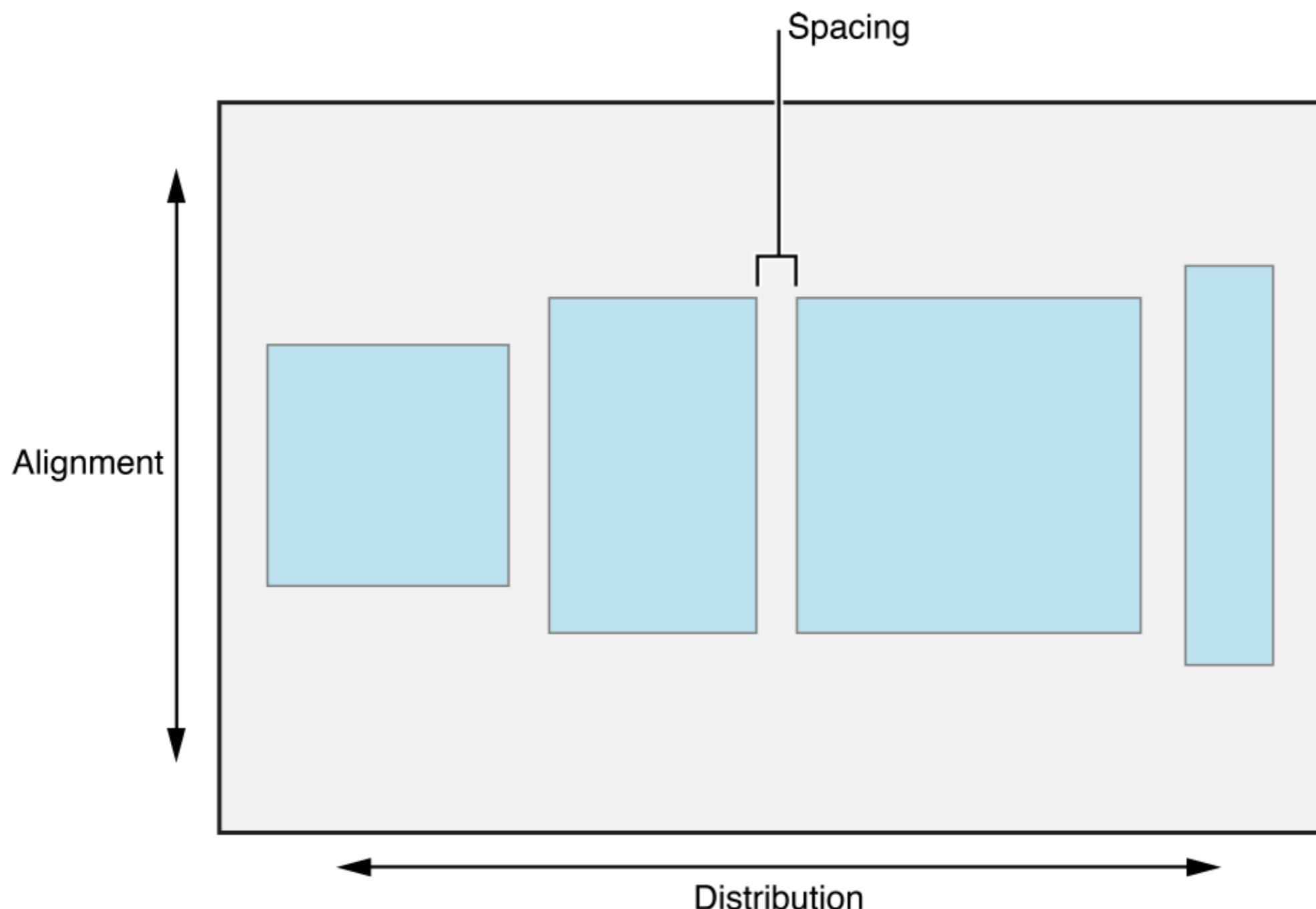
distribution - the distribution of the arranged views **along the stack view's axis**

alignment - The alignment of the arranged subviews
perpendicular to the stack view's axis

axis - horizontal or vertical

spacing - space between subviews

stack view properties

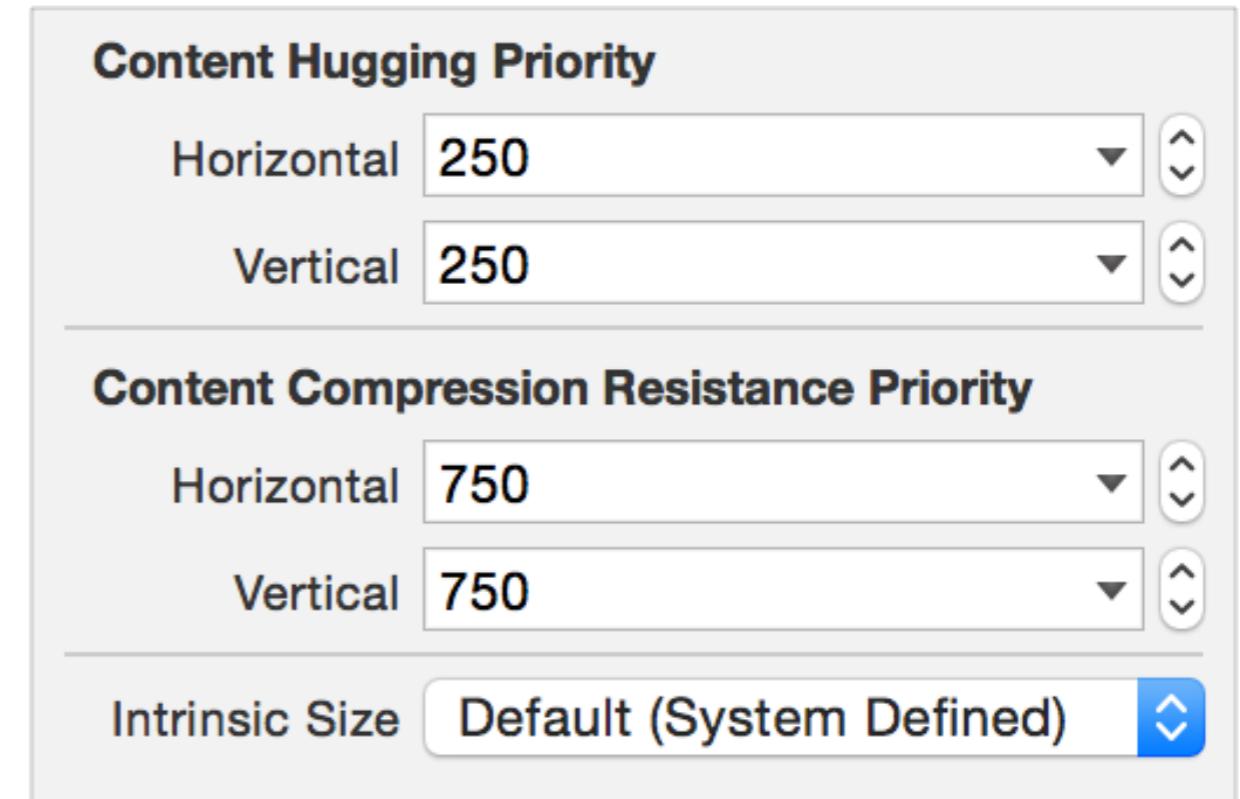


hugging / compression

most of the time - you can avoid setting this

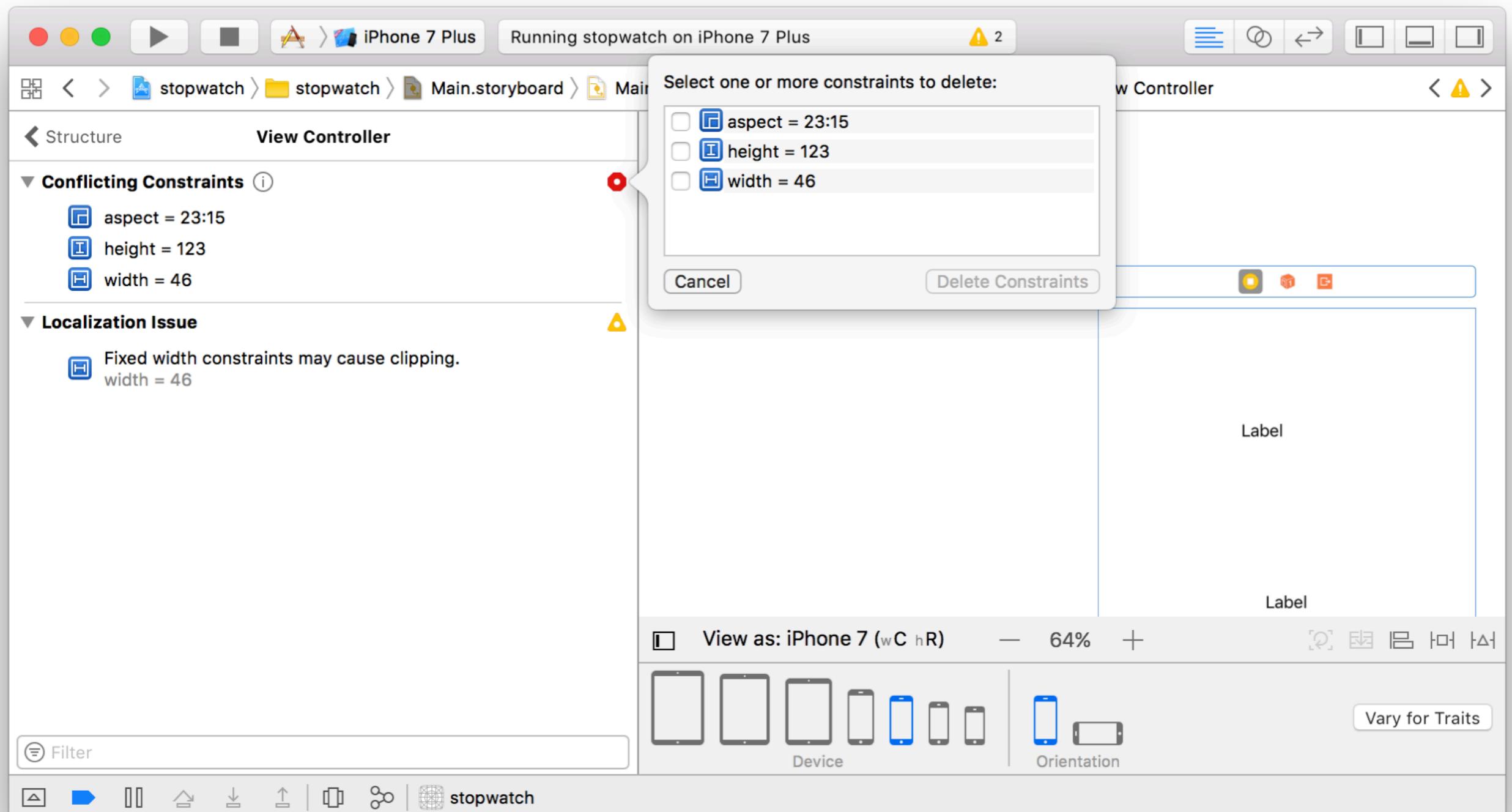
hugging - how much view *does not* want to grow

compression - how much view *does not* want to shrink



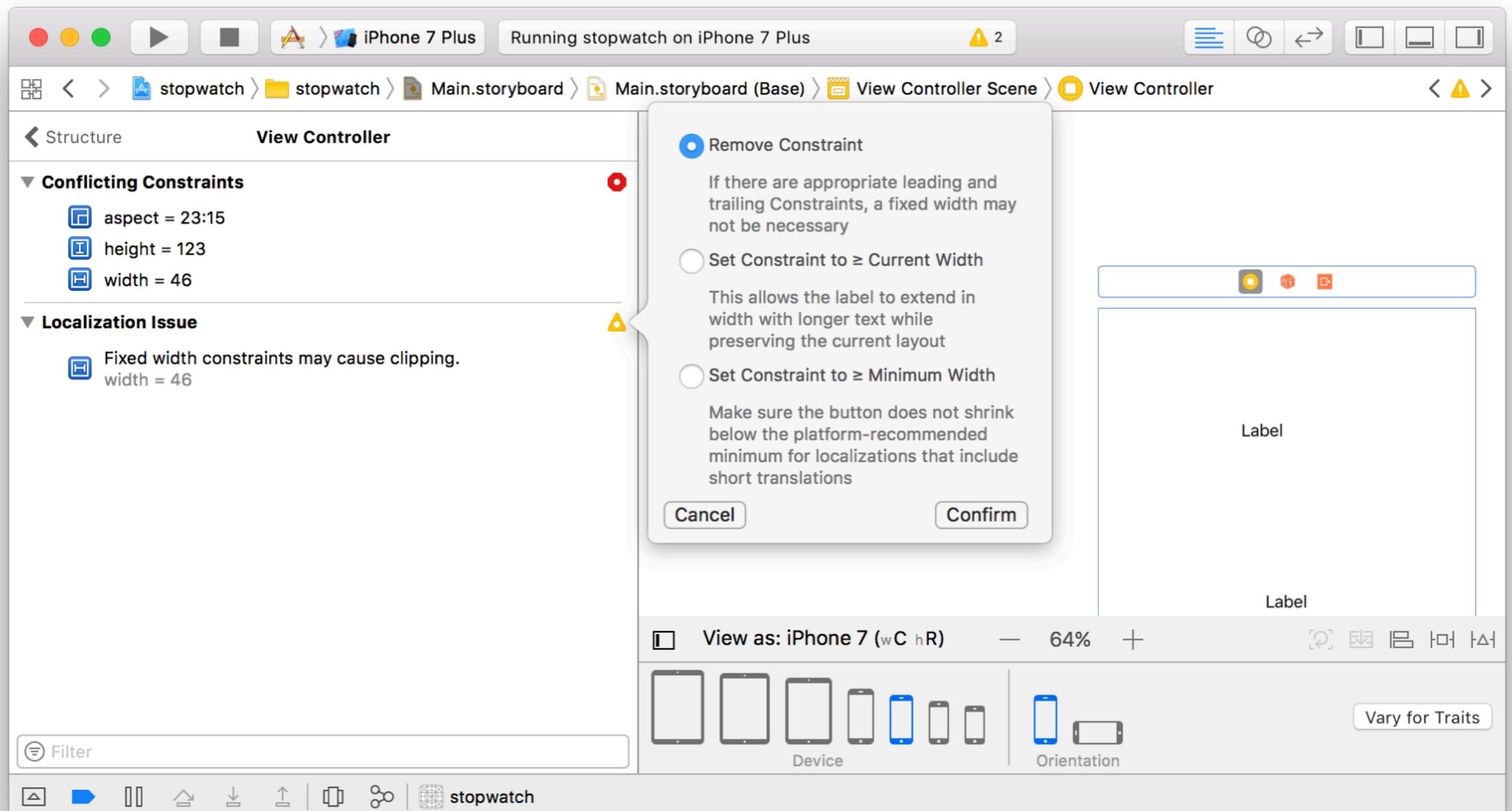
**some AutoLayout
tricks**

resolving conflicts



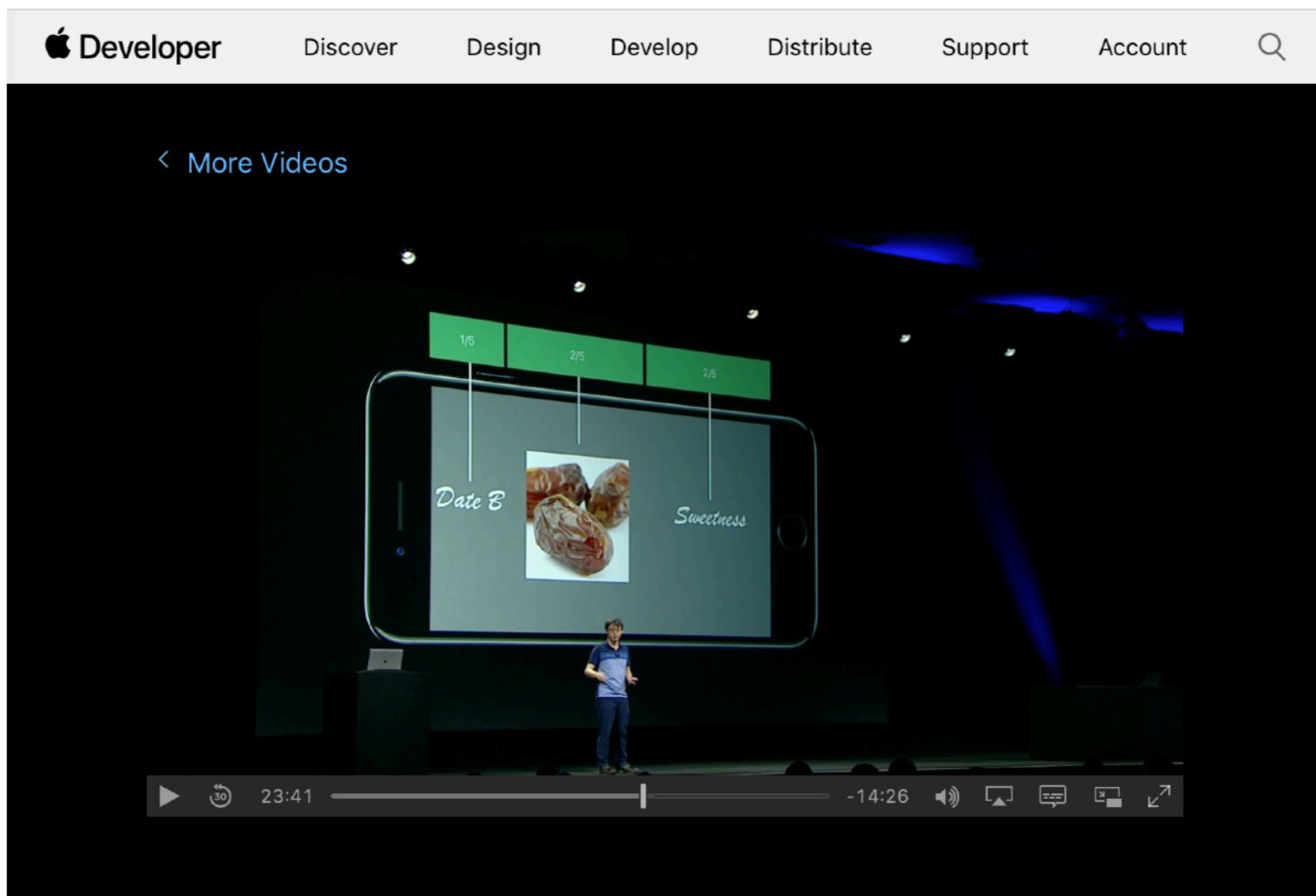
tap on warning icon to reveal tips to resolve your issue

resolving localization issues



spacer views

spacer views - hidden UIViews to enforce proportions



preview mode

Running stopwatch on iPhone 7

View Controller Scene

View Controller

View

Safe Area

Stack View

Button

IMG_6278

Label

Constraints

First Responder

Exit

Storyboard Entry Point

View as: iPad Pro 12.9" (wR hR) 30% +

Device Orientation Adaptation Vary for Traits

Filter stopwatch

Preview Main.storyboard (Preview)

Main.storyboard (Preview)

Button

Label

Label

iPad Pro 10.5" | Portrait | Full Screen

English

The screenshot displays the Xcode interface with the storyboard editor open. The top navigation bar shows the project name 'stopwatch' and the target device 'iPhone 7'. The main workspace shows a storyboard scene with a 'View Controller Scene' containing a 'View Controller' object. Inside the view controller, there is a 'View' which includes a 'Safe Area' and a 'Stack View'. The stack view contains a 'Button' with the label 'Button' and an image of a white dog sitting on a tree branch. Below the stack view is a 'Label' with the text 'Label'. The bottom of the screen shows the Xcode interface with various tools and settings.

preview mode

Running stopwatch on iPhone 7

View Controller Scene

View Controller

View

Safe Area

Stack View

Button

IMG_6278

Label

Constraints

First Responder

Exit

Storyboard Entry Point

View as: iPad Pro 12.9" (wR hR) 30% +

Device Orientation Adaptation Vary for Traits

Filter

stopwatch

Running stopwatch on iPhone 7

Preview Main.storyboard (Preview)

Button

Button

Label

Label

Label Label

iPad Pro 10.5" | Portrait | Full Screen

Double-Length Pseudolanguage

pseudolanguage - make sure localized strings will appear correctly

The screenshot shows the Xcode interface with the storyboard editor open. The left panel displays the view hierarchy in the Document Outline, including a View Controller Scene, View Controller, View, Safe Area, Stack View, Button, IMG_6278 (image), Label, Constraints, First Responder, Exit, and Storyboard Entry Point. The main canvas shows a stack view containing a button labeled "Button" and a label below it. A preview on the right shows the same layout with the button labeled "Button" and the label below it labeled "Label Label". The preview is set to an iPad Pro 10.5" screen in portrait mode. At the bottom, there are device orientation and adaptation controls, and a note about "Double-Length Pseudolanguage".

preview mode

Running stopwatch on iPhone 7

View Controller Scene

View Controller

View

Safe Area

Stack View

Button

IMG_6278

Label

Constraints

First Responder

Exit

Storyboard Entry Point

view layout on multiple different screen types simultaneously

View as: iPad Pro 12.9" (wR hR) 30% +

Device Orientation Adaptation Vary for Traits

Filter

stopwatch

Multiple Languages

Preview Main.storyboard (Preview)

Button Button

Button

Button

Label Label

Label

Label

iPhone SE | Portrait

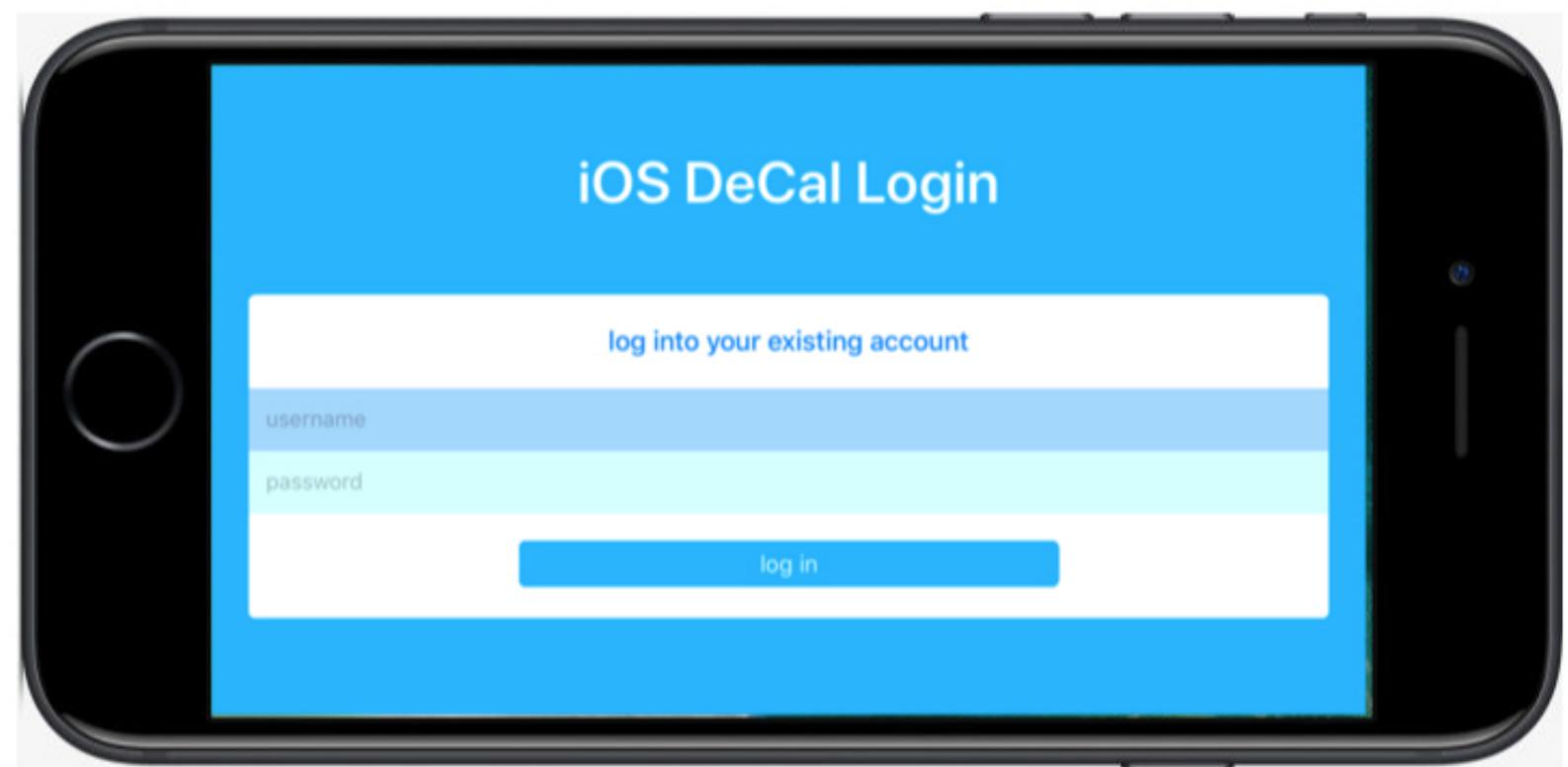
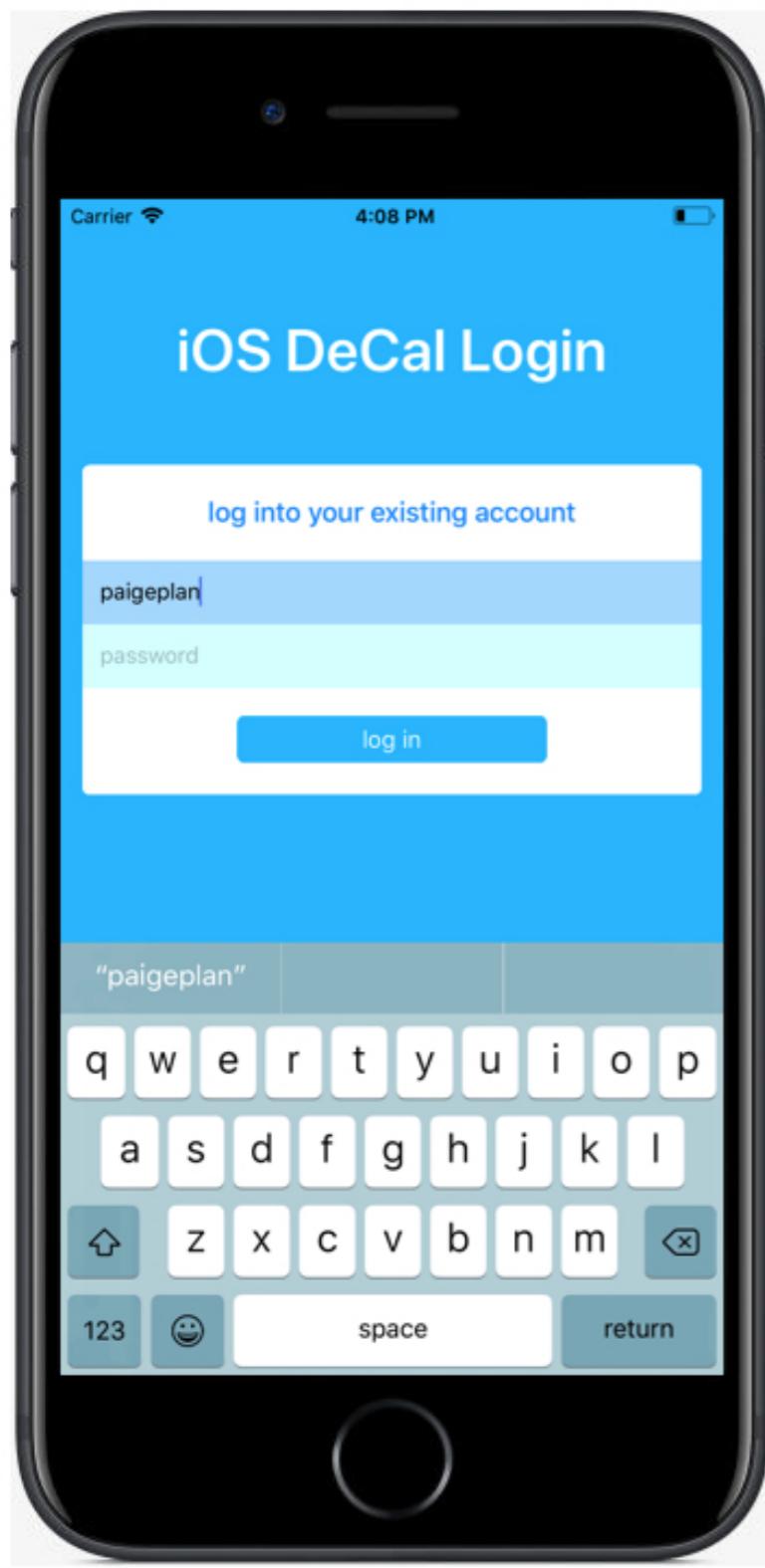
iPad Pro 10.5" | Portrait | Full Screen

The image shows a screenshot of the Xcode interface during the design phase of a mobile application. The main workspace displays a storyboard scene for an iPhone 7. This scene consists of a stack view containing a blue button labeled "Button" and a pink label below it. Below the stack view is a photograph of a white dog sitting on a tree branch. The Xcode interface includes a left-hand sidebar for navigating through the storyboard's components and a bottom toolbar for various editing functions. To the right of the main workspace, a preview panel is open, showing how the same layout would appear on an iPad Pro 10.5" in portrait mode and an iPhone SE in portrait mode. The preview panel also includes labels identifying the button and label components. The overall title of the slide is "preview mode", indicating the focus on the Xcode feature that allows designers to see their work across multiple devices simultaneously.

check-in

check-in today!

- I'll ask you to finish the UI I'm working on
- If you finish before class ends - call me over to check you off
- I'll publish a google form at the end of lecture if you didn't finish / didn't get checked off. You'll need to submit two screenshots of your progress in the form (one of your simulator in portrait, one in landscape)
 - Cmd + S while simulator is open



**didn't finish?
check piazza for a
tutorial video**

hw 2 : hangman
due Monday (9/18) at 11:59 pm

Next Lab : Auto Layout

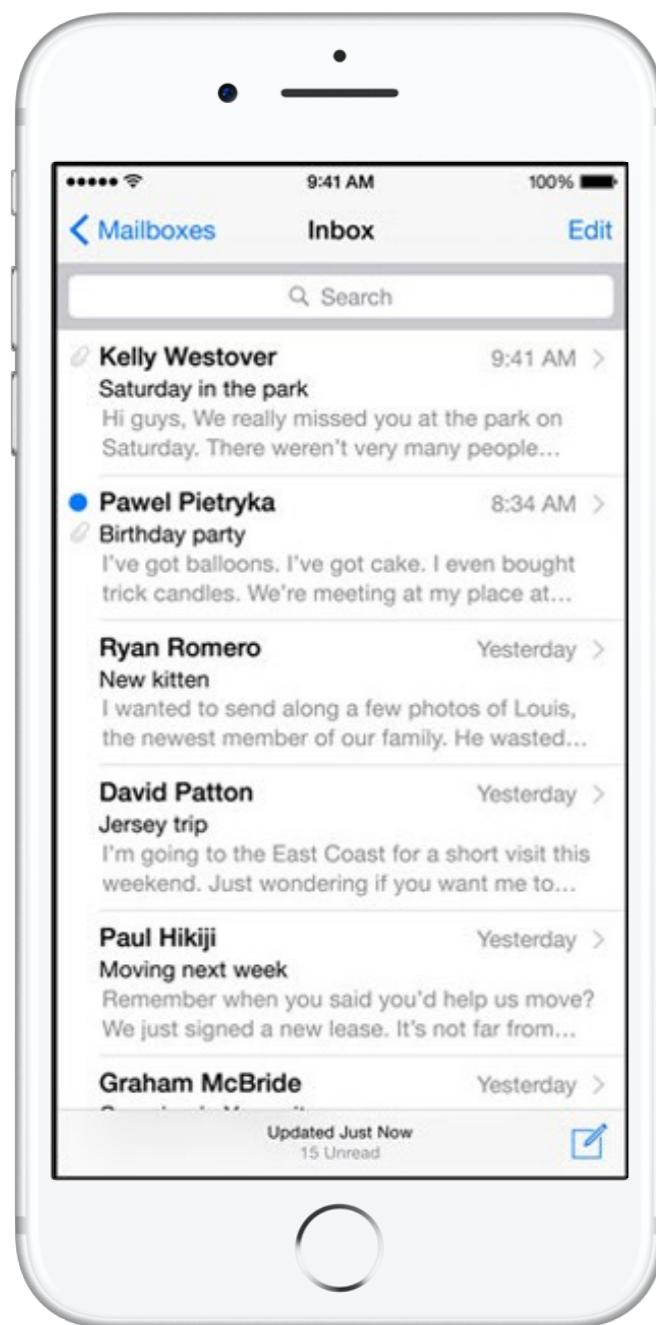
extra slides

... but what if I want a completely different
UI on device rotation?

answer

size classes

size classes - motivation



< Mailboxes **Inbox** Edit

From: [Kelly Westover](#) >

To: [Gilbert Solano](#) > [Graham McBride](#) >

Saturday in the park

September 9, 2014 at 9:41 AM

Hi guys,

We really missed you at the park on Saturday. There weren't very many people there, so it sort of felt like we had the whole place to ourselves.

Kelly Westover 9:41 AM
Saturday in the park
Hi guys, We really missed you at the park on Saturday. There weren't very many people...

Pawel Pietryka 8:34 AM
Birthday party
I've got balloons. I've got cake. I even bought trick candles. We're meeting at my place at...

Ryan Romero Yesterday
New kitten
I wanted to send along a few photos of Louis, the newest member of our family. He wasted...

David Patton Yesterday
Jersey trip
I'm going to the East Coast for a short visit this weekend. Just wondering if you want me to...

Paul Hikiji Yesterday
Moving next week
Remember when you said you'd help us move? We just signed a new lease. It's not far from...

Graham McBride Yesterday
Camping in Yosemite
Updated Just Now
15 Unread

size classes

The screenshot shows the Xcode Interface Builder storyboard editor. At the top, there's a blue toolbar with a "View as:" dropdown set to "iPad Pro 10.5" (wR hR), a zoom slider at 27%, and a "+" button. Below the toolbar is a blue status bar with a smartphone icon and a dropdown menu. The main area displays a storyboard scene with a blue view containing a white "Button" label, a central image of a white dog sitting on a tree branch, and a pink view containing a black "Label". An arrow points from the blue view down to the central image. To the right of the storyboard, a light gray sidebar labeled "No Selection" is visible. A callout bubble from the bottom right corner of the storyboard says "Introduce Variations Based On:" with checkboxes for "Width" and "Height" both checked. Below the storyboard, two cards are shown: "Table View Controller - A controller that manages a table view." and "Table View - Displays data in a list of plain, sectioned, or grouped rows.".

No Selection

Introduce Variations Based On:

- Width
- Height

Vary for Traits

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

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

View as: iPad Pro 10.5" (wR hR) — 27% +

Varying 12 Regular Width and Regular Height Devices

size classes

size classes allow you to create different constraints depending on the device size, orientation, type, etc.

