
Working With Device Rotations

— —

Harrison Packer | Emily Emmons | Sam Kochanski

Rotations on Devices

iPhone

- Support rotation if it complements the design
 - Camera app supports landscape and portrait
 - Weather app only supports portrait
- Never use upside-down portrait
 - Confusing upon incoming call

iPad

- Always support rotation unless absolutely necessary not to
 - iPad inherently rotation agnostic
 - Large enough to support multiple views in a single screen
- Should support upside-down portrait
 - iPads don't receive calls, so orientation is less consequential

Constraints

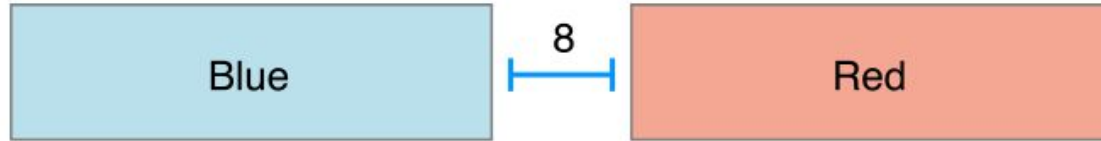
- Define a relationship between two items' **attributes**
- Used to “pin” items in a variable screen space

Example:

- Set **item 1** 16pts from bottom and left of the screen
- Set **item 2** the same width of **item 1** and 2pts above it
- Set **item 3** the same height as **item 1** and its left edge

Regardless of screen size and shape, these relationships will always be true

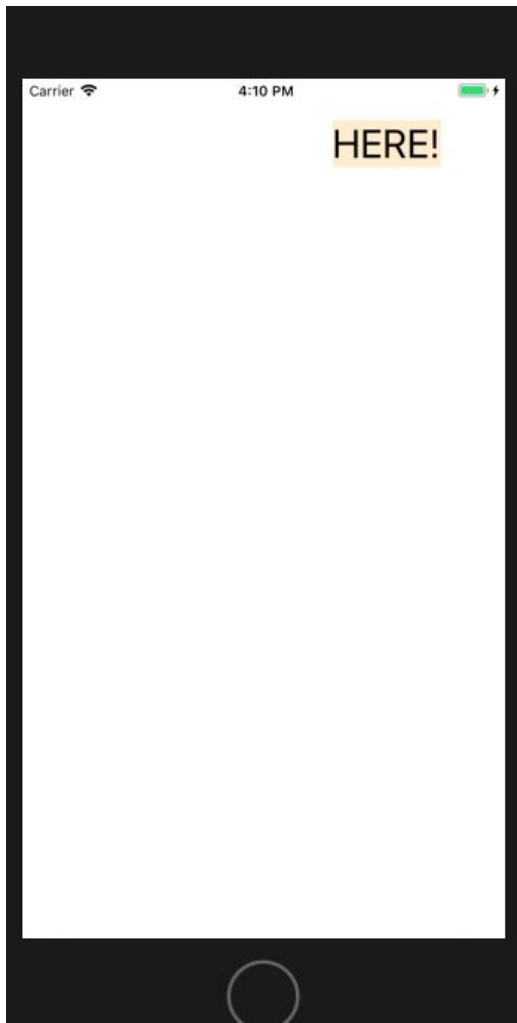
Constraints



$$\underbrace{\text{RedView.Leading}}_{\text{Item 1}} = \underbrace{1.0}_{\text{Multiplier}} \times \underbrace{\text{BlueView.trailing}}_{\text{Item 2}} + \underbrace{8.0}_{\text{Constant}}$$

Relationship

Attribute 2



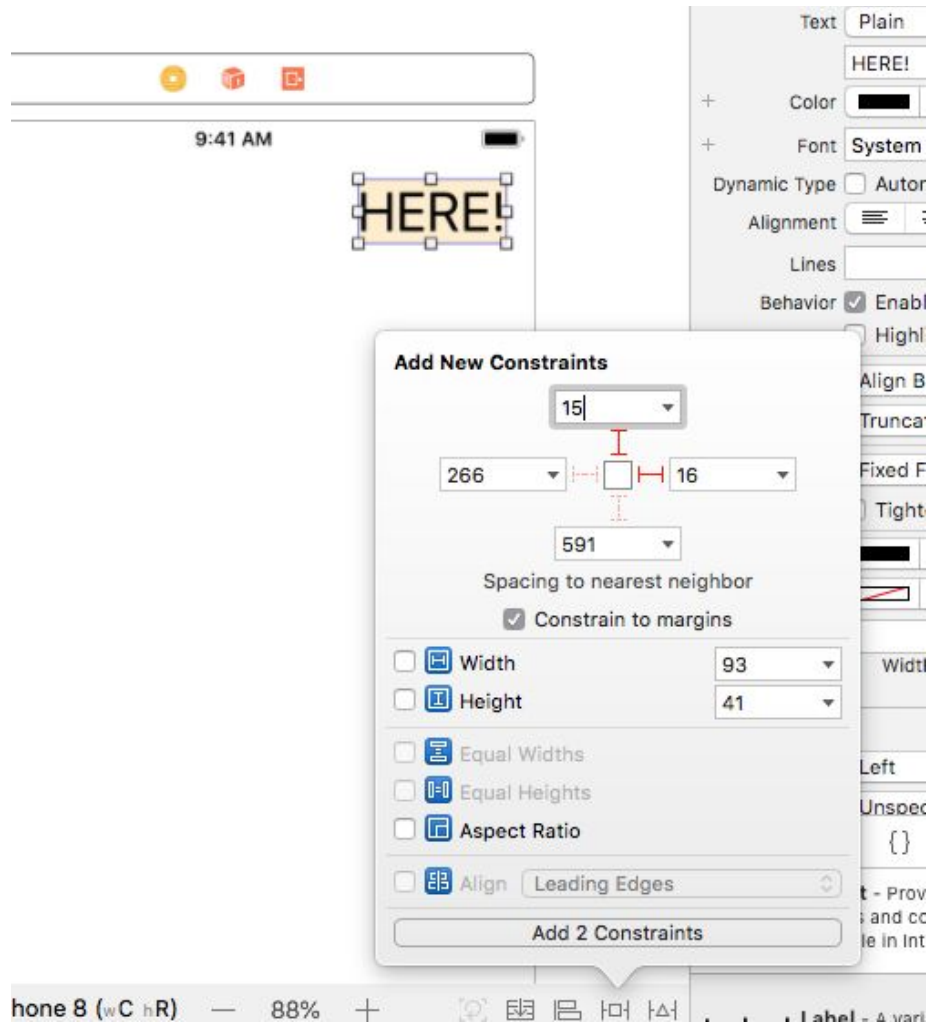
Pinning Constraints

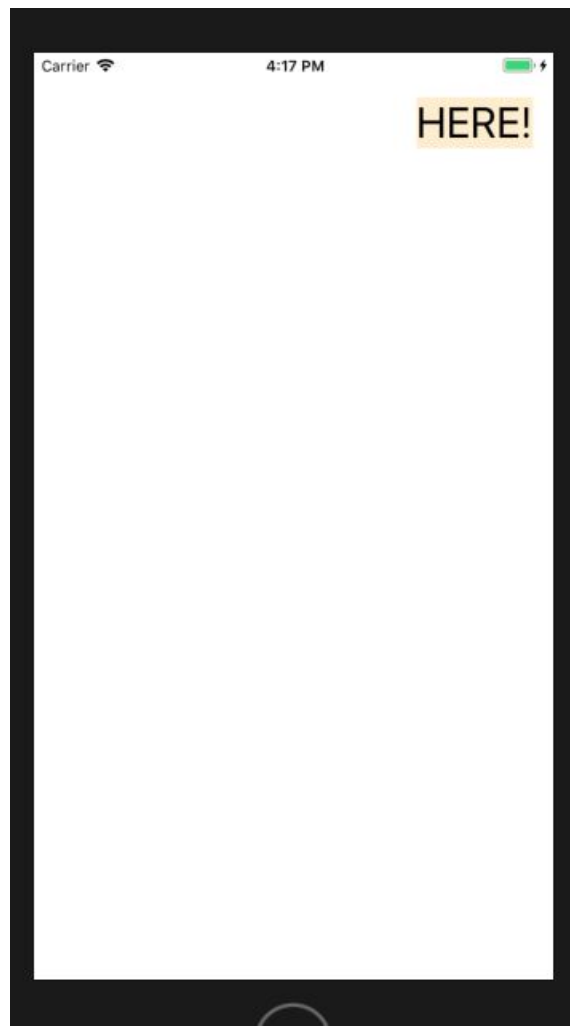
Connects label to Safe Area

Creates 2 constraints

- label 15pts to top
- label 16pts to right

Now...





Size Classes

iOS supports devices with different screen sizes and aspect ratios

To account, interfaces are built with one of two size classes in mind for each dimension

Compact - interface has limited space

Regular - interface has expansive space

Device Configuration

Every device is one of four permutations of size classes for both orientations, denoted by:





w - width, or x axis

h - height, or y axis

C - compact size class

R - regular size class

Example: iPhone 6 portrait **wC hR**
landscape **wR hC**

		Horizontal (width)	
		Compact	Regular
Vertical (height)	Compact	<p>iPhone 4S, 5, 6, 7, SE landscape</p> 	<p>iPhone 6+, 7+ landscape</p> 
	Regular	<p>iPhone portrait</p> 	<p>iPad portrait / landscape</p> 

Multiple Configurations, Single View

XCode lets you to set multiple configurations with different size classes within a single view

Reduces cluttering in project tree

Streamlines and enforces development of device-agnostic interfaces



Configuration Practices

When the current view is

Compact, UI elements should be closer together

Regular, UI elements should fill more space

Example: a view that is **wR hC** (like a landscape iPhone 6 plus) should not require horizontal elements to scroll, and minimize the footprint of vertical elements.

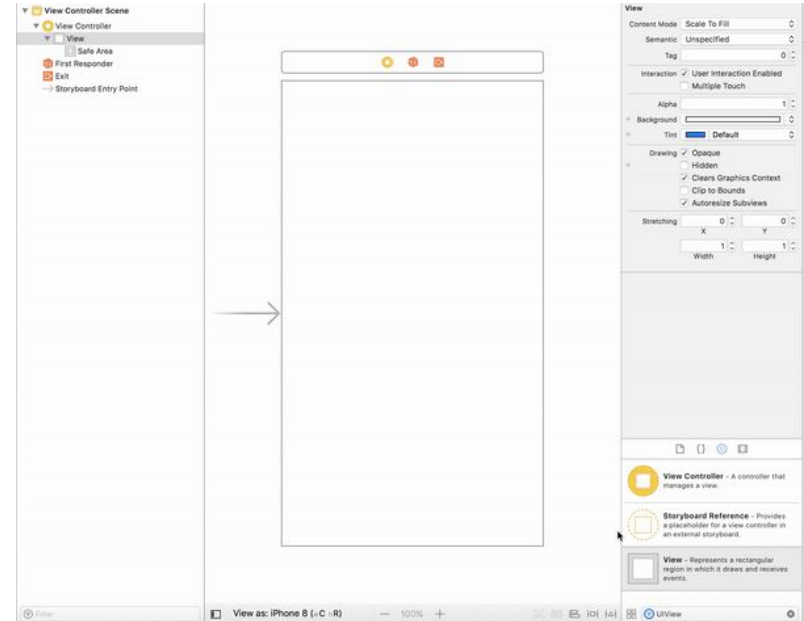
Let's Make an App!

1. Create a new project in XCode
2. Select Single-View-App
3. Give it a name, ex. "Rotation"
4. Click next and create the project
5. Select Main.storyboard

We will be working exclusively in the Storyboard Editor

Create a View Object

Click + drag in a View object from the Object Library to the storyboard view



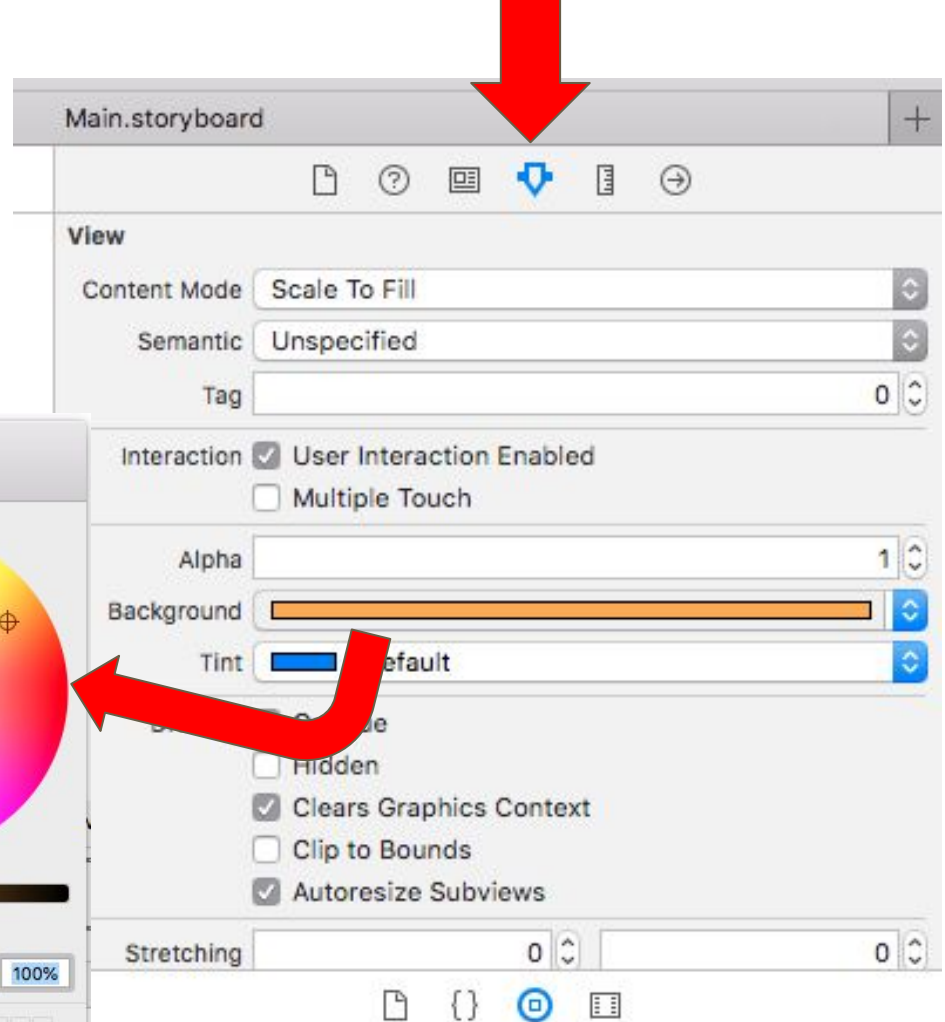
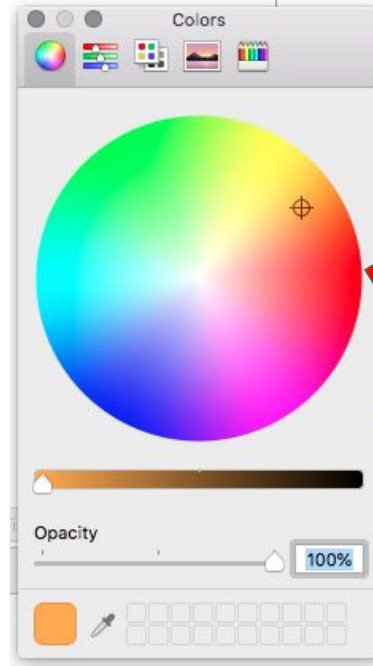
Set View Color

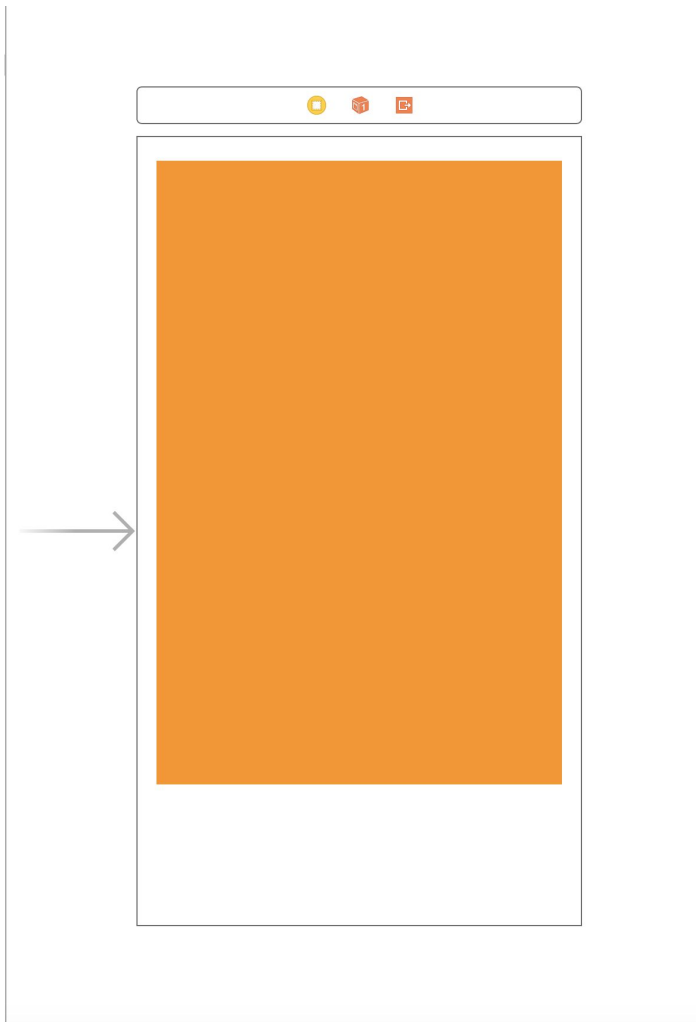
With the View object selected:

Attribute Editor -> View

Click Background to bring up color picker

This will update the View object's fill color





Resize the View

Resize the View by stretching close to the sides and the top

Leave some space at the bottom for the buttons

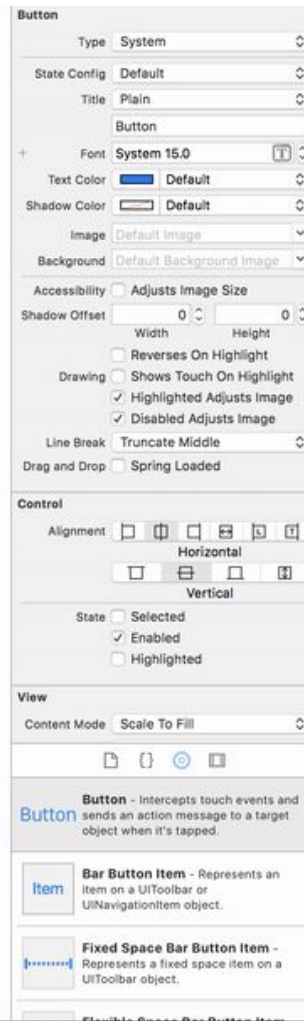
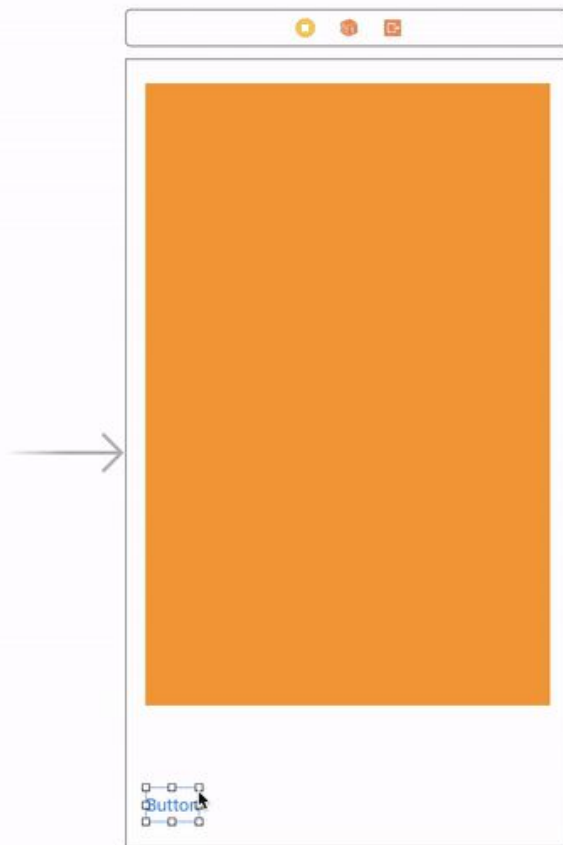
The blue guide lines are your friend! They define an appropriate margin

Create Buttons

Drag a button in and resize it to fit approximately half of blank space below the view

Option + drag to clone the button and drag it

Double click and rename the buttons to "Button 1" and "Button 2"

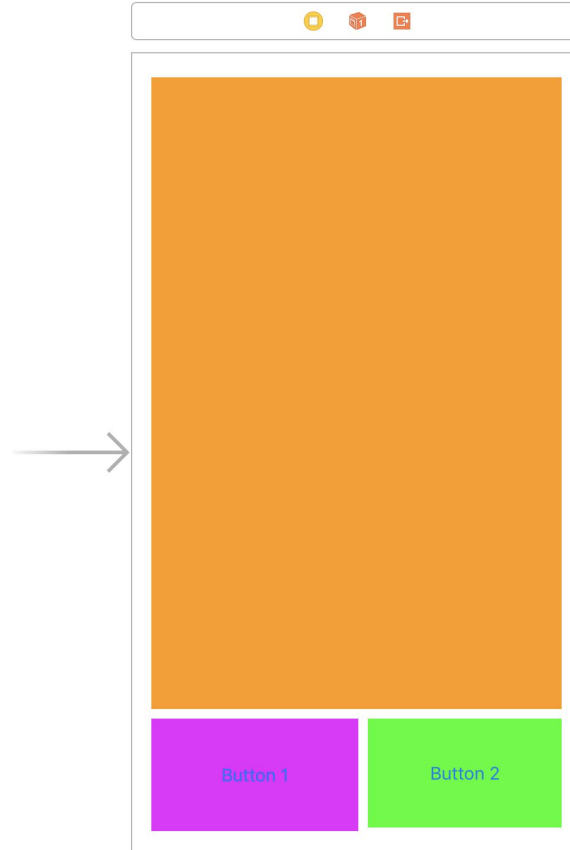


Now You Have a View!

Color the buttons so they are distinctive by selecting them and changing:

Attribute Editor -> View ->
Background

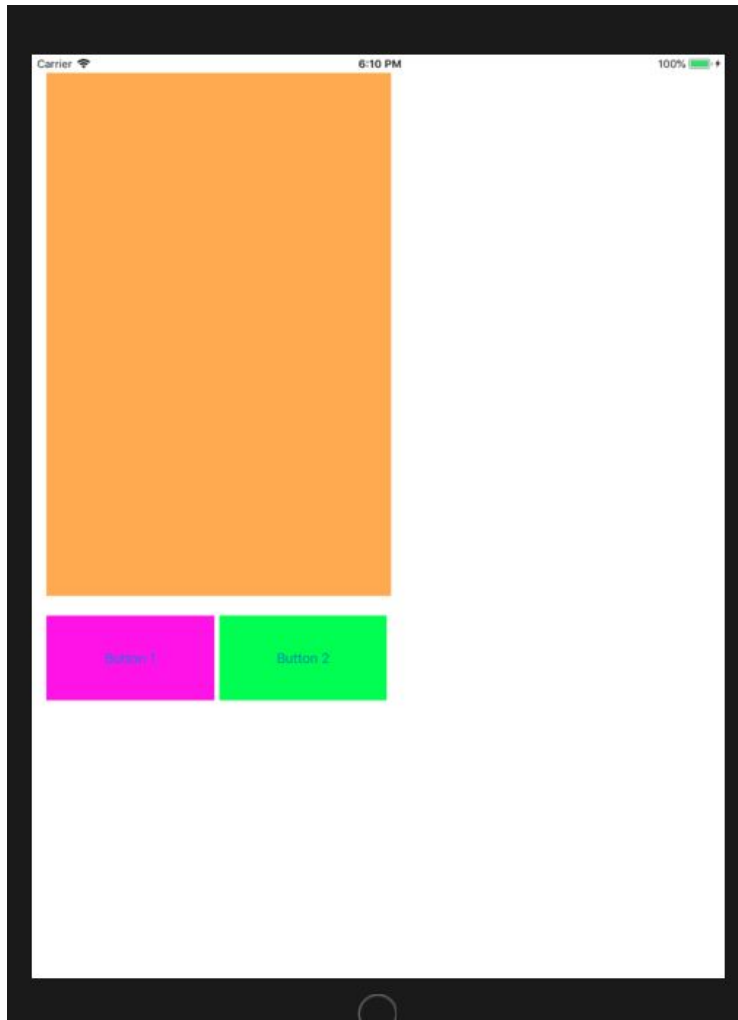
They should be the same size and not overlap with each other or the view object



Try it out!

Run the simulator!
Try different devices!

Since we don't have any constraints, the view doesn't scale or rotate properly for different devices



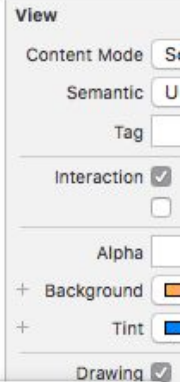
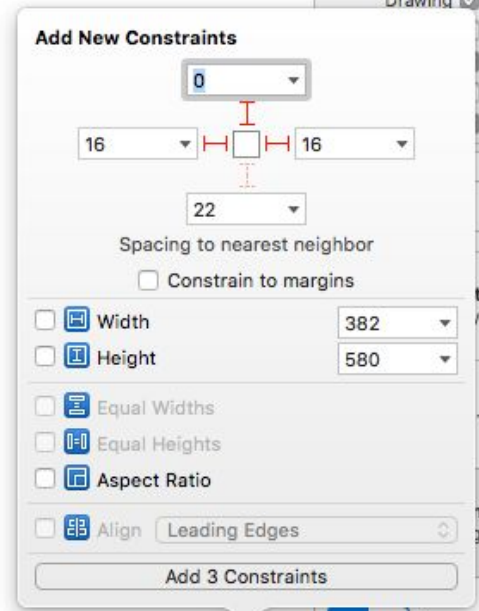
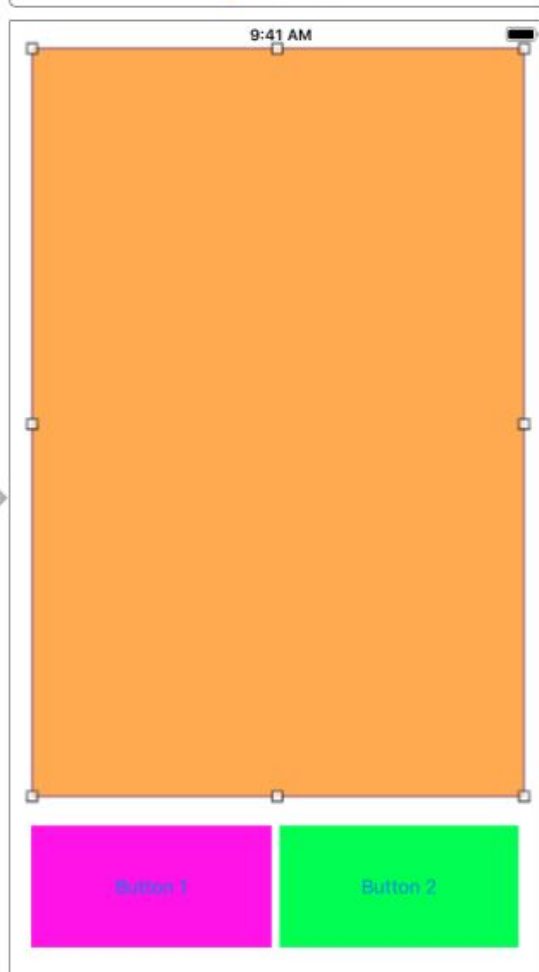
View Pinning

Select the View and the constraints button on the bottom right corner of storyboard

Select left, top, and right red lines in constraint menu

Select “Add 3 Constraints” to apply

This will “pin” it to the top, left, and right sides of the screen

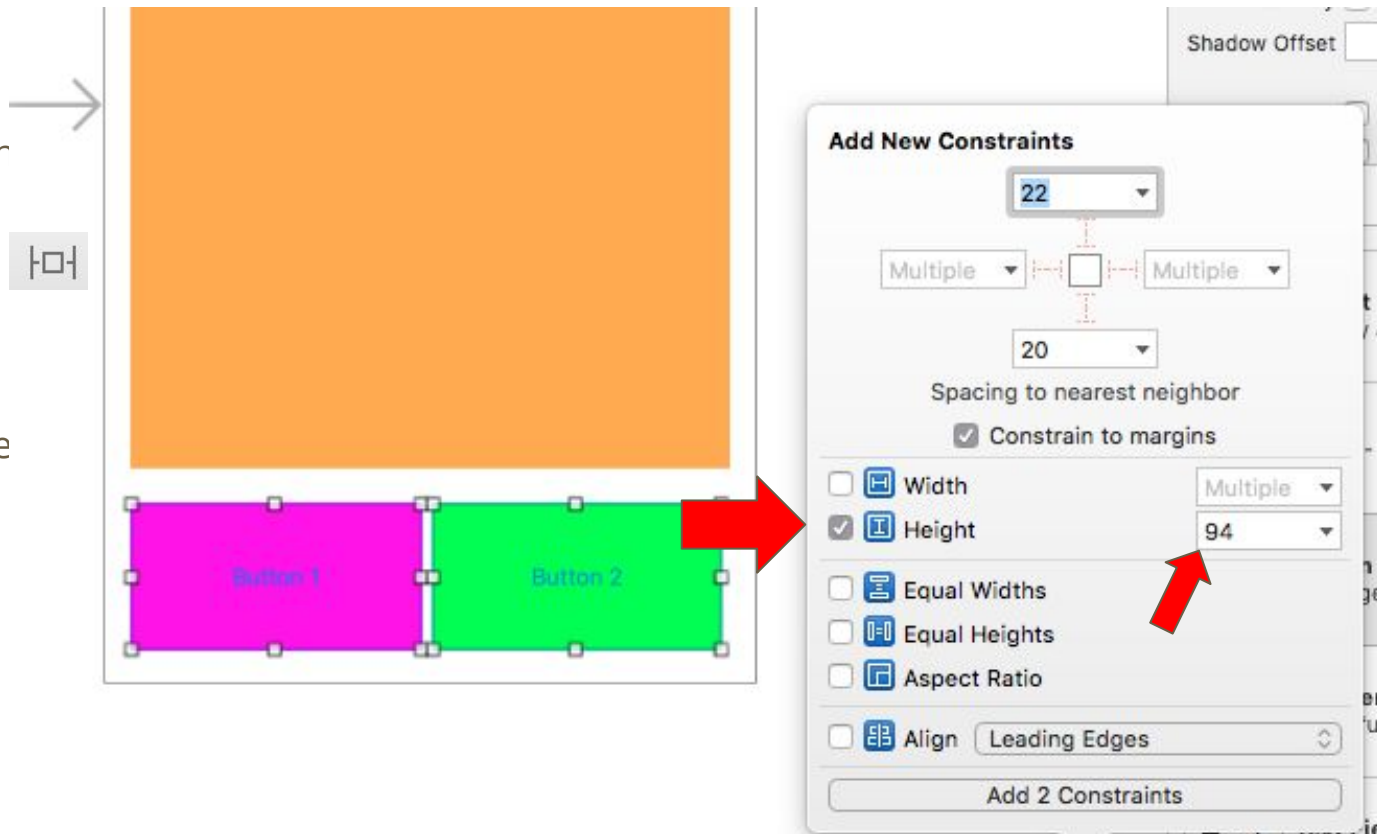


Button Height

Shift + click to select both buttons and open constraint menu

Select Height

Since the buttons are the same height, a number should appear in the box. Otherwise, set a height in pts.



Pinning Button 1

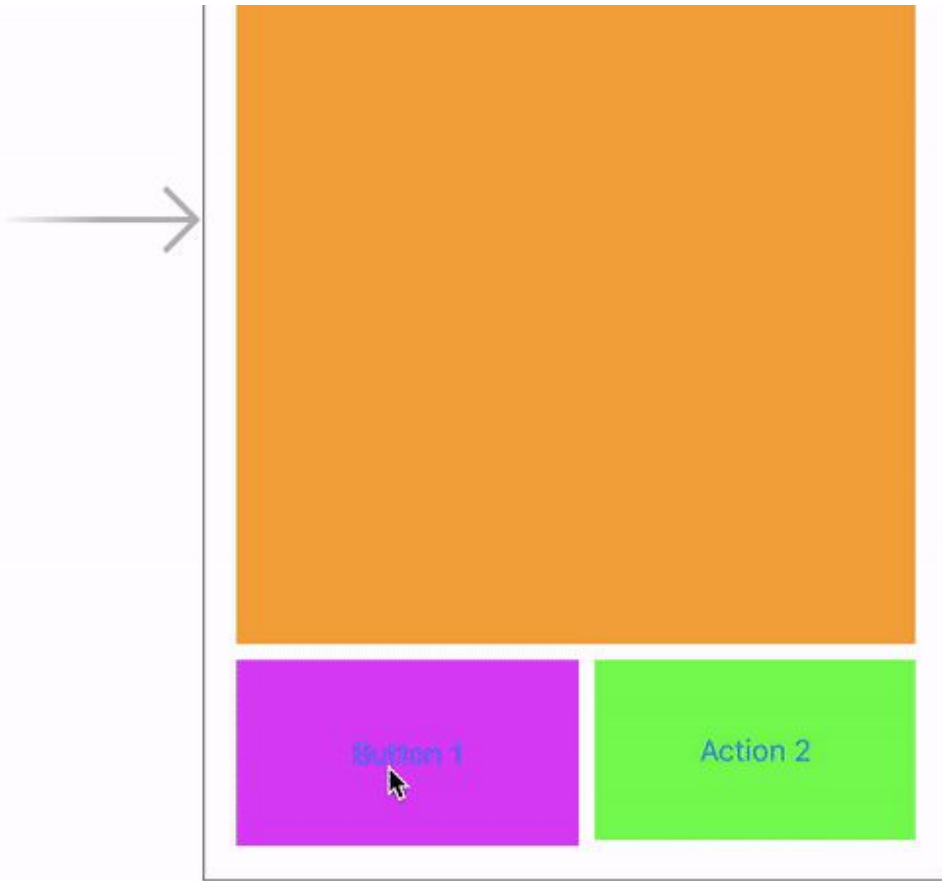
Ctrl + click inside Button 1 and drag
bottom-left corner of storyboard

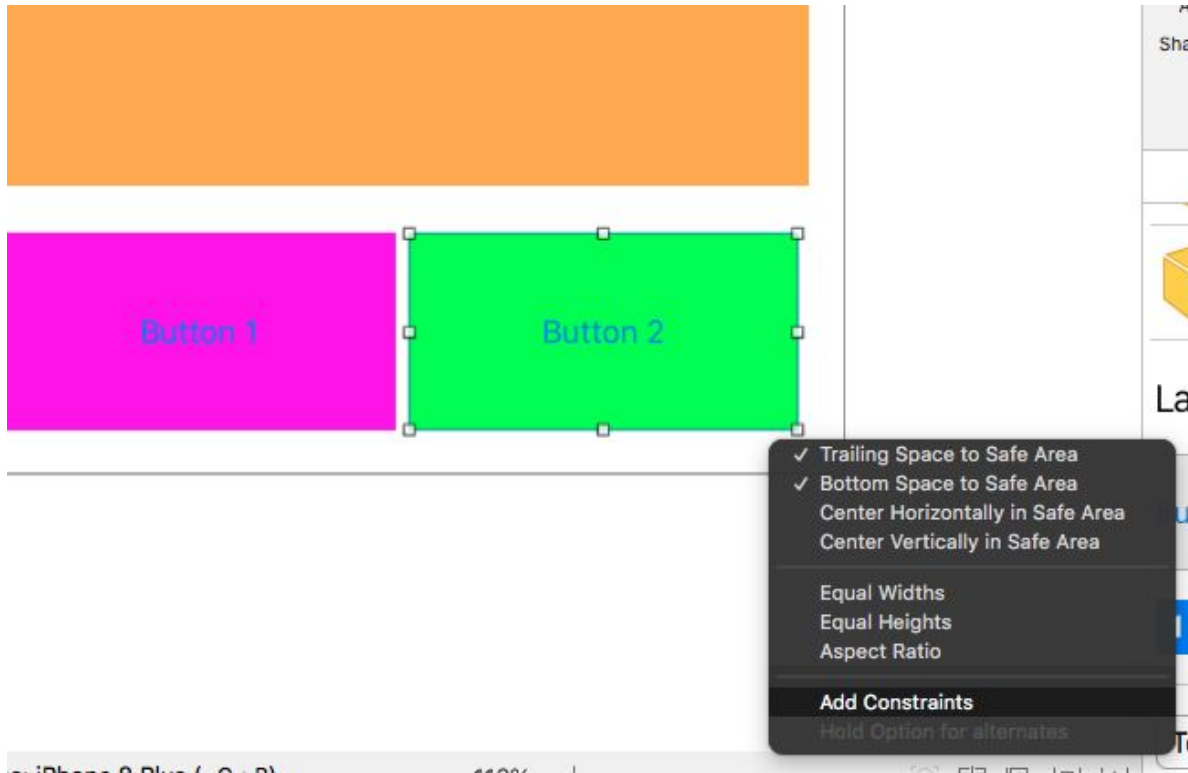
Shift + click

Leading Space to Safe Area and
Bottom Space to Safe Area;
click **Add Constraints**

This may automatically re-scale the
width of the button;

DON'T PANIC - you will fix it later





Pinning Button 2

Same as before in right corner;

Ctrl + click inside Button 2 and drag to bottom-right corner of storyboard

Shift + click

Leading Space to Safe Area
and

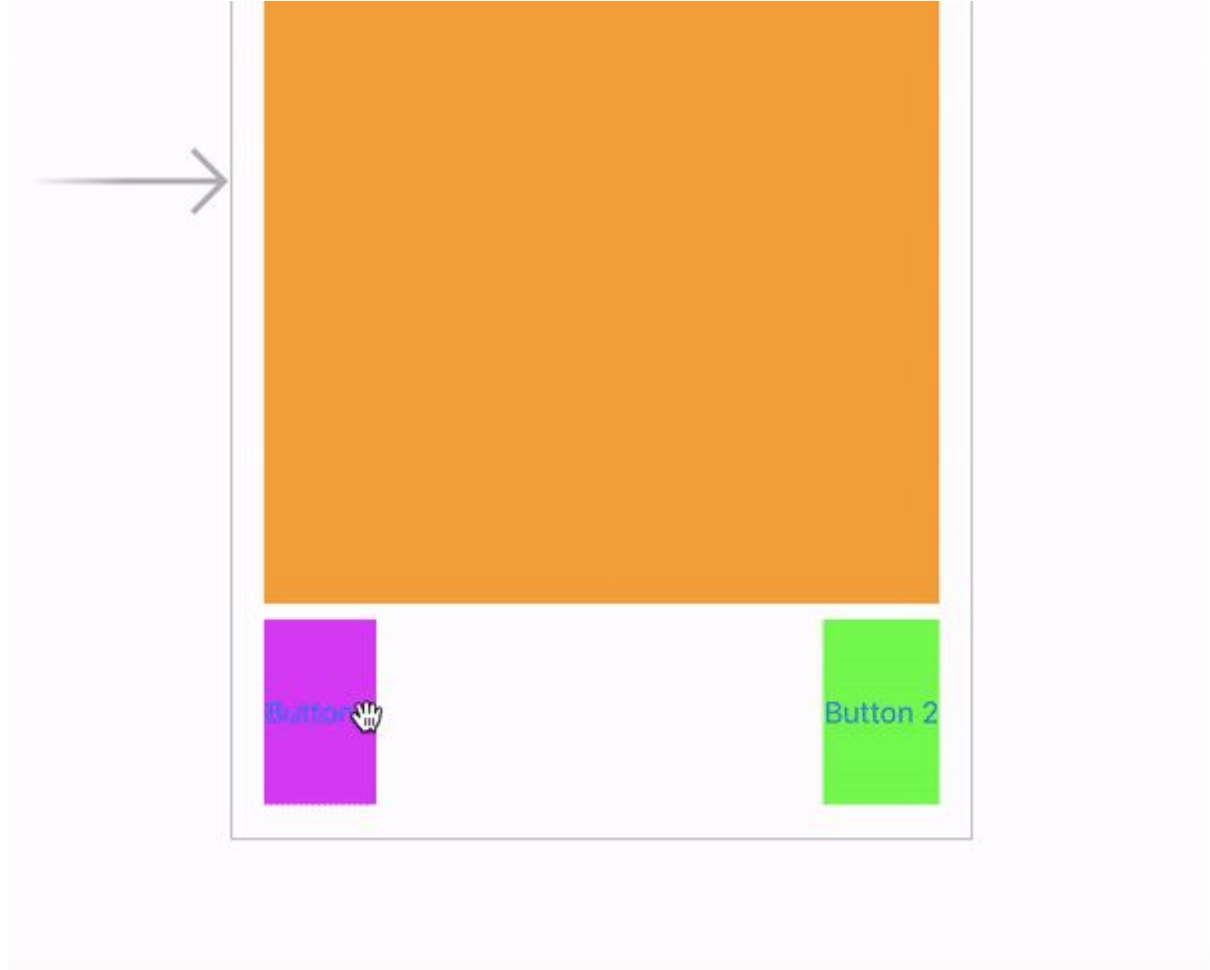
Bottom Space to Safe Area;
click **Add Constraints**

Horizontal Distance

Resize Button 1's right edge to
Button 2's left

Ctrl-click Button 1, drag to
Button 2, and apply "Horizontal
Distance"

This will set the margin
between the two buttons to a
constant size

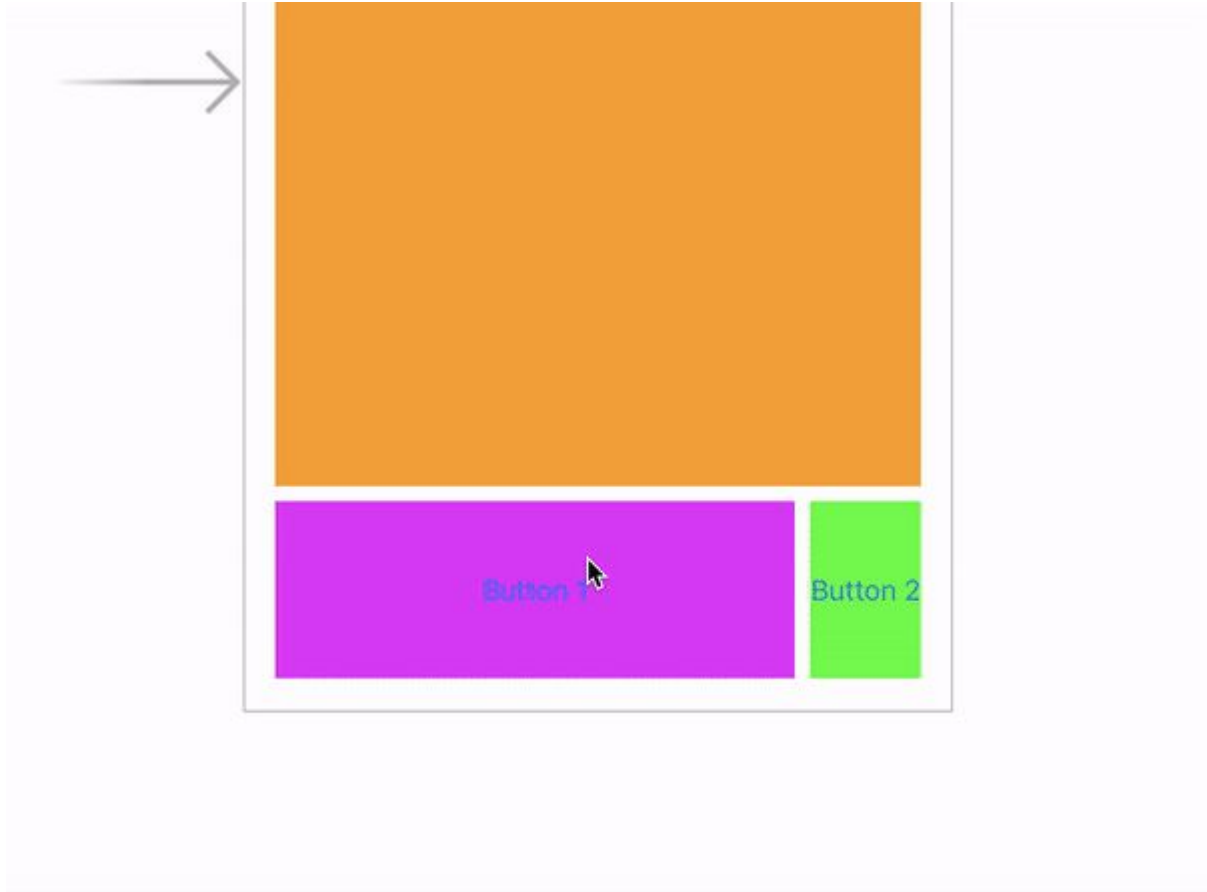


Equal Widths

Ctrl + click Button 1, drag to Button 2, and apply "Equal Widths"

XCode will resize the buttons to have

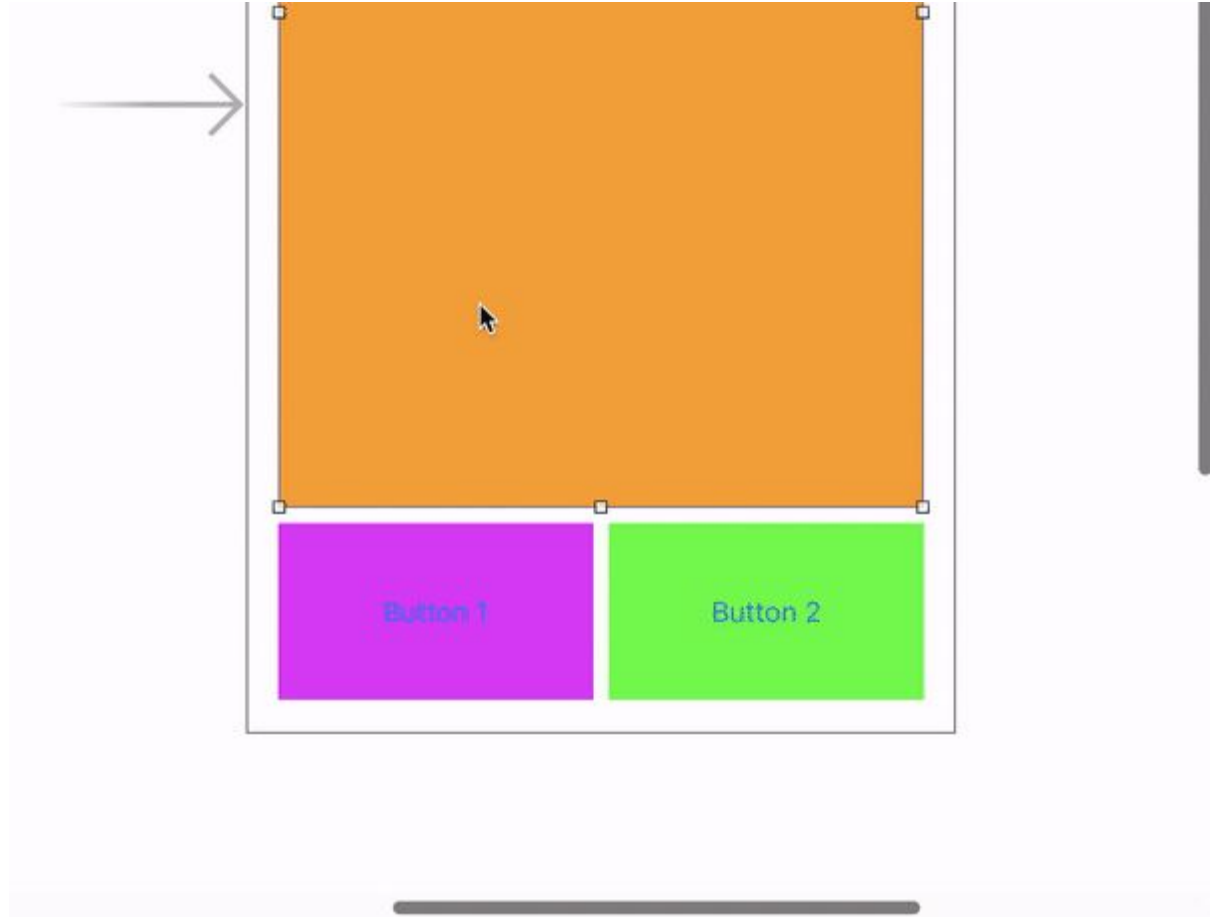
Should suppress any warning
XCode is throwing



View to Button

Ctrl-click View and drag to Button 1, and apply "Vertical Spacing"

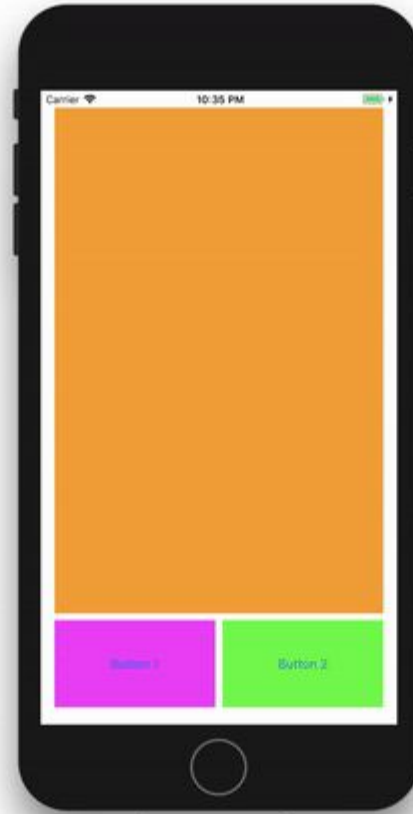
This will fix the bottom of the View to the top of the Button



Try it Out!

The elements should properly fit the screen in both orientations

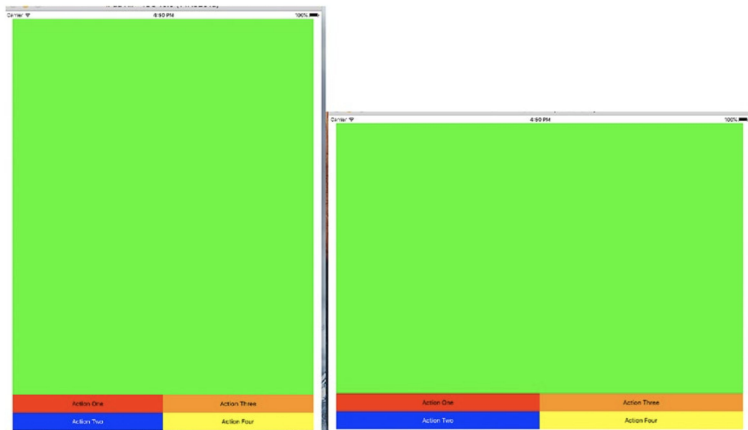
The buttons' widths change depending on the orientation, while the height remains the same



iPad and iPhone device views and button layouts

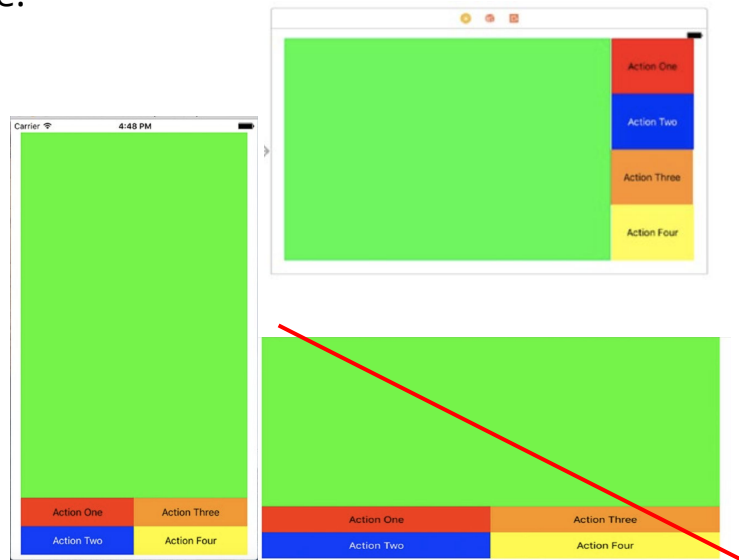
iPad

In any orientation, wR hR, you want a single row of buttons against the bottom of the view



iPhone

In landscape, still a wC hC configuration, you want a single column of buttons pressed up against the right side.



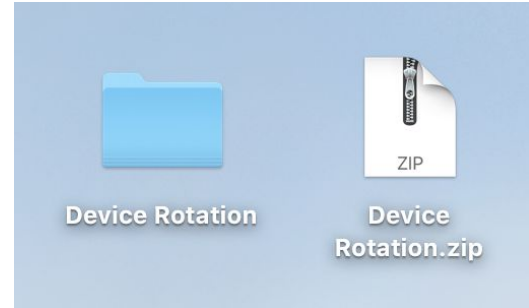
Setting the iPhone Landscape (wC hC) Configuration

First :

- Save your work
- Close your Xcode project (you can just close the window instead of quitting Xcode)
- Create a “master” copy of your project

Create a “master” copy of your project

- Find your project folder and create a compressed version (you can revert to this if you need to)
 - Ctrl-Click on the folder and select “Compress”
- Rename the .zip file “YourProjectBaseline.zip” so you know that this is the original version that works with all devices and orientations.



Create your iPhone Landscape Orientation

- Reopen your project in Xcode and select the landscape mode
- To the right of the Device Configuration Bar, click Vary for Traits
- In the pop-up, select both height and width
- Notice that all devices in the configuration bar are now in landscape mode. You'll develop your UI for this orientation only



Design your iPhone Landscape Orientation

- Delete all of your UI elements (trust me)
- Design your Landscape layout the way you would portrait, but with your buttons stacked vertically on the right side of the screen
- Click “Done Varying”
- Lastly, save this project version before moving on to the iPad and create another compressed copy, renamed as “YourProject_wChC.Zip”

Try it Out!

The buttons appear vertically on the right when in landscape mode

