



# Beginning iOS 10 Development

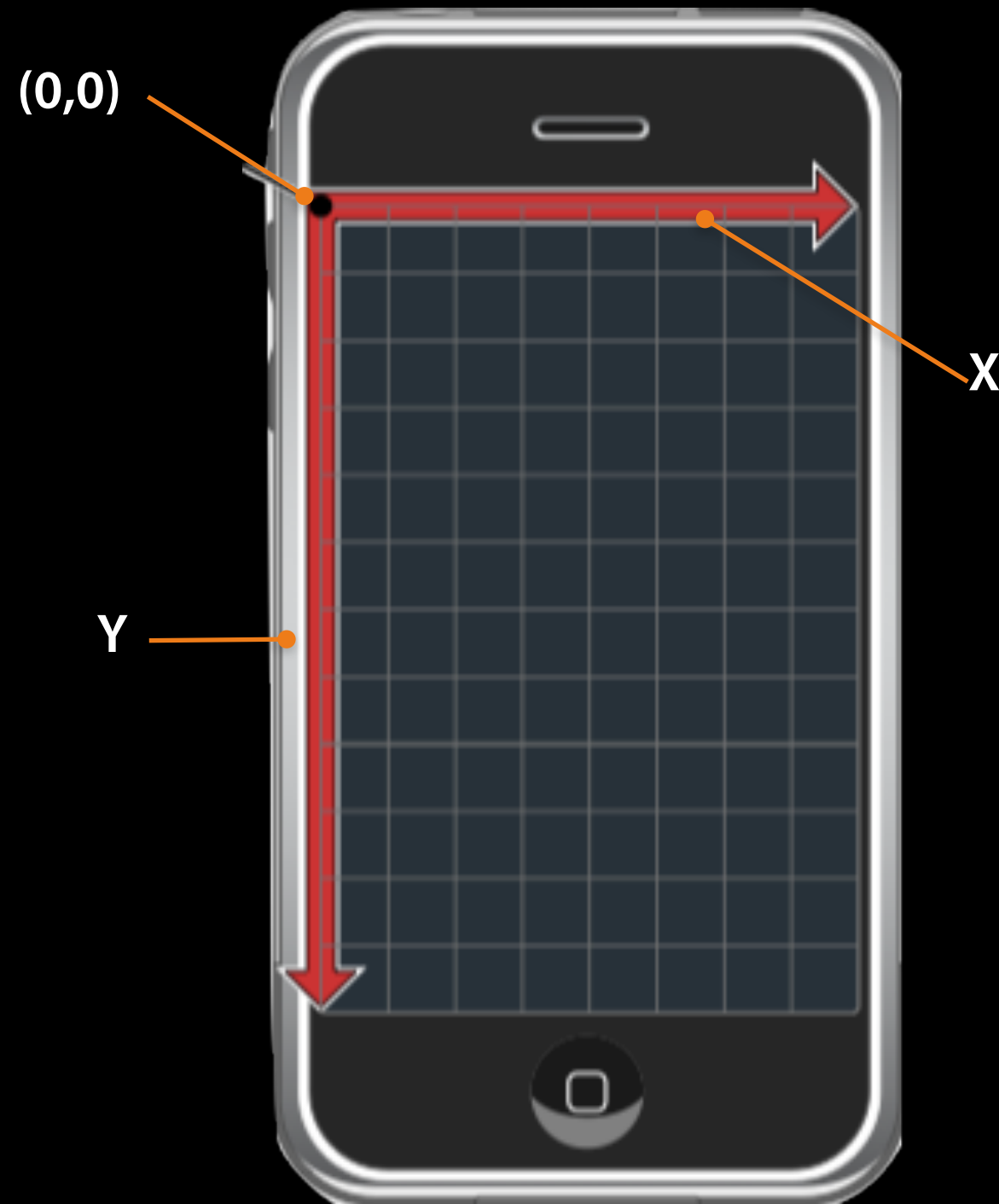
## Views

Yanping Zhao

Nov. 2016

# View Geometry

# UIKit Coordinates



# View Geometry



## frame

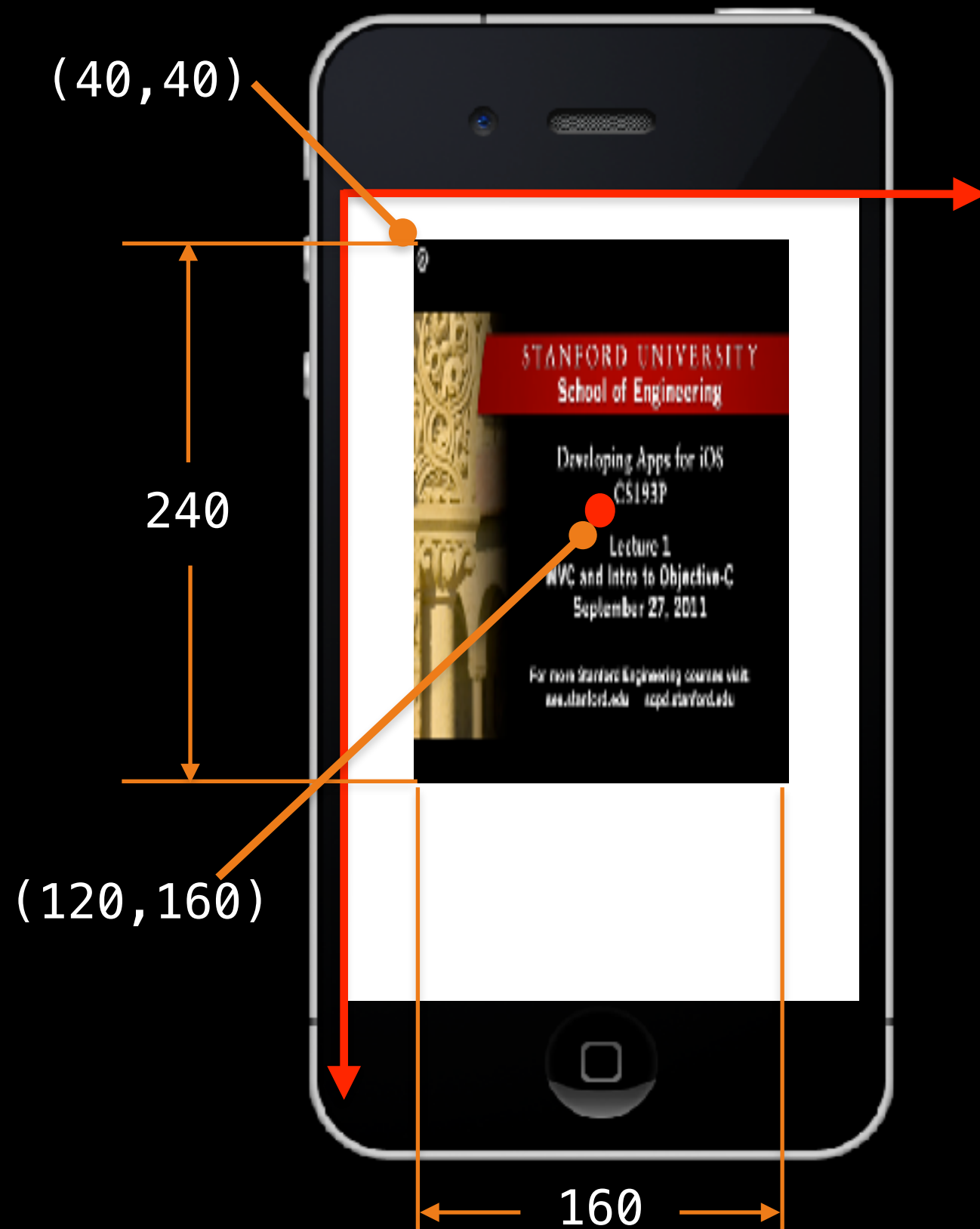
the size and location of the view in its  
superview's coordinate system

CGRect frame;

- CGPoint origin
- CGSize size

```
imageView.frame.origin.x  
imageView.frame.origin.y  
imageView.frame.size.width  
imageView.frame.size.height
```

# View Geometry



## frame

the size and location of the view in its superview's coordinate system

CGRect frame;

- CGPoint origin
- CGSize size

`imageView.frame.origin.x`  
`imageView.frame.origin.y`  
`imageView.frame.size.width`  
`imageView.frame.size.height`

## center

the center point of the view in its superview's coordinate system

CGPoint center;

`imageView.center.x`  
`imageView.center.y`

# The View Geometry



## bound

the size of the view (and its content origin) in the view's own local coordinate system

CGRect bound;

- CGPoint origin
- CGSize size

```
imageView.bound.origin.x  
imageView.bound.origin.y  
imageView.bound.size.width  
imageView.bound.size.height
```

# Points vs. Pixels

- All iOS devices are measured in **pixels**
- All coordinate systems are specified in **points**
- iOS does the mapping automatically

Device	Points vs. Pixels
non-Retina Display	1 point = 1 pixel
Retina Display	1 point = 2 pixel
HD Display	1 point = 3 pixels

# Points vs. Pixels

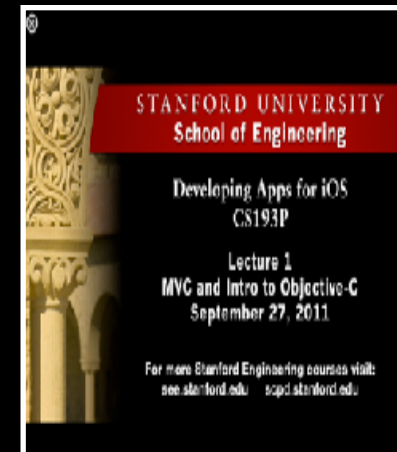
Device	Device Dimensions (pixels)	Screen Dimensions (points)
iPhone 7/6 Plus	1242x2208	414x736
iPhone 7/6	750x1334	375x667
iPhone 5	640x1136	320x568
iPhone 4	640x960	320x480



(40, 40)



## Non-Retina Display



240x160  
cover.png

## Retina Display



480x320  
cover@2x.png

# Content Modes

- The **contentMode** property -- how the view recycles its content in response to the changes in its geometry
  - Non-distorting
  - Distorting



# Content Modes

- The **contentMode** property -- how the view recycles its content in response to the changes in its geometry
  - Non-distorting
  - Distorting



Non-distorting



**ViewContentModeLeft**

# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting



# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting



**Non-distorting**



**ViewContentModeAspectFit**

# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting



# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting



Non-distorting



**ViewContentModeAspectFill**

# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting





# Content Modes

- how the view recycles its content in response to changes in its geometry
  - Non-distorting
  - Distorting



Distorting



**ViewContentModeScaleToFill**

**Auto Layout**

# Auto-layout

- A constraint-based descriptive layout system
- Create UIs adaptive to changes in screen size and device orientation
- View frames (sizes and positions) are calculated automatically
- Each view needs four properties (two in each dimension)

# Constraints

- Geometric properties of views

button1

`button1.width = 30; button1.height = 15;`

- Geometric relationships between views

button1

`button1.width = button2.width;`

button2

Relationships have a coefficient and a constant


`button1.width = 2 * button2.width + 10;`

# Constraints

- Can be equalities and inequalities

 `button1.width >= 120;`

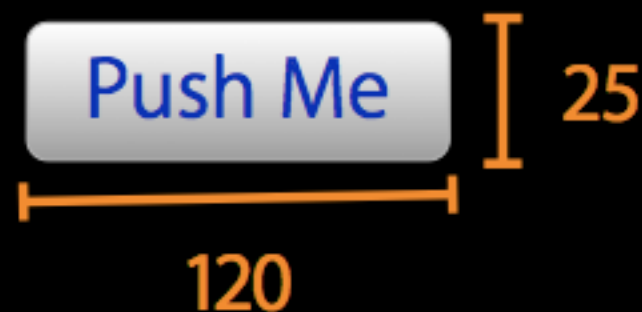
- Constraints have priorities

 `button1.width >= 60 with priority 1000`  
`button1.width = 120 with priority 750`

- `priority = 1000`: the constraint must be met
- `priority < 1000`: the constraint is optional and is satisfied in a best-effort manner

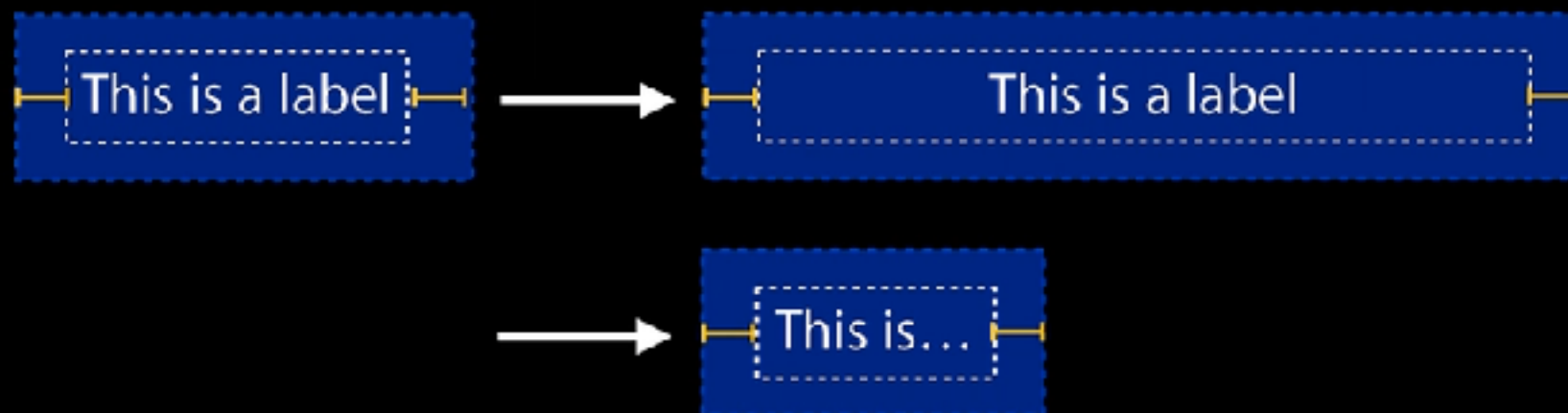
# Intrinsic Content Size

- Some views (e.g., buttons, labels) display content which naturally defines a preferred size of the view
- An intrinsic content size generates two constraints per dimension



# Intrinsic Content Size Properties

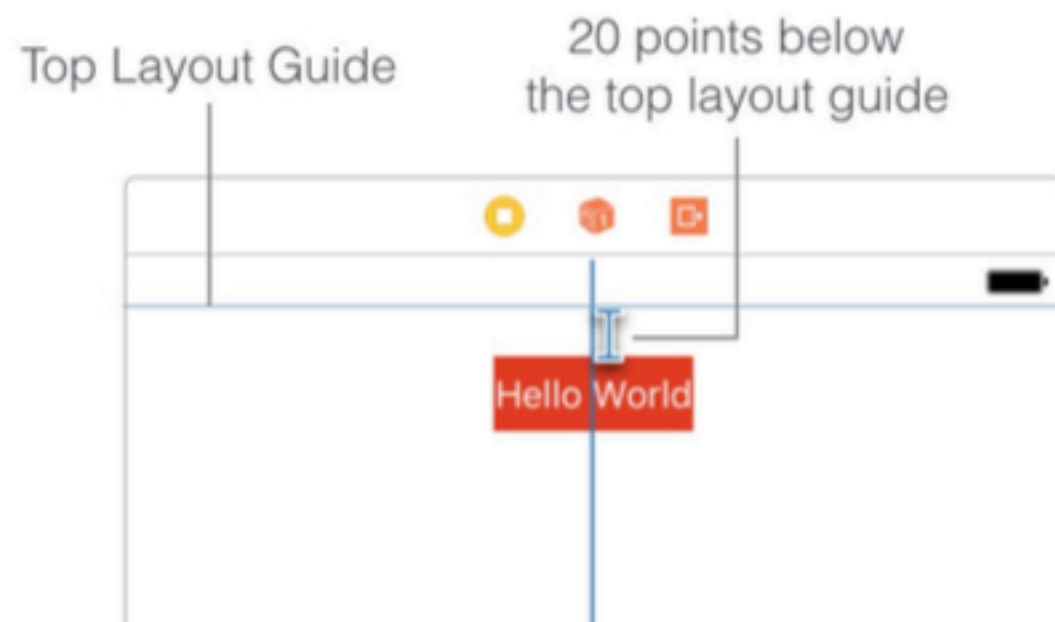
- **Content hugging priority** — how easy a control is allowed to grow
- **Content Compression resistance priority** — How easy a control is allowed to be squished



## Default

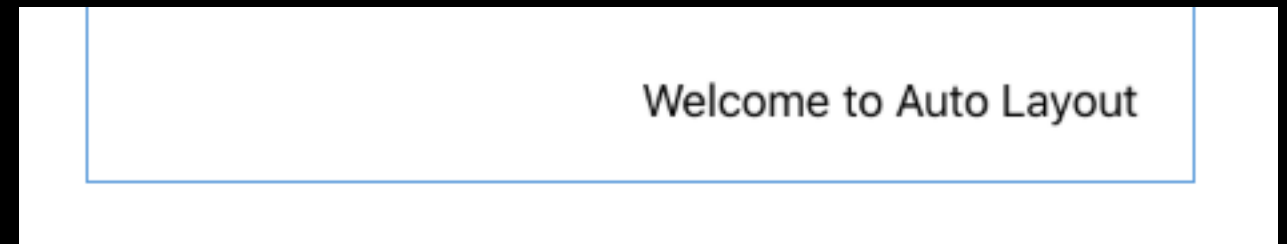
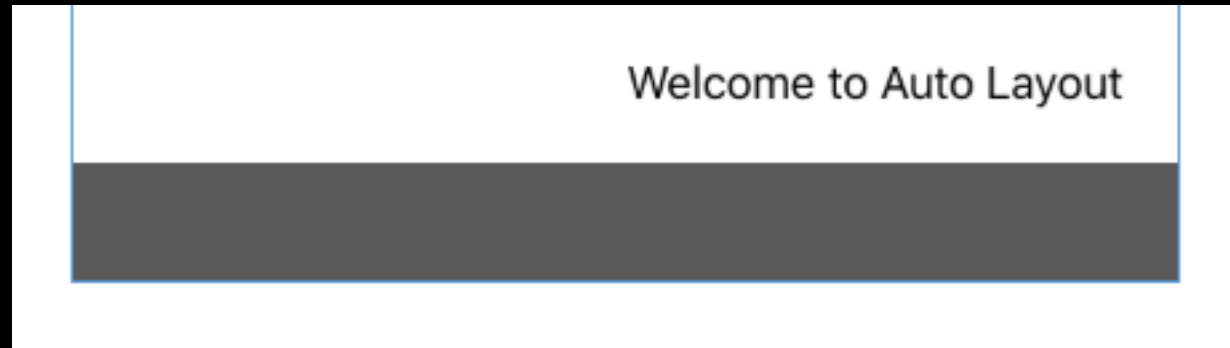
- Low content hugging priority (250) — allow self to grow
- High compression resistance priority (750) — not allow self to shrink

# Top Layout Guide



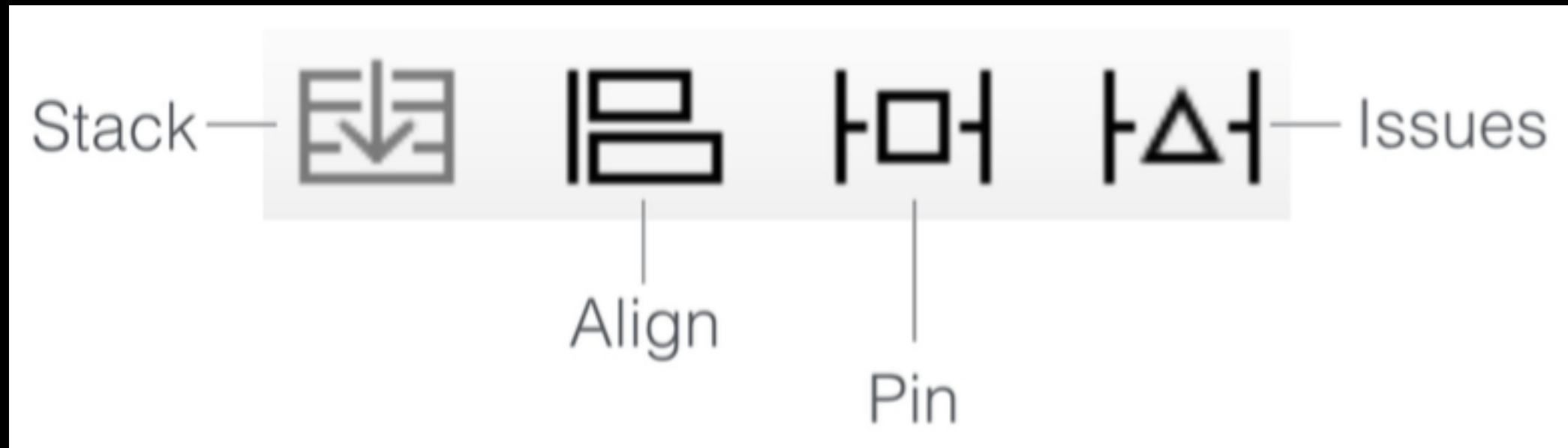


# Bottom Layout Guide



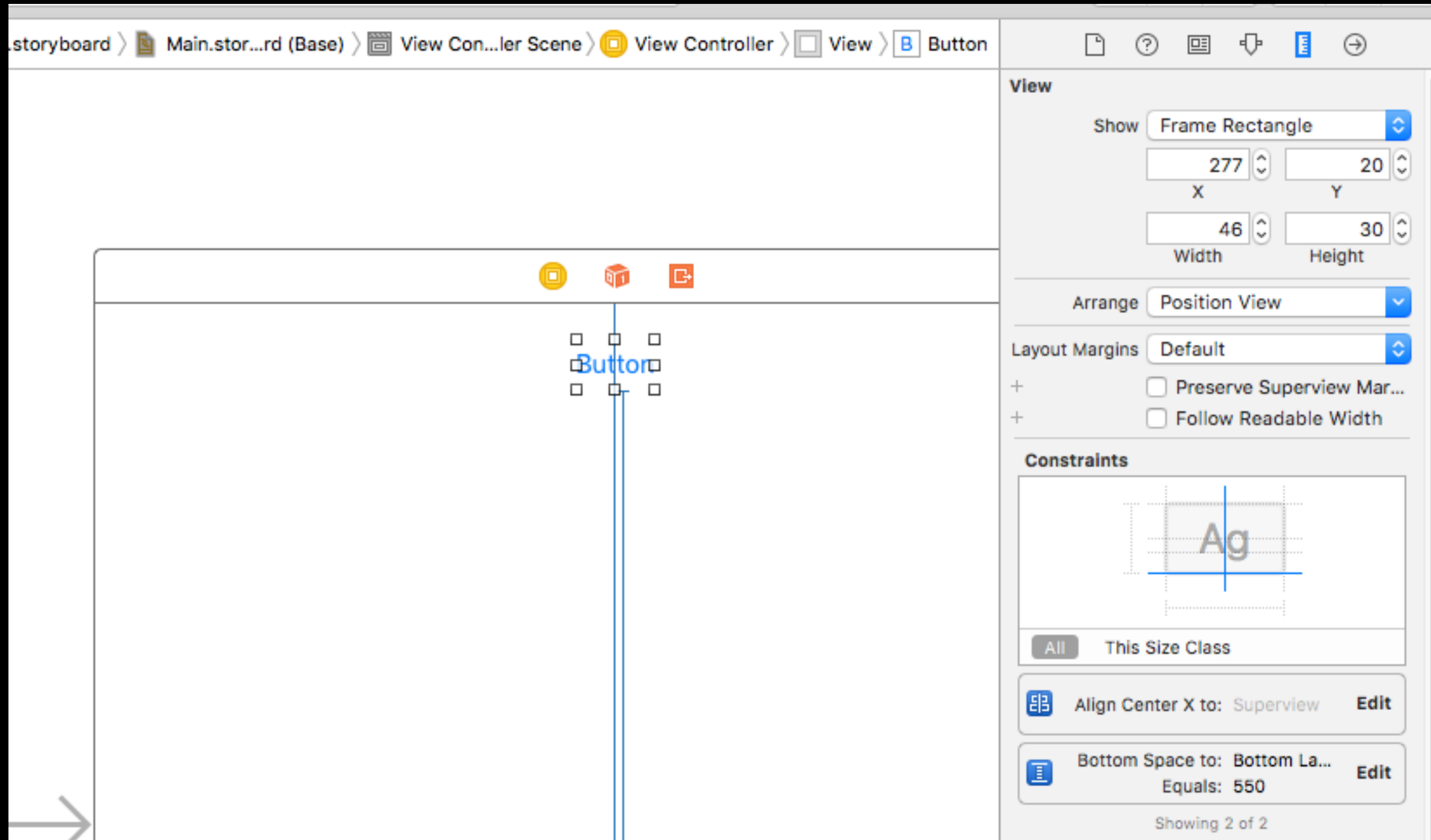
# Define Constraints in Xcode

- Auto-layout bar

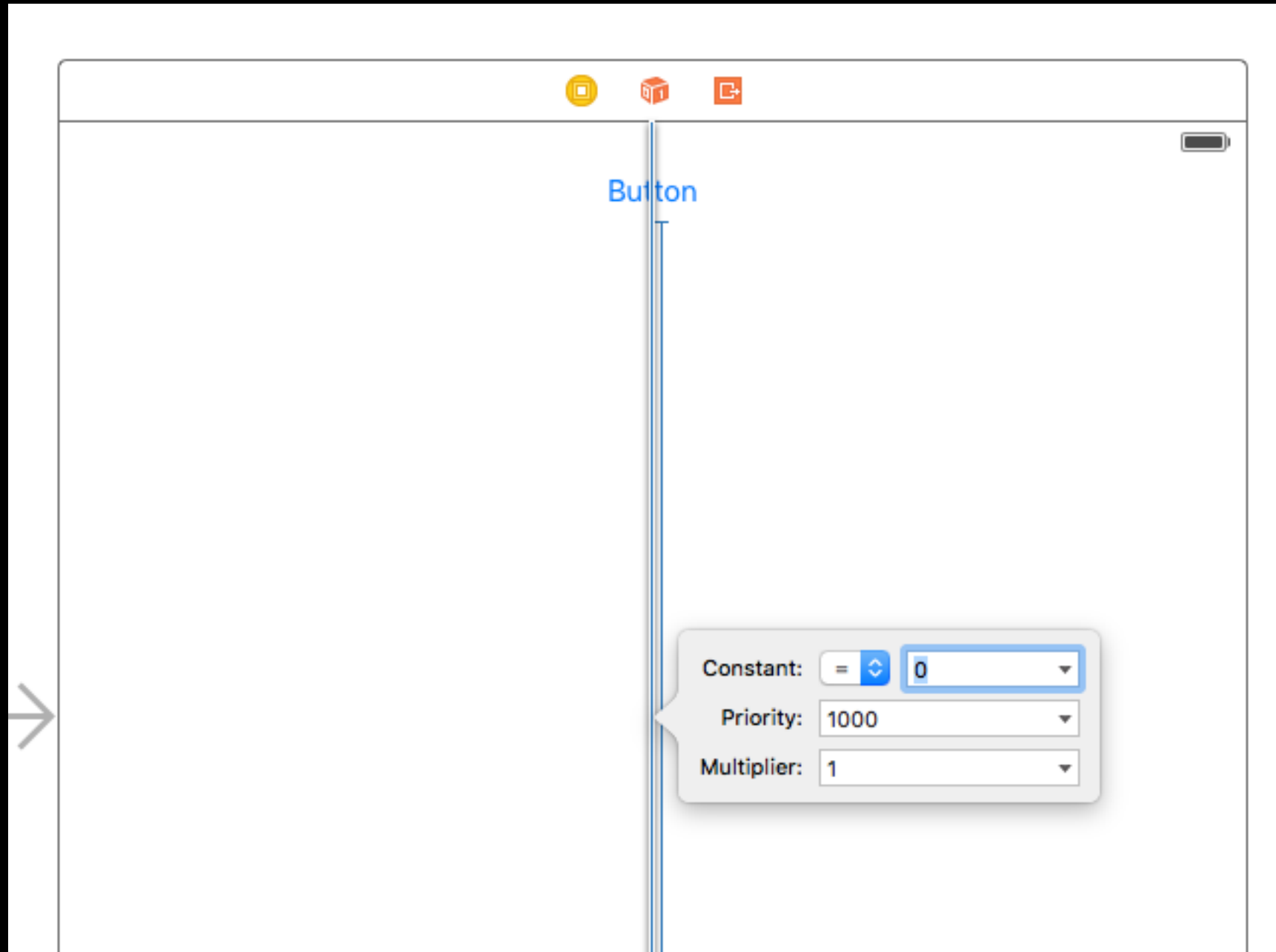


- Ctrl-drag

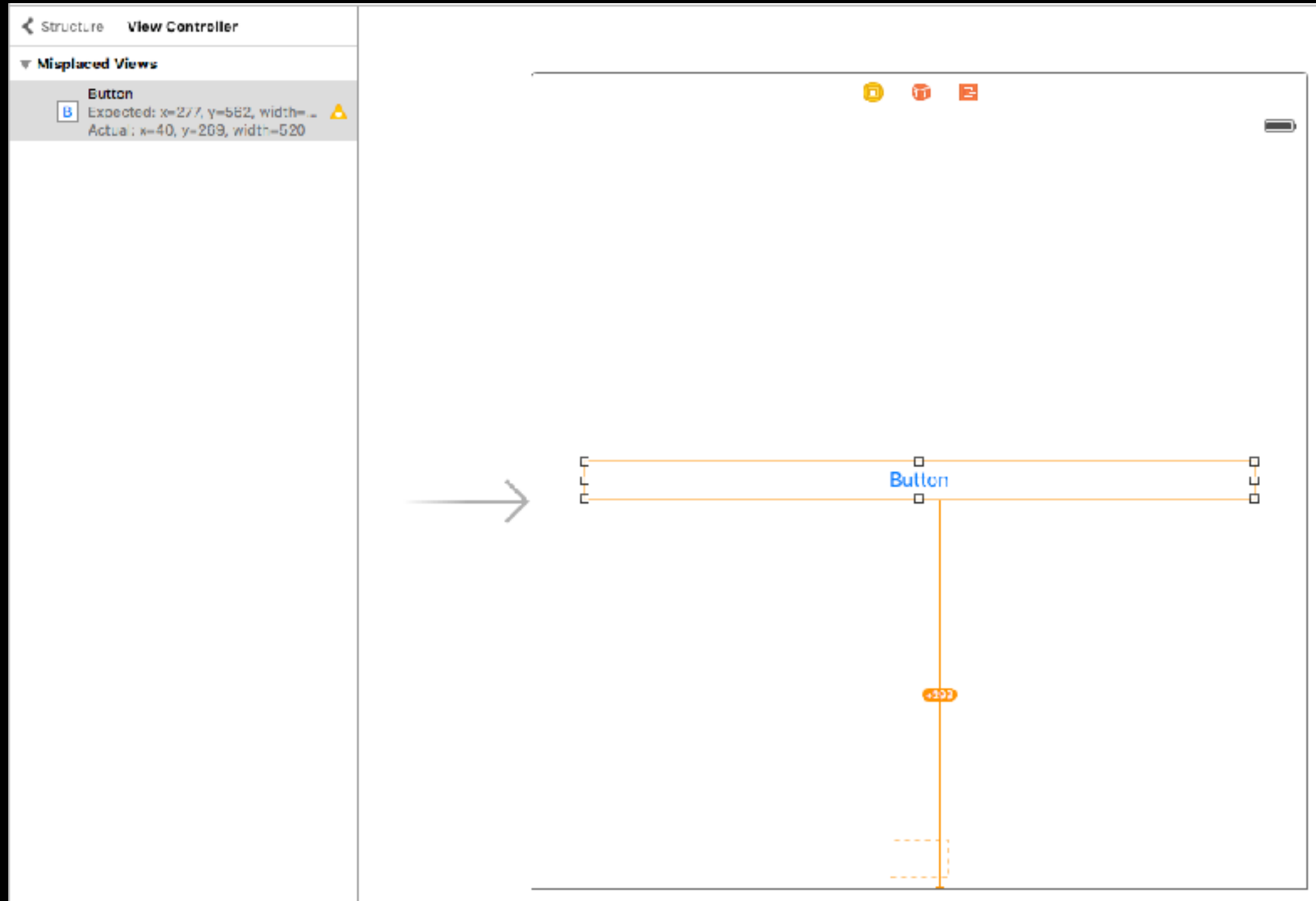
# Editing a Constraint — Attributes Inspector



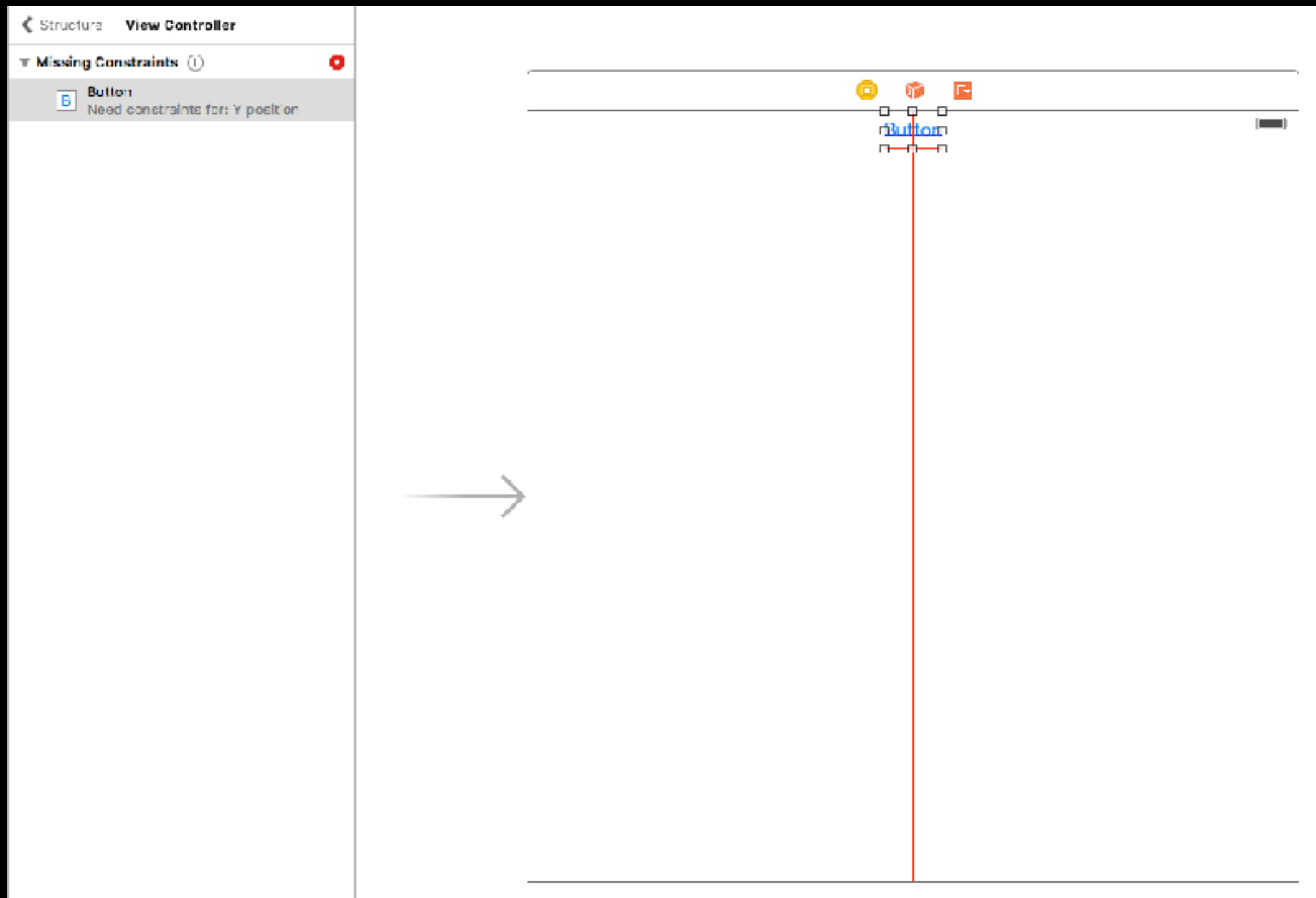
# Editing a Constraint — Constraint Pop-up



# Issues — Ambiguous Constraints

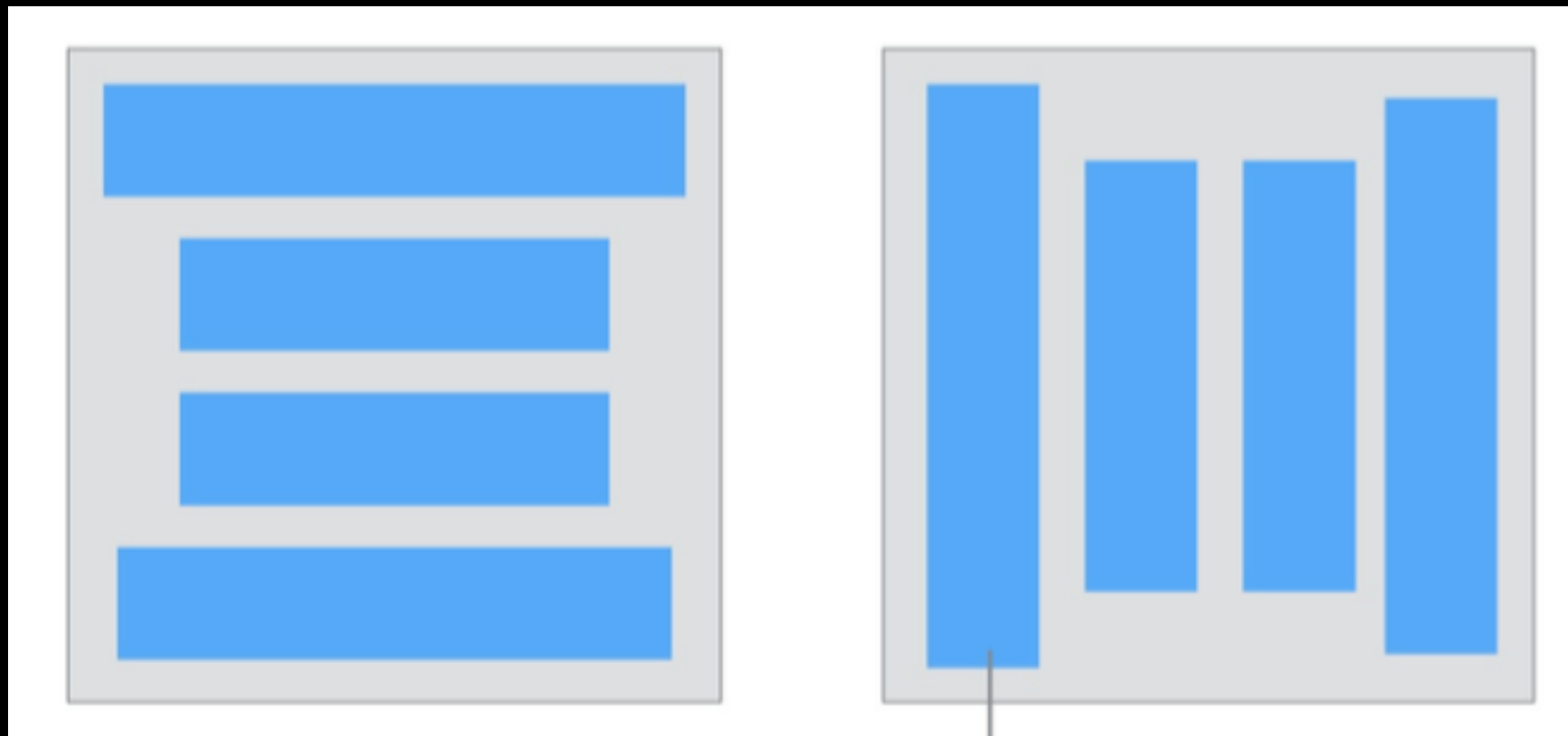


# Issues — Unsatisfactory Constraints

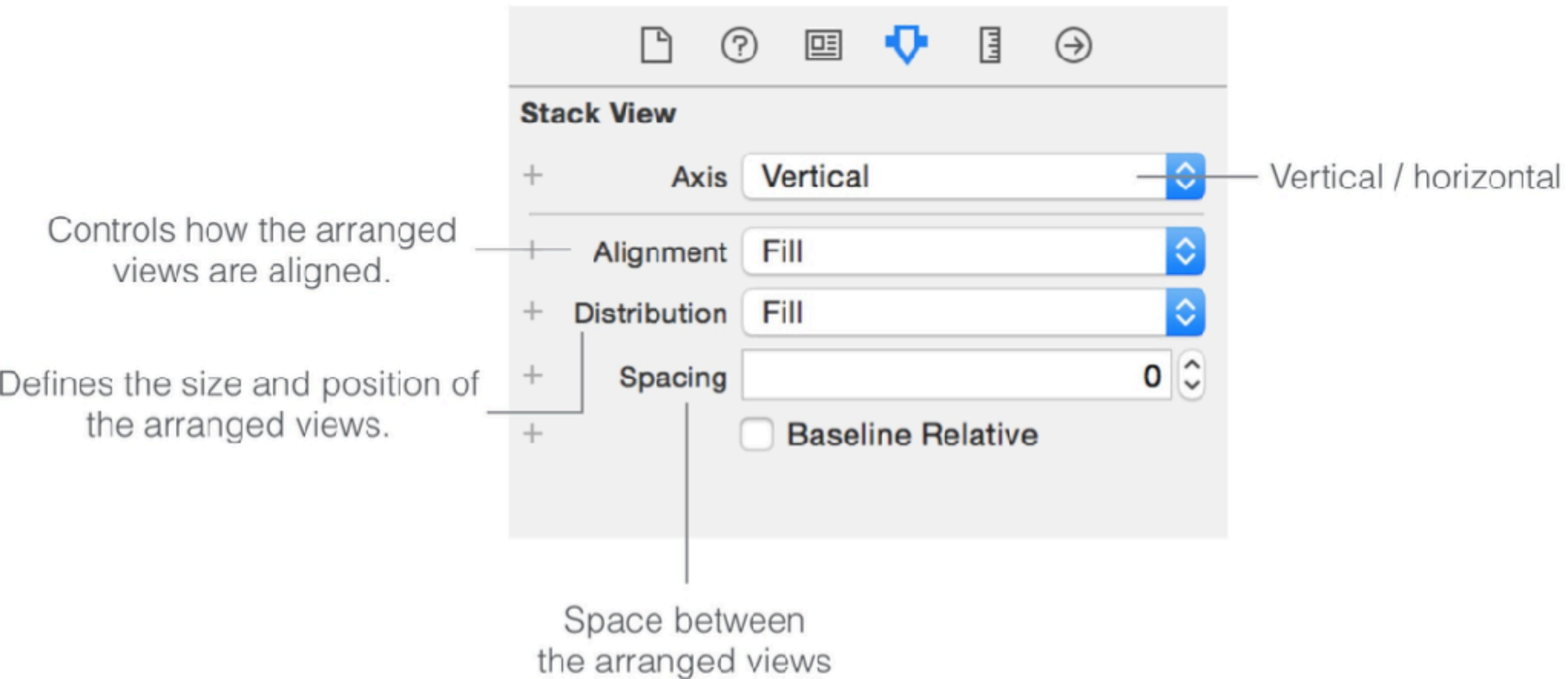


# Stack Views

- A view that groups multiple views in a column or a row
- In most cases, the views inside do not need to specify constraints



# Stack Views





# Stack Views

## Alignment

Alignment	Description
<b>fill</b>	the arranged views are resized to fill the available space perpendicular to the stack view's axis
<b>leading (top)</b>	the arranged views are aligned along the leading edge.
<b>center</b>	the arranged views are aligned with their center
<b>trailing (bottom)</b>	the arranged views are aligned along the trailing edge.

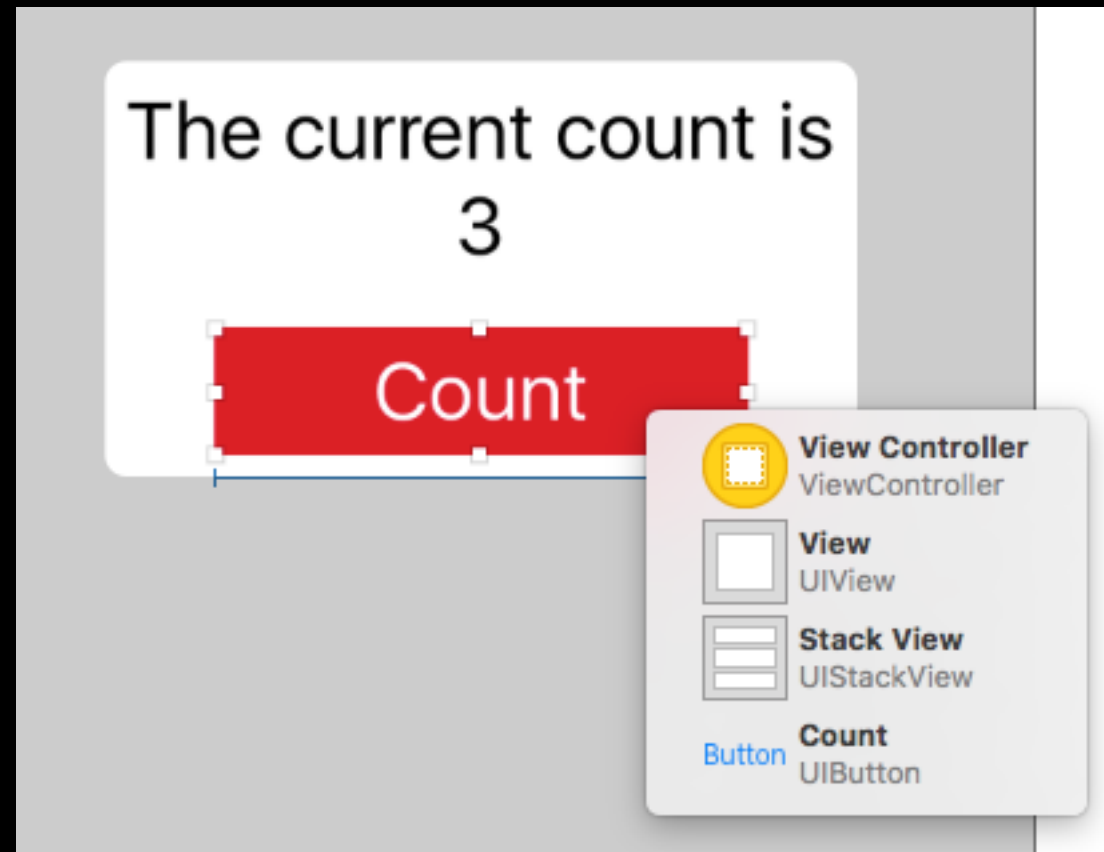
# Stack Views

Distribution — how arranged views fill the space

Alignment	Description
fill	the arranged views are resized
fill equally	the arranged views are resized to the same size
fill proportionally	the arranged views are resized based on their intrinsic content size
equal spacing	the arranged views pads the spacing between them evenly
equal centering	the arranged views have an equal center-to-center spacing along the stack view's axis

# Stack Views

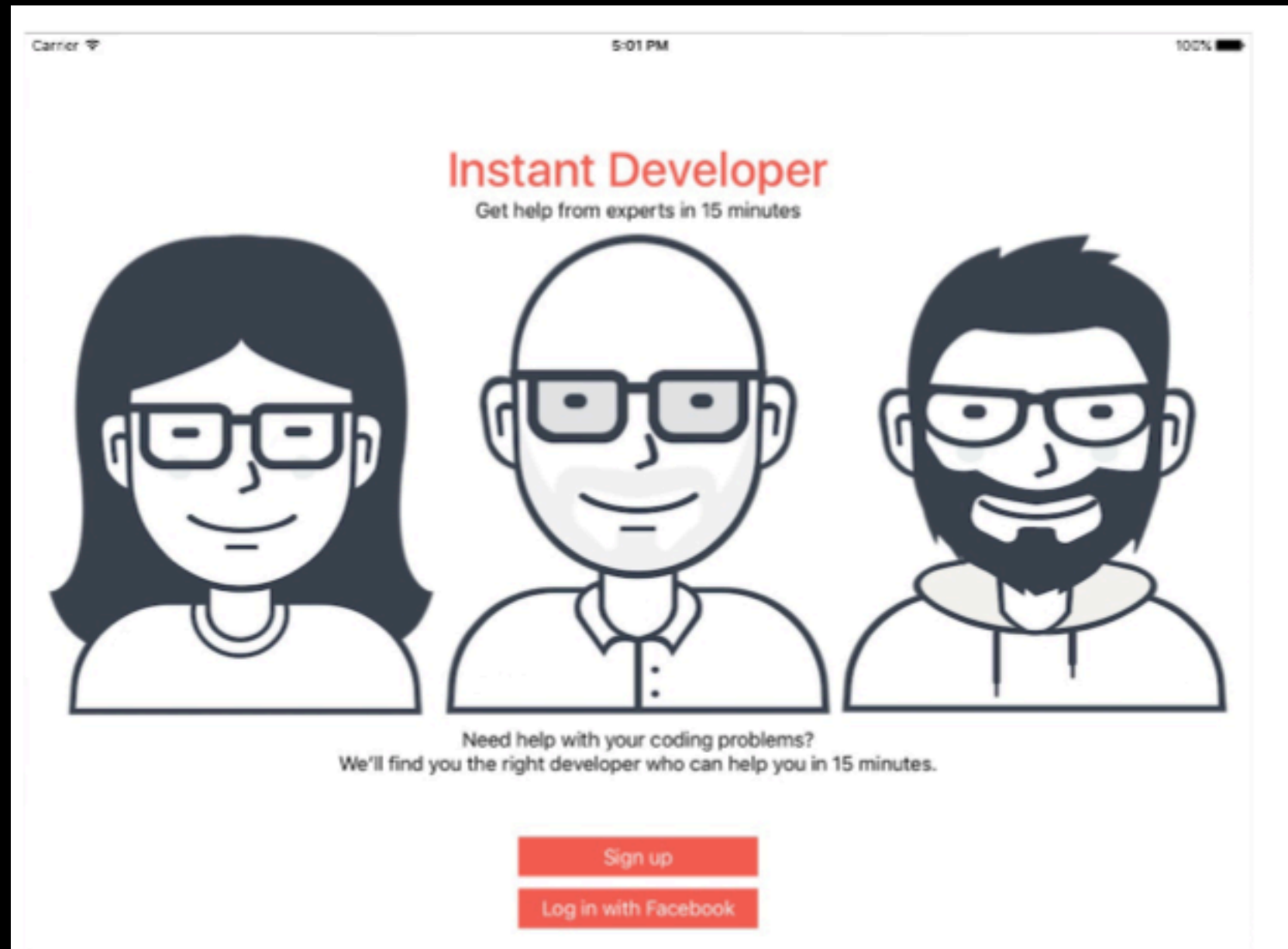
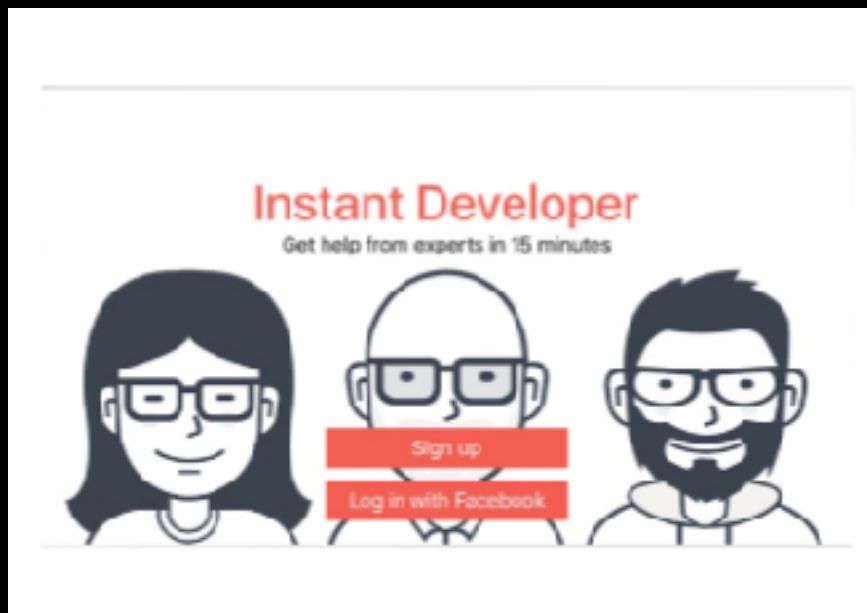
## Short-cut Menu for Selection



Hold Shift + Right Click a view

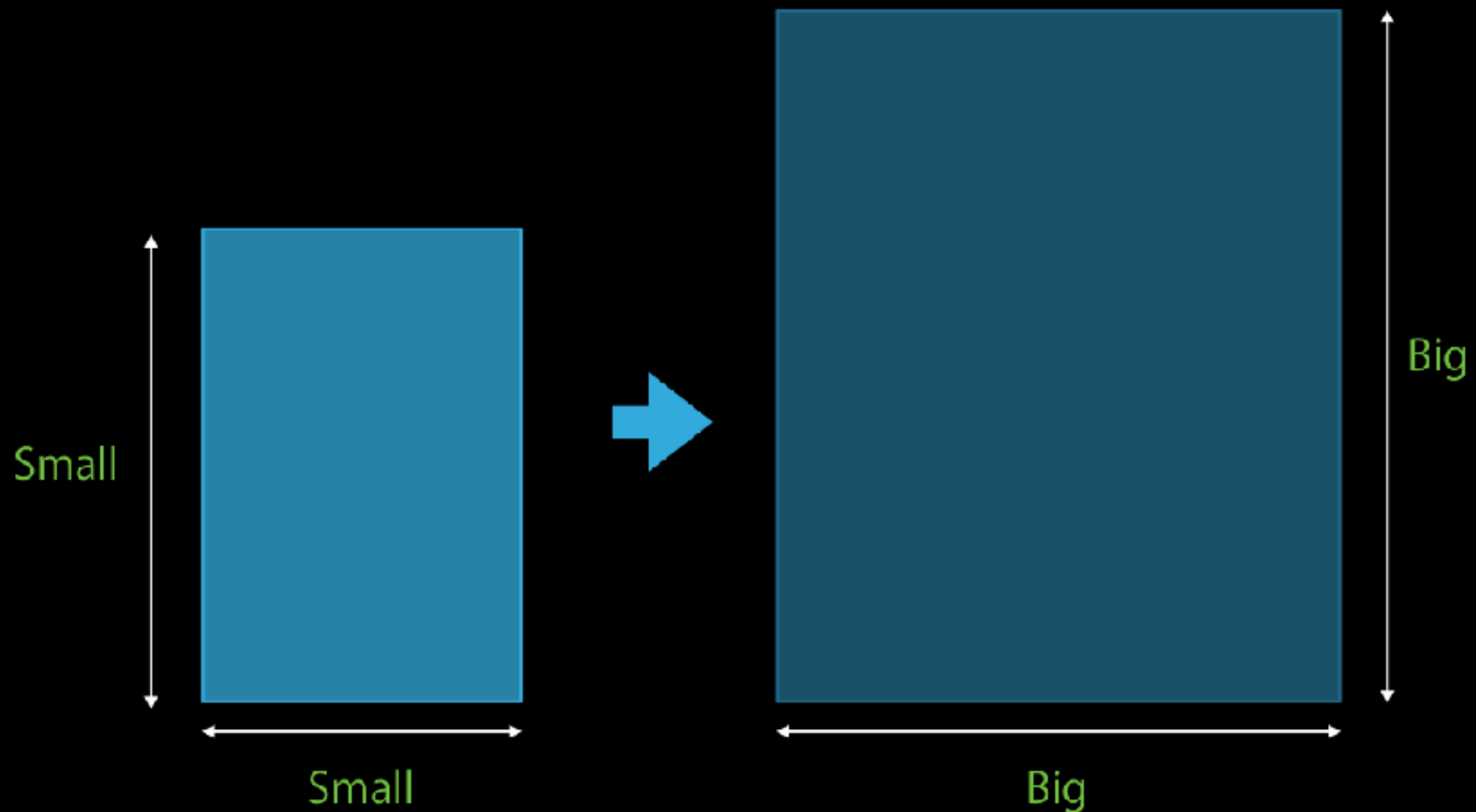
# Adaptive Layout

- An app can adapt its UI to a particular device and/or device orientation







# Size Class

- Coarsely defines the display space available in horizontal and vertical dimensions



# Size Classes

		Horizontal	
Vertical		Regular	Compact
	Regular	 <p>iPad (Portrait &amp; Landscape)</p>	 <p>iPhone Portrait</p>
	Compact	 <p>iPhone 6/7 Plus Landscape</p>	 <p>iPhone 4/5/6 Landscape</p>

# Size Class

Define different values

50

Vertical Space Constraint

First Item Welcome to...to Layout.Top

Relation Equal

Second Item Stack View.Bottom

+ Constant 50

× hC -50

Priority 1000

Multiplier 1

Identifier Identifier

Placeholder ☐ Remove at build time

+ ☒ Installed

# Size Class

Install a constraint/view for certain size class

