Auto Layout with Xcode 5

Chu Si Nguyen - IVC Sec7.1 - May 2014
Collect from WWDC 2012-2013

Why Auto Layout?

The Problem

- Button's frame origin is (124, 396)

Why Auto Layout?

Tree Options to fix

- Auto Layout
  - … via Interface Builder
  - … via Code
- Autosizing attribute
- Programatic

Why Auto Layout?

Why now?

- XCode 5 support much better than ever
- IPhone 6 with multiple screen size

Why Auto Layout?

Base concept

Describe the layout with constrains,
and frame are calculated
automatically

Why Auto Layout?

No more setFrame

- Button's frame origin is (124, 396)

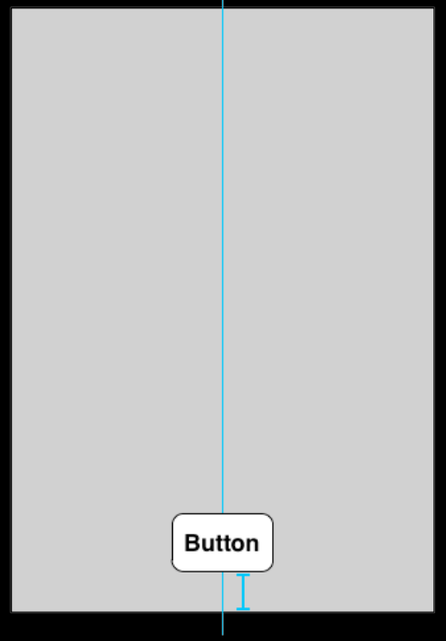
- Button is centered horizontally in its superview
- Button is a fixed distance from the bottom of the superview

- Button.centerX = Superview.centerX
- Button.bottom = Superview.bottom – <padding>

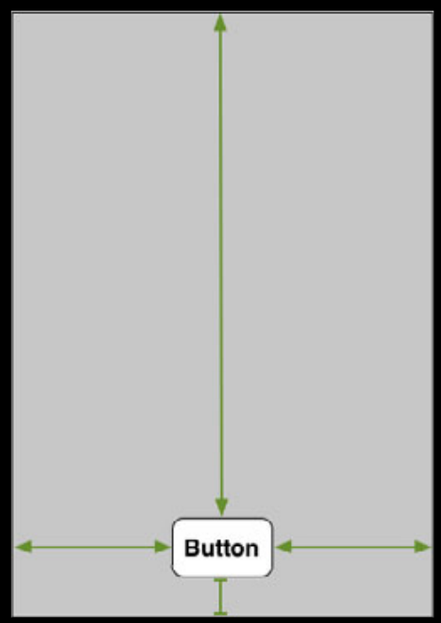
Why Auto Layout?

I can do that which Springs and Struts!

Constraints



Autoresizing Mask



Agenda

- Setting up Constraint-Based layout
- The Visual Format Language
- Easy with XCode 5
- Things can go wrong

Create you constraint

Describe Layout with constraints

item1.attribute1 = multiplier x item2.attribute2 + constant

```
+ (id)constraintWithItem:(id)item1
               attribute:(NSLayoutAttribute)attribute1
               relatedBy:(NSLayoutRelation)relation
                  toItem:(id)item2
               attribute:(NSLayoutAttribute)attribute2
              multiplier:(CGFloat)multiplier
                constant:(CGFloat)constant;
```

Create you constraint

Describe Layout with constraints

item1.attribute1 = multiplier x item2.attribute2 + constant

- Button.centerX = Superview.centerX

```
[NSLayoutConstraint constraintWithItem:button
                             attribute:NSLayoutAttributeCenterX
                             relatedBy:NSLayoutRelationEqual
                                toItem:superview
                             attribute:NSLayoutAttributeCenterX
                            multiplier:1.0
                              constant:0.0]
```

- Button.bottom = Superview.bottom – <padding>

```
[NSLayoutConstraint constraintWithItem:button
                             attribute:NSLayoutAttributeBottom
                             relatedBy:NSLayoutRelationEqual
                                toItem:superview
                             attribute:NSLayoutAttributeBottom
                            multiplier:1.0
                              constant:-padding]
```

Create you constraint

Add them to view

NSLayoutConstraint.h

AppKit

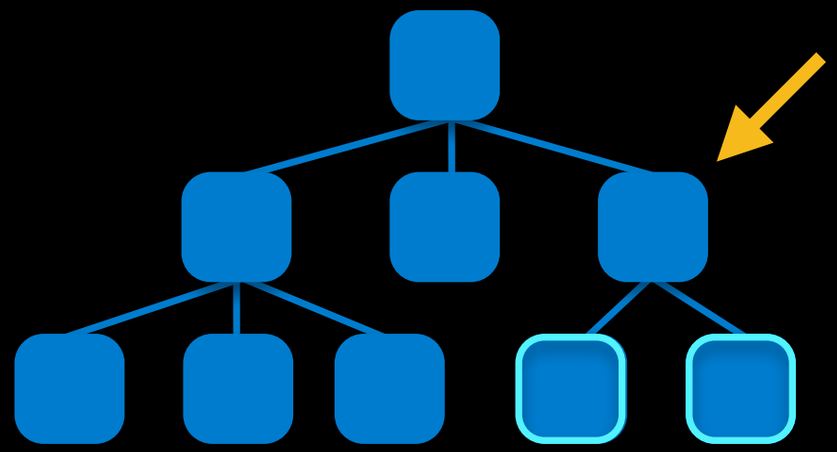
UIView.h

UIKit

```
- (void)addConstraint:(NSLayoutConstraint *)constraint;
```

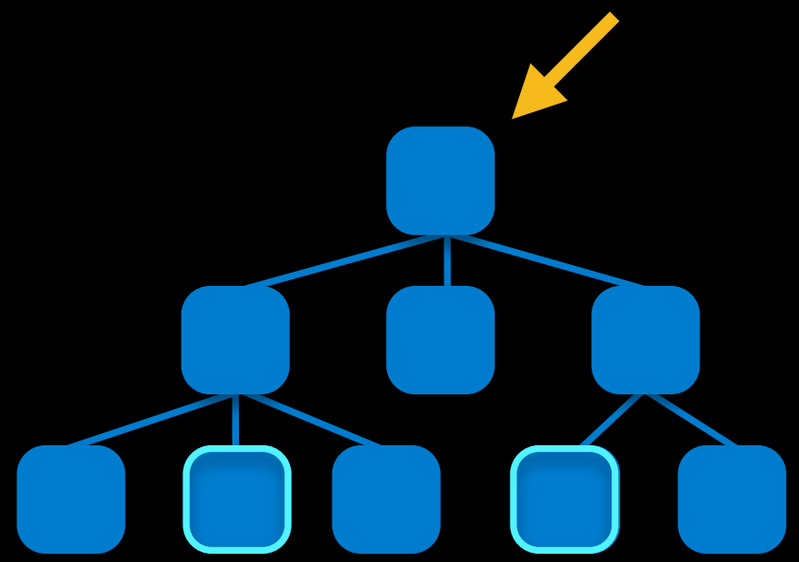
Create you constraint

Which view?



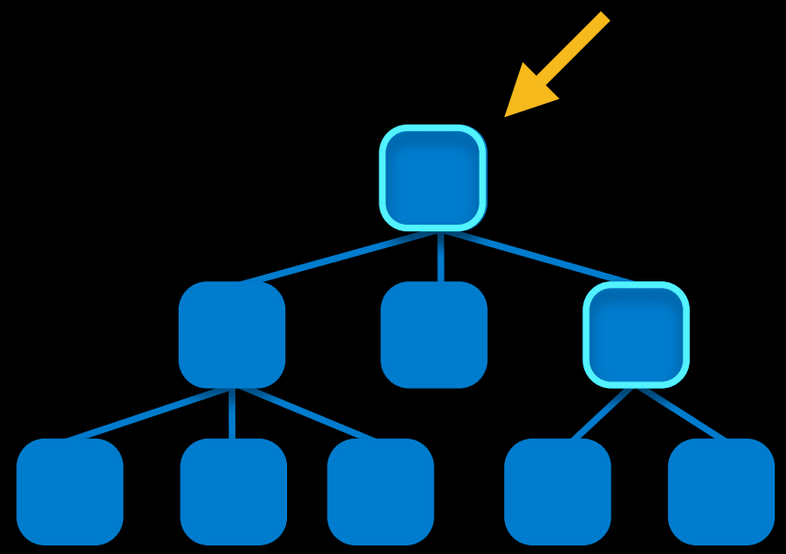
Create you constraint

Which view?



Create you constraint

Which view?



Create you constraint

Update constraint

NSView

```
-setNeedsDisplay:
-setNeedsLayout:
-setNeedsUpdateConstraints:
```

UIView

```
-setNeedsDisplay
-setNeedsLayout
-setNeedsUpdateConstraints
```

NSWindow

```
-layoutIfNeeded
```

NSView

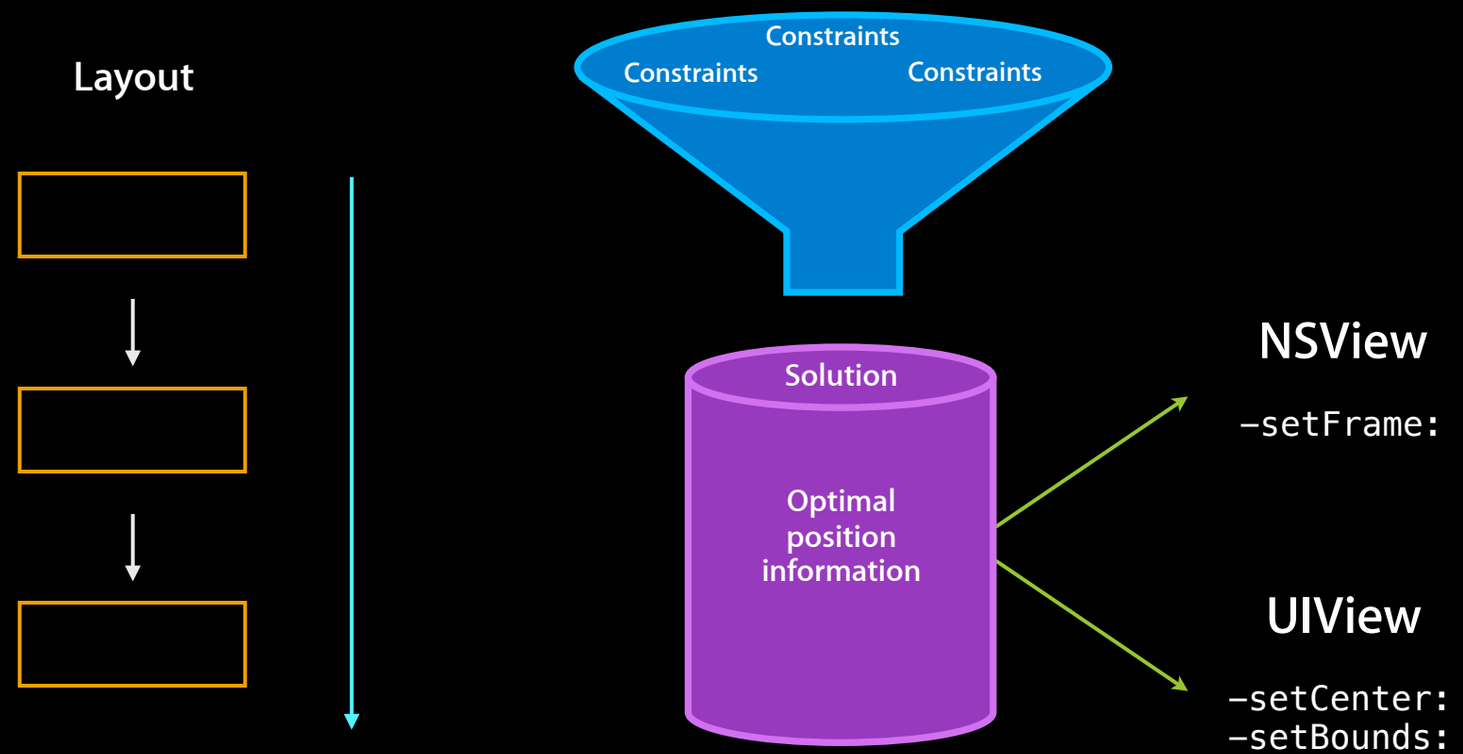
```
-layoutSubtreeIfNeeded
```

UIView/UIWindow

```
-layoutIfNeeded
```

Create you constraint

Behind the Scenes



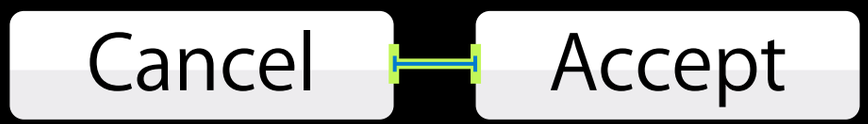
Create you constraint

Somethings should know

- Can apply to any two views, regardless of view hierarchy
- Can establish minimums and maximums with inequalities
- Can be prioritize
- Can define own frame through “intrinsicContentSize”
- Have content hugging priority
- Have content compression resistance priority

The Visual Format Language

Another way describe constraint



The Visual Format Language

Another way describe constraint

The Visual Format Language

Another way describe constraint

```
[cancelButton]-[acceptButton]
```

The Visual Format Language

Another way describe constraint

```
[NSLayoutConstraint constraintsWithVisualFormat:
        @"[cancelButton]-[acceptButton]"
options:0 metrics:nil views:viewsDictionary];
```

The Visual Format Language

Another way describe constraint

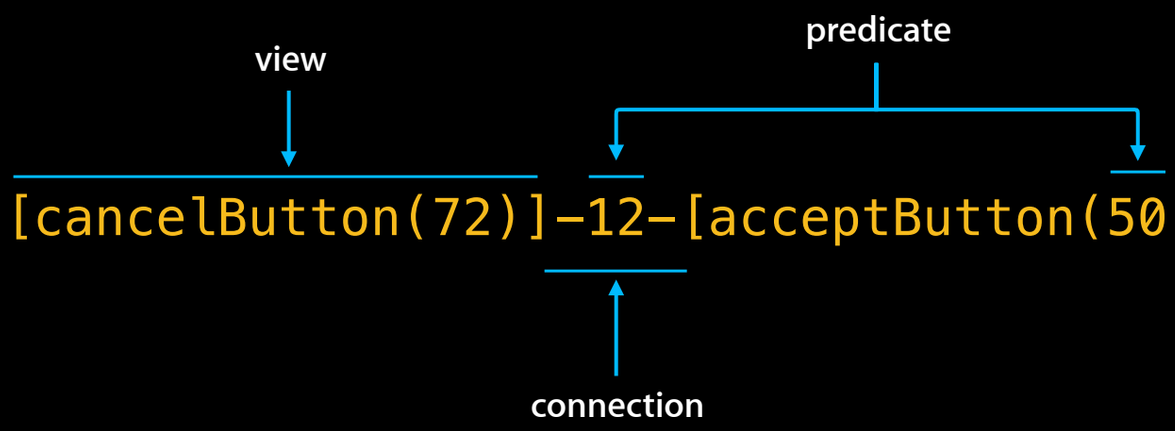
```
[NSLayoutConstraint constraintsWithVisualFormat:
      @”[cancelButton]-[acceptButton]”
 options:0 metrics:nil views:viewsDictionary];
```

```
UIButton *cancelButton = …
UIButton *acceptButton = …
viewsDictionary =
   NSDictionaryOfVariableBindings(cancelButton,acceptButton);
```

```
(lldb) po viewsDictionary
{
    acceptButton = "<UIButton: 0x4004c0>";
    cancelButton = "<UIButton: 0x4004ab>";
}
```

The Visual Format Language

Some Samples

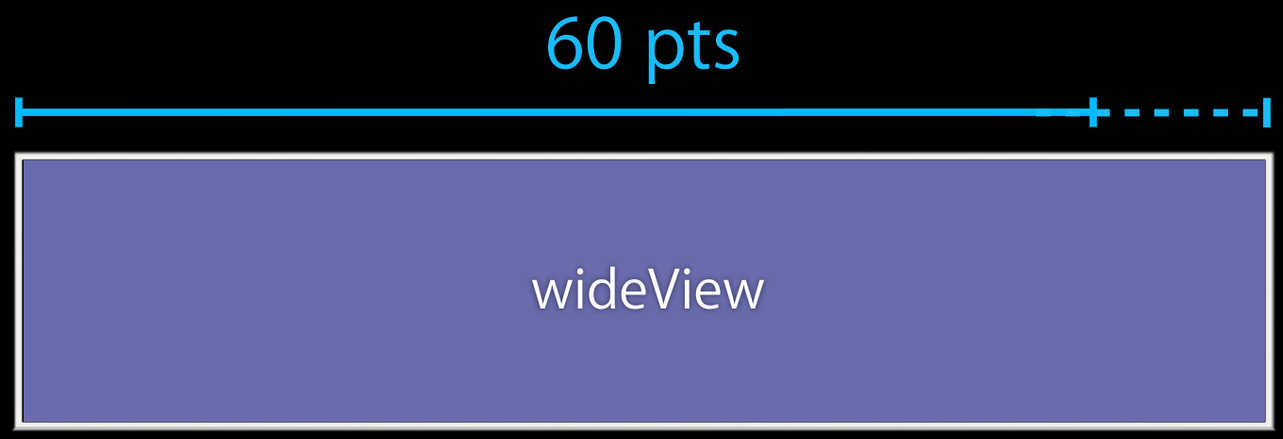


The Visual Format Language

Some Samples

Inequality, Priority	[wideView(>=60@700)]
Vertical: Flush Views, Equal Heights	V:[redBox][yellowBox(==redBox)]
Combination	H:|-[Find]-[FindNext]-[FindField(>=20)]-|

[wideView(>=60@700)]

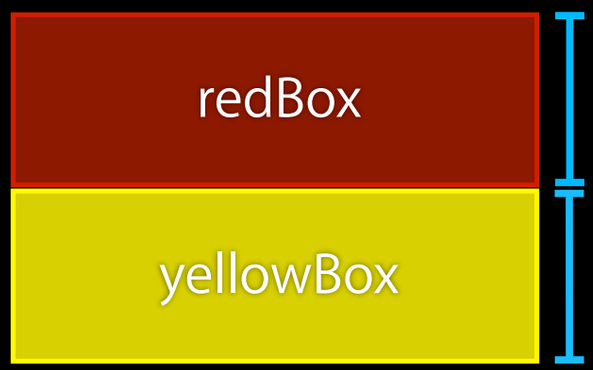


The Visual Format Language

Some Samples

Inequality, Priority	[wideView(>=60@700)]
Vertical: Flush Views, Equal Heights	V:[redBox][yellowBox(==redBox)]
Combination	H:|-[Find]-[FindNext]-[FindField(>=20)]-|

V:[redBox][yellowBox(==redBox)]



The Visual Format Language

Some Samples

Inequality, Priority	[wideView(>=60@700)]
Vertical: Flush Views, Equal Heights	V:[redBox][yellowBox(==redBox)]
Combination	H:|-[Find]-[FindNext]-[FindField(>=20)]-|

H:|-[Find]-[FindNext]-[FindField(>=20)]-|

The Visual Format Language

For more informations

https://developer.apple.com/library/ios/documentation/userexperience/conceptual/AutolayoutPG/VisualFormatLanguage/VisualFormatLanguage.html

Easy with Xcode 5

New ways

2011

2012

2013

Easy with Xcode 5

DEMO

- Life Cycle
- New way to add Constraints
- Align tool
- Pin tool
- Resolve Auto Layout issues tool
- Preview screen
- Helping function

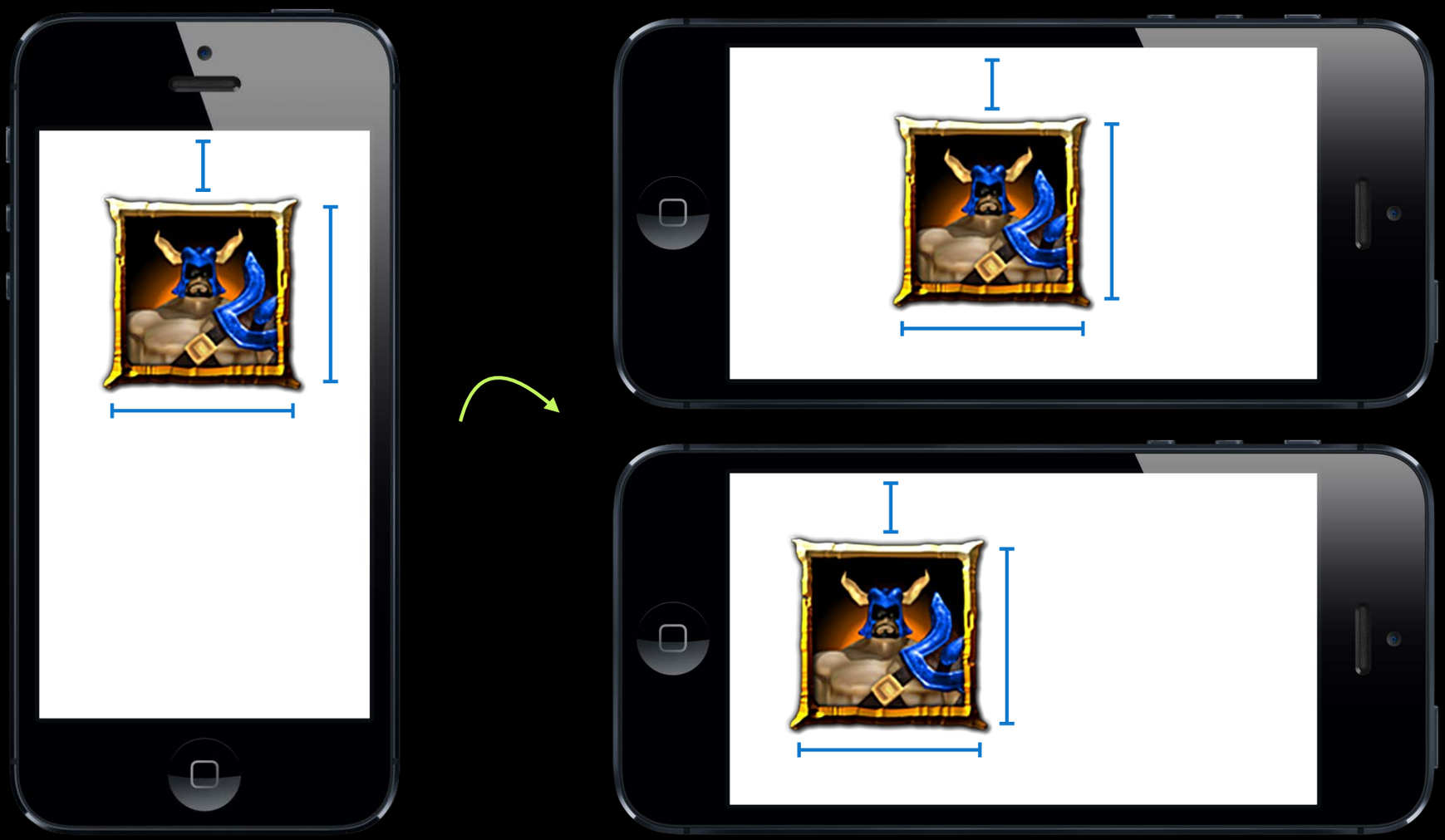
Things can go wrong

What's wrong?

- Ambiguous Frames: not enough information
- Conflicting Constraints: too much information
- Misplaced Views: mismatched position or size

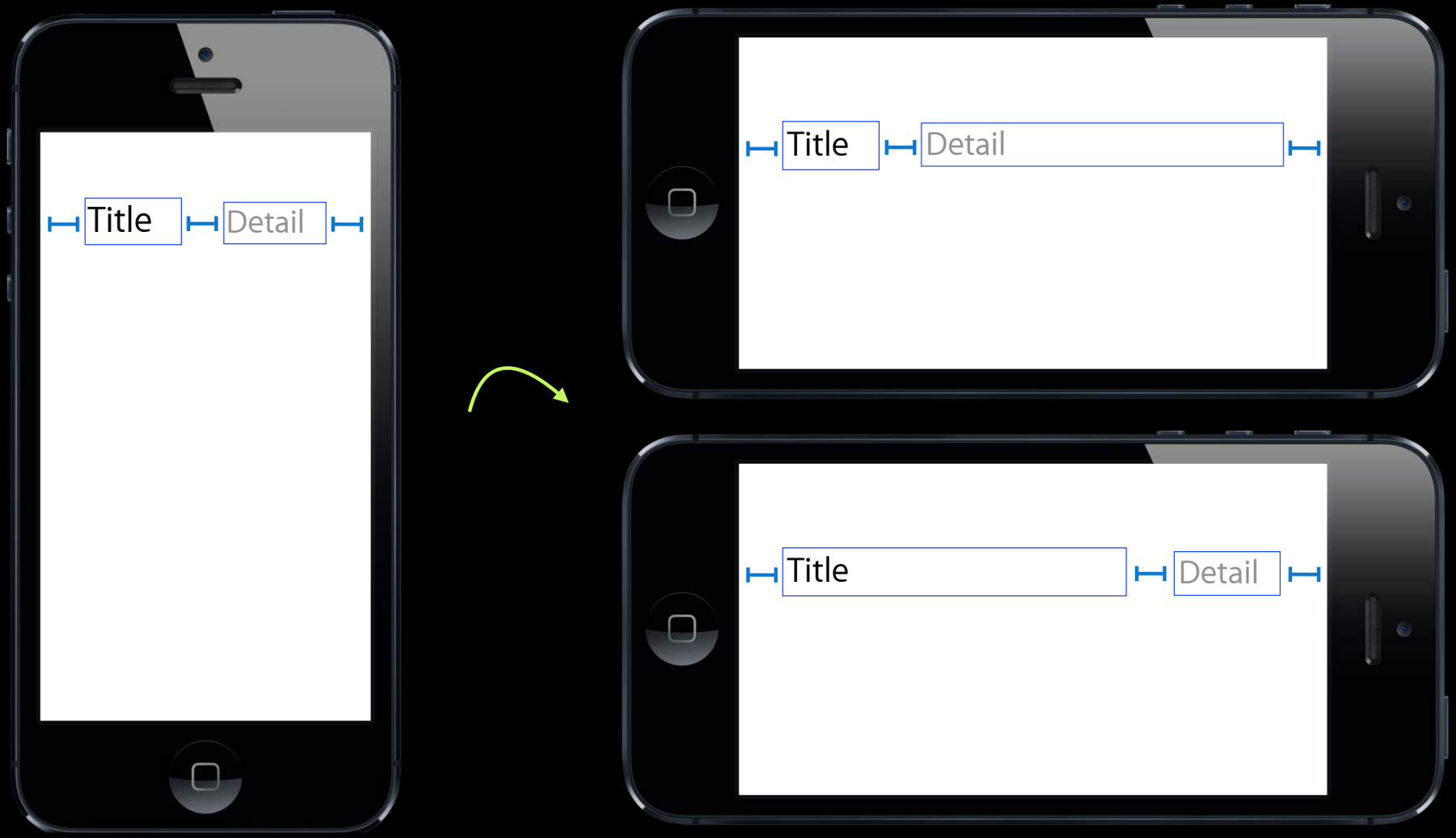
Things can go wrong

Ambiguous Frames



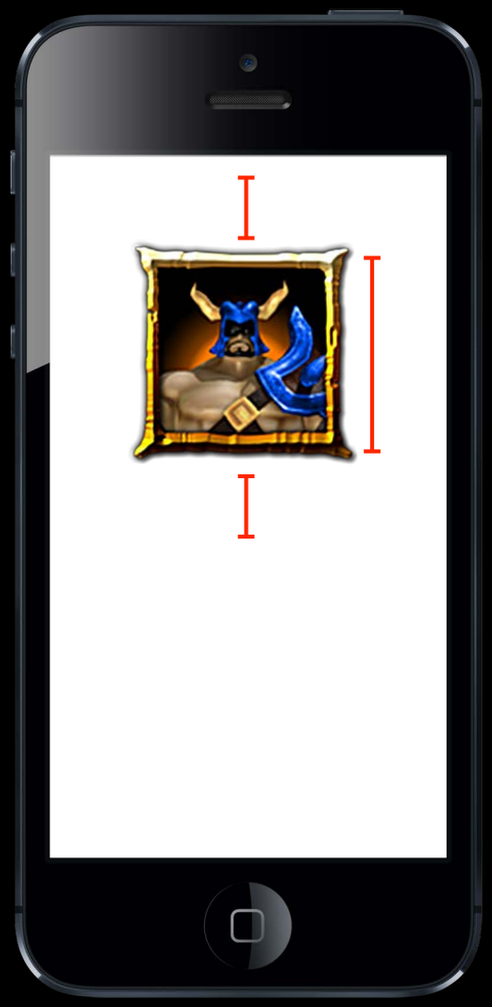
Things can go wrong

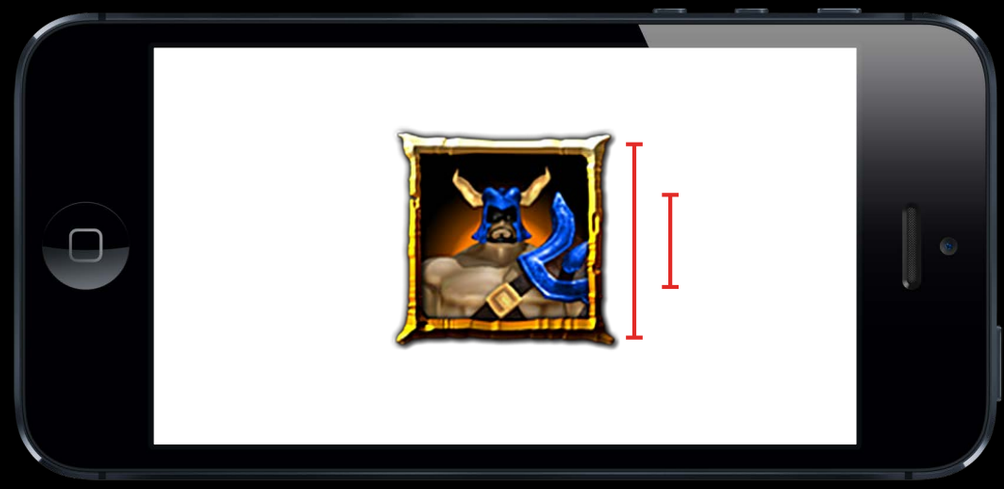
Ambiguous Frames



Things can go wrong

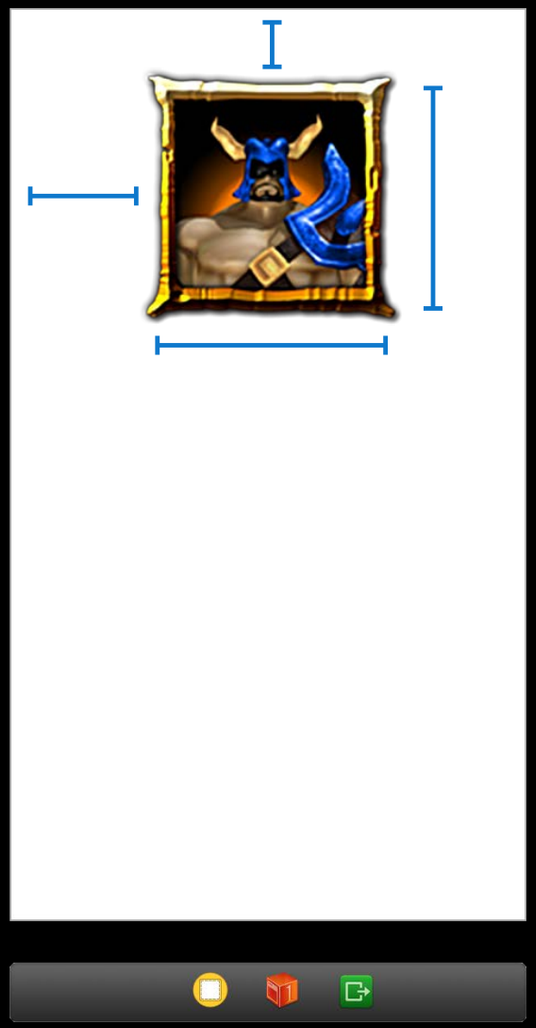
Conflicting Constraints





Things can go wrong

Misplaced Views



Things can go wrong

DEMO

End - Thank you!