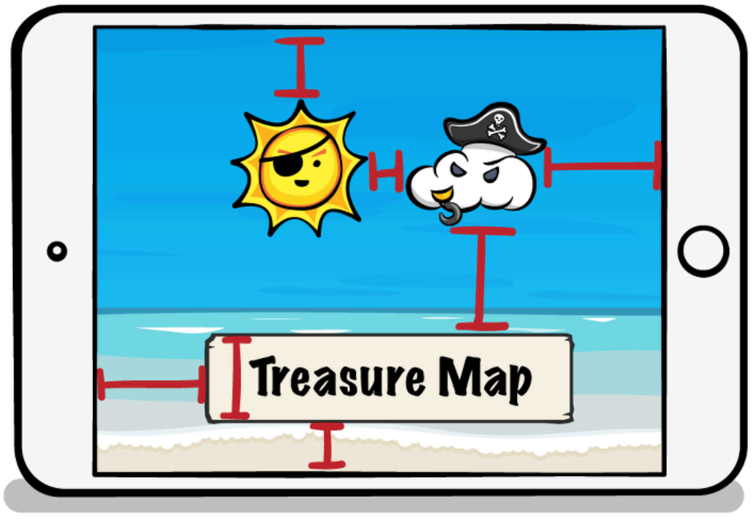
MASTERING AUTO LAYOUT



HANDS-ON CHALLENGES

Mastering Auto Layout

Jerry Beers

Challenge #4: Constraints in Code

By Jerry Beers

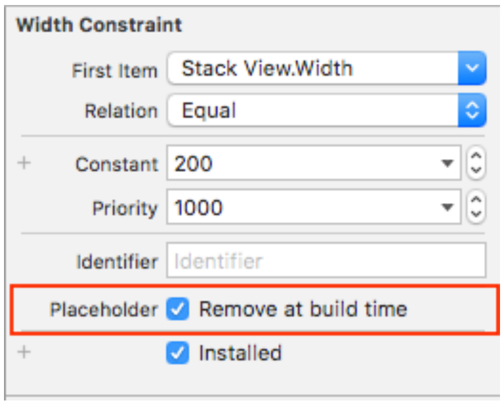
In this challenge, you're going to learn a few concepts. Primarily, you'll use a stack view to replace all the code we created in the demo to do the layout. But you'll also see how to mix using Interface Builder and code to accomplish your layout. Doing your layout in code isn't an all-or-nothing proposition. You can get it most of the way there visually in Interface Builder and then just add the parts that need logic in code.

Adding the Stack View

First, in Interface Builder, drag a horizontal stack view onto the **weather view controller scene**. You want that to be centered, so add vertical and horizontal centering constraints between the stack view and its superview.

At runtime, you'll add views to the stack view and they will determine the size, but you don't ever want it to overlap the edges, so add a >= 0 constraint between the leading edge and the superview's margin and the trailing edge and the superview's margin.

Because Interface Builder wants to know the size of the stack view, but we won't determine that until runtime, you're going to add **placeholder** constraints to the stack view. These get removed at build time, but basically tell Xcode, "don't worry about these attributes, they'll be defined later". Of course, if you do that and then don't follow through, your layout will have problems at runtime. But this is a technique that allows you to mix setting up your layout in Interface Builder and code. Add a width and height constraint to the stack view, and edit those constraints. Double click on the width constraint and select the **remove at build time** checkbox.



Width Constraint

First Item: Stack View.Width
Relation: Equal
Constant: 200
Priority: 1000
Identifier: Identifier
Placeholder: [✓] Remove at build time
[✓] Installed

Finally, create an outlet for the stack view in WeatherViewController, called stackView.

Deleting a Bunch of Code

Now comes the fun part, removing all the code you'll no longer need. At the top of WeatherViewController, remove the guides we no longer need:

```
var containerGuide = UILayoutGuide()
var spacerGuides = [UILayoutGuide]()
var imageConstraints = [NSLayoutConstraint]()
```

Then, completely remove the `viewWillTransition` and `setupImageConstraints` methods. And in `viewDidLoad`, remove these lines:

```
containerGuide.identifier = "container"
view.addLayoutGuide(containerGuide)
```

and remove:

```
if day > 0 {
    let spacer = UILayoutGuide()
    spacer.identifier = "spacer\(day)"
    view.addLayoutGuide(spacer)
    spacerGuides.append(spacer)
}
```

then replace this line:

```
view.addSubview(weatherImage)
```

with this:

```
stackView.addArrangedSubview(weatherImage)
```

Adding the image view to the stack view `arrangedSubviews` property will tell the stack view to manage layout for that image view.

And then we'll add the few constraints we want to keep and configure the stack view. At the bottom of `viewDidLoad`, replace this line:

```
setupImageConstraints(forSize: view.bounds.size)
```

with these:

```
// 1
for imageView in imageViews {
    imageView.heightAnchor.constraint(equalTo:
imageView.widthAnchor).isActive = true
}
// 2
stackView.distribution = .fillEqually
stackView.spacing = 8
```

This does the following:

1. Create a 1:1 aspect ratio constraint for each image
2. Configure the stack view to have some spacing between each image, but then fill the rest of the space equally

Conclusion

Look at all the code you were able to replace. This is much simpler, easier to follow, and easier to understand. Even in code, stack views are very easy to use and really do a lot for you.