Custom Collection View Layouts

Hands-on Challenges

Custom Collection View Layouts
Hands-on Challenges

Copyright © 2015 Razeware LLC.

Challenge G: Unclipping

The image view is now clipped to the container view, which removes the unwanted whitespace from the left and right sides of the cell, but ideally the cell should extend the full width of the collection view so you can't see any of it's four corners.

You can't change the size of the cell itself because of the gotcha covered in the previous episode, but since the container view already extends beyond the bounds of the cell, you may not need to.
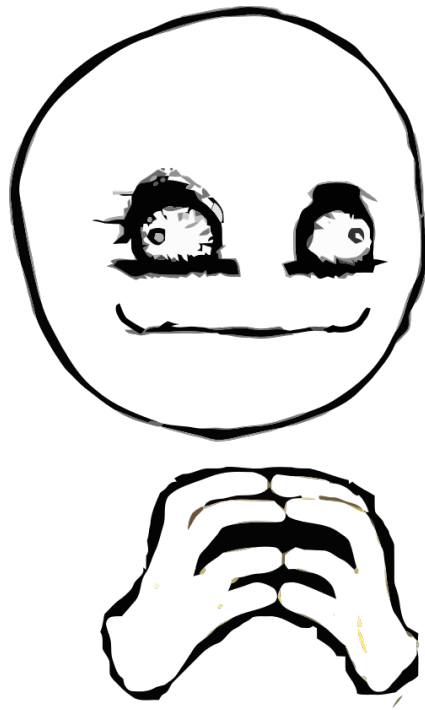
I may not? Tell me more...

Your challenge this time is to prevent the cell from clipping it's subviews, and then to add a gradient overlay to the container view to help the cell's text be more legible.

> **Hint**: For the gradient, have a poke around in the Views group in the Project Navigator as you might find something you can use there. :]

Before you turn the page for our solution, be sure to give it a try for yourself first!

Solution

Unclipping the cell

Open **TutorialCell.swift** and add the following to the bottom of the class:

```swift
override func awakeFromNib() {
  super.awakeFromNib()
  clipsToBounds = false
}
```

Here you're telling the cell *not* to clip it's subviews to it's bounds. You do this in `awakeFromNib()` because it only needs to be done once, when the cell is first created, as beyond that the cells are recycled by the collection view.

Build and run. You'll see that the image view is no longer clipped to the bounds of the cell:
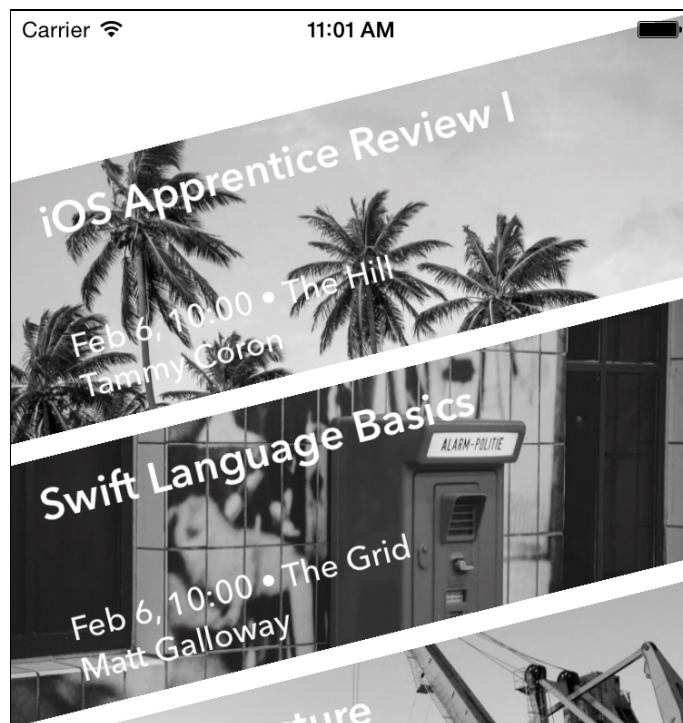

As the container view has the same height and centre as the cell, and it *is* clipping it's subviews, the image view now *appears* to fill the entire cell.

Some of the cells have background photos which are much lighter that others making the session details less legible – you'll fix that now!

Legible text

Open **Main.storyboard** and drag a **View** from the Object Library into the **Container** in the Document Outline, making sure it appears just below the **Image View**:
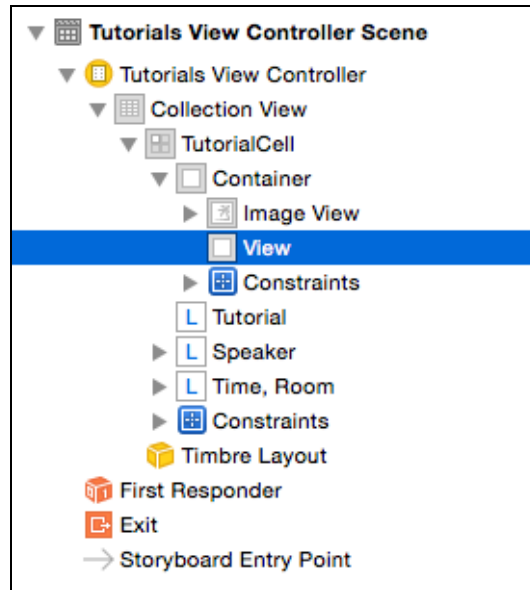

And with the view still selected, in the Identity inspector change Class to `GradientView`. This is a `UIView` subclass that's available in the sample project, which draws a horizontal gradient, and allows you to configure the two colors used for the gradient in Interface Builder, as it makes use of both `@IBDesignable` and `@IBInspectable`.

Open the Attributes inspector and change Gradient View\Start Color to **Black** with an Opacity of **75%**, and change Gradient View\End Color to **Clear Color**. While you're there also change View\Background to **Clear Color**:
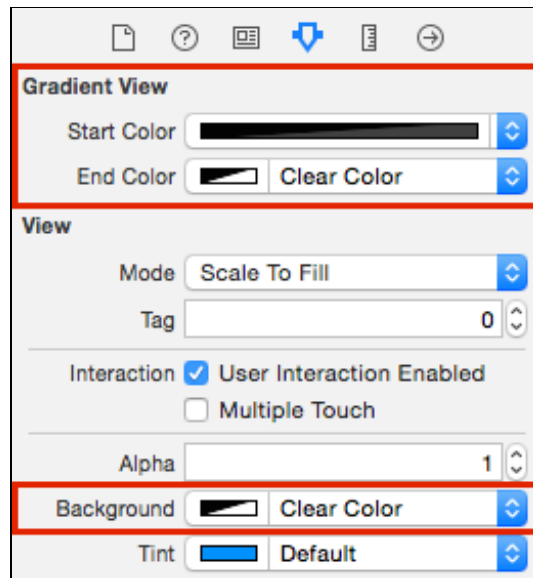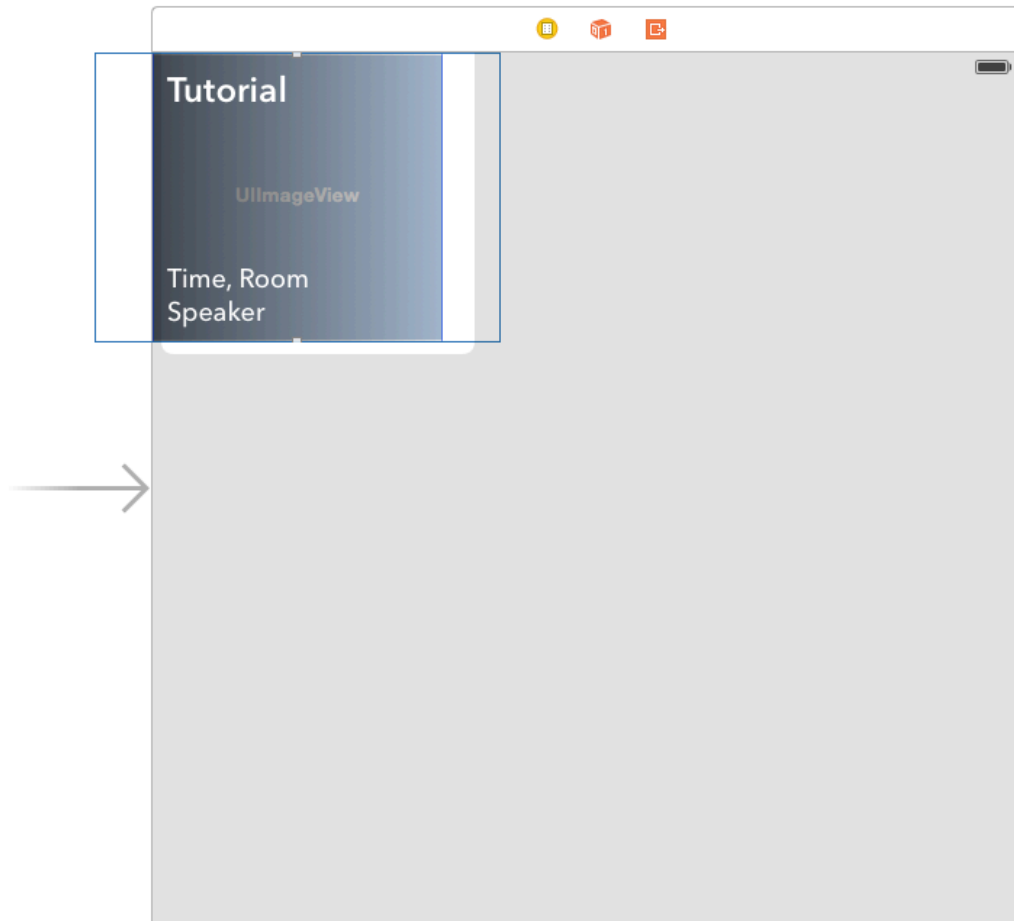
Next, click the **Pin** button at the bottom of the storyboard canvas and enable **Leading**, **Top**, **Trailing**, and **Bottom** spacing, and set all values to 0:
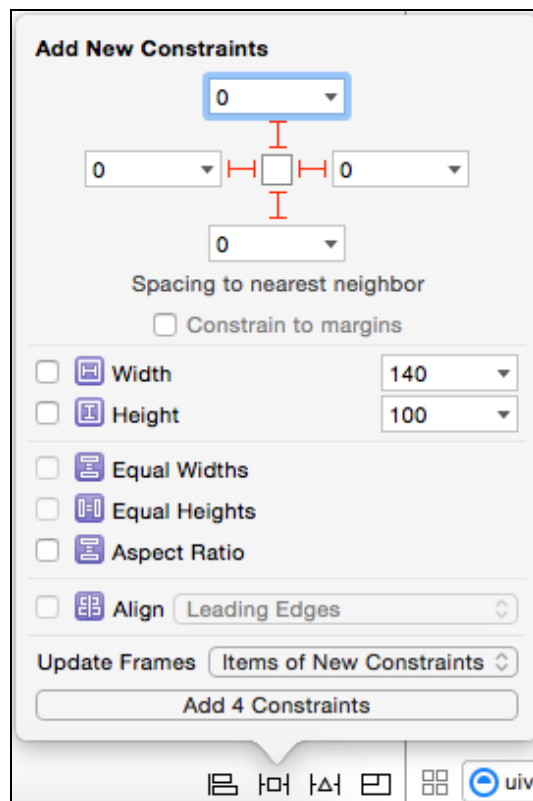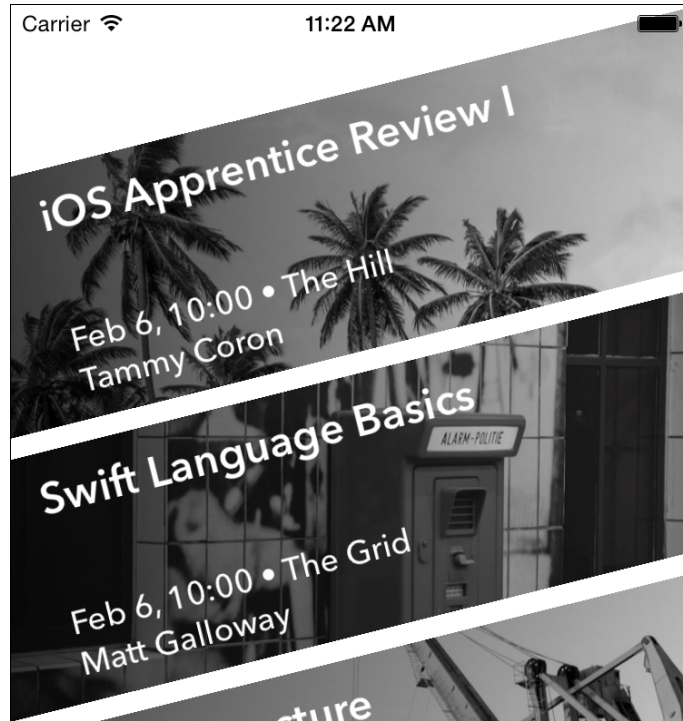

Then change Update Frames to **Items of New Constraints** and click **Add 4 Constraints**. Your storyboard should now look like the following:

And with that, build and run. You'll notice the cells now have a subtle gradient between the photo and the session details, making them much easier to read:

Congratulations. You have now unclipped the cells so the photos appear to stretch the full bounds of the cell, as well as adding a subtle gradient between the photo and the session details to help make the text more legible.

In the next video, the final one of the series, you'll add a really cool parallax effect to the cells, which will wrap up the Timbre layout.