# Custom Collection View Layouts

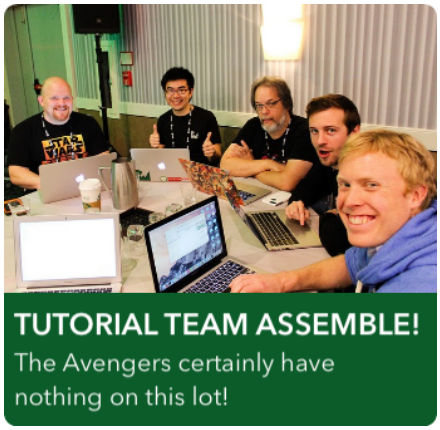
Hands-on Challenges

# Custom Collection View Layouts Hands-on Challenges



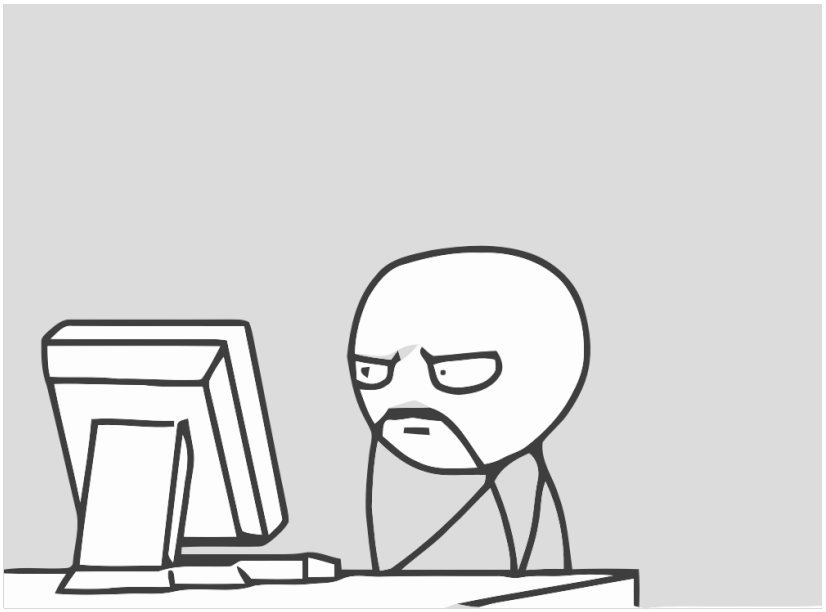
# Challenge B: Comments

The annotation for each photo should contain both a caption *and* a comment, but so far you've only implemented the former.




Your challenge this time is to add the comment to the annotation area of the cell. But don't be mislead, as it's not quite as easy as it may first appear. Since some comments are naturally longer than others, the comment label should be sized according to the length of the comment so the text doesn't get truncated. This will obviously have a knock on effect on the overall height of the cells, and should be taken into account when calculating the layout.

Time to get your thinking caps on.

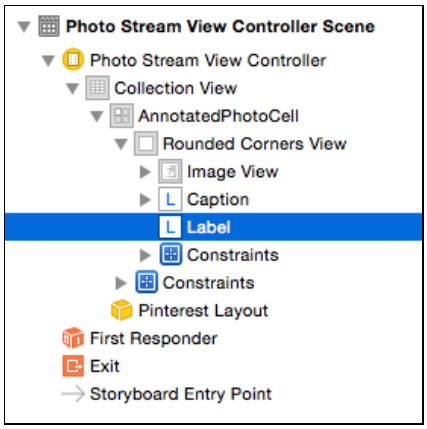


Before you turn the page for our solution, be sure to give it a try for yourself first!

# Solution

## Updating the Interface

Open **Main.storyboard** and drag a **Label** from the Object Library into the Document Outline and drop it just below the Caption label. Your scene should now look like the following:




The reason you've dragged the label into the Document Outline was because it needed to be a subview of the "Rounded Corners View", and this is the most straight forward way to gurantee that.

Select the new label, and in the Attributes inspector make the following changes:

• Change Text to the placeholder text **Comment**;

• Change Font to **Avenir Next Regular**, with a size of **10**;

• Change Lines to **0**.

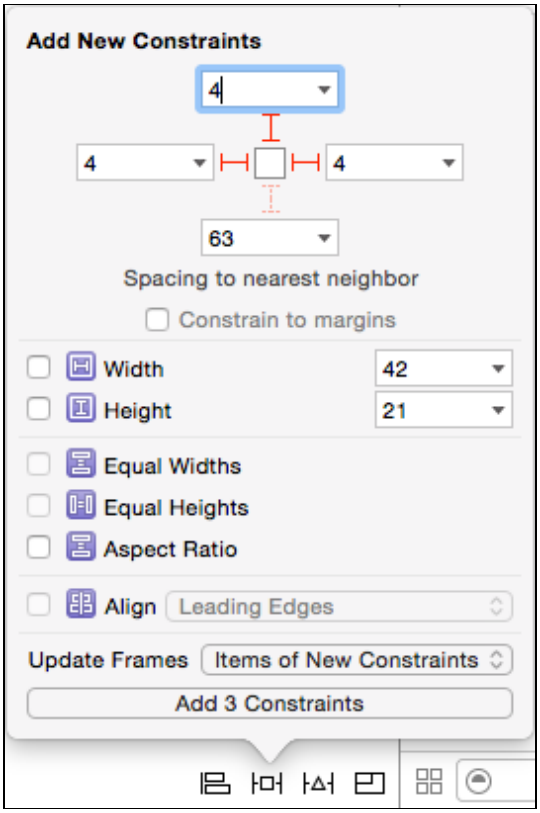
Then, drag the label down and position it under the Caption label.

Using the **Pin** button at the bottom of the storyboard canvas add the following layout contraints, making sure the Top distance is associated with the Caption label and not any of the other views:

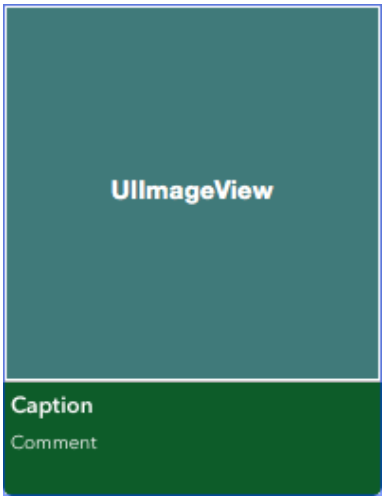
• Leading enabled and set to **4**;

• Top enabled and set to **4**;

• Trailing enabled and set to **4**.




Make sure **Update Frames** is set to **Items of New Constraints** and click **Add Constraints**. Your cell should now look like the following:




You haven't set a height for the label because you set the Lines property to zero, which means the label will set its own height based on its content – this is exactly the behavior you want!

With that done, it's time to move onto updating the layout delegate.

## Updating the Layout Delegate

Open **PhotoStreamViewController.swift** and replace the existing implementation of `collectionView(_:heightForAnnotationAtIndexPath:withWidth:)` with the following:

```
// 1
let photo = photos[indexPath.item]
// 2
let font = UIFont(name: "AvenirNext-Regular", size: 10)!
// 3
let commentHeight = photo.heightForComment(font, width: width)
// 4
let height = 4 + 17 + 4 + commentHeight + 4
return height
```

Here's the play-by-play of what's happening:

1. Get the corresponding `Photo` object for the given index path;
2. Create an instance of `UIFont` with the same attributes as the Comment label you created in Interface Builder;
3. Ask the photo object to calculate the height of its comment and return it, using the font from #2 and restricting it to the given width, which in this case is the inset column width passed by the layout;
4. Add 3 lots of 4-point padding, and the fixed height of the Caption label to the comment height and return it. The 4-point padding represents the spacing between each of the items in the annotation area, as well as the top and bottom spacing.

Now the delegate is all setup to calculate and return the proper height of the comment there's just one last thing to update – the `UICollectionViewCell` subclass.

## Updating the UICollectionViewCell Subclass

Open **AnnotatedPhotoCell.swift** and add the following outlet just below the existing ones:

```
@IBOutlet private weak var commentLabel: UILabel!
```

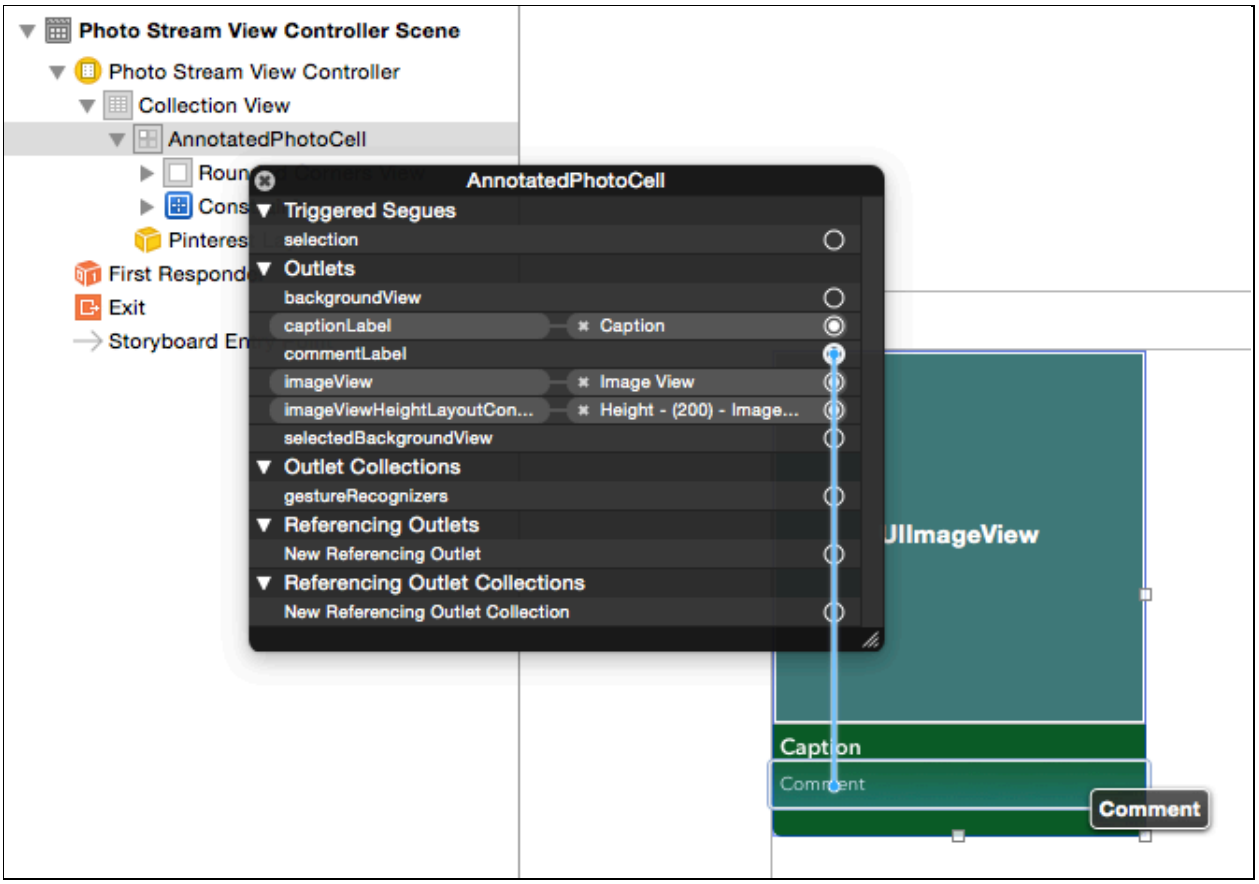
This will be used to reference the Comment label you added earlier in Interface Builder.

Now, add this to the bottom of the `if` block in the `didSet` observer of the `photo` property:

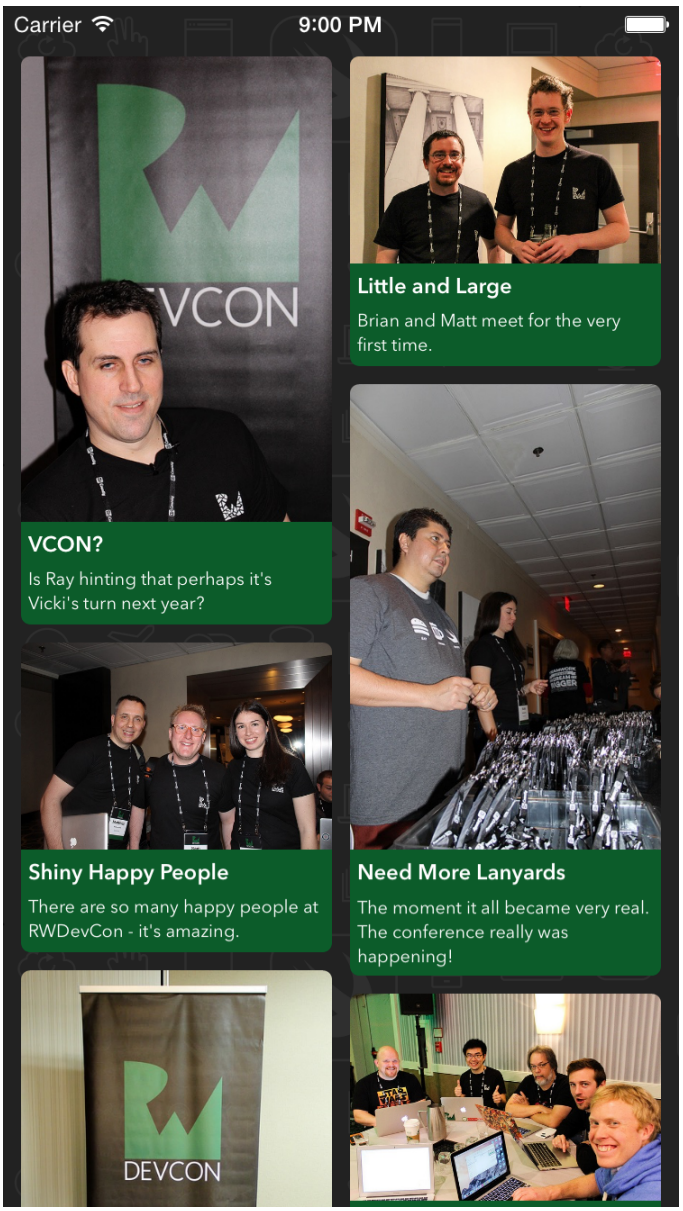
```
commentLabel.text = photo.comment
```

This simply updates the text of the Comment label whenever you set the photo on the cell.

Finally, open **Main.storyboard** and right-click on **AnnotatedPhotoCell** to invoke the Segues and Outlets popup. Drag from the `commentLabel` outlet to the Comment label to connect the two:




Build and run. Your cells will now display the photo's caption in a perfectly sized label, and the cell's overall size will have been adjusted accordingly:




Congratulations! You now have a fully equipped annotation to sit alongside your photos, and the collection view cells are all sized appropriatley based on their content. It's *almost* a perfect replication of the popular Pinterest layout :]