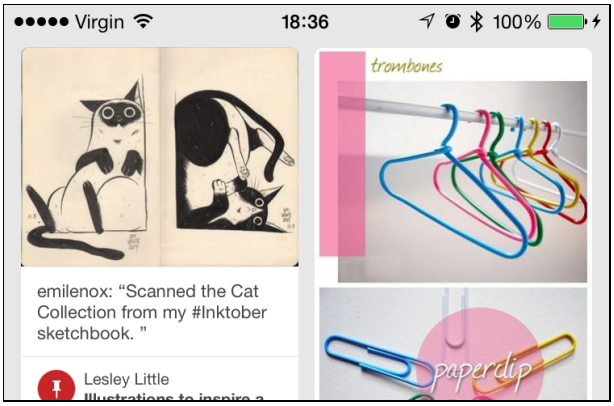
Custom Collection View Layouts

Hands-on Challenges

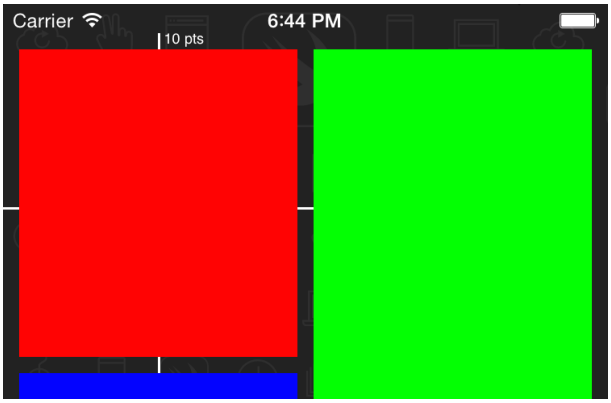
Custom Collection View Layouts Hands-on Challenges

Challenge A: Insets & Padding

As it stands, all the cells are packed in quite tightly, but if you download and take a look at the Pinterest app you'll see the the content should be inset from the collection view around all four edges, and the cells themselves should also have a similar amount of padding around their four edges:



Your challenge this time is to adapt the layout so it takes the collection view's `contentInset` property into account, before moving on to add custom cell padding padding. Be sure that you make the resulting space equal around all sides of each cell:



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

Solution

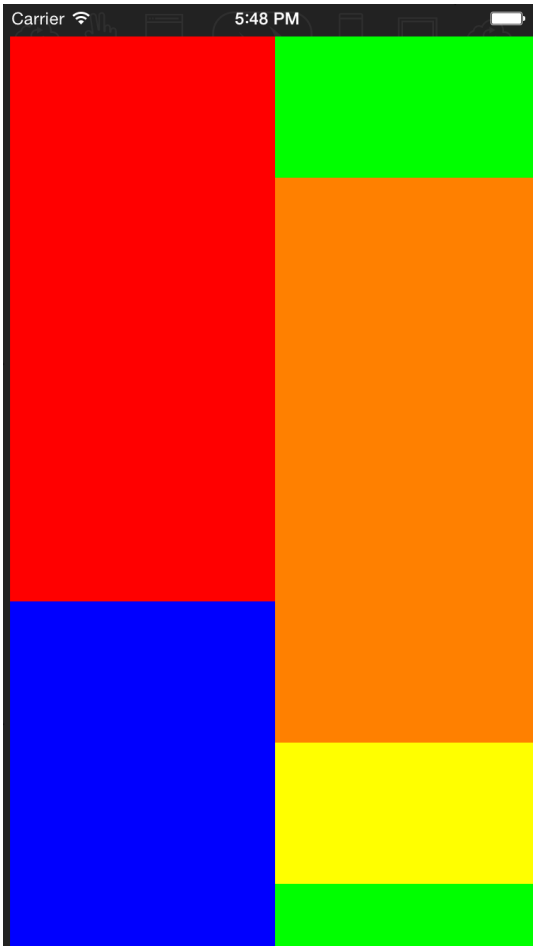
Content Insets

Open **PhotoStreamViewController.swift** and add the following to `viewDidLoad()`, just below where you set the background color of the collection view:

```
collectionView!.contentInset =
  UIEdgeInsets(top: 23, left: 5, bottom: 10, right: 5)
```

Here you're simply insetting the content of the collection view by the specified amounts, which are in points. The `top` value is the largest since you want to take the status bar into account, and the `left` and `right` values are half the bottom value because the other half will be added by the cell padding, which will be appiled to the four edges of the cell. This will result is equal spacing around all four edges of the cell.

Build and run. You'll see that whilst the content *is* offset, it's not quite what we were looking for:



It appears at first glance that the insets have only been applied to the top and left edges, but if you scroll horizontally you'll see that they have in fact been applied to the right edge as well, but this has had the unwanted side-effect of increasing the overall width of the collection view.

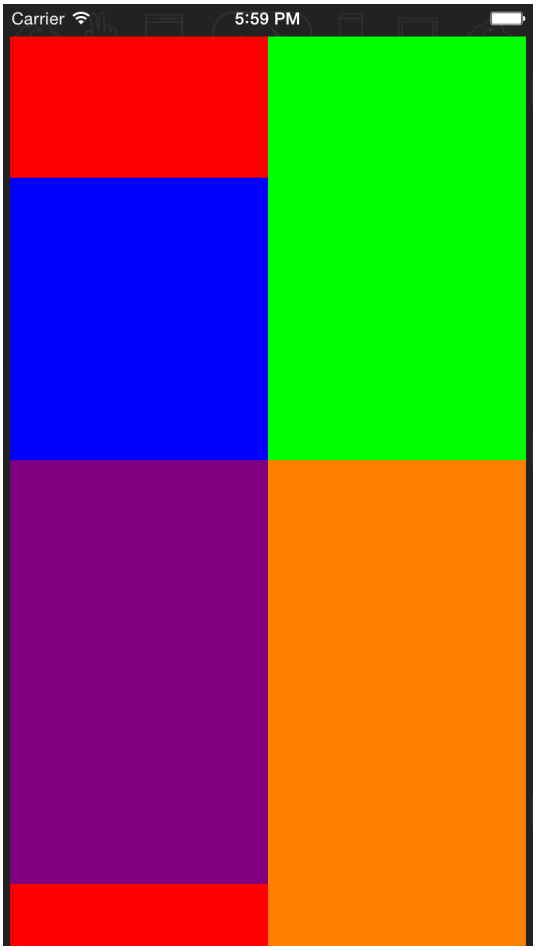
Luckily the fix is an easy one – you simply need to update the layout so it takes the `contentInset` property into account.

Open **PinterestLayout.swift** and replace the getter for the `width` property with the following:

```
let insets = collectionView!.contentInset
return CGRectGetWidth(collectionView!.bounds) - (insets.left +
insets.right)
```

Here you're updating the `width` property so it takes any insets into account. This is the best place to do it since you use the `width` property to calculate the width of each column.

Build and run. You'll see the content is now properly inset around all four edges:



With that done, it's time to move onto the cell padding.

Cell Padding

Still working in **PinterestLayout.swift**, add the following public property to the top of the class, just below the existing properties:

```
var cellPadding: CGFloat = 0
```

This property will allow you to set the desired cell padding outside of the layout itself.

Then, in `prepareLayout()`, add the following directly below where you create the `frame` variable:

```
let insetFrame = CGRectInset(frame, cellPadding, cellPadding)
```

Here you're creating a new rect based on `frame`, but one that's inset by the `cellPadding` property around all four edges. By using `CGRectInset(rect:dx:dy:)` the center point of the rect remains unchanged, which is important in this context as you don't want to move the cell, just shrink it.

Next, change the line:

```
attributes.frame = frame
```

To the following:

```
attributes.frame = insetFrame
```

All you're doing here is telling the layout attributes to use the new, inset frame rather than the original frame.

Build and run. And…nothing's changed!

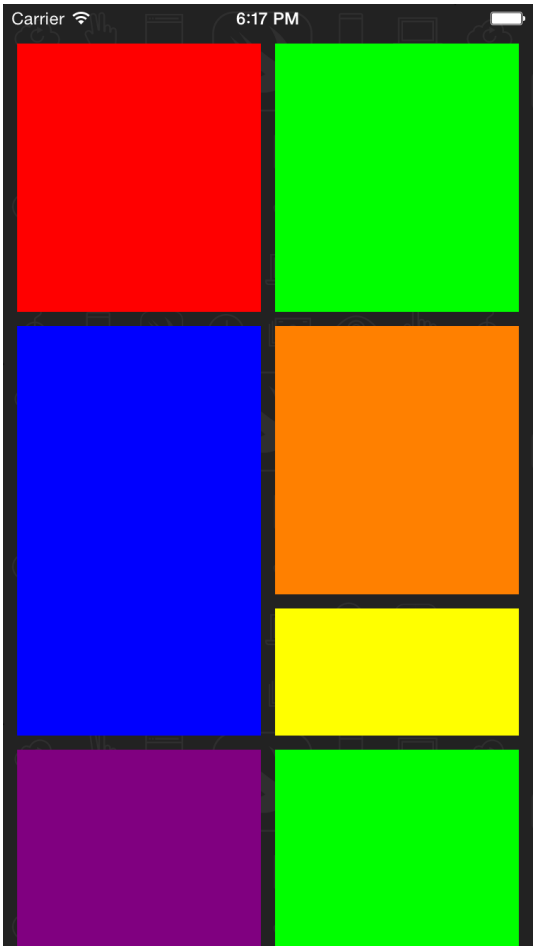
That’s because you need to actually set the new `cellPadding` property.

Open **PhotoStreamViewController.swift** and add the following to `viewDidLoad()`, just below where you set the number of columns on the layout:

```
layout.cellPadding = 5
```

This sets the cell padding to 5 points, which is applied to all four edges of the cell, and results in equal padding between the edges of the collection view and the cells, and between the cells themselves.

Build and run. You’ll see the cell padding is now working as intended:



Congratulations! You now have a layout that takes the collection view's `contentInset` property into account, as well as applying custom cell padding to all four edges of each cell.