My Own ScrollView

Learning Objectives

We're going to build our own ScrollView using an UIView and pan gestures. While in real life you would use a UIScrollView over building one from scratch using UIView, its good to understand how it works and realize how Apple engineers went about building it to makes our lives easier.

Setup

- Create a new iOS project named: `MyScrollView`
- Use the Single View template

Source Control: Remember to create and push your project to github.

Instructions

- Create a UIView and set the frame to be the bounds of your root View.
- Add four UIViews to that View

    - A red UIView at (20,20) x, y coordinates and with width 100 and height 100
    - A green UIView at (150,150) x, y coordinates and with width 150 and height 200
    - A blue UIView at (40,400) x, y coordinates and with width 200 and height 150
    - A yellow UIView at (100,600) x, y coordinates and with width 180 and height 150
- Run the app on iPhone 5 simulator and check that you cannot see the yellow box yet.
- Move the origin of the bounds of your root view down 100 point in the y direction, you should now see part of the yellow box. (Hint: Do this in viewDidAppear)

It looks as though the view has moved down by 100 points, and this is in fact true in relation to its own coordinate system. The view's actual position on the screen (or in its superview, to put it more accurately) remains fixed, however, as that is determined by its frame, which has not changed. However, adjusting the bounds's origin is equivalent to moving down such that another part of it becomes visible through the view. And this is exactly what UIScrollView does when it scrolls. Note that from the perspective of the user it appears as though the view's subviews are moving, although their positions in terms of the view’s coordinate system (in other words, their frames) remain unchanged.

Now, let’s build a ScrollView.

- Add a class called MyScrollView which inherits from UIView
- You need to add two things to your custom UIView
  - A CGSize property called contentSize
  - A PanGestureRecognizer
- In the method that handles events from the PanGestureRecognizer, look for how far your have panned. Then, move (translate) the view's frame, but do not permit values that would violate contentSize.
- Refactor your code so that the boxes are added as subviews to MyScrollView and set the contentSize.

Note: We implemented the basics of UIScrollView but there is a lot more to the real UIScrollView than just this. Momentum scrolling, bouncing, scroll indicators, zooming, and delegate methods are just some of the features we have not implemented here.