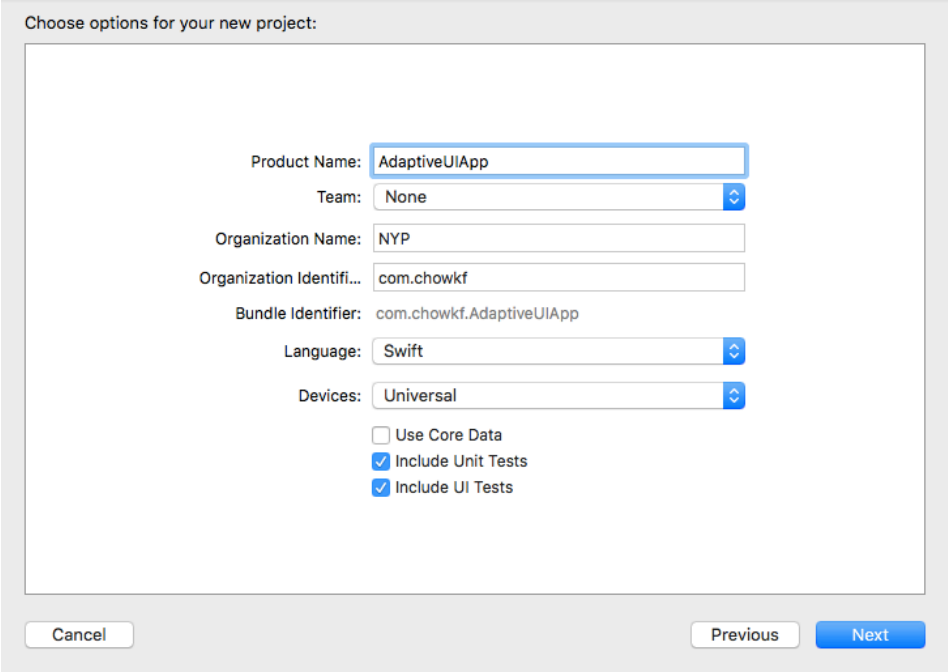
Practical 07: Adaptive UIs

In this lab, we will design a simple form that allows the users to key details about a movie. This form will be used in subsequent practicals.

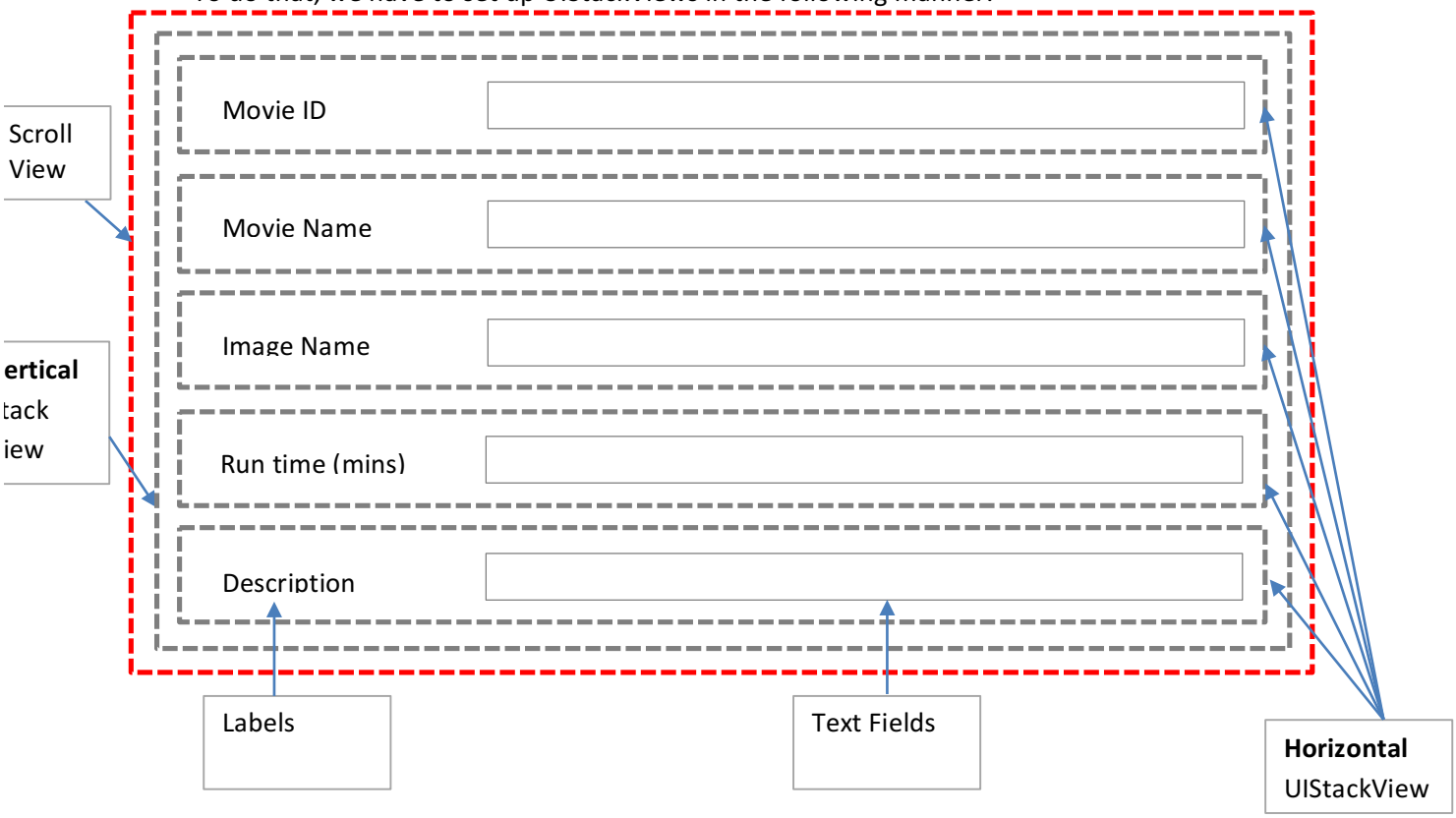
Section 1: Adding Constraints

1. Using Xcode, click on **Create a new Xcode project**.
2. Choose a **Single View Application template** for this practical.
3. Next, configure this project and name is as **AdaptiveUIApp**. For device, choose **Universal**.

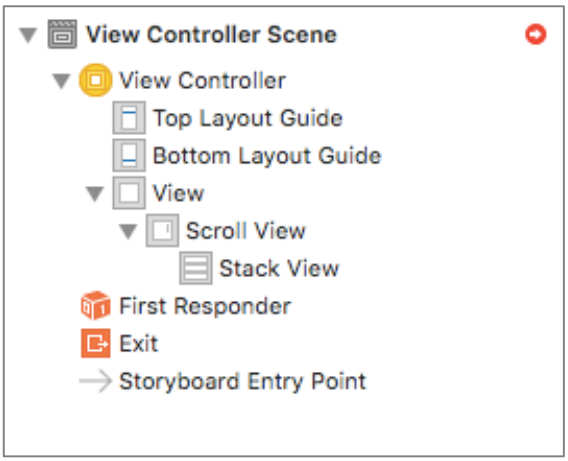


4. At the Project Navigator, click on the **Main.storyboard** to open it in the Interface Builder.
5. We will want to build the following layout as shown below:

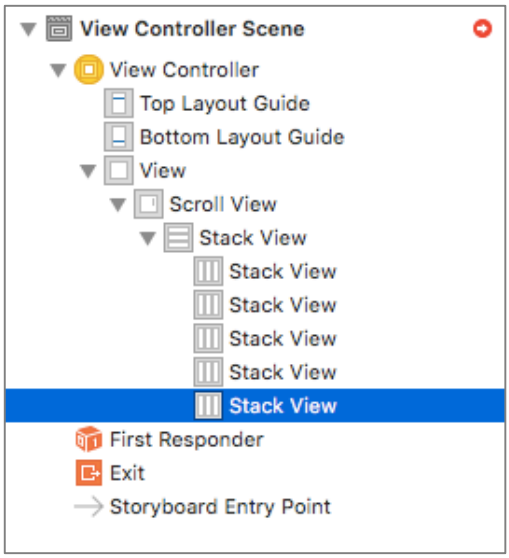
To do that, we have to set up UIStackViews in the following manner:



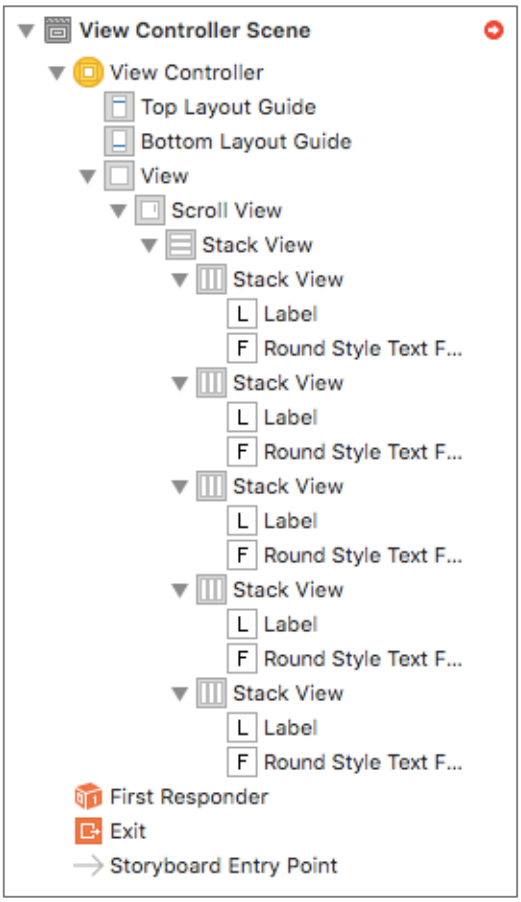
6. In the Object Library at the bottom right of your XCode screen, search for Scroll View and drag it into the structure tree of your storyboard. Then search for Vertical Stack View and drag it into the Scroll View. Don't worry about the position in your view controller yet.



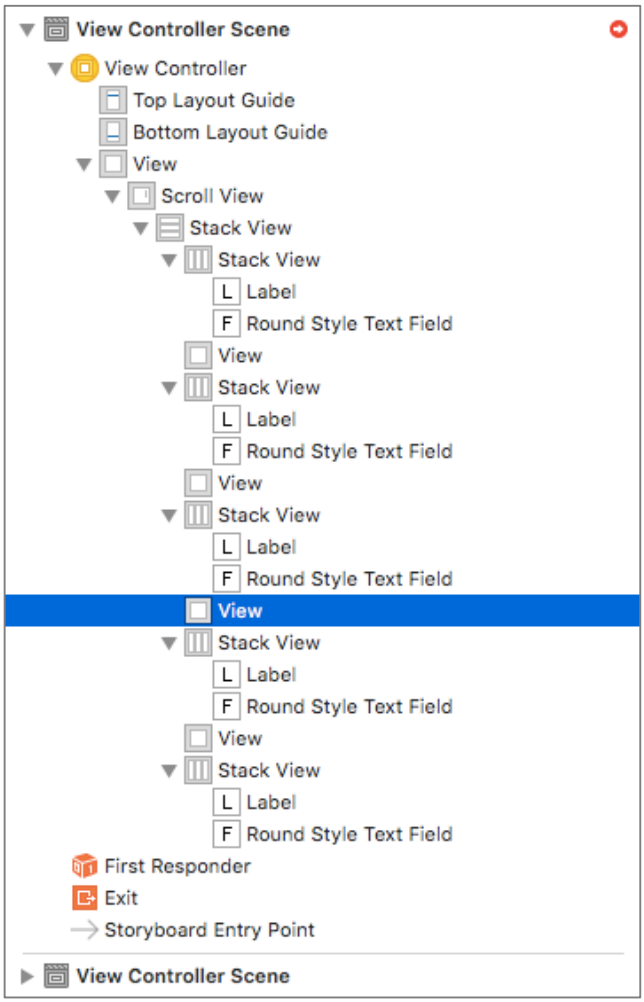
7. Drag 5 Horizontal Stack Views underneath the Vertical Stack View in the structure tree of your Storyboard.



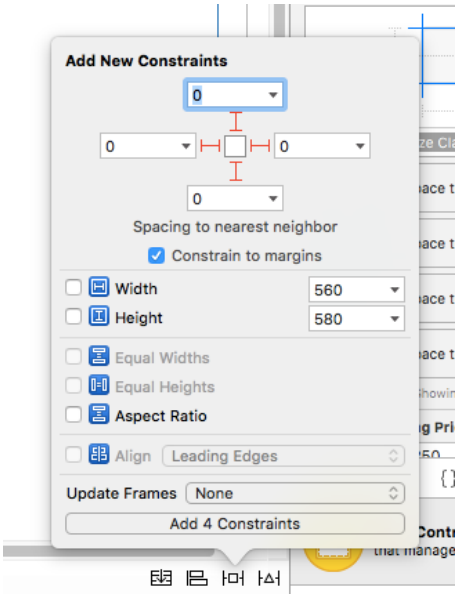
8. Then drag 1 Label and 1 Text Field beneath each of the 5 Horizontal Stack Layouts such that your structure tree looks like the following:



9. Finally, let's add 4 UIViews, one between each Horizontal Stack View like this:



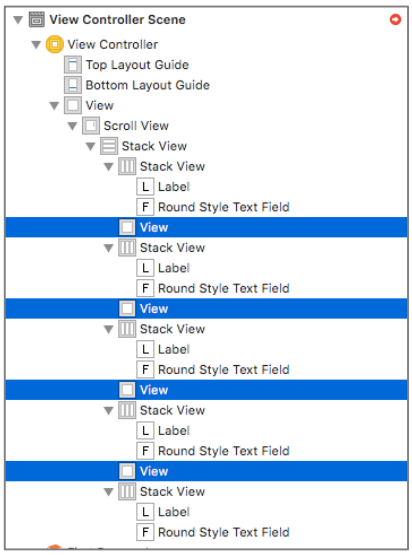
10. We will want to fit the Scroll View to the edges of the screen. To do so, click on the Scroll View in your structure tree, then click the Pin button. Set the top, left, right, bottom constraints to **0**. Change Update Frames to **All Frames in Container**, then click **Add 4 Constraints**.



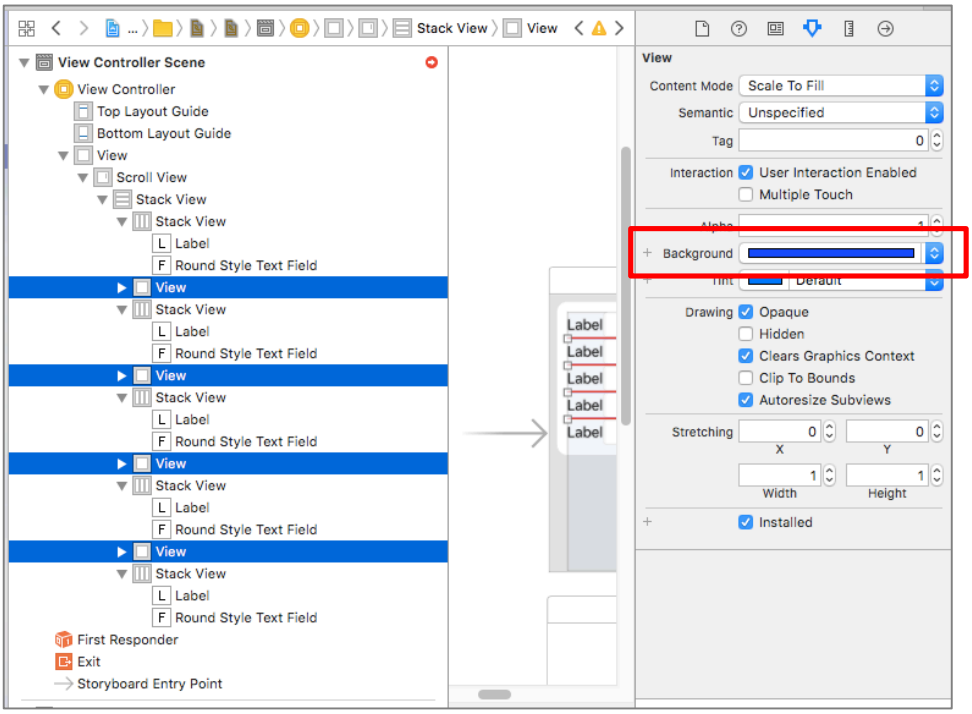
11. We will also want the Vertical Stack View to fit into the Scroll View. To do so, click on the Vertical Stack View, then click the Pin button. Set the top, left, right, bottom constraints to **0**. Change Update Frames to **All Frames in Container**, then click **Add 4 Constraints**.

12. Ctrl-click from the Vertical Stack View to the Scroll View and select **Equal Widths**.

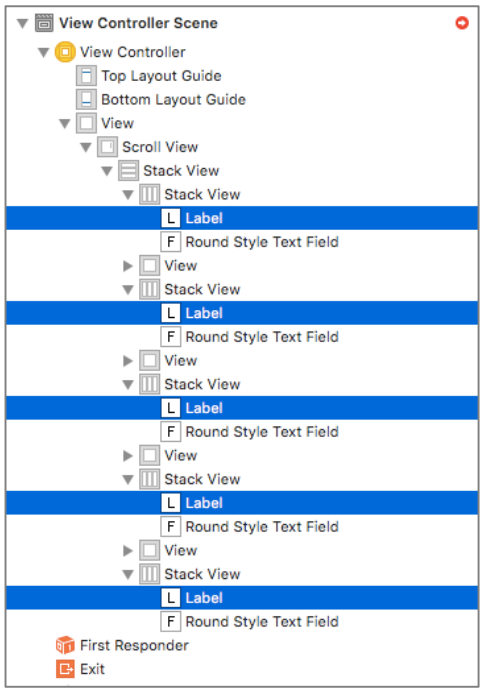
13. Next we are going to make use of the 4 UIVIews we added to draw horizontal lines across the screen by setting a height constraint of 1. To do so, hold your Command key and click on the 4 UIViews in the structure tree.



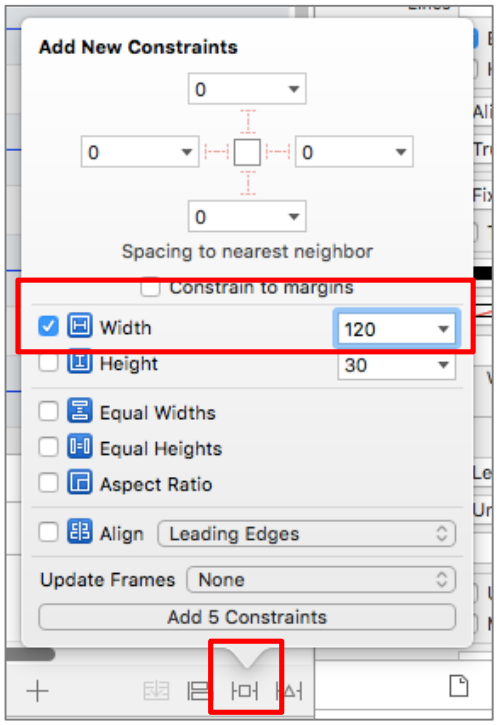
14. Click on the Pin button, set the Height to **1**, and click **Add Constraints**.
15. Then go to the Attributes inspector, and set the background color to **Blue**, or whatever colour you like.



16. Now we want some spacing between the Labels and the Text Field, such that the Text Fields' left edges are aligned vertically. What we can do is to set a Width constraint to each of the Labels. To do so, hold down your Command key and select all the 5 labels in your structure tree.

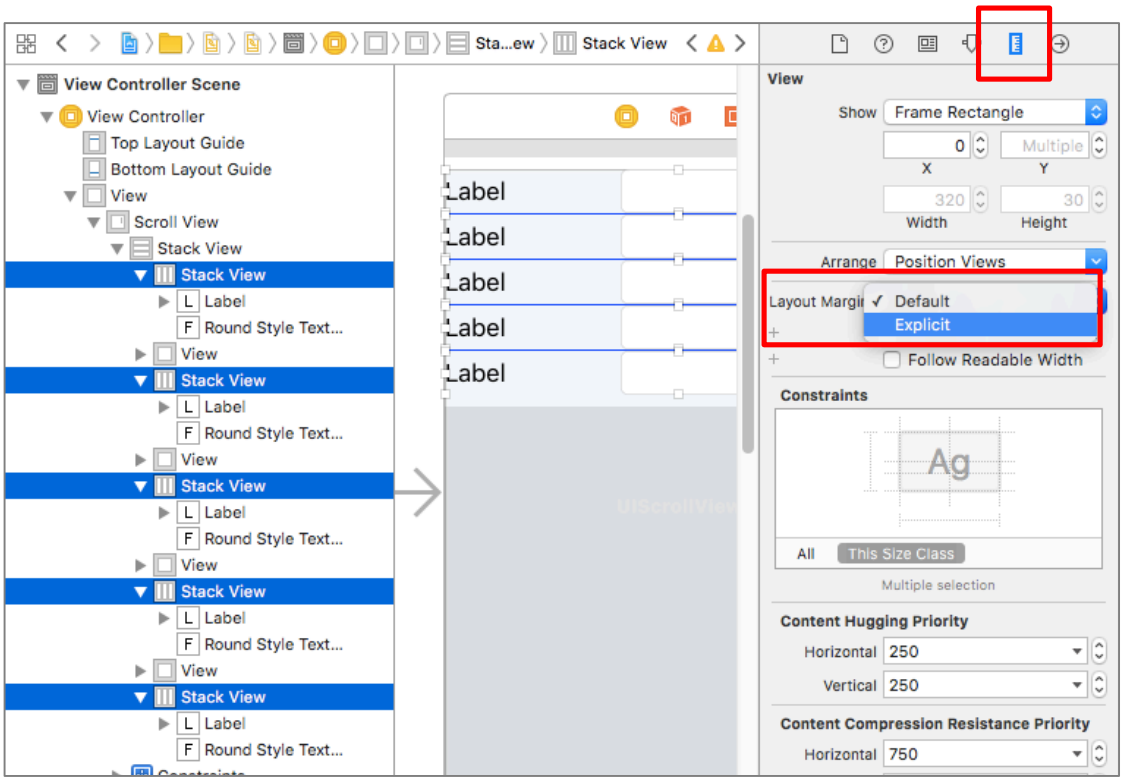


17. Click on the Pin button, and set the Width to **120**, and click **Add 5 Constraints**.

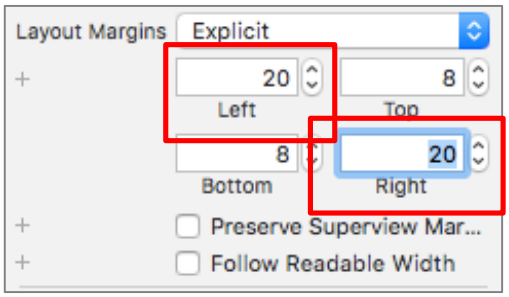


18. By this time, your Storyboard should look like this:

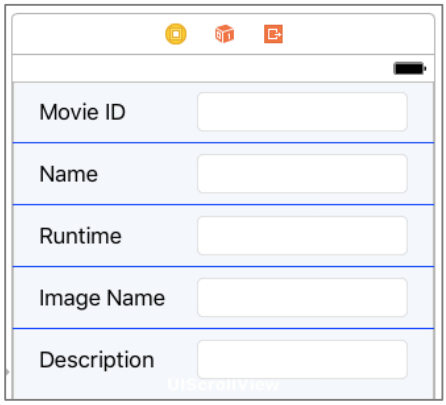
19. We would still like to add some spacing between the Horizontal Stack Views. And there are a number of ways to achieve this. For this example, we are going to add some margins inside the Horizontal Stack Views. To do so, hold down your Command key, select all 5 Horizontal Stack Views, click the **Size Inspector**, change the Layout Margins to **Explicit**.



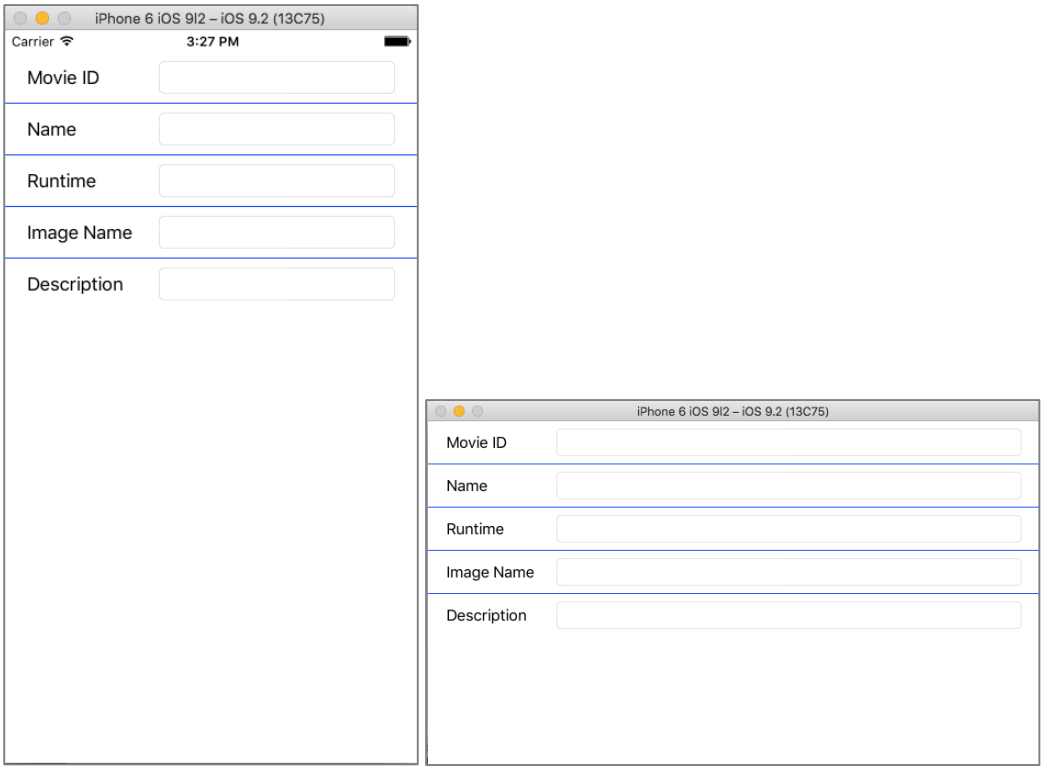
20. Then enter **20** for the left and right margins:



21. Finally, modify the labels so that they look like the following:

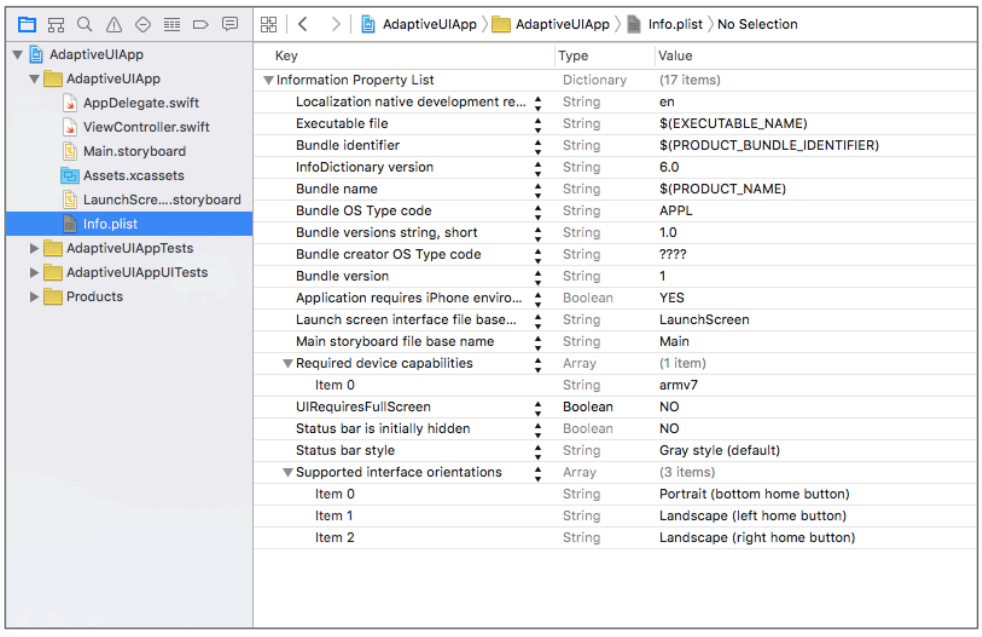


22. Run your app in the simulator. Change the orientation between Portrait and Landscape to see the layout adjust itself.



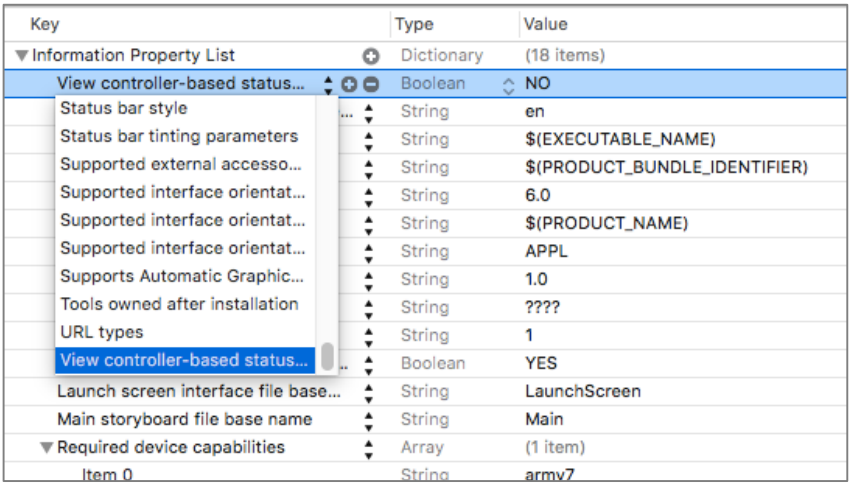
23. You will notice that in the landscape mode, the status bar is missing. This is new behavior since iOS 8. To enable the status bar to appear, you must do two things.

24. First, select the Info.plist file in your project navigator.

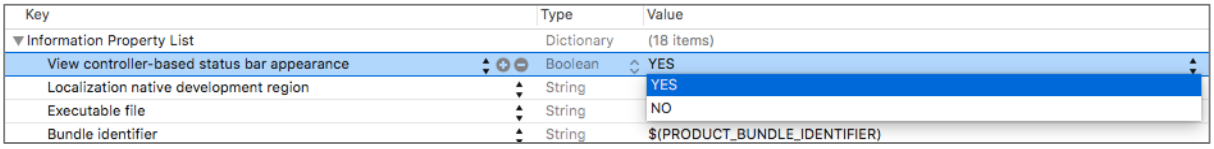


25. In the middle pane, click on the '+' icon in the "Information Property List" line. (You have to mouse over to see the '+' icon appear).

26. A dropdown list will appear. Select the "View controller-based status bar appearance".



27. Set the value to YES.



28. The open the ViewController.swift file, and add the following highlighted code.

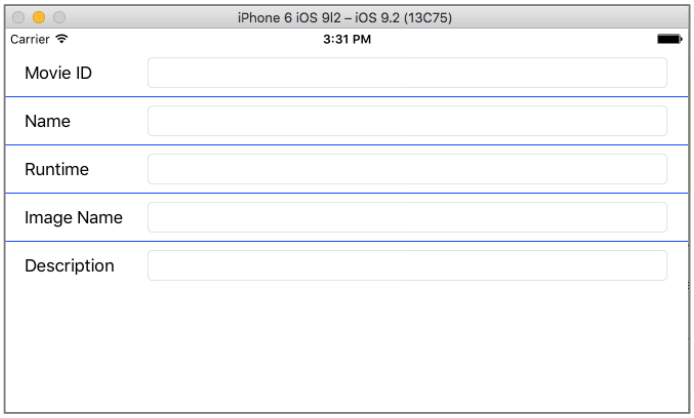
```
import UIKit

class ViewController: UIViewController {

    ...
    override var prefersStatusBarHidden: Bool
    {
        get
        {
            return false
        }
    }

}
```

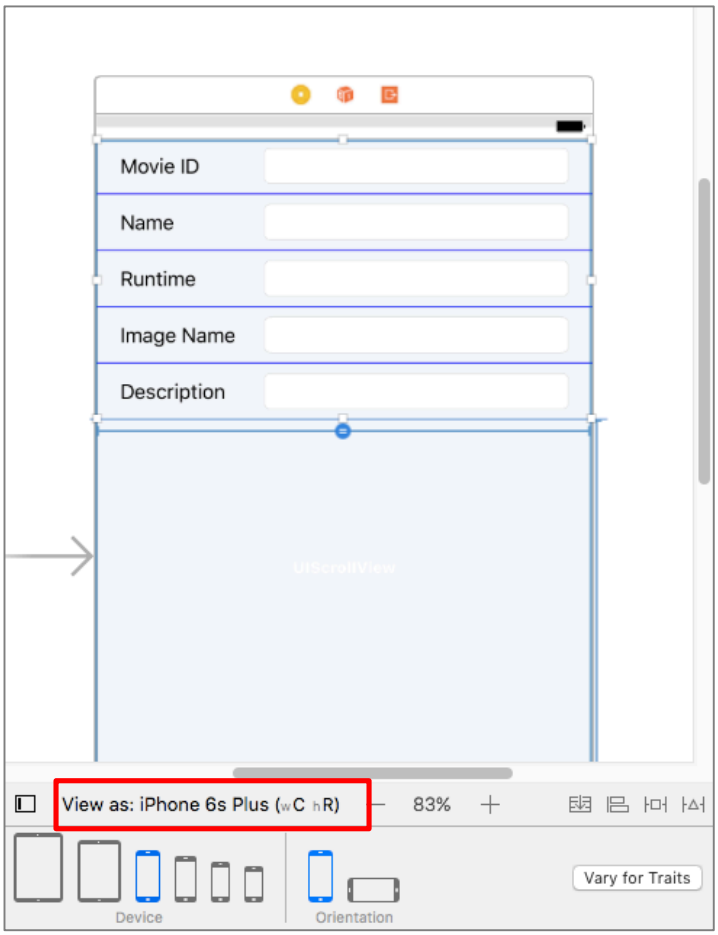
29. Then you should be able to see the status bar appear in the landscape orientation.


iPhone 6 iOS 9l2 – iOS 9.2 (13C75)
Carrier
3:31 PM
Movie ID
Name
Runtime
Image Name
Description

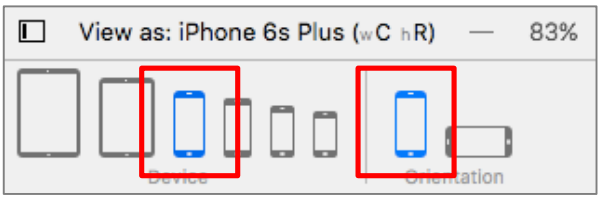
Section 2: Trait Variations

You can set a different layout on different device traits. In this practical we will turn all Horizontal Stack Layouts into Vertical Stack Layouts automatically on devices / orientations with compact widths, using the Trait Variations capability of the XCode design.

30. At the bottom of the Storyboard designer, click on the View as: IPhone… to show the panel that allows you to select devices, orientations and adaptations.

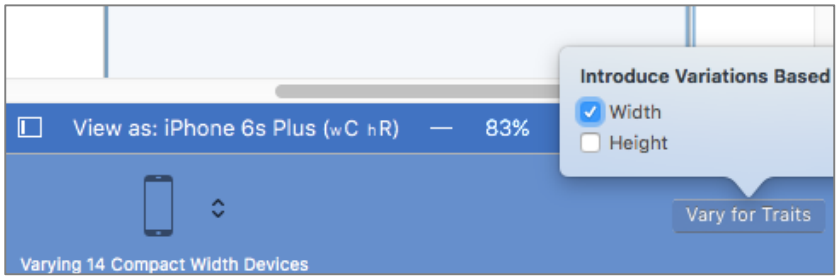


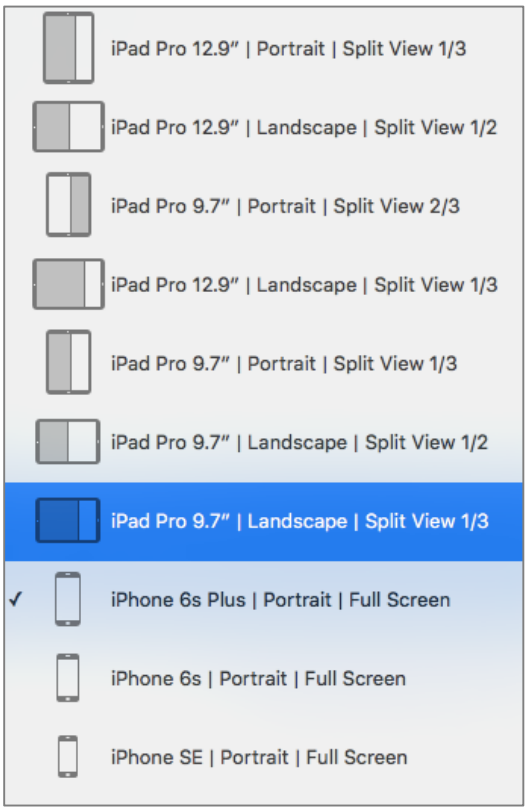
31. Select any combination device and orientation with a compact width (indicated with a wC). If you can't find a suitable device, select **iPhone 6s Plus** for the device, **Portrait** for the orientation.



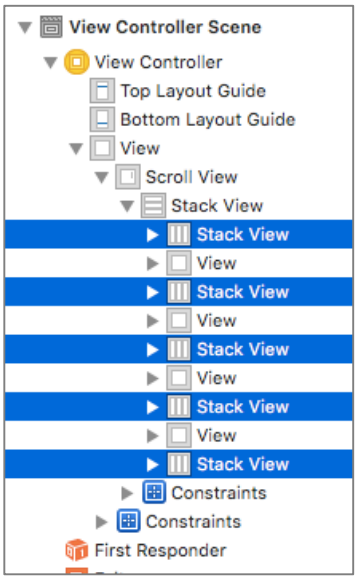
32. Click on Vary for Traits. When the pop-over appears, select Width. This is because we only want to vary the attributes specific to devices / orientations whose widths are the same trait as the device we have selected.

    When you do that, XCode will filter out all devices / orientations that are compact width in our case.

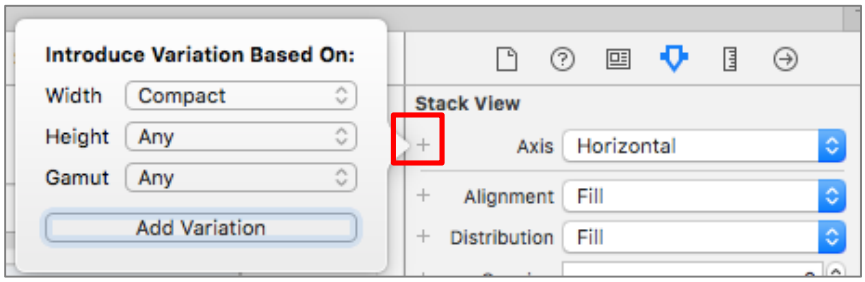




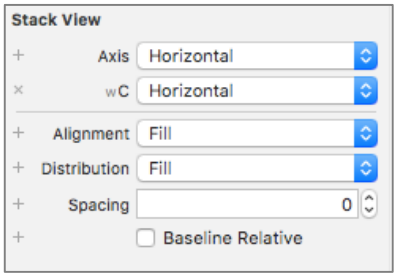
33. Now, we will want iOS to display our Horizontal Stack Views as Vertical Stack Views when the width is compact. To do so, hold down your Command key, and select all Horizontal Stack Views.



34. Click the Attributes Inspector, and the **+** sign to the left of the **Axis** property.

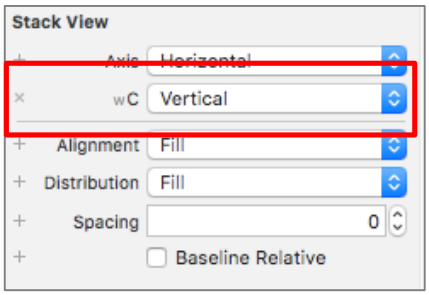


35. Since we are adding a variation based on devices / orientations with compact width, the conditions in the pop-over are exactly what we need. So we click **Add Variation**. When you do that, you will see the following appear in the Attributes Inspector.

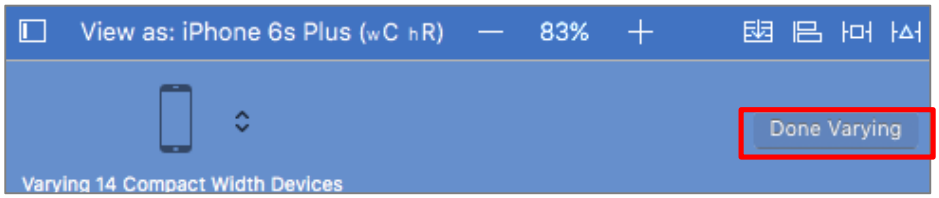


This means that the Axis property will take on a different value when the device / orientation has a compact width.

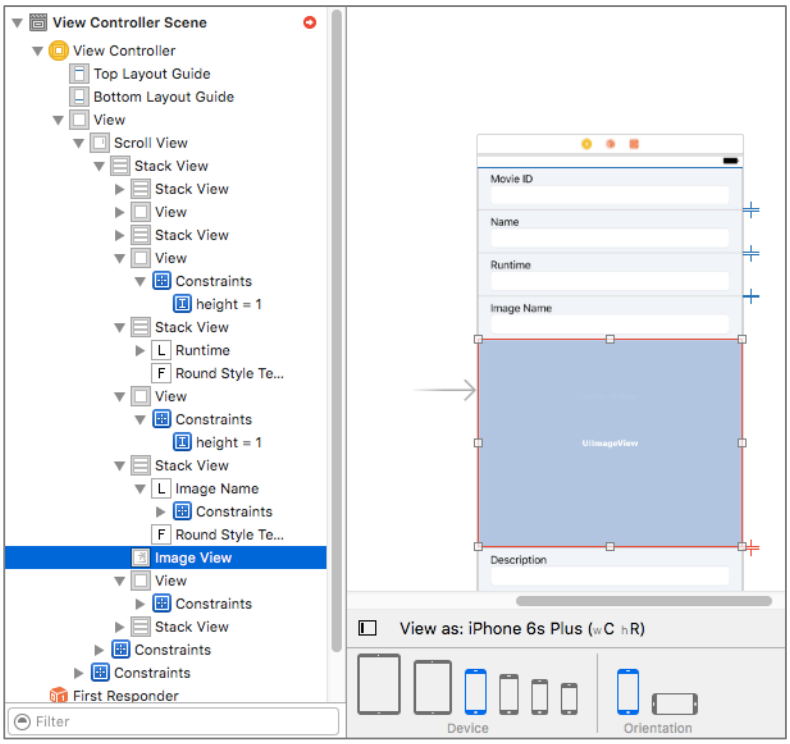
36. Here, we will change the Axis for wC to **Vertical**.



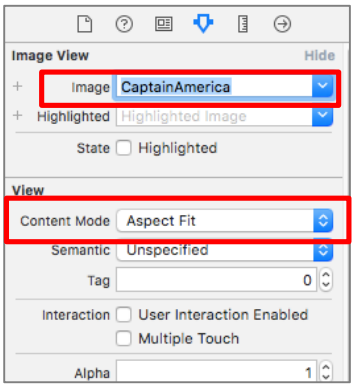
37. Click on **Done Varying**.



38. Back in the default layout, try switching the devices / orientations and see how the layouts in your Storyboard changes.
39. Download any image from the internet and add it into your **assets.xcassets**.
40. Add a new Image View between your 4th Horizontal Stack View and the next View, in this way:



41. Click the Attributes Inspector for the image view and select the image you have added into your assets.xcassets, and set the Content Mode to **Aspect Fit**.



42. Try to run your app in an iPhone 6 simulator and landscape mode and see the scroll view in action.

Challenge

1. Making use of the above, are you able to design your app using purely Trait Variations, Stack Views and Auto Layout Constraints such the ImageView appears in the following orientations: