Lab 02: Collection Views in iOS

Lab Goals

The goal in this lab will be to explore Collection Views as a way to provide complex layout of multiple screen items and learn how to work with them. Topics covered include:

- Collection View - Background and Selected Views
- Collection View - Supplementary View
- Collection View - Decoration View - Highlighting
- Collection View - Building custom cells and layout views
- Collection View - Decoration View - Background

The code introduced in this chapter is already present in the samples, but it's commented out. Remember you can use *cmd + /* hot key to comment and uncomment whole sections of code at a time by selecting the lines of code and then using the hot key. All of the lab steps have been commented with the comments similar to `// TODO: Step1 - instructions`. It is very easy to locate the code in the labs by using the Tasks Pane. Open the Task Pane by clicking on the Tasks icon in the lower right of Xamarin Studio.

Steps

Start up your environment

1. Launch Xamarin Studio.
2. File > Open...
3. Navigate to /Tables and Collection Views in iOS/Lab 02 resources/Lab 02 Begin/ and double-click on the CollectionViewsiOS solution.

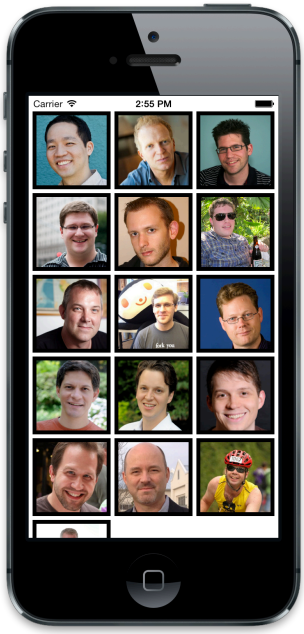
This solution has code implementing a Collection View of speakers from the Xamarin's Evolve conference. All of the code is already present in the solution, but some of the key parts have been commented out so we can learn by enabling the code as we go. In this lab we'll take the approach of explaining the lab steps as we do them and then we'll view the results in the iOS Simulator.

Demo1 – building a basic Collection View

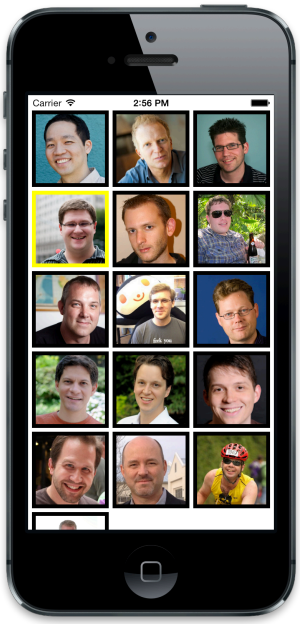
Start by making sure the first project (CollectionViewsiOS_demo1) is the default. Right-click on the project name and choose **Set As Startup Project**.

1. If you click the Play button to build and run the project before we start, the iOS Simulator will launch and display a blank screen. We need to implement the collection view by uncommenting the code present in the example to make it work.

2. Double-click on **CollectionViewController.cs** and uncomment the section of code under `//TODO: Step 1a: create and initialize a UICollectionViewFlowLayout` by using the cmd +/ hot key:
3. File > Save All
4. Click the Play button to build and run the project. The iOS Simulator will launch and display a Collection View with a black frame around the images. Setting the Background View to black and scaling the image down by 10% achieved this effect. The Collection View should now resemble:



5. Tap one of the images and the yellow Selected Background view should be visible:

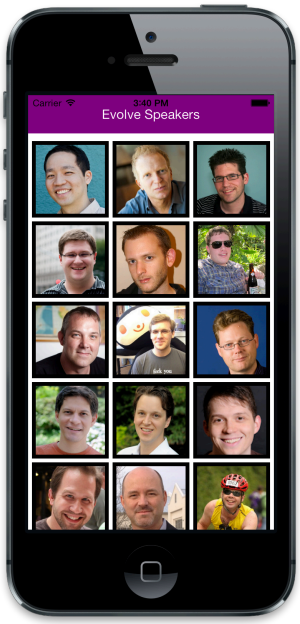


6. Press the Stop button to halt the iOS Simulator.

Demo2 - Header Supplementary View

Let's add a Supplementary View to act as a header for our Collection View.

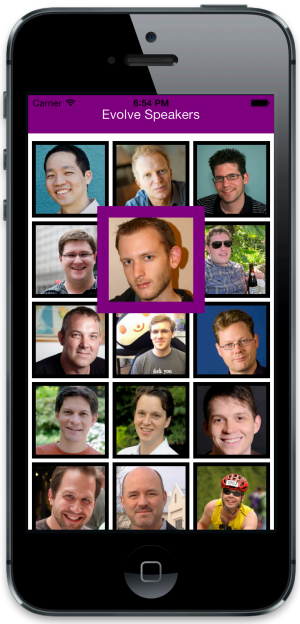
1. Open Header.cs and uncomment the entire class declaration under the comment `//TODO: Step 2a: uncomment to implement a class that builds a header Supplementary View`
2. Open **CollectionViewController.cs** and uncomment the section of code under `//TODO: Step2b: uncomment to register the class for the Supplementary View` to register the Supplementary class so it may be reused.
3. Uncomment all of the code under `//TODO: Step 2c: uncomment to get a header instance to use for the Supplementary View` to create the Supplementary View to be used as a header.
4. Double-click on **AppDelegate.cs** to open it.
5. Uncomment all of the code under `Step 2d: uncomment the next 2 lines to adjust for adding "header" Supplementary View` to uncomment the class to use for the Supplementary View.
6. File > Save All
7. Click the Play button to build and run the project. The iOS Simulator will launch and display our Collection View and should resemble the following:



8. Press the Stop button to halt the iOS Simulator.

Demo3 - Decoration View - Highlighting

1. Double-click on **EvolveCollectionViewController.cs** and uncomment the section of code under `//TODO: Step 3a: uncomment to enable highlighting during a touch and unhighlight on touch up.`
2. File > Save All
3. Click the Play button to build and run the project. The iOS Simulator will launch.
4. Click and hold on one of the images and `ItemHighlighted ()` will trigger magnifying the image slightly and surrounding it with a purple border. The highlighted image should resemble the following:



5. Lift your "finger" off the image and the `ItemUnhighlighted` method will return the highlighted image to its original state.
6. Press the Stop button to halt the iOS Simulator.

Demo4 – Custom layout

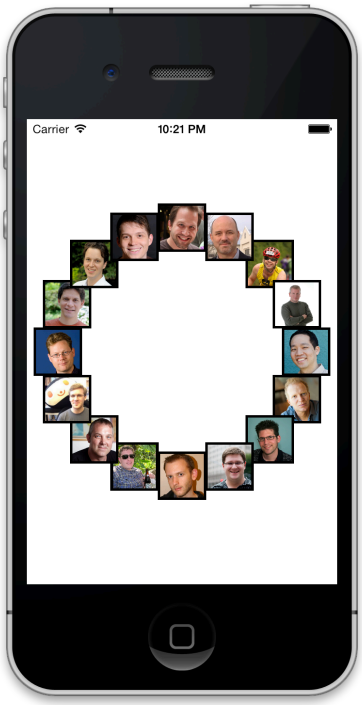
1. Double-click on **CustomLayout.cs** and uncomment all of the code in the section under `//TODO: Step 4a: uncomment code building custom, circular Collection View Layout`

The code in this file creates a circular layout of the speaker images.

2. Double-click on **AppDelegate.cs** and uncomment all of the code in the section of under `//TODO: Step 4b: uncomment to implement a left swipe gesture to switch between Collection View Layouts.`

The code in this section creates a `UISwipeGestureRecognizer` to detect a left swipe that we use to switch between the default `UICollectionViewFlowLayout` and the custom Circular layout we just uncommented.

3. File > Save All
4. Click the Play button to build and run the project. The iOS Simulator will launch displaying the same grid style Collection View.
5. Somewhere on the Collection View screen, do a *Left Swipe* (touch somewhere on the right side of the screen, drag straight-left across the screen, and then lift your finger off the screen.) If the system detected a left swipe the screen layout should resemble the following:



Demo5 – Background decoration view

1. Double-click on **CustomDecorationView.cs** and uncomment the entire class definition under `//TODO: Step 5a: build a background view with a MapView`
2. Double-click on **CustomLayout.cs**
3. Uncomment the RegisterClassForDecorationView declaration under `//TODO: Step5b: register the CustomDecorationView`
4. Uncomment method declaration for LayoutAttributesForDecorationView under `//TODO: Step 5c: add layout attributes for decoration view` so that the background decoration view's layout attributes are available to the collection view when it renders.
5. In the LayoutAttributesForElementsInRect method, uncomment the lines under `//TODO: Step 5d` and `//TODO: Step 5e`. This adds the background decoration view into the list of layout attributes returned by this method
6. File > Save All
7. Click the Play button to build and run the project. The iOS Simulator will launch displaying the same grid style Collection View.
8. Somewhere on the Collection View screen, do a *Left Swipe* (touch somewhere on the right side of the screen, drag straight-left across the screen, and then lift your finger off the screen.) When the collection view layout changes from the grid to the custom circle, you should now see the map of Austin, Texas appear in the background (as shown below).

Congratulations!

In this lab, we implemented Collection Views including Background and Selected Views. We also implemented a Supplementary View that served as a header for our Collection View. Next, we added highlighting to our cells. Finally, we implemented a custom Collection View Layout and a gesture recognizer to switch between a typical grid cell layout and a custom, circular cell layout using a Decoration View with a map view as a background.