Camera App Part 4

Add a CollectionView

1. Drop a Collection View on to your Storyboard

   > CAUTION: DO NOT drop the Collection View inside of the Scroll View. Drop it below it.

2. Add the following Constraints to the Collection View

   a. Vertical Spacing to Bottom Layout Guide with a value of **0** (Zero)

   b. Leading Space to Container with no Margin and a value of **0** (Zero)

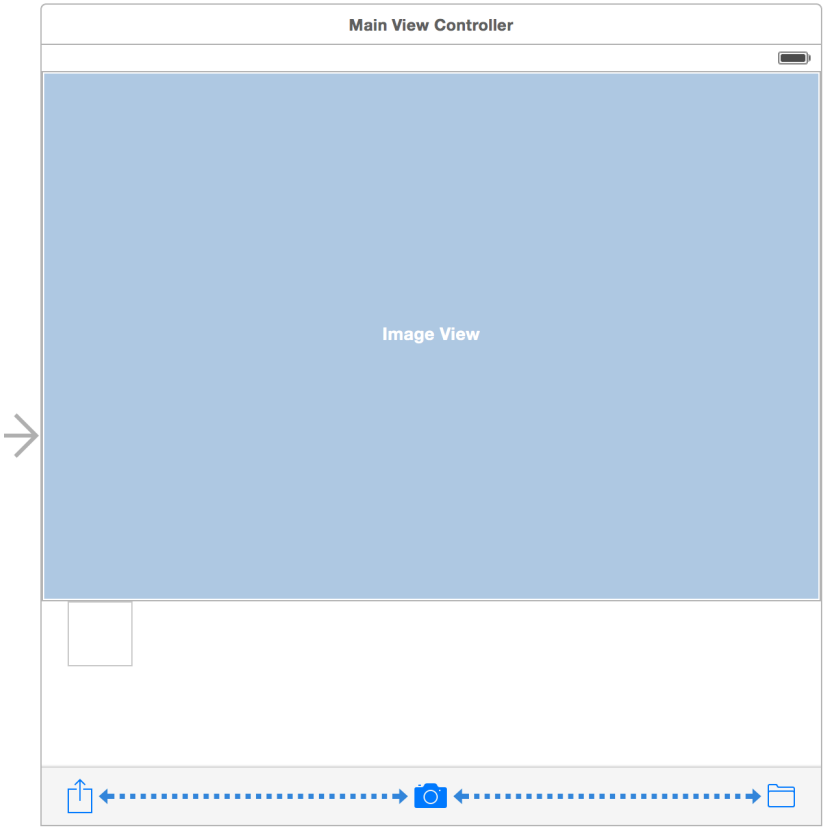
   c. Trailing Space to the Container with no Margin and a value of **0** (Zero)

   d. Height with a value of **170**

Change the Constraints of the Scroll View to Accommodate the Collection View

1. Remove the *Bottom Space to: Bottom Layout Guide* constraint
2. Add a new *Vertical Spacing* Constraint to the Collection View with a value of 0 (Zero)

At this point, your Storyboard should look similar to this:



Set the Main View Controller as the Data Source for the Collection View

1. In the *Document Outline* in the Storyboard, Ctrl+Drag the Collection View to the Main View Controller and select *DataSource* from the context menu

Create an Outlet for the Collection View

1. Create an IBOutlet for the Collection View in the Main View Controller and name it *previewCollectionView*

Adjust the size of the Cell

Notice that there is a square in the top left of the Collection View. That is your Collection View Cell.

1. Select the Collection View
2. In the *Size Inspector*, change the cell size from Width:50 Height:50 to **Width:128 Height:169.5**

Add an Image View to the Cell

1. Drag an Image View on to the Cell
2. Add the following Constraints to the Image View

   a. Top Space to the Container Margin (of the Collection View Cell) with a value of **0** (Zero)

   b. Equal Widths to the Collection View with a value of **0** (Zero)

   c. Center Horizontally in Container with a value of **0** (Zero)

   d. Height with a value of **128**

Create a Custom Cell Class

1. Create a new UICollectionViewCell (Cocoa Touch) class and name it *PreviewCollectionViewCell*
2. Go back to the Collection View Cell in the Storyboard. In the *Identity Inspector*, set the Class to *PreviewCollectionViewCell*
3. In the *Attributes Inspector* set the *Identifier* property to *PreviewCellReuseID*

Create an Outlet for the Image View

1. With the Collection View Cell selected in the Storyboard, open the *Assistant Editor*
2. Ensure that the file that is associated with the
3. Create an IBOutlet for the Image View in the Collection View Cell and name it *previewImageView*
4. Close the *Assistant Editor*

Create a Mutable Array to House the Images the User Takes

1. Add a new private Array of UIImages named *imageStore* to the Main View Controller

```
private var imageStore : [UIImage]!
```

2. Initialize the *imageStore* in the *viewDidLoad* method.

```
override func viewDidLoad() {

    self.imageStore = [UIImage]()

    ...
}
```

Implement the Collection View's Data Source

1. Open the *MainViewController.swift* file
2. Set the MainViewController class to implement the *UICollectionViewDataSource* protocol.

```
class MainViewController: UIViewController,
UIImagePickerControllerDelegate, UINavigationControllerDelegate,
UICollectionViewDataSource {
```

> NOTE: Collection Views work very similarly to Table Views. They have a Datasource and Delegate protocol too.

Implement Data Source Methods

1. Add the following two empty methods

```
func collectionView(collectionView: UICollectionView,
cellForItemAtIndexPath indexPath: NSIndexPath) ->
UICollectionViewCell {

}

func collectionView(collectionView: UICollectionView,
numberOfItemsInSection section: Int) -> Int {

}
```

> NOTE: These two methods mimic the Table View Data Source methods, *tableView:cellForRowAtIndexPath:* and *tableView:numberOfItemsInSection*

2. In the *collectionView:numberOfItemsInSection:* method, return the count of the imageStore

```
func collectionView(collectionView: UICollectionView,
numberOfItemsInSection section: Int) -> Int {

    return self.imageStore.count
}
```

3. In the *collectionView:cellForItemAtIndexPath:* method, add the UIImage at the indexPath provided to the PreviewCollectionViewCell

Now that we have a collection view showing the data, let's put some data in the Image Store

Add New Images to Image Store

1. In the *imagePickerController:didFinishPickingImage:* method, add the following code to add the image to the Image Store and refresh the Collection View

```
func imagePickerController(picker: UIImagePickerController,
didFinishPickingImage image: UIImage, editingInfo: [String :
AnyObject]?) {

...

self.imageStore.insert(image, atIndex: 0)

self.previewCollectionView.reloadData()

...

}
```

Implement the Collection View Delegate

To switch between the images, we will implement the Collection View Delegate to capture when the user selected an image in the Collection View

1. Set the MainViewController class to implement the *UICollectionViewDelegate* protocol.

```
class MainViewController: UIViewController,
UIImagePickerControllerDelegate, UINavigationControllerDelegate,
UICollectionViewDataSource, UICollectionViewDelegate {
```

2. Implement the method that's called when a cell is touched

```
func collectionView(collectionView: UICollectionView,
didSelectItemAtIndexPath indexPath: NSIndexPath) {

}
```

3. In the *collectionView:didSelectItemAtIndexPath:* method, set the selected image to be the main one.

```
func collectionView(collectionView: UICollectionView,
didSelectItemAtIndexPath indexPath: NSIndexPath) {

    let image = self.imageStore[indexPath.item]

    self.displayImageView.image = image
}
```

Tidy Up the UI to Make it Cleaner Looking

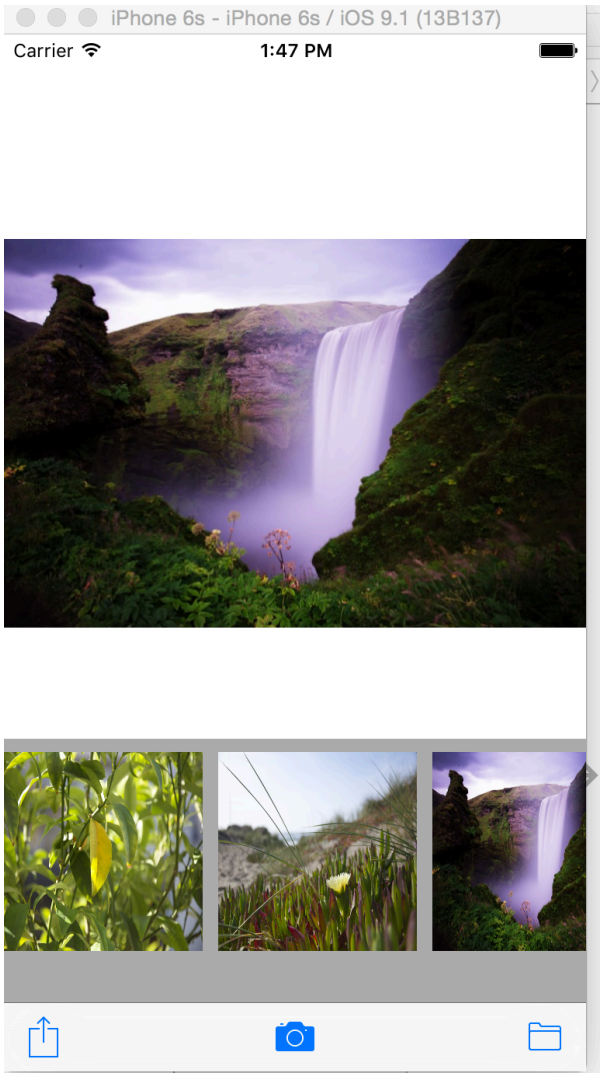
1. Set the background of the Collection View *Light Gray Color*
2. Change the *Scroll Direction* of the Collection View to be **Horizontal**

   a. In the Attributes Inspector of the Collection View, find *Scroll Direction* and change its value to *Horizontal*
3. If the Image Store is empty, do not show the Collection View

   a. Set the *alpha* property of the Collection View to **0.0** in the *viewDidLoad* method

   b. Set the *alpha* property of the Collection View to **1.0** after we add an image to the Image Store

At this point, your app should look similar to this:


iPhone 6s - iPhone 6s / iOS 9.1 (13B137)
Carrier
1:47 PM