EE 443/593

Mobile Application Development
Spring 2018

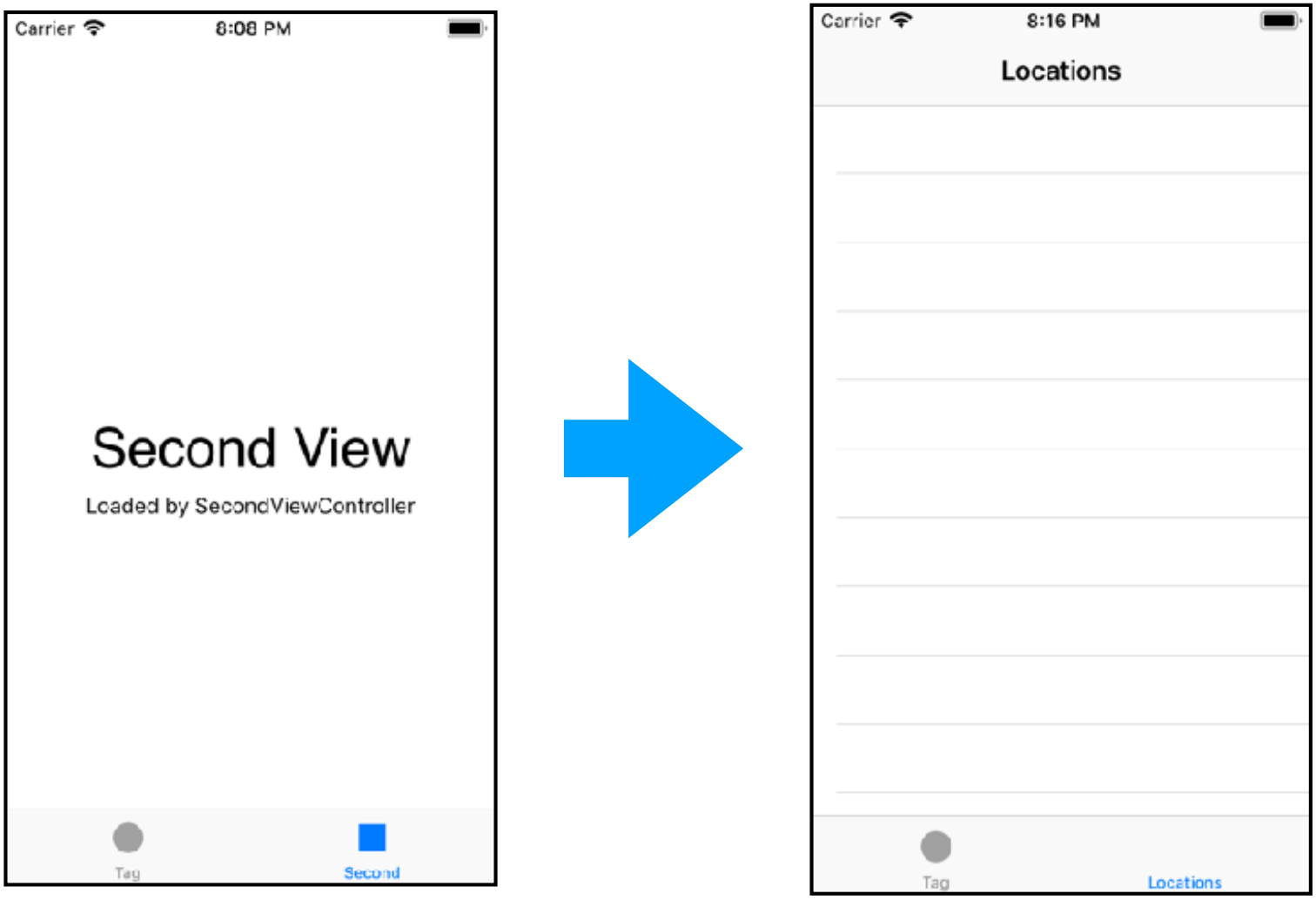
Chapter 28:
The Locations Tab

Cayden Wilson

Outline

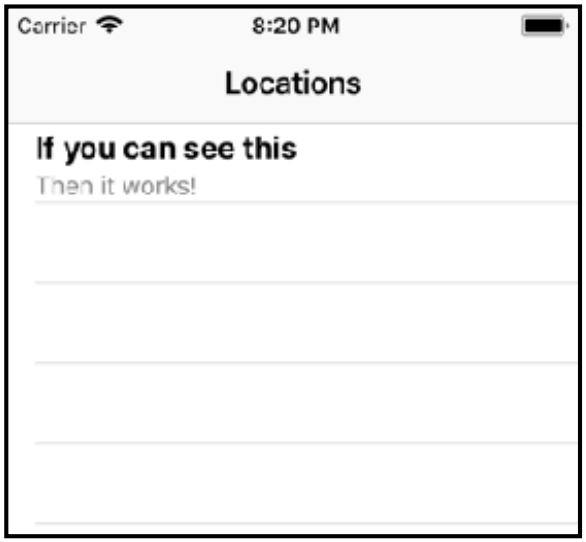
- Locations tab
- Create custom table view cell subclass
- Edit locations
- NSFetchedResultsController
- Delete Locations
- Table view sections

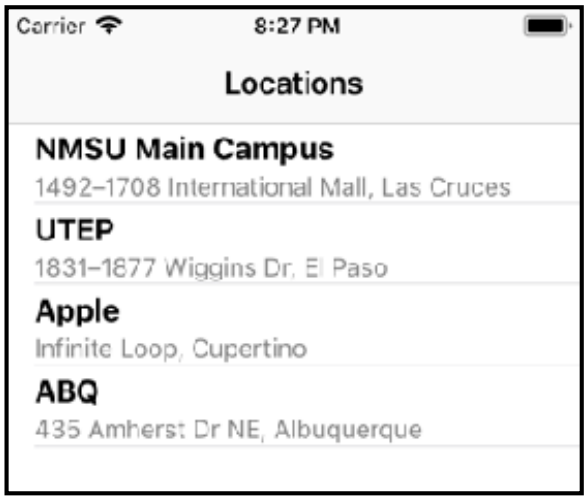
Locations Tab



Create custom table view cell subclass

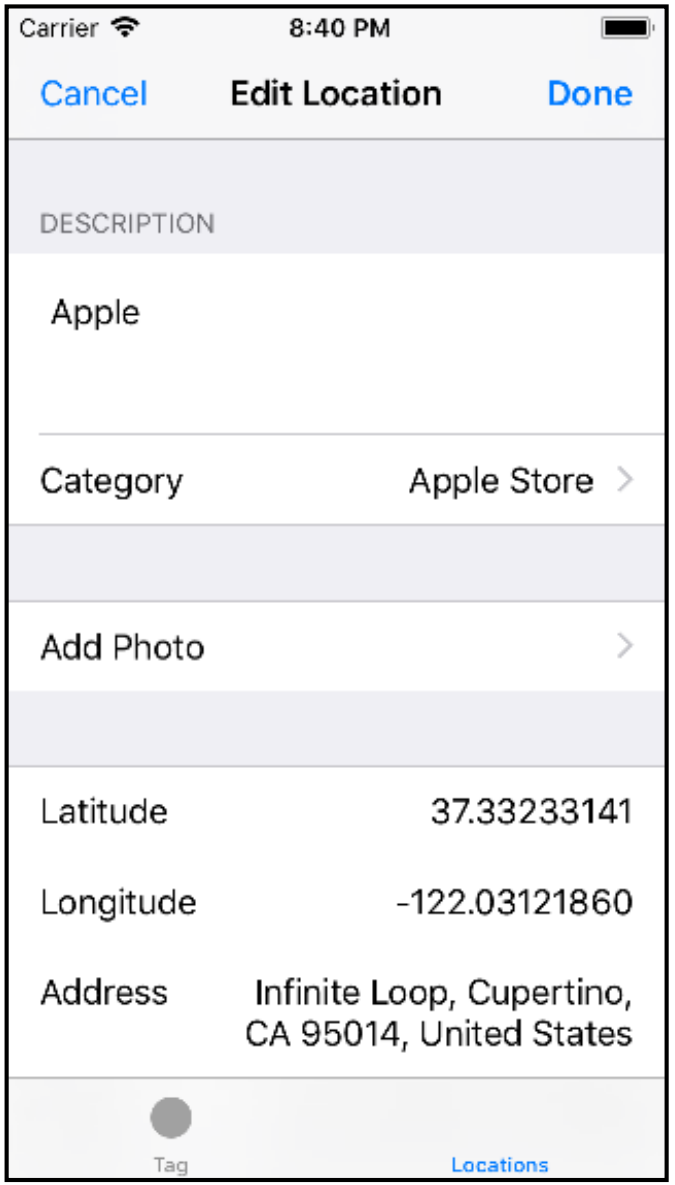
- Design the table view cell
- The basic table view controller
- Display Locations from data store
- Create a custom table view cell subclass





Edit Locations

- Create edit segue
- Set up the edit view controller
- Fix the edit screen



NSFetchedResultsViewController

- Use NSFetchedResultsViewController

```
lazy var fetchedResultsController:
    NSFetchedResultsController<Location> = {
      let fetchRequest = NSFetchRequest<Location>()

      let entity = Location.entity()
      fetchRequest.entity = entity

let fetchedResultsController = NSFetchedResultsController(
        fetchRequest: fetchRequest, managedObjectContext:
self.managedObjectContext,
        sectionNameKeyPath: "category", cacheName:
“Locations")
```

NSFetchedResultsViewController -Extensions

```
extension LocationsViewController:
NSFetchedResultsControllerDelegate {

  func controllerWillChangeContent(_ controller:
NSFetchedResultsController<NSFetchRequestResult>) {
    print("*** controllerWillChangeContent")
    tableView.beginUpdates()
  }

  func controller(_ controller:
NSFetchedResultsController<NSFetchRequestResult>,
                  didChange anObject: Any, at indexPath:
IndexPath?, for type: NSFetchedResultsChangeType,
                  newIndexPath: IndexPath?) {
```

NSFetchedResultsViewController -Extensions (cont.)

```
{
switch type {
    case .insert:
      print("**** NSFetchedResultsChangeInsert (object)")
      tableView.insertRows(at: [newIndexPath!], with: .fade)

    case .delete:
      print("*** NSFetchedResultsChangeDelete (object)")
      tableView.deleteRows(at: [indexPath!], with: .fade)

    case .update:
      print("*** NSFetchedResultsChangeUpdate (object)")
      if let cell = tableView.cellForRow(at: indexPath!) as?
LocationCell {
        let location = controller.object(at: indexPath!) as! Location
        cell.configure(for: location)
      }

    case .move:
      print("*** NSFetchedResultsResultsChangeMove (object)")
      tableView.deleteRows(at: [indexPath!], with: .fade)
      tableView.insertRows(at: [newIndexPath!], with: .fade)
  }
}
```

Delete Locations

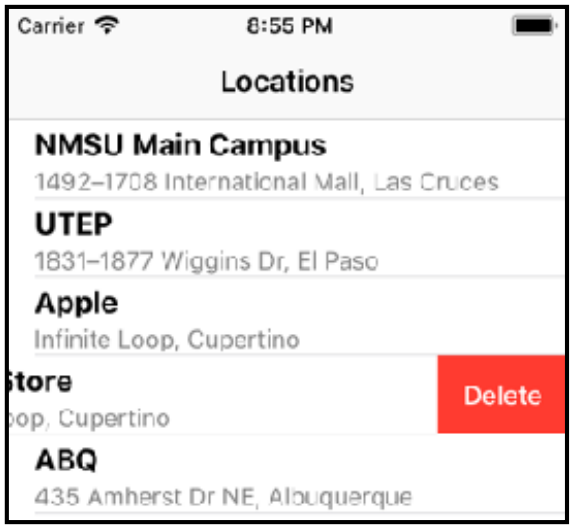
- Two ways
  - Swipe left

```
  override func tableView(_ tableView: UITableView,
commit editingStyle: UITableViewCellEditingStyle,
forRowAt indexPath: IndexPath) {

    if editingStyle == .delete {
      let location =
fetchedResultsController.object(at: indexPath)
      managedObjectContext.delete(location)
      do {
        try managedObjectContext.save()
```

  - "Edit" button

```
navigationItem.rightBarButtonItem = editButtonItem
```



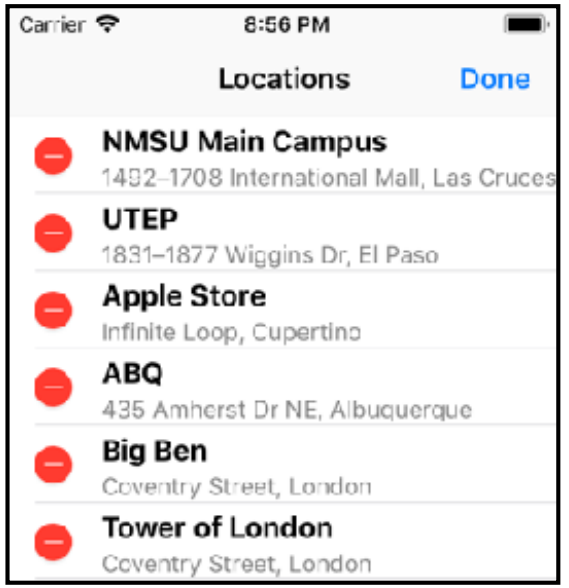
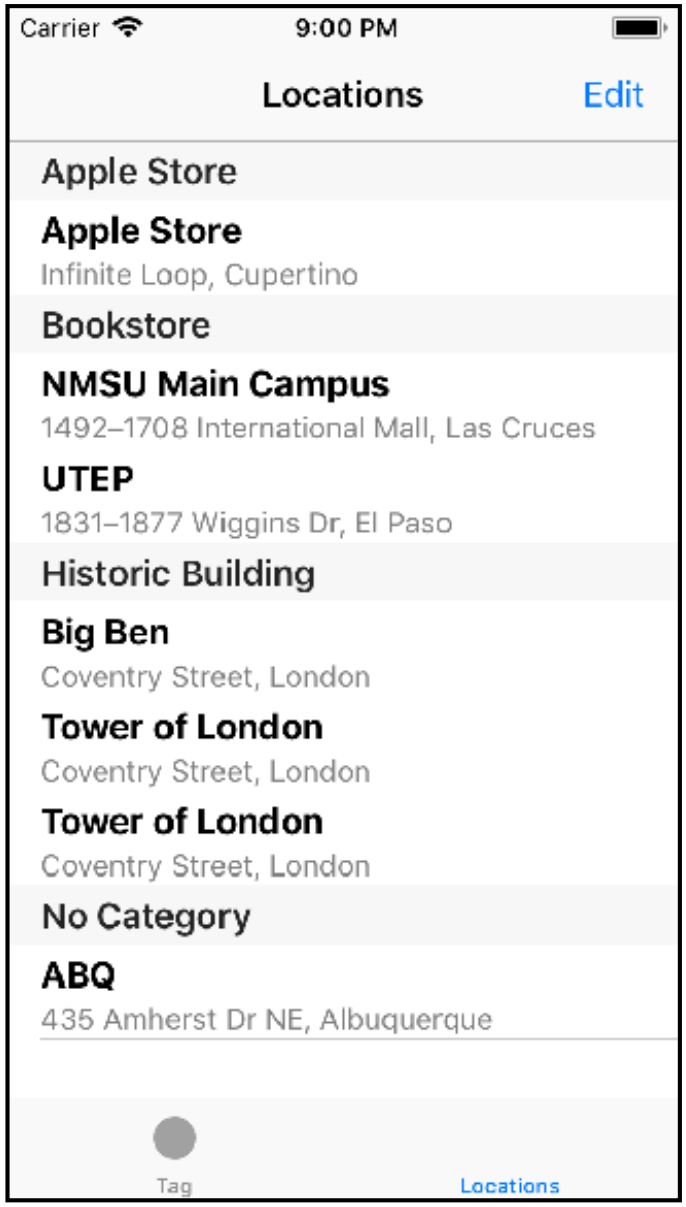


Table View Sections

- Sections help to sort cells based on a value
- “Category”, “Alphabetical”, “Date”, etc
- Allows easy navigation when more data then can be displayed

```
override func tableView(_ tableView: UITableView,
                        titleForHeaderInSection
section: Int) -> String?
```



Graphing in iOS