Collection Views

Dr. Sarah Abraham

University of Texas at Austin
CS329e
Spring 2019
What is a Collection View?

- Presents an ordered set of data items in a flexible layout
- Subclass of UIScrollView (like UITableView)
- Generalized notion of a table view
  - Uses view cells
  - Displays from a data source
  - Not limited to tabular rows
Collection View Example
Collection View Controller

- UICollectionView is an available class
- Also possible to hook UICollectionView into a standard UIViewController
  - Create IBOutlet for collection view in view controller source code
  - Connect outlet to view controller in Interface Builder
  - Connect collection view’s data source and delegate to parent view controller
class ViewController: UIViewController, UICollectionViewDataSource, UICollectionViewDelegate, UICollectionViewDelegateFlowLayout {

    @IBOutlet weak var collectionView: UICollectionView!

    /* String identifier used to dynamically load cells into collection */
    let identifier = "CellIdentifier"

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
Collection View in IB
UICollectionViewDataSource

- Delegate to feed content into collection view
- Cell must have matching identifier in both Xcode and IB
- Returns information about items in the collection
- Required methods for implementation:
  - collectionView(_: numberOfItemsInSection:)
  - collectionView(_: cellForItemAtIndexPath:)

(collectionView(_: numberOfItemsInSection:))
(collectionView(_: cellForItemAtIndexPath:))
Data Source Protocol

- `func collectionView(_: collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int`
  - Data source provides collection with number of items in given section

- `func collectionView(_: collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell`
  - Data source provides collection with the cell at indexPath
Other Functions

- `numberOfSections(inSection:)`
  - Data source provides total number of sections in the collection view

- `collectionView(_:viewForSupplementaryElementOfKind:atIndexPath:)`
  - Data source provides a supplementary view to display in the collection view
    - Used for adding header and footer content

- `collectionView(_:moveItemAt:to:)`
UICollectionViewDelegate

- Handles events that interact with the cells
  - Cell selection
  - Cell highlighting
  - Addition and removal of views
  - Layout changes
  - Cell actions
  - Collection view focus
Collection View Layout

- UICollectionViewFlowLayout
- Collection views not restricted to tabular format
- Grid is default but can be modified for great deal of flexibility
- Contains delegate protocol: UICollectionViewDelegateFlowLayout
  - Defines layout appearance
  - Cell-spacing, scroll direction etc
Flow Layout Delegate Methods

- All optional
- Set dimensions of specified cell
- Set margins around a section
- Set margins between items (rows and columns)
- Set size of header and footer around section
func collectionView(UICollectionView, layout: UICollectionViewLayout, insetForSectionAtIndex: Int)

func collectionView(UICollectionView, layout: UICollectionViewLayout, minimumLineSpacingForSectionAt: Int)

func collectionView(UICollectionView, layout: UICollectionViewLayout, minimumInteritemSpacingForSectionAt: Int)
UICollectionViewFlowLayout

//MARK: UICollectionViewFlowLayout

func collectionView(_ collectionView: UICollectionView, layout collectionViewLayout: UICollectionViewLayout, sizeForItemAt indexPath: IndexPath) -> CGSize {
    return CGSize(width: 60, height: 60)
}

func collectionView(_ collectionView: UICollectionView, layout collectionViewLayout: UICollectionViewLayout, insetForSectionAt section: Int) -> UIEdgeInsets {
    return UIEdgeInsetsMake(0, 0, 0, 0)
}
CollectionView Cells

- UICollectionViewCell displays content
- Added as subviews to collection view
- Subclass to make cell appearance customizable
  - Create in Xcode
  - Customize in Interface Builder
- Must register cell with reuseIdentifier
func collectionView(collectionView: UICollectionView, cellForItemAtIndexPath indexPath: NSIndexPath) -> UICollectionViewCell {

    let cell = collectionView.dequeueReusableCellWithIdentifier(identifier, forIndexPath: indexPath) as! UICollectionViewCell

    return cell

}