EE443 / EE593
Mobile Application Development

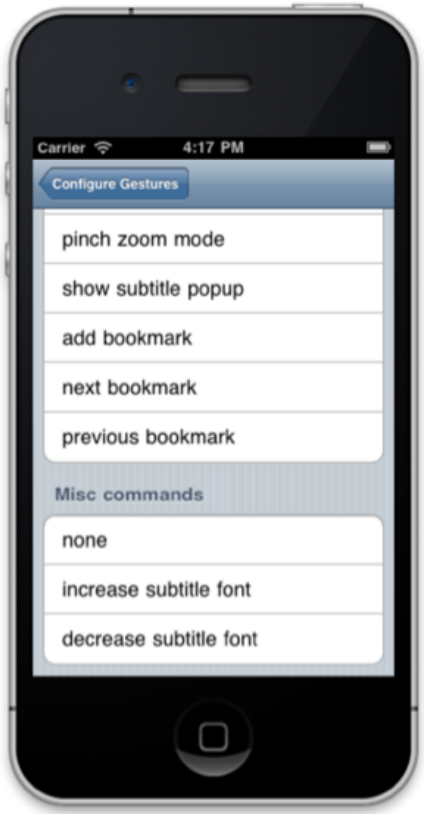
Kousei Richeson

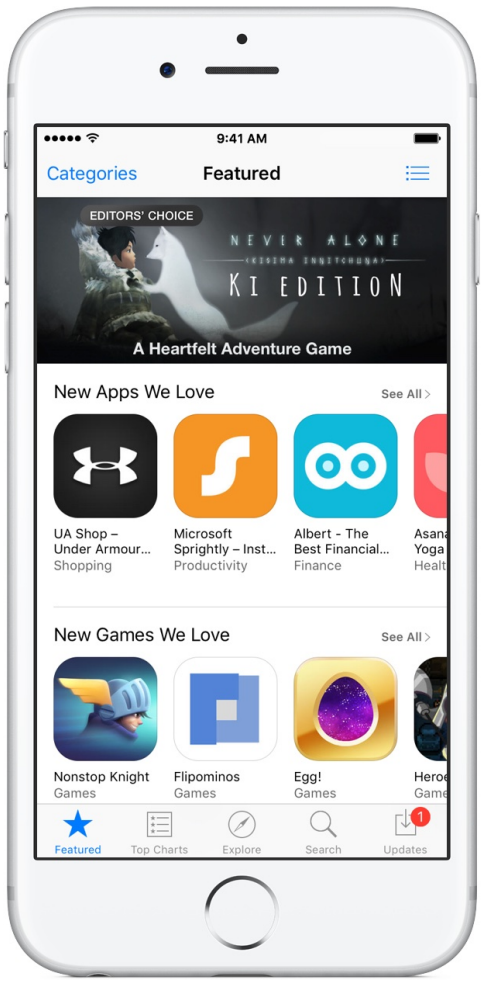
Lecture 6:
Chapter 9: Table Views

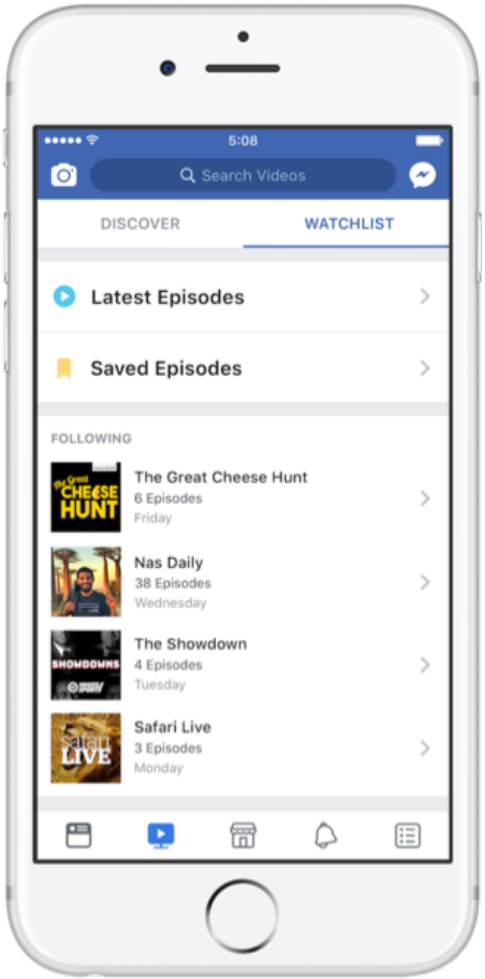
Kousei Richeson

What are Table Views?

- First introduced in iOS 2.0 (2008)
- Continually improved each year
- It is an object in UIKit
- Displays data in lists
- Inherits from UIScrollView
- Each Table View row is a **UITableViewCell**
- Editable & Removable




9:41 AM
Categories
Featured
EDITORS' CHOICE
NEVER ALONE
KI EDITION
A Heartfelt Adventure Game
New Apps We Love
See All
UA Shop – Under Armour...
Shopping
Microsoft Sprightly – Inst...
Productivity
Albert - The Best Financial...
Finance
New Games We Love
See All
Nonstop Knight
Games
Flipominos
Games
Egg!
Games
Featured
Top Charts
Explore
Search
Updates


5:08
Search Videos
DISCOVER
WATCHLIST
Latest Episodes
Saved Episodes
FOLLOWING
The Great Cheese Hunt
6 Episodes
Friday
Nas Daily
38 Episodes
Wednesday
The Showdown
4 Episodes
Tuesday
Safari Live
3 Episodes
Monday

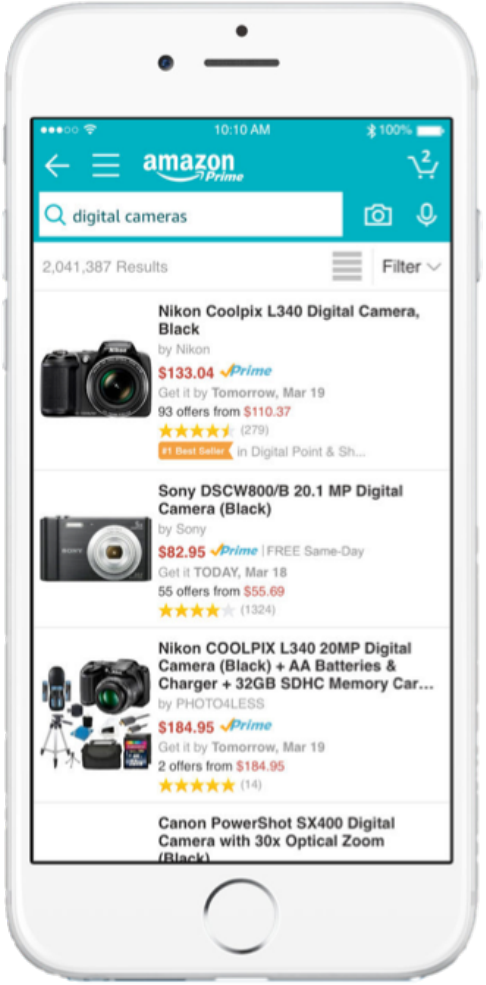

10:10 AM
100%
amazon Prime
digital cameras
2,041,387 Results
Filter
Nikon Coolpix L340 Digital Camera, Black
by Nikon
$133.04 Prime
Get it by Tomorrow, Mar 19
93 offers from $110.37
(279)
in Digital Point & Sh...
Sony DSCW800/B 20.1 MP Digital Camera (Black)
by Sony
$82.95 Prime | FREE Same-Day
Get it TODAY, Mar 18
55 offers from $55.69
(1324)
Nikon COOLPIX L340 20MP Digital Camera (Black) + AA Batteries & Charger + 32GB SDHC Memory Car...
by PHOTO4LESS
$184.95 Prime
Get it by Tomorrow, Mar 19
2 offers from $184.95
(14)
Canon PowerShot SX400 Digital Camera with 30x Optical Zoom

Table Views Sections

- Table Views can have **sections**.
- In the Apple Music app, each letter in the alphabet has its own section.

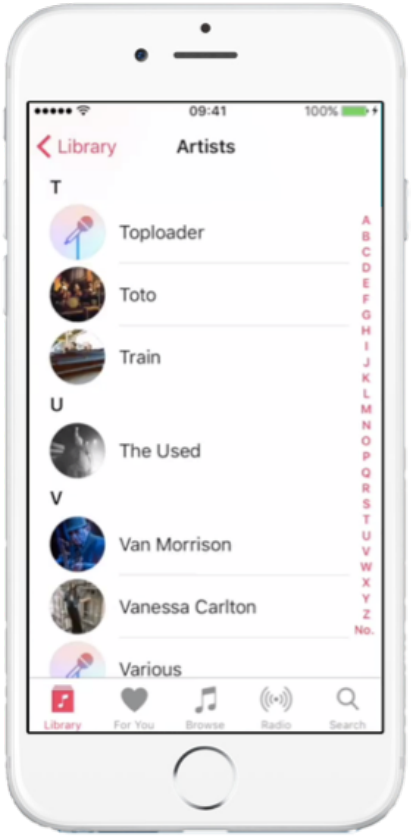
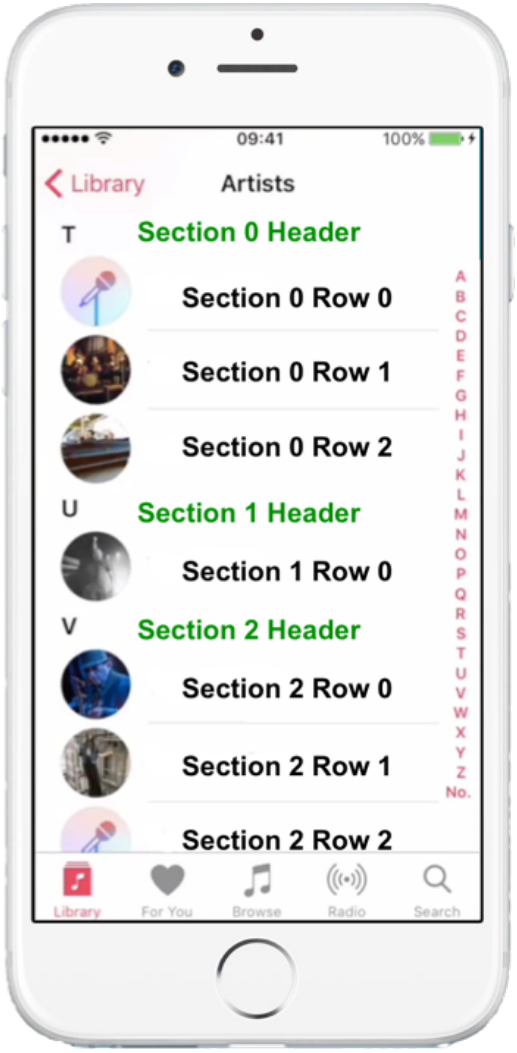


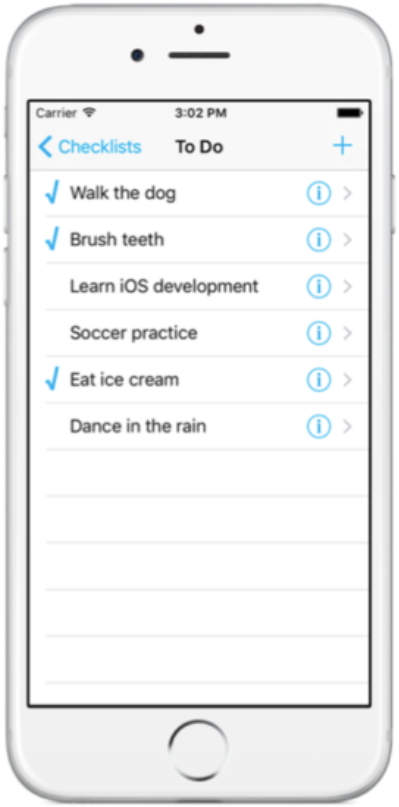
Table Views Rows

- Every row in a Table View has an index path.
- An **index path** is made up of a section and a rows starting from 0.
- You can customize cells!



Project: Checklists App

- In this lecture, you utilize Table Views in order to make the Checklists App!



Programming To-Do List
Chapter 9 App Demo

1. Put a Table View on the app's screen.

2. Conform to Table View protocols.

3. Put data into the table view.

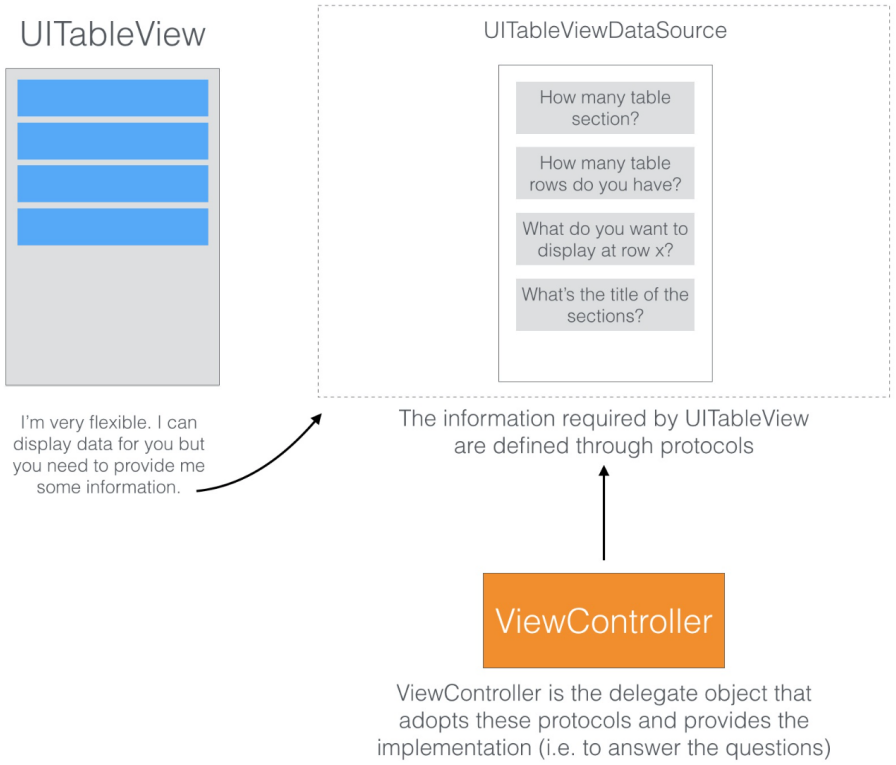
4. Allow the user to tap a row in the table to toggle a checkmark on and off.

What is UITableViewDataSource?

- It is a **required protocol** to display information/data.

- UITableViewDataSource **requires** 2 functions**.**

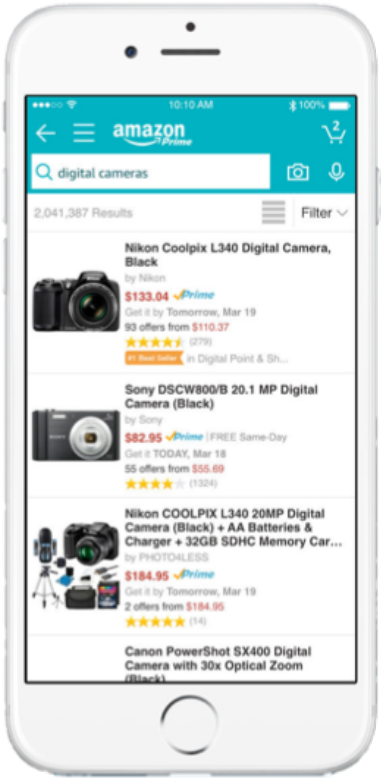
1. cellForRowAtIndexPath( )
2. numberOfRowsInSection( )



DequeueReusableCells

- Every time you return a plain UITableView Cell in cellForRowAtIndexPath( ), you will **create** new cells in memory.

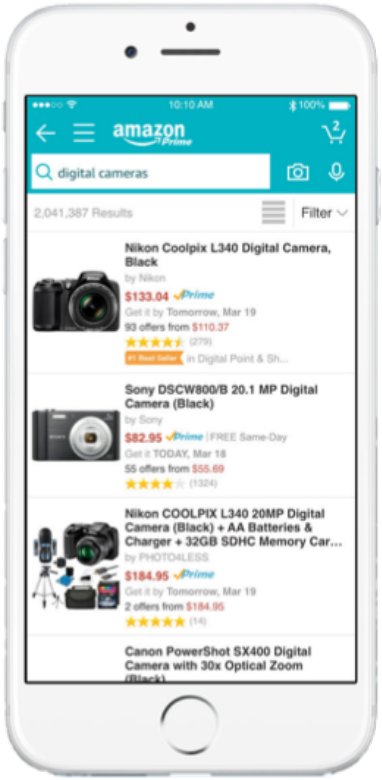
- What do you think will happen if we have 100,000 high quality picture cells in our table view?



DequeueReusableCells (cont.)

- You will run into memory problems and your app will **crash**!

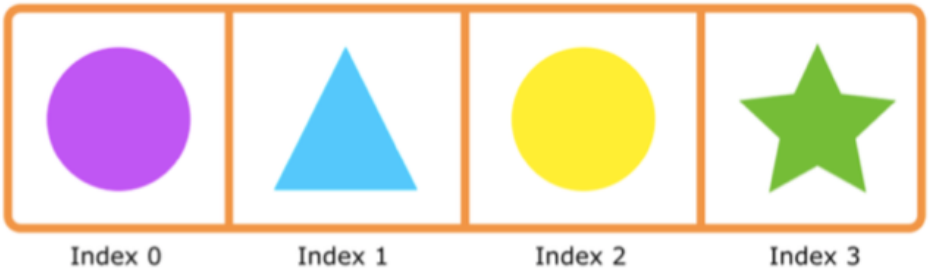
- Use tableView.dequeueReusableCell( ) in order to tell the table view that whenever a cell is off the screen, it will be **replaced** with new data.



Populating Cells with Arrays (Instead of Else-Ifs)

- Instead of manually assigning values to cell text labels, lets use an array to populate it!

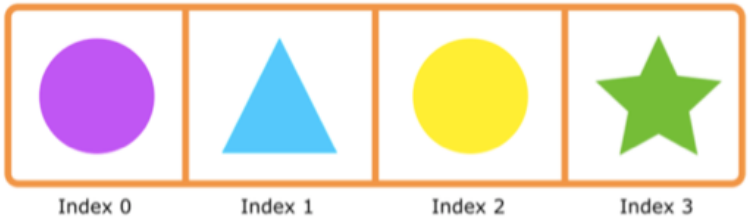
- Change:
  - cell.textLabel?.text = “text”
  - to:
  - cell.textLabel?.text = array[0];



Populating Cells with Arrays (cont.)

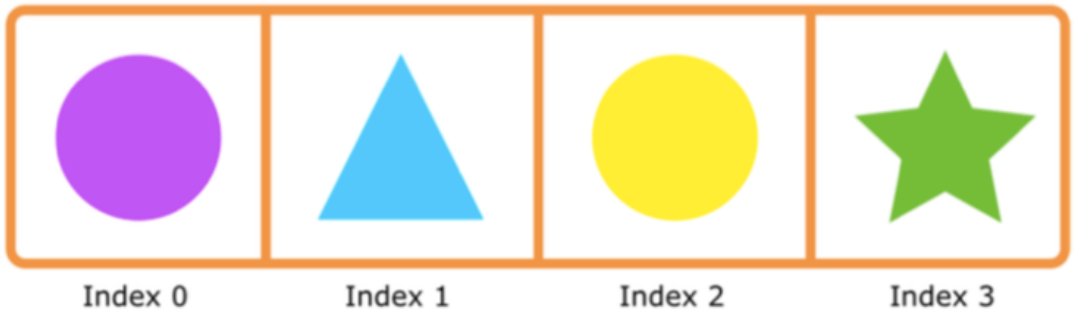
- Now let's **dynamically** set each unique cell with a different index of the array!

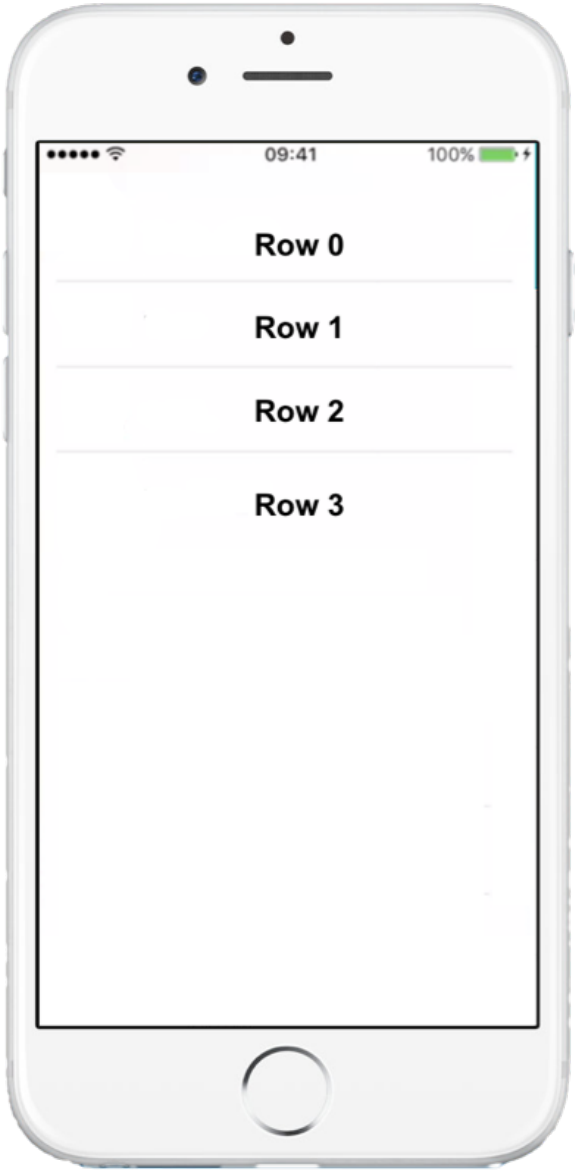
- Change:
  - cell.textLabel?.text = array[0]
  - to:
  - cell.textLabel?.text = array[indexPath.row]



What does this code do?

```
cell.textLabel?.text = array[indexPath.row]
```

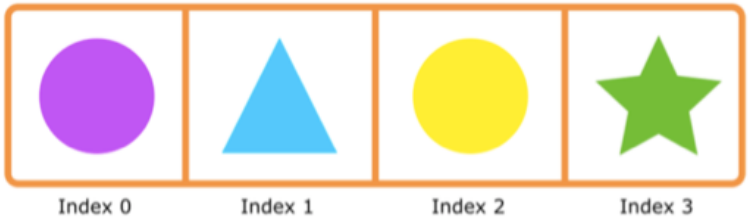




Populating Cells with Arrays (cont.)

- **Why does our app crash!?**

- If there are more rows than the array length, the array will go out of bounds!

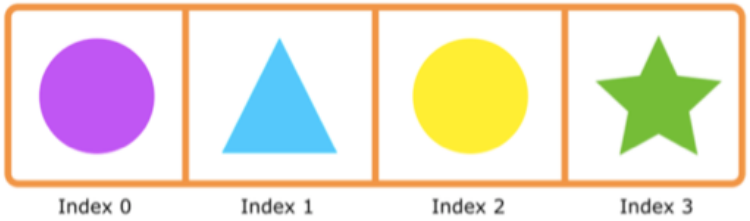


- We need a way to tell the array to loop back to the beginning of the array when it hits the end.

Populating Cells with Arrays (cont.)

- Lets use **Modulus**!

- Modules gives us the remainder of a division problem. (ex: 32 % 6 = 2)

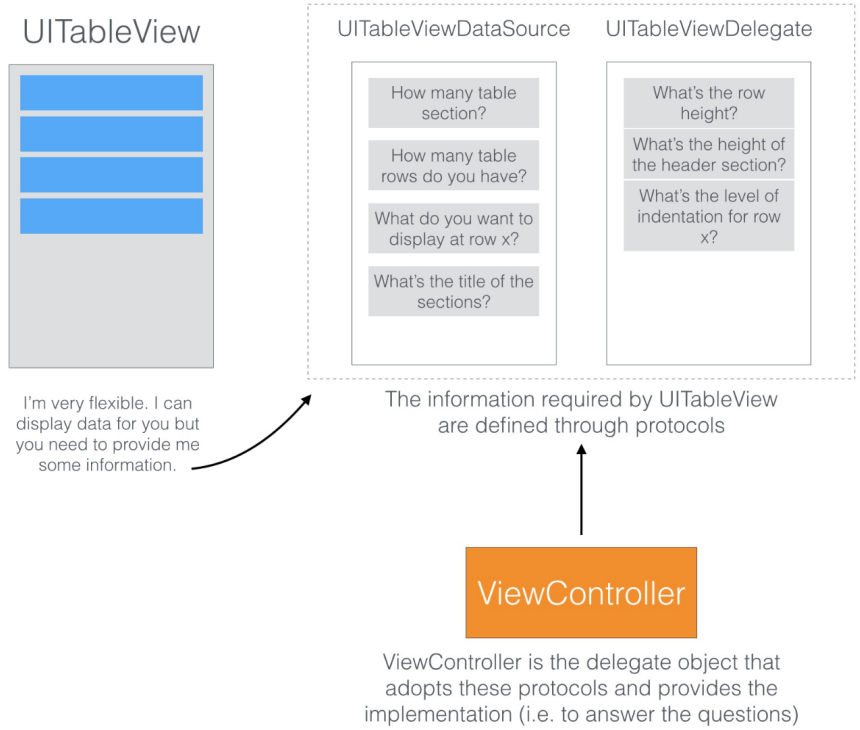


- Use this piece of code:

```
cell.textLabel?.text = array[indexPath.row % array.count]
```

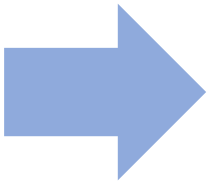
What is UITableViewDelegate?

- It is a **required protocol** for interaction and customization.
- UITableViewDataSource **does not require** any functions**.**



Utilizing Extensions

```swift
class CheckListViewController: UIViewController {

    var emojis = ["🍕🍕🍕", "😃", "😎", "🏆", "🐶", "🔥"]

}


extension CheckListViewController: UITableViewDataSource {

    public func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return 100
    }


    public func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let myCustomCell = tableView.dequeueReusableCell(withIdentifier: "myCell", for: indexPath)
        myCustomCell.textLabel?.text = emojis[indexPath.row % emojis.count]
        return myCustomCell
    }

}


extension CheckListViewController: UITableViewDelegate {

    public func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

        if let cell = tableView.cellForRow(at: indexPath) {

            if cell.accessoryType == .checkmark {
                cell.accessoryType = .none
            } else {
                cell.accessoryType = .checkmark
            }

        }

        tableView.deselectRow(at: indexPath, animated: true)


    }
}
```



```swift
class ChecklistViewController: UIViewController {

    let emoji = ["🍕🍕🍕","😃","😎","☺️","🐶"]

}
```

```swift
extension CheckListViewController: UITableViewDataSource {

    public func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return 100
    }


    public func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let myCustomCell = tableView.dequeueReusableCell(withIdentifier: "myCell", for: indexPath)
        myCustomCell.textLabel?.text = emojis[indexPath.row % emojis.count]
        return myCustomCell
    }

}
```

```swift
extension CheckListViewController: UITableViewDelegate {

    public func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

        if let cell = tableView.cellForRow(at: indexPath) {

            if cell.accessoryType == .checkmark {
                cell.accessoryType = .none
            } else {
                cell.accessoryType = .checkmark
            }

        }

        tableView.deselectRow(at: indexPath, animated: true)


    }
}
```

Utilizing Extensions (cont.)

- It is bad practice to “clump” all your code in the ViewController class.
- Extensions helps increase readability of your code.
- Extend any class by typing:
  - `extension ViewController { }`
- Let’s use this method to separate our protocol functions!

We are Done!

- In chapter 10, we will build a data model to fix our bug!

Any Questions???

What is the Difference Between a TableViewController and a Table View?

- TableViewController sets up your delegate & datasource automatically and gives you some boiler plate code.
- TableViewController are **not** resizable and only saves about 10 minutes in the setup phase.
- You can implement **anything** a TableViewController can implement with an embedded Table View in a normal ViewController
- Table Views **are** resizable

Re-watch this lecture on my GitHub or YouTube Channel!

Check out:

https://github.com/kricheso

or

https://www.youtube.com/watch?v=uhDHlQg1zNo