UITableViews

UITableView

- An object for creating/ managing lists

- Needs two things in order function:
  - delegate (UITableViewDelegate)
  - dataSource (UITableViewDataSource)

UITableViewDelegate

- No **required** functions
- Can be set programmatically in the **ViewController**
- Can be set via Storyboard in **InterfaceBuilder**
- Offers options for managing attributes of a **UITableView**, including:
  - Cell customization (size/design/etc)
  - Header/Footer Views
  - User input (this is a big one!!!) ex:

```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    // Use this function to interpret a cell being pressed
}
```

Using The Delegate

```
import UIKit

class ViewController: UIViewController {
    @IBOutlet private weak var tableView: UITableView!

    override func viewDidLoad() {
        super.viewDidLoad()
        tableView.delegate = self // Can also be set in Storyboard
    }
}

extension ViewController: UITableViewDelegate {
    func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        // Implement method
    }
}
```

UITableViewDataSource

- The bridge between the UITableView and what it's displaying
- Can be set programmatically in the **ViewController**
- Can be set via Storyboard in **InterfaceBuilder**
- **Two required functions**:

```
func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    // Function used to notify the tableView of how many items are in its list
    return model.someArrayOfThings.count
}

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    // Function used to create/decorate UITableViewCells for a given item
    let item = model.someArrayOfThings[indexPath.row]
    let cell = UITableViewCell()
    cell.textLabel?.text = item.text
    cell.detailTextLabel?.text = item.detail

    return cell
}
```

Using The Data Source

```swift
import UIKit

class ViewController: UIViewController {
    @IBOutlet private weak var tableView: UITableView!

    private var model: SomeModel!

    override func viewDidLoad() {
        super.viewDidLoad()
        tableView.dataSource = self // Can also be set in Storyboard
    }
}

extension ViewController: UITableViewDataSource {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        // Return the count of your data set
        return model.someArrayOfThings.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Implement this method to create/ dequeue and return some `UITableViewCell`
        return UITableViewCell()
    }
}
```

Dequeueing Reusable UITableViewCell

- Optimizes performance of **UITableView**
- Used by the delegate to acquire an already allocated cell, in lieu of allocating a new one
- **Requires** a registered identifier
  - Can be set in **Interface Builder**
  - Can be registered programmatically
- **Returns** a **UITableViewCell** object with the associated `identifier` or `nil` if no such object exists in the reusable-cell queue
- Accessed via call to:

```
func dequeueReusableCell(withIdentifier identifier: String) -> UITableViewCell?
```

  - Dequeues an existing cell if one is available or creates a new one using the class or nib file you previously registered. If no cell is available for reuse and you did not register a class or nib file, this method returns `nil`

Using dequeueReusableCell(withIdentifier:)

```swift
import UIKit

final class ViewController: UIViewController {
    @IBOutlet private weak var tableView: UITableView!

    override func viewDidLoad() {
        super.viewDidLoad()
        tableView.dataSource = self
    }
}

extension ViewController: UITableViewDataSource {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return AstrologicalSign.allCases.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let astrologicalSign = AstrologicalSign.allCases[indexPath.row]

        //swiftlint:disable:next force_cast
        let cell = tableView.dequeueReusableCell(withIdentifier: "MyCustomCell") as! MyCustomCell
        cell.nameLabel.text = astrologicalSign.displayName
        cell.descriptionLabel.text = astrologicalSign.description

        return cell
    }
}
```