

CSC 471 / 371
Mobile Application
Development for iOS



Prof. Xiaoping Jia
School of Computing, CDM
DePaul University
xjia@cdm.depaul.edu
[@DePaulSWEEng](https://twitter.com/DePaulSWEEng)

Table & Navigation
Views



Outline

- Table views
 - Static table views
 - Dynamic table views
- Navigation views

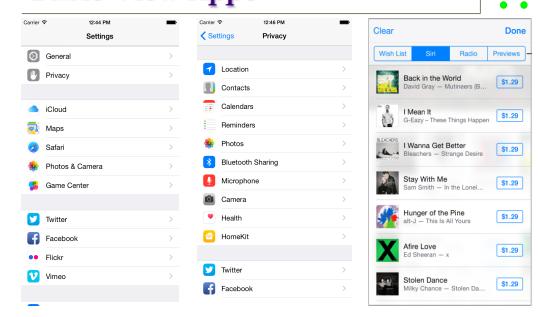


DEPAUL UNIVERSITY 3

Table Views

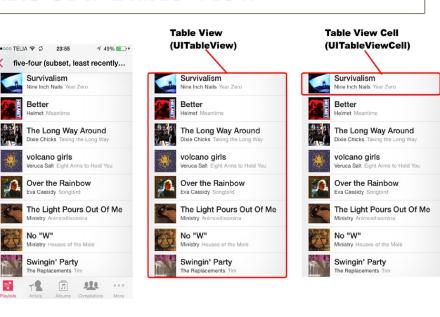


Table View Apps

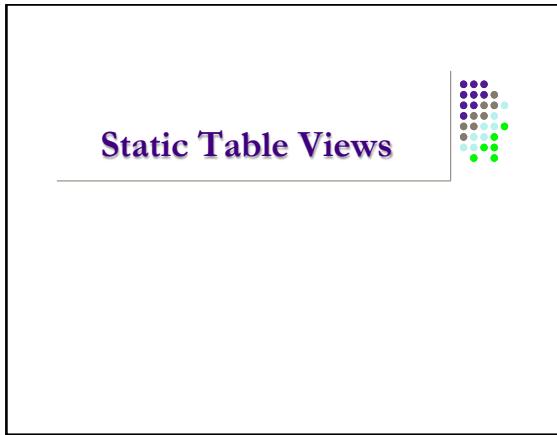


DEPAUL UNIVERSITY 5

Parts of a Table View



DEPAUL UNIVERSITY 6



Use Table View

- View: UITableView
- Controller: UITableViewController
- Two types of Table View
 - Static:
 - Fix number of rows
 - Each row can have a unique design
 - Dynamic:
 - Flexible number of rows, data driven
 - Can accommodate a large number of rows
 - A fixed number of designs of rows – prototypes

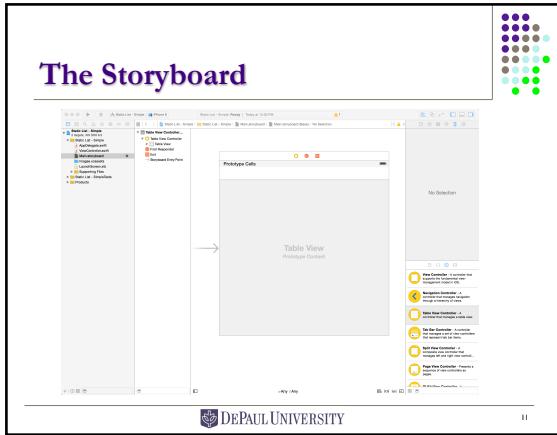
DEPAUL UNIVERSITY 8



New App: Static List – Simple

- New Project | Single View App
- Add a Table View Controller scene to the storyboard
 - Move the Start Arrow to the Table View Controller scene
 - Delete the original View Controller scene and the View Controller class (swift)

DEPAUL UNIVERSITY 10



Add Table View Controller

- Add a new custom Table View Controller
- New File | iOS Source | Cocoa Touch class
- New file options:
 - Class: **StaticTableViewController**
 - Superclass: **UITableViewcontroller**
 - Uncheck: Also create XIB file
 - Language: Swift

DEPAUL UNIVERSITY 12

Set Table View Controller Class

- In the storyboard, select the *Table View Controller*
- In the *Identity Inspector*, set the class to **StaticTableViewController**

DEPAUL UNIVERSITY 13

Design Static Table Cells

- In the storyboard, select the *Table View*
- In the *Attribute Inspector*, change the *Content* field from *Dynamic Prototypes* to *Static Cells*
- Edit the static cells
 - Add widgets and constraints in each table cell
 - Name field:
 - keyboard type: *Default*
 - Phone field:
 - keyboard type: *Numbers and Punctuations*

DEPAUL UNIVERSITY 14

Design Static Table Cells

DEPAUL UNIVERSITY 15

The Static Table View Controller

- Comment out methods
 - `numberOfSectionsInTableView`
 - `tableView:numberOfRowsInSection`
- Only needed for dynamic lists
- Connect outlets

```

@IBOutlet weak var nameField: UITextField!
@IBOutlet weak var phoneField: UITextField!
@IBOutlet weak var primaryContact: UISwitch!
@IBOutlet weak var maxStorage: UISlider!
@IBOutlet weak var storageLabel: UILabel!

```

DEPAUL UNIVERSITY 16

The Static Table View Controller

- Connect an action to the “Did End on Exit” event of both text field

```

@IBAction func doneEditing(sender: UITextField) {
    sender.resignFirstResponder()
}

```

- Connect an action to the slider

```

@IBAction func sliderChanged(sender: UISlider) {
    storageLabel.text =
        "Maximum Storage: \((Int(sender.value))G"
}

```

DEPAUL UNIVERSITY 17

The Submit Action

```

@IBAction func formSubmitted(sender: UIButton) {
    let title = "Static List"
    let name = nameField.text ?? ""
    let phone = phoneField.text ?? ""
    let message = "Name: \(name)\nPhone: \(phone)\n" +
        "Primary contact: \(primaryContact.isOn)\n" +
        "Maximum Storage: \(Int(maxStorage.value))G"
    let alertController = UIAlertController(
        title: title, message: message,
        preferredStyle: .Alert)
    let okayAction = UIAlertAction(title: "Okay",
        style: .Default, handler: nil)
    alertController.addAction(okayAction)
    presentViewController(alertController,
        animated: true, completion: nil)
}

```

DEPAUL UNIVERSITY 18

A Variation – Static Table View with Groups

DEPAUL UNIVERSITY

19

App: Static List – Groups

- In storyboard, select *Table View*
- In the *Attribute Inspector*,
 - Change *Style* from *Plain* to *Grouped* (optional)
 - Change *Sections* to 4
- In each *Table View Section*
 - Remove duplicate cells
- Enter the *Header* and *Footer* for each *Table View Section*

DEPAUL UNIVERSITY

20

Dynamic Table Views

Dynamic Table Views

- The *Table View* may contain a flexible number of rows
 - Data driven, usually backed by an Array or a database
 - Designed to efficiently handle very large tables
 - Table view *data source* and *delegate*
- The rows are populated from the data
 - The design of the rows are based on *prototype cells*
 - One or more prototype cells must be defined
 - Prototype cells are identified by the *reusable cell identifiers*

DEPAUL UNIVERSITY

22

A Simple Table View App – Wine List – Basic

- A simple *Table View* app
- A list of wines, titles only.
- The wine data is stored in an array
- Use the *Basic* table view cell as the prototype cell
- Scrollable and selectable
- No other action.

DEPAUL UNIVERSITY

23

New App: Wine List – Basic

- New Project | Single View App
- Add a *Table View Controller* scene to the storyboard
 - Move the *Start Arrow* to the *Table View Controller* scene
 - Delete the original *View Controller* scene and the *View Controller* class (swift)
- Add a new custom *Table View Controller* class
 - New File | iOS Source | Cocoa Touch class
 - Class: `WineListViewController`
 - Superclass: `UITableViewController`
- Set the class of the *Table View Controller* to the new class

DEPAUL UNIVERSITY

24

The Table View

- Select the Table View, in the Attribute Inspector
 - Set Content to Dynamic Prototypes
 - Set Prototype Cells to 1

DEPAUL UNIVERSITY 25

The Prototype Cell

- Select the Table View Cell, in the Attribute Inspector
 - Set Style to Basic
 - Set Identifier to basic

DEPAUL UNIVERSITY 26

The Table View Controller – The Data Source

```
let wines = [ "Barbera", "Cabernet Sauvignon", ...
  "Syrah", "Viognier", "Zinfandel" ]
```

```
class WineListViewController: UITableViewController {
    override func numberOfSectionsInTableView(
        tableView: UITableView) -> Int {
        return 1
    }
    override func tableView(tableView: UITableView,
        numberOfRowsInSection section: Int) -> Int {
        return wines.count
    }
}
```

DEPAUL UNIVERSITY 27

The Table View Controller – Configure Cells

- Uncomment and modify the method

```
override func tableView(tableView: UITableView,
    cellForRowAtIndexPath indexPath: NSIndexPath)
    -> UITableViewCell {
    let cell = tableView.dequeueReusableCellWithIdentifier(
        "basic", forIndexPath: indexPath)
    // Configure the cell...
    cell.textLabel?.text = wines[indexPath.row]
    return cell
}
```

DEPAUL UNIVERSITY 28

Customize the Prototype Cells

- Select the Title label in the prototype cell
- Customize the font and color

DEPAUL UNIVERSITY 29

The Table View Controller – Handle Row Selection

```
override func tableView(tableView: UITableView,
    didSelectRowAtIndexPath indexPath: NSIndexPath) {
    let title = "Wine List"
    let message = "You have selected \(wines[indexPath.row])"
    let alertController = UIAlertController(title: title,
        message: message, preferredStyle: .Alert)
    let okayAction = UIAlertAction(title: "Okay",
        style: .Default, handler: nil)
    alertController.addAction(okayAction)
    presentViewController(alertController, animated: true,
        completion: nil)
    self.tableView.deselectRowAtIndexPath(indexPath,
        animated: true)
}
```

DEPAUL UNIVERSITY 30

Prototype Cell with a Subtitle – Wine List – Subtitle

- A Table View app using Subtitle prototype cells with
 - A subtitle and
 - An image
- A list of wines with titles, types, and descriptions.
 - A custom Wine class
 - An array of Wine objects



DEPAUL UNIVERSITY

31

The Wine Class

```
class Wine {
    enum Type: String {
        case Red = "red"
        case White = "white"
        case Rosé = "rose"
        case Sparkling = "sparkling"
    }
    var name: String
    var type: Type
    var shortDescription: String
    var longDescription: String
    init(name: String, type: Type, shortDescription: String,
         longDescription: String) { ... }
}
```

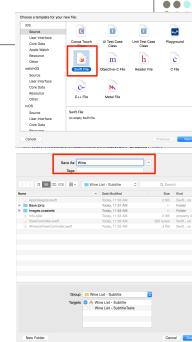
Each wine type will be represented by a prototype cell with an image. The raw values correspond to the reusable identifier of the cell.

DEPAUL UNIVERSITY

33

The Wine Class

- New File | iOS Source | Swift File
- Click Next
- Choose a file name: Wine
- Click Create

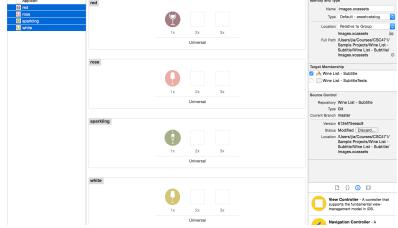


DEPAUL UNIVERSITY

32

The Prototype Cells

- Add images for each type of wines to the assets catalog



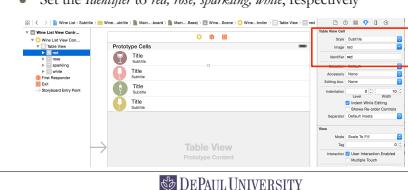
DEPAUL UNIVERSITY

35

The Prototype Cells

- In storyboard, select the Table View,
 - Set Prototype Cells to 4
- Select each Table View Cell, edit each cell
 - Set Style to Subtitle
 - Select an image
 - Set the Identifier to red, rose, sparkling, white, respectively

Correspond to the raw values of the wine type enum.



DEPAUL UNIVERSITY

36

The Table View Controller – The Data Source

- Same as before.

```
class WineListViewController: UITableViewController {
    override func numberOfSectionsInTableView(
        tableView: UITableView) -> Int {
        return 1
    }
    override func tableView(tableView: UITableView,
        numberOfRowsInSection section: Int) -> Int {
        return wines.count
    }
    ...
}
```

DEPAUL UNIVERSITY

37

The Table View Controller – Configure Cells

```
override func tableView(tableView: UITableView,
    cellForRowAtIndexPath indexPath: NSIndexPath)
-> UITableViewCell {
    let wine = wines[indexPath.row]
    let cell = tableView.dequeueReusableCellWithIdentifier(
        wine.type.rawValue, forIndexPath: indexPath)

    // Configure the cell...
    cell.textLabel?.text = wine.name
    cell.detailTextLabel?.text = wine.shortDescription
    return cell
}
```

DEPAUL UNIVERSITY

38

Run ...

The icon/cells are selected based on the wine type

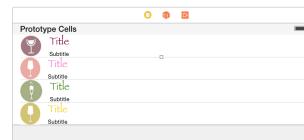


DEPAUL UNIVERSITY

39

Customize Prototype Cells

- Change the font and color of the *Title* label

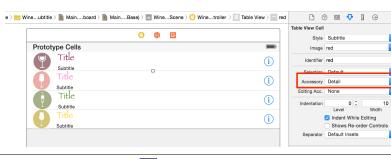


DEPAUL UNIVERSITY

40

Accessories of the Cell

- Each Table View Cell also has an optional accessory button.
- Accessory types: *Disclosure Indicator*, *Detail Disclosure*, *Checkmark*, *Detail*
- You may associate an action to accessory button tap

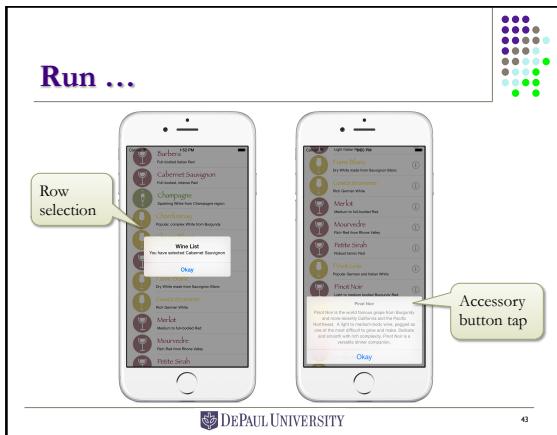


DEPAUL UNIVERSITY

41

The Table View Controller – Handle Accessory Button Tap

```
override func tableView(tableView: UITableView,
    accessoryButtonTappedForRowWithIndexPath indexPath:
    NSIndexPath) {
    let wine = wines[indexPath.row]
    let title = wine.name
    let message = wine.longDescription
    let alertController = UIAlertController(title: title,
        message: message, preferredStyle: .ActionSheet)
    let okayAction = UIAlertAction(title: "Okay",
        style: .Default, handler: nil)
    alertController.addAction(okayAction)
    presentViewController(alertController, animated: true,
        completion: nil)
    self.tableView.deselectRowAtIndexPath(indexPath,
        animated: true)
}
```



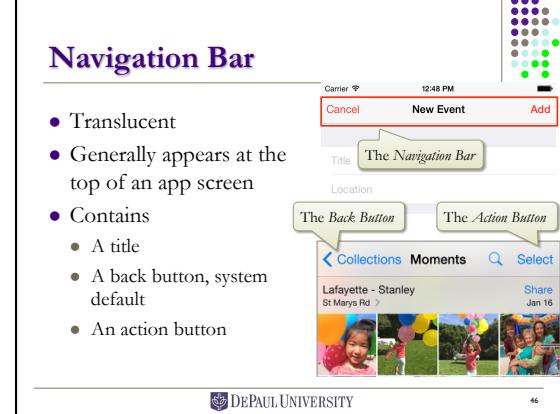
Navigation Controller

Navigation Controller

- Class: `UINavigationController`
- Displays a *navigation bar*, with a back button
- Manages a stack of view controllers
 - The first view controller is the *root view controller*
 - Go one level down
 - Push a view controller to the stack
 - Go one level back up
 - Pop the view controller at the top
- Commonly used with table views
 - To present a hierarchical structure

Navigation Bar

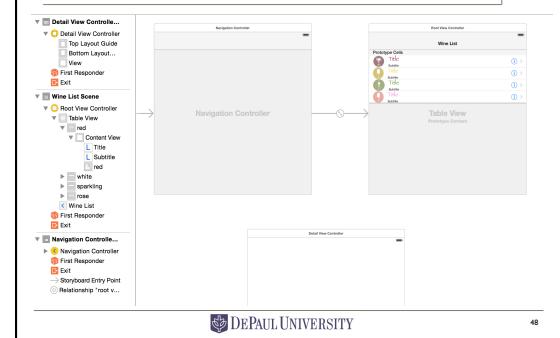
- Translucent
- Generally appears at the top of an app screen
- Contains
 - A title
 - A back button, system default
 - An action button



Wine List – Navigation + Selection

- New Project | Single View App
- Add a new *Navigation Controller*
 - It's a combo of a *Navigation View Controller* and a *Table View Controller*, the *root view controller*
 - Move the start arrow to the *Navigation View Controller*
- Add a new subclass of *Table View Controller*
 - `WineListViewController` and link it to the *Root View Controller*
 - Change the title from *Root View Controller* to *Wine List*
- Add the *Wine* class and the wine list data
- Rename the *View Controller* to `DetailViewController`

The Navigation and Table View



The Detail View Scene

- Two labels for the name and the detail descriptions of the selected wine



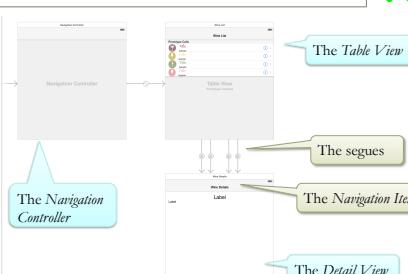
DEPAUL UNIVERSITY 49

Segues to Detail View – Selection

- Add segues from the *Wine List* scene to the *Wine Detail* scene
 - Ctrl-Drag from each *Prototype Cell* in the *Wine List* scene to the *Wine Detail* scene
 - Choice of *Selection Segue* or *Accessory Action*
 - Choose: *Selection Segue* | *Show*
- Perform segue when a cell is selected
- Add a *Navigation Item* to the *Wine Detail* scene, which will appear on the *Navigation Bar*

DEPAUL UNIVERSITY 50

The Segues From Table View to Detail View



DEPAUL UNIVERSITY 51

The Detail View Controller

```
class DetailViewController: UIViewController {
    @IBOutlet weak var titleLabel: UILabel!
    @IBOutlet weak var descriptionLabel: UILabel!
    var wine: Wine?

    override func viewDidAppear(animated: Bool) {
        if let w = wine {
            titleLabel.text = w.name
            descriptionLabel.text = w.longDescription
        }
    }
}
```

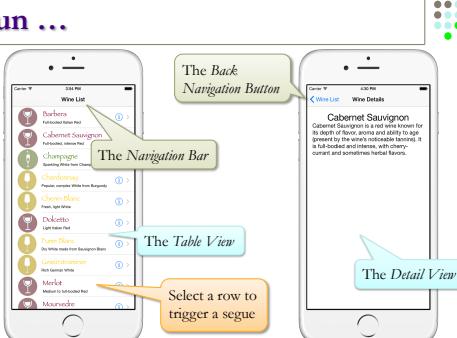
DEPAUL UNIVERSITY 52

The Wine List View Controller – Prepare Segue for Selection

```
override func prepareForSegue(segue: UIStoryboardSegue,
                             sender: AnyObject?) {
    if let detailViewController =
        segue.destinationViewController
        as? DetailViewController {
        if let indexPath =
            self.tableView.indexPathForSelectedRow() {
            detailViewController.wine =
                wines[indexPath.row]
        }
    }
}
```

DEPAUL UNIVERSITY 53

Run ...



DEPAUL UNIVERSITY 54

Wine List – Navigation + Accessory

- A variation of the last app
- Use the *accessory buttons* of the table cells to trigger segues
- Same set up as the last app, except
 - The segues from the *Wine List* scene to the *Wine Detail* scene
 - The *prepare for segue* method in the *Wine List View Controller*

DEPAUL UNIVERSITY

55

The Wine List View Controller – Prepare Segue for Accessory

```
override func prepareForSegue(segue: UIStoryboardSegue,
                             sender: AnyObject?) {
    if let detailViewController =
        segue.destinationViewController
            as? DetailViewController {
        if let cell = sender as? UITableViewCell {
            if let indexPath =
                self.tableView.indexPathForCell(cell) {
                detailViewController.wine =
                    wines[indexPath.row]
            }
        }
    }
}
```

When an accessory button is tapped, the row is not selected. Must use the *sender* to determine which row is tapped.

DEPAUL UNIVERSITY

57

Segues to Detail View – Accessory

- Add segues from the *Wine List* scene to the *Wine Detail* scene
- Ctrl-Drag from each *Prototype Cell* in the *Wine List* scene to the *Wine Detail* scene
- Choice of *Selection Segue* or *Accessory Action*
- Choose: *Accessory Action* | *Show*

Perform segue when an accessory button is tapped

DEPAUL UNIVERSITY

56

Prepare Segue for Accessory – A Pyramid of Ifs

```
override func prepareForSegue(segue: UIStoryboardSegue,
                             sender: AnyObject?) {
    if let detailViewController =
        segue.destinationViewController
            as? DetailViewController {
        if let cell = sender as? UITableViewCell {
            if let indexPath =
                self.tableView.indexPathForCell(cell) {
                detailViewController.wine =
                    wines[indexPath.row]
            }
        }
    }
}
```



DEPAUL UNIVERSITY

59

Prepare Segue for Accessory – An Alternative with Guards

```
override func prepareForSegue(segue: UIStoryboardSegue,
                             sender: AnyObject?) {
    guard let detailViewController =
        segue.destinationViewController
            as? DetailViewController
        else { return }
    guard let cell = sender as? UITableViewCell
        else { return }
    guard let indexPath =
        self.tableView.indexPathForCell(cell)
        else { return }
    detailViewController.wine =
        wines[indexPath.row]
}
```

Guard against the condition being false. The else block must exit the current scope.

DEPAUL UNIVERSITY

60

Sample Code

- Static List – Simple.zip
- Static List – Groups.zip
- Wine List – Basic.zip
- Wine List – Subtitle.zip
- Wine List – Navigation+Selection.zip
- Wine List – Navigation+Accessory.zip

Next ...

- Multi-threading
- Two-dimensional graphics drawing
- Touch events and gestures