CSC 471 / 371 Mobile Application Development for iOS

Prof. Xiaoping Jia
School of Computing, CDM
DePaul University
xjia@cdm.depaul.edu
@DePaulSWEng

Table & Navigation Views

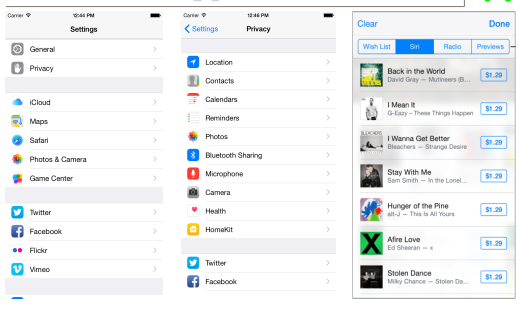
Outline

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

DEPAUL UNIVERSITY 3

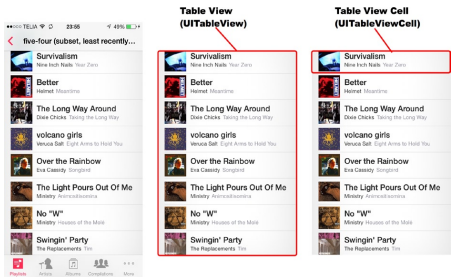
Table Views

Table View Apps



DEPAUL UNIVERSITY 5

Parts of a Table View

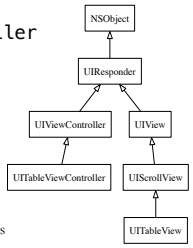


DEPAUL UNIVERSITY 6

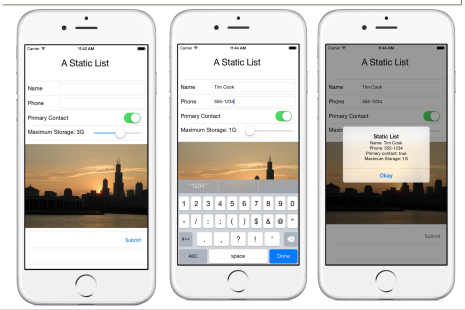
Static Table Views

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

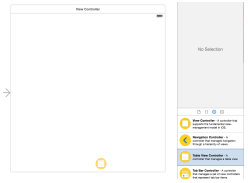


A Static Table View App

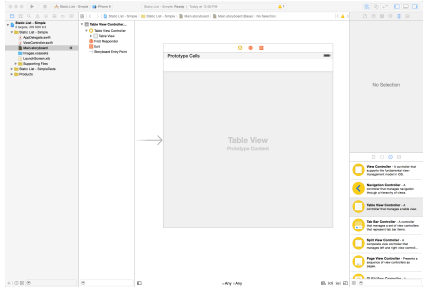


New App: Static List – Simple

- 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)

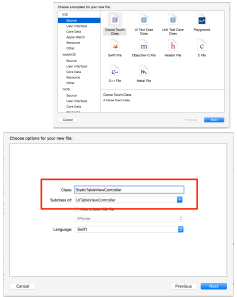


The Storyboard



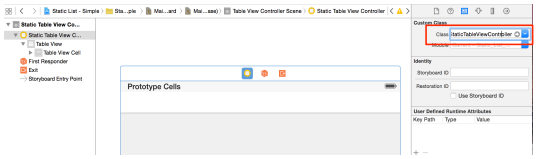
Add Table View Controller

- 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



Set Table View Controller Class

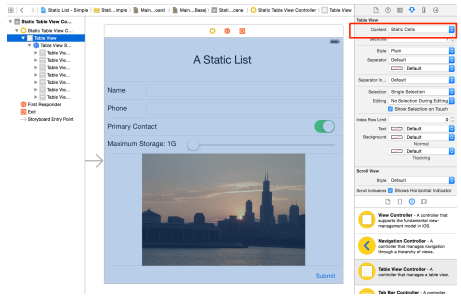
- In the storyboard, select the *Table View Controller*
  - In the *Identity Inspector*, set the class to `StaticTableViewController`



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*

Design Static Table Cells



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!
```


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()
}
```

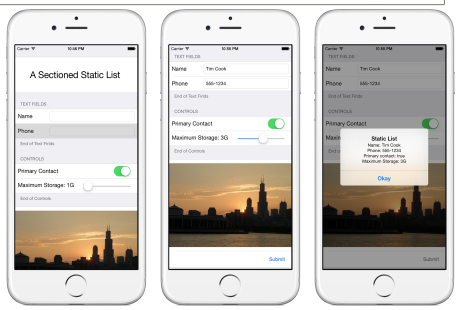
- Connect an action to the slider

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


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.on)\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)
}
```

A Variation – Static Table View with Groups



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*

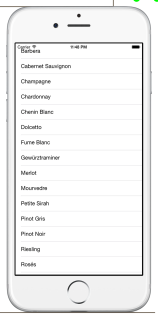
Dynamic Table Views

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 vary 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*

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.

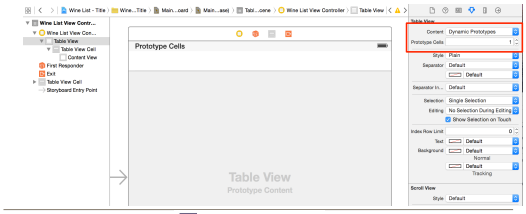


New App: Wine List – Basic

- 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

The Table View

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

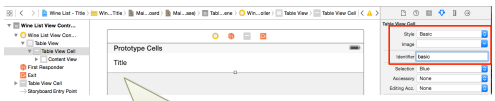


DePaul University 25

The Prototype Cell

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

The *reusable cell identifier*



The *Basic*-style cell with a *Label* for the title.

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

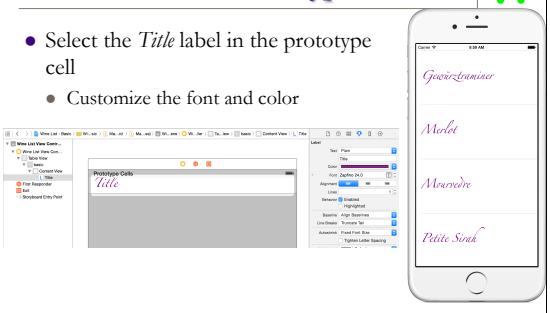
```
override func tableView(tableView: UITableView,
             cellForRowAtIndexPath indexPath: NSIndexPath)
             -> UITableViewCell {
    let cell = tableView.dequeueReusableCellWithIdentifier(
                 "basic", forIndexPath: indexPath)
                 as UITableViewCell

    // Configure the cell...
    cell.textLabel?.text = wines[indexPath.row]
    return cell
}
```

The *reusable cell identifier*

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 Section

```
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)
}
```

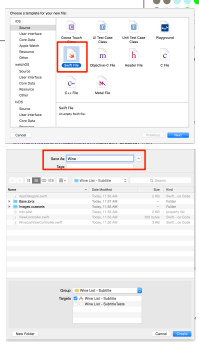
DePaul University 30

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

The Wine Class

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



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) { … }
}
```

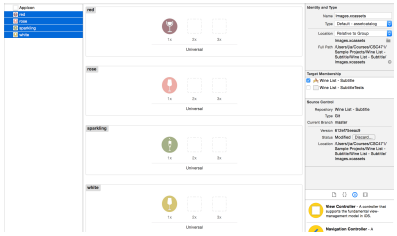
Each wine type will be represented by a prototype cell with an image. The raw values correspond to the *reusable identifier* of the cell.

The Wine List Data

```
let wines = [
    Wine(name: "Barbera",
        type: .Red,
        shortDescription: "Full-bodied Italian Red",
        longDescription: "Barbera is a red wine …"),
    Wine(name: "Cabernet Sauvignon",
        type: .Red,
        shortDescription: "Full-bodied, intense Red",
        longDescription: "Cabernet Sauvignon is a red …"),
    …
    Wine(name: "Zinfandel",
        type: .Red,
        shortDescription: "Medium to full-bodied Red",
        longDescription: "Zinfandel is a medium to …"),
]
```


The Prototype Cells

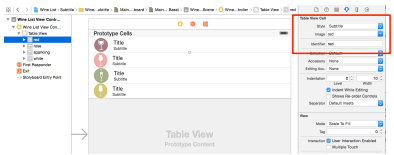
- Add images for each type of wines to the assets catalog



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.



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

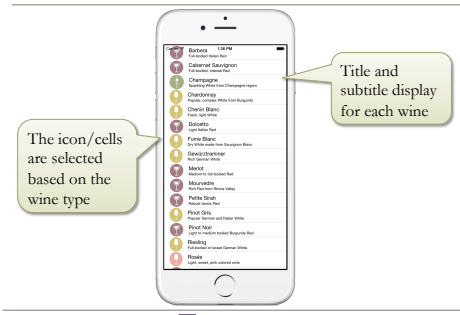
The Table View Controller – Configure Cells

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

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

The *reusable cell identifier*

DePaul University 38

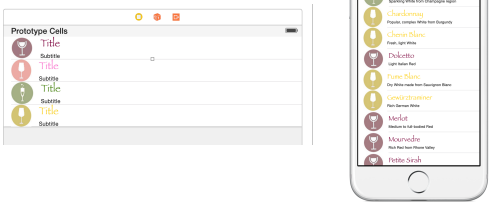
Run …



DePaul University 39

Customize Prototype Cells

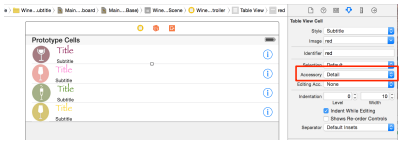
- Change the font and color of the *Title* label



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

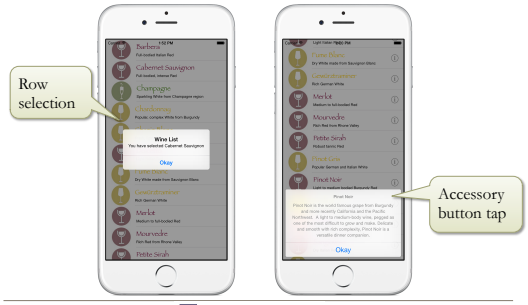


DePaul University 41

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)
}
```

Run …



DePaul University 43

Navigation Controller

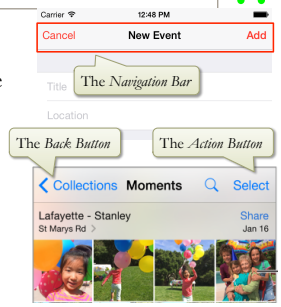
Navigation Controller

- 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

DePaul University 45

Navigation Bar

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



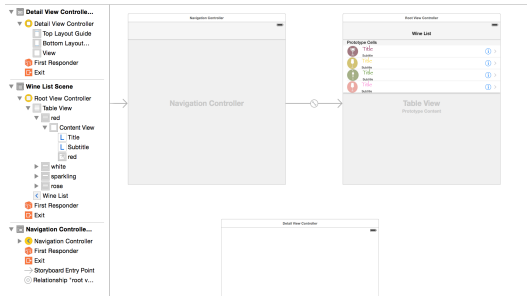
DePaul University 46

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`

DePaul University 47

The Navigation and Table View



DePaul University 48

The Detail View Scene

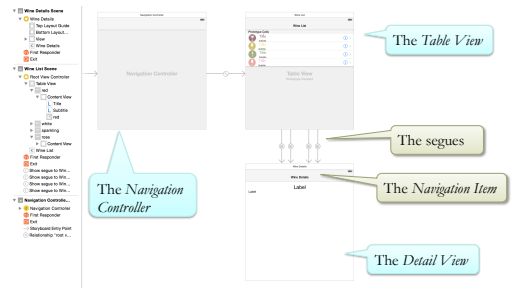
- Two labels for the name and the detail descriptions of the selected wine

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*
- Add a *Navigation Item* to the *Wine Detail* scene, which will appear on the *Navigation Bar*

Perform segue when a cell is selected

The Segues From Table View to Detail View



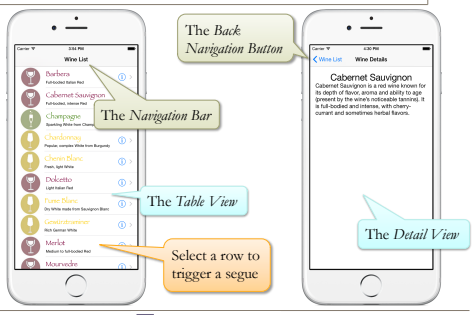
The Detail View Controller

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

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


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]
        }
    }
}
```


Run …



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*

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*

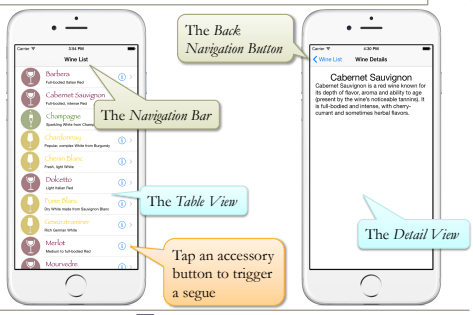
Perform segue when an accessory button is tapped

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.

Run …



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