CSC 471 / 371 Mobile Application Development for iOS

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

1

Adaptive Design

2

Outline

- Size classes
- Split view controller

DePaul University 3

Size Classes

4

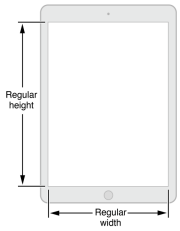
Size Classes

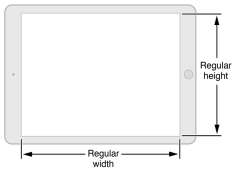
- Introduced in iOS 8
- A *size class* identifies a relative amount of display space for the height and for the width.
- Two size classes: *regular* and *compact*.
  - The *regular* size class – expansive space
  - The *compact* size class – constrained space.
- Each display environment is characterized by a *horizontal* size class and a *vertical* size class.
- iOS automatically makes various layout changes when the size classes of a display environment change.

DePaul University 5

The Size Classes of iPad

- Portrait: Regular x Regular
- Landscape: Regular x Regular

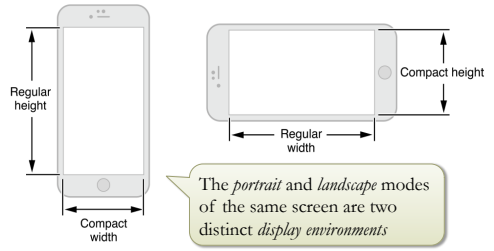




DePaul University 6

The Size Classes of an iPhone 6+

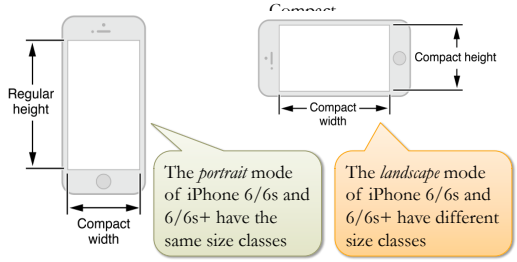
- Portrait: Compact x Regular
- Landscape: Regular x Compact



The *portrait* and *landscape* modes of the same screen are two distinct *display environments*

The Size Classes of iPhone 6

- Portrait: Compact x Regular
- Landscape: Compact x Compact



The *portrait* mode of iPhone 6/6s and 6/6s+ have the same size classes

The *landscape* mode of iPhone 6/6s and 6/6s+ have different size classes

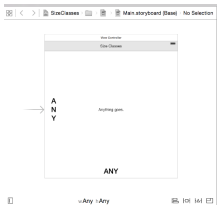
Adaptive Design with Size Classes

- You can design adaptive UI based on the size classes
- Xcode allows selection of size classes for the width and height from
  - Compact, Any, Regular
- Widgets and constraints can be *installed* or *uninstalled* for specific size classes
  - When a widget or constraint is *uninstalled* the associated object *does not exist* in the view hierarchy

The Default Size Classe

- The default size class of the Interface Builder is

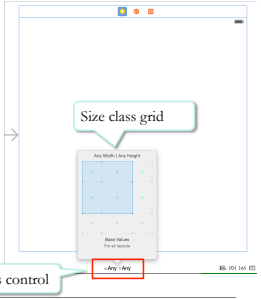
  Any Width | Any Height
  - All widgets are installed for screens of all size classes
  - The same set of constraints are installed, i.e., in effect, for screens of all size classes



Select Size Classes

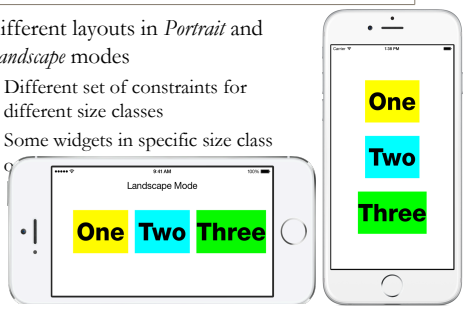
- Use the size class control
- Move the mouse to select from the 3 × 3 grid
- The square shape of the scene represents the default size class

  w Any | h Any



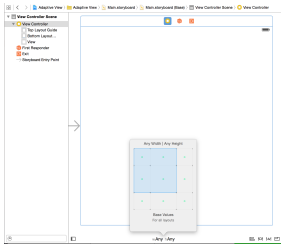
The Adaptive View Demo

- Different layouts in *Portrait* and *Landscape* modes
  - Different set of constraints for different size classes
  - Some widgets in specific size class



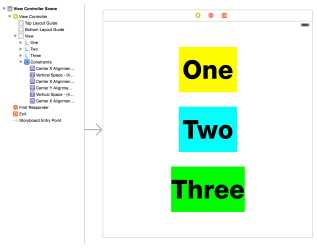
The Adaptive View

- Start with a single view app
- Use the Default size class
- Design the UI for the default size class first
  - The Portrait mode
  - Customize for the landscape size class later



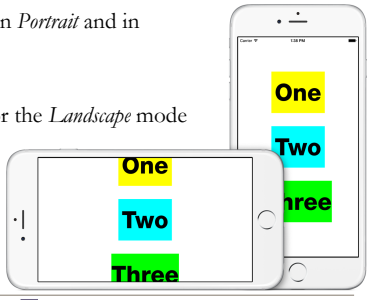
The Portrait Model UI Design

- The constraints:
  - Center X Alignment:
    View – One
    View – Two
    View – Three
  - Center Y Alignment:
    View – Two
  - Vertical Space:
    One – Two
    Two – Three



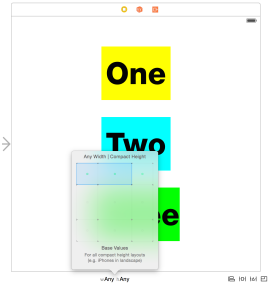
Run …

- Same layout in *Portrait* and in *Landscape*
- Not adaptive
- Not suited for the *Landscape* mode



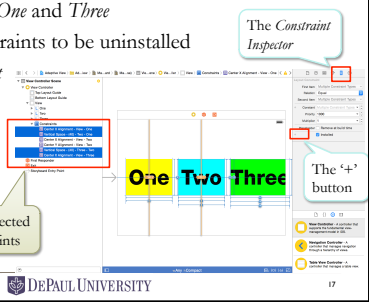
Change the Size Class

- Use the Size Class control to set the size class to
  w Any | h Compact
- The size class that fits the *Landscape* mode of all iPhones, including iPhone 6+



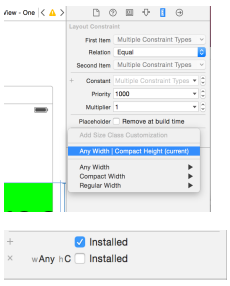
Uninstall Constraints

- Keep only the constraints for *Two*. Uninstall the constraints for *One* and *Three*
- Select the constraints to be uninstalled
- In the *Constraint Inspector*
  - Click the '+'



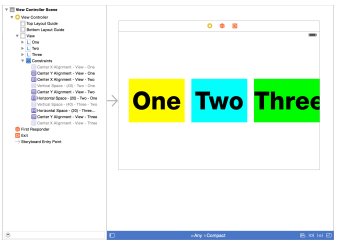
Uninstall Constraints

- Add a size class
  Any Width | Compact Height
- Uncheck the *installed* box for
  W Any h C
- Now, all the selected constraints are uninstalled for the size class
  Any Width | Compact Height



Install New Constraints

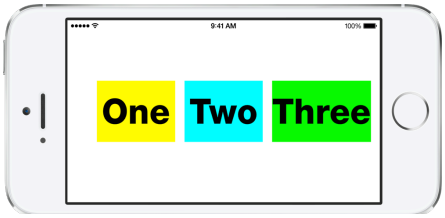
- Add constraints
  - Center Y Alignment
    - View – One
    - View – Three
  - Horizontal Space
    - One – Two
    - Two – Three
- These constraints will only be installed for Any | Compact



DePaul University 19

Run …

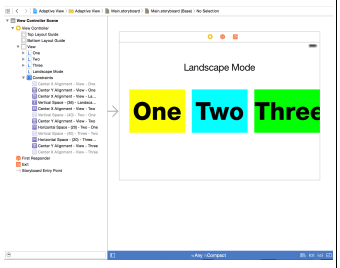
- The Landscape model looks better



DePaul University 20

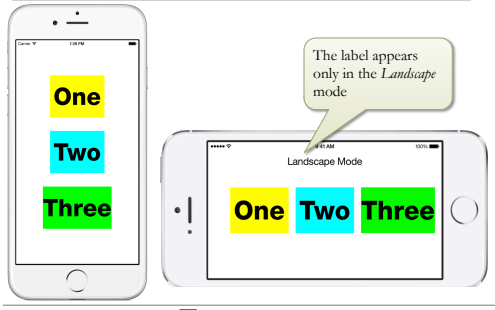
Add a Widget

- Add a Label and constraints
  - Center X Alignment
  - Vertical Space to top



DePaul University 21

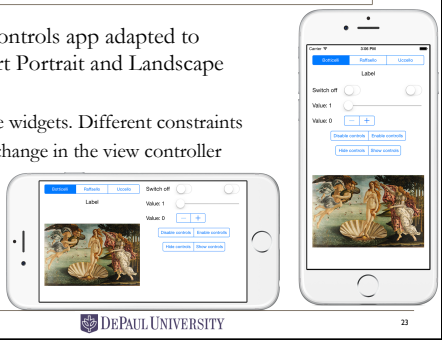
Run …



DePaul University 22

A More Complex Adaptive Design

- The Controls app adapted to support Portrait and Landscape layout
  - Same widgets. Different constraints
  - No change in the view controller



DePaul University 23

A More Complex Adaptive Design

- Divide the view into several regions, i.e., containers
- Each container maintain its own set of constraints.
- Only the top level constraints need to be adapted

DePaul University 24

Split View Controller

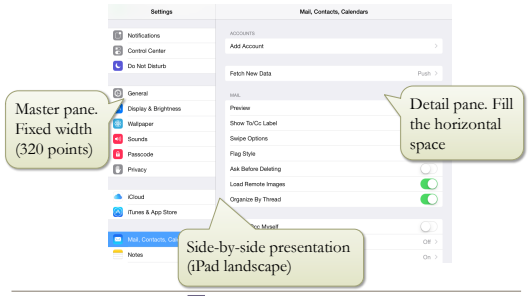
25

Split View Controller

- Designed for iPad. Works on all iOS 8 devices
  - iPad only for iOS 7 or earlier
- Manages two child view controllers – master/detail.
  - Several presentation styles based on screen dimensions
  - Side-by-side, one at a time, partial overlay

DePaul University 26

Split View Controller Example



DePaul University 27

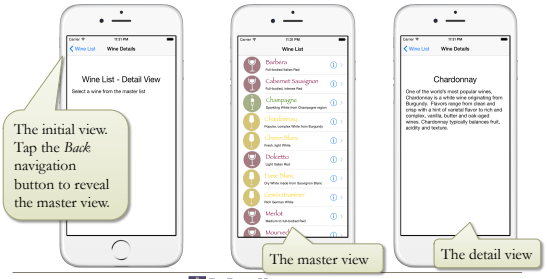
Wine List – Split View

- Another variation of the *Wine List* app using a split view controller
  - When running on iPhone: nearly identical with the version using navigation view
  - when running on iPad: automatically adapt the presentation style

DePaul University 28

Wine List – Split View

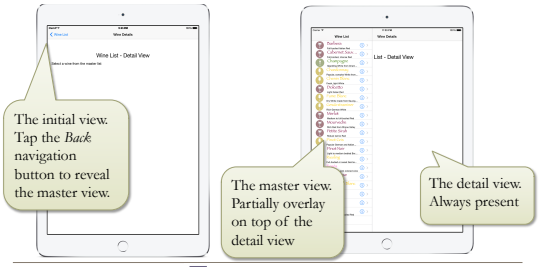
- Running on iPhone – one view controller at a time



DePaul University 29

Wine List – Split View

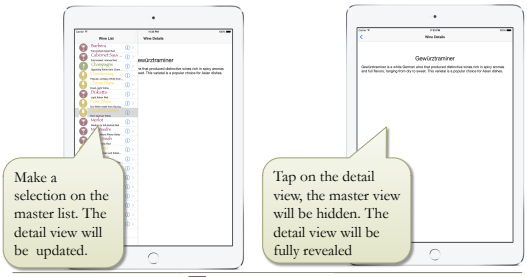
- Running on iPad Air, *Portrait* mode.



DePaul University 30

Wine List – Split View

- Running on iPad Air, *Portrait* mode.



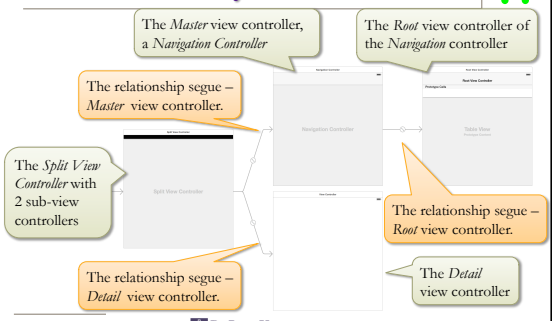
Wine List – Split View

- Running on iPad Air, *Landscape* mode.

Create a Split View App

- Start with a new single view app
- In storyboard, add a *Split View Controller* (a 4-controller bundle)
  - and move the start arrow to the *Split View Controller*
- The *Split View Controller* has two sub-view controllers
  - The *Master* view controller, which is a *Navigation Controller* with its root view controller
  - The *Detail* view controller

The Split View Controller – The Initial Storyboard

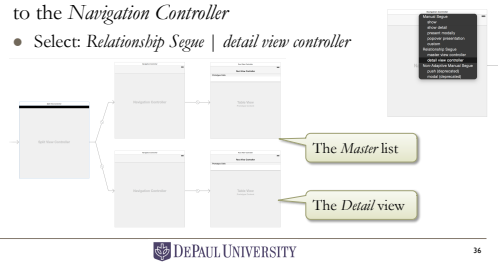


Replace the Detail View Controller

- The initial detail view controller is a simple view controller.
  - No navigation bar when running in iPad *Portrait* mode.
  - Use a *right-swipe* gesture to reveal the master list. No visual clue.
- Replace it with a *Navigation Controller*
  - Has a navigation bar, and *Back* button to reveal the master list.

Replace the Detail View Controller

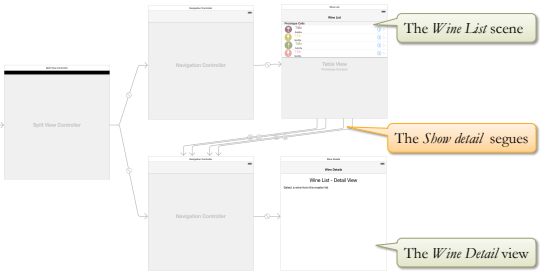
- In storyboard, add a *Navigation Controller*
- Add a segue (Ctrl-Drag) from the *Split View Controller* to the *Navigation Controller*
  - Select: *Relationship Segue | detail view controller*



Edit the Wine List Scenes

- Nearly the same design and logic from the *Wine List – Navigation+Selection* app
  - The *Wine List* and *Wine Detail* scenes and view controllers
- **Different** segues from the cells in the *Wine List* scene to the *Navigation Controller*
  - Choose: *Selection Segue* | *Show detail* (instead of *Show*)

The Completed Storyboard



The Wine Detail View Controller

- Only change needed in the *Detail View Controller*
  - To set up the *Back* navigation button

```
override func viewWillAppear(animated: Bool) {
    navigationItem.leftBarButtonItem =
        splitViewController?.displayModeButtonItem()
    navigationItem.leftItemsSupplementBackButton = true
    if let w = wine {
        titleLabel.text = w.name
        descriptionLabel.text = w.longDescription
    }
}
```


The Wine List View Controller

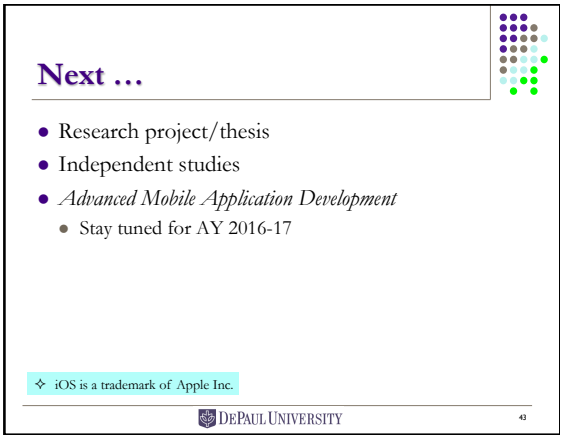
- Only change needed in the *Wine List View Controller*
  - To display the alert as a *Popover*

```
override func tableView(tableView: UITableView,
        accessoryButtonTappedForRowWithIndexPath
        indexPath: NSIndexPath) { …
    let alertController = UIAlertController(…)
    let popover = alertController.popoverPresentationController
    if let tableView = self.view as? UITableView {
        popover?.sourceView =
            tableView.cellForRowAtIndexPath(indexPath)
    }
    presentViewController(alertController, animated: true,
        completion: nil)
    self.tableView.deselectRowAtIndexPath(…)
}
```

Action Sheet as a Popover

Sample Code

- Adaptive View.zip
- Controls – Rotate.zip
- Wine List – Split View.zip


Next …
Research project/thesis
Independent studies
Advanced Mobile Application Development
Stay tuned for AY 2016-17
iOS is a trademark of Apple Inc.
DePaul University
43