

CSC 471 / 371
Mobile Application
Development for iOS



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

Tabbed Views & Picker Views

Outline

- Tabbed views
- Picker views



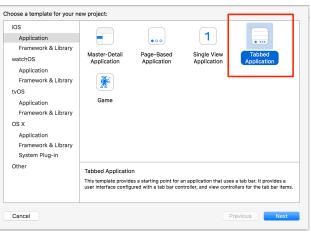
Tabbed Views

Tabbed Apps



Create a Tabbed App

- Use the “*Tabbed Application*” template
- Create an app with two tabs



Storyboard of the Tabbed App

- Ready to run
- Each tab can be customized
- Add tabs by adding
 - view controllers and
 - relationship* segues

DEPAUL UNIVERSITY 7

Add a New Tab

- Add a new scene
 - Drag and drop a new *View Controller* in the storyboard
- Add a new Swift class
 - Class: `ThirdViewController`
 - Subclass of: `UIViewController`
 - Uncheck: *Also create XIB file*
- In the storyboard
 - Select the new *View Controller*
 - Use the *Identity Inspector* to change the Class to: `ThirdViewController`

DEPAUL UNIVERSITY 8

Add a Relationship Segue

- Add a segue from the *Tab Bar Controller* to the new *View Controller*.
- Select: *Relationship Segue* | *View Controllers*
- Ctrl-drag from the *Tab Bar Controller* to the *Third* view controller.
- Select *Relationship Segue* | *view controllers*

DEPAUL UNIVERSITY 9

The Storyboard with Three Tabs

- The *Third* tab is added and linked
- The tabs can be customized

DEPAUL UNIVERSITY 10

Customize the Title and Icon of the New Tab

- Select the *tab item* at the bottom of the new scene
 - Use the *Attribute Inspector*
- Option 1:
 - Use *System Items*, e.g., *featured*
- Option 2:
 - Use a *Custom* item
 - Add an icon in *images.xcassets*
 - Choose a *title* for the tab item

DEPAUL UNIVERSITY 11

Run ... the Tabbed App

- The *First* tab
- The *Featured* tab

DEPAUL UNIVERSITY 12

“Order My Sub” App Using Pickers

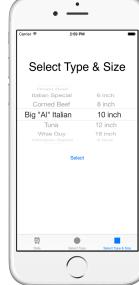


This slide shows the title “Order My Sub” App Using Pickers. Below the title is a placeholder area for the app's interface.

“Order My Sub” App



- A three-tab app using pickers
 - A date picker to pick date and time
 - A simple one-component picker to choose the type of sub to order
 - A two-component picker to choose the type and the size of the sub to order
- Using popup alerts and action sheets
 - Using closures for actions
- Sharing data between tabs



DEPAUL UNIVERSITY 14

Using a Date Picker



- Start with the *Tabbed App* template
- Add a third tab with a *Date Picker*
- Add new *View Controller* class: *DateViewController*
- Link to the third *View Controller* in the storyboard
- Customize the third tab
 - Change the tab image and title
 - Add a *Date Picker* and other controls in the view



DEPAUL UNIVERSITY 15

The “Pick a Date & Time” Scene



- Connect outlets to
 - The *Segmented Control*
 - The *Date Picker*
- Connect actions to
 - The *Segmented Control*
 - The “Select” Button



DEPAUL UNIVERSITY 16

The Date View Controller – the Outlets and Actions



```
class DateViewController: UIViewController {
    @IBOutlet weak var options: UISegmentedControl!
    @IBOutlet weak var datePicker: UIDatePicker!
    @IBAction func optionSelected(sender: UISegmentedControl) {
    }
    @IBAction func datePicked(sender: UIButton) { ... }

    override func viewDidLoad() {
        super.viewDidLoad()
        options.selectedSegmentIndex = 0
    }
}
```

Set the initial state of the *Segmented Control*

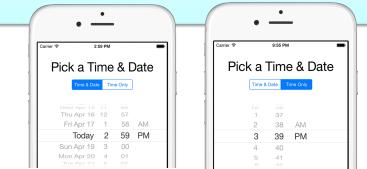


DEPAUL UNIVERSITY 17

Set the Date Picker Mode



```
@IBAction func optionSelected(sender: UISegmentedControl) {
    if sender.selectedSegmentIndex == 0 {
        datePicker.datePickerMode = .DateAndTime
    } else if sender.selectedSegmentIndex == 1 {
        datePicker.datePickerMode = .Time
    }
}
```



DEPAUL UNIVERSITY 18

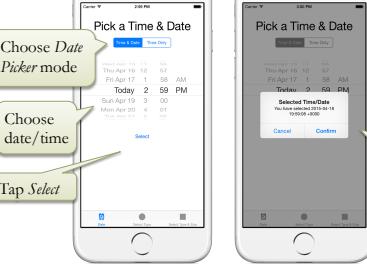
Action for Picking Date

```
@IBAction func datePicked(sender: UIButton) {
    let title = "Selected Time/Date"
    let message = "You have selected \\"(datePicker.date)""
    let alertController = UIAlertController(title: title,
                                           message: message, preferredStyle: .Alert)
    let cancelAction = UIAlertAction(title: "Cancel",
                                    style: .Cancel, handler: nil)
    let okayAction = UIAlertAction(title: "Confirm",
                                  style: .Default, handler: nil)
    alertController.addAction(cancelAction)
    alertController.addAction(okayAction)
    presentViewController(alertController, animated: true,
                          completion: nil)
}
```

DEPAUL UNIVERSITY

19

Select a Time and a Date



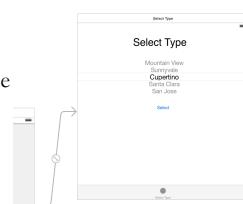
An alert popup with a *Cancel* and a *Confirm* button.

DEPAUL UNIVERSITY

20

The “Select Type” Scene – Single Component Picker

- Use *View Controller* class: `FirstViewController`
- Add a *Picker View* and a *Button* in the view
- Connect an outlet to the *Picker*
- Connect an actions to the “Select” *Button*



DEPAUL UNIVERSITY

21

The First View Controller

- The outlet and the action

```
class FirstViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
    @IBOutlet weak var picker: UIPickerView!
    @IBAction func selected(sender: UIButton) { ... }
    ...
}
```

DEPAUL UNIVERSITY

22

Populate Data in the Picker

- Use the *Delegate* pattern
 - The *Picker View* will delegate the task of populating the data to this *View Controller*.
- This *View Controller* must conform to two protocols
 - `UIPickerViewDelegate`
 - `UIPickerViewDataSource`

```
class FirstViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
    ...
}
```

DEPAUL UNIVERSITY

23

Connection to the Delegate

- The connection between the *Picker View* and its *data source* and *delegate* must be explicitly set
- In storyboard, select the *Picker View*
- Select the *Connection Inspector*
- Connect the outlets named *dataSource* and *delegate* to the *View Controller*



DEPAUL UNIVERSITY

24

The Single Component Picker

- Populate the *Picker View* data in the *View Controller*

```
let subs = [
    "Blockbuster",
    "Roast Beef",
    ...,
    "American"
]

class FirstViewController: UIViewController,
    UIPickerViewDataSource, UIPickerViewDelegate {
    ...
}
```

- An array of strings used to populate the *Picker View*.
- In the *global* scope, i.e., the *internal* scope. Visible inside the project. Shared among view controllers.

DEPAUL UNIVERSITY

25

Implement the Data Source

- The methods of the `UIPickerViewDataSource` protocol

```
func numberOfComponentsInPickerView(
    pickerView: UIPickerView) -> Int {
    return 1
}

func pickerView(pickerView: UIPickerView,
    numberOfRowsInComponent component: Int)
-> Int {
    return subs.count
}
```

DEPAUL UNIVERSITY

26

Implement the Delegate

- The methods for the `UIPickerViewDelegate` protocol

```
func pickerView(pickerView: UIPickerView,
    titleForRow row: Int,
    forComponent component: Int)
-> String? {
    return subs[row]
}
```

DEPAUL UNIVERSITY

27

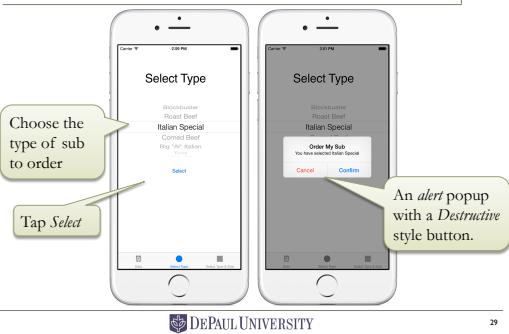
The Select Action

```
@IBAction func selected(sender: UIButton) {
    let title = "Order My Sub"
    let message = "You have selected
\(`subs[picker.selectedRowInComponent(0)]`)"
    let alertController = UIAlertController(title: title,
        message: message, preferredStyle: .Alert)
    let cancelAction = UIAlertAction(title: "Cancel",
        style: .Destructive, handler: nil)
    let okayAction = UIAlertAction(title: "Confirm",
        style: .Default, handler: nil)
    alertController.addAction(cancelAction)
    alertController.addAction(okayAction)
    presentViewController(alertController, animated: true,
        completion: nil)
}
```

DEPAUL UNIVERSITY

28

The Test Run ...

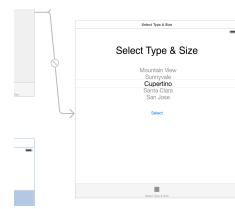


DEPAUL UNIVERSITY

29

The “Select Type & Size” Scene – Double Component Picker

- Use *View Controller* class: `SecondViewController`
- Add a *Picker View* and a *Button* in the view
- Connect an outlet to the *Picker*
- Connect an actions to the “Select” *Button*
- Connect the *dataSource* and the *delegate* of the *Picker View*



DEPAUL UNIVERSITY

30

The Second View Controller

- The outlet and the action
- This View Controller also conforms to two protocols
 - UIPickerViewDelegate
 - UIPickerViewDataSource

```
class SecondViewController: UIViewController,
    UIPickerViewDataSource, UIPickerViewDelegate {
    @IBOutlet weak var picker: UIPickerView!
    @IBAction func selected(sender: UIButton) { ... }
}
```

DEPAUL UNIVERSITY

31

The Double Component Picker

- Populate the Picker View data in the View Controller

```
let sizes = [
    "6 inch",
    "8 inch",
    "10 inch",
    "12 inch",
    "16 inch",
    "3 foot"
]
class SecondViewController: UIViewController,
    UIPickerViewDataSource, UIPickerViewDelegate {
    ...
}
```

An array of strings used to populate the second component of the Picker View.

DEPAUL UNIVERSITY

32

Implement the Data Source

- The UIPickerViewDataSource protocol

```
func numberOfComponentsInPickerView(
    pickerView: UIPickerView) -> Int {
    return 2
}
func pickerView(pickerView: UIPickerView,
    numberOfRowsInComponent component: Int)
    -> Int {
    if component == 0 {
        return subs.count
    } else {
        return sizes.count
    }
}
```

DEPAUL UNIVERSITY

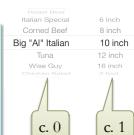
33

Implement the Delegate

- The UIPickerViewDelegate protocol

```
func pickerView(pickerView: UIPickerView,
    titleForRow row: Int,
    forComponent component: Int)
    -> String? {
    if component == 0 {
        return subs[row]
    } else {
        return sizes[row]
    }
}
```

Select Type & Size



DEPAUL UNIVERSITY

34

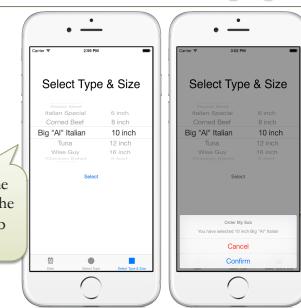
The Select Action

```
@IBAction func selected(sender: UIButton) {
    let title = "Order My Sub"
    let message = "You have selected
    \"(sizes[picker.selectedRowInComponent(1)])
    \$(subs[picker.selectedRowInComponent(0)])\""
    let alertController = UIAlertController(title: title,
        message: message, preferredStyle: .ActionSheet)
    let cancelAction = UIAlertAction(title: "Cancel",
        style: .Destructive) { action in ... } // Action for Cancel
    let confirmAction = UIAlertAction(title: "Confirm",
        style: .Default) { action in ... } // Action for Confirm
    alertController.addAction(cancelAction)
    alertController.addAction(confirmAction)
    presentViewController(alertController, animated: true,
        completion: nil)
}
```

DEPAUL UNIVERSITY

35

The Action Sheet Popup



An action sheet popup with a Destructive style button and a Default style button.

DEPAUL UNIVERSITY

36

The Select – Cancel Action

```
@IBAction func selected(sender: UIButton) {
    ...
    let cancelAction = UIAlertAction(title: "Cancel",
        style: .Destructive) { action in
        let alertController = UIAlertController(
            title: "No Problem", message: "Select again.",
            preferredStyle: .Alert)
        alertController.addAction(okayAction)
        alertController.addAction(cancelAction)
        self.presentViewController(alertController,
            animated: true, completion: nil)
    }
    ...
}
```

DEPAUL UNIVERSITY

37

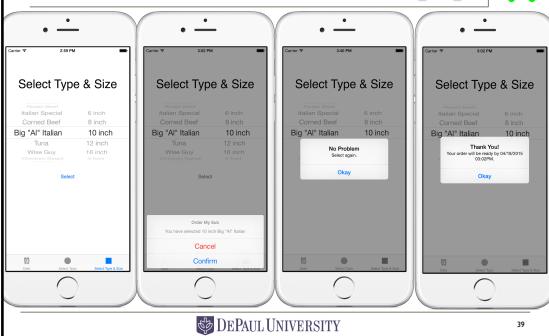
The Select – Confirm Action

```
let confirmAction = UIAlertAction(title: "Confirm",
    style: .Default) { action in
    let okayController = UIAlertController(
        title: "Thank You!", message: "Your order ...",
        preferredStyle: .Alert)
    let okayAction = UIAlertAction(title: "Okay",
        style: .Default, handler: nil)
    okayController.addAction(okayAction)
    self.presentViewController(okayController,
        animated: true, completion: nil)
}
alertController.addAction(cancelAction)
alertController.addAction(confirmAction)
...
}
```

DEPAUL UNIVERSITY

38

The Cancel and Conform Popup



DEPAUL UNIVERSITY

39

One More Refinement

- Use the date/time selected at “Pick a Time & Data” tab as the pick-up time.
- Format the confirm message of the “Pick Type & Size” tab using the pick-up time.
- To pass data between tabs, we use a global variable

```
var date = NSDate() // A global variable
class DateViewController: UIViewController {
    ...
}
```

DEPAUL UNIVERSITY

40

Pick a Time and Date – Handle Confirm Action

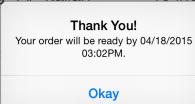
```
@IBAction func datePicked(sender: UIDatePicker) {
    let title = "Selected Time/Date"
    let message = "You have selected \(datePicker.date)"
    let alertController = UIAlertController(title: title,
        message: message, preferredStyle: .Alert)
    let cancelAction = UIAlertAction(title: "Cancel",
        style: .Cancel, handler: nil)
    let okayAction = UIAlertAction(title: "Confirm",
        style: .Default) {
        action in date = self.datePicker.date
    }
    alertController.addAction(cancelAction)
    alertController.addAction(okayAction)
    presentViewController(alertController, animated: true,
        completion: nil)
}
```

DEPAUL UNIVERSITY

41

Pick Type & Size – The Select – Confirm Action

```
let confirmAction = UIAlertAction(title: "Confirm",
    style: .Default) { action in
    let dateFormatter = NSDateFormatter()
    dateFormatter.dateFormat = "MM/dd/yyyy hh:mm"
    let dateString = dateFormatter.stringFromDate(date)
    let okayController = UIAlertController(
        title: "Thank You!",
        message: "Your order will be ready by \(dateString).",
        preferredStyle: .Alert)
    ...
}
```



Okay

DEPAUL UNIVERSITY

42

Sample Code

- Tabbed App.zip
- Order My Sub.zip



DEPAUL UNIVERSITY

43

Next ...

- Static table views
- Dynamic table views
- Navigation views



❖ iOS is a trademark of Apple Inc.

DEPAUL UNIVERSITY

44