Notifications

Mobile Application Development in iOS

School of EECS

Washington State University

Instructor: Larry Holder
Outline

• Alerts
• Internal notifications
• Local notifications
• Remote (push) notifications
Alerts

• **UIAlertController**
  
  – `init` *(title, message, preferredStyle)*
    
    • `preferredStyles`: `.alert` *(popover)*, `.actionSheet` *(bottom)*
  
  – `addAction` *(UI.AlertAction)*
    
    • `init` *(title, style, handler)*
      
      – Style: `.default`, `.cancel`, `.destructive`
  
  – `preferredAction` *(bold on `.alert` style)*
  
  – `addTextField` *(handler)*
let alert = UIAlertController(title: "Next Action", message: "Choose your next action.", preferredStyle: .alert)

let goAction = UIAlertAction(title: "Go", style: .default, handler: { (action) in
    // execute some code when this option is selected
    print("Go!")
})

let stopAction = UIAlertAction(title: "Stop", style: .destructive,
    handler: { (action) in
    print("Stop!")
})

let cancelAction = UIAlertAction(title: "Cancel", style: .cancel,
    handler: { (action) in
    print("Cancel.")
})
Alerts: Demo

... 

```swift
alert.addAction(goAction)
alert.preferredAction = goAction // only affects .alert style
alert.addAction(stopAction)
alert.addAction(cancelAction)

present(alert, animated: true, completion: nil)
```

preferredStyle: `.alert`

preferredStyle: `.actionSheet`

Next Action
Choose your next action.

- Go
- Stop
- Cancel

Next Action
Choose your next action.

- Go
- Stop
- Cancel
Alerts with Text Fields

```swift
var loginAlert: UIAlertController!

func initializeLoginAlert() {
    loginAlert = UIAlertController(title: "Login",
                                  message: "Enter your username and password.", preferredStyle: .alert)
    loginAlert.addTextField(configurationHandler: usernameTextFieldHandler)
    loginAlert.addTextField(configurationHandler: passwordTextFieldHandler)
    loginAlert.addAction(UIAlertAction(title: "Login", style: .default, handler: { (action) in
        let username = self.loginAlert.textFields?[0].text
        let password = self.loginAlert.textFields?[1].text
        print("username = \(username!), password = \(password!)")}))
}

func usernameTextFieldHandler (_ textField: UITextField) {
    textField.placeholder = "Username"
}

func passwordTextFieldHandler (_ textField: UITextField) {
    textField.placeholder = "Password"
    textField.isSecureTextEntry = true
}

@IBAction func alertWithText(_ sender: UIButton) {
    self.present(loginAlert, animated: true, completion: nil)
}
```

Mobile Application Development in iOS 6
Internal Notifications

• NotificationCenter
  – addObserver
  – removeObserver
  – post(Notification)

• Notification names
  – developer.apple.com/documentation/foundation/notification.name
Internal Notifications

• Example: Detect change to UserDefaults

```swift
// In viewDidLoad
let center = NotificationCenter.default
center.addObserver(
    self,
    selector: #selector(ViewController.settingsChanged),
    name: UserDefaults.didChangeNotification,
    object: nil)

// In ViewController
@objc func settingsChanged (_ notification: NSNotification) {
    print("settingsChanged")
}
```
Internal Notifications

• Example: Detect when app becomes active

```swift
// In viewDidLoad
let center = NotificationCenter.default
center.addObserver(self,
    selector: #selector(ViewController.appActive),
    name: .UIApplicationDidBecomeActive, object: nil)

// In ViewController
@objc func appActive (_ notification: NSNotification) {
    print("app active")
}
```
Internal Notifications

• Define you own

```swift
// In viewDidLoad
let center = NotificationCenter.default
center.addObserver(self,
    selector: #selector(ViewController.coolDetected),
    name: NSNotification.Name("coolNotification"), object: nil)

// In ViewController
@objc func coolDetected (_ notification: NSNotification) {
    print("cool")
}

// Anywhere
NotificationCenter.default.post(name:
    NSNotification.Name("coolNotification"), object: nil)
```
Local Notifications

• User Notification Center
  – import UserNotifications
  – UNUserNotificationCenter.current()

• Request authorization to use notifications
  – requestAuthorization()

• Handle changes to authorizations
  – getNotificationSettings()
Local Notifications: Request Authorization

```swift
import UserNotifications // In AppDelegate.swift

// In didFinishLaunchingWithOptions
let center = UNUserNotificationCenter.current()
center.requestAuthorization(options: [.alert, .badge, .sound])
  { (granted, error) in
    // Enable or disable features based on authorization
  }
```

“NotifyDemo” Would Like to Send You Notifications
Notifications may include alerts, sounds, and icon badges. These can be configured in Settings.

Don’t Allow  Allow
Local Notifications: Handle Authorization Changes

// In ViewController

var alertNotificationsOkay = false // Similarly for badge and sound

func checkNotifications() {
    let center = UNUserNotificationCenter.current()
    center.getNotificationSettings { (settings) in
        self.alertNotificationsOkay = false
        if settings.alertSetting == .enabled {
            self.alertNotificationsOkay = true
        }
    }
}

// In AppDelegate

func applicationWillEnterForeground(_ application: UIApplication) {
    let vc = self.window?.rootViewController as! ViewController
    vc.checkNotifications()
}

Mobile Application Development in iOS
Scheduling Notifications

• (1) Create content
  – UNMutableNotificationContent

• (2) Create trigger
  – Based on time interval, date/time, location
  – UNTimeIntervalNotificationTrigger
  – UNCalendarNotificationTrigger
  – UNLocationNotificationTrigger
Scheduling Notifications

• (3) Create request
  – UNNotificationRequest

• (4) Schedule notification
  – UNUserNotificationCenter.add(request)
func scheduleNotification() {
    let content = UNMutableNotificationContent()
    content.title = "Hey!"
    content.body = "What’s up?"
    content.userInfo["message"] = "Yo!"
    // Configure trigger for 5 seconds from now
    let trigger = UNTimeIntervalNotificationTrigger(timeInterval: 5.0,
    repeats: false)

    // Create request
    let request = UNNotificationRequest(identifier: "NowPlusFive",
    content: content, trigger: trigger)

    // Schedule request
    let center = UNUserNotificationCenter.current()
    center.add(request) { (error : Error?) in
        if let theError = error {
            print(theError.localizedDescription)
        }
    }
}
Receiving Notifications

• AppDelegate conforms to

UNUserNotificationCenterDelegate

  – UNUserNotificationCenter.current().delegate = self

  – didReceive()
    • Called if app in background or not running

  – willPresent()
    • Called if app running in foreground
### Receiving Notifications

```swift
class AppDelegate: UIResponder, UIApplicationDelegate, UNUserNotificationCenterDelegate {

    // In didFinishLaunchingWithOptions
    UNUserNotificationCenter.current().delegate = self

    func userNotificationCenter(_ center: UNUserNotificationCenter, 
        didReceive response: UNNotificationResponse, 
        withCompletionHandler completionHandler: @escaping () -> Void) {

        print("user responded to notification while in background")
        // Do stuff with response here (non-blocking)
        let vc = self.window?.rootViewController as! ViewController
        vc.handleNotification(response) 
        completionHandler()
    }

    // In ViewController
    func handleNotification (_ response: UNNotificationResponse) {
        let message = response.notification.request.content.userInfo["message"] as! String
        self.messageLabel.text = message
    }
```

Called even if app wasn’t running.
Receiving Notifications While App in Foreground

```swift
// In AppDelegate
func userNotificationCenter(_ center: UNUserNotificationCenter, willPresent notification: UNNotification, withCompletionHandler completionHandler: @escaping (UNNotificationPresentationOptions) -> Void) {

    print("received notification while in foreground; display?")
    completionHandler([.alert]) // no options ([]) means no notification
}
```

Mobile Application Development in iOS
Local Notifications: Other Options

• Configure different categories of notifications
  – Add custom actions: UNNotificationAction
  – Create category: UNNotificationCategory

• Add custom sounds (< 30 seconds)
  – UNNotificationSound
Remote (Push) Notifications

- Enable push notifications capability in app
- Enable Remove Notifications Background Mode
- Configure push notifications on developer account
- Create SSL certificate
- Register for remote notifications
- Retain device token
- Use device token + SSL certificate to send notifications to Apple’s server
  - Will then be delivered to app on device
Enable Push Notifications

• This will also create App ID with Push Notifications set to Configurable
Enable Remote Notifications

- Background Modes
Configure Push Notifications

Certificate, Identifiers & Profiles

iOS App IDs

- Name: XC edu wsu NotifyDemo
- Identifier: edu.wsu.NotifyDemo

Application Services:
- Service: Development, Distribution
- App Groups: Disabled, Disabled
- Associated Domains: Disabled, Disabled
- Data Protection: Disabled, Disabled
- Game Center: Enabled, Enabled
- HealthKit: Disabled, Disabled
- HomeKit: Disabled, Disabled
- iCloud: Disabled, Disabled
- In-App Purchase: Enabled, Enabled
- Inter-App Audio: Disabled, Disabled
- Personal VPN: Disabled, Disabled
- Push Notifications: Configurable, Configurable
- Wallet: Disabled, Disabled
- Wireless Accessory Configuration: Disabled, Disabled

Push Notifications

- Configurable

Apple Push Notification service SSL Certificates

To configure push notifications for this iOS App ID, a Client SSL Certificate that allows your notification server to connect to the Apple Push Notification Service is required. Each iOS App ID requires its own Client SSL Certificate. Manage and generate your certificates below.

Development SSL Certificate

Create an additional certificate to use for this App ID.

Production SSL Certificate

Create an additional certificate to use for this App ID.

Wallet
- Disabled

Wireless Accessory Configuration
- Disabled
Create SSL (1)

- Create Certificate Signing Request (CSR)
  - Follow directions on CSR developer page
Create SSL (2)

- Upload CSR file
- Generate certificate
- Download certificate
- Double-click certificate to add to keychain
Create SSL (3)

- Export certificate from keychain
- Convert to PEM file

```
$ openssl pkcs12 -in PushCert.p12 -out PushCert.pem -nodes -clcerts
Enter Import Password:
MAC verified OK
```
Register for Remote Notifications

// In didFinishLaunchingWithOptions
// Register with APNs
UIApplication.shared.registerForRemoteNotifications()

func application(_ application: UIApplication,
didRegisterForRemoteNotificationsWithDeviceToken deviceToken: Data) {
    let deviceTokenString = deviceToken.reduce("", {
        $0 + String(format: "%02X", $1)"
    })
    print("device token = \(deviceTokenString)"
    // Something like (64 hex characters):
    // 87AE522E54CCCE0D7B126F0795A4B629B6ADAA095FF608CE4D0E854473F47E21
}

func application(_ application: UIApplication,
didFailToRegisterForRemoteNotificationsWithError error: Error) {
    // The token is not currently available.
    print("Remote notification support is unavailable due to error: \(error.localizedDescription)"
}
Send Push Notifications

• Example using Python PyAPNs

(https://github.com/djacob/PyAPNs)

```python
from apns import APNs, Payload

apns = APNs(use_sandbox=True, cert_file='PushCert.pem')

token_hex = '87AE522E54CCCE0D7B126F0795A4B629B6ADAA095FF608CE4D0E854473F47E21'
payload = Payload(alert="Hello World!", sound="default", badge=1,
    custom={'message': 'This is Apple.'}) # userInfo

apns.gateway_server.send_notification(token_hex, payload)
```
Resources

• Alerts
  – developer.apple.com/documentation/uikit/uialertcontroller

• Internal notifications
  – developer.apple.com/documentation/foundation/nsnotificationcenter

• Local and Remote (Push) Notifications
  – developer.apple.com/notifications/