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` (`UIAlertAction`)
    - `init` (title, style, handler)
      - Style: `.default`, `.cancel`, `.destructive`
  - `preferredAction` (bold on `.alert` style)
  - `addTextField` (configurationHandler)
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

...  

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

![Alerts: Demo](image-url)
func loginAlert() {
    let alert = UIAlertController(title: "Login",
                                  message: "Enter username and password.",
                                  preferredStyle: .alert)
    alert.addTextField(configurationHandler: { (textField) in
        textField.placeholder = "Username"
    })
    alert.addTextField(configurationHandler: { (textField) in
        textField.placeholder = "Password"
        textField.isSecureTextEntry = true
    })
    alert.addAction(UIAlertAction(title: "Login", style: .default,
                                  handler: { (action) in
            let username = alert.textFields?[0].text
            let password = alert.textFields?[1].text
            print("username = \(username!), password = \(password!)")
        })))
    present(alert, animated: true, completion: nil)
}
Internal Notifications

- Notify observers of an event
  - Observer and event must be in the same process
- NotificationCenter
  - addObserver
  - addObserver
  - post(Notification)
- Notification names
Internal Notifications

• Example: Detect change to UserDefaults
  – Only when app in foreground

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

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

• Example: Detect when app becomes active

```
// In viewDidLoad
NotificationCenter.default.addObserver(self, selector: #selector(appActive), name: UIApplication.didBecomeActiveNotification, object: nil)

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

• Define you own

```swift
// In viewDidLoad
NotificationCenter.default.addObserver(
    self, 
    selector: #selector(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 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
        if settings.alertSetting == .enabled {
            self.alertNotificationsOkay = true
        } else {
            self.alertNotificationsOkay = false
        }
    }
}

// In AppDelegate
func applicationDidBecomeActive(_ application: UIApplication) {
    let navVC = self.window?.rootViewController as! UINavigationController
    let mainVC = navVC.topViewController as! ViewController
    mainVC.checkNotifications()
}
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) 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
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")
        // Do stuff with response here (non-blocking)
        let navVC = self.window?.rootViewController as! UINavigationController
        let mainVC = navVC.topViewController as! ViewController
        mainVC.handleNotification(response)
        completionHandler()
    }

    // In ViewController
    func handleNotification(_ response: UNNotificationResponse) {
        let message = response.notification.request.content.userInfo["message"] as! String
        print("received notification message: \(message)")
    }

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

// 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
}
Local Notifications: Other Options

• Add sound (< 30 seconds)
  – UNNotificationSound

• Set app icon badge number
  – UNMutableNotificationContent.badge

• Configure different categories of notifications
  – Add custom actions: UNNotificationAction
  – Create category: UNNotificationCategory
Remote (Push) Notifications

- Enable push notifications capability in app
- Enable Remote 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 (didReceive, willPresent)
- Only works on real device (not simulator)
Enable Push Notifications

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

• Background Modes
Configure Push Notifications

Mobile Application Development in iOS
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

Mobile Application Development in iOS
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

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

func application(_ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    // Convert each byte to hex string and concatenate them
    let deviceTokenString = deviceToken.reduce("", { $0 + String(format: "%02X", $1) })
    print("device token = \\
           (deviceTokenString)"
    )
    // Something like (64 hex characters):
    // 87AE522E54CCCE0D7B126F0795A4B629B6ADAA095FF608CE4D0E854473F47E21

    return true
}

func application(_ application: UIApplication,
    didRegisterForRemoteNotificationsWithDeviceToken deviceToken: Data) {
    // Debug 输出 device token
    print("device token = \(deviceToken)")
}

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 (python2 only)

(\texttt{github.com/djacobs/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/notifications

• Local and remote (push) notifications
  – developer.apple.com/documentation/usernotifications