Notifications

Mobile Application Development in iOS

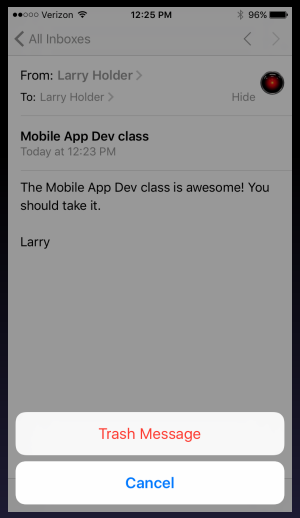
School of EECS

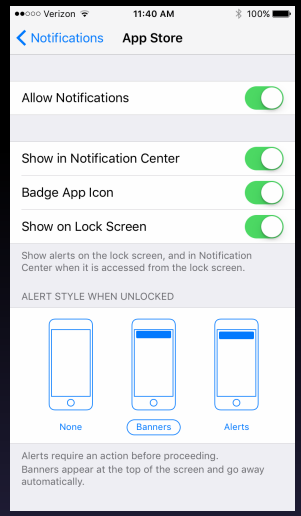
Washington State University

Instructor: Larry Holder

Outline

- Alerts
- 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
  - var preferredAction
  - addTextField (handler)

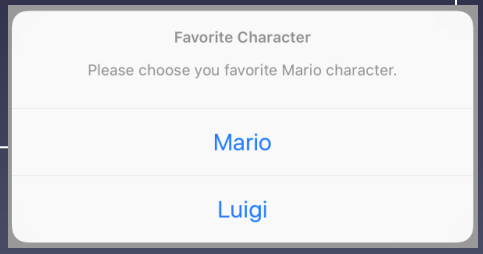
Alerts: Demo

```
let alert = UIAlertController(title: "Favorite Character",
  message: "Please choose you favorite Mario character.",
  preferredStyle: .actionSheet)

alert.addAction(UIAlertAction(title: "Mario", style: .default,
  handler: { (action) in
    // execute some code when this option is selected
    print("Favorite character is Mario")
  }))

alert.addAction(UIAlertAction(title: "Luigi", style: .default,
  handler: { (action) in
    // execute some code when this option is selected
    print("Favorite character is Luigi")
  }))

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



Alerts with Text Fields

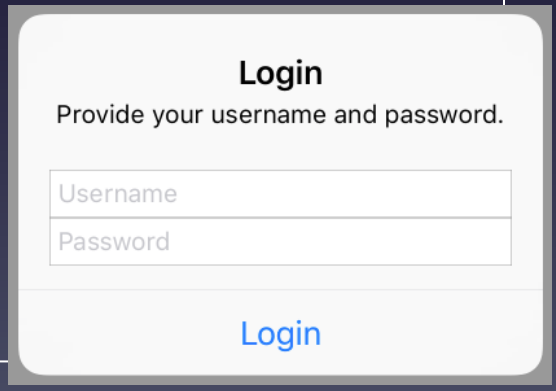
```
var loginAlert: UIAlertController!

func initializeLoginAlert() {
  loginAlert = UIAlertController(title: "Login",
    message: "Provide 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)
}
```



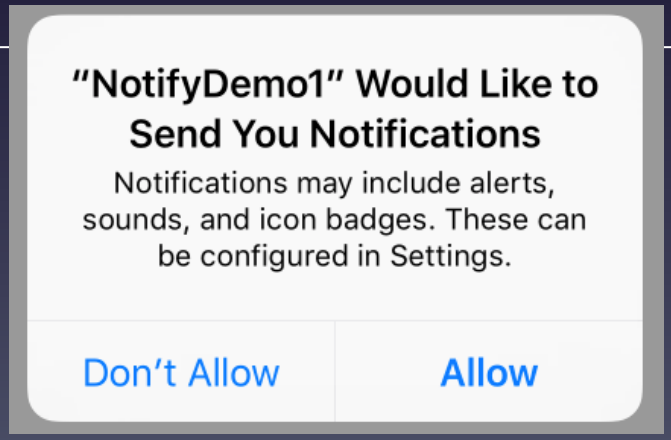
Local Notifications

- UNUserNotificationCenter
- Request authorization to use notifications
- Handle changes to authorizations

Local Notifications: Request Authorization

```
import UserNotifications // ususally in AppDelegate.swift

// Usually in application: didFinishLaunchingWithOptions
let center = UNUserNotificationCenter.current()
center.requestAuthorization(options: [.alert, .badge, .sound])
  { (granted, error) in
    // Enable or disable features based on authorization.
    let vc = self.window?.rootViewController as! ViewController
    vc.notificationsOkay = granted
  }
```



Local Notifications: Handle Authorization Changes

```
// Call from applicationDidEnterForeground or before notification
func checkIfNotificationsStillOkay() {
  let center = UNUserNotificationCenter.current()
  center.getNotificationSettings(completionHandler:
    self.handleNotificationSettings)
}

func handleNotificationSettings (notificationSettings:
UNNotificationSettings) {
  if ((notificationSettings.alertSetting == .enabled) &&
      (notificationSettings.badgeSetting == .enabled) &&
      (notificationSettings.soundSetting == .enabled)) {
    self.notificationsOkay = true
  } else {
    self.notificationsOkay = false
  }
}
```

Receiving Notifications

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

  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.handleNotification1(response)

    completionHandler()
  }
}
```

Called even if app wasn't running.

```
class ViewController: UIViewController {
  func handleNotification1 (_ response: UNNotificationResponse) {
    let message = response.notification.request.content.userInfo["message"]
                    as! String
    self.messageLabel.text = message
  }
}
```

Receiving Notifications While App in Foreground

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

  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
  }

...
}
```

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)`

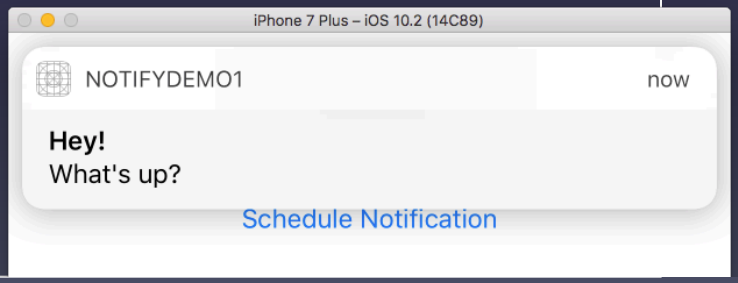
Scheduling Notifications

```
func scheduleNotification1() {
  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)
    }
  }
}
```



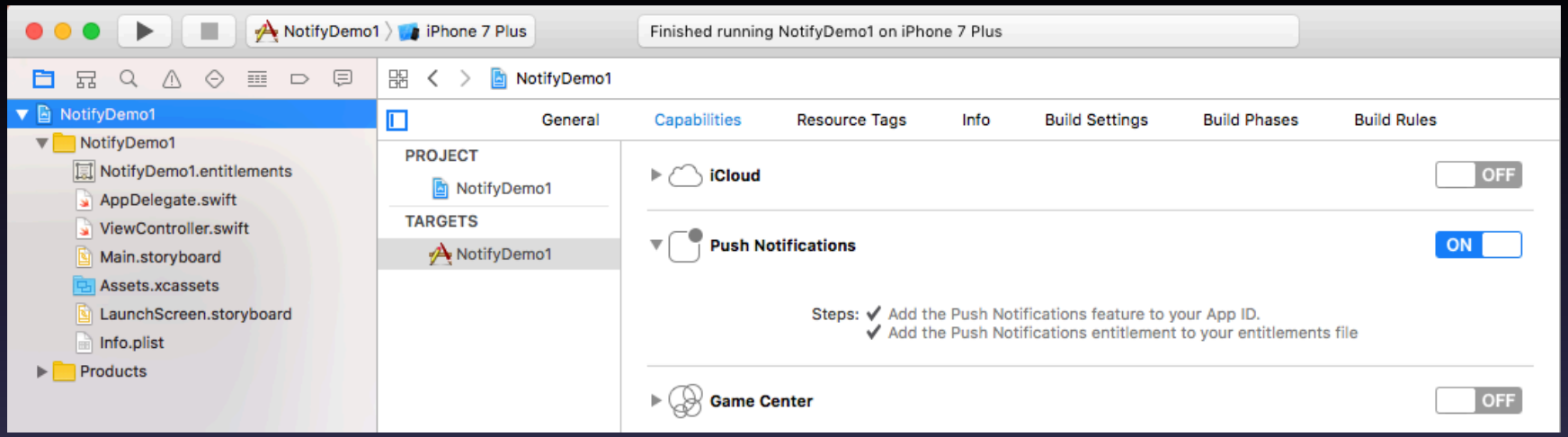
Local Notifications: Other Options

- Configure different categories of notifications
  - Add custom actions
  - `UNNotificationCategory`
  - `UNNotificationAction`
- Add custom sounds (< 30 seconds)
  - `UNMutableNotificationContent.sound`

Remote (Push) Notifications

- Enable push notifications capability
- Register for remote notifications
- Retain device token
- Create SSL certificate
- Use device token + SSL certificate to send notifications to Apple's server
  - Will then be delivered to app on device

Enable Push Notifications



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)")
}

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

Resources

- Alerts
  - https://developer.apple.com/ios/human-interface-guidelines/ui-views/alerts/
- Local and Remote (Push) Notifications
  - https://developer.apple.com/notifications/
- Good push notifications tutorial
  - https://www.raywenderlich.com/123862/push-notifications-tutorial