Settings

Mobile Application Development in iOS
School of EECS
Washington State University
Instructor: Larry Holder
Outline

• In-app settings
  – UserDefaults
• Device settings
  – Settings Bundle
• Settings across devices and app groups
  – iCloud key-value store
Local Settings: UserDefaults

- Persistent key-value store for app
  - Accessed via singleton class instance: UserDefaults.standard

- Setting values
  - UserDefaults.standard.set(value: <type>, forKey: String)
  - <type>: Bool, Int, Float, Double, URL?, Any?

```swift
let userName: String = "larry" // safer to force the type
let sortItems: Bool = true
let timeLimit: Int = 5
UserDefaults.standard.set(userName, forKey: "userName")
UserDefaults.standard.set(sortItems, forKey: "sortItems")
UserDefaults.standard.set(timeLimit, forKey: "timeLimit")
```
Sidebar: Singleton Class

• Create one, and only one, instance of a class

```swift
class Globals {
    static let sharedInstance = Globals()

    // Properties and methods
    var userName = ""
    var sortItems = true
    var timeLimit = 5

    // Prevent calling init() directly on class
    private init() {}
}

// Access anywhere
Globals.sharedInstance.userName = "larry"
```
UserDefaults

• Getting values
  – UserDefaults.standard.<type>(forKey: String)
  – <type>: bool, integer, float, double, string, url, dictionary, array, data, object
  – Returns nil if not set, except for bool, integer, float, double

• Removing values
  – UserDefaults.standard.removeObject(forKey: String)

```swift
// Usually in AppDelegate didFinishLaunchingWithOptions
if (UserDefaults.standard.object(forKey: "sortItems") == nil) {
    UserDefaults.standard.set(true, forKey: "sortItems")
}
// Later...
let sortItems = UserDefaults.standard.bool(forKey: "sortItems")
```
Settings Demo

Navigation Controller

Table View
Static Content

Mobile Application Development in iOS
Device Settings: Settings Bundle

- Add Settings Bundle to app
Settings Bundle

- Modify root.plist to create settings
Settings Bundle

Mobile Application Development in iOS
Accessing Settings Bundle

• Set in device’s Global Settings

• Access in UserDefaults as before

```swift
let category = UserDefaults.standard.string(forKey: "category")
let version = UserDefaults.standard.string(forKey: "version")

UserDefaults.standard.set("movies", forKey: "category")
UserDefaults.standard.set("1.0", forKey: "version")
```

• Warning: Device settings undefined until changed
Accessing Settings Bundle

• How to detect when device settings changed?

• In AppDelegate:

```swift
func applicationWillEnterForeground(_ application: UIApplication) {
    let navController = self.window?.rootViewController as! UINavigationController
    let settingsTVC = navController.topViewController as! SettingsTableViewController
    settingsTVC.resetSettingsView() // Our method to update view
}
```

• Or, send notification to app when device settings changed (next time...)
Open Device Settings from App

```swift
if let settingsURL = URL(string: UIApplication.openSettingsURLString) {
    if UIApplication.shared.canOpenURL(settingsURL) {
        UIApplication.shared.open(settingsURL, options: [:], completionHandler: nil)
    }
}
```
Settings across Apps: iCloud

• iCloud
  – Store key-values, documents, data
  – Automatically synced across all devices/apps

• CloudKit
  – Public storage for all users
  – Private storage per user
  – Requires manual synchronization

• App can be notified of changes (next time...)

Mobile Application Development in iOS 13
iCloud Key-Value Store

- Settings across apps & devices
- Add iCloud capabilities to app
- Need an App ID with iCloud entitlements
iCloud Key-Value Store (1)

• Create new project
• Give it a unique name, not previously used to generate App ID
• In Capabilities, turn on iCloud
  – Key-value storage already checked
iCloud Key-Value Store (1)
iCloud Key-Value Store (2)

- Login to developer account
- Will need at least one real device
- Make sure device is registered on developer account
- Under Certificates, make sure you have one for your development environment
  - Should be auto-generated
- Create iCloud Container and register
iCloud Key-Value Store (2)

Registering your iCloud Container lets you use the iCloud Storage APIs to enable your apps to store data and documents in iCloud, keeping your apps up to date automatically.

iCloud Container Description

Description: iCloud Demo Container
You cannot use special characters such as @, *, ", "

Identifier
Enter a unique identifier for your iCloud Container, starting with the string 'iCloud'.

ID: iCloud.edu.wsu.iCloudDemo
We recommend using a reverse-domain name style string (i.e., com.domainname.appname).
iCloud Key-Value Store (3)

• Create explicit App ID for this app
  – Wildcard App ID won’t work for iCloud use
  – Check iCloud App Services (include CloudKit)

• Edit App ID
  – Select and assign your iCloud container
  – Register
iCloud Key-Value Store (3a)

### Registering an App ID

The App ID string contains two parts separated by a period (.) — an App ID Prefix that is defined as your Team ID by default and an App ID Suffix that is defined as a Bundle ID search string. Each part of an App ID has different and important uses for your app. Learn More

#### App ID Description

- **Name:** iCloud Demo
  
  You cannot use special characters such as @, &, *, ', "

#### App ID Prefix

- **Value:** 3JN3V4335U (Team ID)

#### App ID Suffix

- **Explicit App ID**
  - If you plan to incorporate app services such as Game Center, In-App Purchase, Data Protection, and iCloud, or want a provisioning profile unique to a single app, you must register an explicit App ID for your app.

  To create an explicit App ID, enter a unique string in the Bundle ID field. This string should match the Bundle ID of your app.

  Bundle ID: `edu.wsu.iCloudDemo`

  **We recommend using a reverse-domain name style string (i.e., com.domainname.appname). It cannot contain an asterisk (*)**.

- **Wildcard App ID**
  - This allows you to use a single App ID to match multiple apps. To create a wildcard App ID, enter an asterisk (*) as the last digit in the Bundle ID field.

  Bundle ID: `Example: com.domainname.*`

---

### App Services

Select the services you would like to enable in your app. You can edit your choices after this App ID has been registered.

#### Enable Services:

- **App Groups**
- **Associated Domains**
- **Data Protection**
  - Complete Protection
  - Protected Unless Open
  - Protected Until First User Authentication
- **Game Center**
- **HealthKit**
- **HomeKit**
- **iCloud**
  - Compatible with Xcode 5
  - Include CloudKit support (requires Xcode 6)
- **In-App Purchase**
- **Inter-App Audio**
- **Personal VPN**
- **Push Notifications**
- **Wallet**
- **Wireless Accessory Configuration**
iCloud Key-Value Store (3b)
iCloud Key-Value Store (4)

- Create Provisioning Profile
  - iOS App Development
  - Select newly-created App ID
  - Select your development certificate(s)
  - Select your devices
  - Give profile a name

- Back in Xcode
  - Uncheck automatically manage signing
  - Select new provisioning profile
iCloud Key-Value Store (4a)
iCloud Key-Value Store (4b)
iCloud Key-Value Store (4c)
iCloud Key-Value Store (4d)
// Initialize key-value store
var keyStore = NSUbiquitousKeyValueStore()

// viewDidLoad
keyStore.synchronize()

// Set a key to a value
keyStore.set(value, forKey: "keyName")
keyStore.synchronize() // optional if you can wait

// Read value of key (mind your latency)
if let storedString = keyStore.string(forKey: "keyName") {
    print("Cloud Value = \(storedString)"")
} else {
    print("Cloud Value = empty")
}
iCloud Key-Value Store: Testing

• Run app on simulator and device (or multiple devices)
  – All signed in to same iCloud account
• Changes via simulator propagate to device(s)
• Propagation to simulator requires "Debug ➔ Trigger iCloud Sync" each time
Resources

• User Defaults
  – developer.apple.com/documentation/foundation/userdefaults

• iCloud and CloudKit
  – developer.apple.com/documentation/documentation/cloudkit