Mobile Application Development
Aileen Pierce

DATA PERSISTENCE

Data Persistence

- Model objects written to hold data should support archiving.
- Archiving model objects lets you write objects to a file and then read them back in.
- The most common ways to handle archiving in iOS are:
  - Property lists
  - Object archives
  - SQLite3
  - Core Data

Sandbox

- Your app sees the iOS file system like a normal UNIX file system
- Every app gets its own /Documents directory which is referred to as its sandbox
- Your app can only read and write from that directory
  - Security
  - Privacy
  - Cleanup

Sandbox

- In the Finder go into your home directory
- Go to Library/Developer/CoreSimulator/Devices/*Device UDID*/data/Containers/Data/Application
  - each app has subdirectories
    - Documents-app sandbox to store its data
    - Library-user preferences settings
    - Tmp-temp files
- The same file structure exists on devices

File System

- **`NSSearchPathForDirectoriesInDomain()`** is a C function that will locate a directory.
- Retrieve the path to the Documents directory

```
let path = NSSearchPathForDirectoriesInDomains
(NSSearchPathDirectory.DocumentDirectory,
NSSearchPathDomainMask.AllDomainsMask, true)
```

  - Returns an array with the documents directory in index 0

- Create a string with the full path name

```
let dir = paths[0] as NSString
let
file=dir.stringByAppendingPathComponent(filename)
```

File System

- The **`NSFileManager`** class enables you to perform many generic file-system operations
  - Check to see if a file exists
  - Manipulate files (move, copy, delete, etc)
- The **`UIApplication`** class provides a centralized point of control and coordination for iOS apps.

Property Lists

- Only certain objects can be stored in property lists and then written to a file using the **`writeToFile(atomically)`** method
  - **`NSArray`**
  - **`NSDictionary`**
  - **`NSData`**
  - **`NSString`**
  - (and the mutable versions of the above)
  - **`NSNumber`**
  - **`NSDate`**

Communication

In iOS there are four common patterns for objects to communicate

1. Target-Action: a single object calls a single method when a single event occurs
2. Delegation: an object responds to numerous methods to modify or add behavior
3. Notification: Register an object to be notified when an event occurs
4. Key-Value Observing (KVO): register to be one of many objects notified when single property of another object changes.

Notifications

- A notification is a callback mechanism that can inform multiple objects when an event occurs
- A **`NSNotificationCenter`** manages the notification process
- Objects register for the notifications they're interested in
- Notification senders post notifications to a notification center.
- The notification center notifies any objects registered for that notification.