

Announcements

- **Lab 3 is due on Wednesday by 11:59 PM**

Today's Topics

- **Property Lists**
- **iPhone's File System**
- **Archiving Objects**
- **SQLite**
- **Web Services**

Storage on the iPhone

Property Lists

Property Lists

- **Convenient way to store a small amount of data**
 - Arrays, dictionaries, strings, numbers, dates, raw data
 - Human-readable XML or binary format
- **UserDefaults class uses property lists under the hood**



When Not to Use Property Lists

- **More than a few hundred KB of data**
 - Loading a property list is all-or-nothing
- **Complex object graphs**
- **Custom object types**

Property List Demo

iPhone's File System

Keeping Applications Separate



Why Keep Applications Separate?

- Security
- Privacy
- Cleanup after deleting an app

Home Directory Layout

- Each app has its own set of directories
- <Application Home>
 - MyApp.app
 - MyApp
 - MainViewController.storyboard
 - Somelimage.png
 - Documents
 - Library
 - Caches
 - Preferences
- Applications only read and write within their home directory

Demo



File Paths in Your Application

```
// Basic directories
let homeDir = NSHomeDirectory()
let tmpDir = NSTemporaryDirectory()

// Documents directory
let paths =
    NSSearchPathForDirectoriesInDomains(.documentDirectory,
                                        .userDomainMask, true)

let documentsPath = paths[0]

// <Application Home>/Documents/foo.plist
let filePath = docsDir + "/foo.plist"
```

Including Writable Files with Your App

- **Many applications want to include some starter data**
- **But application bundles are code signed**
 - You can't modify the contents of your app bundle
- **To include a writable data file with your app...**
 - Build it as part of your app bundle
 - On first launch, copy it to your Documents directory

Archiving Objects

Archiving Objects

- **Next logical step from property lists**
 - Include arbitrary classes
 - Complex object graphs
- **Utilize the Codable protocol introduced in Swift 4**

Archiving Example

```
struct Book: Codable {
    var title: String
    var author: String
    var pageCount: Int

    enum CodingKeys: String, CodingKey {
        case title; case author; case pageCount
    }

    init(title: String, author: String, pageCount: Int) {
        self.title = title; self.author = author; self.pageCount = pageCount
    }

    func encode(to encoder: Encoder) throws {
        ...
    }

    init(from decoder: Decoder) throws {
        ...
    }
}
```


Archiving Example

```
func encode (to encoder: Encoder) throws {
    var container = encoder.container(keyedBy: CodingKeys.self)

    try container.encode(title, forKey: "title")
    try container.encode(author, forKey: "author")
    try container.encode(pageCount, forKey: "pageCount")
}
```

Archiving Example

```
init (from decoder: Decoder) throws {
    let container = try decoder.container(keyedBy: CodingKeys.self)
    title = try container.decode(String.self, forKey: .title)
    author = try container.decode(String.self, forKey: .author)
    pageCount = try container.decode(Int.self, forKey: .pageCount)
}
```

Archiving & Unarchiving using Codable

- **Creating an archive**

```
let data = try PropertyListEncoder().encode(books)
let success = NSKeyedArchiver.archiveRootObject(data, toFile:"path/to/archve")
```

- **Decoding an archive**

```
guard let data= NSKeyedUnarchiver.unarchiveObject(withFile: "path/to/archive")
  as? Data else { return }
let books = try PropertyListDecoder.decode([Book].self,from data)
```

Archiving Demo

SQLite

SQLite

- **Complete SQL database in an ordinary file**
- **Simple, compact, fast, reliable**
- **No server**
- **Great for embedded devices**
 - Included on the iPhone platform

When Not to Use SQLite

- **Multi-gigabyte databases**
- **High concurrency (multiple writers)**
- **Client-server applications**
- **“Appropriate Uses for SQLite”**
<http://www.sqlite.org/whentouse.html>

More on SQLite

- **“SQLite in 5 Minutes Or Less”**
– <http://www.sqlite.org/quickstart.html>
- **“Intro to the SQLite C Interface”**
– <http://www.sqlite.org/cintro.html>
- **FMDB SQLite Wrapper**
– <https://github.com/ccgus/fmdb>

SQLite Demo - FMDB

Core Data

- **Object-graph management and persistence framework**
 - Makes it easy to save and load model objects
 - Properties
 - Relationships
 - Higher-level abstraction than SQLite or property lists

Web Services

Your Application & The Cloud

- Store & access remote data
- May be under your control or someone else's
- Many Web 2.0 apps/sites provide developer API

Integrating with Web Services

- **Non-goal of this class: teach you all about web services**
 - Plenty of tutorials accessible, search on Google
- **Many are exposed with XML or JSON**
- **High level overview of parsing these types of data**

XML

Options for Parsing XML

- **XMLParser**
 - Event-driven API
 - <https://developer.apple.com/reference/foundation/xmlparser>
- **SWXMLHash**
 - Simple Swift XML Parsing
 - <https://github.com/drmohundro/SWXMLHash>

JSON

JavaScript Object Notation

- **More lightweight than XML**
- **Looks a lot like a property list**
 - Arrays, dictionaries, strings, numbers
- **Open source json-framework wrappers for Swift and Objective-C**

What does a JSON string look like?

```
{  
  "instructor" : "Todd Sproull",  
  "students" : 20,  
  "itunes-u" : true,  
  "midterm-exam" : null,  
  "assignments" : [ "WhatATool",  
                   "HelloPoly"]  
}
```

More on JSON

- **“Introducing JSON”**
 - <http://www.json.org/>
- **Encoding and Decoding Custom Types**
 - https://developer.apple.com/documentation/foundation/archives_and_serialization/encoding_and_decoding_custom_types
- **JSON Editor**
 - <https://www.jsoneditoronline.org/>

Recap

- **Property lists**
 - Quick & easy, but limited
- **Archived objects**
 - More flexible, but require writing a lot of code
- **SQLite and Core Data**
 - Elegant solution for many types of problems
- **XML and JSON**
 - Low-overhead options for talking to “the cloud”

JSON Demo

More on JSON

- **“Introducing JSON”**
 - <http://www.json.org/>
- **SwiftyJSON**
 - <https://github.com/SwiftyJSON/SwiftyJSON>
- **JSON Editor**
 - <http://www.jsoneditoronline.org/>

Recap

- **Property lists**
 - Quick & easy, but limited
- **Archived objects**
 - More flexible, but require writing a lot of code
- **SQLite and Core Data**
 - Elegant solution for many types of problems
- **XML and JSON**
 - Low-overhead options for talking to “the cloud”

Firestore Demo

Cloud Firestore