Learning Summary Report

Due 10 am Monday 14 Nov 2011

Create Data-Driven Mobile Application Assignment (HIT3329)

By Chao Zhao 6555985

# Overview

Learning summary reports summaries the main topics and knowledge that have covered from the Create Data Driven Mobile Application with my opinions and thoughts. The summary organised topic areas by weekly learning. The development on mobile device is a new concept for me. I have never developed any application on mobile platform and tablet. The mobile device has touchable screen that different with desktop. The performance and power is limited. The developers should use limited resource to provide good quality software to users. The official SDK is called Xcode. I think the most wonderful parts are interface builder. With Interface builder, developers can draw UI elements by mouse instead draw programmatically. The Objective C official supported languages in iOS platform and Mac. It is a dynamic typing languages that complier will not check the type in complier errors. Developers can use core data to mange the local data. Advantages of core data are persistence, simple and powerful. However, it is not a database. The Application should prepare to work with airplane mode. This is due to mobile phone has unreliable networking.

# Self-Assessment

|  |  |  |  |
| --- | --- | --- | --- |
| **Intended Learning Outcome (ILO 1):** Build a data-driven mobile application that reads data from Web APIs, interprets the data on the device, and publishes user data back via an API. | | | |
| **Adequate** | **Good** | **Outstanding** | **Exemplary** |
| Weekly assignments |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Intended Learning Outcome (ILO 2):** Design and build a mobile application that degrades gracefully under limiting conditions and maintains a good user experience. | | | |
| **Adequate** | **Good** | **Outstanding** | **Exemplary** |
| Weekly assignments | Assignment 7 | Assignment 7 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Intended Learning Outcome (ILO 3):** Create a mobile application that provides speed and a good user experience by caching data and making asynchronous API calls. | | | |
| **Adequate** | **Good** | **Outstanding** | **Exemplary** |
| Weekly assignments | Assignment 6 |  |  |

# Reflection

## Operation System on Mobile Device

In First Week, we are introduced iOS operation system. The mobile operation system is not like a PC or Mac. It has limit environment compare a desktop, likes

* Limit battery life,
* Small screen without WIMP (Windows, Icons, Menus, Pointers).
* Limit memory and CPU
* Poor signal (GPS, WIFI, 3G network)

The phone is very tiny. It is a good design for portable. However, the battery may dead anytime. User needs to use finger to control application via pinch, click (touch), swipe and multiple touching. The Application should provide large icons to users to easily click. The memory and CPU is reduced than desktop. Developers need to reduce the use of resource. Due to a mobile device, it uses wireless networking. Wireless is very week than wired cable. A app in mobile device should automatically recover failures.

Also, the Objective C and SDK (Xcode) are introduced. Objective C is a dynamic typing language. It pass message instead of ‘call’. The Xcode is official SDK to develop App in iOS. It integrated Interface Builder to drawing interface by mouse.

## Objective C

In the week 2 lectures, detailed Objective C is introduced to everyone. Objective C syntax is different with almost other languages.

[aObject doSomething:value];

In the Objective C, getter and setter method can implement with @property and @synthesize. The Objective C will automatically create the getter and setter method.

The manual memory management is very important concept in iOS development. As I mentioned in wee 1, the mobile device has limited memory, developer must allocate the system memory manually instead of automatically. The automatically memory management does not release memory immediately.

The Objective C uses reference counting to manage memory usage. Below are the rules.

1. You own any object you create
2. You take ownership of an object using retain
3. When no longer needed, you must relinquish ownership of an object you own (release)
4. You must not relinquish ownership objects you do not own

The way that Objective C manages memory is different with other languages that I learnt in the past, like C/C++, Java.

## Delegate and Protocol

In week 3 lectures, I learnt delegate and Protocols in Objective C. It likes Interface in Java and multiple (virtual) inherent in C++. The delegate is a design pattern, which means an object needs to perform an action, but it needs other object’s help. It makes the code weak coupling. With the decoupling relationships, the code can be easily tested, modified and replaced.

I have some experience in C++ and Java. In C++, I have never use virtual inherent in software development. In Java, I used interface (protocol) to implement my application. I thought that delegate could make code more complicated. I found many bugs and error with delegate. However, It makes each component independents, developer can modify each component more easily. It is useful in object-oriented programming.

Last topic is application delegate. Within a mobile device, operation system need save or suspend application states to save the power resource. It is a huge different with desktop operation system. As I never consider this issue in desktop application development. The method is declare in

[AppName]AppDelegate.h/.m.

When the application states are changed, the method will pass message automatically to application object. Application can save object in memory to disk storage or load to memory. (See details about assignment 6)

## UIKit

In the week 4 lectures, I learnt how to draw UIViews and how to navigate between them. MVC (Model-View-Controller) can help me to have a better understanding with UIKit. Model is data source in application; view is XIB (or nib) file; controller is subclass of UIViewController.

The very good design pattern is called Fly Weight Pattern introduced. With this pattern, a UITableView use a queue to save the cell that scroll off screen. When a cell scrolls in screen, cell queues. System just change contents then display in the screen. Apple designed this pattern due to mobile device’s limit memory. It saves much memory especially in a large number of cells, without create all cells in a table view.

The UINavigationController is used to push or pop a UIVIew. It works like stack. The current UIView is always on the top of stack.

## Core Data

Now, the social network is very popular in the world. To design an application, developers need manage data. The Application needs to load data from disk to memory or save data from memory to disk. Apple design Core Data to help developer to mange data. However, Core Data is a database. In the Core Data, entity is represent table in database. It could have many entities and relationships in application. We can use Core Date to create, retrieve, update, delete (CRUD), sort and search data objects.

If developers want to mange a large amount of data, they need use SQLite. SQLite is local database system.

## Networking

Networking is important part of mobile devices, however mobile devices have unreliable networking. The network may dead due to weak signal. As a developer, we need to recovery any failures. The solution is saving data from Internet in disk storage and avoids requesting duplicated data. This will save users’ power and memory resource. 3G Internet is slow. It may take few seconds to download or upload to Internet. The Application needs to provide visual feedback to users that the application is not dead. A UIActivityIndicatorView, UIProgressView, UIAlertView and UILabel can indicate user networking’s progress. The networking transmission, local disk storage ,GPS locating should run in multiple threads. Otherwise, user may feel the application is dead. It will decrease the user experience.

## Data Management

To communication between server and client, the data is formatted. To improve networking transmission, the JSON format is introduced in mid-2000s. It is much faster than XML at parsing speed and data size. (Figure 1)

The iOS SDK does not support native parsing JSON format. We can use some library somebody already made. It can save our work.

When the application is downloading data source from Internet. User may cancel the threads via press ‘back’ button in application’s UI. The UIView is released, but the thread still sends to feedback to UIView. The Application will crush. The good application should cancel the thread in background.

## UI & UX

To design an application, developer must consider user interface (UI) and user experience (UX).

The bad interface may cause 1 start review in ‘App Store’, Apple provide Human Interface Guide (HIG). It can help developer to build a better UI.

* Users don’t like alert view, the alerts should avoid as below reason:
* Feedback for process finished
* Ask for confirmation of user-initiated actions, used action sheets instead
* Avoiding Springboard

No matter how nice is the icon, the springboard should be avoided, and it takes users more time to search items. List style of table view should be used.

* Apple recommend the minimum tip-size is 44 x 44 points
* Visual feedback with a long time process

UIActivityIndictorView and UIProgressView should use to indicate process states.

* Orientation – more contents not different contents

When orientation is changed. More contents should load to view. App should not ask user to change the orientation except games application.

# Mind Map



# Reference:

*Advanced Memory Management Programming Guide*, Apple, Accessed on 9th November 2011, <http://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/MemoryMgmt/Articles/mmRules.html>

*iOS Human Interface Guidelines*, Apple, accessed on 9th November 2011, <http://developer.apple.com/library/ios/#documentation/userexperience/conceptual/mobilehig/Introduction/Introduction.html>

# Appendixes

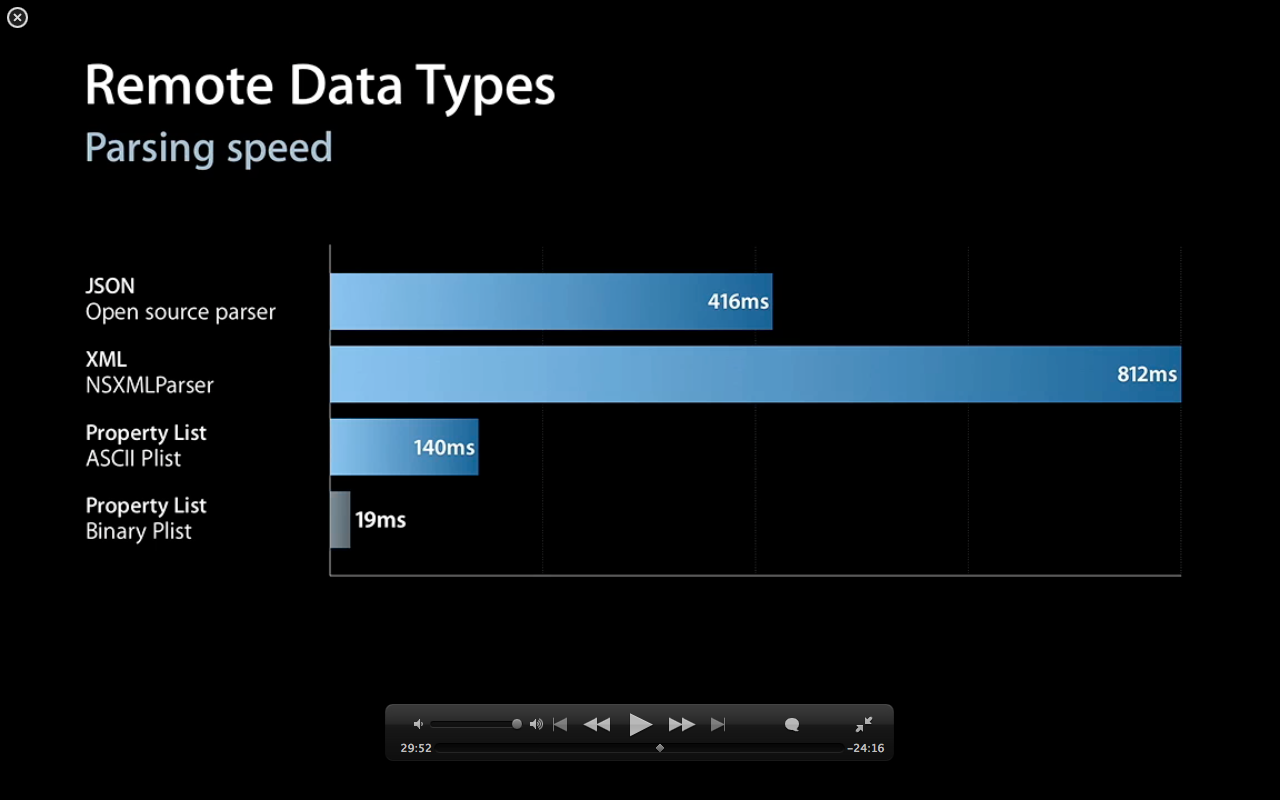


Figure 1

Source: WWDC 2010 - Build a Server-driven User Experience