CSC 540 Mobile App Development II

Adam Steele
DePaul University

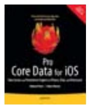
Outline

Administrivia
iPhone App Design
Project
- Annotated Bibliography

Assignment
- Assignment #3

Administrivia

Book(s)
- Primary Text



ISBN:9781430233558
http://library.books24x7.com.ezproxy2.lib.depaul.edu/bookshelf.asp
Do a search for the book

Administrivia

Book(s)
- Supplemental Text

ISBN:9781430230243
http://library.books24x7.com.ezproxy2.lib.depaul.edu/bookshelf.asp
Do a search for the book

- We will also make use of readings and papers
  - **Designing From Both Sides of the Screen,** Ellen Isaacs, ISBN 978-0672321511

Administrivia

Grading

• Assignments	50%
• Final project	40%
• **Proposal, Paper and Presentation**	
• Attendance and Participation:	10%

Participation for DL students will be evaluated based on submissions to the forums

Plagiarism & Incompletes
- Review relevant sections of website

iPhone App Design

iPhone Platform

iPhone App Design

Screen Resolution

320x480
- Old devices iPhone, 3G, 3GS, iPod Touch

640x960
- iPhone4 and 4th gen iPodTouch

1024x768
- iPad

DePaul CDM 7

iPhone App Design

We are going to look at cross-platform issues in this class
- iOS
  - **iPhone, iPad**
- Other platforms
  - **Custom Frameworks**
  - **New Web technologies (e.g HTML5 & jQuery Mobile, etc.)**

DePaul CDM 8

iPhone App Design

Main language iOS uses is Objective-C (Obj-C)
- SDK, frameworks, libraries, samples are in Obj-C
- First part of this class is 100% Obj-C

Obj-C is an old language
- C++ (pre processor) and Obj-C were competing
- 20 years ago
- Now it's back

DePaul CDM 9

iPhone App Design

Java
- The iPhone does not support Java applications of any kind. Steve Jobs has been quoted as saying
  - **"Java's not worth building in. Nobody uses Java anymore. It's this big heavyweight ball and chain.“**
- Java is heavily used in Android development
- C# the new and improved Microsoft version of Java and is a mainstay for Windows Phone (WP7)

DePaul CDM 10

iPhone App Design

Invention of C programming language
- Dennis Ritchie at AT&T (Bell Labs) invented C early 1970's

Objective-C

- Brad Cox designed Objective C in early 1980's
  - **Based on Smalltalk (used to program the Xerox Alto)**
  - **Added objects to C (originally from Simula 67)**

Contemporary to C++
- Invented in 1979, by Bjarne Stroustrup starting in 1979 at Bell Labs (originally a preprocessor)
- Originally called "C with classes"

DePaul CDM 11

iPhone App Design

Competing solutions
- C++ & Obj-C (C++ won out in the 80's and 90's)

NeXT Software
- Early adopter of Obj-C in 1988
- Apple acquired NeXT Step in 1996
  - **When Steve Jobs returned (post Scully)**

Obj-C became the basis for their Operating system
- Dev language for all current Apple OS platforms

DePaul CDM 12

iPhone App Design

Apple needed

- Memory managed language
- Different and proprietary
- Keeping it unique, keeps them in the Apple family
  - **Locked-in (but the prison is very comfortable)**

iPhone

- Obj-C – Enabled a break away from standard web dev./app paradigm
- Start over with new model of interaction
  - **Especially on the mobile platform**
    - iPod, iPhone, iPad

DePaul CDM 13

iPhone App Design

Need to develop on a Mac PC

- Why?
  - **They sell software and hardware**
  - **Get more people to buy their stuff**

I have a PC but not a mac

- Use the labs at school or borrow a Mac
  - **Cheap MacBook, Mac mini**
- Mac clones (Hackintoshes) don't always work
  - **No excuses**
    - Your code needs to compile and run in XCode

DePaul CDM 14

iPhone App Design

DePaul has a Educational Developer's License

- No need to purchase the developer's license
- You can do everything with this license except for publishing to the store and getting beta drops.

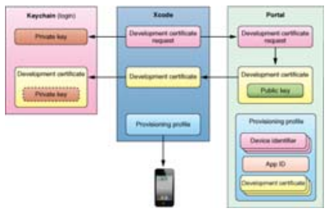
Dev Site:

- http://developer.apple.com/ios/
- Pretty much everything is on here (docs, code, *etc*.)
- I have sent out emails inviting you to the Dev Site

DePaul CDM 15

iPhone App Design

Provisioning



- You only need this if you want to run on a device
  - **The iOS emulator will be just fine for the moment**

DePaul CDM 16

iPhone App Design

UUID

- How do I get the UUID of my iPod Touch or iPhone or iPad
  - **Need that crazy number for hardware target**
- Answer
  - **Use iTunes look at the serial number, click over it, write it down.**
    - If you hit ctrl-C / AppleKey-C in iTunes when the UUID is displaying it will actually copy it to the clipboard even though it isn't highlighted.
  - Use the Organizer open your device from xCode, the number is selectable

DePaul CDM 17

iPhone App Design

Interface Editor

- Creates graphical entities
- Associates to links in source code
- Reduces programming

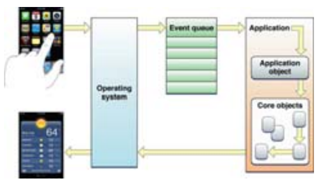
Cause great frustration

- Better than most GUI development
- Hard to debug
- We will avoid this for the moment
  - **To and extent we will be able to skip over it and use StoryBoards**

DePaul CDM 18

iPhone App Design

Application Event Loop



DePaul CDM 19

iPhone App Design

MVC

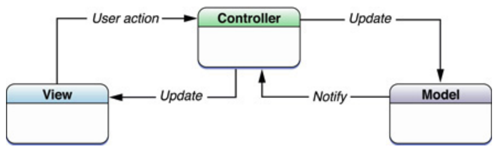
- Model-View-Controller
  - Logical way to divide up GUI's
- Model
  - The classes that hold your application's data
- View
  - Made up of the windows, controls, and other elements that the user can see and interact with
- Controller
  - Binds the model and view together
  - Application logic that decides how to handle the user's input

DePaul CDM 20

iPhone App Design

Model-View-Controller Paradigm

- One of the Gang-of-Four (GoF) Patterns

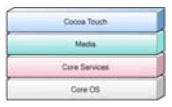


DePaul CDM 21

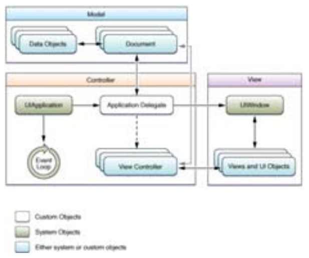
iPhone App Design

Fundamentals

IOS Layers



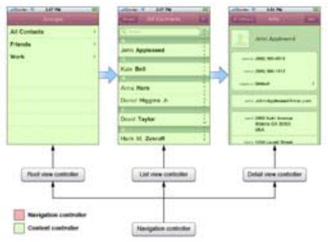
Object Layers (MVC)



DePaul CDM 22

iPhone App Design

Multiple Controllers



DePaul CDM 23

iPhone App Design

Cocoa Touch

- When you write Coco Touch applications
  - Create view components using interface builder (XIB or NIB) files
  - Sometimes modify interface in code
  - Might subclass existing views and controls
- Controller component
  - Usually composed of class that your create
  - Specific to the application
- Code you write needs to be associated to a specific category
  - Model, View, or Controller

DePaul CDM 24

iPhone App Design

Widgets

- Outlet
  - **Controller class can refer to objects in the nib**
  - **Special kind of instance variable called an Outlet**
- Outlets are instance variables that are declared using the keyword **`IBOutlet`**.
  - **`IBOutlet UIButton *myButton;`**
- Any instance variable your create and want to connect to an Object in the nib, must be preceded by **`IBOutlet`** keyword

DePaul CDM 25

iPhone App Design

Actions

- Interface objects in our Nib file can be set up to trigger special methods in our controller class.
  - **Special methods are called Actions.**
- **Actions** are methods that are part of your controller class
  - **Keyword `IBAction`**
- Example
  - **`-(IBAction)doSomething:(id)sender;`**

DePaul CDM 26

iPhone App Design

Delegates:

- Classes that take responsibility for doing certain things on behalf of another object.

Every iPhone application has one and only one instance of UIApplication,

- Responsible for the application's run loop
- Handles application-level functionality
  - **Such as routing input to the appropriate controller class**

DePaul CDM 27

iPhone App Design

`*.pch`
- **Precompiled header**

`*.m`
- **objective-C files**

`main.m`
- **main() function – jump off point for the application**

`*.plist`
- **Property List**

`*.xib`
- **Resources - contains images, sounds, movies and data**
- **Nib file**
  - **Nib files come from OSX's predecessor NextStep**
  - **So many things start with NS or N**

DePaul CDM 28

iPhone App Design

Sections

- Interface section
  - **Describes the data members**
  - **Describes the types of methods**
  - **Keyword: `@interface`**
- Implementation section
  - **Contains the code for the methods**
  - **Keyword: `@implementation`**
- Program section
  - **Code to solve specific problems**
  - **Can be spread across several files**
  - **No keywords**

DePaul CDM 29

iPhone App Design

Factory method

- Creates the class

Interface format

```
@interface  className: ParentClass
    {
    Data variables;
    Member declarations;
    }
    Method declarations;
@end
```

DePaul CDM 30

iPhone App Design

Object / Instancing

- Instance methods
- Class methods
- Applied method changes state of the object

Key concepts

- Objects are unique representations from a class
- Each object contains some information (data)
  - **Typically private to that object**
- Methods provide the means of accessing and changing that data

Syntax

- [ClassOrInstance method];

Message / Receiver

- Sending a message - asks a class to perform a method.
- Recipient of a message is the receiver
  - **[receiver message]**

iPhone App Design

Method arguments

```
- (void) setNumber: (int) n;
```

What they represent:

- `-`
  - **Method type**
- `(void)`
  - **Return type**
- `setNumber`
  - **Method name**
- `:`
  - **Method takes arguments**
- `(int)`
  - **Argument type**
- `n;`
  - **Argument name**

iPhone App Design

```
myFraction = [[Fraction alloc] init];
```

- Inner expression evaluated first

```
NISAutoreleasePool *pool = [[NISAutoreleasePool alloc] init];
```

- Do you understand what this is?

```
[myFraction setNumerator: 1];
```

- Argument 1 is sent to myFraction.setNumerator()

iPhone App Design

int – reserved word

- Can't be used as a variable

Conventions

- Capitalize 1st letter for Classes
- Lower case for methods and data

Instance variables

@interface name : parent

```
{
int x;
int y;
}
```

- X & Y are instance variables

iPhone App Design

Read access

- `myVar = [someObject foo];`
- `myVar = someObject.foo;`
- Same as Java, C++, C#

Write access

- `someObject.foo = myVar;`
- `[someObject setFoo:myVar];`

iPhone App Design

- **Get**

```
    -(id) foo
     {
        Return foo;
     }
```

- Set

```
  -(void)
    setFoo:(id) aFoo
     {
 If(aFoo != foo) {
    [aFoo retain];
    [foo release];
     Foo = aFoo;
   }
}
```

- **`@property` combined with `@synthesize` tell the compiler to create access for the instance variable.**
- **Property needs to be set before the synthesize method is called!**

iPhone App Design

Random Notes
- Skip terminal use, only use xCode development

Syntax
- // comments
  - **same as C++**
- /* long multi-line … */
  - **Same as C**

String Danger
- `@"The quick brown fox jumps over the lazy dog"`
  - **Const NSString object**
- `"The quick brown fox jumps over the lazy dog"`
  - **Const C-Style String**
- Missing the @ sign is a big difference

NSLog(x)
- X is an argument
- Just like C/C++

DePaul CDM 37

iPhone App Design

Properties of avariable
- Example
  - `@property (nonatomic, retain) UILabel *statusText;`
- retain / assign
  - **Retain**
    Keep the instance variable from being flushed from memory
  - **Assign (default behavior)**
    Garbage collected
- nonatomic / atomic
  - **Multi-threaded behavior**
  - **`atomic` is default**
    Access is mutex-locked

DePaul CDM 38

iPhone App Design

[pool drain]
- Release memory

return 0
- Just like main()

NSLog
- Just like printf, sprintf, - formatted strings
- \n, %i – special characters
- Arguments work

DePaul CDM 39

iPhone App Design

void print()
- Instance method "-"

+int getTotalTrinkets()
- Class method "+"

Return types in parentheses
- -(int) foo()
  - **Return int**
- +(void) init()
  - **No return**

DePaul CDM 40

iPhone App Design

Data encapsulation
- What is this?
  - **Why do we like it?**
- Provides a layer of insulation

Accessors
- Get functions
  - `-(int) numerator;`
- Set functions
  - `-(void) setNumerator: (int) d;`

DePaul CDM 41

iPhone App Design

Allocation
- Method 1:
  - `Dog *fido;`
  - `fido = [Dog alloc];`
  - `fido = [fido init];`
- Method 2:
  - `Dog *Fido = [[Dog alloc] init];`
- Method 3:
  - `Dog *Fido = [Dog new];`
  - **Does the same as method 1 or 2**

DePaul CDM 42

iPhone App Design

Autorelease
- Allocate and release objects as needed.
- Supports shared objects between data structures
- Free shared object when the reference counts goes to zero.

Advantages:
- You do not have to think about memory as much
- Safer pattern, less likely to leak
- Some functionality in easier to use data structures

Disadvantages:
- Slower, release doesn't happen immediately
- Abstracted isolated malloc and free

iPhone App Design

Autorelease
- Is a delay release command
- Allocations are release in a pool on event boundaries

Allows pattern
- One object creates allocation
- Different object is responsible:
  - **retain and release object**

Very useful for newly created objects

iPhone App Design

Ownership of memory is critical
- Fragmentation is implied
- Life time of objects are shared
  - **No coordination**
    Loose coupling
- Free memory immediately or delayed
  - **release – immediate**
  - **autorelease – delayed or postponed**

iPhone App Design

Create objects in groups
- Use collectively
- Release collectively

Custom memory pool
- Create a collection of objects
- Object get return to the collection to be reused
- User manages this manager object
  - **One time initialization**
  - **Never release memory**

Avoid fragmentation

iPhone App Design

Autorelease pools are released
- When relinquishing ownership of the objects that have been added to the pool.

This frequently has the effect of disposing of temporary objects that have accumulated up to that point
- End of the event cycle, or during a loop when you create a large number of temporary objects. .

iPhone App Design

Garbage collected environment
- Release is a no-op
- NSAutoreleasePool provides a drain method behaves the same as calling release
  - **Drain does the garbage collection**
- Uses a mark and sweep process
  - **Stops all threads as the garbage collection happens**
  - **Timer or resource based trigger**

iPhone App Design

Many factory methods for convenience

- One general guideline

If you didn't allocate it or retain it

- **DON'T** release it

iPhone App Design

iPhone

- Uses alloc and dealloc for their memory manager
- Same as malloc / free in C

NSObjects

- Contain retain count
- If count > 0
  - **allocation is alive**
- If count == 0
  - **allocation is released**

iPhone App Design

Ref Count

- **`alloc, copy, new`**
  - **Create object**
  - **Retain count == 1**
- **`retain`**
  - **Increments retain count by 1**
- **`release`**
  - **Decrements retain count by 1**
- When retain count is 0
  - **`dealloc` gets called**

    Freeing the memory

iPhone App Design

If you overload **`dealloc`**

- Need to add the actual dealloc itself
  - **`[super dealloc]`**

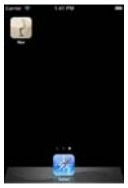
General rule

- Never call dealloc directly except for above case

Guideline

- If you create the memory
  - **You clean up the memory through release**

iPhone App Design



Assignment #3

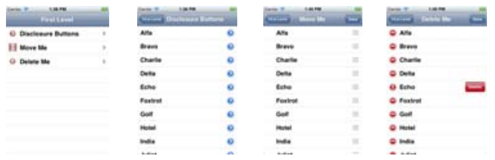
Change icon

- Add icon to your resources
  - **Name it icon.png**
- 57x57 texture
- PNG – Portable Network Graphics format
- Can use photo editor to resize and save as .png
- There are online image converters
  - **http://www.coolutils.com/online/image-converter/**

iPhone App Design

Assignment #3

- For the first assignment should be fairly easy
  - **This is a warm-up esp. for people who are not familiar with Obj-C**
- Take the Nav program that is discussed in chapter 9 (of the Orange book)
  - **Only implement three controllers (DisclosureButton, MoveMe, DeleteMe)**



iPhone App Design

Assignment #3

- Change all the data to reflect the Maritime alphabet
  - **Alfa, Bravo, Charlie, Delta, Echo, Foxtrot, Golf, Hotel, India, Juliet, Lima, Montreal, *etc.***
- For extra geek points
  - **If you change the data on one screen, it should update on the others**
  - **Hint: `viewDidLoad` and `viewDidUnload`**

DePaul CDM 55

Project

Tentative Schedule

- Week (today) – Discuss Project Proposal in Class
  - **DL students will have a wiki and we will set up a group Skype**
- Week 5 – Submit Project Proposal
- Week 7 – Present Project Progress
- Week 8 – Present Draft of Project
- Week 10 – Project Presentations
- Week 11 – Final Project Report due

DePaul CDM 56

Project

Course Project

- Should communicate an about mobile apps/devices that is of interest to your colleagues in class
- May be done independently, or in groups of two
- Ideas may come from past or current experience
- Should produce results that could be generalized and possibly published
  - **Develop a mobile app prototype that has principles of design that can be generalize**
  - **Discuss some aspect of mobile computing**
  - **Evaluate an application or a device (provide ideas for changes)**
  - **Something wild**

DePaul CDM 57

Project

Course Project

- The project can be a mix of both research and development
  - **This is a continuum**

Research Paper ⟷ Implementation Project

(7 pages) (2 or 3 pages + code)

DePaul CDM 58

Project

Project Proposal

- Explicitly declare what you plan to do for the project.
- Road map for completing the project
- Allows the instructor to provide feedback and suggestions.

Requirements

- The proposal should be about one to two pages in length.
- It should cover:
  - **Brief description of the topic.**
  - **Project participants and their roles.**
  - **Questions the project will address.**
  - **Activities that need to be performed.**
  - **Results that need to be collected.**
  - **How the results will answer the project's questions.**
  - **A timeline for accomplishing the project's goals.**

DePaul CDM 59

Project Proposal

Annotated Bibliography (Initial)

- 2-7 entries
- Each reference should have all bibliographic information (format: ACM, IEEE, etc.)
- ACM is preferred
  - **http://www.cs.ucy.ac.cy/~chryssis/specs/ACM-refguide.pdf**
  - **http://www.ieee.org/documents/ieeecitationref.pdf**
- Each reference should have a short (1-2 sentence) summary:

Robert Fabricant. 2005. Incorporating guidance and rewards into a handheld-device user experience. In *Proceedings of the 2005 conference on Designing for User eXperience* (DUX '05). AIGA: American Institute of Graphic Arts, New York, NY, USA, , Article 30.

Fabricant's paper discusses the design of a device to reduce the users stress by deep-breathing and bio-feedback. The team used "persuasive design" techniques to reinforce the user's behavior. Etc.

DePaul CDM 60