Objective C Primer

With Slides from Kevin Layer
Overview

- Objective-C is an object oriented language.
- follows ANSI C style coding with methods from Smalltalk
- There is no formal written standard
  - Relies mostly on libraries written by others
- Flexible almost everything is done at runtime.
  - Dynamic Binding
  - Dynamic Typing
  - Dynamic Linking
Inventors

• Brad Cox and Tom Love (1981)
• Cox thought something like Smalltalk would be very useful to application developers

| rectangles aPoint collisions |
rectangles := OrderedCollection
    with: (Rectangle left: 0 right: 10 top: 100 bottom: 200)
    with: (Rectangle left: 10 right: 10 top: 110 bottom: 210).
aPoint := Point x: 20 y: 20.
collisions := rectangles select: [:aRect | aRect containsPoint: aPoint].

• Cox modified a C compiler and by 1983 he had a working Object-oriented extension to C called OOPC.
Development

• 1986: Objective-C (via Stepstone)
• 1988: Steve Jobs obtain a licence for NeXT
• NeXTSTEP
  – Objective-C
  – Derived from BSD Unix
  – Make interface design easier
• 1995: full right from Stepstone
OPENSTEP API

• Developed in 1993 by NeXT and Sun
• An effort to make NeXTSTEP-like Objective-C implementation available to other platforms.
• In order to be OS independent
  – Removed dependency on Mach Kernel
  – Made low-level data into classes
• Paved the way for Mac OS X, GNUstep
Apple and Mac OS X

- NeXT is taken over by Apple in 1996 and put Steve Jobs and his Objective-C libraries to work
- Redesigned Mac OS to use objective-C similar to that of NeXTSTEP
- Developed a collection of libraries named “Cocoa” to aid GUI development
- Release Mac OS X (ten), which was radically different than OS 9, in March 2001
The Cocoa API

• Primarily the most frequently used frameworks nowadays.
• Developed by Apple from NeXTSTEP and OPENSTEP
• Has a set of predefined classes and types such as NSnumber, NSstring, Nsdate, etc.
• NS stands for NeXT-sun
• Includes a root class NSObject where words like alloc, retain, and release come from
• For iOS: Cocoa Touch
Dynamic Language

- Almost everything is done at runtime
- Uses dynamic typing, linking, and binding
- This allows for greater flexibility
- Minimizes RAM and CPU usage
To Import or Include?

`#import head.h`

- C/C++’s `#include` will insert `head.h` into the code even if it’s been added before.
- Obj-C’s `#import` checks if `head.h` has been imported beforehand.
Messages

• Almost every object manipulation is done by sending objects a message.

• Two words within a set of brackets, the object identifier and the message to send.

[Identifier message ]

• Because of dynamic binding, the message and receiver are joined at runtime.
Basic syntax structure

C++ syntax

```cpp
void function(int x, int y, char z);
Object.function(x, y, z);
```

Objective-C syntax

```objective-c
-(void) function:(int)x, (int)y, (char)z;
[Object function:x, y, z];
```
Keyword: id

- The word ‘id’ indicates an identifier for an object much like a pointer in C++
- This uses dynamic typing
- For example, if Pen is a class...

```c
extern id Pen;

id myPen;

myPen = [Pen new ];
```

(Cox, 59)
Memory Allocation

• Objects are created dynamically through the keyword, “alloc”
• Objects are dynamically deallocated using the words “release” and “autorelease”
• autorelease deallocates the object once it goes out of scope.
• New in iOS: ARC (automatic reference counting)
Prototyping functions

• When declaring or implementing functions for a class, they must begin with a + or -
• + indicates a “class method” that can only be used by the class itself.
• - indicates “instance methods” to be used by the client program (public functions).
#import <Foundation/Foundation.h>
@interface Person : NSObject {
    NSString * _name;
    int _age;
}
-(void)setAge:(int)number;
-(void)setName:(NSString *) next;
-(int) age;
-(NSString *) name;
@end
Class Definition (Implementation)

#import “person.h”
@implementation Person
-(void)setAge:(int) age
    {_age = age;}
-(void)setName:(NSString*) newname {
    _name = newname;
}
-(int) age
    {return _age;}
-(NSString *) name
    {return _name;}
@end
C++ VS. Objective-C

- Adds OOP, metaprogramming and generic programming to C
- Comes with a std library
- Has numerous uses
- Large and complex code for OOP

- Only adds OOP to C
- Has no standard library; is dependant on other libraries
- Mostly used for application building
- Simpler way of handling classes and objects
Useful Objects/functions

• **NSString**
  
  `@"a string"`

• **NSLog (like printf)**

  ```
  NSLog("%@ %d", name, age)
  ```

• **NSArray**

  ```
  NSArray * array = @["a", "b"];
  ```

• **NSMutableArray**

  ```
  ```
Objective-C 2.0

• In October 2007, Apple Inc. releases Objective-C 2.0 for Mac OS 10.5 (Leopard)
• Adds automatic garbage collection
• Instance Methods (public functions) are defined differently using `@property`
Automatic Reference Counting

• Memory Locations
  – Stack (frame)
  – Heap (all objective-C objects)

• Ownership
  – local variables
    \[ \text{NSString * str} = \text{Astring} \Rightarrow \text{str owns Astring} \]
  – instance variables
    \[ \text{Anobject._str} = \text{Astring} \Rightarrow \text{Anobject owns Astring} \]
ARC

• *Memory leak*: keep an object that has no owner.

• *Premature deallocation*: free up an object that is still owned by someone
Loss of Ownership

• Re-assignment
• Setting the pointer to nil
• The owner is destroyed
• Removing an object from a collection
  [items removeObject: item]
Strong vs. Weak Reference

- Strong Reference: own the object
- Weak Reference: does not own the object (but just point to it)
  - To avoid strong reference cycle
  - Define with `__weak`
    - `__weak NSString * str;`
References

• Cox, Brad. *Object Oriented Programming: an Evolutionary Approach*
• Sebesta, Robert. *Concepts of Programming Languages*
• Apple Inc. *Apple Developer Connection*
  http://developer.apple.com
• Stevenson, Scott. *Theocacao*
• Various Authors. *Wikipedia: the Free Encyclopedia*
  http://en.wikipedia.org