More OO, Start of Functional Programming

CS152

Chris Pollett

Oct. 27, 2008.

Outline

- Objective-C
- Functional Programming

Introduction

- Last day, we talked about two different ways that you could try to add object orientation to C:
  - Structs with functions pointers, bindings, and syntactic sugar
  - Functions with static members acting as fields, using a message parameter to the function to say what method to invoke. The static members are arrays, one for each field of the class, and an object_id is used to say which object among all instances of the class we are talking about.
- C++ takes the first approach. The second approach is more akin to Objective-C.

Objective-C

- Objective-C was created by Cox and Love in the mid-1980s.
- It was popularized by NextStep Corp, which was later bought by Apple.
- Objective-C is used to program Mac's, iPhones, etc. There is also a GNU version.
- It is an extension of C. So C programs are Objective-C programs.

Basic of an Objective-C Program

- Header Files use the extension .h
- Implementations use the extension .m
- A .h file typically might be used to define an interface of a class. This might have the format:

```
@interface classname : superclassname {
  int my_field;
   // instance variables can be any C type or  the generic id type.
}
+classMethod1; // + means class method; - means instance method
+(return_type)classMethod2;
+(return_type)classMethod3:(param1_type)parameter_varName;
-(return_type)instanceMethod1:(param1_type)param1_varName
     :(param2_type)param2_varName;
-(return_type)instanceMethod2WithParameter: (param1_type)param1_varName
     andOtherParameter:(param2_type)param2_varName;
@end
```

Implementation Files

- To implement the functions you use an implementation file and the syntax:

```
@implementation classname
+classMethod {
   // implementation
}
-instanceMethod {
   // implementation
}
@end
```

Invoking Methods

- A call obj.method(parameter); in C++ might look like [obj method:parameter]; in Objective-C.
- You can have multiple parameters:

  [obj method:parameter1 paramName2: parameter2]
- To instantiate a class you can send either the new message or instantiate in two steps:

  MyObject * o = [MyObject new];

  MyObject * o = [[MyObject alloc] init];
- You can override the init method of your class to make a new constructor.

Example Objective-C Program

```
// Interface File
#import <objc/Object.h>

@interface MyHello : Object {
   int myNumber;
}
-setNumber:(int)aNumber;
-sayHello;

@end
```

Example Program cont'd

```
// implementation file
#import <stdio.h>
#import "MyHello.h"
@implementation MyHello
-setNumber:(int)aNumber {     myNumber =
  aNumber;
}
-sayHello{
   printf("Hello! %d\n", myNumber);
}
-(id) init { //self is like this in Java
    self = [super init];
    if (self) { myNumber = 0; }
    return self;}
@end
```

Objective-C Example Last Part.

```
#import "MyHello.h"
int
main(void){
MyHello *hello = [MyHello new];
[hello setNumber:10];
   //set the number to echo
[hello sayHello];
return 0;
}
/* To compile in gcc could type:
gcc -x objective-c -Wno-import main.m MyHello.m
   -lobjc     */
```

Will do iPhone Demo.

Functional Programming

- We are now going to talk about functional programming languages.
- These programming languages view programs as functions.
- Given two sets X,Y a function f associates to each value in X a value in Y.
- This might be written f: X --> Y
- The mapping of a particular value might be written y = f(x).
- In order to make a language out of the notion of function. We need (1) to be able to start with a base set of functions and **define** new ones; (2) we need to be able to **apply** functions to values.