FIU | Computing & Information Sciences
FLORIDA INTERNATIONAL UNIVERSITY

Mobile Application Development

lecture6

Spring 2012 - COP 4655 U1
M/W 6:25pm – ECS 138
Steve Luis

Agenda

- More class principles

@class

- Compiler declaration used in your interface file .h
- Forward declaration
- Resolve circular references
- Compiling efficiency

Rule of thumb

- Only #import the super class or protocols in header files.
- #import all classes you send messages to in implementation.
- Forward declarations for everything else.

@class

Class Square refers to Circle and class Circle refers to Square.
If you use #import in .h file creates a Circular reference

Instead use:

```
#import <Foundation/Foundation.h>
@class Square;
@interface Circle : NSObject {
   [...]
}
[...]
-(NSString *) fitsInside:(Square *)shape;
@end
```

Then use #import “Square.h” after you import “Circle.h” file

Local Variables

```
@interface LVExample: NSObject
{
   int a;
   int b;
}

… skip to @implementation

- (void) offsetObjects
{
   int x = a;
   int y =b;

   a = x + y;
   b = x – y;
}
```

Local variables

- Declared within methods
- Must be initialized
- Are released after the method is executed.

Method Arguments

- Only a copy of the original value passed to object.
- Example below moodVal does not change after method is executed

```
[…]

- (void) smileBig: (int) wide
{
  wide = wide + 10;
}

[…]

[myFace smileBig: smileSize];
```

Static Keyword

- Use static to have Local variables retain value
- Initialized only once

```
- (int) ticketsDispensed
{
    static int ticketCount = 10;  // will not be reset
    ticketCount = ticketCount -1;
```

Self Keyword

- Refer to the object that is the receiver of the current message

- example

```
@interface LVExample: NSObject
{
  int a;
  int b;
}

… skip to @implementation
-     (void) normalize
{
     a = a * 100;
     b = b * 100;
}

- (void) offsetObjects
{
  int x = a;
  int y =b;

  [self normalize];
  a = x + y;
  b = x – y;
}
```

Find the right method

When you send a message to an object

1. The class of the object is checked for a match first.
2. The parent is checked next.
3. Continue checking parents until root class.
4. If not found generate an error.

Overriding Methods and Keyword Super

- Child class method with the same name of the Parent Class method overrides the inherited definition.
- The new method must have the same return type, and take the same number/type of arguments.
- You can send a message to super to execute an overridden method.

```
@interface ClassA: NSObject
{
   int a;
}
-  (void) initVar ;
-------------------------------------------------
@interface ClassB: ClassA


-  (void) initVar;
```

```
@implementation ClassA
-  (void) initVar
{
  a= a + 10;
}
------------------------------------------
@implementation ClassB
-  (void) initVar
{
     a = 10;
     [super initVar];
}
```

Abstract Classes

- Defined specifically for sub-classing only.
- Never expected to create an instance from it.
- Design pattern used in Foundation class.

```
@interface CommandUnit : NSObject
{
   int unitID;
}
-  (void) workFlowBlue ;
@end
```

Assignment

- Read Kochan chapters 7,8, and 9
- Due Monday, Feb. 6th.
- Don’t forget to complete Program #2 and Participation #2 by Weds. At 11pm.