

```

- (NSData *)requestWeatherDataWithCity:(NSString *)city
{
    NSLog(@"requestWeatherData:withCity:%@", city);
    NSError *error;
    NSMutableString *url = [[NSMutableString alloc] initWithString:@"http://api.openweathermap.org/data/2.5/weather?q="];
    [url appendString:city];
    NSURL *weatherService = [NSURL URLWithString:url];

```

Objective-C

```

    NSHTTPURLResponse *res = nil;
    NSData *returnData = [NSURLConnection sendSynchronousRequest:[NSURLRequest requestWithURL:weatherService] timeoutInterval:10.0 error:&error];
    if ((res != nil) && ([res statusCode] == 503))
        NSBeep();

    if (error) {
        NSLog(@"%@", error);
    }

```

```

    NSLog(@"xml: %@", [NSString stringWithFormat:@"%s", [returnData UTF8String]]);

```


What

- Superset to C for object-oriented programming
- Extremely dynamic
- Based on Smalltalk
- Primary language for development on OS X

Disclaimer

- Learning C is very important
- More complex than scripting languages
- Get a friend to help you!

Code Anatomy

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Declaration

```
@interface Tweet: NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Class name

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Superclass


```
@interface Tweet : NSObject {
```

```
    unsigned identifier;  
    NSString *sender;  
    NSString *text;
```

```
}
```

```
+ (id)tweetWithIdentifier:(unsigned)identifier;
```

```
- (unsigned)identifier;
```

```
- (NSString *)sender;
```

```
- (NSString *)text;
```

```
- (void)markAsFavorite:(BOOL)flag;
```

```
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;
```

```
@end
```

Instance variables

```
@interface Tweet : NSObject {  
    unsigned identifier;  
    NSString *sender;  
    NSString *text;  
}
```

```
+ (id)tweetWithIdentifier:(unsigned)identifier;
```

```
- (unsigned)identifier;  
- (NSString *)sender;  
- (NSString *)text;  
  
- (void)markAsFavorite:(BOOL)flag;  
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;
```

```
@end
```

Method declaration

```
@interface Tweet : NSObject {  
    unsigned identifier;  
    NSString *sender;  
    NSString *text;  
}
```

```
⊕(id)tweetWithIdentifier:(unsigned)identifier;
```

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;
- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

```
@end
```

Method type: class method

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Return type: the any type

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Method name

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Parameter type and name


```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Method declaration

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Method type: instance method

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Return type

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message fromSource:(NSString *)source;

@end
```

Method name and parameters...

```
@interface Tweet : NSObject {
    unsigned identifier;
    NSString *sender;
    NSString *text;
}

+ (id)tweetWithIdentifier:(unsigned)identifier;

- (unsigned)identifier;
- (NSString *)sender;
- (NSString *)text;

- (void)markAsFavorite:(BOOL)flag;
- (void)replyWithText:(NSString *)message (fromSource:(NSString *)source);

@end
```

...and more name and parameters

Sending messages


```
[someTweet replyWithText:@"Hello" fromSource:@"myClient"];
```

```
[[someTweet] replyWithText:@"Hello" fromSource:@"myClient"];
```

Object

```
[someTweet(replyWithText:@"Hello" fromSource:@"myClient"]];
```

First piece of method name

```
[someTweet replyWithText:@"Hello" fromSource:@"myClient"];
```

First parameter

```
[someTweet replyWithText:@"Hello" (fromSource:@"myClient"];
```

Second part of method name

```
[someTweet replyWithText:@"Hello" fromSource:@"myClient"];
```

Second parameter

- [NSArray indexOfObjectInRange:]
- [NSObject setValue:forKey:]

Methods = Messages

**Method
Invocations = Message
Sends**

More later

Implementing your class

@implementation Tweet

```
- (void)markAsFavorite:(BOOL)flag {  
    if (flag) {  
        [magicTwitterService markTweetWithIdentifierAsFavorite:identifier];  
    } else {  
        [magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];  
    }  
}  
...  
@end
```


`@implementation Tweet`

```
- (void)markAsFavorite:(BOOL)flag {  
    if (flag) {  
        [magicTwitterService markTweetWithIdentifierAsFavorite:identifier];  
    } else {  
        [magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];  
    }  
}  
...  
@end
```

Class implementation

```
@implementation Tweet
- (void)markAsFavorite:(BOOL)flag {
    if (flag) {
        [magicTwitterService markTweetWithIdentifierAsFavorite:identifier];
    } else {
        [magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];
    }
}
...
@end
```

Class name

```
@implementation Tweet
```

```
- (void)markAsFavorite:(BOOL)flag {
```

```
    if (flag) {
```

```
        [magicTwitterService markTweetWithIdentifierAsFavorite:identifier];
```

```
    } else {
```

```
        [magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];
```

```
    }
```

```
}
```

```
...
```

```
@end
```

Method signature

@implementation Tweet

- (void)markAsFavorite:(BOOL)flag {

if (flag) {

[magicTwitterService markTweetWithIdentifierAsFavorite:identifier];

} else {

[magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];

}

}

...

@end

Method body

```
@implementation Tweet
- (void)markAsFavorite:(BOOL)flag {
    if (flag) {
        [magicTwitterService markTweetWithIdentifierAsFavorite:identifier];
    } else {
        [magicTwitterService unmarkTweetWithIdentifierAsFavorite:identifier];
    }
}
...
@end
```

Remember the instance variables

```
@interface MagicTweet : Tweet {  
}  
@end
```

```
@implementation MagicTweet  
- (void)markAsFavorite:(BOOL)flag {  
    if (random() % 2 == 1) {  
        [self replyWithMessage:@"I LOVED IT!" fromSource:@"magic land"];  
    }  
    [super markAsFavorite:flag];  
}  
@end
```

```
@interface MagicTweet : Tweet {  
}  
@end  
  
@implementation MagicTweet  
- (void)markAsFavorite:(BOOL)flag {  
    if (random() % 2 == 1) {  
        [self replyWithMessage:@"I LOVED IT!" fromSource:@"magic land"];  
    }  
    [super markAsFavorite:flag];  
}  
@end
```

Invoke superclass implementation

Making new objects


```
NSString *string = [[NSString alloc] init];
```

```
NSString *string = [[NSString alloc] init];
```

Variable declaration

```
NSString *string = [[NSString alloc] init];
```

Object allocation

```
NSString *string = ([[NSString alloc] init]);
```

Object initialization

```
NSString *string = ([[NSString alloc] initWithString:@"New String"]);
```

Object initialization

Foundation

Base class library for Objective-C

Object
Strings
Arrays
Dictionaries
Memory management
Threading
Timers
File management
Network access

NSObject
NSString
NSArray
NSDictionary
NSThread
NSTimer
NSFileHandle
NSStream

Memory management

Reference counting

- [NSObject retain]
- [NSObject release]

Autorelease pools

`NSAutoreleasePool`

Simple rules

-init... and -get... methods return retained objects
Everything else is autoreleased

Deallocation

When your retain count hits 0, your
-dealloc
method is called

Tons to explore, but not this week

Housekeeping

Interfaces go in .h files
Implementations go in .m files

Use `#import` directive to
bring interface into your file

Nicer than C's `#include`

**Declare only public methods
in interface; keep private
methods in implementation**

A Simple Program

```
#import <Foundation/Foundation.h>

@interface Welcomer : NSObject {
    NSString *name;
}

- (id)initWithName:(NSString *)aName;
- (void)welcome;
@end
```

Welcomer.h

```
#import "Welcomer.h"

@implementation Welcomer
- (id)initWithName:(NSString *)aName {
    self = [super init];
    if(self) {
        name = [aName retain];
    }
    return self;
}

- (void)dealloc {
    [name release];
    [super dealloc];
}

- (void)welcome {
    NSLog(@"Hello, %@!", name);
}
@end
```

Welcomer.m

```
#import "Welcomer.h"
#import <Foundation/Foundation.h>

int main(int argc, char **argv) {
    Welcomer *w = [[Welcomer alloc] initWithName:@"Steve Jobs"];
    [w welcome];
    [w release];

    return 0;
}
```

main.m

```
gcc -lobjc -framework Foundation -o Welcomer Welcomer.m main.m
```



```
# ./welcomer
```

```
Welcomer[18429] 4/1/08 12:00:00 AM: Hello, Steve Jobs!
```

```
#
```

More constructs!

Categories

Add methods to existing classes

```
#define TWOOSH_LENGTH (140)

@interface Tweet (TwooshSupport)
- (BOOL)isTwoosh;
@end

@implementation Tweet (TwooshSupport)
- (BOOL)isTwoosh {
    return [text length] == TWOOSH_LENGTH;
}
@end
```

```
#define TWOOSH_LENGTH (140)

@interface Tweet (TwooshSupport)
- (BOOL)isTwoosh;
@end

@implementation Tweet (TwooshSupport)
- (BOOL)isTwoosh {
    return [text length] == TWOOSH_LENGTH;
}
@end
```

Category name

Protocols

**Define interfaces for other classes
to implement**


```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end
```

```
@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end
```

```
@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end

@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end

@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

Protocol declaration

```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end

@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end

@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

Protocol name

```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end
```

```
@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end
```

```
@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

Protocol methods

```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end

@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end

@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

Protocol adherence

```
@protocol JSONEncodable
- (NSString *)JSONContents;
@end

@interface Record <JSONEncodable> {
    ...
}
...
- (NSString *)JSONContents;
@end

@interface JSONTransport
- (void)sendObjectToRemoteHost:(id <JSONEncodable>)object;
@end
```

Protocol adherence

Properties

Quick syntax for declaring setters and getters


```
@interface Tweet : NSObject {
```

```
    ...
```

```
}
```

```
@property (readonly) NSString *text;
```

```
@end
```

```
@implementation Tweet
```

```
@synthesize text;
```

```
@end
```

```
@interface Tweet : NSObject {  
    ...  
}
```

```
@property (readonly) NSString *text;
```

```
@end
```

```
@implementation Tweet  
@synthesize text;  
@end
```

Property declaration

```
@interface Tweet : NSObject {  
    ...  
}  
  
@property (readonly) NSString *text;  
  
@end  
  
@implementation Tweet  
@synthesize text;  
@end
```

Parameters

```
@interface Tweet : NSObject {  
    ...  
}  
  
@property (readonly) NSString *text;  
  
@end  
  
@implementation Tweet  
@synthesize text;  
@end
```

Business as usual

```
@interface Tweet : NSObject {  
    ...  
}  
  
@property (readonly) NSString *text;  
  
@end  
  
@implementation Tweet  
@synthesize text;  
@end
```

Tell compiler to write your method for you

[tweet text]

vs

tweet.text

Language patterns

Delegation

The friendlier alternative to subclassing

Class clusters

Factory methods that return
different private implementations

Target-action

Send my object a message of my choosing
when something interesting happens

Model

Model View

**Model
View
Controller**

Model
Controller
View

One more thing...

Selectors

Names for methods

Similar to symbols in Smalltalk

Declare with @selector



© 2008 Benjamin Stiglitz
ben@tanjero.com