That strange beast of Objective-C

Yes, there's more than [square brackets] !

Roberto Gamboni • Antonio Malara

TOMTOM Developers Day

Lunch Menu

- History
- Object Messaging
- Dynamism
- Forwarding
- Categories

History

70s	**C** – AT&T Bell Labs • 1972 **Smalltalk** – Xerox Parc • 1972
80s	**C++** – AT&T Bell Labs • 1983 **Objective-C** – Stepstone Inc • 1986
90s	**NeXTStep** – NeXT Inc • 1989 **OpenStep** – NeXT Inc • 1994
00s	**Cocoa** – Apple Inc • 2001 **Objective-C 2.0** – Apple Inc • 2006

Objective-C

- Thin layer over C: it supports only object messaging
- Dynamic
- Single inheritance plus multiple interfaces
- There is no standard library!
- Leverage C, not call it Evil
- Expressive

Foundation

- Most used Objective-C library
- Base classes and language support
- Data structures
- Runloops and I/O
- Not required!!

Glimpse of syntax

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

@interface ExampleClass
{
  Class isa;
  int   counter;
}

+ initialize;
+ alloc;

- free;
- count;
@end
```

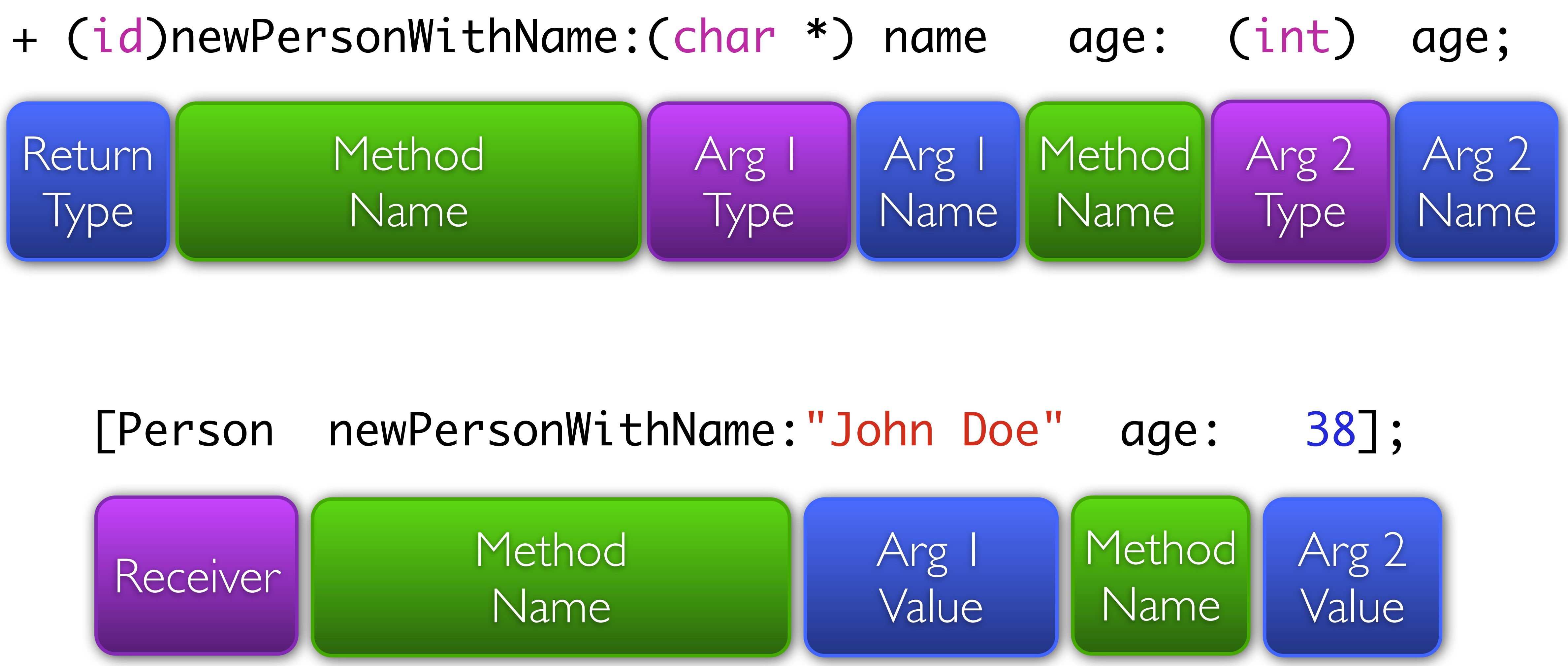
```
@implementation ExampleClass

+ initialize;
{ return self; }

+ alloc;
{ class_createInstance(self, 0); }

- free;
{ object_dispose(self); }

- count;
{ printf("Hello world %d\n",
           counter++); }
@end
```

Reality is easier

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

@interface ExampleClass : NSObject {
  int counter;
}

- count;
@end

@implementation ExampleClass

- count {
  printf("Hello world %d\n", counter++);
}
@end
```

TOMTOM Developers Day

Object Messaging



```
+ (id)newPersonWithName:(char *) name   age:  (int)  age;
```

```
[Person  newPersonWithName:"John Doe"  age:   38];
```

Object Messaging

Obj-C

```
// Declaration
+ (id)newPersonWithName:(char *)name age:(int)age;

// Usage
[Person newPersonWithName:"John Doe" age:38];
```

C++

```
// Assuming we could use “:” in method names
// Declaration
id newPersonWithName:age:(char * name, int age);

// Usage
Person::newPersonWithName:age:("John Doe", 38);
// or
Person->newPersonWithName:age:("John Doe", 38);
```

Object Messaging

- Classes are objects too: meta-class managed by runtime
- No constructor mechanism.
- Sending an object to the null pointer is safe
- Converted to a call to `objc_msgSend()`

How much dynamism?

- Dynamic Method Resolution

  Object dynamically typed *at runtime*

  You can modify a class *at runtime*

Not my responsability

- Message Forwarding

  Transparently forward the message call to another object

  Simplify implementation of design patterns

Categories

- Extend Classes without resorting subclassing
- Adapt existing Classes to requirements
- Code Maintainability/Management
- Every instance of the class is modified

Categories (2)

```
@interface NSMutableArray (UtilityAdditions)
- (void) shuffle;
- (void) reverse;
@end

@interface NSMutableArray (StackAdditions)
- (void) push: (id) anObject;
- (id) pop;
@end

@interface NSMutableArray (QueueAdditions)
- (id) pull;
@end
```

Coding time

Unleashing the beast

Real world usage

TOMTOM Developers Day

DEV DAY 2010

DRIVE THE BUSINESS

Dinner Menu

- KVO
- Blocks
- GCD
- Zombies

KVO

- Naming conventions matters
- Observer pattern built-in
- Classes are modified at runtime

Blocks

- Inline code passed to other functions
- Callback much easier to use
- Compact code
- Control structures (like Ruby and more) and concurrency primitive (like Erlang)

GCD

- Underlying mechanism that makes multithreading easier
- very fast, efficient and light on the system

Zombies

- Memory region is never marked as free
- Message to freed object are controlled
- Code predictable and debuggable

Coding time again

Open Source

libs	license	open source
objc-runtime	APSL	✓
GC libauto	Apache	✓
libdispatch	Apache	✓
Core Foundation	APSL	✓

More at: http://www.opensource.apple.com/

Thank You!
TOMTOM Developers Day
DEV DAY 2010
DRIVE THE BUSINESS