

Tagged Beyond Belief

Robin Kunde
me@robinkunde.com
@robinkunde

About Me

- Development Manager @ bizjournals.com
- Oversee Backend/Cloud/SaaS
- Developed and maintain The Business Journals app

A Mystery

```
[[NSString alloc] initWithUTF8String:"hellohello"];  
[[NSString alloc] initWithUTF8String:"hellohellq"];
```

- First statement more than twice as fast as second one
- Unfortunately, Foundation is not open source
- Reason is tagged pointers

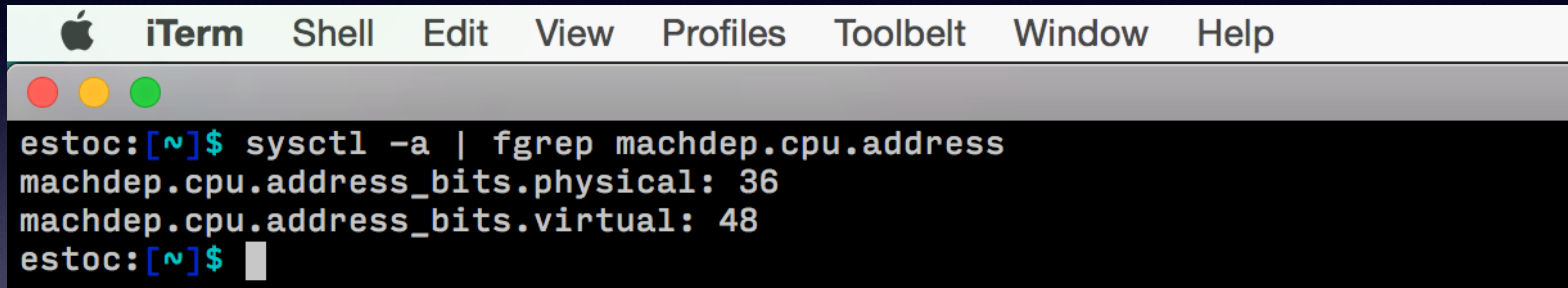
What's a pointer?

- Integer pointing to memory address on heap
- Simply means nth byte from start
- 0 has special meaning, not pointing to anything
- Depending on context, represented by nil, null, or nullptr
- Size depends on memory architecture of target platform
- 64-bit wide on modern systems, incl. OS X and iOS

What's a tagged pointer?

- Not all pointer bits are actually ever used
- Modern x64 CPUs don't have enough address lines
- `sysctl -a | fgrep machdep.cpu.address`

What's a tagged pointer?



```

  Apple  iTerm  Shell  Edit  View  Profiles  Toolbelt  Window  Help
  ● ● ●
estoc:[~]$ sysctl -a | fgrep machdep.cpu.address
machdep.cpu.address_bits.physical: 36
machdep.cpu.address_bits.virtual: 48
estoc:[~]$
```

The image shows a screenshot of an iTerm terminal window. The title bar at the top includes the Apple logo, the name 'iTerm', and several menu items: 'Shell', 'Edit', 'View', 'Profiles', 'Toolbelt', 'Window', and 'Help'. Below the title bar is a grey bar with three colored window control buttons (red, yellow, green). The terminal area has a black background with white text. It shows a command prompt 'estoc:[~]\$' followed by the command 'sysctl -a | fgrep machdep.cpu.address'. The output consists of two lines: 'machdep.cpu.address_bits.physical: 36' and 'machdep.cpu.address_bits.virtual: 48'. The prompt 'estoc:[~]\$' is shown again at the bottom with a white cursor block.

What's a tagged pointer?

- Not all pointer bits are actually ever used
- Modern x64 CPUs don't have enough address lines
- `sysctl -a | fgrep machdep.cpu.address`
- Allocated memory required to be aligned
- Memory address must be dividable by certain power of 2
- In 8-byte aligned pointer, lowest 3 bits are always 0
- Tagging is the process of storing metadata in unused bits

WHY?!?

- Short answer: Speed and ease of development
- Long answer: ARC

Transitioning

- With OS X 10.5, Apple introduced Objective-C 2.0
- Added 64-bit support and garbage collection for OS X
- Some low level features deprecated in 32-bit mode, removed in 64-bit

Changes to Objective-C

If you are writing low-level code that targets the Objective-C runtime directly, you can no longer access an object's `isa` pointer directly. Instead, you need to use the runtime functions to access that information.

The isa pointer

⌵ ▶ ⏏ ⏏ ⏏ ⏏ ⏏ | ⏏ | test > Thread 1 > 0 main

▶ **A** **argv** = (const char **) 0x7fff5fbff880

A **argc** = (int) 1

▼ **L** **test** = (__NSCFConstantString *) @"spacejam"

▼ **__NSCFString**

▼ **NSMutableString**

▼ **NSString**

▼ **NSObject**

isa = (Class) __NSCFConstantString

The isa pointer

- Points to location of Class object in memory
- Only property of NSObject
- Retain count is not actually stored in NSObject
- ARC emits more retain/release messages than a person would
- Changing NSObject would come at a huge compatibility cost

The Magic isa pointer

LSB

(Example only. Actual implementation is fluid.)

1	Is this a tagged pointer?
1	Object has/had an associated reference?
1	Object has C++/ARC destructor?
30	Upper 30 bits of class pointer
9	Magic Number (011010010)
1	Object is/was weakly referenced?
1	Object is deallocating?
1	Object has external retain count?
19	Inline retain count

MSB

General		Capabilities	Info	Build Settings	Build Phases	Build Rules
Basic	All	Combined	Levels	<div>Q</div>		
▼ Architectures						
Setting		Bizjournals AppStore				
Additional SDKs						
Architectures		Standard architectures (armv7, arm64) - \$(ARCHS_STANDARD) ⬆				
Base SDK		Latest iOS (iOS 8.4) ⬆				
▼ Build Active Architecture Only		<Multiple values> ⬆				
AdHoc		No ⬆				
AppStore		No ⬆				
Debug		Yes ⬆				
► Supported Platforms		iOS ⬆				
Valid Architectures		arm64 armv7 armv7s				

Tagging, with a vengeance

- Allocating the memory for objects is expensive
- At best, call to `malloc()` returns spare memory
- At worst, syscall to kernel to provide more memory
- We can replace entire objects with tagged pointers

Look at all this room for data!

LSB

1	Is this a tagged pointer?
3	What kind of object is this?
60	Payload

MSB

Look at all this room for data!

0000 0000 0000 1111 0111

^ ^ ^

| | tag bit

| |

| tagged pointer class index (3)

|

payload (15)

Magic Objects

0	NSAtom
1	Unused
2	NSString
3	NSNumber
4	NSIndexPath
5	NSManagedObjectID
6	NSDate
7	Unused

Yeah, but...

- Why does this actually work?
- How can you call methods on non-objects?
- Object Oriented Programming is just an abstraction
- You group related data and functions into classes
- Compiler transforms abstractions into machine code

How do methods even?

```
class Number
{
    var value = 0

    func add(numberToAdd: Int) -> Number
    {
        value += numberToAdd

        return self
    }
}
```


How do methods even?

```
struct Number
{
    Class* isa;
    int value;
}
```

```
Number* function add(Number* self, int numberToAdd)
{
    self->value += numberToAdd;

    return self;
}
```


How do methods even?

```
Number* function add(Number* self, int numberToAdd)
{
    if (self & 1 == 1) {
        // extract value from self
        // add numberToAdd to decoded value
        // encode new value
        // return as new tagged pointer
    }

    self->value += numberToAdd;

    return self;
}
```


Make ObjC Swifter (Swiftier?)

- Pointers are value types
- Copy on assignment, pass by value
- Tagged pointer objects are also value types
- Like Swift structs, can call methods on them
- No reference counting necessary
- However, still use message passing

Demo Time!

- Let's build an NSTaggedPointerString object by hand

NSTaggedPointerString

LSB

1	Is this a tagged pointer?
3	What kind of object is this? (5)
4	Length of string (1 to 11 characters)
56	String data, variable encoding (8bit for 1-7 character, 6b 8-9, 5b 10-11)

MSB

NSTaggedPointerString

- 56 bits allows 7 8-bit ASCII characters
- Beyond that, squeeze in more characters by changing character set
- 6-bit character set only encodes 64 most commonly used characters
- 5-bit character set encodes 32 most commonly used characters