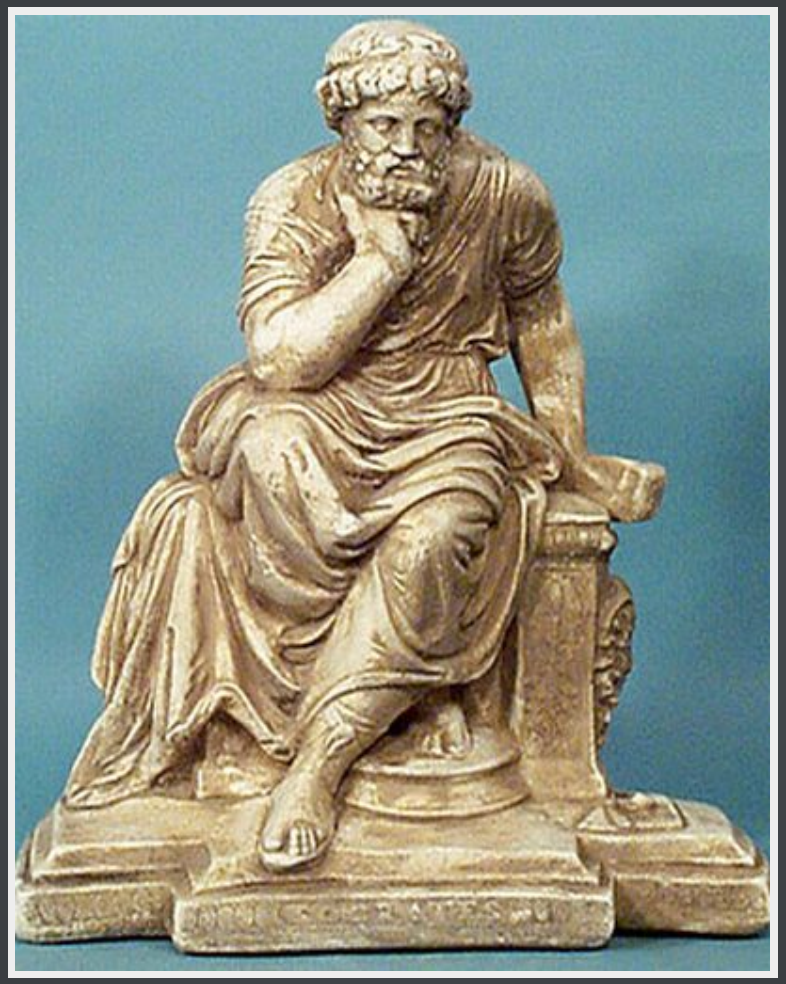
LOW LEVEL THINKING IN HIGH LEVEL PROGRAMMING

Błażej Marcinkiewicz

Łódź wiOSłuje #6

WHAT WILL THIS TALK BE ABOUT?



AGENDA

1. High level assembly - C
2. What can go wrong in C++?
3. What's so different in shading languages?
4. How can it run so fast? - ObjC runtime

BRAIN TEASER FOR A GOOD START

```c
#include <stdio.h>
void foo(void)
{
        int a;
        printf("%d\n", a);
}
void bar (void)
{
        int a = 42;
}

int main(void)
{
        bar();
        foo();
}
```

cc foo.c && ./a.out
42

HAVE YOU EVER WONDERED...

```c
#include <stdio.h>

void foo(void)
{
        int a;
        printf("%d\n", a);
}

int main(void)
{
        foo();
}
```

```c
#include <stdio.h>

void foo(void)
{
        static int a;
        printf("%d\n", a);
}

int main(void)
{
        foo();
}
```

WHY IS IT SO BIG?

```c
#include <stdio.h>

struct X
{
        int a;
        char b;
        int c;
};

int main(void)
{
        printf("%zu\n", sizeof(int));
        printf("%zu\n", sizeof(char));
        printf("%zu\n", sizeof(struct X));
}
```

THE SPIRIT OF C

There are many facets of the spirit of C, but the essence is a community sentiment of the underlying principles upon which the C language is based. Some of the facets of the spirit of C can be summarized in phrases like (Introduction to C Rationale):

- Trust the programmer.
- Don't prevent the programmer from doing what needs to be done.
- Keep the language small and simple.
- Provide only one way to do an operation.
- Make it fast, even if it is not guaranteed to be portable.

WHAT ABOUT C++?

```
struct A
{
        A() { printf("A()\n"); };
        A(int a) { printf("A(int a)\n"); };
        ~A() { printf("~A()\n"); };
};
struct B
{
        B(int b) { a = b; };
        B(long l) : a(l) { };
        A a;
};
int main()
{
        printf("1\n");
        { B b(int(2)); }
        printf("2\n");
        { B b(long(2)); }
        return 0;
}
```

EXCEPTIONS AND RTTI

- Exceptions mechanism is intended to handle error situations.
- RTTI - run time type information. Allows dynamic check if object is instance of a specific class.

VIRTUAL WORLD

```
#include <stdio.h>
struct X
{
        int a;
        char b;
        int c;

        virtual void setA(int v) { a = v; }
        int getA() { return a; }
};

int main(void)
{
        printf("%zu\n", sizeof(struct X));
        return 0;
}
```

SCRAPING THE SILICON

- GPU is designed to execute many simple operations at once.
- GPUs have totally different assembly, though shading languages look similar to C.
- Heaviest operation - global memory read.
- GPUs are optimized for image processing operations.

MADNESS

```
float main(float x : TEXCOORD) : SV_Target
{
        return (x + 1.0f) * 0.5f;
}
```

```
float main(float x : TEXCOORD) : SV_Target
{
        return x * 0.5f + 0.5f;
}
```

FUN FACTS

```
float main(float2 x : TEXCOORD) : SV_Target
{
        return abs(a.x) * abs(a.y);
}
```

```
float main(float2 x : TEXCOORD) : SV_Target
{
        return abs(a.x * a.y);
}
```

A BIT OF SORCERY

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y  = number;
    i  = * ( long * ) &y;       // evil floating point bit level hacking
    i  = 0x5f3759df - ( i >> 1 );               // what the fuck?
    y  = * ( float * ) &i;
    y  = y * ( threehalfs - ( x2 * y * y ) );   // 1st iteration

    return y;
}
```

WHY COMPILER DIDN'T OPTIMIZE IT?

- Results might not be the same.
- May introduce INF or NaN.
- Compiler cannot break the rules, you've written the code so you probably knew what you wanted to do.

WHAT ABOUT MOBILE?

- Objective C 3rd most popular programming language (TIOBE).
- Superset of C language.
- Smalltalk style messaging.

LONG TIME AGO...

- Capture image from front-facing camera and map eyes position to looking direction in a 3D scene.
- Do it fast <=> 60fps.
- Do it on mobile.

SO CLOSE...

```
@interface ETKernel {
        float *_kernelValues;
}

- (float)getKernelValueAtPosition:(CGPoint)position;
@end
```

OBJC RUNTIME

Every message:

```
[self doSomethingTo:var1];
```

Is converted into C function call:

```
objc_msgSend(self, @selector(doSomethingTo:), var1);
```

WE NEED TO GO DEEPER

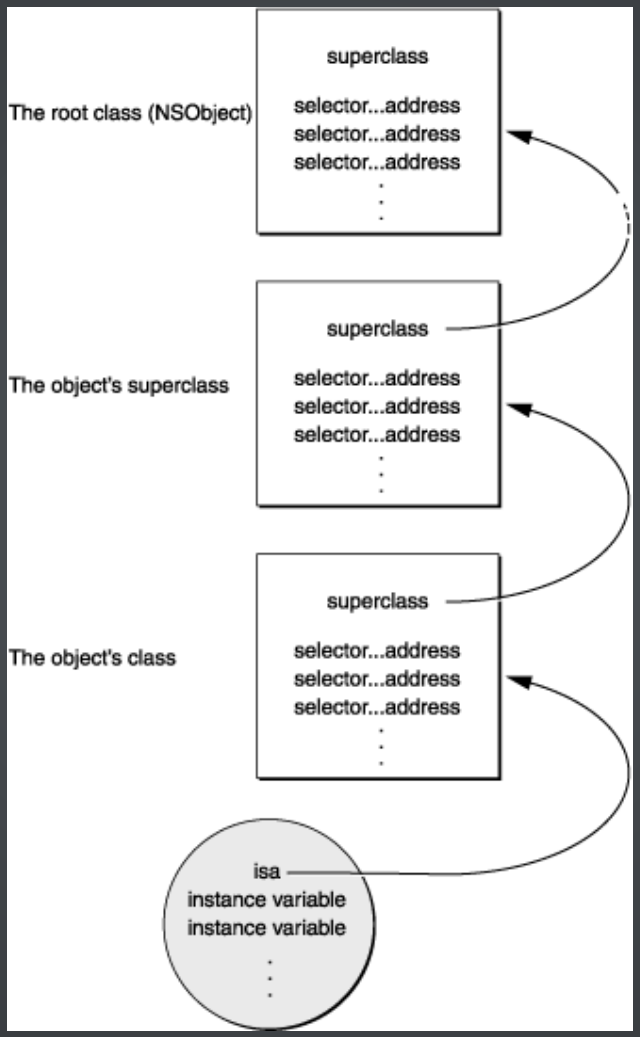
```
id objc_msgSend(id receiver, SEL name, arguments...);
```

- Check if receiver is not nil.
- Check if receiver responds to selector.
- Handle KVO notifications.
- ...

HOW OBJC OBJECTS ARE BUILT

- Associative containers
- Every object contains pointer to superclass
- Retain count.
- Dispatch table

DISPATCH TABLE



WHY SHOULD I CARE?

- Performance impact on every method call (also properties).
- UIScrollView scrolling speed.

WHY MY SCROLL VIEW ISN'T SCROLLING SMOOTHLY?

- Deep view hierarchy.
- Too much alpha blending.
- Too many allocations.

WHEN PROFILER CHEATS...

- Default output of Instruments is hard to read
- Hide missing symbols, Hide system libraries filter
- What are we missing?


AT&T
1:58 PM
Qwilt
My Qwilt
70 albums • 2,091 photos
Today
3:45pm
Weekends are great
14 photos
Belltown
11
5
3 days ago
Swimming in the river
14 photos
AT&T
1:58 PM
My Qwilt
Mar 29, 2013
Weekends are great
24 photos
Belltown

IMAGE LOADING PITFALLS

- [UIImage imageNamed:] vs [UIImage imageWithContentsOfFile:]
- Multilevel image cache

DECOMPRESSION SICKNESS

- Use CoreGraphics kCGImageSourceShouldCache flag
- Draw image in background to force decompression
- Watch out for memory

ANIMATIONS

- Problem - load&run ~150 frames of animation
- Challenge - keep it smooth
- Solution - show first frame, and run animation after scrolling is finished ;]

CONCLUSION

- Remove all properties from code?
- Should everyone know that?
- Don't turn off thinking while writing code.

Q&A

CONTACT

Błażej Marcinkiewicz
blazej.marcinkiewicz@polidea.com
@blazejmar