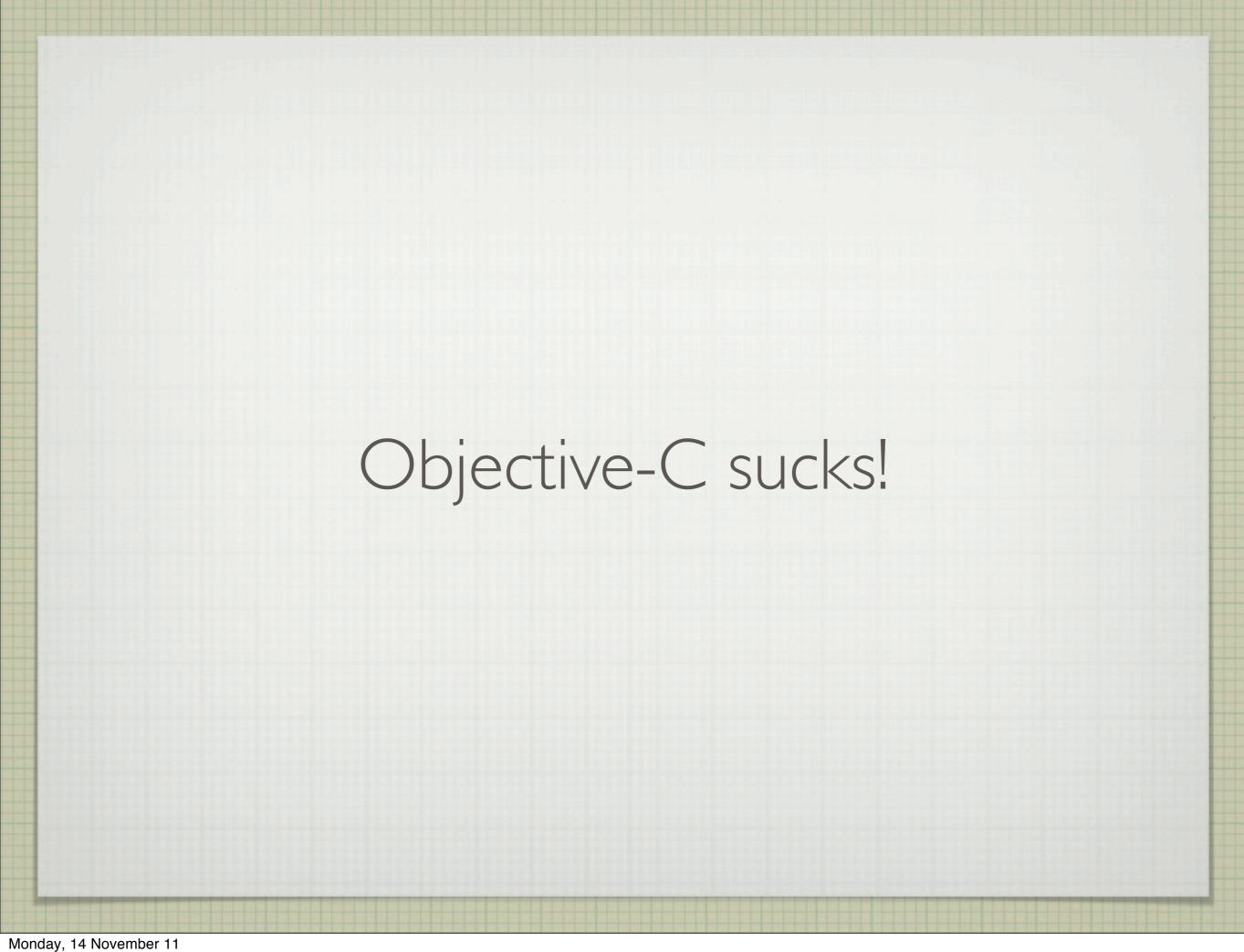
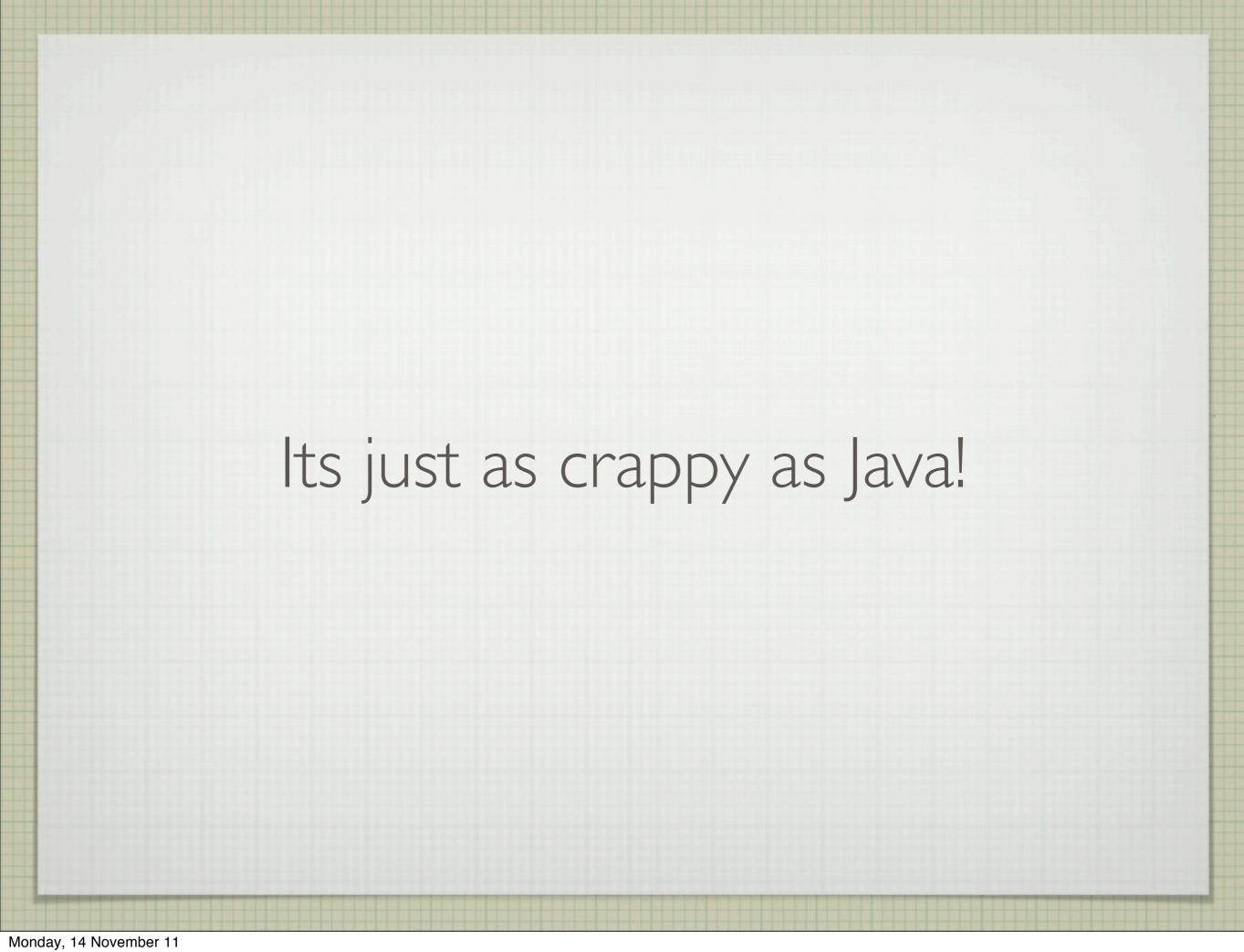
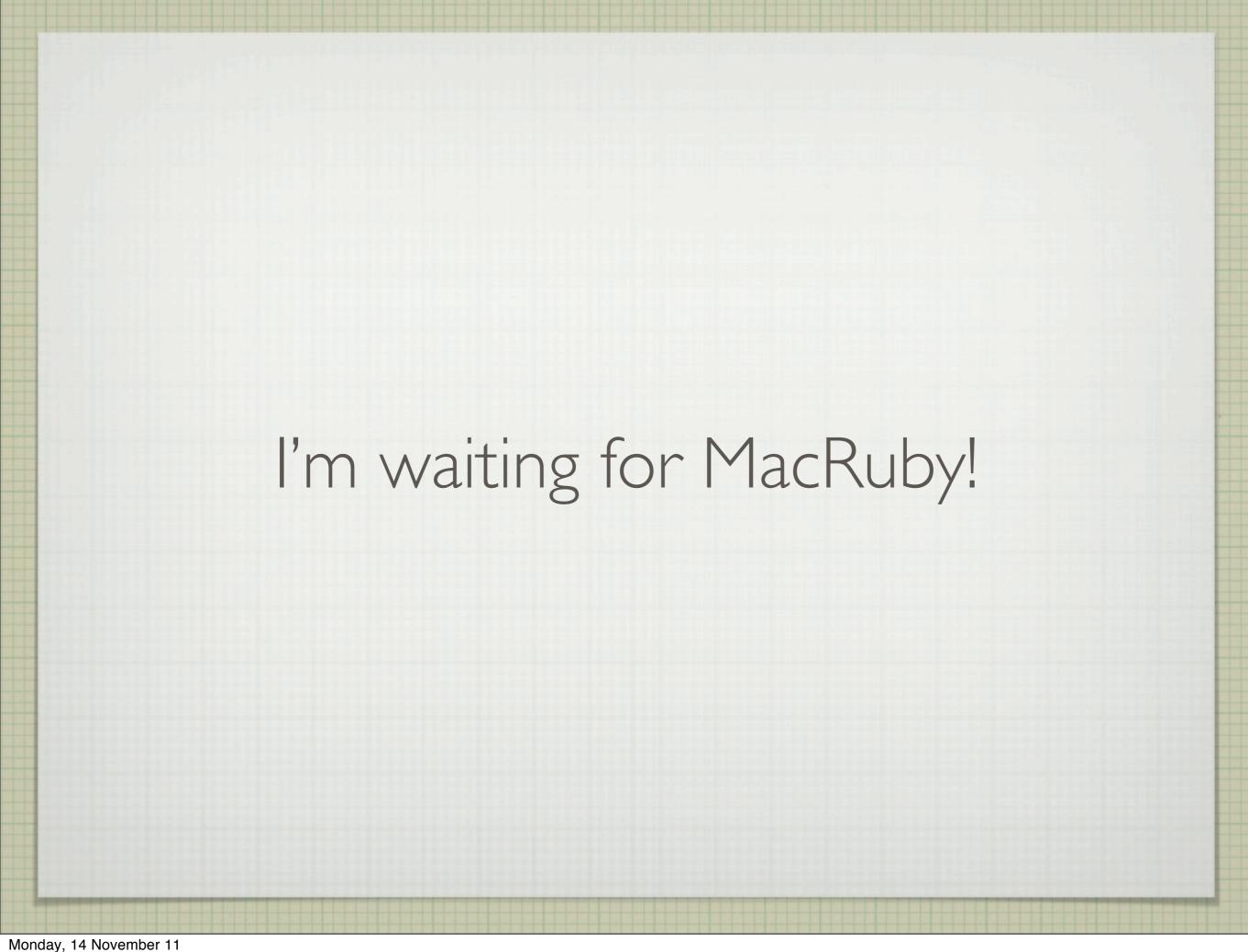
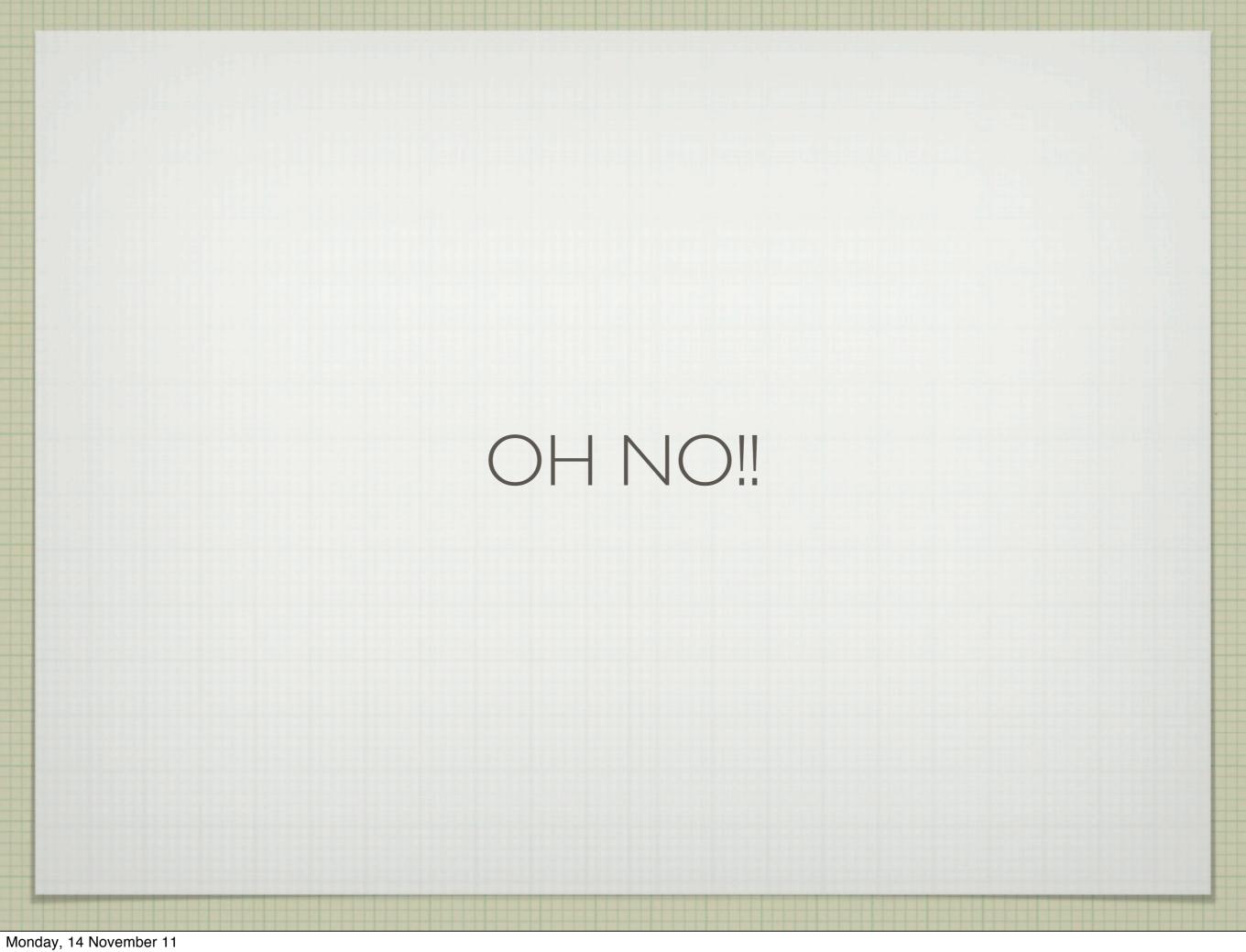
MY ADVENTURES IN OBJECTIVE-C

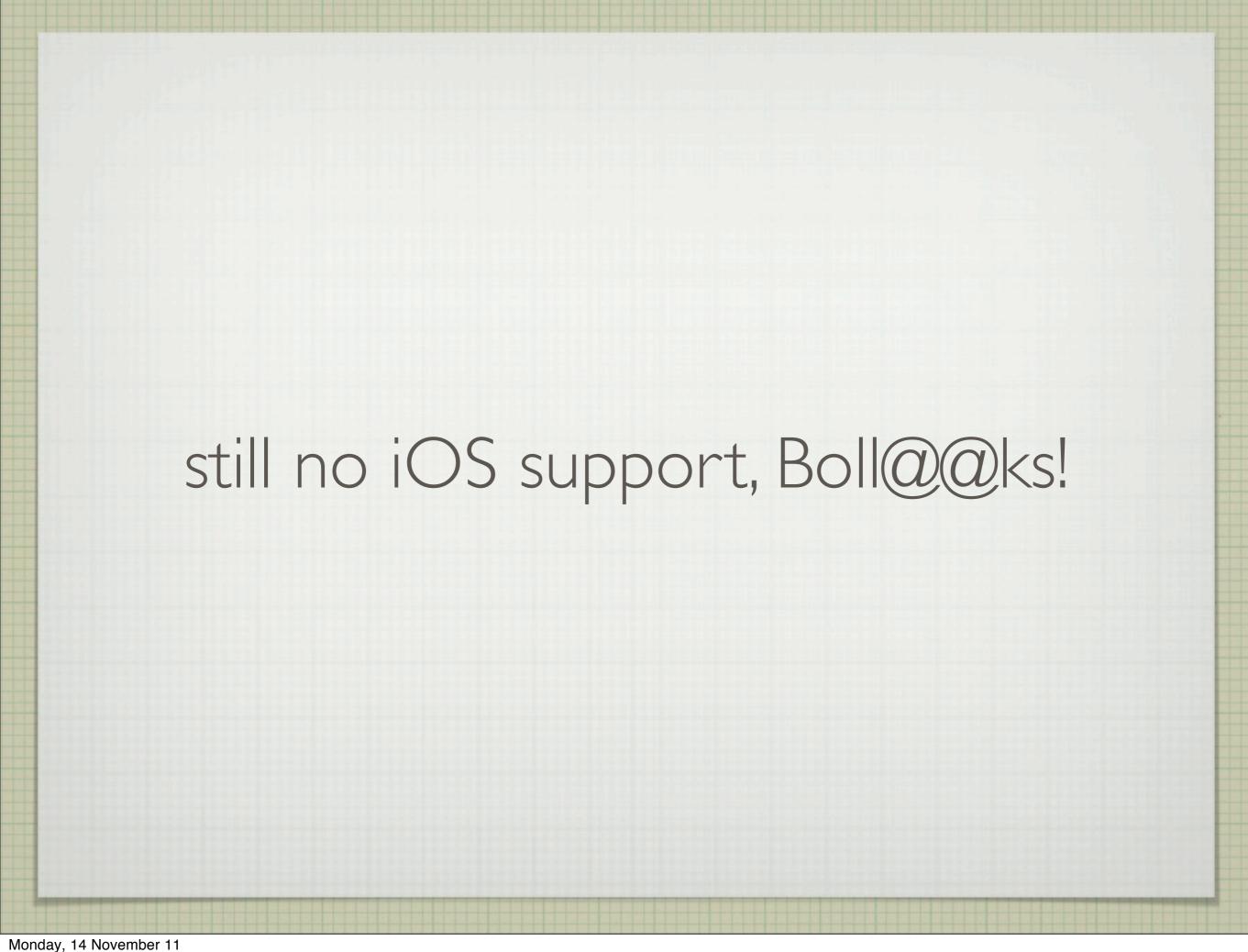
@abdels

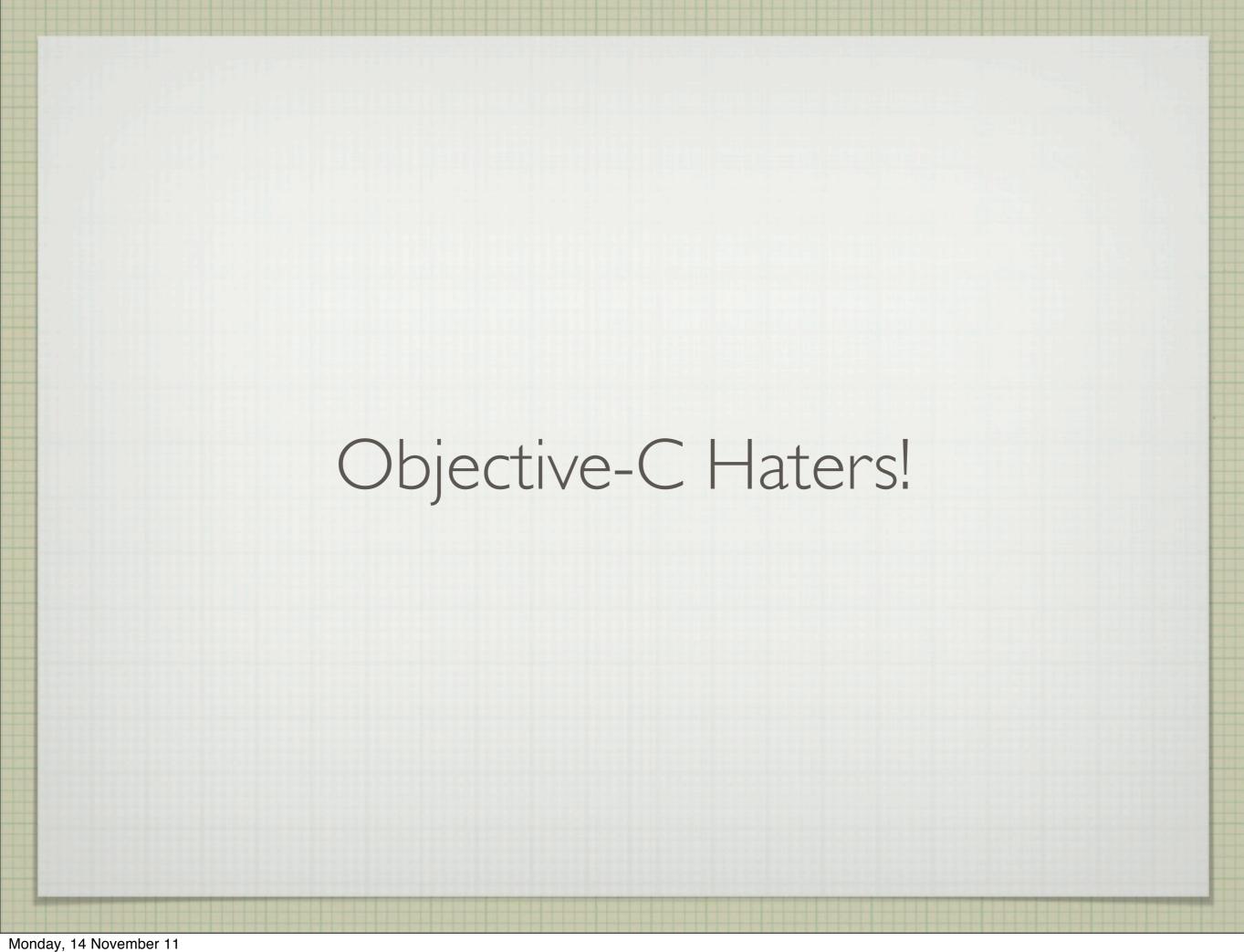


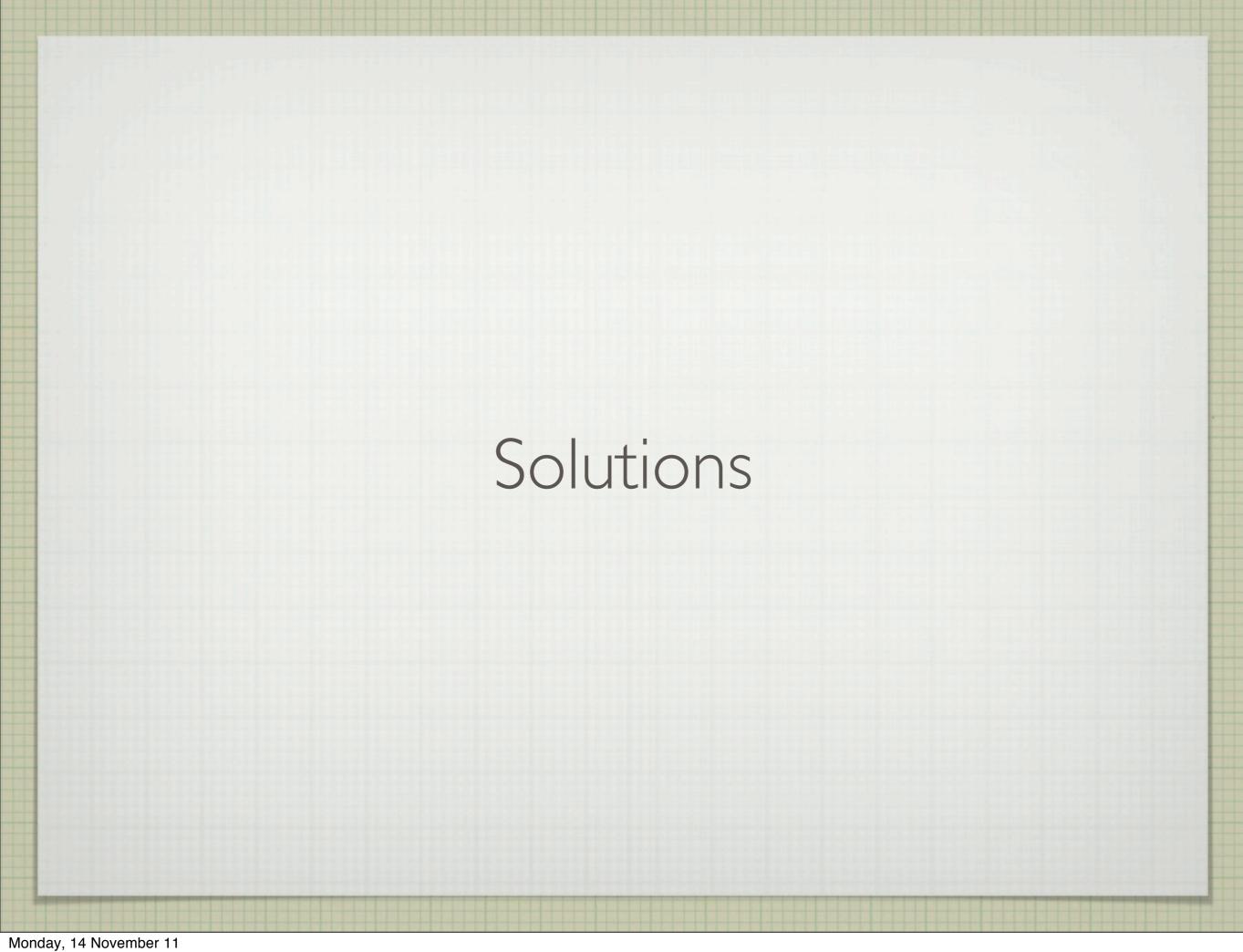


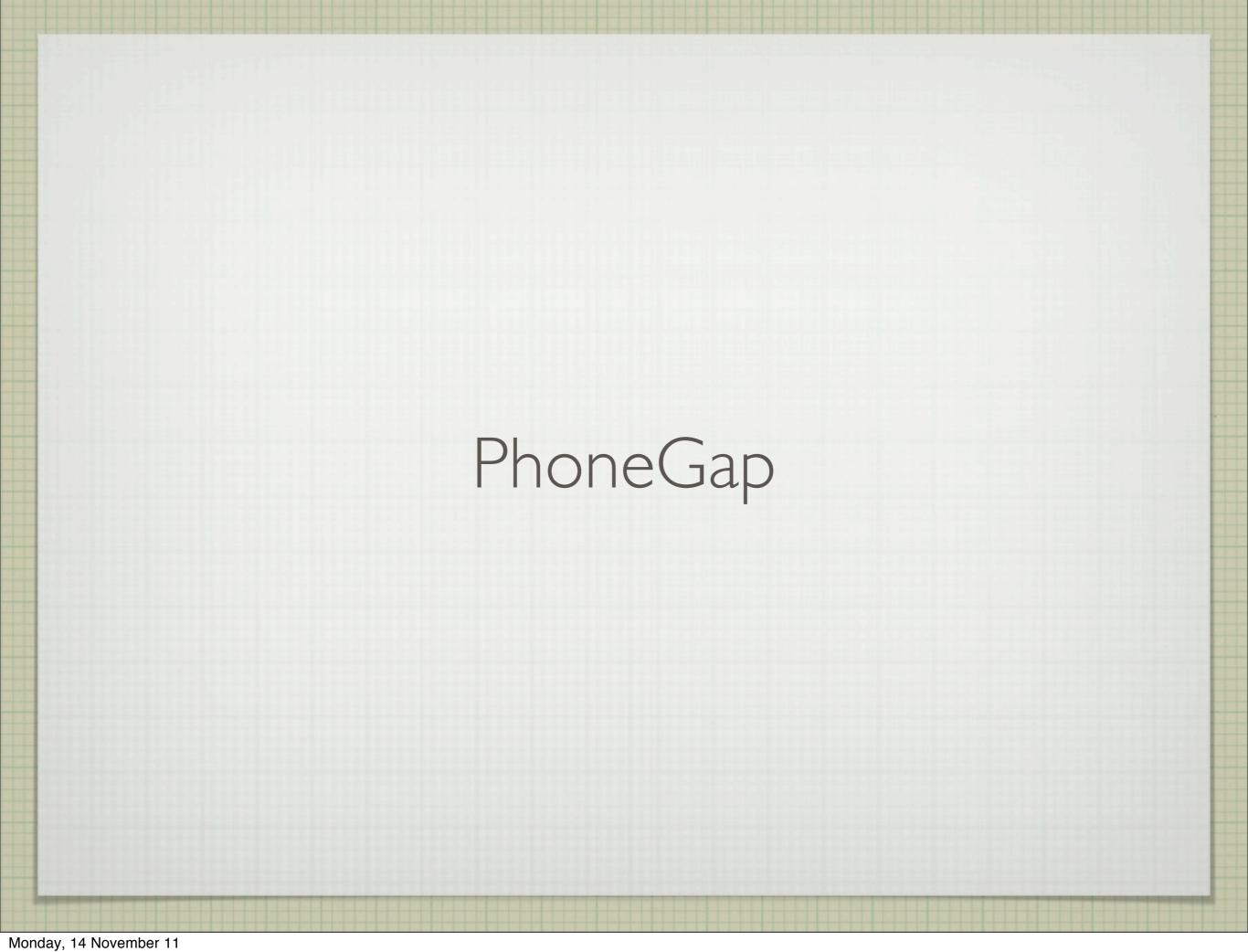




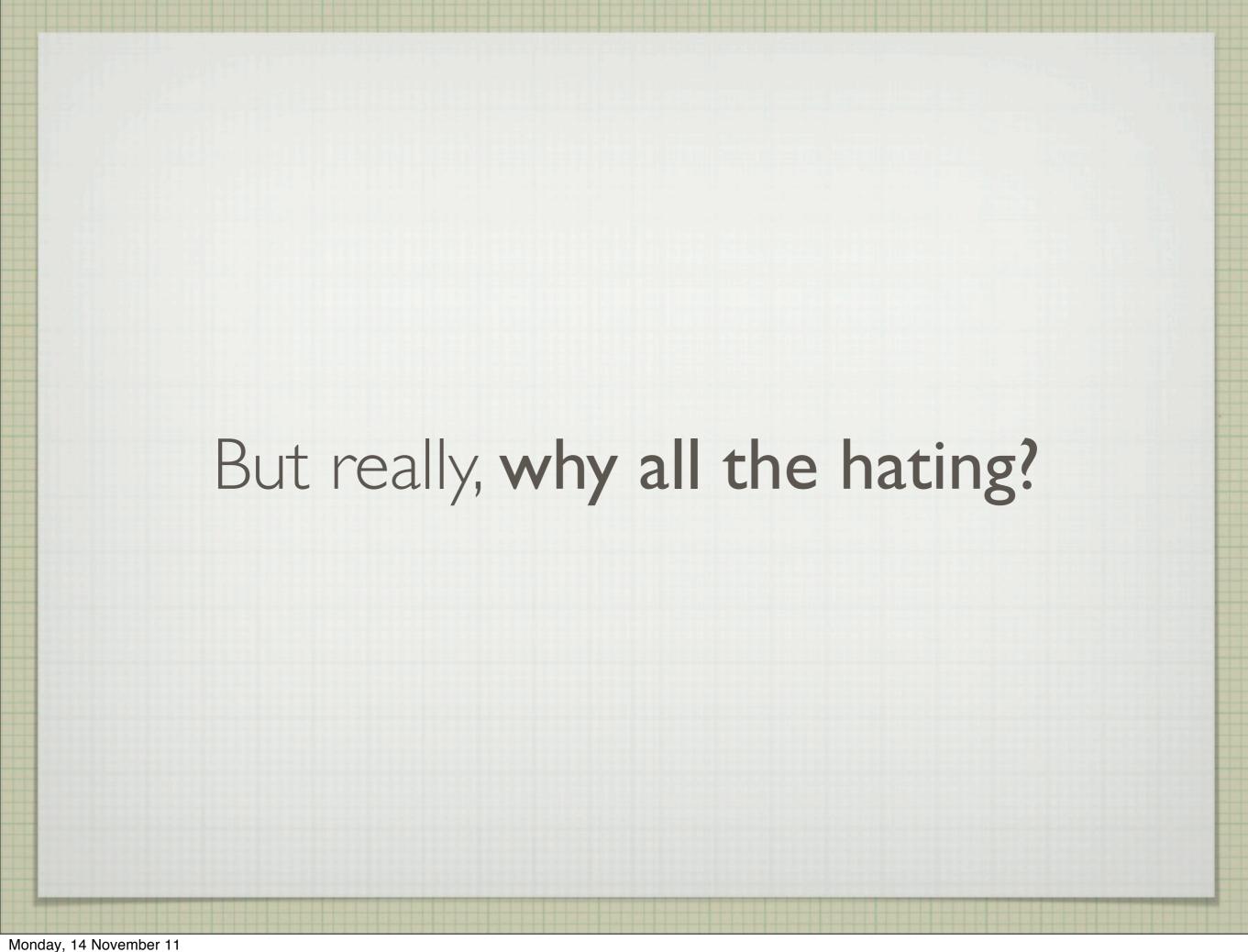








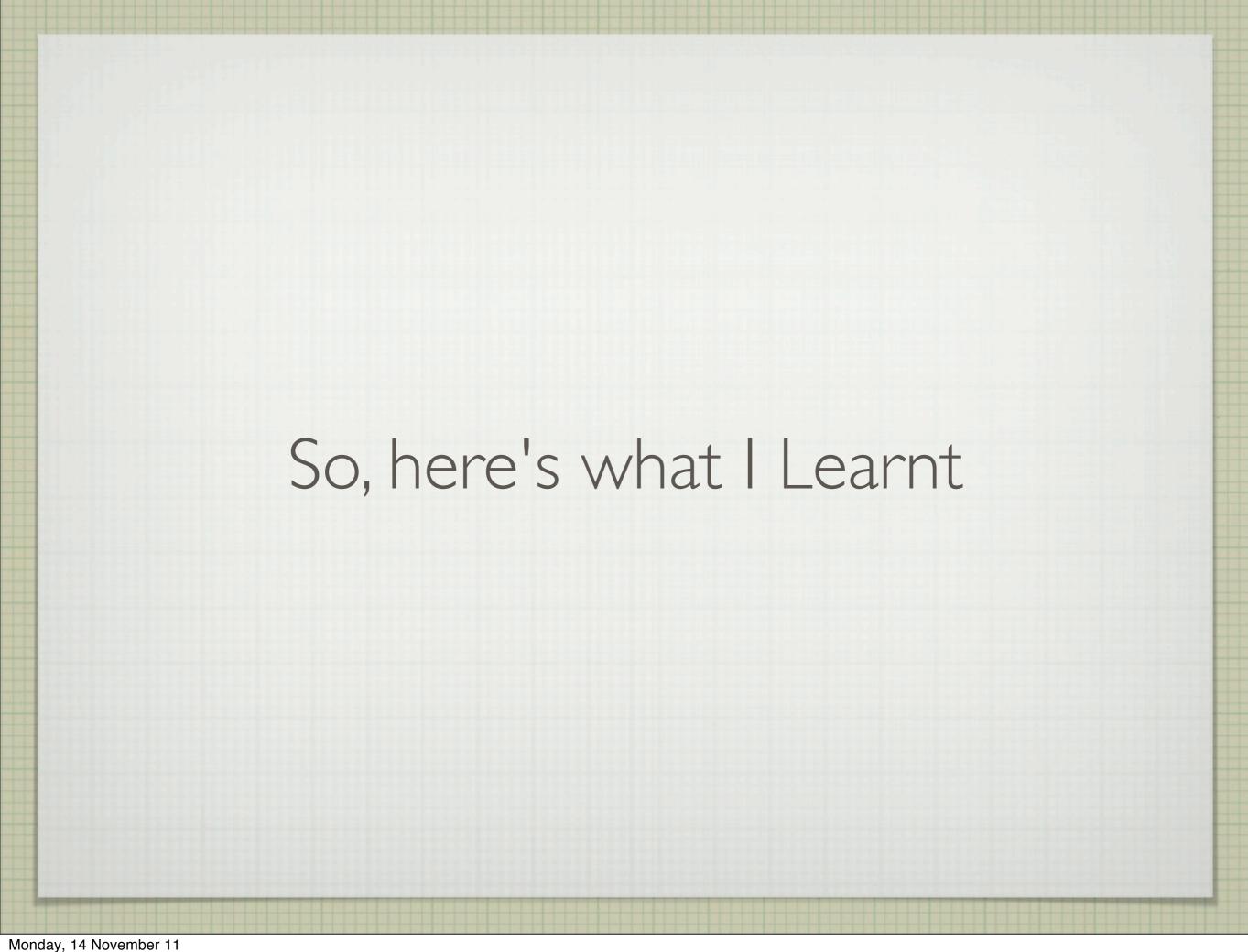


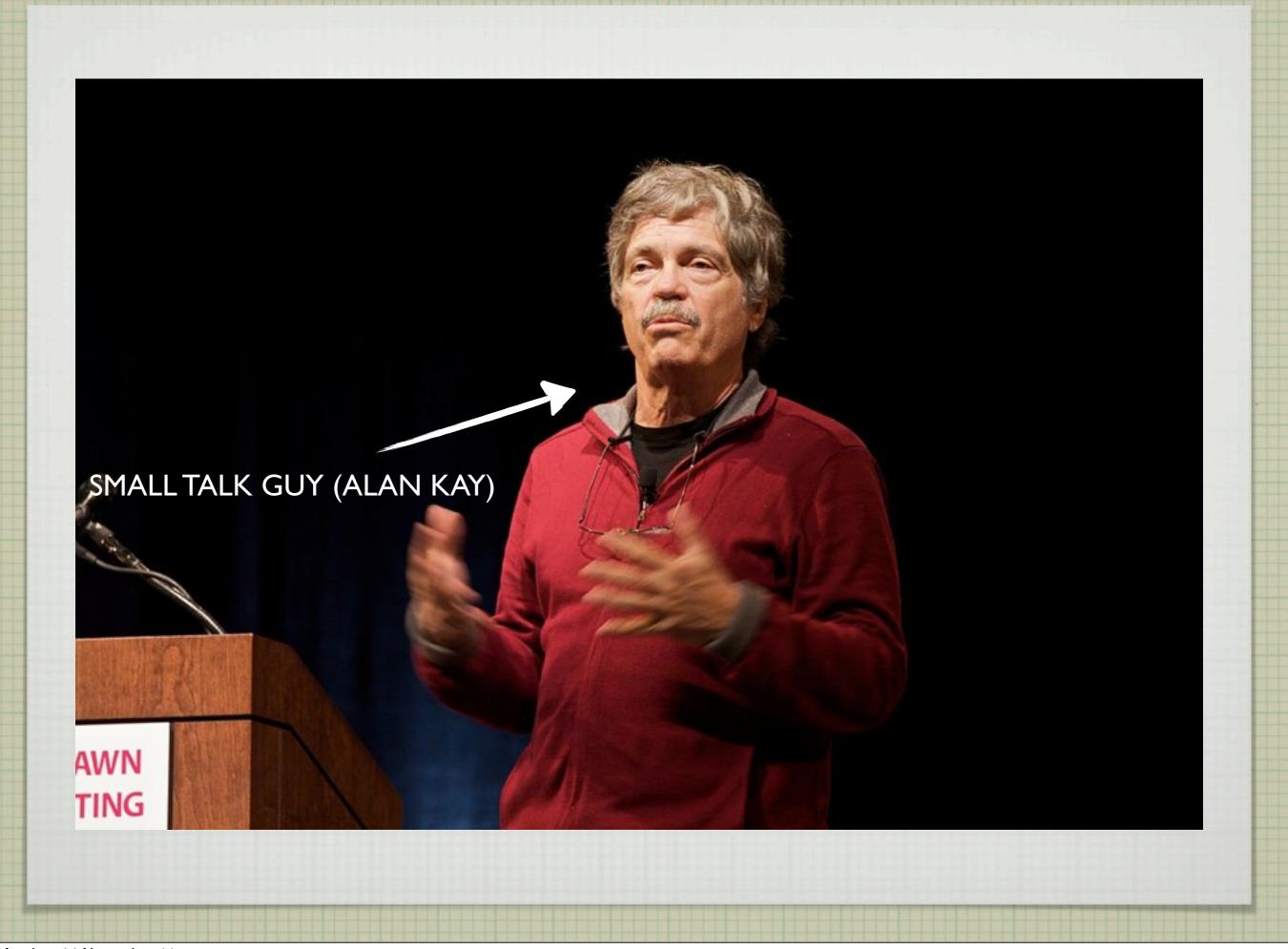


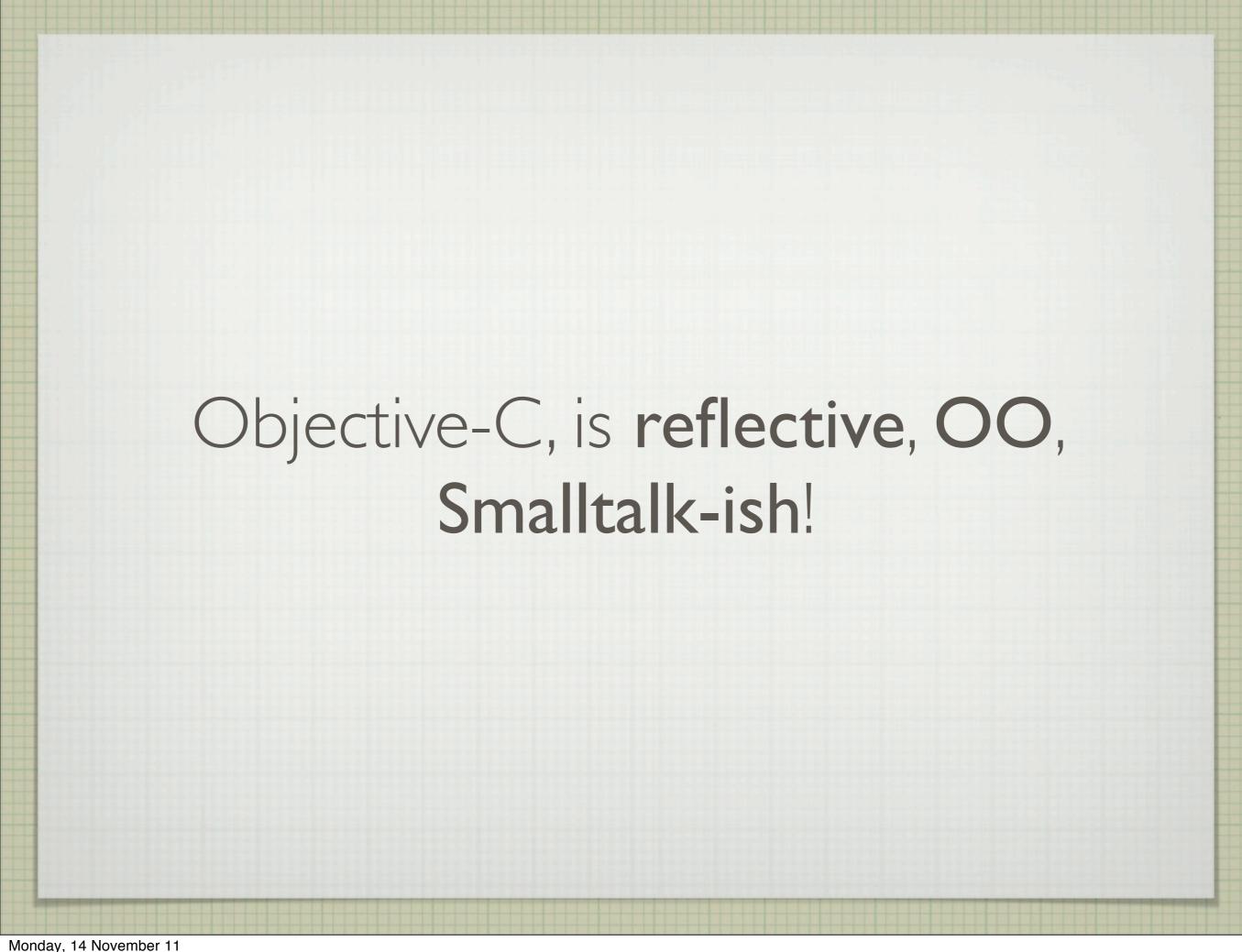


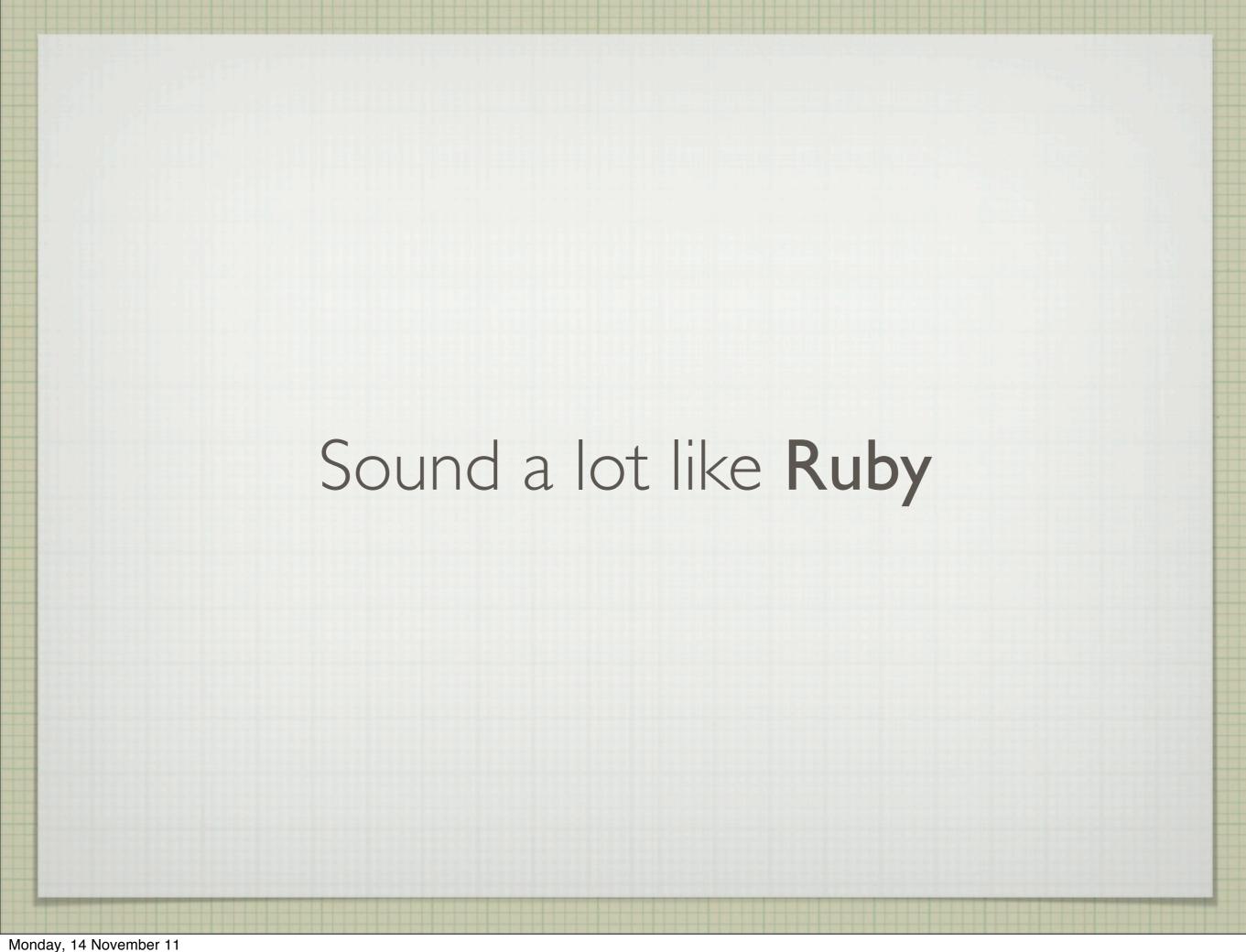
That's pride f**king with you. F**k pride. Pride only hurts. It never helps. You fight through that sh*t!

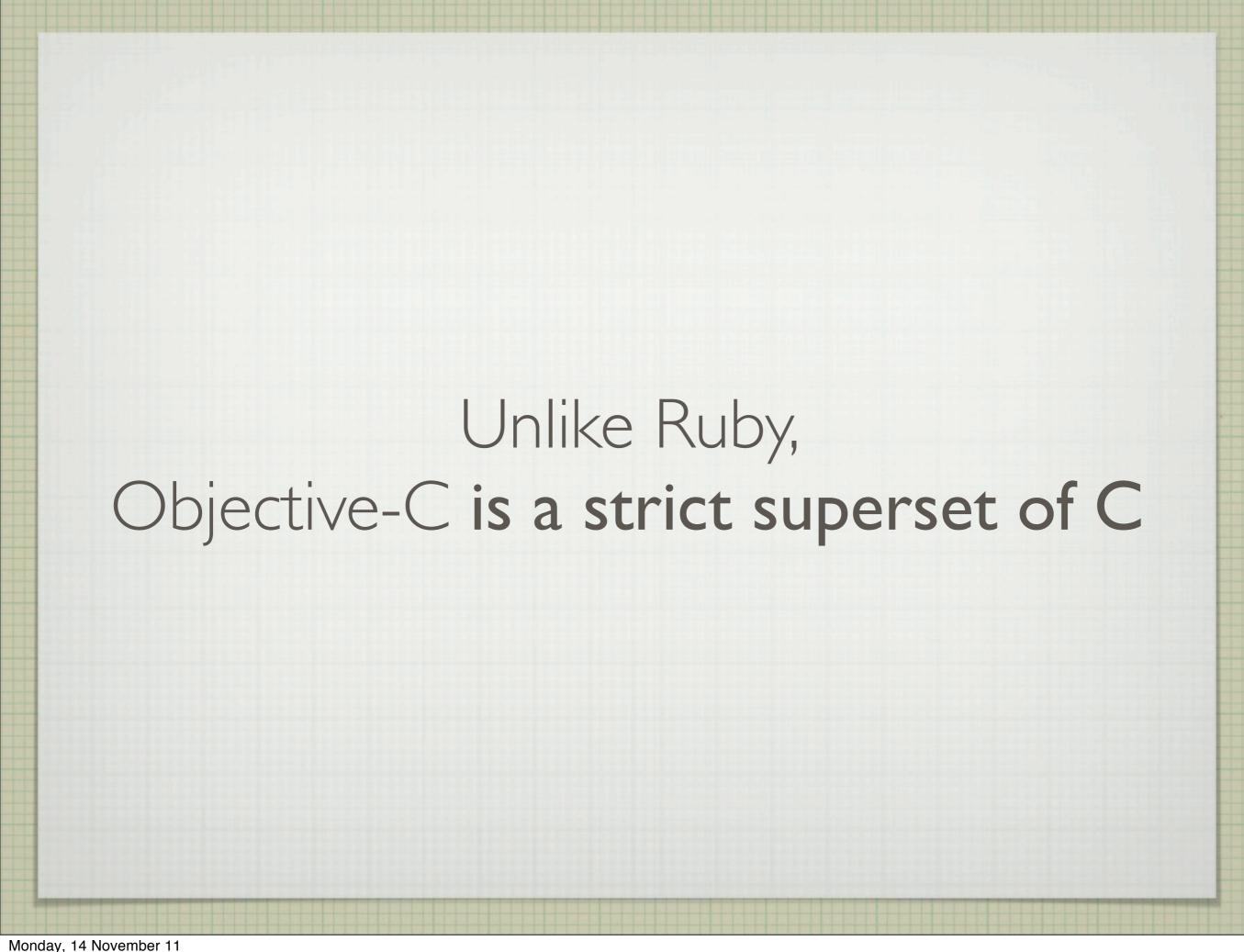












Object Message passing:

Obj-C

[world say:@"hello"];

Ruby

world.say("hello")

Object Message passing:

Obj-C

[world say:@"hello"];

[world performSelector:@selector(say:) withObject:@"hello"]

objc_sendMsg(id object, SEL selector)

Object Message passing:

Ruby

world.send(:say, "hello")

Non-strict typing (Duck Typing?)

Obj-C

id world = [[World alloc] init];

[world peace];

Here 'id' is a pointer to any object.

Non-strict typing (Duck Typing?)

Obj-C

World *world = [[World alloc] init];

[world peace];

This ensures method compiler checks!

Object declaration

```
Obj-C
```

MyClass * myObject = [[MyClass alloc] init];

```
Ruby (pseudo code)

class Object
  def self.new(*args)
  self.alloc.initialize(*args)
  end
end
```

Interface/Implementation (the C thing)

```
world.h (header file)
#import < UIKit/UIKit.h>
@interface World : NSObject {
 NSString *foo;
@property (copy, retain) NSString *foo;
-(NSString *) say:(NSString *)something;
@end
```

Interface/Implementation (the C thing)

```
world.m (implementation file)
#import "world.h"
@implementation World
@synthesize foo;
-(NSString *) say:(NSString *)something{}
- (void)dealloc {
 [foo release];
 [super dealloc];
@end
```

Properties aka attr_accessor

```
@interface World : NSObject {
    NSString *foo;
    ...
    @property (copy, retain) NSString *foo;
```

@end

@implementation World @synthesize foo;

@end

Properties aka attr_accessor

@synthesize: creates dynamic setters and getters and performs the necessary memory allocation.

```
-(void)setFoo:(Foo *)s {
    if(foo != s) {
        [foo release];
        foo = [s retain];
    }
}
-(Foo *)foo { return foo; }
```

Interfaces with Protocols

- @protocol Shopper
- (void)recession;
- (void)boom;
- @end
- @interface World : NSObject <Shopper> @end

Monkey Patching with Categories

```
- (Ullmage *)roundedCornerlmage;
@end
Ullmage+RoundedCorner.m
@implementation Ullmage
(RoundedCorner)
- (Ullmage *)roundedCornerlmage {
 //Make rounded corners
@end
```

@interface Ullmage (RoundedCorner)

Ullmage+RoundedCorner.h

We Have Blocks!

```
NSArray *films = [NSArray arrayWithObjects:@"Reservoir Dogs", @"Pulp Fiction", @"Kill Bill", nil];
```

```
[films enumerateObjectsUsingBlock:^(id object, NSUInteger
index, BOOL *stop) {
    NSLog(@"%@ film at index %d", object, index);
}];
```

We Have Blocks!

```
int (^negative)(int) = ^(int number) {
    return number * -1;
};
int result = negative(2);
```

REFERENCES

- HTTP://EN.WIKIPEDIA.ORG/WIKI/OBJECTIVE-C
- HTTP://EN.WIKIPEDIA.ORG/WIKI/MACRUBY
- HTTP://OFPS.OREILLY.COM/TITLES/9781449380373/INDEX.HTML
- HTTP://BLOG.PHUSION.NL/2010/03/24/OBJECTIVE-C-FOR-RUBY-DEVELOPERS-UN-NOT-SO-PETIT-INTERLUDE-1/
- ☐ HTTP://GOSHAKKK.NAME/BLOG/2011/08/19/OBJECTIVE-C-RUBYISTS-INSIGHT/
- ☐ HTTP://GOOGLE-STYLEGUIDE.GOOGLECODE.COM/SVN/TRUNK/ OBJCGUIDE.XML



THE SUPER OWL

http://superowl.me
@thesuperowl

SIGN UP:)

@abdels https://github.com/the-experimenters Monday, 14 November 11