Łukasz Adamczak
http://czak.pl

MacRuby

czyli Cocoa bardziej ludzkie

Cocoa

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

Cocoa.h

Cocoa

- Zbiór obiektowych API dla OS X, platforma runtime
- NeXTSTEP - rok 1989 (NS...)
- Foundation, GUI, dźwięk, grafika, sieć, i18n, ...
- Napisane w Objective-C/C, dostępne z Objective-C

Cocoa c.d.

```
// Stworzenie okna
NSWindow *window =
  [[NSWindow alloc]
    initWithContentRect:NSMakeRect(0, 0, 100, 100)
              styleMask:NSTitledWindowMask | NSClosableWindowMask
                backing:NSBackingStoreBuffered
                  defer:0
                 screen:0];

// Ustawienie tytułu
[window setTitle:@"Hello, World"];

// Wyświetlenie
[window display];
[window orderFrontRegardless];
```

[Objective-C]

Ruby a Objective-C

```ruby
# konstrukcja Stringa
s = "2 + 2 = #{2+2}"



# długość
s.length

# znak na 2giej pozycji
s[2]

# łączenie Stringów
s + "!!"
```

Ruby 1.9

```objc
// konstrukcja NSStringa
NSString *s =
  [NSString stringWithFormat:
    @"2 + 2 = %i", 2+2];

// długość
[s length];

// znak na 2giej pozycji
[s characterAtIndex:2];

// łączenie NSStringów
[s stringByAppendingString:@"!!"];
```

Objective-C

Ruby a Objective-C, c.d.

```ruby
# konstrukcja tablicy
a = ["pies", "kot", "małpa"]



# ostatni element
a.last

# czy zawiera "wilk"?
a.include? "wilk"

# dodanie elementu
a << "wilk"
```

Ruby 1.9

```objc
// konstrukcja tablicy
NSMutableArray *a =
  [NSMutableArray arrayWithObjects:
    @"pies", @"kot", @"malpa", nil];


// ostatni element
[a lastObject];

// czy zawiera "wilk"?
[a containsObject:@"wilk"];

// dodanie elementu
[a addObject:@"wilk"];
```

Objective-C

Ruby a Objective-C, c.d.

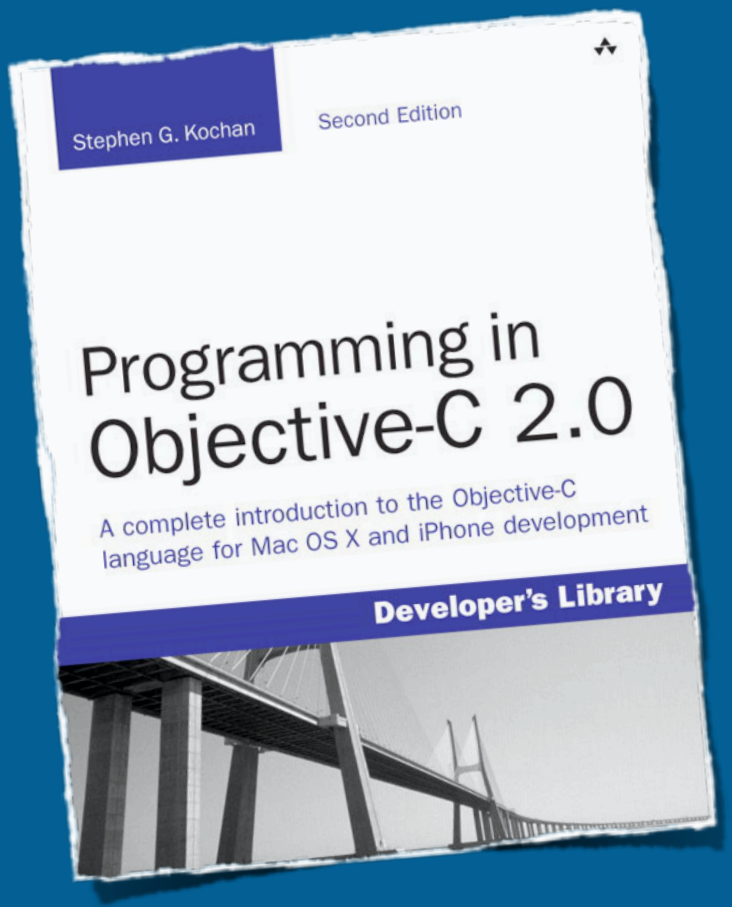
```ruby
# konstrukcja hasha
h = { "name" => "Anna Kowalska",
      "age"  => 34 }
```

Ruby 1.9

```objc
// konstrukcja hasha
NSDictionary *h =
  [NSDictionary
    dictionaryWithObjectsAndKeys:
      @"Anna Kowalska", @"name",
      [NSNumber numberWithInt:34], @"age",
      nil];
```

Objective-C

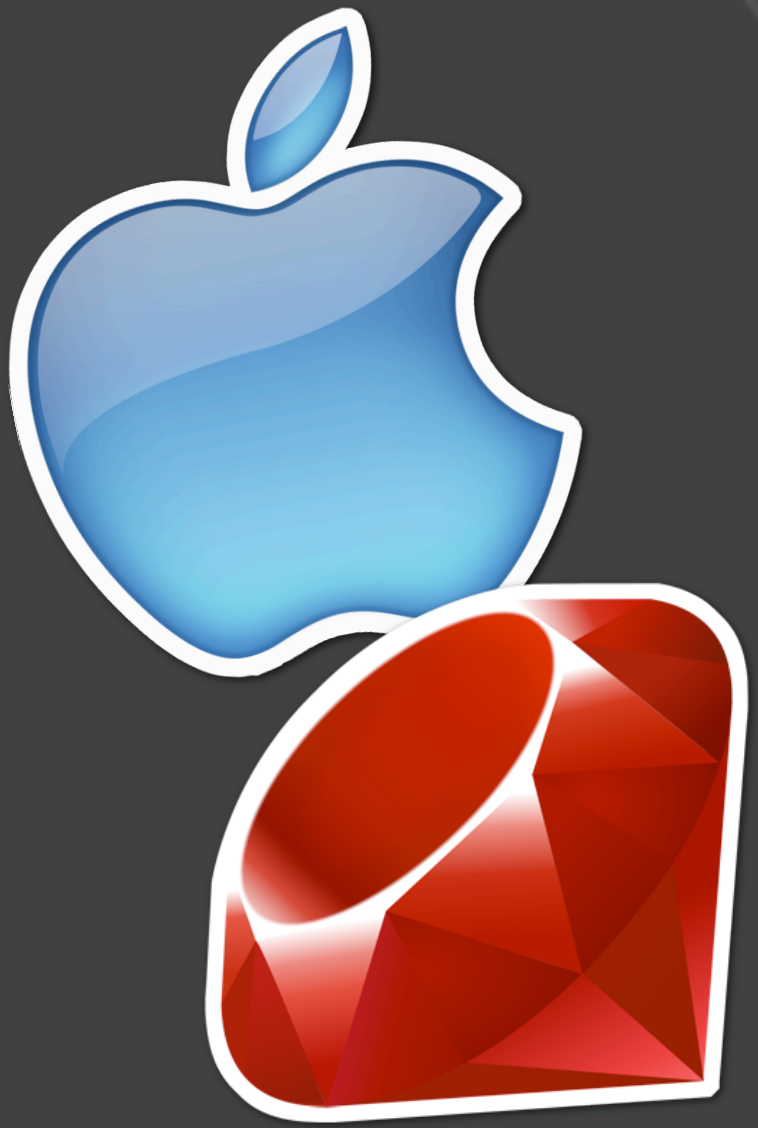
Objective-C



- Podstawowy język dla OS X
- Jedyny słuszny dla iPhone
- Podobieństwa z Rubym
- C => wydajność
- 2.0 o wiele wygodniejszy

Apple ♥ Ruby

Apple ♥ Ruby

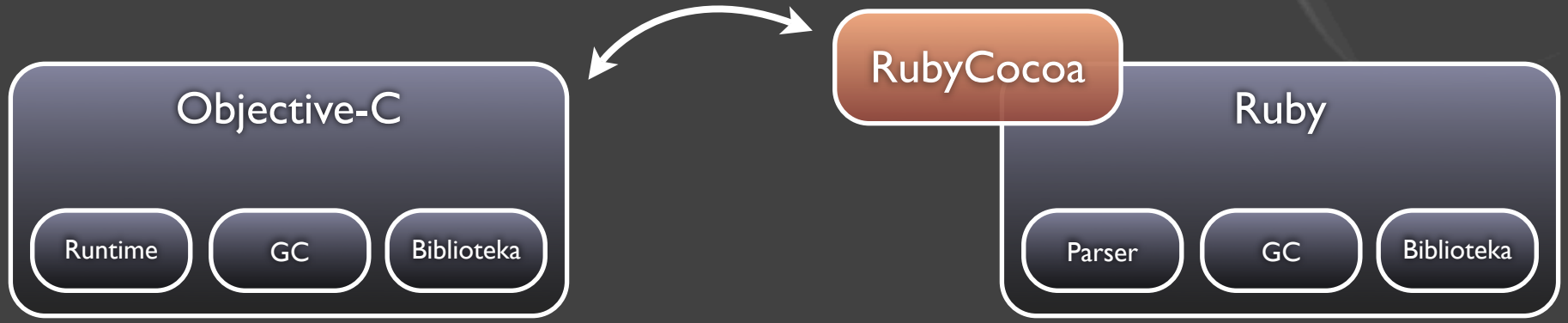


- Ruby w OS X od 2002
- 10.5:
  - Ruby 1.8.6
  - RubyGems
  - Rails 1.2.6
- 10.6:
  - Ruby 1.8.7
  - Rails 2.2

RubyCocoa

- Most między Rubym a Objective-C
- 2001, Hisakuni Fujimoto
- 2006, wsparcie Apple
- Stabilne, supportowane
- Integracja z Xcode

RubyCocoa, c.d.



- Różnice w komunikatach
- 2 Garbage Collectory
- Konwersja obiektów
- Różnice w wątkach

MacRuby


Ruby
Parser
GC
Biblioteka
Objective-C
Biblioteka
Runtime
GC
Biblioteka

MacRuby

- Projekt open-source Apple od 2007
- Kompletna implementacja Ruby 1.9
- Liga JRuby, IronRuby
- Integracja z Cocoa

MacRuby, c.d.

MacRuby, c.d.

- Każdy obiekt w Rubym jest NSObjectem
- Bez proxy, bez konwersji
- Runtime Objective-C
- "Yes, MacRuby is supposed to replace RubyCocoa"

- Laurent Sansonetti

Jak to działa - instalacja

- **Wersja 0.4 stabilna**
- **Komendy** `mac*`
- **Przykłady**
  `/Developer/Examples/Ruby/MacRuby`
- **Framework**
  `/Library/Frameworks/MacRuby.framework`
- **2 szablony do Xcode**

Jak to działa - kod

```
# Framework, z którego chcemy korzystać
framework 'Cocoa'       # albo 'QTKit', 'WebKit', ...

# [h setObject:@"Mariola" forKey:@"name"];
h.setObject "Mariola", forKey:"name"
h.setObject "Mariola", :forKey => "name"
h["name"] = "Mariola"

# initWithContentRect:styleMask:backing:defer:
window.initWithContentRect frame,
  styleMask:NSBorderlessWindowMask,
    backing:NSBackingStoreBuffered,
      defer:false
```

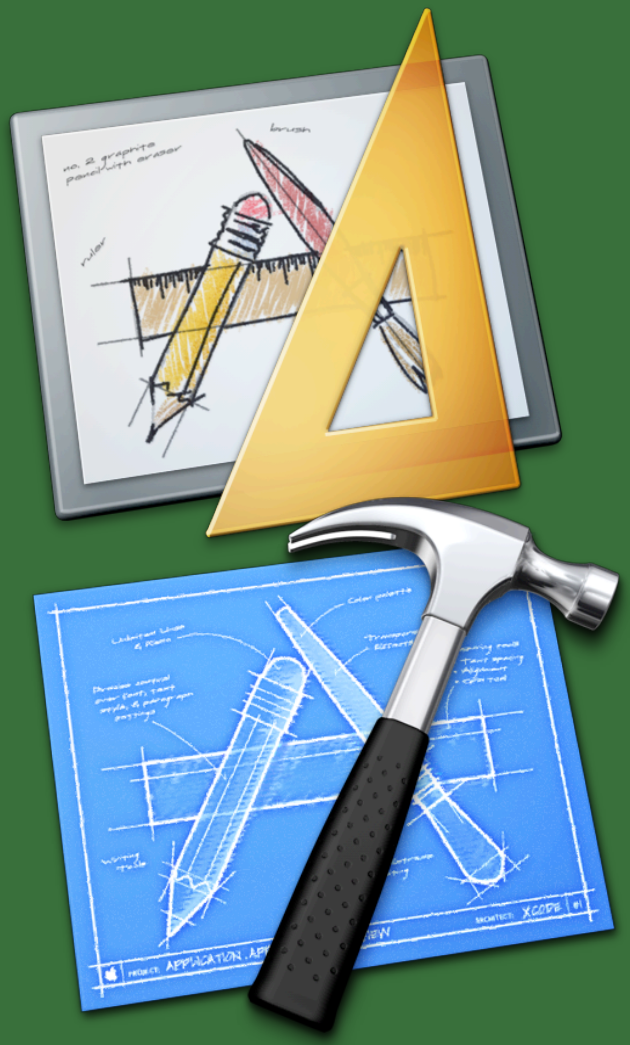
Jak to działa - kod

```
# Skróty setX i isX
# [imageView setImage:myImage];
imageView.image = myImage

# [imageView isEditable];
imageView.editable?


# Struktury jako klasy & dostęp do stałych
p = NSPoint.new(10, 30)
r = NSRect.new(p, NSZeroSize)
```

Jak to działa - Xcode & IB



- Szablony aplikacji w Xcode
- Klasy Ruby rozpoznawane w Interface Builderze
- Outlets
  `attr_accessor, attr_writer`
- Actions
  `def action(sender)`

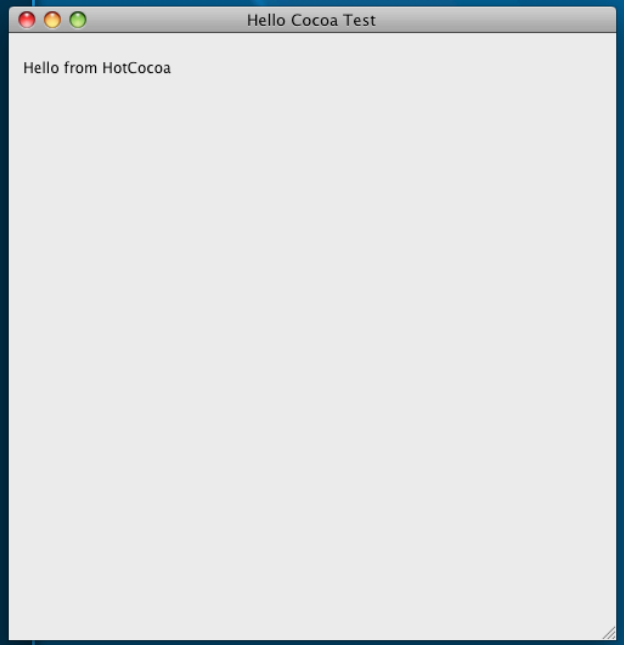
HotCocoa

- **Uproszczona budowa GUI**
- **Komenda**
  `$ hotcocoa myapp`
- **Biblioteka**
  `require 'hotcocoa'; include HotCocoa`
- **Metody skrótowe, "smart defaults"**
- **HotConsole**
  http://github.com/vincentisambart/hotconsole

HotCocoa, c.d.

```
require 'hotcocoa'; include HotCocoa

application do |app|
  window :frame => [100, 100, 500, 500],
         :title => "Hello Cocoa Test" do |win|
    win << label(:text => "Hello from HotCocoa",
                 :layout => {:start => false})
    win.will_close { exit }
  end
end
```



MacRuby

Uwaga

- **RubyGems**
- **Typy C zwracane lub przekazywane do Ruby'ego**
  `patrz: BridgeSupport`
- **Niezabezpieczony kod**
  `MyApp.app/Contents/Resources/lib/*.rb`

Materiały

- http://macruby.org
- http://rubyonosx.com
- http://www.merbist.com
- http://www.comprovisation.com
- http://www.cocoadevcentral.com
- http://peepcode.com/products/meet-macruby
- http://www.johnmacshea.org/examples/GameRelated
- http://lists.macosforge.org/mailman/listinfo.cgi/macruby-devel
- http://nubyonrails.com/articles/macruby-presentation-from-rubyfest
- http://rubyconf2008.confreaks.com/macruby-ruby-for-your-mac.html
- http://www.slideshare.net/mattetti/macruby-when-objectivec-and-ruby-meet
- http://www.slideshare.net/mattetti/macruby-hotcocoa-presentation-by-rich-kilmer
- http://rubyconf2008.confreaks.com/os-x-application-development-with-hotcocoa.html
- http://media.fngtps.com/video/2009/rubyonosx2009-02-hotcocoa_introduction-rich_kilmer.torrent

Dziękuję