Wykład nr 1
Język Ruby i framework Rails

dr inż. Aleksander Smywiński-Pohl

Elektroniczne Przetwarzanie Informacji
Konsultacje: czw. 14.00-15.30, pokój 3.211

Historia Rubiego i Railsów

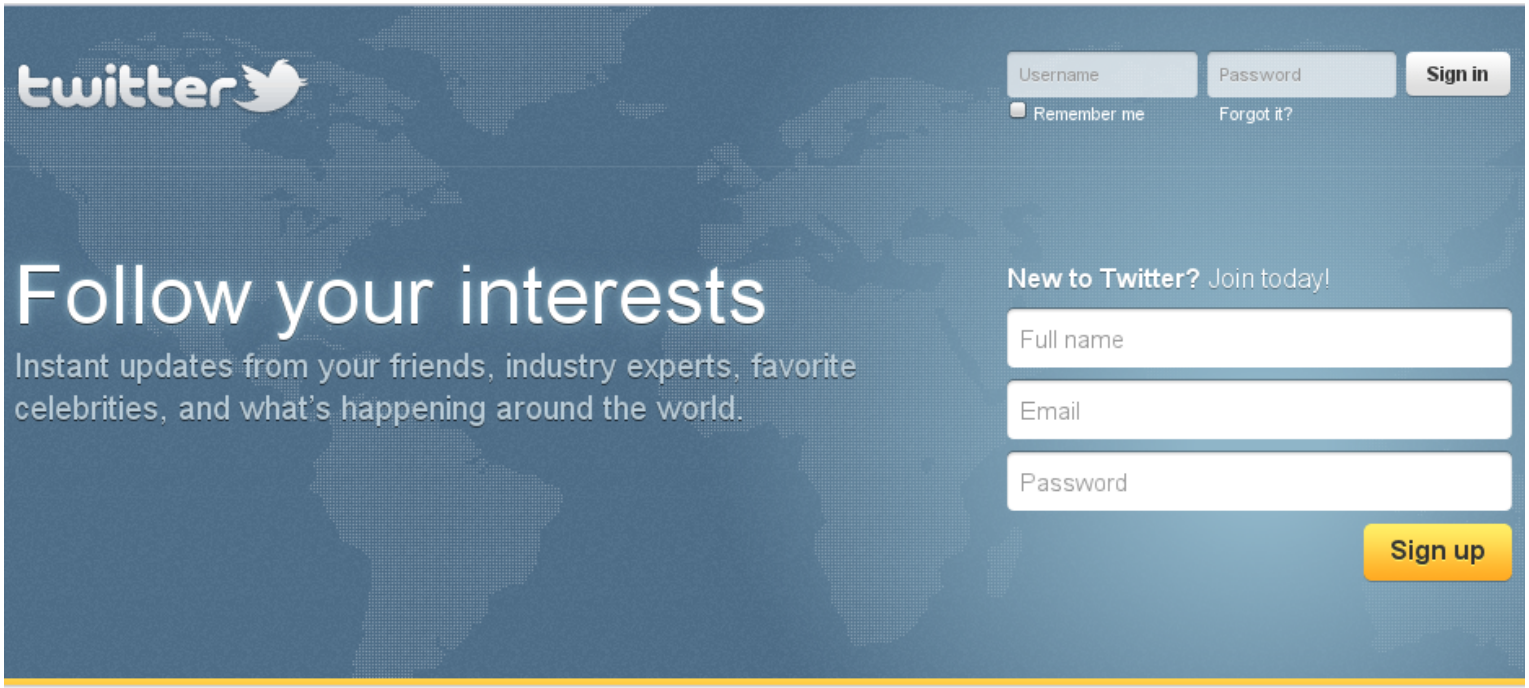
- 1993 – Yukihiro Matsumoto San, Japonia
- 1995 – pierwsze wydanie, lista dyskusyjna w Japonii
- 1996 – Ruby wydanie 1.0
- 1999 – pierwsza angielska lista dyskusyjna
- 2000 – pierwsza angielska książka na temat Rubiego
- 2004 – początek pracy nad Ruby on Rails, David Heinemeier Hansson, Dania
- 2005 – Ruby on Rails wydanie 1.0
- 2007 – Ruby wydanie 1.9.0
- 2013 – Ruby wydanie 2.0.0
- 2013 – Ruby on Rails wydanie 4

Otoczenie Rubiego

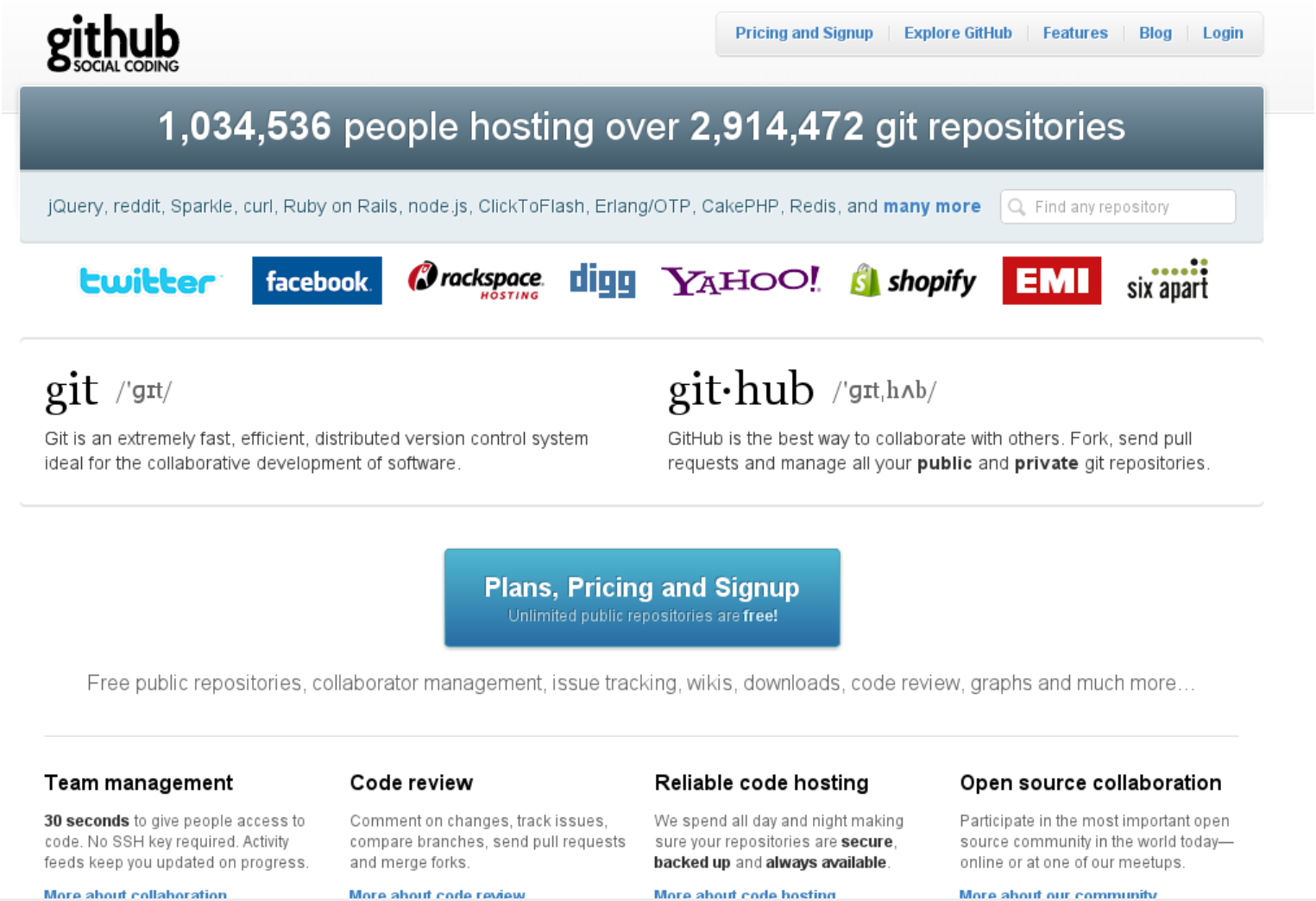
- C/C++ – systemy operacyjne, systemy wbudowane, narzędzia systemowe
- Java – aplikacje korporacyjne i duża plikacje serwerowe
- C# – programy dla systemu Windows, aplikacje webowe
- Objective-C – programy dla systemów OS X i iOS
- PHP – aplikacje webowe
- Python, Perl – programowanie skryptowe, programowanie systemowe, aplikacje webowe
- JavaScript – programowanie po stronie klienta (w przeglądarce), ostatnio również po stronie serwera (node.js)
- ActionScript – programowanie po stronie klienta (Flash)
- Swift – nowy język na platformy iOS i OS X wspierany przez Apple

Sukcesy Rubiego/Railsów – Basecamp

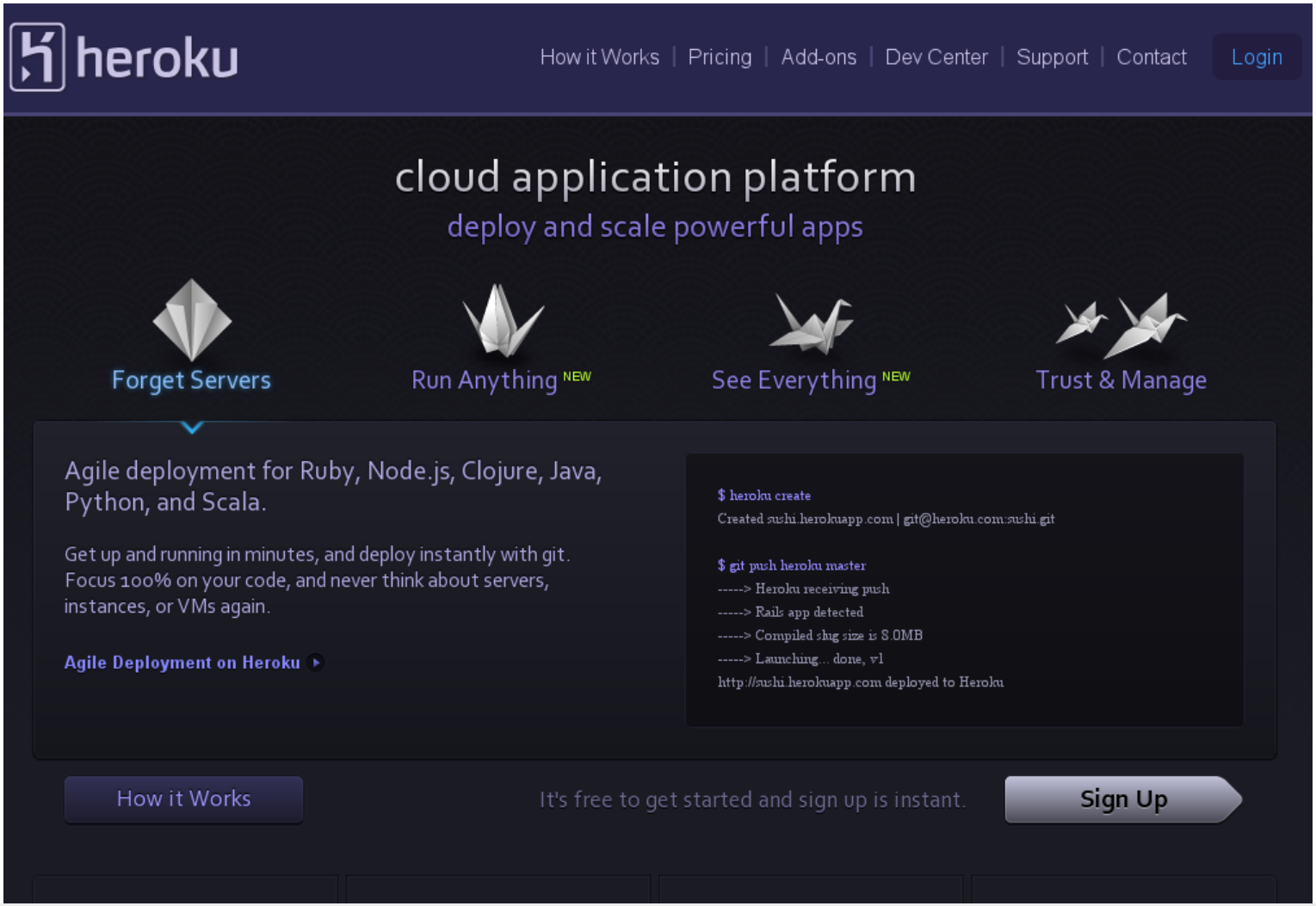
Sukcesy Rubiego/Railsów – Twitter



Sukcesy Rubiego/Railsów – Github

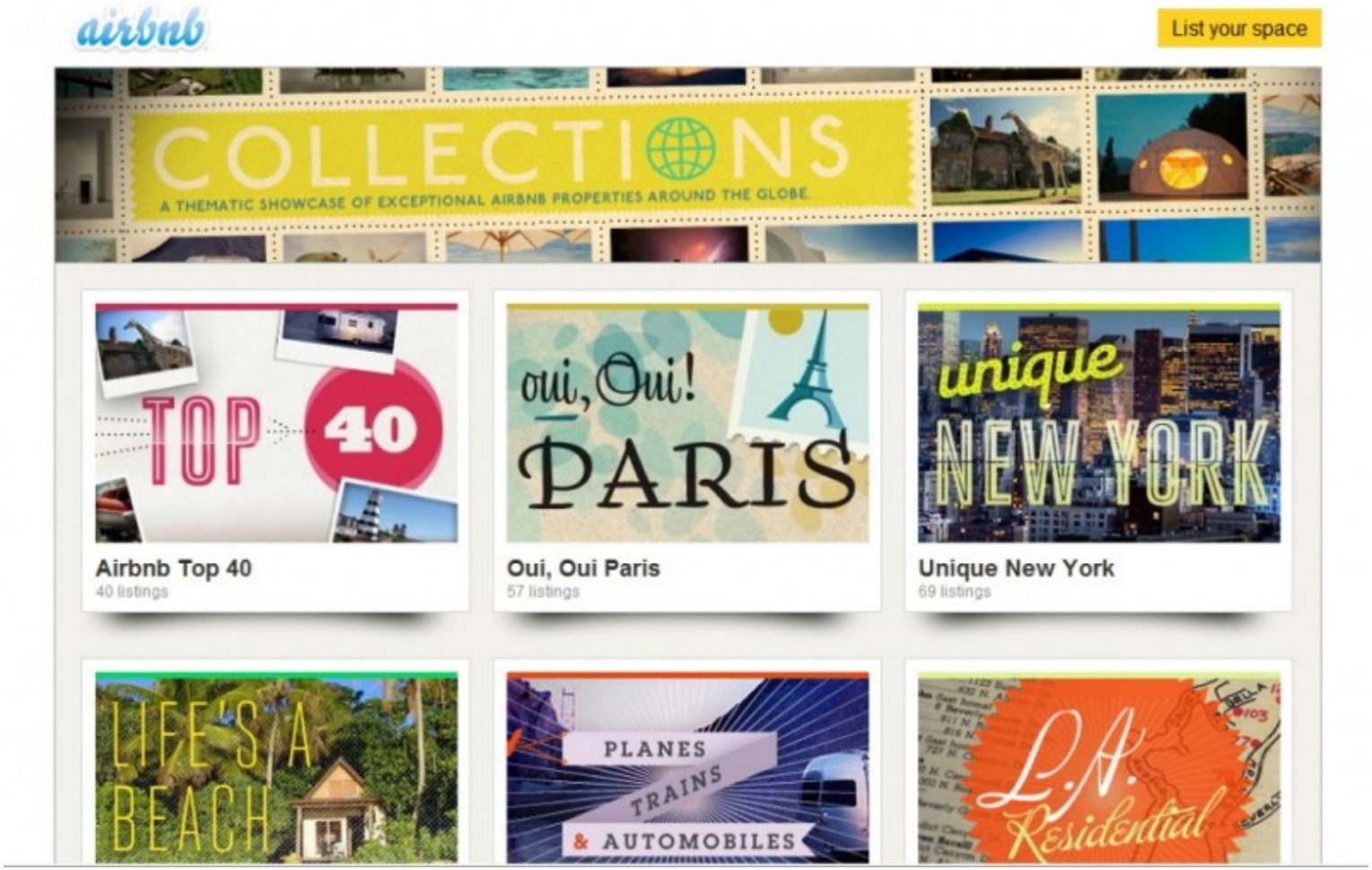


Sukcesy Rubiego/Railsów – Heroku



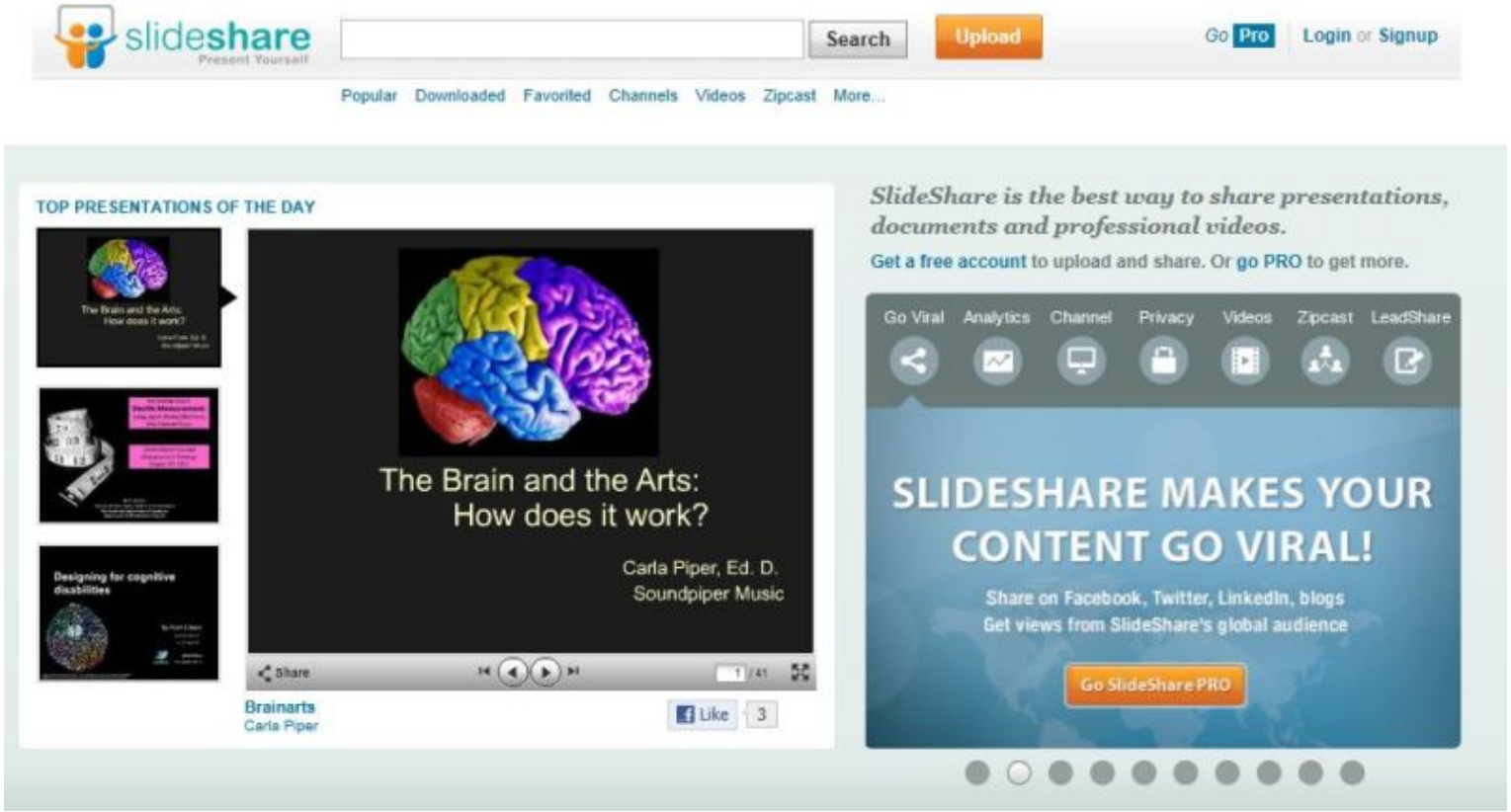
Sukcesy Rubiego/Railsów – Groupon

Sukcesy Rubiego/Railsów – AirBnB

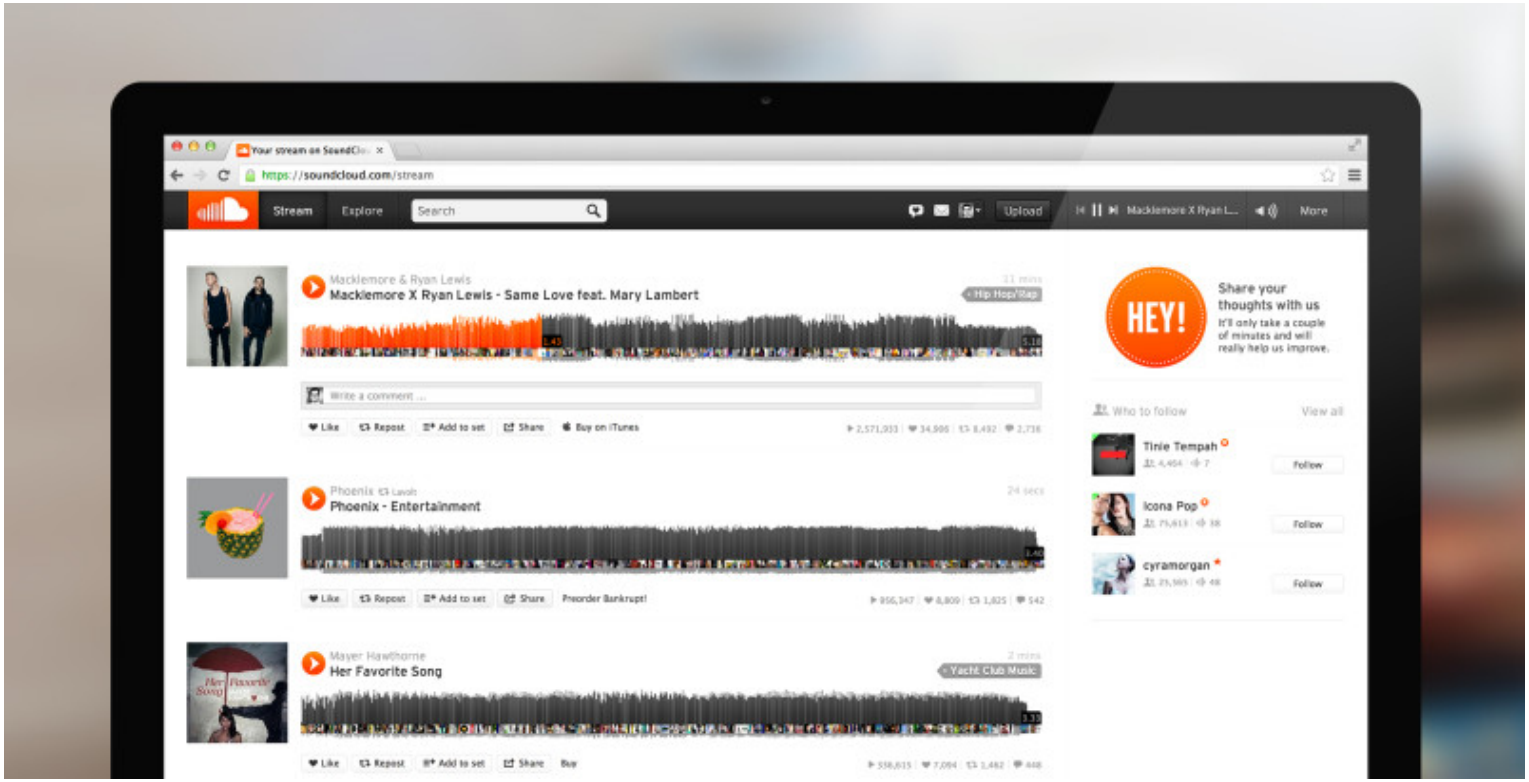


Sukcesy Rubiego/Railsów – bloomberg

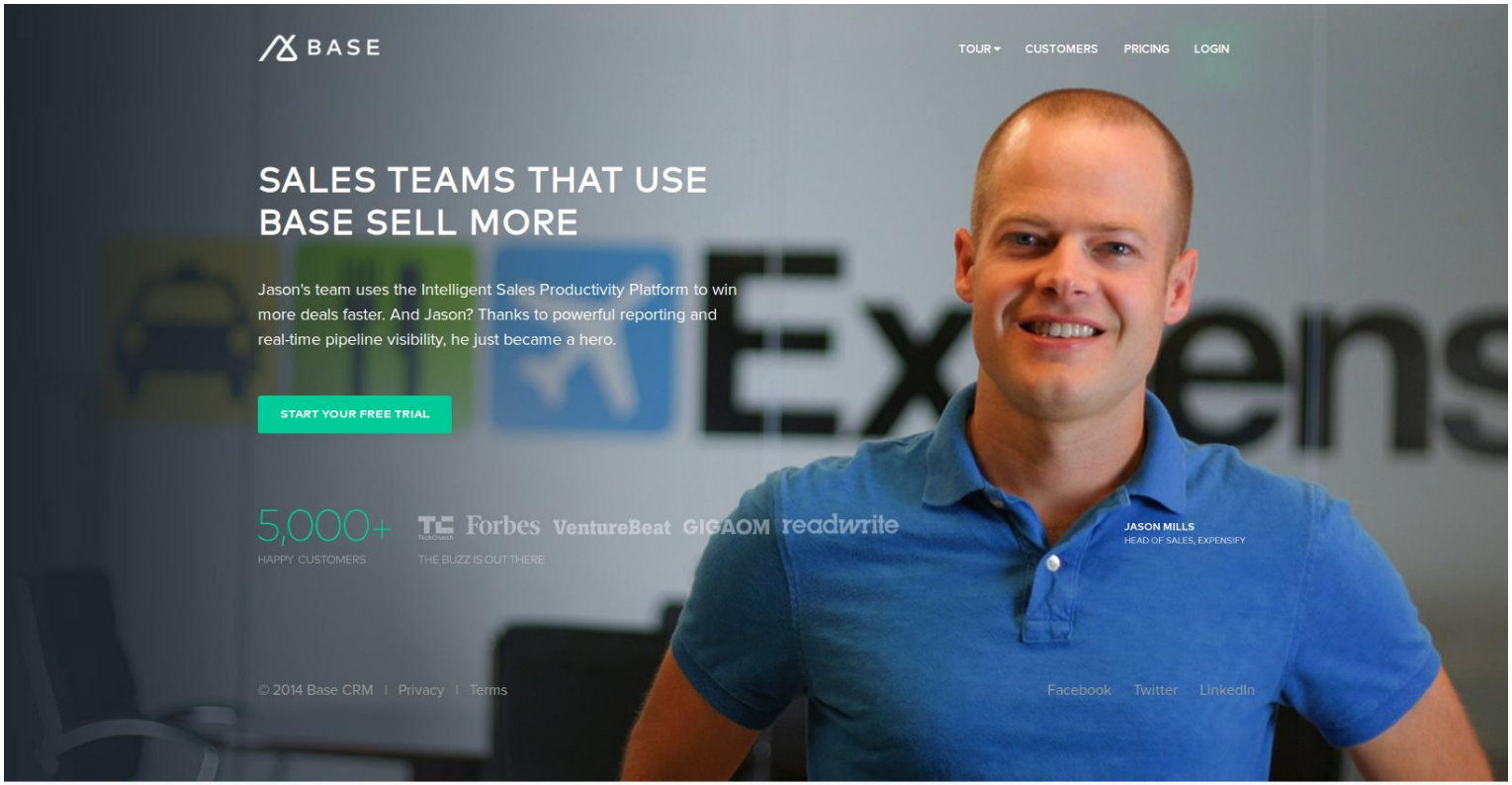
Sukcesy Rubiego/Railsów – Slideshare



Sukcesy Rubiego/Railsów – Soundcloud



Sukcesy Rubiego/Railsów – Base



Sukcesy Rubiego/Railsów – Kontomierz

Sukcesy Rubiego/Railsów – Sentymetr

Praca

- Ogłoszenia na `forum.rubyonrails.pl`
- Zwykle podawane są widełki płacowe.
- Jak zdobyć pracę?
  - odpowiadać na pytania na stackoverflow.com
  - rozwijać opensourcowy projekt na githubie
  - mieć dużo plusów na LinkedIn w działce Ruby/Ruby on Rails
- Pracodawcy często sami się zgłaszają.
- Rails Girls Summer of Code.

Cechy języka

- 100% obiektowo zorientowany
- dziedziczenie jednobazowe
- interpretowany
- dynamicznie typizowany
- silnie typizowany
- funkcjonalny styl programowania
- przeciążanie operatorów
- *garbage collector*
- mechanizm wyjątków
- bloki i domknięcia
- metaprogramowanie

100% obiektowo zorientowany

wszystko jest obiektem, w tym:

```
nil                 # wartość pusta
nil.nil?            #=> true
true                # wartość logiczna
true.nil?           #=> false
1                   # liczby
1.zero?             #=> false
"abc"               # łańcuchy znaków
"abc".empty?        #=> false
[1,2,3]             # tablice
[1,2,3][0]          #=> 1
/a|b/               # wyrażenia regularne
/a|b/.match("a")    #=> 0
lambda { }          # bloki kodu (funkcje anonimowe)
```

Interpreter

```
ruby-1.9.2-p290 > a = 10
 => 10
ruby-1.9.2-p290 > b = 5
 => 5
ruby-1.9.2-p290 > a + b
 => 15
ruby-1.9.2-p290 > def suma(a,b)
ruby-1.9.2-p290 ?>  a + b
ruby-1.9.2-p290 ?>  end
 => nil
ruby-1.9.2-p290 > suma(1,2)
 => 3
ruby-1.9.2-p290 >
```

Dynamiczna typizacja

```ruby
a = [1,2,3]
a[1]                    #=> 2

a = "ala ma kota"
a[1]                    #=> "l"

def sum(a,b)
  a + b
end
sum(1,2)                #=> 3
sum([1],[2,3])          #=> [1,2,3]
```

Silna typizacja

```ruby
a = 10
b = "5"
a + b
# TypeError: String can't be coerced into Fixnum
# from (irb):3:in `+'

a = 10
b = 5.0
a + b                #=> 15.0

if 0
  puts "Zero"        #=> "Zero"
end
```

Funkcjonalny styl programowania

```ruby
str = "ala ma kota"
method = str.method(:empty?)
method.call                          #=> false

closure = lambda {|a,b| a + b}
closure.call(1,3)                    #=> 4

def multiply(multiplier)
  lambda {|a| a * multiplier }
end
three_times = multiply(3)
three_times.call(5)                  #=> 15
three_times.call(-2)                  #=> -6
```

Mechanizm wyjątków

```ruby
begin
  file = File.open("nie_ma_mnie.txt")
  file.each do |line|
    puts line
  end
rescue Exception => ex
  puts "Wystąpił wyjątek #{ex}"
end
```

Ewaluacja wyrażeń

```ruby
name = "Ala"
length = name.size > 5 ? "długie" : "krótkie"

length =
  if name.size > 5
    "długie"
  else
    "krótkie"
  end
```

Przegląd składni

- brak średników (pod warunkiem, że nie umieszczasz wielu poleceń w jednej linii, co jednak jest odradzane)
- brak wymogów co do wcięć (oczywiście właściwe wcięcia poprawiają czytelność kodu)
- brak deklaracji typów – wystarcza inicjowanie zmiennych
- 1-linijkowe komentarze zaczynają się znakiem #
- specjalny *obiekt* reprezentujący wartość pustą zwany `nil`
- wszystko jest wyrażeniem (również `if`, `case`, itp.)
- opcjonalne nawiasy

Materiały (1/2)

- Oficjalna strona języka `http://www.ruby-lang.org/pl/`
- Pierwsze wydanie popularnej 'książki z kilofem' `http://www.ruby-doc.org/docs/ProgrammingRuby/`
- Najdziwniejszy podręcznik programowania na świecie `http://mislav.uniqpath.com/poignant-guide/`
- Wprowadzenie do Rubiego mojego autorstwa `http://apohllo.pl/dydaktyka/ruby/intro`
- Dokumentacja podstawowych klas Rubiego i Rails `http://railsapi.com/`
- Webowy interfejs do testowania wyrażeń regularnych `http://rubular.com`

Materiały (2/2)

- Przewodniki po Ruby on Rails (en. – 4.1) `http://guides.rubyonrails.org`
- Ruby toolbox - przydatne biblioteki Rubiego i rozszerzenia frameworku Rails `http://ruby-toolbox.com/`
- Książka „Ruby: programowanie" David Flanagan, Yukihiro Matsumoto (jest w bibliotece WZKS)
- Książka (z kilofem) „Programming Ruby (2nd edition): The Pragmatic Programmers' Guide" Dave Thomas, Andy Hunt, Chad Fowler
- Książka „Ruby. Tao programowania w 400 przykładach" Hal Fulton

Podziękowania dla:

- Agnieszki Figiel, za udostępnienie prezentacji w postaci plików źródłowych
- Marka Kowalcze oraz Jakuba Kuźmy z grupy SRUG (srug.pl), za pomoc przy kolorowaniu składniu w Latex'u