Moving from C++ to Ruby

Leslie Brown

Career “Path”

- MIT - Aerospace Engineering
- Lockheed Martin - Systems Engineer
- Freedom High School - Math Teacher
- SimiGon - Software Engineer (more C++)
- Modernmeal - Ruby on Rails / Ember.js

C++

- Statically Typed
- Compiled
- Runs fast!

Ruby

- Dynamically Typed
- Interpreted
- Relatively slow

Ruby was written in C

What are they used for?

C++

HALO 3

F-35 LIGHTNING II

CERN

Adobe

NASA

Ruby

Homebrew

The missing package manager for OS X

RAILS

CHEF

metasploit®

Basecamp®

airbnb

GROUPON®

“Learn Ruby and C. With Ruby you can write programs fast, with C you can write fast programs.”

–Henry Maddocks

“C++ is only efficient when it is running. It is not efficient when you have a deadline to finish a project and deliver working code.”

–Paul Lutus

Different Strengths
Different Uses

Both powerful, but in different ways

First Impressions

- I don’t have to compile my code?
- Where are the header files?
- Where are the semicolons?
- I don’t have to declare the data type?
- Where are the constructors and destructors?
- Duck type?!?

Little Things

Ruby	C++
`nil`	`NULL`
`this`	`self`
`.`	`->`
`<`	`:`
`require`	`#include`

Little Things (cont.)

- .any? .gsub!
- puts(“Hello World”) or puts “Hello World”
- 0 evaluates to true
- attributes / public member variables
- modules / namespaces

More Substantial Differences

- Can add and redefine methods and variables at runtime
- No explicit references: a variable is an automatically dereferenced name for an object
- No multiple inheritance (but mixins exist)
- Everything is an object
- Everything has a value (even if it’s nil)

Everything is an object

```
irb(main):004:0> 123.class
=> Fixnum
irb(main):005:0> 123.class.superclass
=> Integer
irb(main):006:0> 123.class.superclass.superclass
=> Numeric
irb(main):007:0> 123.class.superclass.superclass.superclass
=> Object
```

Everything has a value (even if it's nil)

```
irb(main):001:0> 3 + 5
=> 8
irb(main):002:0> if true then "thingy" end
=> "thingy"
irb(main):003:0> if false then "thingy" end
=> nil
irb(main):004:0> x = 3
=> 3
```

Everything has a value

```
x = 5
y = 10

z = if x > y
      true
    else
      false
    end

z #=> false
```

Making Life Easier

- irb is great
- Enumerable mixin
  - vector<T>::const_iterator iter vs .each
  - select, sort, include? map, inject/reduce
- Strings: trim, to_s, #{}, split, regex …

side by side code

Hello World

hello_world.c × | fizzbuzz.cpp × | reverse_strin × | Person.h × | Person.cpp × | person_exam × | substrings.c ×

```cpp
#include <iostream>

int main()
{
  std::cout << "Hello, World!\n";
}
```

C++

hello_world.rb × | reverse_string.rb × | substrings.rb × | fizzbuzz.rb × | Person.rb ×

```ruby
puts "Hello World"
```

Ruby

Line 1, Column 19 | Spaces: 2 | Ruby

FizzBuzz

hello_world. × | fizzbuzz.cp × | reverse_strir × | Person.h × | Person.cpp × | person_exar × | substrings.c ×

```cpp
#include <iostream>
using namespace std;

int main ()
{
  for(int i = 1; i <= 100; i++)
  {
    if(i % 3 == 0 && i % 5 == 0)
      cout << "FizzBuzz" << endl;
    else
    if(i % 3 == 0)
      cout << "Fizz" << endl;
    else
    if(i % 5 == 0)
      cout << "Buzz" << endl;
    else cout << i << endl;
  }
  return 0;
}
```

C++

hello_world.rb × | fizzbuzz.rb × | reverse_string.rb × | substrings.rb × | Person.rb ×

```ruby
(1..100).each do |x|

  if x%3 == 0 && x%5 == 0
    puts 'FizzBuzz'
  elsif x%3 == 0
    puts 'Fizz'
  elsif x%5 == 0
    puts 'Buzz'
  else
    puts x
  end

end
```

Ruby

Line 1, Column 1 | Spaces: 2 | Ruby

String Reverse

hello_world. × | fizzbuzz.cp × | reverse_strir × | Person.h × | Person.cpp × | person_exar × | substrings.c ×

```cpp
#include <iostream>
using namespace std;

int main( )
{
    char str[80];

    cout << "Enter string: ";
    cin.getline(str, 80);

    int l; //Hold length of string
    for(l = 0; str[l] != '\0'; l++);    //finding length of string

    int temp;
    for(int i = 0, j = l - 1; i < l/2; i++, j--)
    {
        temp = str[i];
        str[i] = str[j];
        str[j] = temp;
    }

    cout << "Reverse string: " << str << endl;

    return 0;
}
```

C++

hello_world.rb × | fizzbuzz.rb × | reverse_string.rb × | substrings.rb × | Person.rb ×

```ruby
print "Enter string: "
str = gets.reverse!
puts "Reverse string: #{str}"

```

Ruby

Line 1, Column 1 | Spaces: 2 | Ruby

Substring Search

hello_world. × | fizzbuzz.cp × | reverse_strir × | substrings.c × | Person.h × | Person.cpp × | person_exar ×

```cpp
#include <iostream>
#include <string>

// returns count of non-overlapping occurrences of 'sub' in 'str'
int countSubstring(const std::string& str, const std::string& sub)
{
    if (sub.length() == 0) return 0;
    int count = 0;
    for (size_t offset = str.find(sub); offset != std::string::npos;
   offset = str.find(sub, offset + sub.length()))
    {
        ++count;
    }
    return count;
}


int main()
{
    std::cout << countSubstring("university ruby", "rub")    << '\n';
    std::cout << countSubstring("penpen is the best penguin!", "pen")        << '\

    return 0;
}
```

C++

hello_world.rb × | fizzbuzz.rb × | reverse_string.rb × | substrings.rb × | Person.rb ×

```ruby
def count_substrings(string, substring)
  string.scan(substring).length
end

puts count_substrings("university ruby", "rub")
puts count_substrings("penpen is the best penguin!", "pen")

```

Ruby

Line 1, Column 1 | Spaces: 4 | C++

Person Class

Person.h

```cpp
#include <string>
#include <vector>

using std::string;
using std::vector;

class Person
{

private:
  string firstName;
  string familyName;
  vector<string> emailAddresses;

public:
  Person(string firstName, string familyName);
  string getFullName();
  void addEmailAddress(string email);
  vector<string> getEmailAddresses();

};
```

C++

Person.cpp

```cpp
#include "Person.h"

Person::Person(string firstName, string familyName)
: firstName(firstName), familyName(familyName)
{}

string Person::getFullName()
{
  return firstName + " " + familyName;
}

void Person::addEmailAddress(string email)
{
  emailAddresses.push_back(email);
}

vector<string> Person::getEmailAddresses()
{
  return emailAddresses;
}
```

C++

person_example.cpp

```cpp
#include <iostream>
#include "Person.h"

using std::cout;
using std::endl;

void show(vector<string> v) {
  cout << "---" << endl;
  vector<string>::iterator iter;
  for (iter = v.begin(); iter != v.end(); iter++) {
    cout << *iter << endl;
  }
  cout << "---" << endl;
}

void usePerson() {
  Person p("Arthur","Dent");
  cout << "Person fullName: " << p.getFullName() << endl;
  p.addEmailAddress("dent@earth.com");
  p.addEmailAddress("dent@earth.co.uk");
  show(p.getEmailAddresses());
}

int main (int argc, char* const argv[]) {
  usePerson();
  return 0;
}
```

C++

Person.rb

```ruby
class Person
  def initialize(firstName,familyName)
    @firstName = firstName
    @familyName = familyName
    @emailAddresses = []
  end

  def getFullName
    @firstName + " " + @familyName
  end

  def addEmailAddress(address)
    @emailAddresses << address
  end

  def getEmailAddresses
    Array.new(@emailAddresses)
  end
end

person = Person.new("Arthur","Dent")
puts person.getFullName
person.addEmailAddress "arthur@earth.com"
emails = person.getEmailAddresses
puts emails.length
puts emails[0]
person.addEmailAddress "arthur@HoG.com"
emails = person.getEmailAddresses
puts emails.length
puts emails[0..-1]
```

Ruby

Line 1, Column 1 Spaces: 2 C++

Thanks to Dr. Graham Roberts for this example

Habits to Break

```
even_array = []
array.each do |i|
  even_array << i if i % 2 == 0
end
```

```
even_array = array.select{ |i| i % 2 == 0 }
```

Habits to Break

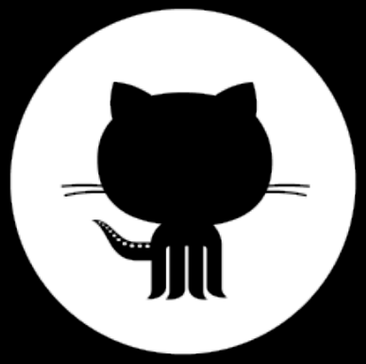
```
sum = 0
for i in 0..array.length-1
  sum += array[i]
end
```

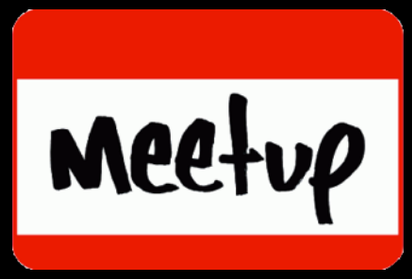
```
sum = 0
array.each do |a|
  sum += a
end

array.reduce(0) { |sum, i| sum + i }
array.reduce { |sum, i| sum + i }
array.reduce(0, :+)
array.reduce(:+)
```

Good Riddance

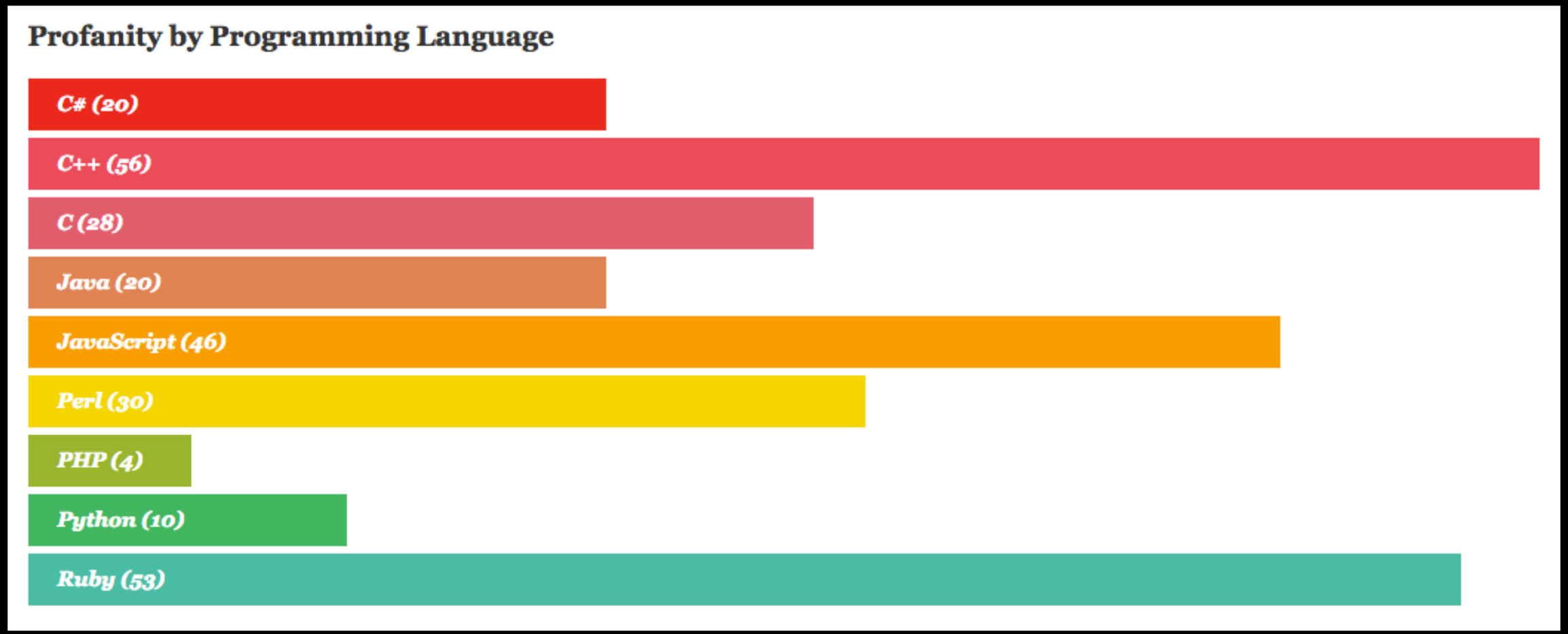
- linker errors
- writing getters and setters
- pointers … allocation / deallocation
- windows tithe day
- IDE?

Culture



My People



http://andrewvos.com/2011/02/21/amount-of-profanity-in-git-commit-messages-per-programming-language/

Final Thoughts

- C++ before Ruby
- Appreciate Ruby!
- The Ruby Way
- OOP

References / Further Reading

- **To Ruby From C and C++** https://www.ruby-lang.org/en/documentation/ruby-from-other-languages/to-ruby-from-c-and-cpp

- **The Road to Ruby from C++** http://www.devx.com/RubySpecialReport/Article/34497

- **ruby-talk thread** https://www.ruby-forum.com/topic/88609

- http://www.ibm.com/developerworks/linux/library/os-sixrubyfeatures/index.html
- http://blog.petersobot.com/rewriting-in-cpp-for-fun-speed-and-masochism
- http://blog.flatironschool.com/the-road-to-ruby-from-c
- http://chrismdp.com/2012/01/why-i-switched-from-ruby-back-to-c-plus-plus
- **Bjarne Stroustrup's list of C++ Applications** http://www.stroustrup.com/applications.html

Comparing Languages

- http://rosettacode.org
- http://web.archive.org/web/20100420080552/http://www.dmh2000.com/cjpr/
- http://www.dmoz.org/Computers/Programming/Languages/Comparison_and_Review/
- http://www0.cs.ucl.ac.uk/staff/G.Roberts/courses2005_6/1008/Slides3.pdf
- http://www.hammerprinciple.com/therighttool/items/ruby/c-2