Ruby - The way back to Productivity!

Michael Neumann

mneumann@ntecs.de

http://www.ntecs.de/blog

Table of Contents

1 XP, UML, Agile gurus about Ruby

1.0.1 Kent Beck

- *I always knew that one day Smalltalk would replace Java. I just didn't know it would be called Ruby.*

1.0.2 Martin Fowler

- *When I ask the question "do you think you're significantly more productive in Ruby rather than Java/C#", each time I've got a strong 'yes'.*
- *[…], names like the both the Prags, Justin Gehtland, Bruce Tate, David Geary et al should be enough to make Ruby worth looking at.*
- *I like the language a lot.*

1.0.3 Ron Jeffries

- *"In letzter Zeit arbeite ich mit Ruby und mag es sehr gerne."*
- *"[…] und ich hoffe, dass die Welt lernen wird, dass dynamisch typisierte Sprachen wie Ruby und Smalltalk produktiver sind als statisch typisierte."*

Kent Beck groups.google.de/group/comp.lang.ruby/msg/46ad5844760121bd (http://groups.google.de/group/comp.lang.ruby/msg/46ad5844760121bd)

Martin Fowler www.martinfowler.com/bliki/EvaluatingRuby.html (http://www.martinfowler.com/bliki/EvaluatingRuby.html) www.martin-fowler.com/bliki/MovingAwayFromXslt.html (http://www.martinfowler.com/bliki/MovingAwayFromXslt.html)

When I wrote the software for this Bliki (on a long flight) I did it in Ruby. […] My conclusion from this exercise was that using Ruby for XML transforms was much easier than using XSLT.

Jon Jeffries www.approximity.com/produktiver_programmieren/jeffries_de.html (http://www.approximity.com/produktiver_programmieren/jeffries_de.html)

Brian Marick www.pragmaticprogrammer.com/titles/bmsft/ (http://www.pragmaticprogrammer.com/titles/bmsft/) (3 books)

2 Try Ruby Online

tryruby.hobix.com/ (http://tryruby.hobix.com/)

3 RubyJS - Ruby to Javascript Compilation

The HTML:

src/rubyjs/index.html

```
<html>
  <body>
    <script language="javascript" src="hw.js"></script>
    <a href="#" onclick="main();">say hello</a>
    <div id="out"/>
  </body>
</html>
```

The "client-side" Ruby code:

src/rubyjs/hw.rb

```
require 'rwt/DOM'
require 'rwt/HTTPRequest'
require 'json'

class HelloWorld
  def self.main
    out = DOM.getElementById('out')
    HTTPRequest.asyncGet('/json') do |response|
      DOM.setInnerText(out, JSON.load(response).inspect)
    end
  end
end
```

The server-side Ruby code:

src/rubyjs/server.rb

```ruby
require 'mongrel'
require 'json'

class JsonHandler < Mongrel::HttpHandler
  def process(request, response)
    response.start(200) do |header, output|
      output << JSON.dump([{"Hello" => "rubyconf"}])
    end
  end
end
```

Test it:

localhost:3000/ (http://localhost:3000/)

4 Key aspects of Ruby

- Ruby was designed for *humans*
  - Readable
  - Consistent
- Lets you concentrate on the problem.
- Ruby is pragmatic.

5 Readability

Simple things like:

```
# count the 0‘s!
100000000  # vs.
100_000_000
```

Where do you put your parens in C/C++/Java?

```
if (...) {
  // code
}

if (...)
{
  // code
}

if (...)
  // code
```

In Ruby/Eiffel/Ada:

```
if condition
  # code
end

# Ruby
expr if condition
```

6 Variable Naming

Can you say wether variable `i` below is a *local variable*, a *global variable*, an *instance variable* or a *constant*?

```
// somewhere deep inside a Java/C++/C# method
if (i == 1)
// ...
```

In Ruby:

```
if i == 1     # (1) a local variable
if $i == 1    # (2) a global variable
if @i == 1    # (3) an instance variable
if I == 1     # (4) a constant
```

That is *soooo* ugly:

```
public class X {
  private int i;

  public X(_i) {
    i = _i;
  }

  public X(i) {
    this.i = i;
  }
}
```

7 Class and Method Naming

From object oriented analysis we know that:

- Classes = *nouns*

  ```
  Animal Account File Logger Observer Class
  ```

- Methods = *verbs*

  ```
  miau read new open do_this_or_that delete
  ```

In German and in Ruby nouns are written capitalized. In most language, including Ruby, verbs are written lowercase.

8 Readability by Conventions

"?", "!" and "=" can be part of the method name:

- `question?`

  Return value is either `true` or `false`.

```
if anArray.empty?
  ...
end

if File.exist?('/tmp/a')
  ...
end
```

- `dangerous!`

  Be careful! Mostly used to mark destructive updates.

```
str = "  hallo welt  "
str.strip   # => "hallo welt"
str         # => "  hallo welt  "

# destructive version: strip!
str.strip!
str         # => "hallo welt"
```

9 Readability by Conventions (2)

- `attribute=`

  A setter method (setBlah in Java)

```
class A
  def value
    @value
  end

  def value=(value)
    @value = value
  end
end

a = A.new
a.value = 5
a.value      # => 5

a.value=(6)
a.value      # => 6
```

10 Readability by Conventions (3)

- `[]`, `[]=`

  "Array" access.

```
a = ["a", "b", "c"]
a[0]       # => "a"
a[1] = 4
a[1]       # => 4
```

  Again: plain methods!

```
class MyArray
  def [](idx)
    # ...
  end

  def []=(idx, value)
    # ...
  end
end

m = MyArray.new
m[0]
m["test"]
m[MyArray.new] = 5
m[if true then x else y end] = 1.4
```

11 Readability by Custom Control Structures

Even a child can understand:

```
1.upto(10) do |i|
  print i
end

5.downto(1) do |i|
  print i
end

10.times do
  print "hello world"
end

["a", "b", "c"].each do |e|
  print e
end
```

But would it understand this?

```
for (int i=1; i <= 10; i++) {
  System.out.println(i);
}

for (int i=5; i > 1; i--) {
  System.out.println(i);
}
```

Can *YOU* state that the second is correct (within 5 seconds)? I can't!

12 Evil for-loops

- They really hurt my brain (your too?)
- And introduce possible (off-by-one (http://www.google.com/search?hl=de&client=opera&rls=en&hs=ho6&q=off-by-one-error+linux&btnG=Suche&lr=)) errors.
- Evil `while` as well. They are not made for humans:

```
# what????
while a and not b
  # do something
end
```

- Instead use infinite loops with breaks:

```
loop do
  break if not a
  break if b
end
```

- `loop` is actually a method in Ruby:

```
def loop
  while true
    yield
  end
end
```

13 Build your own loops!

- DRY - Don’t Repeat Yourself! How many occurences of:

```
for (int i=0; i<arr.getLength(); i++) { /* ... arr[i] */ }
```

- Encapsulate instead!

```
class Array
  def each
    for i in 0 ... self.length
      yield self[i]
    end
  end
end

[1,2,3].each do |elem|
  # ...
end
```

- Need indices?

```
  def each_with_index
    for i in 0 ... self.length
      yield self[i], i
    end
  end

[1,2,3].each_with_index {|elem, i| ... }
```

14 Not just loops...

- Bad:

```
f = File.open('/tmp/test', 'w+')
f.write("test")
# ...
f.close
```

  Exception can lead to a file not being closed (=> running out of files).

- Ugly:

```
// do you remember how to open a file in Java?
f = ...

try {
  // do something with "f"
}
finally {
  f.close();
}
```

- Why Ugly?

15 Not just loops (2)

- Encapsulate resposibility for "freeing" resources locally!!!
- Finish what you start!
- Good (DRY):

```
class File
  def File.open_with_block(name, mode)
    f = File.open(name, mode)
    begin
      yield f
    ensure
      f.close
    end
  end
end

File.open_with_block('/tmp/test', 'w+') do |f|
  # ...
end
```

- Luckily this comes with Ruby by default:

```
File.open('/tmp/test', 'w+') do |f|
  # ...
end
```

16 Understandability

- Principle of Least Surprise (actually that of the languages' creator ;-).
- Documentation Matters (http://www.ruby-doc.org/core/)!!! Java's docs (http://java.sun.com/j2se/1.4.2/docs/api/java/io/FileReader.html) sucks, because no examples!
- If something doesn't work, you're pissed off in Java! Inherent complexity. For example: "Umlauts" ;-)
- In Ruby, it once happen that someone answered faster that speed of light ;-)

---

www.ruby-doc.org/core/ (http://www.ruby-doc.org/core/) www.ruby-doc.org/stdlib/ (http://www.ruby-doc.org/stdlib/) ferret.davebalmain.com/api/ (http://ferret.davebalmain.com/api/) api.rubyonrails.org/ (http://api.rubyonrails.org/)

17 Consistency

In Ruby everything is

- an *object*
- an *expression*
- a *method* call (99.999%)

18 Consistency (2)

Classes are regular objects. There are no constructors, only methods!

src/classes_are_objects.rb

```
class Test
end

# instanciate a new "Test"
obj = Test.new

# treat classes like objects
[Test, String, Array].each do |klass|
  p klass.new
end

# create a class "by hand"
klass = Class.new
p klass.class      # => Class
```

Ruby has meta-classes. But I don’t want to go into detail :)

19 Consistency (Truth)

Truth of values:

```
# we are in Python ;)
if 0:
  print "will never be executed"

if "":
  print "will never be executed"

if []:
  print "will never be executed"
```

The *absolute Truth* of values:

```
# we are back in Ruby ;)
if 0.zero?
  print "will never be executed"

if "".empty?
  print "will never be executed"

if [].empty?
  print "will never be executed"
```

- Only `nil` and `false` evaluate to `false`. Everything else is `true` in Ruby.

20 OO with Ruby

Single-Inheritance + Mixins.

src/seq.rb

```ruby
# Enumerable comes by default with Ruby
module Enumerable
  def select
    arr = []
    each {|e| arr << e if yield e }
    return arr
  end

  def collect
    arr = []
    each {|e| arr << yield(e) }
    return arr
  end
end

class Sequence
  include Enumerable   # defines collect, select, reject...

  def initialize(from=1, to=nil)
    @from, @to = from, to
  end

  def each
    i = @from
    loop {
      yield i
      i += 1
      break if @to and i > @to
    }
```

```ruby
  end
end

p Sequence.new(1,10).select{|i| i%2 == 0}.collect{|i| i**2}
```

21 Examples...

22 Parsing Wikipedia Metadata

Extract Metadata from Wikipedia:

src/snippets/wikipedia_meta.rb

```
require 'hpricot'   # HTML parser
require 'open-uri'  # treat any kind of URI like a file

NAME = readline.split.map{|i| i.capitalize}.join('_')
URL = "http://de.wikipedia.org/wiki/#{NAME}"

doc = Hpricot(open(URL).read)
(doc / "td.metadata-label").each {|e|
  puts "#{e.inner_text.ljust(20)} #{e.next_sibling.inner_text}"
}
```

23 Generating PDF

Easy:

src/snippets/pdf_writer.rb

```
require 'pdf/writer'

pdf = PDF::Writer.new
pdf.select_font 'Helvetica'
pdf.fill_color Color::RGB::Red
pdf.add_text(200, 400, 'Hallo Ruby!', 72, 45)
STDOUT << pdf.render
```

www.artima.com/rubycs/articles/pdf_writer.html (http://www.artima.com/rubycs/articles/pdf_writer.html)

24 Distributed Ruby

Like Java's RMI but only the fraction of the code size.

The server:

src/drb/drb_server.rb

```
require 'drb'

# The object that we want to be accessible from the client
ExposedObj = [1,2,3, [4,5,6], "test", proc { p "." }, proc {|i| i + 1}]

DRb.start_service('druby://localhost:3456', ExposedObj)
puts "Server started"
DRb.thread.join
```

The client:

src/drb/drb_client.rb

```
require 'drb'

DRb.start_service

# A reference to the remote object (which resides on the server)
$remote_obj = DRbObject.new(nil, 'druby://localhost:3456')

p $remote_obj.last.call(1) # => 1 + 1 = 2

# Enter interactive Ruby to play a bit more...
require 'irb'; IRB.start
```

25 Great Ruby Tools

Interactive Ruby Interpreter

```
irb
```

Ruby information - Query documentation

```
ri Array#join
ri Hash.new
```

Rubygems Package manager

```
gem list
gem install rails
```

Generate your own "gem" and upload it to rubyforge.org/ (http://rubyforge.org/). Everyone can now install it using `gem`.

Ruby make

```
rake
rake db:stop
```

Ruby documentation generator (like javadoc)

```
rdoc
```

26 Ever wrote your own Webserver?

- Multi-threaded
- In Five minutes? 14 LoC?

Here we go:

src/webserver/webserver.rb

```
require 'socket'
server = TCPServer.new('localhost', 9000)
puts "Listening on localhost:9000"
loop do
  Thread.new(server.accept) {|socket|
    begin
      if socket.gets =~ /^GET \/([^\s]*) HTTP/
        socket << "HTTP/1.0 200 OK\r\n\r\n" + File.read($1)
      end
    ensure
      socket.close
    end
  }
end
```

Lets see if it works:

localhost:9000/index.html (http://localhost:9000/index.html) localhost:9000/webserver.rb (http://localhost:9000/webserver.rb) localhost:9000//etc/rc.conf (http://localhost:9000//etc/rc.conf) ;-)

27 Automating Google

What are the top results for a Google query?

src/snippets/query_google.rb

```
require 'hpricot'   # HTML parser
require 'open-uri'  # treat any kind of URI like a file

doc = Hpricot(open("http://www.google.de/search?hl=de&q=#{readline}").read)
(doc / '.g .l').each {|e|
  puts "-----------------------------------------------"
  puts e.inner_text, e[:href]
}
```

28 Domain Specific Languages

Sequel, a DSL for generating SQL and accessing databases.

src/sequel/ex.rb

```
require 'sequel'

# Sqlite memory database
DB = Sequel.open 'sqlite:/'

# Table items
items = DB[:items]

p items.sql
# SELECT * FROM items

p items.filter {:category == 'ruby'}.order(:price).sql
# SELECT * FROM items WHERE (category = 'ruby') ORDER BY price

p items.filter {:price.in(100..200) && :category.nil?}.sql
# SELECT * FROM items WHERE ((price >= 100 AND price <= 200) AND
#                           (category IS NULL))
```

More infos here:

code.google.com/p/ruby-sequel/ (http://code.google.com/p/ruby-sequel/)

29 Interfacing with C is easy

Write the C interface:

src/c_extension/kit.c

```
#include "ruby.h"

VALUE Kit_rocks(VALUE self)  // method Kit#rocks
{
  return rb_str_new2("Ruby rocks!");
}

void Init_kit()
{
  VALUE cKit = rb_define_class("Kit", rb_cObject);
  rb_define_method(cKit, "rocks", Kit_rocks, 0);
}
```

Create the Makefile and compile it:

```
ruby -rmkmf -e "create_makefile 'kit'"
make
```

Ready to go:

```
require 'kit'
p Kit.new.rocks
```

30 Mixing Ruby and C++

Cplus2Ruby allows neatless integration of C++ code within Ruby.

- Calling C++ methods from Ruby and vice versa.
- Handles Garbage Collection, Property initialization and Compilation automatically.

src/cplus2ruby/example.rb

```
require 'cplus2ruby'

class NeuralEntity < Cplus2Ruby_
  property :id
end

class Neuron < NeuralEntity
  property :potential,       'float'
  property :pre_synapses

  def initialize
    self.pre_synapses = []
  end

  method_c :stimulate, {at: 'float', weight: 'float'}, %{
    // This is C++ Code
    @potential += at * weight;
  }
end

Cplus2Ruby.compile_and_load('inspire')
n = Neuron.new
n.potential = 1.0
n.stimulate(1.0, 2.0)
p n.potential # => 3.0
```

31 Calculating PI

Is Ruby really slow?

src/pi.rb

```
def calc_pi
  q, r, t, k = 1, 0, 1, 1
  loop do
    n = (3*q+r) / t
    if (4*q+r) / t == n then
      yield n
      q, r, t, k = 10*q, 10*(r-n*t), t, k
    else
      q, r, t, k = q*k, q*(4*k+2)+r*(2*k+1), t*(2*k+1), k+1
    end
  end
end

calc_pi {|n| print n; $stdout.flush }
```

Keep in mind that every "/", "*", "+" or "==" is a method call that has to look up the method in a hash table.

32 Julia Sets

```
ruby19 -rcomplex -e'c,m,w,h=Complex(-0.75,0.134),50,400,200;\
  puts"P6\n#{w} #{h}\n255";(0...h).each{|j|(0...w).each{|i|n,\
  z=0,Complex(0.9*i/w,0.9*j/h);while n<=m&&(z-c).abs<=2;z=z*z+c;n+=1\
  end;print [20+n*10,0,rand*99].pack("C*")}}'|display
```

33 Interpreter

- Ruby 0.0.0 - 1.8.x: MRI - Matz Ruby Interpreter)
- Ruby 1.9: "YARV" Bytecode
- Rubinius www.rubini.us/ (http://www.rubini.us/) (EngineYard)
- IronRuby: www.ironruby.net/ (http://www.ironruby.net/) (Microsoft)
- JRuby: jruby.codehaus.org/ (http://jruby.codehaus.org/) (Sun)

say hello