Corey Haines, Journeyman Developer

Ruby

It really is Love

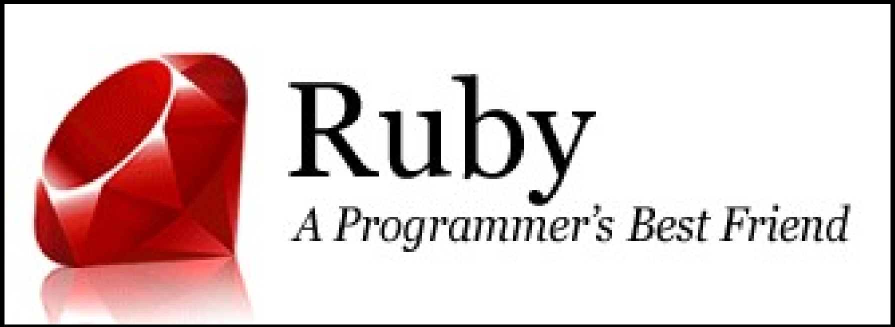
Thanks to RubySpec

Not About

IronRuby.net

(will have some links at the end, though)

About



Not About

Convincing you to use Ruby

About

Things I love in Ruby

duck-typing

open classes

(almost) everything is executable ruby code

Should You Learn Ruby?

Me

Corey Haines

Journeyman Developer

C# Background

'What would happen if?'

On to Ruby

“...trying to make Ruby natural, not simple.”

- Yukihiro Matsumoto

“Matz”

image courtesy Jim Lindley on flickr

Qualities

Simple Syntax

Object-Oriented

Duck-typing

Blocks

Open Classes

Initializers

```
arr = [1, 6, 2, 3, 5]

arr2 = [“Element”, “Another one”]

lookup = { :ruby => “love”,
           :c_sharp => “good too”,
           :dot_net => “super-ific” }
```

Classes

```
class Rectangle
  attr_accessor :width, :height

  def initialize(width, height)
    self.width = width
    self.height = height
  end

  def area
    width * height
  end
end
```

```
>> r = Rectangle.new(5, 20)
=> #<Rectangle:0x3691d8 @width=5,
@height=20>
>> r.area
=> 100
```

Qualities

Simple Syntax
Object-Oriented
Duck-typing
Blocks
Open Classes

Everything is an object

```
>> 9.succ
=> 10
>> 9.nil?
=> false
>> 9.between? 8, 10
=> true
>> 9.class
=> Fixnum
```

Everything is an object

```
>> 9.succc
=> 10
>> 9.nil?
=> false
>> 9.between? 8, 10
=> true
>> 9.class
=> Fixnum
>> Fixnum.class
=> Class
```

Wha??

Ever written this?

```
public void MakeItQuack<T>(T quacker)
where T : ICanQuack
{
    quacker.Quack();
}
```

Really Want

```
public void MakeItQuack<T>(T quacker)
where T can quack
{
    quacker.quack();
}
```

Qualities

Simple Syntax
Object-Oriented
Duck-typing
Blocks
Open Classes

Duck-typing

Walks like a duck

Quacks like a duck

Must be a duck?

Well, No

But, we can interact with it like a duck!

And Pretend!

```
def make_it_quack(quacker)
  quacker.quack();
end
```

Type != Class

Behavior/Interaction-Orientation

Qualities

Simple Syntax
Object-Oriented
Duck-typing
Blocks
Open Classes

Blocks

```
a = [5, 7, 10, 24]

a.each do |num|
  puts num
end

b = a.map do |num|
  num * 2
end

puts b.inspect
```

Accepting Blocks

```
def five_times
  yield 1
  yield 2
  yield 3
  yield 4
  yield 5
end

five_times do |num|
  puts num
end
```

Accepting Blocks

```
def five_times(&block)
  block.call(1)
  block.call(2)
  block.call(3)
  block.call(4)
  block.call(5)
end

five_times do |num|
  puts num
end
```

Qualities

Simple Syntax
Object-Oriented
Duck-typing
Blocks
Open Classes

Open Classes

Say goodbye to sealed/virtual/override/argh

```
def convert(to_convert)
    return nil if to_convert.nil?
    return to_convert if to_convert.empty?
    do_conversion(to_convert)
end
```

As you wish

```
def NilClass
    def empty?
        true
    end
end
```

```
def convert(to_convert)
    return to_convert if to_convert.empty?
    do_conversion(to_convert)
end
```

More Readable?

```
current_time = 17
```

```
current_time = 5.pm
```

```
class Fixnum
  def pm
    self + 12
  end
end
```

Type != Class (redux)

a = “coreyhaines@gmail.com;me@coreyhaines.com”

Type != Class (redux redux)

```
a = “coreyhaines@gmail.com;me@coreyhaines.com”
a.extend(EmailAddressList)

puts a.email_addresses.inspect

a.each_address do |address|
    Mailer.send_email_to(address)
end
```

Remember!

With great power comes great responsibility

Qualities

Simple Syntax
Object-Oriented
Duck-typing
Blocks
Open Classes

Awesomeness!!

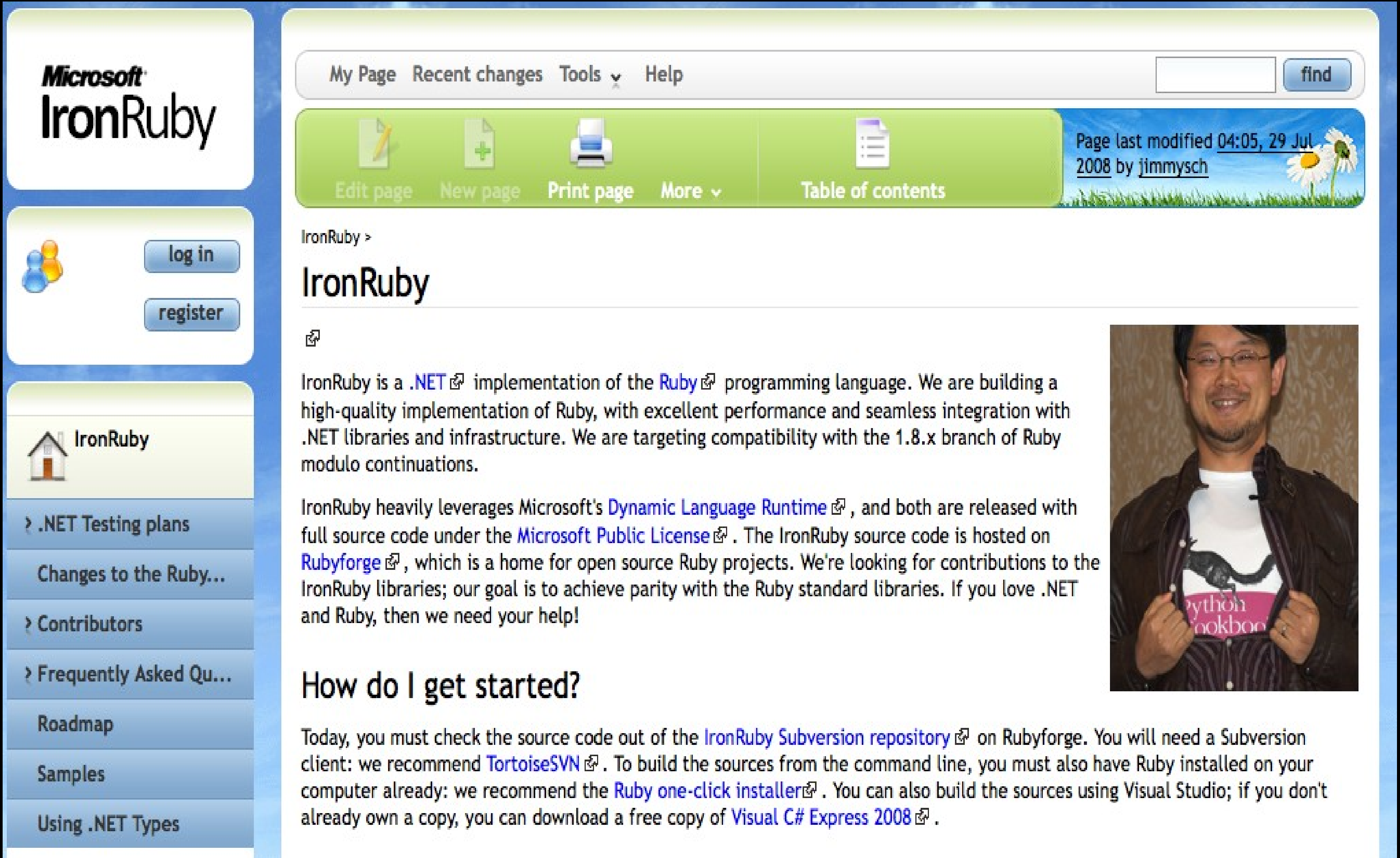
Examples

Builder

Examples

ActiveRecord

IronRuby.net



Page last modified 04:05, 29 Jul 2008 by jimmysch

IronRuby >

IronRuby

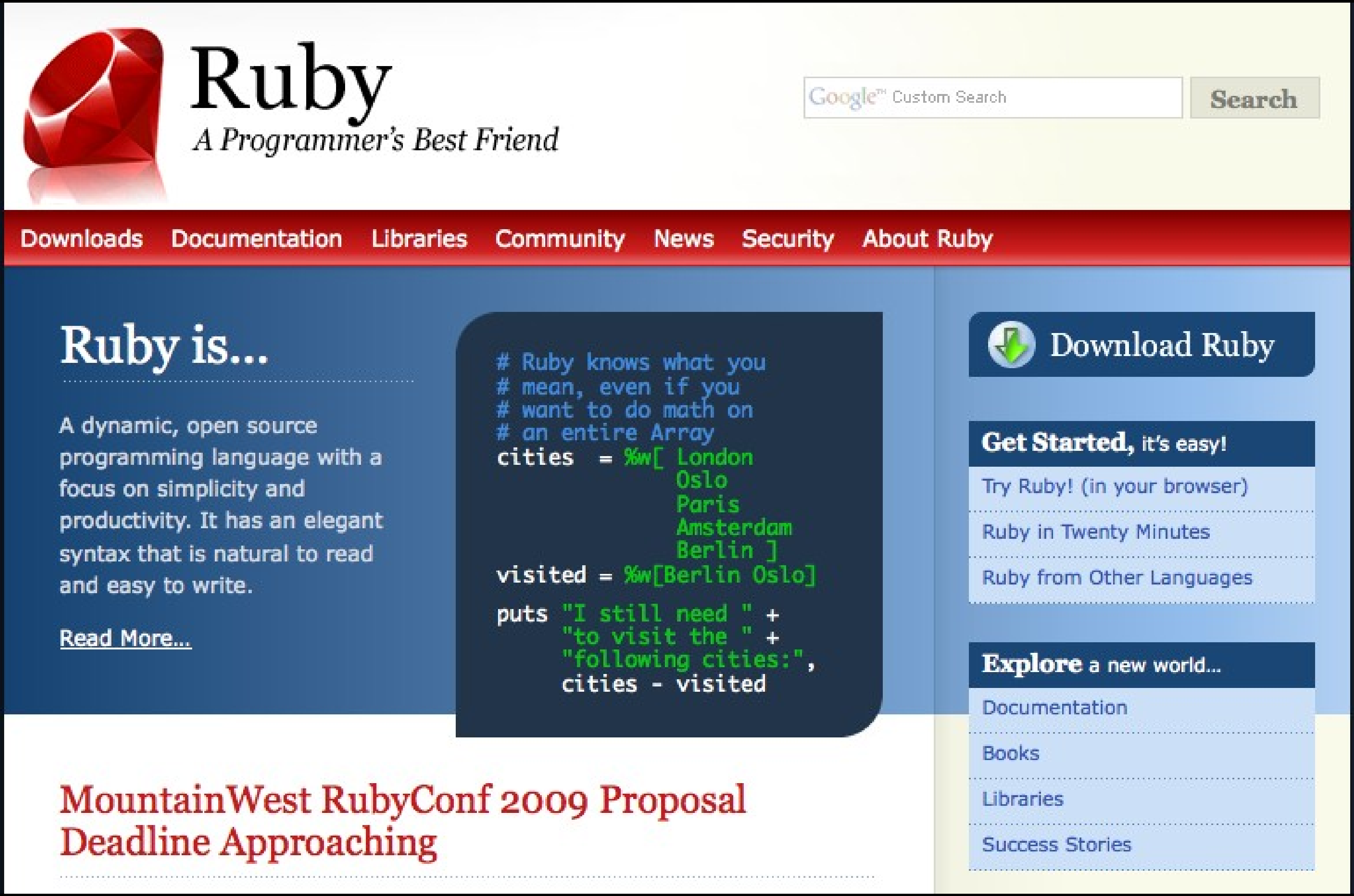
IronRuby is a .NET implementation of the Ruby programming language. We are building a high-quality implementation of Ruby, with excellent performance and seamless integration with .NET libraries and infrastructure. We are targeting compatibility with the 1.8.x branch of Ruby modulo continuations.

IronRuby heavily leverages Microsoft's Dynamic Language Runtime, and both are released with full source code under the Microsoft Public License. The IronRuby source code is hosted on Rubyforge, which is a home for open source Ruby projects. We're looking for contributions to the IronRuby libraries; our goal is to achieve parity with the Ruby standard libraries. If you love .NET and Ruby, then we need your help!

How do I get started?

Today, you must check the source code out of the IronRuby Subversion repository on Rubyforge. You will need a Subversion client: we recommend TortoiseSVN. To build the sources from the command line, you must also have Ruby installed on your computer already: we recommend the Ruby one-click installer. You can also build the sources using Visual Studio; if you don't already own a copy, you can download a free copy of Visual C# Express 2008.

RubyLang.org



Ruby is...

A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

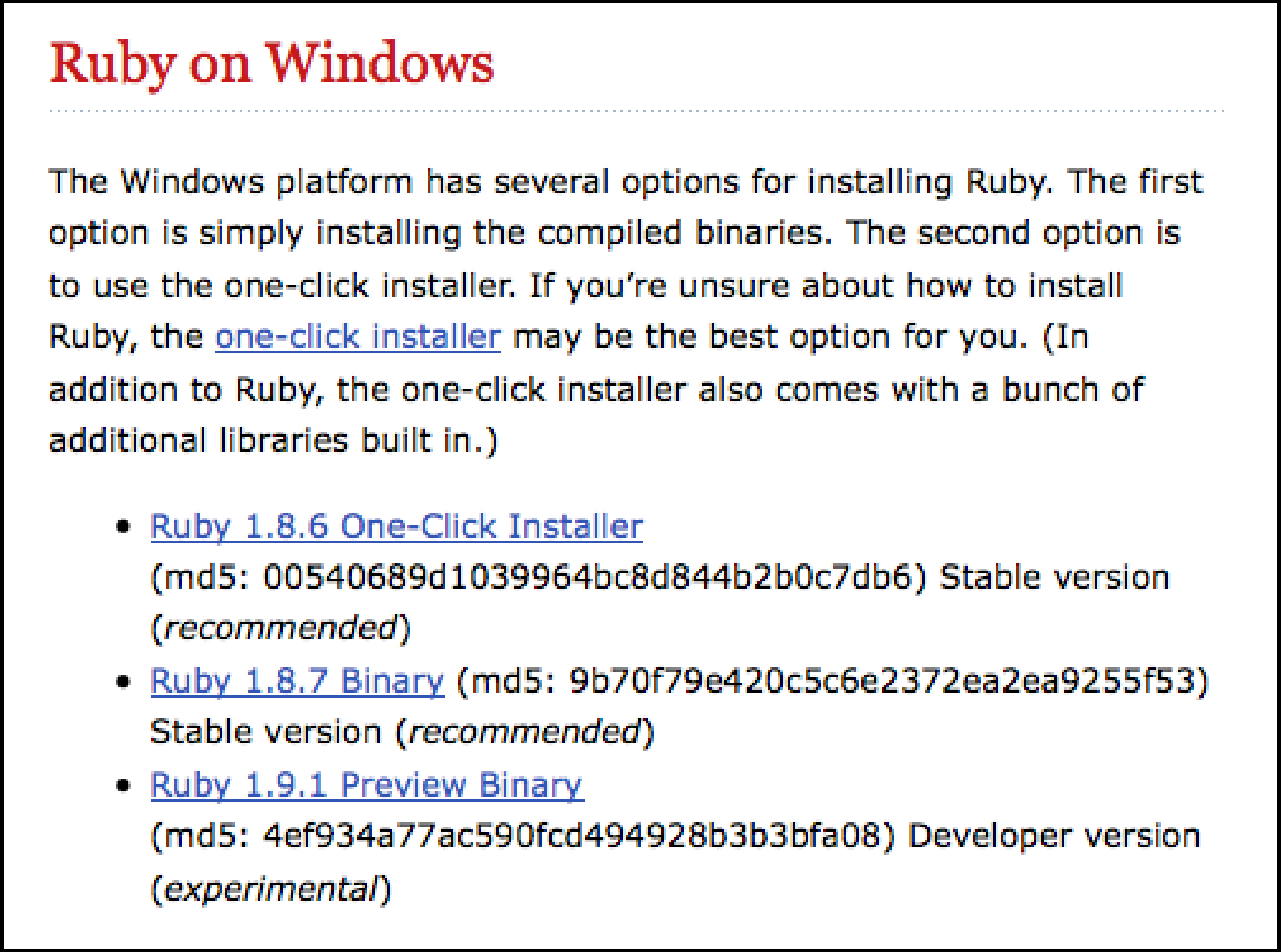
Read More...

```
# Ruby knows what you
# mean, even if you
# want to do math on
# an entire Array
cities  = %w[ London
              Oslo
              Paris
              Amsterdam
              Berlin ]
visited = %w[Berlin Oslo]

puts "I still need " +
     "to visit the " +
     "following cities:",
     cities - visited
```

MountainWest RubyConf 2009 Proposal Deadline Approaching

Windows? No Problem!



Ruby on Windows

The Windows platform has several options for installing Ruby. The first option is simply installing the compiled binaries. The second option is to use the one-click installer. If you're unsure about how to install Ruby, the one-click installer may be the best option for you. (In addition to Ruby, the one-click installer also comes with a bunch of additional libraries built in.)

- Ruby 1.8.6 One-Click Installer (md5: 00540689d1039964bc8d844b2b0c7db6) Stable version (*recommended*)
- Ruby 1.8.7 Binary (md5: 9b70f79e420c5c6e2372ea2ea9255f53) Stable version (*recommended*)
- Ruby 1.9.1 Preview Binary (md5: 4ef934a77ac590fcd494928b3b3bfa08) Developer version (*experimental*)

Corey Haines
Journeyman Developer

email: coreyhaines@gmail.com
twitter: @coreyhaines
blog(s): www.coreyhaines.com
programmingtour.blogspot.com