

What is the difference between Ruby 1.8 and Ruby 1.9

Asked 16 years, 4 months ago Modified 10 years, 8 months ago

Viewed 43k times



103



I'm not clear on the differences between the "current" version of Ruby (1.8) and the "new" version (1.9). Is there an "easy" or a "simple" explanation of the differences and why it is so different?

ruby

ruby-1.9

ruby-1.8

Share

Improve this question

Follow

edited Nov 9, 2010 at 14:09



Josh Lee

177k ● 39 ● 276 ● 280

asked Aug 22, 2008 at 1:32



salt.racer

22.3k ● 14 ● 45 ● 51

- 1 I assume you mean Ruby 1.8.6 - Ruby 1.8.7 has a lot of library constructs from 1.9. – [Andrew Grimm](#) May 3, 2010 at 7:44

Any more I consider the 1.8 versions the "old" versions, and 1.9.2+ the "current" ones. I only use 1.8.7 for compatibility checks but develop in 1.9.2. – [the Tin Man](#) Nov 9, 2010 at 16:25

5 @Telemachus: The link's broken. – [Andrew Grimm](#) Sep 7, 2011 at 2:46

1 @Telemachus, Andrew Grimm -- this archive.org link works -- web.archive.org/web/20090423003136/http://eigenclass.org/... – [J.Merrill](#) Feb 9, 2013 at 3:31

4 Answers

Sorted by:

Highest score (default)



Sam Ruby has a [cool slideshow that outline the differences](#).

168



In the interest of bringing this information inline for easier reference, and in case the link goes dead in the abstract future, here's an overview of Sam's slides. The slideshow is less overwhelming to review, but having it all laid out in a list like this is also helpful.



Ruby 1.9 - Major Features

- Performance
- Threads/Fibers
- Encoding/Unicode
- gems is (mostly) built-in now
- if statements do not introduce scope in Ruby.

What's changed?

Single character strings.

Ruby 1.9

```
irb(main):001:0> ?c  
=> "c"
```

Ruby 1.8.6

```
irb(main):001:0> ?c  
=> 99
```

String index.

Ruby 1.9

```
irb(main):001:0> "cat"[1]  
=> "a"
```

Ruby 1.8.6

```
irb(main):001:0> "cat"[1]  
=> 97
```

{"a","b"} No Longer Supported

Ruby 1.9

```
irb(main):002:0> {1,2}
```

```
SyntaxError: (irb):2: syntax error, unexpected ',', ex
```

Ruby 1.8.6

```
irb(main):001:0> {1,2}  
=> {1=>2}
```

Action: Convert to {1 => 2}

`Array.to_s` Now Contains Punctuation

Ruby 1.9

```
irb(main):001:0> [1,2,3].to_s  
=> "[1, 2, 3]"
```

Ruby 1.8.6

```
irb(main):001:0> [1,2,3].to_s  
=> "123"
```

Action: Use `.join` instead

Colon No Longer Valid In When Statements

Ruby 1.9

```
irb(main):001:0> case 'a'; when /\w/: puts 'word'; end  
SyntaxError: (irb):1: syntax error, unexpected ':',
```

```
expecting keyword_then or ',' or ';' or '\n'
```

Ruby 1.8.6

```
irb(main):001:0> case 'a'; when /\w/: puts 'word'; end  
word
```

Action: Use semicolon, then, or newline

Block Variables Now Shadow Local Variables

Ruby 1.9

```
irb(main):001:0> i=0; [1,2,3].each {|i|}; i  
=> 0  
irb(main):002:0> i=0; for i in [1,2,3]; end; i  
=> 3
```

Ruby 1.8.6

```
irb(main):001:0> i=0; [1,2,3].each {|i|}; i  
=> 3
```

Hash#index **Deprecated**

Ruby 1.9

```
irb(main):001:0> {1=>2}.index(2)  
(irb):18: warning: Hash#index is deprecated; use Hash#[]  
=> 1
```

```
irb(main):002:0> {1=>2}.key(2)
=> 1
```

Ruby 1.8.6

```
irb(main):001:0> {1=>2}.index(2)
=> 1
```

Action: Use Hash.key

`Fixnum.to_sym` **Now Gone**

Ruby 1.9

```
irb(main):001:0> 5.to_sym
NoMethodError: undefined method 'to_sym' for 5:Fixnum
```

Ruby 1.8.6

```
irb(main):001:0> 5.to_sym
=> nil
```

(Cont'd) Ruby 1.9

```
# Find an argument value by name or index.
def [](index)
  lookup(index.to_sym)
end
```

svn.ruby-lang.org/repos/ruby/trunk/lib/rake.rb

Hash Keys Now Unordered

Ruby 1.9

```
irb(main):001:0> {:a=>"a", :c=>"c", :b=>"b"}  
=> {:a=>"a", :c=>"c", :b=>"b"}
```

Ruby 1.8.6

```
irb(main):001:0> {:a=>"a", :c=>"c", :b=>"b"}  
=> {:a=>"a", :b=>"b", :c=>"c"}
```

Order is insertion order

Stricter Unicode Regular Expressions

Ruby 1.9

```
irb(main):001:0> /\x80/u  
SyntaxError: (irb):2: invalid multibyte escape: /\x80/
```

Ruby 1.8.6

```
irb(main):001:0> /\x80/u  
=> /\x80/u
```

`tr` and `Regexp` Now Understand Unicode

Ruby 1.9

```
unicode(string).tr(CP1252_DIFFERENCES, UNICODE_EQUIVAL
  gsub(INVALID_XML_CHAR, REPLACEMENT_CHAR).
  gsub(XML_PREDEFINED) {|c| PREDEFINED[c.ord]}
```

pack **and** **unpack**

Ruby 1.8.6

```
def xchr(escape=true)
  n = XChar::CP1252[self] || self
  case n when *XChar::VALID
    XChar::PREDEFINED[n] or
    (n>128 ? n.chr : (escape ? "&##{n};" : [n].pack(
  else
    Builder::XChar::REPLACEMENT_CHAR
  end
end
unpack('U*').map {|n| n.xchr(escape)}.join
```

BasicObject **More Brutal Than** **BlankSlate**

Ruby 1.9

```
irb(main):001:0> class C < BasicObject; def f; Math::P
NameError: uninitialized constant C::Math
```

Ruby 1.8.6

```
irb(main):001:0> require 'blankslate'
=> true
irb(main):002:0> class C < BlankSlate; def f; Math::PI
=> 3.14159265358979
```


Action: Use ::Math::PI

~~Delegation Changes~~

~~Ruby 1.9~~

```
irb(main):002:0> class C < SimpleDelegator; end  
=> nil  
irb(main):003:0> C.new('').class  
=> String
```

~~Ruby 1.8.6~~

```
irb(main):002:0> class C < SimpleDelegator; end  
=> nil  
irb(main):003:0> C.new('').class  
=> C  
irb(main):004:0>
```

~~[Defect 17700](#)~~

Use of \$KCODE Produces Warnings

Ruby 1.9

```
irb(main):004:1> $KCODE = 'UTF8'  
(irb):4: warning: variable $KCODE is no longer effective  
=> "UTF8"
```

Ruby 1.8.6

```
irb(main):001:0> $KCODE = 'UTF8'
```

```
=> "UTF8"
```

`instance_methods` Now an Array of Symbols

Ruby 1.9

```
irb(main):001:0> {}.methods.sort.last  
=> :zip
```

Ruby 1.8.6

```
irb(main):001:0> {}.methods.sort.last  
=> "zip"
```

Action: Replace `instance_methods.include?` with `method_defined?`

Source File Encoding

Basic

```
# coding: utf-8
```

Emacs

```
# -*- encoding: utf-8 -*-
```

Shebang

```
#!/usr/local/rubybook/bin/ruby  
# encoding: utf-8
```

Real Threading

- Race Conditions
 - Implicit Ordering Assumptions
 - Test Code
-

What's New?

Alternate Syntax for Symbol as Hash Keys

Ruby 1.9

```
{a: b}  
  
redirect_to action: show
```

Ruby 1.8.6

```
{:a => b}  
  
redirect_to :action => show
```

Block Local Variables

Ruby 1.9

```
[1,2].each {|value; t| t=value*value}
```

Inject Methods

Ruby 1.9

```
[1,2].inject(:+)
```

Ruby 1.8.6

```
[1,2].inject {|a,b| a+b}
```

`to_enum`

Ruby 1.9

```
short_enum = [1, 2, 3].to_enum  
long_enum = ('a'..'z').to_enum  
loop do  
  puts "#{short_enum.next} #{long_enum.next}"  
end
```

No block? Enum!

Ruby 1.9

```
e = [1, 2, 3].each
```

Lambda Shorthand

Ruby 1.9

```
p = -> a, b, c {a+b+c}  
puts p.(1, 2, 3)  
puts p[1, 2, 3]
```

Ruby 1.8.6

```
p = lambda {|a, b, c| a+b+c}  
puts p.call(1, 2, 3)
```

Complex Numbers

Ruby 1.9

```
Complex(3, 4) == 3 + 4.im
```

Decimal Is Still Not The Default

Ruby 1.9

```
irb(main):001:0> 1.2-1.1  
=> 0.09999999999999999
```

Regex “Properties”

Ruby 1.9

```
/\p{Space}/
```

Ruby 1.8.6

```
/[:space:]/
```

Splat in Middle

Ruby 1.9

```
def foo(first, *middle, last)
  (->a, *b, c {p a-c}).(*5.downto(1))
end
```

Fibers

Ruby 1.9

```
f = Fiber.new do
  a, b = 0, 1
  Fiber.yield a
  Fiber.yield b
  loop do
    a, b = b, a+b
    Fiber.yield b
  end
end
```

```
end
10.times {puts f.resume}
```

Break Values

Ruby 1.9

```
match =
  while line = gets
    next if line =~ /^#/
    break line if line.find('ruby')
  end
```

“Nested” Methods

Ruby 1.9


```
def toggle
  def toggle
    "subsequent times"
  end
  "first time"
end
```

HTH!

Share Improve this answer

Follow

edited Nov 21, 2013 at 20:52

 **Michael Stalker**
1,367 ● 9 ● 15

answered Aug 22, 2008 at 1:53



Tim Sullivan

16.9k ● 12 ● 75 ● 120

3 HTH == "Hope that helps". I had to look that up. Can you point me to an answer you provided, where you didn't hope that it would help? – [Chris Wesseling](#) Apr 2, 2013 at 14:08

Or did you mean Happy to help? Please edit accordingly. Or remove it, if your happiness/hopefulness counts for all your answers. ;-) – [Chris Wesseling](#) Apr 2, 2013 at 14:09 ✎

10 No, I'm going to leave it as-is. As a percentage of the total answer, it's minuscule and either interpretation is fine with me. Thanks. – [Tim Sullivan](#) Apr 2, 2013 at 15:15

According to the link you provided, the change in delegation was a bug and has been fixed. – [Kyle Strand](#) Aug 2, 2013 at 18:46

What about the inconsistency between "Range.include?" method? Ruby v1.8.7 gives a different result than v1.9 – [Lucas Pottersky](#) Sep 16, 2013 at 17:09



12

One huge difference would be the move from Matz's interpreter to [YARV](#), a bytecode virtual machine that helps significantly with performance.



Share Improve this answer

answered Aug 22, 2008 at 3:11

Follow



Sören Kuklau

19.9k ● 8 ● 55 ● 90





4

Many now recommend [The Ruby Programming Language](#) over the Pickaxe - more to the point, it has all the details of the 1.8/1.9 differences.



Share Improve this answer

answered Sep 15, 2009 at 9:31

Follow



[Dave Everitt](#)

17.8k ● 7 ● 68 ● 100



- 1 I like them both. The Pickaxe book discusses some things more fully but The Ruby Programming Language is good when you want to quickly scan for something. It's closer to a "nutshell" book. – [the Tin Man](#) Nov 9, 2010 at 16:26



1

Some more changes:

Returning a splat singleton array:



```
def function
  return *[1]
end
```



```
a=function
```



- ruby 1.9 : [1]
- ruby 1.8 : 1

array arguments

```
def function(array)
  array.each { |v| p v }
```

```
end  
function "1"
```

- ruby 1.8: "1"
- ruby 1.9: undefined method `each' for "1":String

Share Improve this answer

answered Apr 8, 2014 at 15:13

Follow



[Wim Yedema](#)

41 ● 3
