



# Introducing Ruby 2.5: A Full List of Additions and Updates



## Table of Contents


---

Introduction	3
New Feature in Hash: slice	6
New Feature in Hash: transform_keys	6
Keyword arguments on Struct.new	6
Pattern argument on Enumerable methods	8
Rescue on blocks	9
Top-level constant look-up is removed	10
New Feature in Pathname: glob	11
New Feature in Dir: children and each_child	11
New Feature on Kernel: yield_self	12
Backtrace and error message	12
pp.rb is automatically loaded	12
New Feature in String: delete_prefix and delete_suffix	13
New Feature in Array: append and prepend	14

## Table of Contents

---

New Feature in KeyError: receiver and key	15
New class: FrozenError	16
New Feature in ERB: result_with_hash	17
New Feature in SecureRandom: alphanumeric	18
New Feature in Set: to_s , ===	18
New Feature in Set: reset	19
New Feature in File: ltime	20
Rubygems updated to 2.7.3	21
Gemification of standard	12
ubygems.rb removed from stdlib	22
Global VM Lock enhancements	22
Ruby OpenSSL is upgraded to 2.1	22
Conclusion	23



After more than 20 years, Ruby continues to grow, thanks to a strong open source community and the fact that more businesses are deploying business-critical web applications built on Ruby. The most recent version, Ruby 2.5, was released on Christmas Day in 2017, and offers a number of new features, many of which are outlined in this e-book.

With every new release, Ruby continues to deliver the same core benefits that Yukihiro “Matz” Matsumoto envisioned in 1993 when he created Ruby. Ruby remains popular because it was the first truly object-oriented programming language. It also continues to be one of the most natural programming languages, placing human understanding above machine code, which is why more developers gravitate to Ruby.

## **Programmers continue to fall in love with Ruby for various reasons, including:**



Ruby's syntax is simple and clear. If the Ruby code is well-written, you can follow it just as you could the plot of any book. There aren't syntactical keystrokes, and because it is a natural language, anyone can pick up where a previous coder left off; the documentation is embedded in the code itself.



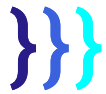
Everything is an object, which simplifies programming without compromising flexibility or intuitive coding.



Ruby rocks on Rails. For many web application developers, it's the Rails framework that gives Ruby its real value. Rails makes assumptions about what every developer needs to start building applications, which means there is less code to write and less repetition. Ruby lets you get started quickly and is perfect for agile web development.

Perhaps one of the biggest reasons programmers love Ruby is the open source developer community. The growing tribe of Ruby programmers are smart, intuitive, and generous. Any time you are stymied by a programming problem, you can reach out for help and you will always get a friendly response. This latest Ruby release includes innovations contributed by the growing community of Ruby developers. The great thing about open source is you get to mine the expertise of thousands of talented developers.

**This e-book was created to highlight the latest features in Ruby 2.5. We hope you find it insightful and useful**



## New Features

### 1. New Feature in Hash: slice

`slice` returns a hash containing the given keys.

```
{ a: 1, b: 2, c: 3, d: 4 }.slice(:a, :b)
=> {:a=>1, :b=>2}
```

### 2. New Feature in Hash: transform\_keys

`transform_keys` and `transform_keys!` returns a new hash with the keys converted using the block.

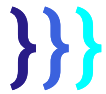
```
{ a: 1, b: 2, c: 3, d: 4 }.transform_keys{ |k| k.to_s }
=> {"a"=>1, "b"=>2, "c"=>3, "d"=>4}
```

### 3. Keyword arguments on Struct.new

You can add `keyword_init: true` to `Struct.new` to be able to use keyword arguments.

Before Ruby 2.5.0, you can't use keyword arguments.

```
Point = Struct.new(:x, :y)
Point.new(3, 5)
=> #<struct Point x=3, y=5>
```



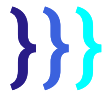
## New Features

You need to pass `x` first, then `y`. This is not a problem when you only have 2 attributes but it's a pain if you have a lot more attributes.

On Ruby 2.5.0, you can add `keyword_init: true`.

```
Point = Struct.new(:x, :y, keyword_init: true)
Point.new(x: 3, y: 5)
=> #<struct Point x=3, y=5>
```

If you use `keyword_init: true`, you can't use `Point.new(3, 5)`. Otherwise you'll get `ArgumentError (wrong number of arguments (given 2, expected 0))`.



## New Features

### 4. Pattern argument on Enumerable methods

You can pass a pattern on `all?`, `any?`, `none?`, and `one?` to make code more concise. If a pattern is passed instead of a block, the `===` method is used.

```
[1, 3.14, 2ri].all?(Numeric)
=> true
```

The code above is equivalent to

```
[1, 3.14, 2ri].all?{ |x| Numeric === x }
=> true
```

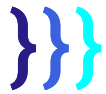
Another common use for this is passing regex.

```
%w{ant bear cat}.none?(/d/)
=> true
```

The code above is equivalent to

```
%w{ant bear cat}.none?{|x| /d/ === x }
=> true
```





## New Features

### 5. Rescue on blocks

You can add `rescue/else/ensure` are now allowed to be used directly with `do/end` blocks

Before Ruby 2.5.0, you need to use `begin` and `end` to be able to use rescue.

```
lambda do
  begin
    raise 'err'
  rescue
    $! # => #<RuntimeError: err>
  end
end.call
```

Starting with Ruby 2.5.0, you can remove `begin` and the corresponding `end` .

```
lambda do
  raise 'err'
rescue
  $! # => #<RuntimeError: err>
end.call
```

It looks cleaner and though Matz isn't a big fan of the syntax, he knows many developers like it.  
So, thank you Matz!



## New Features

### 6. Top-level constant look-up is removed

This code used to work

```
class Cloud; end  
class EngineYard; end  
EngineYard::Cloud
```

On Ruby 2.5, you'll get

```
NameError (uninitialized constant EngineYard::Cloud  
Did you mean?  Cloud)
```

On Ruby 2.4 and earlier, you'll get a warning but it will return the Cloud class.

```
warning: toplevel constant Cloud referenced by EngineYard::Cloud
```

If you encountered this issue before, you already have a workaround for this behavior. This a welcome change as it fixes some Rails autoload issues.

## New Features

### 7. New Feature in Pathname: glob

Pathname already has a `glob` class method. Ruby 2.5 adds a `glob` instance method.

```
require 'pathname'
dir = Pathname.new('.')
dir.glob('*rb')
=> [#<Pathname:a.rb>, #<Pathname:b.rb>, #<Pathname:c.rb>]
```

This code gets all the files ending on rb.

### 8. New Feature in Pathname: children and each\_child

`children` returns the files and directories without `"."` and `".."`.

```
Dir.children '.'
=> ["a.rb", "b.rb", "c.rb", "d.rb"]
```

You can go through each file without `"."` and `".."` using `each_child`

```
Dir.each_child('.').each do |f|
  puts f
end
dir.glob('*rb')
a.rb
b.rb
c.rb
d.rb
```

## New Features

### 9. New Feature on Kernel: `yield_self`

`yield_self` accepts a block and returns the result of the block.

Before Ruby 2.5.0, you need to use `begin` and `end` to be able to use `rescue`.

```
3.yield_self{ |x| x*x }  
=> 9
```

Ruby already has `tap` but it returns the object instead of the result of the block

```
3.tap{ |x| x*x }  
=> 3
```

### 10. Backtrace and error message

The backtrace and error message are printed in reverse order. The error message is bold and underlined.

### 11. New Feature in Pathname: `children` and `each_child`

You no longer have to write `require "pp"` to use it.



## New Features

### 12. New Feature in String: `delete_prefix` and `delete_suffix`

`delete_prefix` removes a string that appears at the beginning of another string.

```
str = "abcdef"
substr = "abc"

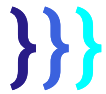
str.delete_prefix(substr)
=> "def"
```

`delete_suffix` removes a string that appears at the end of another string.

```
str = "abcdef"
substr = "def"

str.delete_suffix(substr)
=> "abc"
```

The equivalent bang methods, `delete_prefix!` and `delete_suffix!` also exist. They work the same way as described above and modifies the string with the result.



## New Features

### 13. New Feature in Array: `append` and `prepend`

`append` adds objects to the end of an array. `prepend` adds objects to the beginning. These are aliases to `push` and `unshift`, respectively. These methods are already in Active Support.

```
["a", "b", "c"].append("d")  
=> ["a", "b", "c", "d"]
```

```
["a", "b", "c"].append("d", "e")  
=> ["a", "b", "c", "d", "e"]
```

```
["x", "y", "z"].prepend("w")  
=> ["w", "x", "y", "z"]
```

```
["x", "y", "z"].prepend("v", "w")  
=> ["v", "w", "x", "y", "z"]
```



## New Features

### 14. New Feature in KeyError: receiver and key

You can rescue `KeyError` and call `receiver` and `key` .

```
begin
  h = {a: 1, b: 2}
  h.fetch(:z)
rescue KeyError => e
  p e.receiver #=> {:a=>1, :b=>2}
  p e.key      #=> :z
end
```

```
begin
  ENV.fetch("F00")
rescue KeyError => e
  p e.receiver #=> ENV
  p e.key      #=> "F00"
end
```



## New Features

### 15. New class: FrozenError

```
begin
  "a".freeze << "b"
rescue FrozenError
end
```

If you modify a frozen object, you'll now get a `FrozenError`, a subclass of `RuntimeError`. Before Ruby 2.5, you'll get a `RuntimeError`.





## New Features

### 16. New Feature in ERB: `result_with_hash`

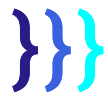
You can rescue `KeyError` and call `receiver` and `key` .

```
require 'erb'
h = {a: 1, b: 2}
erb = ERB.new(<<EOS)
A is <%= a %>
B is <%= b %>
EOS

puts erb.result_with_hash(a: 1, b: 2)
```

The output will be

```
A is 1
B is 2
```



## New Features

### 17. New Feature in ERB: `result_with_hash`

The alphanumeric method generates a random alphanumeric string.

```
require 'securerandom'
p SecureRandom.alphanumeric
=> "IRI4a4sXsVteD1sK"
```

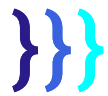
### 18. New Feature in Set: `to_s,===`

`to_s` is an alias of `inspect`

```
require 'set'
set = Set.new ["a", "b"]
=> #<Set: {"a", "b"}>
```

`===` is an alias of `include`

```
set === "b"
=> true
```



## New Features

### 19. New Feature in ERB: `result_with_hash`

**reset** resets the internal state of a state of a set after existing elements are modified.

```
require 'set'
set = Set.new([[]])
=> #<Set: {[[]]}>
set.first << 1
set === [1] # alias to include?
=> false
set.reset
=> #<Set: {[1]}>
set === [1]
=> true
```



## New Features

### 20. New Feature in ERB: `result_with_hash`

`ltime` sets the access and modification times of the link itself instead of its referent

In this example, `link.rb` is a symlink to `a.rb`.

```
File.mtime "a.rb"
=> 2018-02-15 11:43:56 +0800

File.lstat("link.rb").mtime
=> 2018-02-15 11:52:08 +0800

t = Time.now
=> 2018-02-15 11:53:17 +0800

File.lutime t, t, "link.rb"

File.mtime "a.rb" # doesn't change
=> 2018-02-15 11:43:56 +0800

File.lstat("link.rb").mtime
=> 2018-02-15 11:53:17 +0800
```



## General Updates

### 21. Rubygems updated to 2.7.3

Here's a list of major enhancements on Rubygems 2.7.3

- \* Update vendored bundler-1.16.0
- \* Use Bundler for Gem.use\_gemdeps
- \* Add command signin to gem CLI
- \* Add Logout feature to CLI





## General Updates

### 22. Gemification of standard

Here's a list of major enhancements on Rubygems 2.7.3

`cmath csv date dbm etc fcntl fiddle fileutils gdbm`

`ipaddr scanf sbdm stringio strscan webrick zlib`

### 23. `ubygems.rb` removed from `stdlib`

`ubygems.rb` isn't needed since Ruby 1.9 when Ruby started shipping with `zz` built-in.

### 24. Global VM Lock enhancements

These methods and many more now release the GVL:

`Dir.chdir`, `Dir.open`, `File.stat`, `File.exist?`, `File.rename`, `File.chown`

This allows other operations on other threads to proceed.

### 25. Ruby OpenSSL is upgraded to 2.1

Ruby Openssl 2.1 removed support for OpenSSL versions before 1.0.1 and LibreSSL versions before 2.5.



## Conclusion

Despite claims to the contrary, Ruby is far from dead. Although coding purists still balk at Ruby's ability to streamline development, most see its value as the language places usability over limitless customization. In fact, the TIOBE Index of programming languages ranked Ruby number 10, which is the same ranking as last year. The latest RedMonk programming rankings which compare the performance of different languages, put Ruby in fifth place, on a par with C# and C++.

The market value of Ruby on Rails continues to prove itself every day. Ruby on Rails offers a full stack language in a single framework that enables incredible productivity. To keep pace in today's business world, you have to be agile, and that means businesses need agile software development. Without Ruby on Rails, most of today's web applications and e-commerce platforms wouldn't exist.

Ruby on Rails continues to be a viable platform for commercial development. Thanks to a robust and active developer community, Ruby will continue to improve and expand, as shown in the release of Ruby 2.5. There continue to be new coding frameworks introduced to the market, but it's hard to beat the natural language programming of Ruby and the performance of Rails.







**[Learn how Engine Yard can be your Ruby DevOps team...](#)**

Engine Yard is the leading Ruby on Rails full-stack services company specializing in provisioning, managing, monitoring and controlling applications on AWS. Engine Yard specializes in deep partnerships with customers to provide mission-critical support and uptime. Engine Yard has thousands of customers across 58 countries and is headquartered in Austin, Texas.

## Try Out Engine Yard Today

[Start Your Free Trial](#)

### Products

-  DEVSPACES
-  ENGINE YARD
-  FOGBUGZ
-  CODEREVIEW
-  DEVCOACH
-  CODEFIX

### Contacts

info@devgraph.com  
410-220-8290  
401 Congress Avenue  
Austin, TX 78701