
Ruby

What is Ruby?

- Ruby is a programming language designed for simplicity and readability.
- It is widely used for web development, scripting, and general-purpose programming.
- Ruby's elegant syntax and developer-friendly philosophy make it popular among programmers.

Why was Ruby Created?


MY REACTION
When someone says Programming is Easy

- Ruby was created to prioritize developer productivity and enjoyment.
- It aims to make programming feel natural and intuitive.
- Ruby encourages writing code that is easy to understand and maintain.

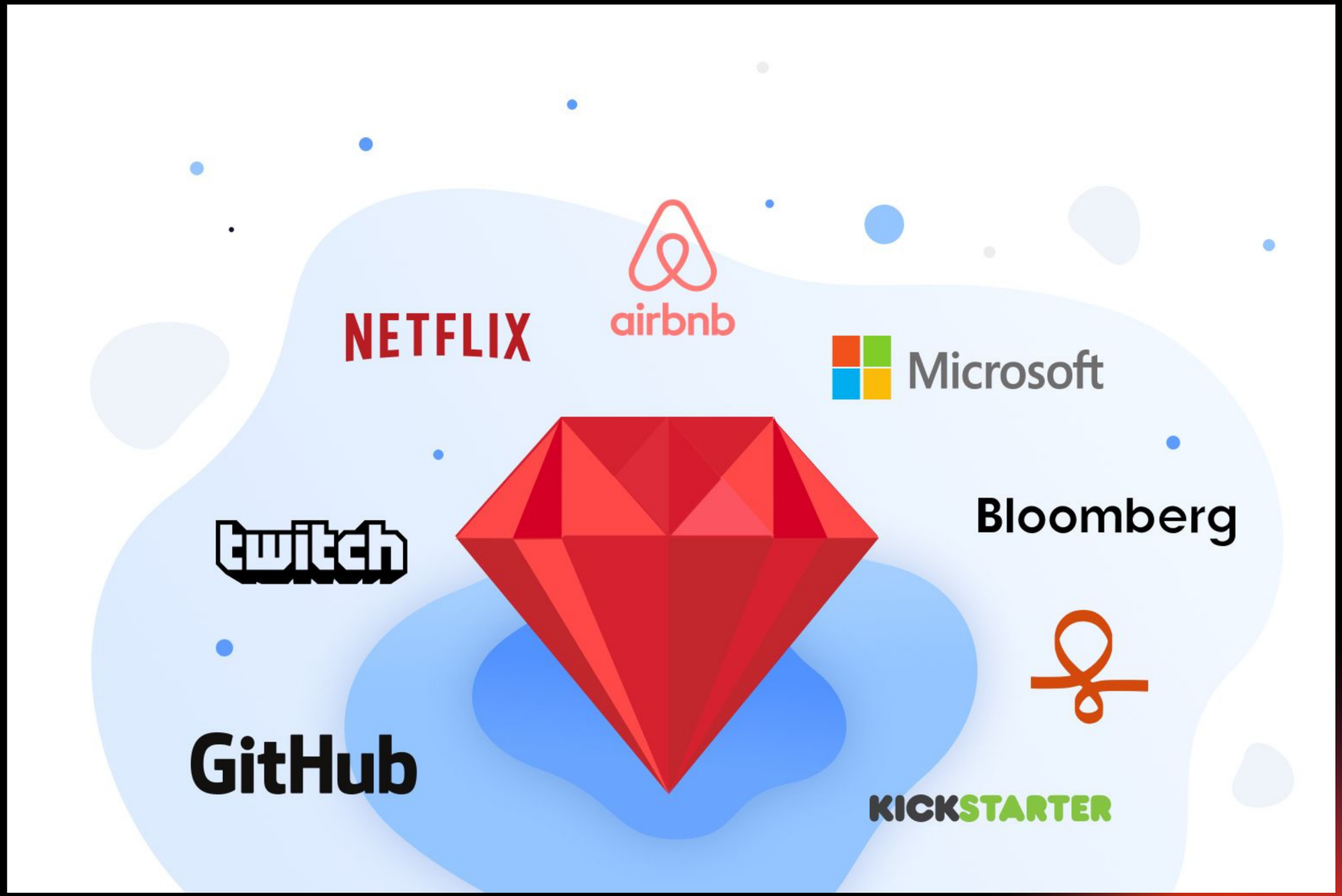
Who created Ruby?

- Ruby was created by Yukihiro Matsumoto, also known as Matz.
- Matz developed Ruby in the mid-1990s in Japan.
- He wanted to create a language that combined the best features of other languages and catered to programmer happiness.

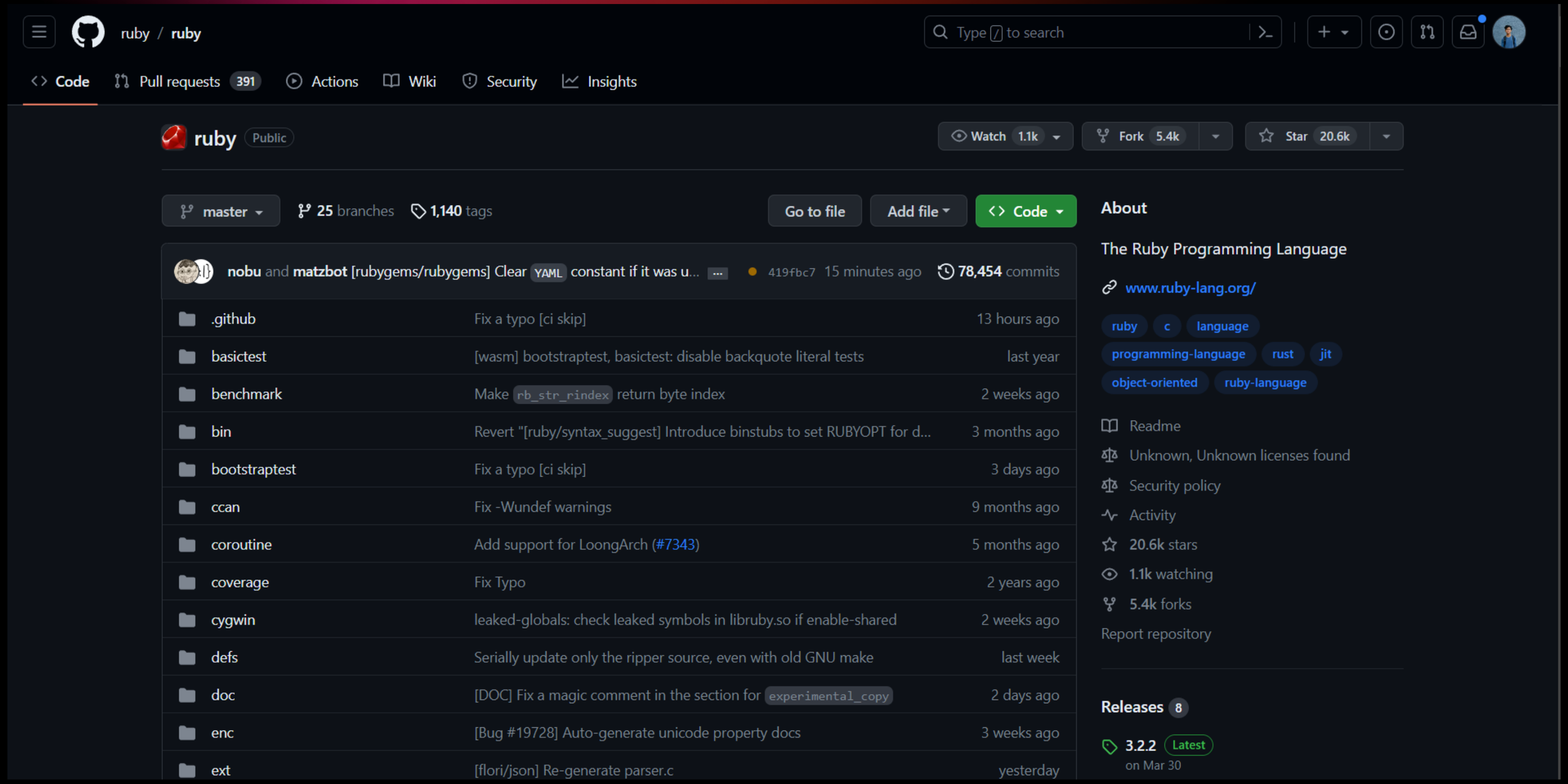

ASTON MARTIN
10
Gulf
HACKETT
LONDON
ASTON MARTIN
RACING
WEC
MICHELIN
Basecamp.com

Where is Ruby used?

- Ruby is used in a variety of applications, with web development being a prominent area.
- Ruby on Rails, a popular web framework built with Ruby, powers many websites and web applications.
- Ruby is also utilized for scripting tasks, automation, and building software tools.


NETFLIX
airbnb
Microsoft
twitch
Bloomberg
GitHub
KICKSTARTER

When was Ruby created and its current status?


ruby / ruby
Type / to search
Code
Pull requests 391
Actions
Wiki
Security
Insights
ruby
Public
Watch 1.1k
Fork 5.4k
Star 20.6k
master
25 branches
1,140 tags
Go to file
Add file
Code
nobu and matzbot [rubygems/rubygems] Clear YAML constant if it was u...
419fbc7 15 minutes ago
78,454 commits
.github Fix a typo [ci skip] 13 hours ago
basictest [wasm] bootstraptest, basictest: disable backquote literal tests last year
benchmark Make rb_str_rindex return byte index 2 weeks ago
bin Revert "[ruby/syntax_suggest] Introduce binstubs to set RUBYOPT for d... 3 months ago
bootstraptest Fix a typo [ci skip] 3 days ago
ccan Fix -Wundef warnings 9 months ago
coroutine Add support for LoongArch (#7343) 5 months ago
coverage Fix Typo 2 years ago
cygwin leaked-globals: check leaked symbols in libruby.so if enable-shared 2 weeks ago
defs Serially update only the ripper source, even with old GNU make last week
doc [DOC] Fix a magic comment in the section for experimental_copy 2 days ago
enc [Bug #19728] Auto-generate unicode property docs 3 weeks ago
ext [flori/json] Re-generate parser.c yesterday
About
The Ruby Programming Language
www.ruby-lang.org/
ruby c language
programming-language rust jit
object-oriented ruby-language
Readme
Unknown, Unknown licenses found
Security policy
Activity
20.6k stars
1.1k watching
5.4k forks
Report repository
Releases 8
3.2.2 Latest
on Mar 30

- Ruby was first released in 1995 by Matz.
- It has since evolved through multiple versions, with the latest stable version being Ruby 3.2.2 (as of March 2023).
- Ruby continues to have an active and enthusiastic community, with ongoing development and updates.


LET'S GET STARTED!


Boy or girl?
Hello, world
programmer
it is

hello world!

```
puts ”hello, world!”
```

vs

```
print ”hello, world!”
```

The main difference between puts and print in Ruby is that puts adds a newline character after printing, while print does not.

Data Types

- Number
- String
- Symbol
- Boolean

1) Numbers

```
# Addition
1 + 1   #=> 2

# Subtraction
2 - 1   #=> 1

# Multiplication
2 * 2   #=> 4

# Division
10 / 5  #=> 2

# Exponent
2 ** 2  #=> 4
3 ** 4  #=> 81

# Modulus (find the remainder of division)
8 % 2   #=> 0  (8 / 2 = 4; no remainder)
10 % 4  #=> 2  (10 / 4 = 2 with a remainder of 2)
```

2) String

```ruby
name = "Odin"

puts "Hello, #{name}" #=> "Hello, Odin"
puts 'Hello, #{name}' #=> "Hello, #{name}"
```

3) Symbol

```
1 | :my_symbol
```

Symbols are different from strings. While strings are sequences of characters and can change their contents, symbols are immutable. This means that symbols always have the same value and can't be modified once created.

Used in keys in hashes.

4) Boolean

The Boolean values true and false represent exactly what you think they do: true represents something that is true, and false represents something that is false.

Variables

1) Local Variables:
   Used to store data within a specific scope.
   Example: name = "John"

2) Instance Variables:
   Used to store data accessible within an instance of a class.
   Example: @age = 25

3) Class Variables:
   Shared among all instances of a class.
   Example: @@count = 10

4) Global Variables:
   Accessed from anywhere in the program.
   Example: $total = 100

5) Constants:
   Used to store values that remain constant.
   Example: PI = 3.14159

Conditional Logic

- if
- else
- else if
- case
- then
- when
- unless

if, elsif, else

```ruby
if attack_by_land == true
  puts "release the goat"
elsif attack_by_sea == true
  puts "release the shark"
else
  puts "release Kevin the octopus"
end
```

case, when, then

```ruby
grade = 'F'

case grade
when 'A'
  puts "You're a genius"
  future_bank_account_balance = 5_000_000
when 'D'
  puts "Better luck next time"
  can_i_retire_soon = false
else
  puts "'YOU SHALL NOT PASS!' -Gandalf"
  fml = true
end
```

unless

```ruby
age = 19
puts "Welcome to a life of debt." unless age < 18

unless age < 18
  puts "Down with that sort of thing."
else
  puts "Careful now!"
end
```

Loops

- loop
- while
- for & ranges
- until

loop

```ruby
i = 0
loop do
  puts "i is #{i}"
  i += 1
  break if i == 10
end
```

while

```ruby
i = 0
while i < 10 do
 puts "i is #{i}"
 i += 1
end
```

for, ranges

```
1  for i in 0..5
2    puts "#{i} zombies incoming!"
3  end
```

until

```ruby
i = 0
until i >= 10 do
 puts "i is #{i}"
 i += 1
end
```

Array

Creating an Array

```
num_array = [1, 2, 3, 4, 5]
str_array = ["This", "is", "a", "small", "array"]
```

Accessing elements

```
str_array = ["This", "is", "a", "small", "array"]

str_array[0]            #=> "This"
str_array[1]            #=> "is"
str_array[4]            #=> "array"
str_array[-1]           #=> "array"
str_array[-2]           #=> "small"
```

Array

Adding and Removing Elements

```ruby
num_array = [1, 2]

num_array.push(3, 4)      #=> [1, 2, 3, 4]
num_array << 5            #=> [1, 2, 3, 4, 5]
num_array.pop             #=> 5
num_array                 #=> [1, 2, 3, 4]
```

Adding and Subtracting Arrays

```ruby
a = [1, 2, 3]
b = [3, 4, 5]

a + b         #=> [1, 2, 3, 3, 4, 5]
a.concat(b)   #=> [1, 2, 3, 3, 4, 5]
```


NYPD
Hashes
Array

Hashes

- Hashes store data in key-value pairs.
- Elements in a hash are accessed using unique keys, not indexes.
- Hashes allow you to quickly look up values based on specific keys.

vs

Array

- Arrays store data in an ordered sequence of elements.
- Elements in an array are accessed using indexes starting from 0.
- Arrays preserve the order of elements and allow easy manipulation.


THANK YOU FOR YOUR ATTENTION
ANY QUESTIONS?
makeameme.org