
Look at all these toys!


Help – it’s Ruby!


All alone


Is Ruby dying?


Where do Rubyists go?

Where do Rubyists go?

Tobias Pfeiffer
@PragTob
pragtob.info

LIEFERY

Where do Rubyists go?

This questionnaire aims to identify which programming languages Ruby programmers are interested in and the reasons why these other languages are appealing to them. This is done out of interest and as part of the preparation for my talk "Where do Rubyists go?" that also wants to take a look at the most popular languages.

Please only answer this questionnaire if you did Ruby in the past or are currently doing Ruby. If you didn't learn any new languages that's also great and a very valuable answer :)

Results will also be published in blog form (probably early 2018). Free text field answers will not be published in full. They're safe with me (Tobi / @PragTob). I might quote extracts of answers but only if there's no personally identifying information.

For the free form text fields please aim to be concise, I want to read and categorize all of them so half a blog post might be interesting but quite challenging for me to go through everything then.

*Required

When did you start learning Programming? (Please specify the year) *

Your answer

When did you start learning Ruby? (Please specify the year) *

Your answer

Why did you get into/learn Ruby in the first place? *

Your answer

Are you currently doing Ruby in your free time? *

- ◯ Yes
- ◯ No

Where do Rubyists go?

673 Responses

This questionnaire aims to identify which programming languages Ruby programmers are interested in and the reasons why these other languages are appealing to them. This is done out of interest and as part of the preparation for my talk "Where do Rubyists go?" that also wants to take a look at the most popular languages.

Please only answer this questionnaire if you did Ruby in the past or are currently doing Ruby. If you didn't learn any new languages that's also great and a very valuable answer :)

Results will also be published in blog form (probably early 2018). Free text field answers will not be published in full. They're safe with me (Tobi / @PragTob). I might quote extracts of answers but only if there's no personally identifying information.

For the free form text fields please aim to be concise, I want to read and categorize all of them so half a blog post might be interesting but quite challenging for me to go through everything then.

*Required

When did you start learning Programming? (Please specify the year) *

Your answer

When did you start learning Ruby? (Please specify the year) *

Your answer

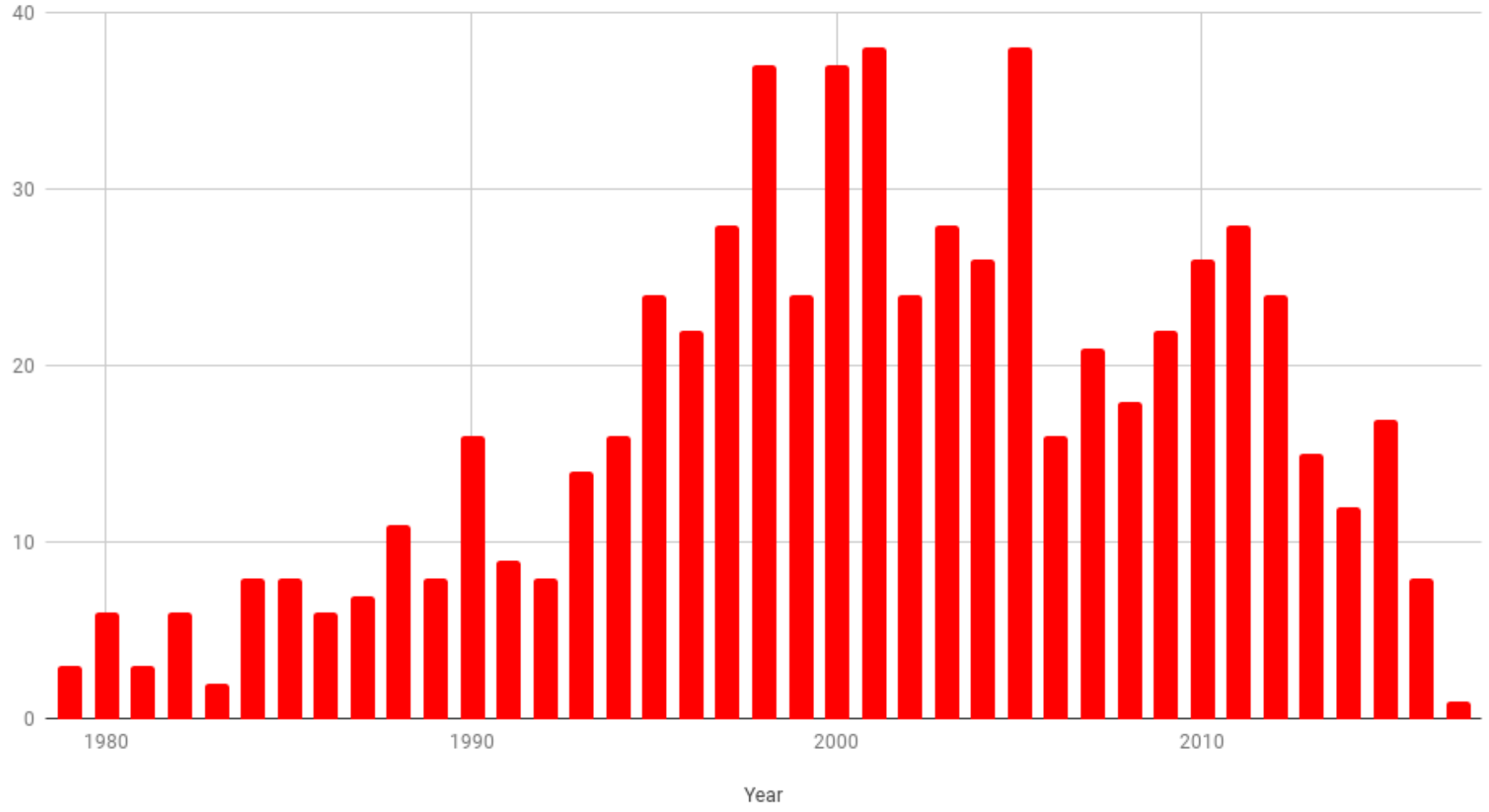
Why did you get into/learn Ruby in the first place? *

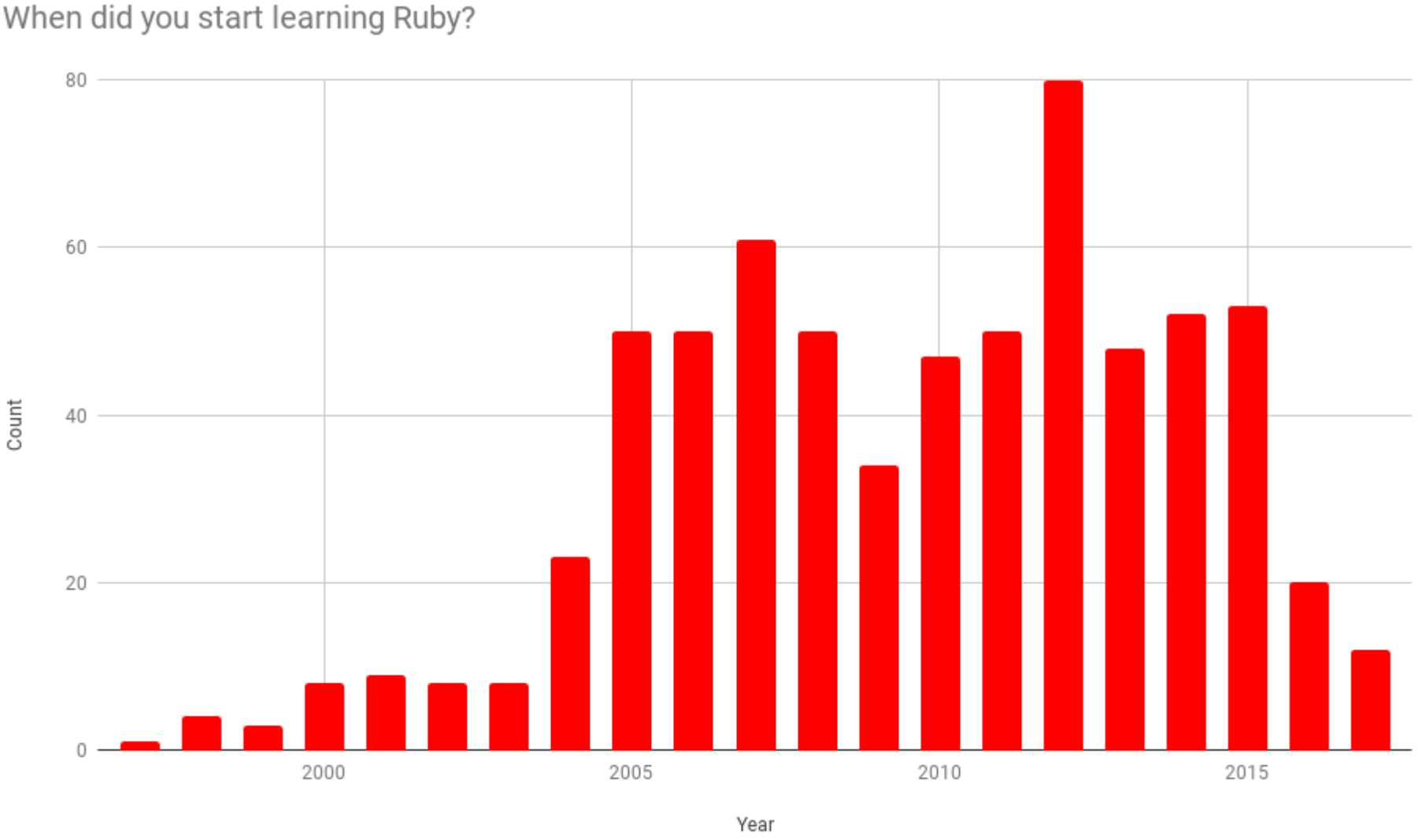
Your answer

Are you currently doing Ruby in your free time? *

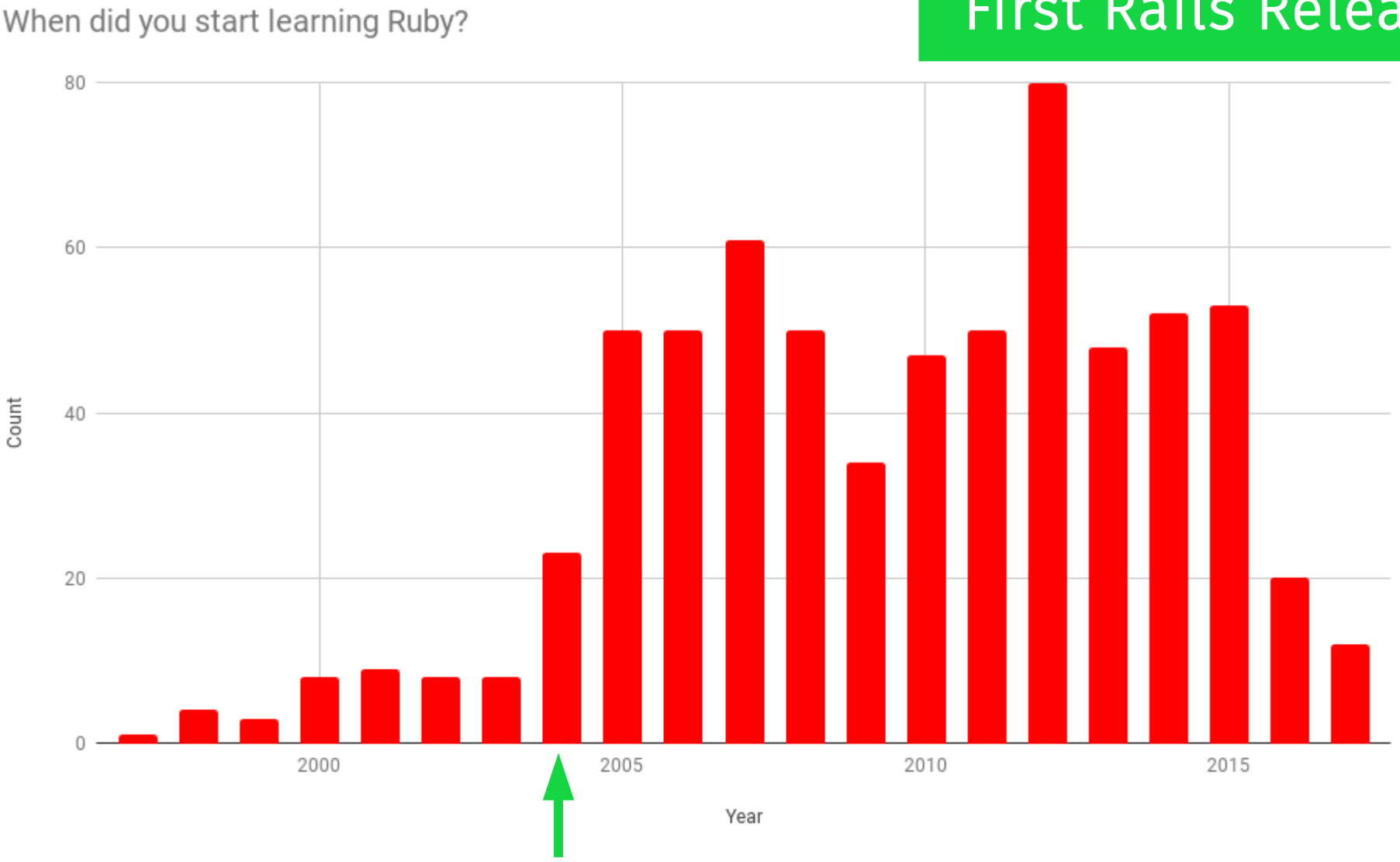
- ◯ Yes
- ◯ No

When did you start learning Programming?

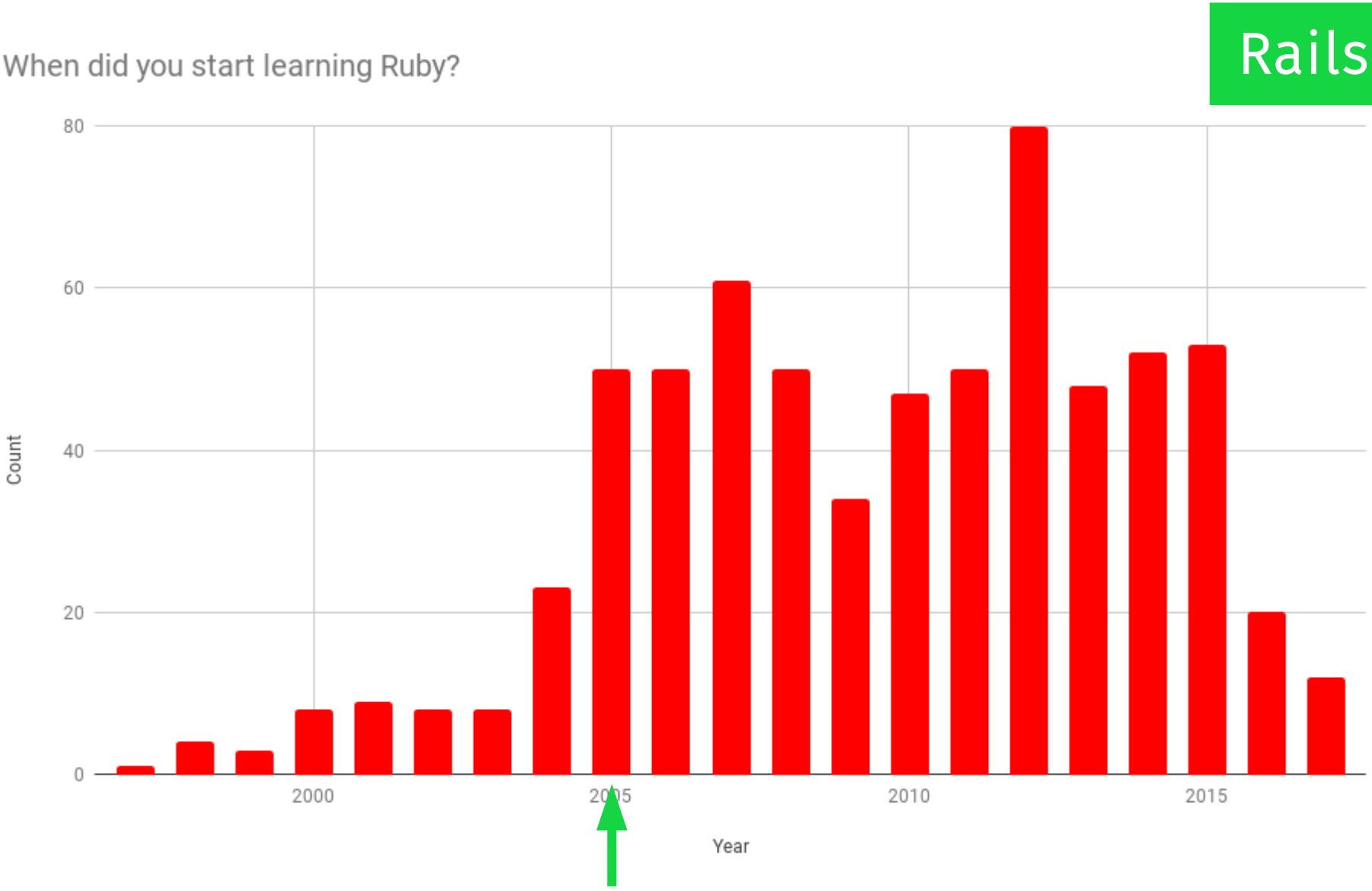



When did you start learning Ruby?
80
60
40
20
0
Count
2000
2005
2010
2015
Year

First Rails Release



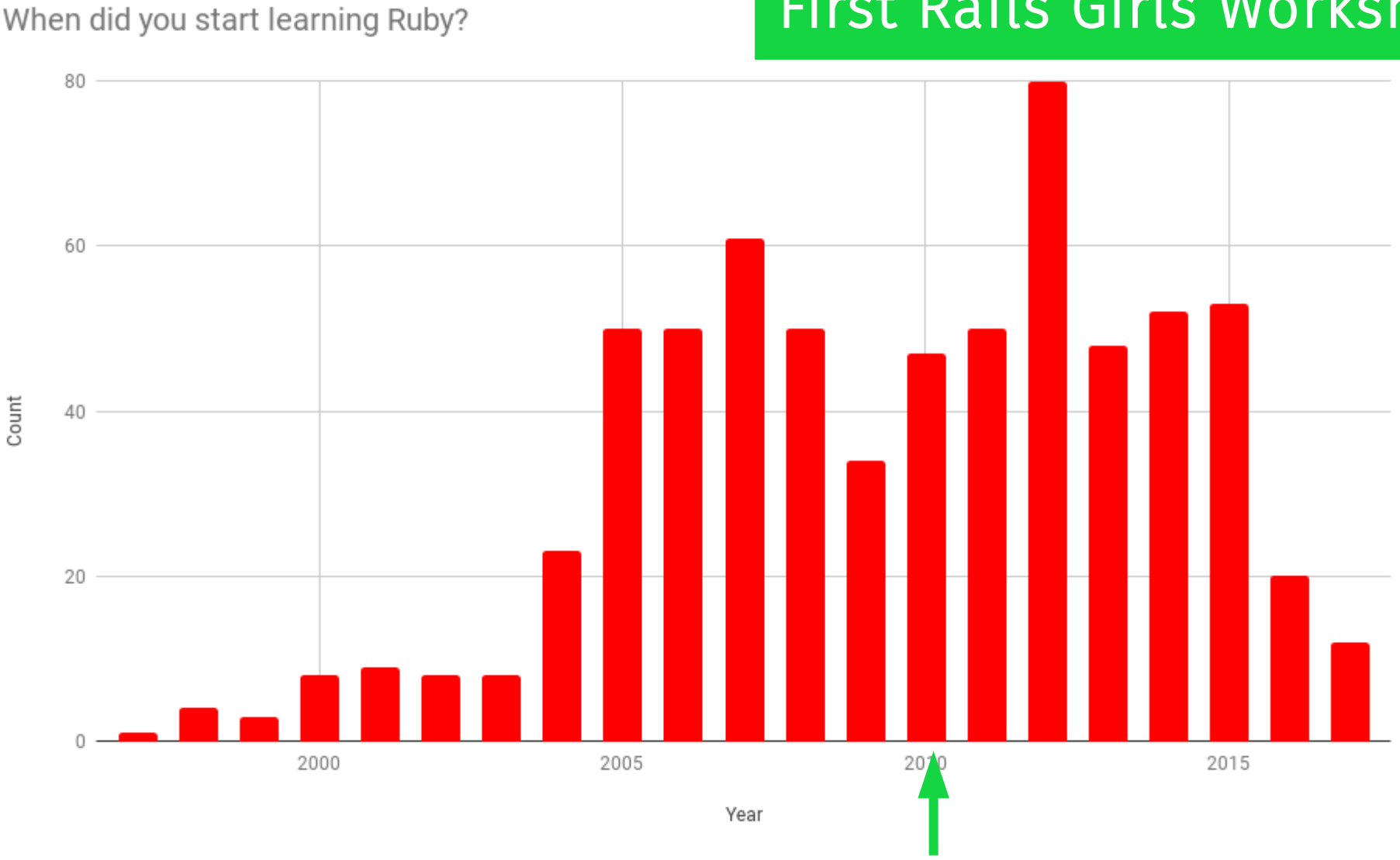
Rails 1.0


When did you start learning Ruby?
80
60
40
20
0
Count
2000
2005
2010
2015
Year

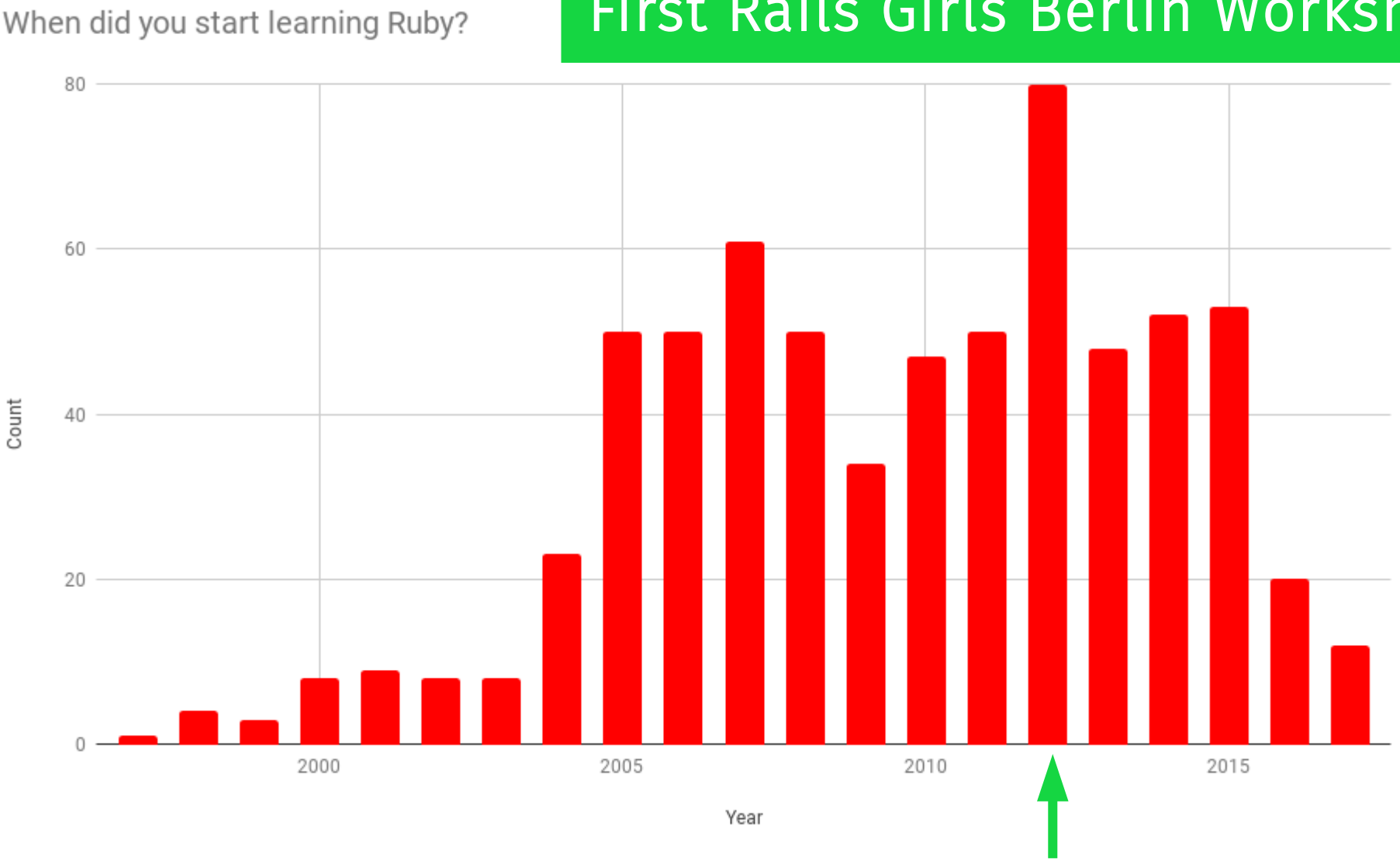
Why did you learn Ruby?

“I had written half of Rails in PHP. Then Rails was announced and it was like a cheat code to a working framework.”

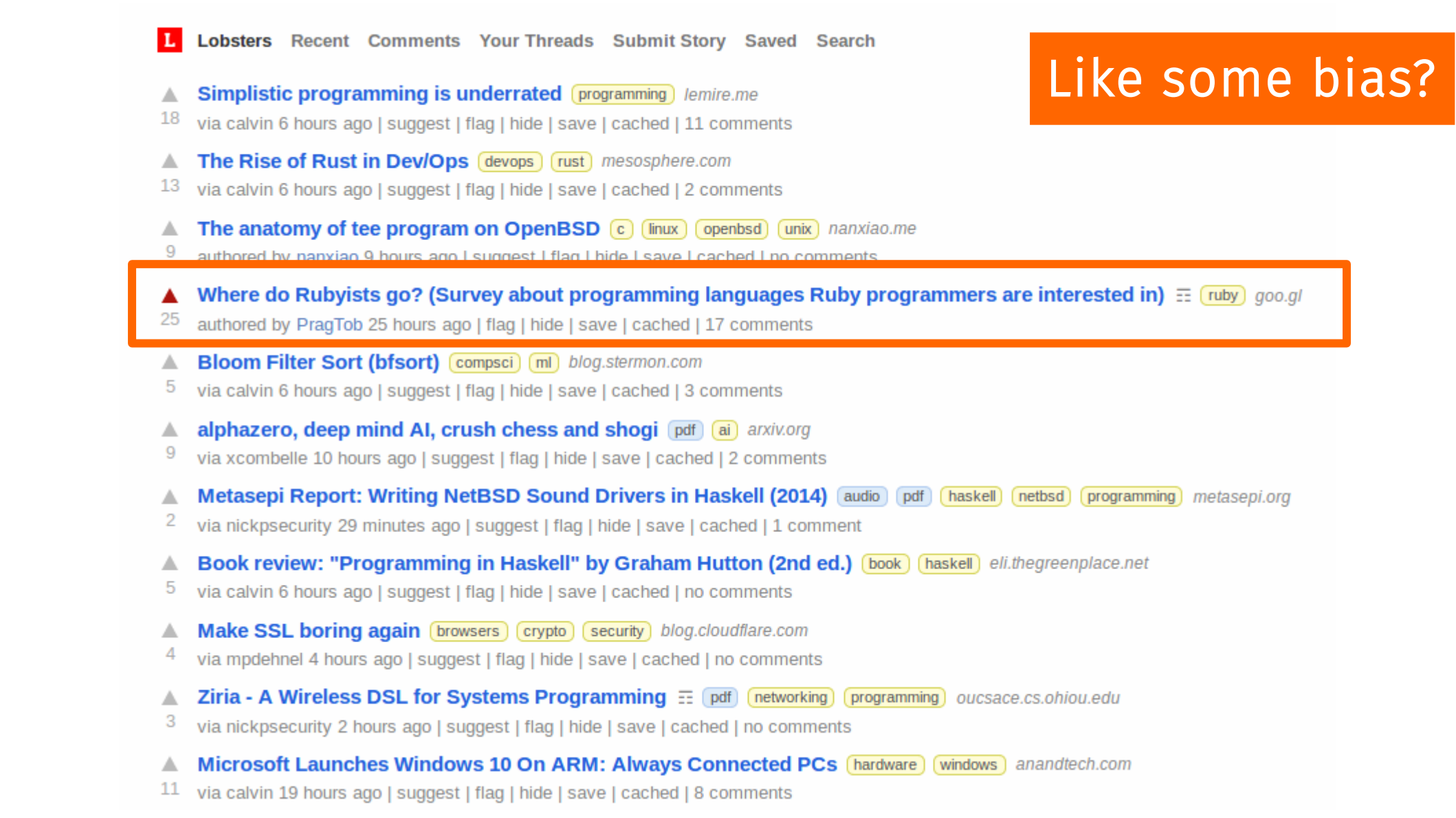
First Rails Girls Workshop



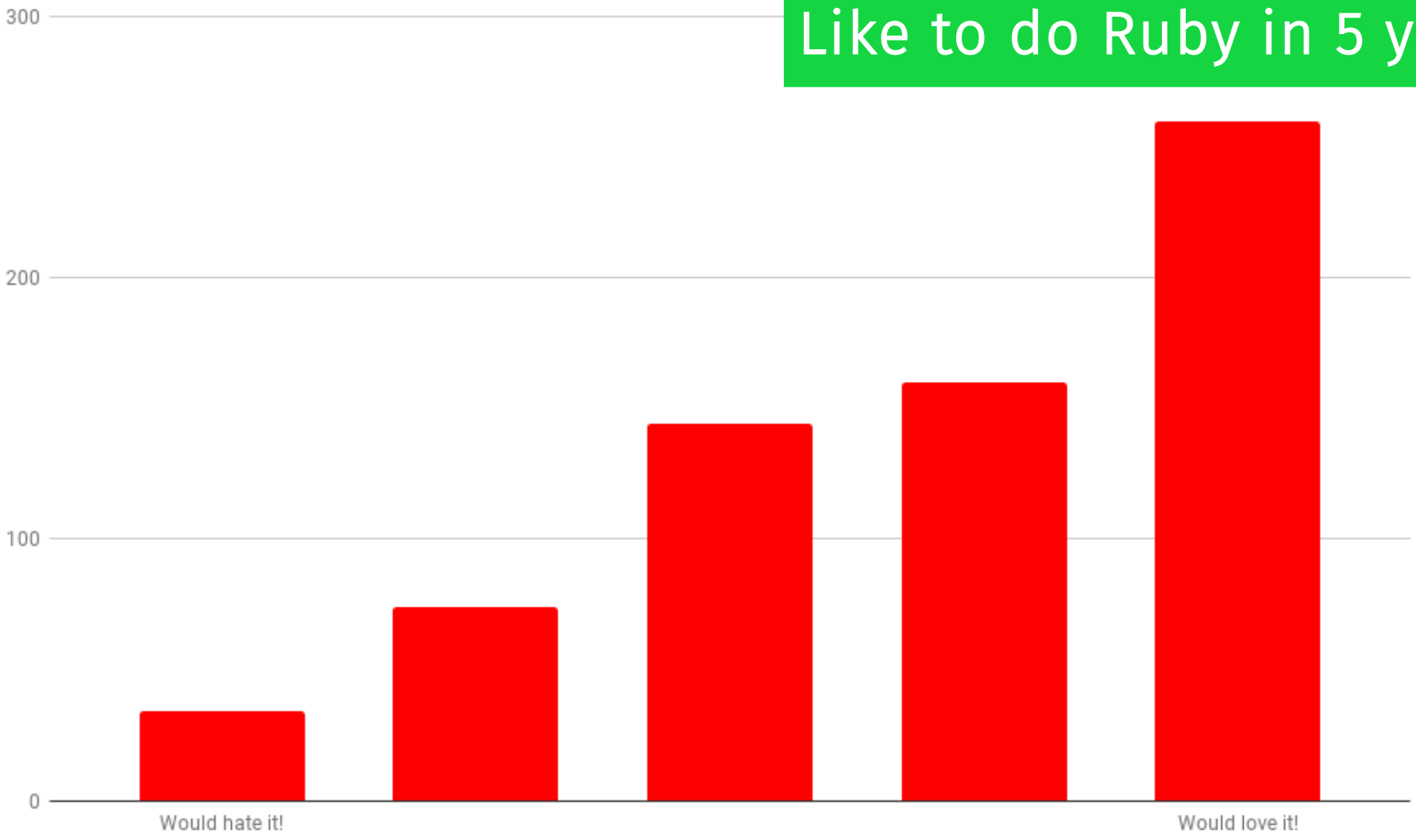
First Rails Girls Berlin Workshop

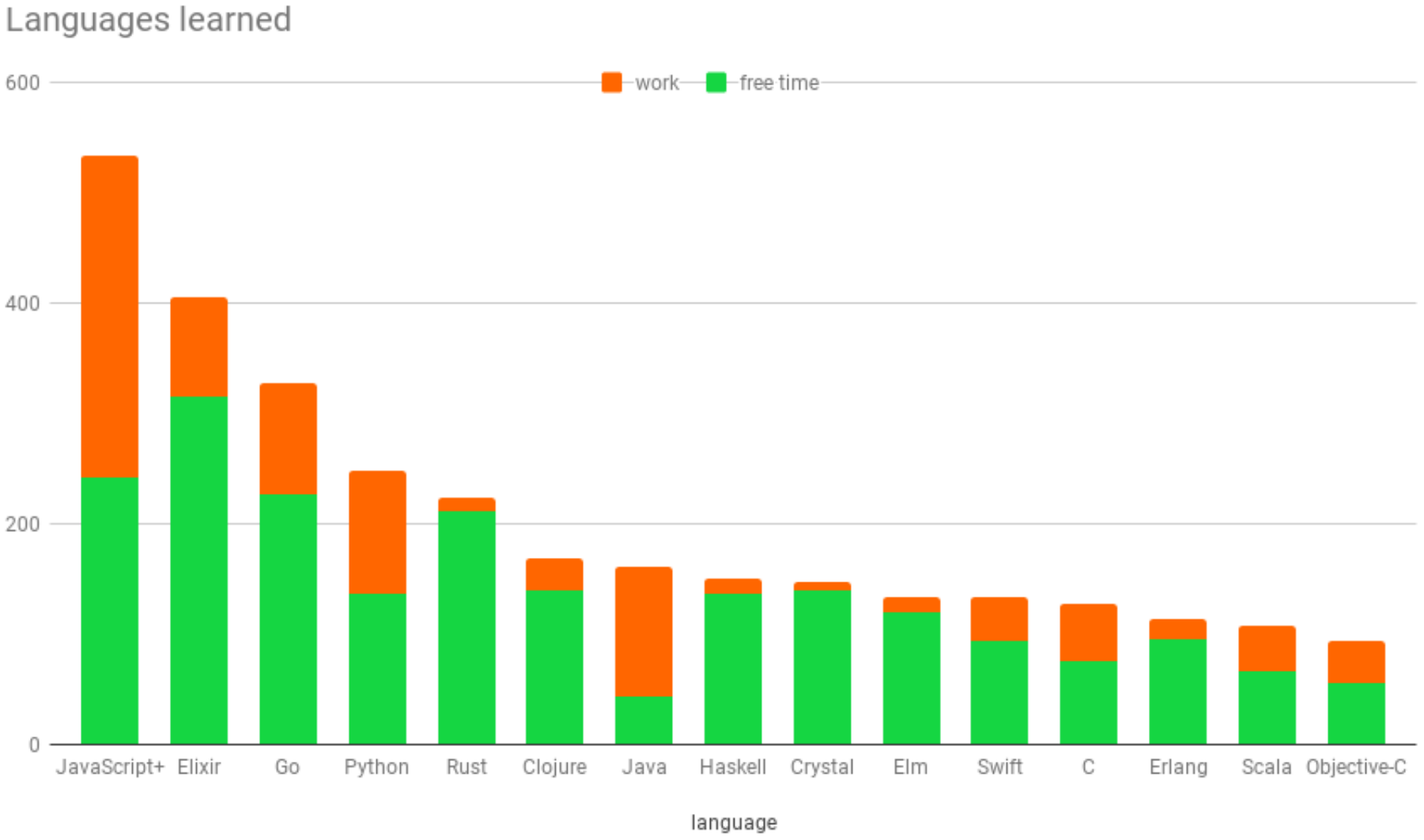


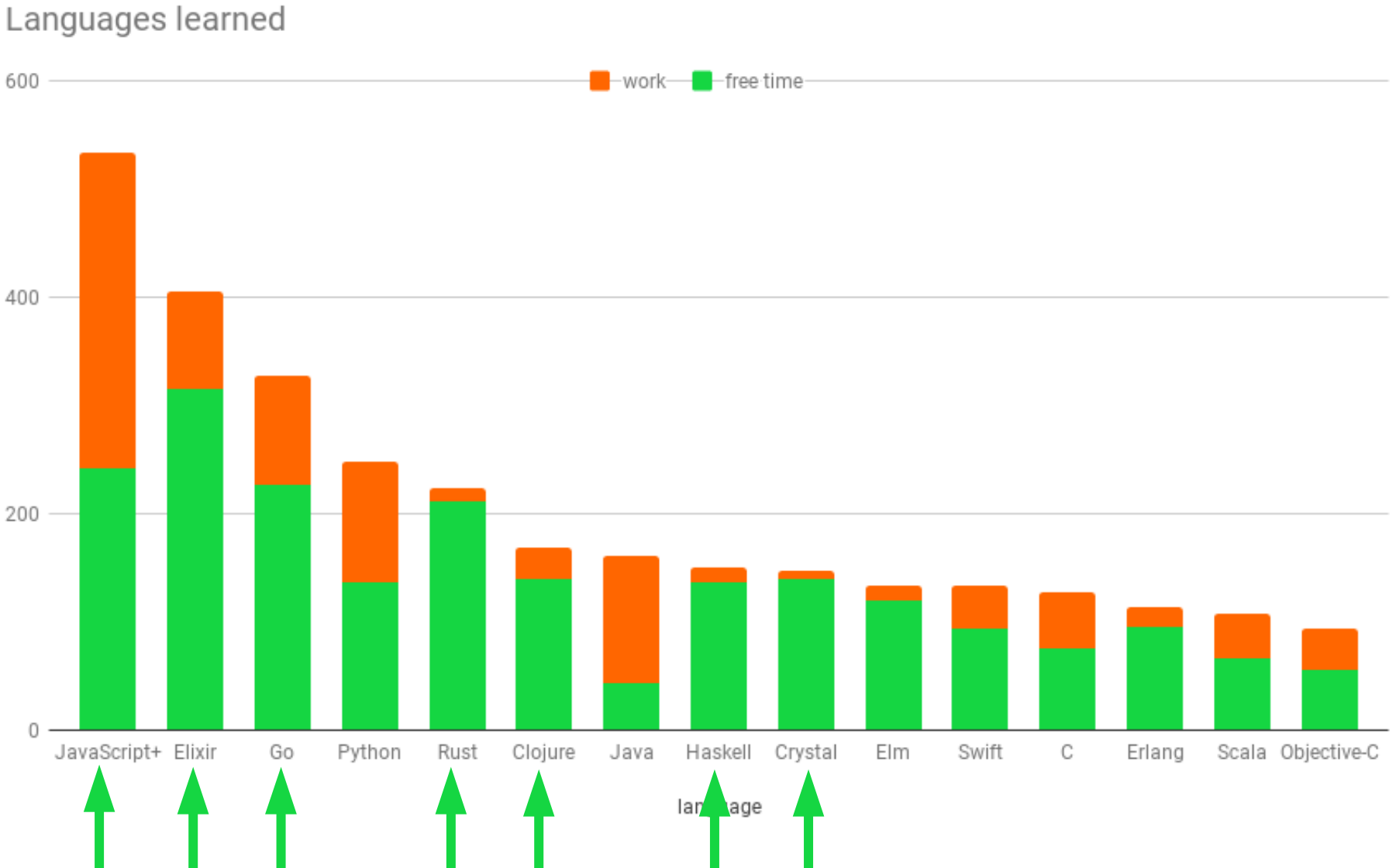
Surveys and Bias


L
Lobsters Recent Comments Your Threads Submit Story Saved Search
Like some bias?
18 Simplistic programming is underrated programming lemire.me
via calvin 6 hours ago | suggest | flag | hide | save | cached | 11 comments
13 The Rise of Rust in Dev/Ops devops rust mesosphere.com
via calvin 6 hours ago | suggest | flag | hide | save | cached | 2 comments
9 The anatomy of tee program on OpenBSD c linux openbsd unix nanxiao.me
authored by nanxiao 9 hours ago | suggest | flag | hide | save | cached | no comments
25 Where do Rubyists go? (Survey about programming languages Ruby programmers are interested in) ruby goo.gl
authored by PragTob 25 hours ago | flag | hide | save | cached | 17 comments
5 Bloom Filter Sort (bfsort) compsci ml blog.stermon.com
via calvin 6 hours ago | suggest | flag | hide | save | cached | 3 comments
9 alphazero, deep mind AI, crush chess and shogi pdf ai arxiv.org
via xcombelle 10 hours ago | suggest | flag | hide | save | cached | 2 comments
2 Metasepi Report: Writing NetBSD Sound Drivers in Haskell (2014) audio pdf haskell netbsd programming metasepi.org
via nickpsecurity 29 minutes ago | suggest | flag | hide | save | cached | 1 comment
5 Book review: "Programming in Haskell" by Graham Hutton (2nd ed.) book haskell eli.thegreenplace.net
via calvin 6 hours ago | suggest | flag | hide | save | cached | no comments
4 Make SSL boring again browsers crypto security blog.cloudflare.com
via mpdehnel 4 hours ago | suggest | flag | hide | save | cached | no comments
3 Ziria - A Wireless DSL for Systems Programming pdf networking programming oucsace.cs.ohiou.edu
via nickpsecurity 2 hours ago | suggest | flag | hide | save | cached | no comments
11 Microsoft Launches Windows 10 On ARM: Always Connected PCs hardware windows anandtech.com
via calvin 19 hours ago | suggest | flag | hide | save | cached | 8 comments

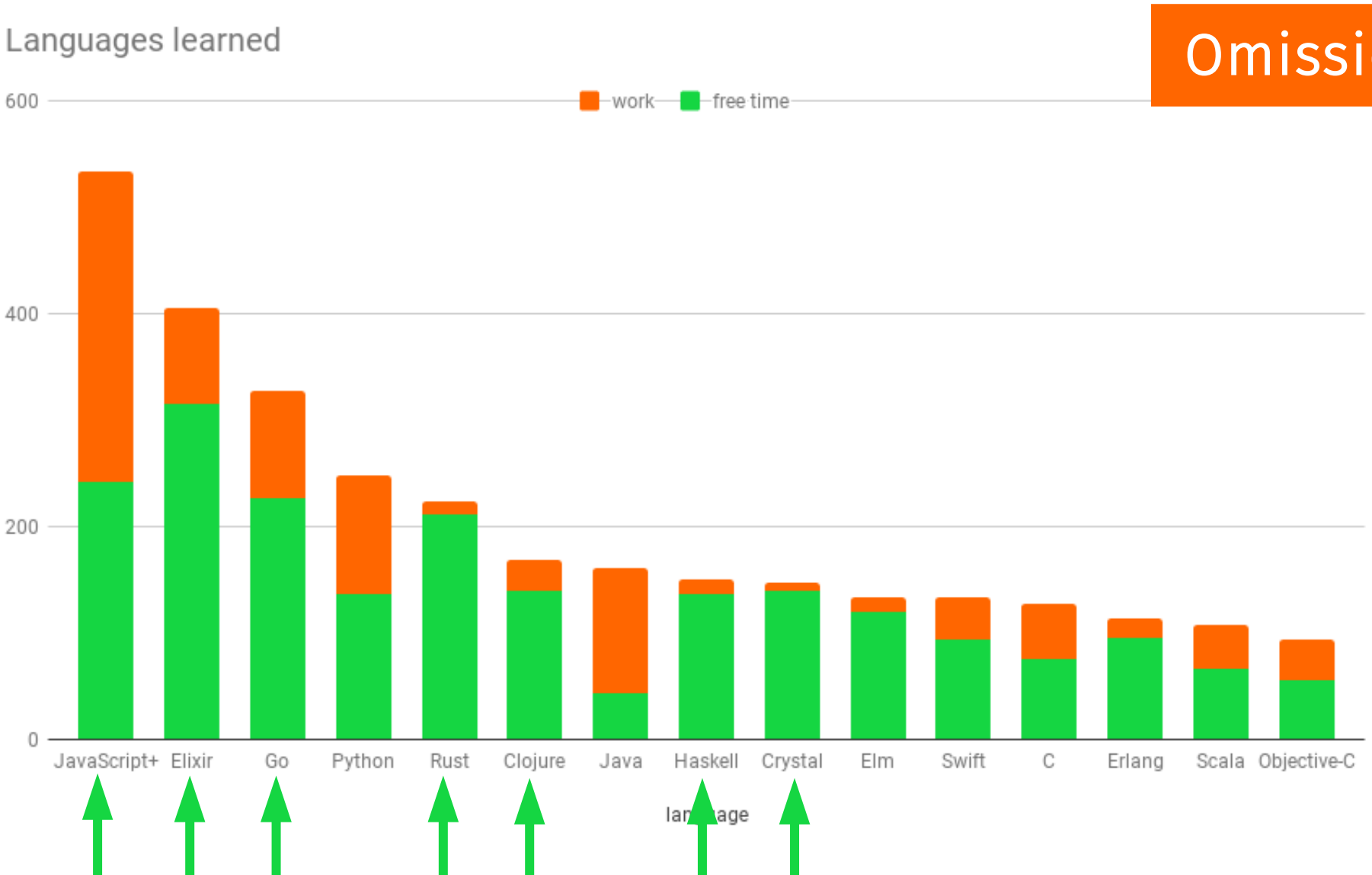
Like to do Ruby in 5 years?




Languages learned
work
free time
600
400
200
0
JavaScript+
Elixir
Go
Python
Rust
Clojure
Java
Haskell
Crystal
Elm
Swift
C
Erlang
Scala
Objective-C
language


Languages learned
work
free time
600
400
200
0
JavaScript+
Elixir
Go
Python
Rust
Clojure
Java
Haskell
Crystal
Elm
Swift
C
Erlang
Scala
Objective-C

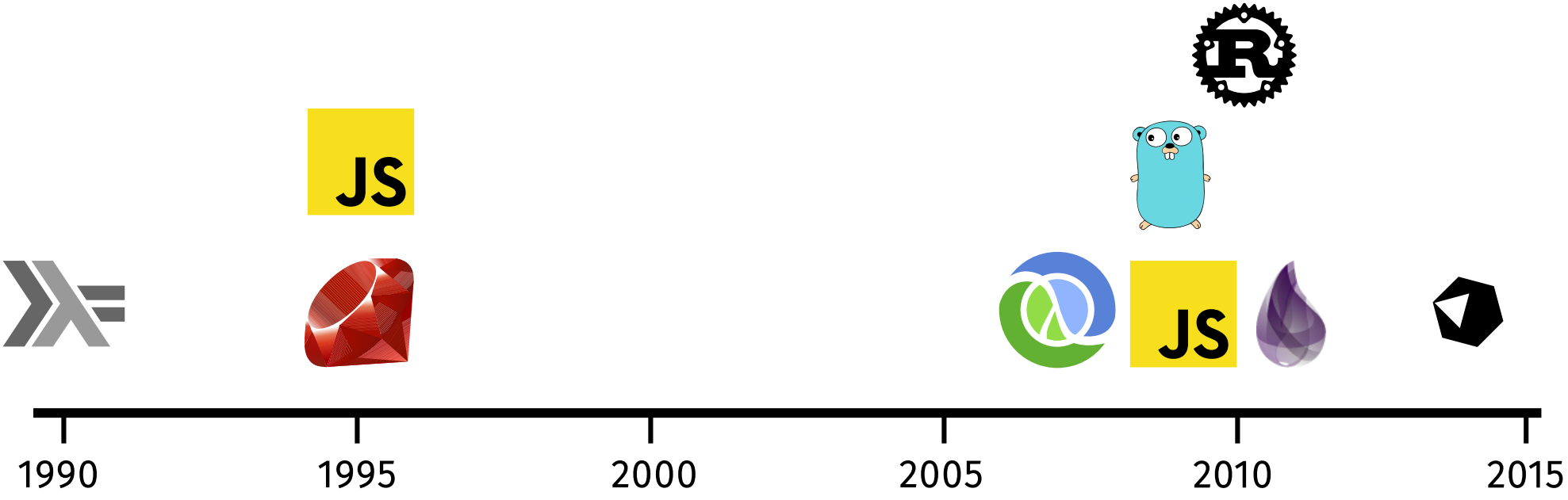
Omissions

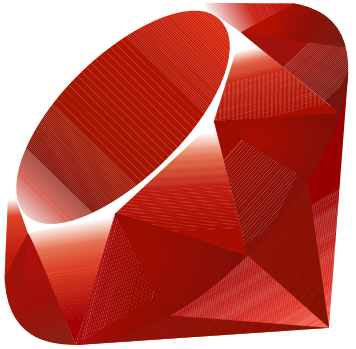

Languages learned
work
free time
600
400
200
0
JavaScript+
Elixir
Go
Python
Rust
Clojure
Java
Haskell
Crystal
Elm
Swift
C
Erlang
Scala
Objective-C

Tools

Disclaimer


Meet & Greet


JS
JS
1990
1995
2000
2005
2010
2015



Name: Ruby

Popular Rubyists: All of them

Known for: Metaprogramming, dynamic, Scripting, web

Self-assessment: *A dynamic, open source programming language with a focus on simplicity and productivity.*

FizzBuzz!

1
2

1
2
Fizz

1
2
Fizz
4
Buzz

1
2
Fizz
4
Buzz
Fizz

1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14

1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz

1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
…

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

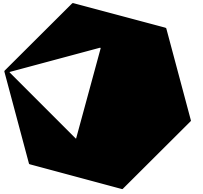
```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```



CRYSTAL

Name:	Crystal
Popular Rubyists:	Erik Berlin, Piotr Szotkowski, Fabio Akita
Known for:	Ruby-like, Performance, Type Inference
Self-assessment:	*Fast as C, slick as Ruby*

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

cp fizzbuzz.rb fizzbuzz.cr

```ruby
def fizzbuzz(n)
  if (n % 15).zero?
    "FizzBuzz"
  elsif (n % 5).zero?
    "Buzz"
  elsif (n % 3).zero?
    "Fizz"
  else
    n
  end
end

(1..100).each {|n| puts fizzbuzz(n)}
```

Name: Elixir

Popular Rubyists: José Valim, Dave Thomas, Xavier Noria

Known for: Erlang VM, Actors, Functional, Phoenix

Self-assessment: *dynamic, functional language designed for building scalable and maintainable applications.*

```elixir
defmodule FizzBuzz do
  def fizzbuzz(n) when rem(n, 15) == 0, do: "FizzBuzz"
  def fizzbuzz(n) when rem(n, 5) == 0, do: "Buzz"
  def fizzbuzz(n) when rem(n, 3) == 0, do: "Fizz"
  def fizzbuzz(n), do: n
end

Enum.each(1..100, fn i -> IO.puts(FizzBuzz.fizzbuzz(i)) end)
```

```elixir
defmodule FizzBuzz do
  def fizzbuzz(n) when rem(n, 15) == 0, do: "FizzBuzz"
  def fizzbuzz(n) when rem(n, 5) == 0, do: "Buzz"
  def fizzbuzz(n) when rem(n, 3) == 0, do: "Fizz"
  def fizzbuzz(n), do: n
end

Enum.each(1..100, fn i -> IO.puts(FizzBuzz.fizzbuzz(i)) end)
```

Name: Haskell

Popular Rubyists: Chad Fowler

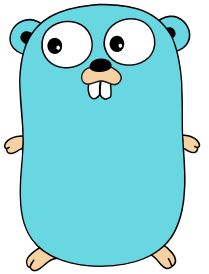
Known for: Type System, Monads, Pure

Self-assessment: *An advanced, purely functional programming language.*

```haskell
main = mapM_ (putStrLn . fizzbuzz) [1..100]

fizzbuzz x
    | x `mod` 15 == 0 = "FizzBuzz"
    | x `mod`  3 == 0 = "Fizz"
    | x `mod`  5 == 0 = "Buzz"
    | otherwise = show x
```

```haskell
main = mapM_ (putStrLn . fizzbuzz) [1..100]

fizzbuzz x
    | x `mod` 15 == 0 = "FizzBuzz"
    | x `mod`  3 == 0 = "Fizz"
    | x `mod`  5 == 0 = "Buzz"
    | otherwise = show x
```

```haskell
main = mapM_ (putStrLn . fizzbuzz) [1..100]

fizzbuzz x
    | x `mod` 15 == 0 = "FizzBuzz"
    | x `mod`  3 == 0 = "Fizz"
    | x `mod`  5 == 0 = "Buzz"
    | otherwise = show x
```



Name: Go

Popular Rubyists: Katrina Owen, Evan Phoenix

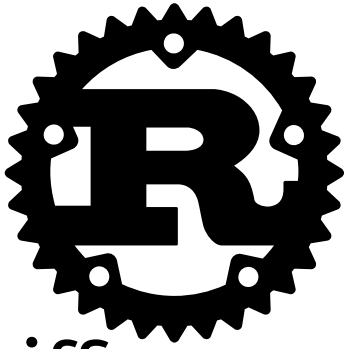
Known for: Goroutines, Simple, No Exceptions, No Generics

Self-assessment: *open source programming language that makes it easy to build simple, reliable, and efficient software.*

```go
package main

import "fmt"
import "strconv"

func FizzBuzz(i int) string {
    switch {
    case i%15 == 0:
        return "FizzBuzz"
    case i%3 == 0:
        return "Fizz"
    case i%5 == 0:
        return "Buzz"
    default:
        return strconv.Itoa(i)
    }
}

func main() {
    for i := 1; i <= 100; i++ {
        fmt.Println(FizzBuzz(i))
    }
}
```

```go
package main

import "fmt"
import "strconv"

func FizzBuzz(i int) string {
    switch {
    case i%15 == 0:
        return "FizzBuzz"
    case i%3 == 0:
        return "Fizz"
    case i%5 == 0:
        return "Buzz"
    default:
        return strconv.Itoa(i)
    }
}

func main() {
    for i := 1; i <= 100; i++ {
        fmt.Println(FizzBuzz(i))
    }
}
```

```go
package main

import "fmt"
import "strconv"

func FizzBuzz(i int) string {
    switch {
    case i%15 == 0:
        return "FizzBuzz"
    case i%3 == 0:
        return "Fizz"
    case i%5 == 0:
        return "Buzz"
    default:
        return strconv.Itoa(i)
    }
}

func main() {
    for i := 1; i <= 100; i++ {
        fmt.Println(FizzBuzz(i))
    }
}
```



Name: Rust

Popular Rubyists: Steve Klabnik, Yehuda Katz, Sean Griffin

Known for: Memory Management, Compiler, Firefox Quantum

Self-assessment: *a systems programming language that runs blazingly fast, prevents segfaults, and guarantees thread safety.*

```rust
fn main() {
    (1..101).for_each(|n| println!("{}", fizzbuzz(n)))
}

fn fizzbuzz(n: i32) -> String {
    match (n % 3, n % 5) {
        (0, 0) => "FizzBuzz".to_string(),
        (0, _) => "Fizz".to_string(),
        (_, 0) => "Buzz".to_string(),
        _ => n.to_string(),
    }
}
```

```rust
fn main() {
    (1..101).for_each(|n| println!("{}", fizzbuzz(n)))
}

fn fizzbuzz(n: i32) -> String {
    match (n % 3, n % 5) {
        (0, 0) => "FizzBuzz".to_string(),
        (0, _) => "Fizz".to_string(),
        (_, 0) => "Buzz".to_string(),
        _ => n.to_string(),
    }
}
```

```rust
fn main() {
    (1..101).for_each(|n| println!("{}", fizzbuzz(n)))
}

fn fizzbuzz(n: i32) -> String {
    match (n % 3, n % 5) {
        (0, 0) => "FizzBuzz".to_string(),
        (0, _) => "Fizz".to_string(),
        (_, 0) => "Buzz".to_string(),
        _ => n.to_string(),
    }
}
```

Name: JavaScript

Popular Rubyists: Yehuda Katz, Jeremy Ashkenas

Known for: Quirks, Async, Compile to

Self-assessment: *a lightweight interpreted or JIT-compiled programming language with first-class functions.*

```
const fizzBuzz = n => {
    if (n % 15 === 0) {
        return "FizzBuzz";
    } else if (n % 3 === 0) {
        return "Fizz";
    } else if (n % 5 === 0) {
        return "Buzz";
    } else {
        return n;
    }
};

for (let n = 1; n <= 100; n += 1) {
    console.log(fizzBuzz(n));
}
```

```
const fizzBuzz = n => {
    if (n % 15 === 0) {
        return "FizzBuzz";
    } else if (n % 3 === 0) {
        return "Fizz";
    } else if (n % 5 === 0) {
        return "Buzz";
    } else {
        return n;
    }
};

for (let n = 1; n <= 100; n += 1) {
    console.log(fizzBuzz(n));
}
```

```
const fizzBuzz = n => {
    if (n % 15 === 0) {
        return "FizzBuzz";
    } else if (n % 3 === 0) {
        return "Fizz";
    } else if (n % 5 === 0) {
        return "Buzz";
    } else {
        return n;
    }
};

for (let n = 1; n <= 100; n += 1) {
    console.log(fizzBuzz(n));
}
```

Name: Clojure

Popular Rubyists: Russ Olsen, Bozhidar Batsov, Arne Brasseur

Known for: *Rich Hickey, Lisp, JVM, ()*

Self-assessment: *a robust, practical, and fast programming language with a set of useful features that together form a simple, coherent, and powerful tool.*

```clojure
(defn fizzbuzz [n]
   (cond
     (zero? (mod n 15)) "FizzBuzz"
     (zero? (mod n 3)) "Fizz"
     (zero? (mod n 5)) "Buzz"
     :else n))

(run! println (map fizzbuzz (range 1 101)))
```

```clojure
(defn fizzbuzz [n]
   (cond
     (zero? (mod n 15)) "FizzBuzz"
     (zero? (mod n 3)) "Fizz"
     (zero? (mod n 5)) "Buzz"
     :else n))

(run! println (map fizzbuzz (range 1 101)))
```

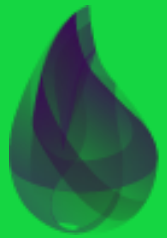
```clojure
(defn fizzbuzz [n]
   (cond
     (zero? (mod n 15)) "FizzBuzz"
     (zero? (mod n 3)) "Fizz"
     (zero? (mod n 5)) "Buzz"
     :else n))

(run! println (map fizzbuzz (range 1 101)))
```


What you got?

Paradigm

Object

Object



Functional

Functional

Procedural

Procedural
Functional

Functional
Procedural

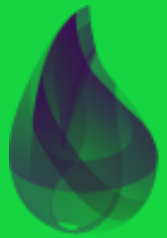
Functional

Parallelism

Parallelism vs Concurrency

Concurrent

Concurrent



Yes

Actors

Yes

Mvar, par, STM

Yes

Goroutines + channels

Yes

Agnostic

JS

Yes/No

Webworkers+

Yes

STM, pmap, Transducers

Performance!

Type System

Dynamic

Static
Inferred++

Dynamic
Optional
Inferred++

Static
Inferred++

Static
Inferred

Static
Inferred

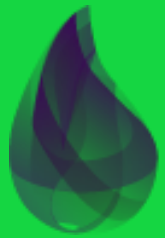
Dynamic
Optional
Inferred++

Dynamic
Optional
Inferred++

Compiled vs Interpreted

Interpreted

Compiled



Compiled

Compiled

Compiled

Compiled

Interpreted

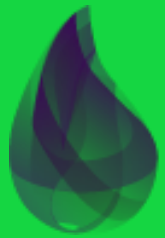
Compiled

Self-hosted


Show me your code

No

Yes



Yes

Yes

Yes

Yes

JS

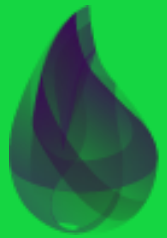
No

No

Garbage-Collection

Yes

Yes



Yes

Yes

Yes

No

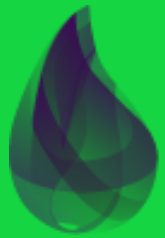
Yes

Yes

Single File Distribution

No

Yes



Yes

Yes

Yes

Yes

JS

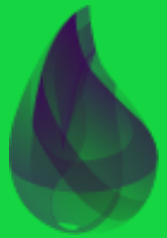
No

No

Ruby-like Syntax

Yes

Yes



Yes

No

No

No

No

No


Parallel


Parallel
Typing


Parallel
Typing
Fast


So, what?


Expand Your Mind
NEVER STOP LEARNING

Why did you learn a new language?

“Ruby's OO model was brain-expanding, and I was seeking more brain-expanding paradigms that would let me think entirely new thoughts.”


Joy


Domain

Tools


Where does
Ruby go?

*“Rails is **strangling Ruby**. In the same way that you don't quit because of a bad company, you **quit because of a bad boss**.”*


Where does
Ruby go?


Parallel
Typing
Fast

*“Ruby is the **best language** I have used over my 30 years programming. I hope Ruby 3 puts **an end to the Ruby is slow meme** once and for all.”*

*“I really like Ruby for **what it is**, and don't think 'adding a type system' or something is the best way to keep Ruby relevant. Don't **morph** Ruby **in to something it's not.**”*


We’re all great!


Explore some new lands!

Enjoy Coding & **Learning** in whatever language…

Tobias Pfeiffer
@PragTob
pragtob.info

LIEFERY