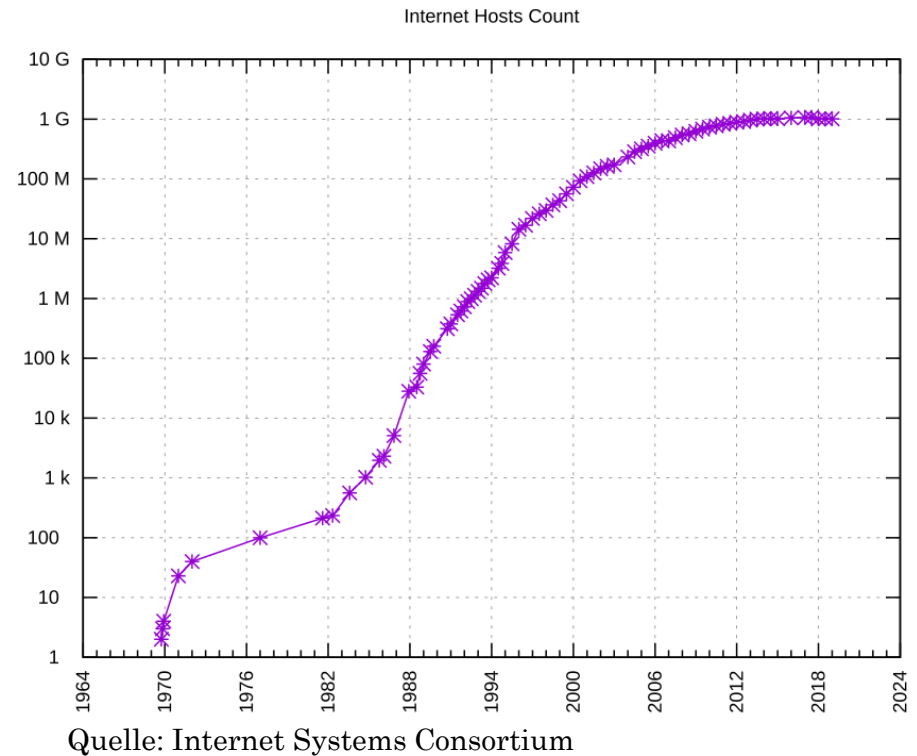


Ruby



Situation in den 1990ern

- Schnelles Wachstum des Internets
 - Steigende Anforderungen an die Webentwicklung
- Produktivität steht zunehmend im Mittelpunkt
- Objektorientierte Programmierung wird zum führenden Paradigma



Ziele und Designphilosophie

Freude am
Programmieren

Produktivität

Intuitives
Verhalten

Einfachheit

Universalität


"I believe that the purpose of life is, at least in part, to be happy. Based on this belief, Ruby is designed to make programming not only easy but also fun. It allows you to concentrate on the creative side of programming, with less stress."
– Yukihiro Matsumoto




Quelle: www.flickr.com

Features

- Einfache Syntax
- dynamische Typisierung
- Garbage Collection
- Objektorientierung
- Block-basierte Programmierung
- Metaprogrammierung
- Exception Handling
- read-eval-print loop



```
1 puts "Enter the lower limit:"
2 lwr = gets.chomp.to_i
3 puts "Enter the upper limit:"
4 upr = gets.chomp.to_i
5
6 for yr in lwr..upr do
7   if yr % 400 == 0
8     puts "#{yr} is a leap year"
9   elsif yr % 4 == 0 && yr % 100 != 0
10    puts "#{yr} is a leap year"
11  else
12    puts "#{yr} is not a leap year"
13  end
14 end
```



```
1 10.times do
2   puts "Hello world!"
3 end
```

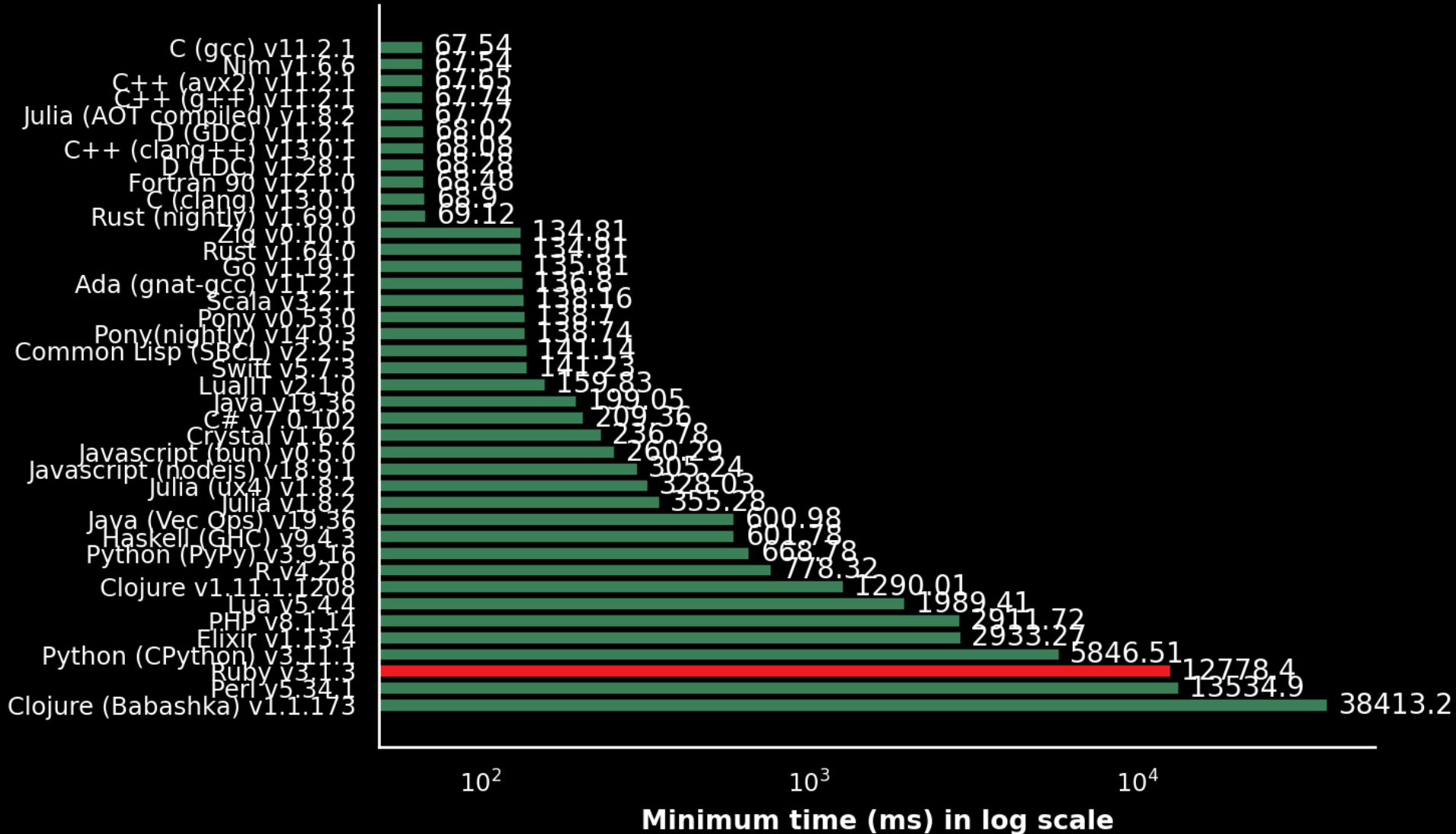
Infrastruktur

- Entwicklung als Open-Source Projekt
 - <https://github.com/ruby/ruby>
- Offizielles Paketverwaltungssystem RubyGems
 - <https://rubygems.org/>

Speed comparison of various programming languages

Method: calculating π through the Leibniz formula 100000000 times

1/24/2025



Performance

! geringe Performance, besonders im Vergleich zu kompilierten Sprachen

>> Verbesserung der Performance durch JIT-Compiler

- Separater Thread kompiliert mehrfach verwendete Codestücke

Anwendungsbereiche von Ruby

- Webentwicklung
- Automatisierung und Scripting
- Datenverarbeitung

Ruby on Rails

- David Heinemeier Hansson, 2005
- MVC Framework
- Convention Over Configuration
- Eingebaute Tools



grep-Utility

- Native Unterstützung regulärer Ausdrücke
- Umfangreiche Möglichkeiten für Dateizugriffe
- Umfangreiche Möglichkeiten zur Manipulation von Daten



```
1 # Das pattern aus den Argumenten abrufen
2 pattern = ARGV.shift
3 regexp = Regexp.new(pattern, regexp_options)
```



```
1 lines.each_with_index do |line, idx|
2   # Überprüfen, ob die Zeile das pattern enthält
3   if line.match?(regexp) do
4     # ...
5   end
6 end
```



```
1 Dir.glob("#{path}/**/*").each do |file|
2   lines = File.open(file).readlines.map(&:chomp)
3 end
```

Quellen

<https://www.ruby-lang.org>

<https://github.com/ruby/ruby>

<https://rubygems.org/>

<https://ruby-doc.com>