Moderne Programmiersprachen: Rust

Softwareentwicklung in der Wissenschaft

Christian Willner

11.05.2020

1 Einleitung
Einordnung und Abgrenzung
Entwicklungsgeschichte
Anwendungsbeispiele

2 Syntax und Umgebung
Ähnlichkeiten zu C
Besonderheiten
Ownership

3 Beispiele
Parallel Computing
Memory Safety

4 Zusammenfassung

5 Literatur

6 Anhang

Rust Mission Statement

Rust is a **systems programming** language focused on three goals: **safety, speed, and concurrency**. [. . .] making it a useful language for a number of use cases other languages aren't good at: **embedding** in other languages, programs with specific **space and time requirements**, and writing **low-level code**, like device drivers and operating systems. It improves on current languages targeting this space by having a number of **compile-time safety checks** that produce no runtime overhead, while eliminating all **data races.** [1][. . .]

[1]`https://doc.rust-lang.org/1.8.0/book/`

Vergleich zu anderen Sprachen

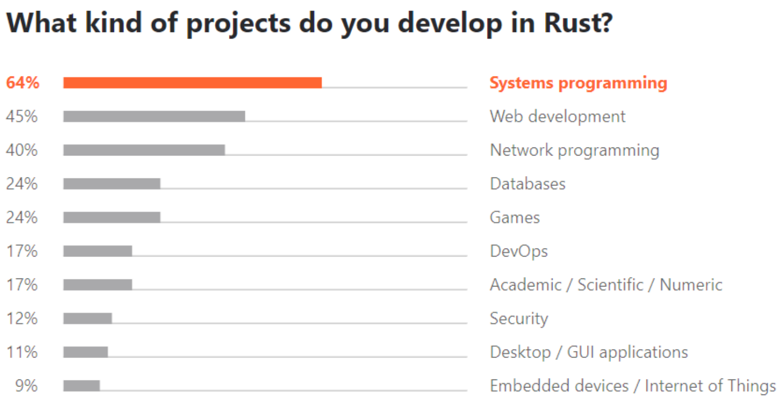
	Rust	Python	C++	Go	Fortran
Seit	2010	1990	1985	2009	1957
Sicherheit	+ +	+	– –	+	–
Geschwindigkeit[2]	++	– –	++	o	+
Nebenläufigkeit[3]	++	-	+	++	+
Zugang[4]	–	++	–	o	–

[2] https://benchmarksgame-team.pages.debian.net/benchmarksgame/which-programs-are-fastest.html

[3] https://www.sas.upenn.edu/~jesusfv/Lecture_HPC_5_Scientific_Computing_Languages.pdf

[4] https://raygun.com/blog/programming-languages/

Typische Anwendungsfelder[5]



[5] https://www.jetbrains.com/lp/devecosystem-2019/rust/

Personal Project to Most beloved language

- Idee und erste Umsetzung durch Graydon Hoare in 2006
- Gesponsort durch Mozilla seit 2009
- Version 1.0 in 2015 [6]
    - "Rust hat mehr Features in der Entwicklung verworfen, als andere Sprachen ingesamt haben"
    - Veröffentlichungen im sechs-Wochen Rhythmus
- Beliebteste Sprache auf Stack Overflow seit 2016 [7]

[6] `http://steveklabnik.github.io/history-of-rust/`

[7] `https://stackoverflow.blog/2019/04/09/the-2019-stack-overflow-developer-survey-results-are-in/`

Browser bis Betriebssystem

- Mozilla Firefox – 8,6% SLOC in Rust[8]
- Dropbox Diskotech – file storage core[9]
- Redox – Unix-like microkernel, komplett in Rust[10]
- Tor Netzwerk – Portierung auf Rust[11]
- Discord – Wechsel von Go zu Rust (kein GC)[12]
- yelp – real time AB testing[13]

[8] https://4e6.github.io/firefox-lang-stats/

[9] https://www.wired.com/2016/03/epic-story-dropboxs-exodus-amazon-cloud-empire/

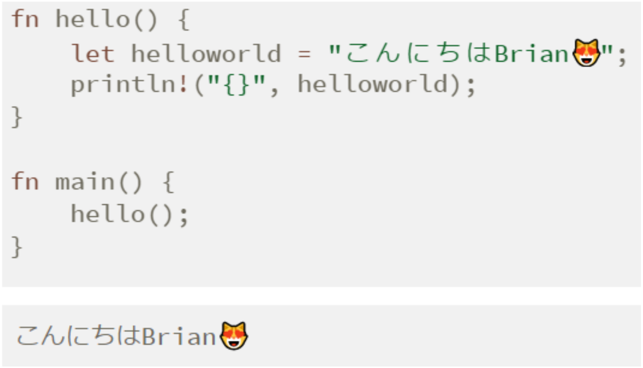
[10] https://redox-os.org/

[11] https://lists.torproject.org/pipermail/tor-dev/2017-March/012088.html

[12] https://blog.discord.com/why-discord-is-switching-from-go-to-rust-a190bbca2b1f

[13] https://www.youtube.com/watch?v=u6ZbF4apABk

Allgemeiner Aufbau



```
fn hello() {
    let helloworld = "こんにちはBrian😻";
    println!("{}", helloworld);
}

fn main() {
    hello();
}
```

```
こんにちはBrian😻
```

Abbildung: Hello Mr. Kernighan

Kontrollstrukturen

```
fn main() {
    let x : [u32; 3] = [47, 5, 23];
    let mut y : bool = true;
    for &n in &x {
        if n > 23 {
            println!("{}", n);
        }
        else {
            y = false;
        }
    }
}
```

Typensystem

```
fn main() {
    let mut x = 9000.0;
    x += 1f64;
    if x > 9000 {
        println!("It's over {}!", x);
    }
}
```

Compiler Fehler

```
error[E0308]: mismatched types
--> src/main.rs:4:12
    |
4 |     if x > 9000 {
    |            ^^^^
    |            |
    |            expected `f64`, found integer
    |            help: use a float literal: '9000.0'

error: aborting due to previous error

For more information about this error,
try 'rustc --explain E0308'.
error: could not compile 'dragonball'.
To learn more, run the command again with --verbose.
```

Cargo – crates.io

- Cargo:
  - Package & Dependency Manager
  - Package Builder
  - Testing Setup
  - Documentation Builder
  - Git bei Projekterstellung
- Crates.io
  - community package registry
  - 39.714+ crates
  - 2,803,740,091+ Downloads

Verschieben und Ausleihen

```
struct Pointer {a: i32, b: i32,}

fn main() {
    let x  = Pointer{a: 23, b: 5};
    let y =  x;
    println!("x.a: {}, x.b: {}", x.a, x.b);
    println!("y.a: {}, y.b: {}", y.a, y.b);
}
```

Compiler Fehler

```
error[E0382]: borrow of moved value: `x`
--> src/main.rs:6:39
    |
4 |     let x  = Pointer{a: 23, b: 5};
    |           - move occurs because `x` has type `Pointer`,
                which does not implement the `Copy` trait
5 |     let y =  x;
    |              - value moved here
6 |     println!("x.a: {}, x.b: {}", x.a, x.b);
    |                                   ^^^ value borrowed
                                          here after move
```

Verschieben und Ausleihen - Korrektur

```
struct Pointer {a: i32, b: i32,}

fn main() {
    let x  = Pointer{a: 23, b: 5};
    let y =  &x;
    println!("x.a: {}, x.b: {}", x.a, x.b);
    println!("y.a: {}, y.b: {}", y.a, y.b);
}
```

Parallelisierung

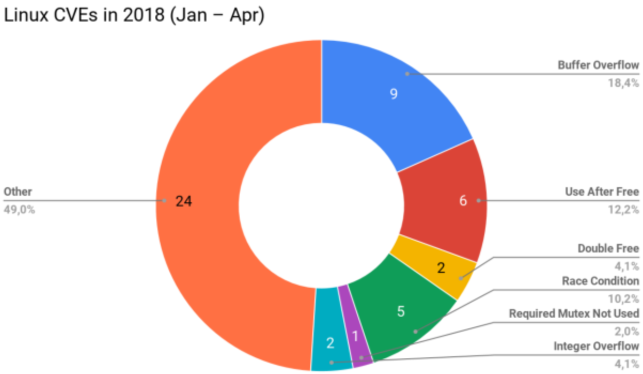
```rust
extern crate rayon;
use rayon::prelude::*;
extern crate rand;
use rand::Rng;
fn main() {
  const VEC_SIZE: usize = 10_000_000;
  let mut vec1: Vec<f64> = (0..VEC_SIZE)
    .map(|_| rng.gen::<f64>()).collect();
  let mut vec2 = vec1.clone();
  vec1.iter_mut()
    .for_each(|p| *p *= p.sin() * p.cos()); // serial
  vec2.par_iter_mut()
    .for_each(|p| *p *= p.sin() * p.cos()); // parallel
}
```

Performance-Vergleich

- Random Double-Array mit 10.000.000 Einträgen.
- Berechne pro Ellement $p = p * (\cos(p) * \sin(p) - \arccos(p))$
- Gesamtzeit zur Berechnung in Sekunden ohne Set-Up:

Sprache	Seriell	Parallel
Rust	0.483	0.196
C	1.114	0.432
Python	16.8	-

Häufigste Fehlerquellen – Common Vulnerabilities and Exposures [14]



[14] https://phil-opp.github.io/talk-konstanz-may-2018/

Double Free, Use after Free

- Lebenszeit festgelegt durch Compiler
    - Lifetime Attribute
    - Scope
- Keine manuelle Allokation

Integer Overflow 1

```
fn main(){
    let x: i8 = 127;
    let _y: i8 = &x + 256;
}
```

Compiler Fehler

```
 error: literal out of range for `i8`
 --> src/main.rs:3:23
  |
3 |     let _y: i8 = &x + 256;
  |                       ^^^
  |
  = note: `#[deny(overflowing_literals)]` on by default

error: aborting due to previous error
```

Integer Overflow 2

```
use std::io;

fn main(){
    let x:i8 = 127;
    let mut guess = String::new();

    io::stdin()
        .read_line(&mut guess)
        .expect("Failed to read line");

    let guess: i8 = guess.trim().parse().expect("ERROR");
    let mut _y = x + guess;
}
```

Runtime Fehler – Panik

```
thread 'main' panicked at 'attempt to add
 with overflow', .\overflow.rs:12:18
note: run with `RUST_BACKTRACE=1` environment
 variable to display a backtrace
```

Warum Rust benutzen?

- Schnelligkeit, Sicherheit, Nebenläufigkeit
- Infrastruktur - Cargo / crates.io
- Community - Doku / Compiler

Informationen

- Webseite: `https://www.rust-lang.org/`
- Einführung: The Rust Programming Language - Steve Klabnik, Carol Nichols (2019)
- Dokumentation: `https://www.rust-lang.org/learn`
- Discord: `https://discord.gg/rust-lang`
- Twitter: `https://twitter.com/rustlang`
- Github: `https://github.com/rust-lang`
    - 2,900+ contributers, 119,600+ commits, 44,500+ stars

Quellen I

[1] `https://doc.rust-lang.org/1.8.0/book/`

[2] `https://benchmarksgame-team.pages.debian.net/benchmarksgame/which-programs-are-fastest.html`

[3] `https://www.sas.upenn.edu/~jesusfv/Lecture_HPC_5_Scientific_Computing_Languages.pdf`

[4] `https://raygun.com/blog/programming-languages/`

[5] `https://www.jetbrains.com/lp/devecosystem-2019/rust/`

[6] `http://steveklabnik.github.io/history-of-rust/`

[7] `https://stackoverflow.blog/2019/04/09/the-2019-stack-overflow-developer-survey-results-are-in/`

Quellen II

[8] https://4e6.github.io/firefox-lang-stats/

[9] https://www.wired.com/2016/03/epic-story-dropboxs-exodus-amazon-cloud-empire/

[10] https://redox-os.org/

[11] https://lists.torproject.org/pipermail/tor-dev/2017-March/012088.html

[12] https://blog.discord.com/why-discord-is-switching-from-go-to-rust-a190bbca2b1f

[13] https://www.youtube.com/watch?v=u6ZbF4apABk

[14] https://phil-opp.github.io/talk-konstanz-may-2018/

Python Code

```python
import random
import math
import time
vec_size = 10000000
arr = [random.random() for _ in range(vec_size)]
start_time = time.time()
for p in arr:
    p *= math.sin(p) * math.cos(p) - math.acos(p)
end_time = time.time()
print("Serial: {:.5f} seconds".format(end_time - start_time))
```

C Code mit OML (1 – Setup)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <omp.h>

int main(){
    srand(time(0));
    int VEC_SIZE = 10000000;
    double *arr = (double*)malloc(VEC_SIZE * sizeof(double));
    double *arr2 = (double*)malloc(VEC_SIZE * sizeof(double));
    int i;
    for (i = 0; i < VEC_SIZE; ++i)
    {
    arr[i] = (double)rand()/(double)(RAND_MAX);
    arr2[i] = arr[i];
    }
```

C Code mit OML (2 – seriell)

```
struct timespec start, finish;
clock_gettime(CLOCK_MONOTONIC, &start);
{
    for (i = 0; i < VEC_SIZE; ++i)
    {
    arr[i] = sin(arr[i]) * cos(arr[i]) - acos(arr[i]);
    }

}
clock_gettime(CLOCK_MONOTONIC, &finish);
double time_taken;
time_taken = (finish.tv_sec - start.tv_sec) * 1e9;
time_taken = (time_taken + (finish.tv_nsec - start.tv_nsec)) * 1e-9;
printf("C Single thread: %f seconds\n", time_taken);
```

C Code mit OML (3 – parallel)

```
    clock_gettime(CLOCK_MONOTONIC, &start);
    #pragma omp parallel
    {
        #pragma omp for schedule(static,VEC_SIZE/4)
        for (i = 0; i < VEC_SIZE; ++i)
        {
            arr2[i] = sin(arr2[i]) * cos(arr2[i]) - acos(arr2[i]);
        }
    }
    clock_gettime(CLOCK_MONOTONIC, &finish);

    time_taken = (finish.tv_sec - start.tv_sec) * 1e9;
    time_taken = (time_taken + (finish.tv_nsec - start.tv_nsec)) * 1e-9;
    printf("C Parallel thread: %f seconds\n", time_taken);

    return 0;
}
```