

Rust für das Betriebssystem

Dr. Peter Dillinger
peter.dillinger@email.de

Inhalt

- Anforderungen an eine Betriebssystemprogrammiersprache
- Die Programmiersprache Rust
- Sicherheitskonzepte in Rust
- Zusammenfassung
- Literatur und Quellen



Digitales Handout

<https://github.com/phd4hd/HSDA>

Anforderungen an eine Programmiersprache

- Hardware-nahe Ausführung
 - CPU spezifische Einstellungen (Stack, IRQ, Page Descriptor Tables)
 - Ohne statische Bindung (Bare-Metal)
- Performant
 - Effiziente Code-Ausführung
 - Reduzierung des Overhead
 - Minimaler Footprint
- Sicherheit
 - Schutz vor Programmierfehler
 - Schutz vor Schwachstellen

Die Programmiersprache Rust

- Systemprogrammiersprache
- Multiparadigmatisch
 - Generisch
 - Nebenläufig
 - Funktional
 - Imperativ
 - Strukturiert
- Sicher
- Praxisnah

```
fn main() {  
    println!("Hello, world!");  
}
```

```
fn fakultaet(i: u64) -> Option<u64> {  
    match i {  
        0 => Some(1),  
        n => match fakultaet(n - 1) {  
            Some(m) => n.checked_mul(m),  
            None => None  
        }  
    }  
}
```

Sicherheitskonzepte in Rust

- Datentypkonvertierung

```
fn main() {  
    println!("Verträglichkeit der Datentypen");  
  
    let x : i8  = 127;  
    let y : i32 = x;  
    let z : f64 = 10;  
  
    println!("x = {}, y = {}, z = {}", x, y, z);  
}
```

Sicherheitskonzepte in Rust

- Ownership

```
fn main() {  
    println!("Wem gehört der String?");  
  
    let s1: String = "Hallo".to_string();  
    let s2: String = s1;  
  
    println!("{}", s1 == s2);  
}
```

Sicherheitskonzepte in Rust

- Referenzen (Pointer)

```
fn pointer() -> &i32 {  
    let N : i32 = 42;  
    return N;  
}
```

Sicherheitskonzepte in Rust

- Speicherverwaltung

```
use std::mem::drop; // ähnlich zu free()

fn main() {

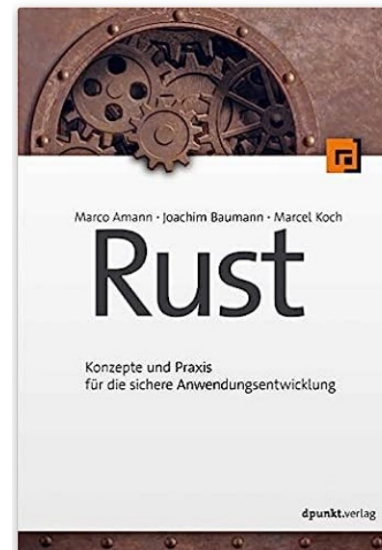
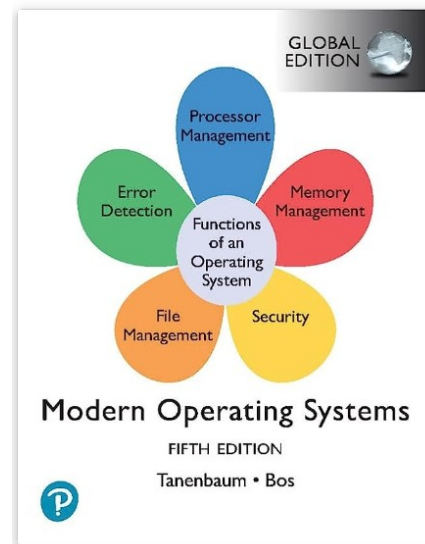
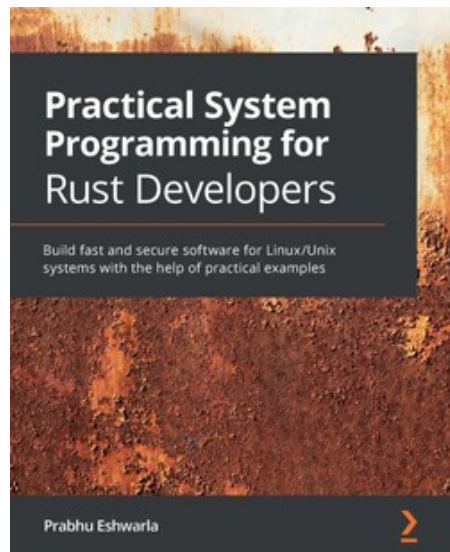
    let s = "Hallo".to_string();
    drop(s);

    println!("s = {}", s);
}
```


Zusammenfassung

- Rust erfüllt viele Anforderungen, ein Betriebssystem zu schreiben
- Linux setzt Rust ein
Rust Programming Language To Land in Linux Kernel 6.1
- Windows setzt Rust ein
Microsoft is busy rewriting core Windows code in memory-safe Rust
- Neues Betriebssystem komplett in Rust
Redox OS: an Operating System Written in Rust







Literatur und Quellen



Rust Playground
(Online Editor & Compiler)



CS 242: Programming Languages,
Fall 2018: Memory safety in Rust

| BEST PROGRAMMING LANGUAGE TO WRITE AN OPERATING SYSTEM | | PRICE | CURRENT STABLE VERSION | TYPING DISCIPLINE |
|--|---|-------|------------------------|--------------------------------------|
| -- |  C | FREE | C'18 | static |
| -- |  Rust www.rust-lang.org | - | 1.66 | - |
| -- |  Nim nim-lang.org | - | 1.6.12 | - |
| -- |  V vlang.io | FREE | - | Strong, static, inferred, structu... |
| -- |  Zig ziglang.org | FREE | 0.10.1 | - |
| -- |  Assembly | - | - | - |

<https://www.slant.co/topics/20492/~programming-language-to-write-an-operating-system>