組み込み Rust 講習会

Rust 編

JIJINBEI

2024-07-20

HiCoder

Table of contents

Rust とは

Rust とは

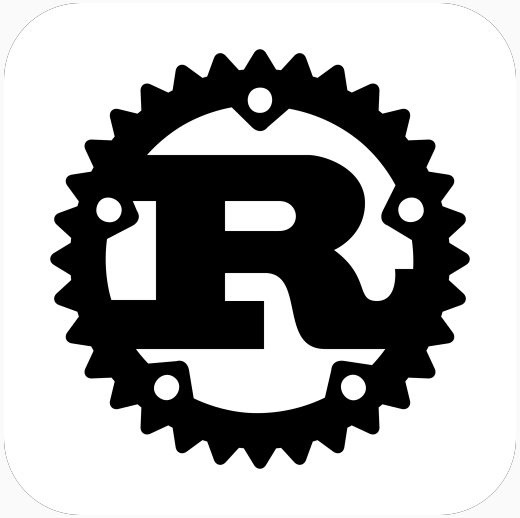
- Mozilla が開発したプログラミング言語
- Rust は最も愛されている言語として 7 年目を迎え、87% の開発者が使い続けたいと答えている。 [link]
- 最近のオープンソースはすべて Rust で書かれていると言っても過言ではない[1]

Rust is God

[1] このスライドも Rust 製の Typst で作成

Rust が好まれている理由

- 安全性
  - 型安全性
  - メモリ安全性
    - **所有権**
    - **ライフタイム**
- 処理速度の速さ
  - VM を使わない
  - メモリ管理を自動で行わない
- 並行処理
  - 所有権によるデータ競合の防止
- バージョンやパッケージ管理
  - cargo



変数

変数

`let` で変数の定義

```rust
fn main() {
    let x = 5;
    println!("The value of x is: {}", x);
    x = 6; // ERROR
    println!("The value of x is: {}", x);
}
```

変数を可変するには、`mut` を使う

```rust
fn main() {
    let mut x = 5;
    println!("The value of x is: {}", x);
    x = 6;
    println!("The value of x is: {}", x);
}
```

データ型

データ型(数値)

- Rust は静的型付け言語

大きさ	符号付き	符号なし
8-bit	i8	u8
16-bit	i16	u16
32-bit	i32	u32
64-bit	i64	u64
arch	isize	usize

Table 1: 整数型

大きさ	浮動小数点
32-bit	f32
64-bit	f64

Table 2: 浮動小数点型

配列

配列

- 配列は同じ型の要素を持つ

```rust
fn main() {
    let a: [i32; 5] = [1, 2, 3, 4, 5];
}
```

`i32` が 5 つの要素を持つ配列 `a` を定義

所有権

所有権の例(変数の所有権の移動)

```rust
fn main() {
    let s1 = String::from("hello");
    let s2 = s1;

    // println!("{}, world!", s1); // value borrowed here after move

    println!("{}, world!", s2);
}
```

参考文献

参考文献

- Rust 公式ドキュメント [link]