

Introduction to Rust Programming Language

Ahmad Saugi (2440078696)







Who is Rust For?

Students

Rust is for students and those who are interested in learning about systems concepts. Using Rust, many people have learned about topics like operating systems development.

Companies

Hundreds of companies, large and small, use Rust in production for a variety of tasks.

Ø Open Source Developers

Rust is for people who want to build the Rust programming language, community, developer tools, and libraries.

People Who Value Speed and Stability

Rust is fast. By speed, we mean both how quickly Rust code can run and the speed at which Rust lets you write programs.

Cargo

A build system and package manager for Rust.

Cargo responsibility:

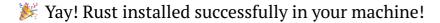
- Building your code
- Downloading the libraries your code depends on
- Updating and removing dependencies

Rust Installation

We install Rust using rustup, a tool to manage Rust versions.

You can install Rust by running this command on terminal:

```
curl --proto '=https' --tlsv1.2 https://sh.rustup.rs -sSf | sh
```



Hello World Rust

- 1. Open terminal
- 2. Create new Rust project. Run in terminal:

```
cargo init helloworld
```

3. Run the project

```
cd helloworld
cargo run
```

Variables and Mutability

Consider this Rust code:

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

Will it run successfully?

Variables and Mutability

.

The answer is NO! It won't run.

In Rust, we need to write the mutable variable explicitly by adding mut keyword when creating a variable.

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

Looping

There are several ways to make looping in Rust:

```
// loop
let mut count = 1;
loop {
  println!("{count}");
  count += 1;
  if count == 5 {
    break;
// while loop
let mut count = 1;
while count <= 5 {</pre>
  println!("{count}");
  count += 1;
// for loop
for count in 1..6 {
  println!("{count}");
```

Match

Match is similar to switch in other programming languages.

```
let number = 11;
println!("Tell me about {}", number);
match number {
    1 => println!("One!"),
    2 | 3 | 5 | 7 | 11 => println!("This is a prime"),
    13..=19 => println!("A teen"),
    _ => println!("Ain't special"),
// output: This is a prime
let boolean = true;
let binary = match boolean {
    false => println!("salah!"),
    true => println!("benar!"),
};
// output: benar!
```

Guessing Game

Let's make a guessing game!

```
use rand::Rng;
use std::cmp::Ordering;
use std::io;
fn main() {
    println!("Guess the number!");
    let secret number = rand::thread rng().gen range(1..=100);
    loop {
        println!("Please input your guess.");
        let mut guess = String::new();
        io::stdin()
            .read_line(&mut guess)
            .expect("Failed to read line");
        let guess: u32 = match guess.trim().parse() {
            0k(num) => num,
            Err( ) => continue,
        };
```

Thank You

Powered by Slidev