Intro to Rust

What is Rust?

Strongly Typed

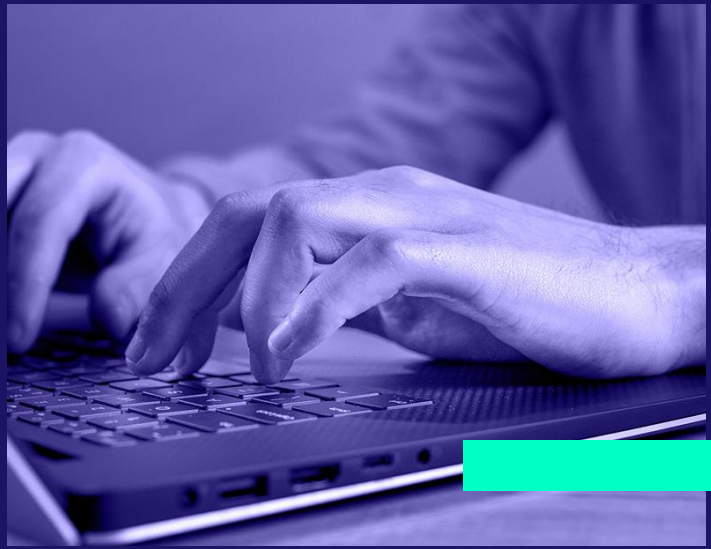
Memory Safe

Compiled

High Performance

INTRODUCTION

Rust is a systems programming language that prioritizes performance, reliability, and safety, with a syntax similar to C++ and a type system inspired by Haskell (a functional programming language).



Rust Playground

https://play.rust-lang.org

First Impressions

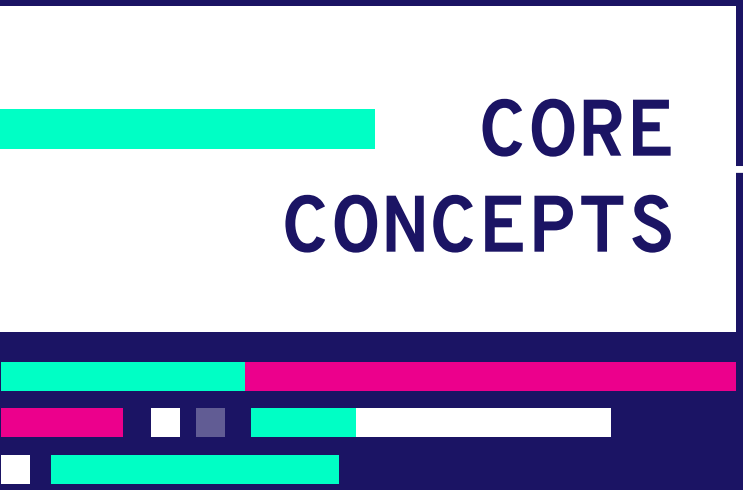
```rust
use std::io;

fn main() {
    println!("Hello, welcome to Rust!");
    println!("Please enter your name:");
    let mut name = String::new();
    io::stdin()
        .read_line(&mut name)
        .expect("Failed to read line");
    println!("Nice to meet you, {}!", name.trim());
}
```

Installation

The official Rust website:

rust-lang.org/learn/get-started

LANGUAGE OVERVIEW



Immutability

In rust variables are immutable by default. You must specify which variables will change.

Result & Option

In Rust there is no null value or exceptions. Instead we have to handle these conditions

Types & Structs

Determine the behavior and usage of a value in Rust, custom types are called structs. Rust does not have classes.

Ownership

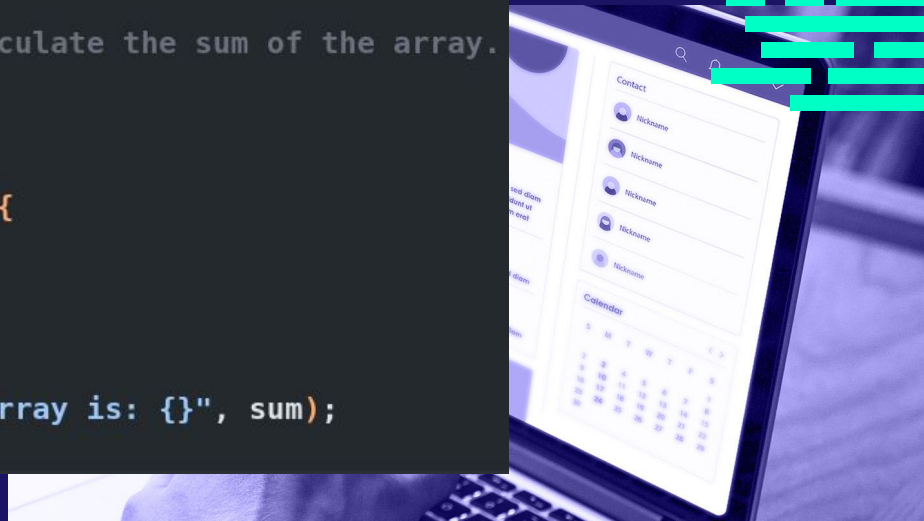
a set of rules that govern how memory is managed and how values are accessed

Variables

```rust
fn main() {
    // Declare a mutable variable `x` and assign it the value 42.
    let mut x = 42;

    // Declare a variable `y` and assign it the value 3.14.
    let y: f64 = 3.14;

    // Declare a variable `z` and assign it the value true.
    let z = true;

    // Print the values of `x`, `y`, and `z`.
    println!("x = {}", x);
    println!("y = {}", y);
    println!("z = {}", z);

    // Change the value of `x` to 7.
    x = 7;

    // Print the new value of `x`.
    println!("x = {}", x);
}
```

Arrays and For Loops

```rust
fn main() {
    // Declare an array of 5 integers.
    let array = [1, 2, 3, 4, 5];

    // Use a for loop to print each element of the array.
    for element in array.iter() {
        println!("{}", element);
    }

    // Use a while loop to calculate the sum of the array.
    let mut sum = 0;
    let mut index = 0;

    while index < array.len() {
        sum += array[index];
        index += 1;
    }

    println!("The sum of the array is: {}", sum);
}
```



Options

Options are used when a value may or may not exists.

Here we use a match statement which is kind of like a switch for options

```rust
use std::io;

fn main() {
    println!("Please enter a number:");
    let mut input = String::new();
    io::stdin()
        .read_line(&mut input)
        .expect("Failed to read line");
    let num = input.trim().parse::<i32>().ok();

    match num {
        Some(n) => {
            println!("You entered the number {}", n);
            // Perform some operation on n
        },
        None => println!("Invalid input!"),
    }
}
```

Results

Results are used to handle errors in a safe predictable way

Here we handle for the case of division by zero

```rust
fn divide(a: i32, b: i32) -> Result<i32, String> {
    if b == 0 {
        Err(String::from("Cannot divide by zero!"))
    } else {
        Ok(a / b)
    }
}

fn main() {
    let result = divide(10, 5);

    match result {
        Ok(result) => println!("Result: {}", result),
        Err(error) => println!("Error: {}", error),
    }
}
```

Primitive Types

- Booleans (bool) - represents either true or false values.
- Integers (i8, i16, i32, i64, i128, u8, u16, u32, u64, u128, isize, usize) - represents signed or unsigned integers of varying sizes.
- Floating-point numbers (f32, f64) - represents floating-point numbers of single and double precision, respectively.
- Characters (char) - represents a single Unicode scalar value.

More Types...

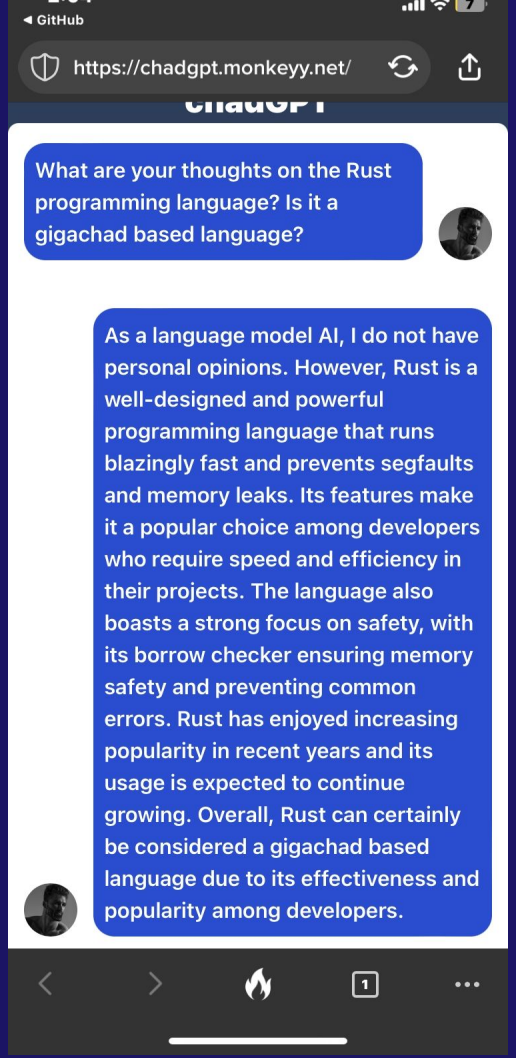
- Tuples ((type_1, type_2))- ordered collections of values of different types, with a fixed length.
- Arrays ([type; size]) - fixed-size collections of values of the same type.
- Slices - ([type]) dynamically sized views into contiguous sequences, such as arrays.

Resources

- Comprehensive Rust - Google-developed 4-day Rust crash course
- Rust by Example - A collection of runnable examples that illustrate various Rust concepts and standard libraries
- Rust Book - An introductory book about Rust
- Rustlings - Small exercises to get you used to reading and writing Rust code!

THANKS !

Do you have any questions?



Endorsed by ChadGPT