

CPSC-354 Report

Michael Masakayan
Chapman University

December 15, 2022

Abstract

Short summary of purpose and content.

Contents

1	Introduction	1
2	Project	1
2.1	Project Outline	1
2.2	Project deliverable 1	2
2.3	Project deliverable 2 Introduction on the language	2
2.3.1	Types	3
2.3.2	Compound Types	3
2.4	Project deliverable 3	4
3	Conclusions	4

1 Introduction

¹ My name is Michael Masakayan, I'm a 4th year computer science major. I transferred from Pasadena City College in 2020. I am interested in cyber security and although I won't be able to get a minor in it I plan on getting a few certificates this year. I enjoy playing video games and like watching movies

2 Project

2.1 Project Outline

I decided to change the outlook of my project and pivot to study another programming language. I originally was going to do a graphical project like plotting Mandelbrot sets but I think that I will be able to show more progress if I apply what we have learned to another language. Now describing the my new project. I originally thought I was going to do a visual project, something like plotting Mandelbrot sets but instead I am going to change to learning and explaining a new programming language Rust. My milestones will include

- Going over the history of the language. Mainly who made it, why it was made, and how it was made. (I think that I will be able to finish this section within the next 2 weeks)

- Short introduction into how to get started in the language. I would like to go over a short snippet of code that maybe I have made and how the language interprets different data types and structures. (I think that I could finish this within 3 or 4 weeks seeing as I most likely need to be comfortable with the language to do this section)
- It's pros and cons, and it's real world application. I would like to go over the trade offs of the language and why people are starting to use this over others. (I think I can complete this with the first mile stone within the first 2/3 weeks.)
- I would also like to talk about how the language is connected to other languages and my experience with it. This is not as important to me as the other mile stones. I think it would be good to round it off and talk about how it connects in actual workflows and working with other languages. (I should be able to complete this within 2 weeks but I do not this is as important of a section and I think I should come back to it if I have time.)

If there are any suggestions on topics I should cover or another language that might be good to do the project in please let me know.

2.2 Project deliverable 1

- **Who made it?** Rust started out of a personal project from Graydon Hoare. He started working on the language. During Hoare's work at Mozilla they sponsored the project in 2009 and pushed ongoing development to be a part of their browser engineer project. And as of 2011 the newer Rust compiler successfully compiled itself. In May of 2015 the Rust Core Team announced Rust 1.0 would be the first stable build for rust. As said in the announcement this would mark the end of their churn and begin the phase of commitment to stability. There were some changes during the 2020 pandemic. In their Aug 18 2020 announcement they addressed the concerns that Rust would be abandoned, they said they would be taking the foundation seriously and planned on getting the trademarks and domain names.
- **Why it was made?** According to Hoare's he wanted to revive some of the good ideas from the early 80's and late 70's competitors. He said that it was also needed because of the more recent consciousness of security people have.

2.3 Project deliverable 2 Introduction on the language

Firstly you will want to go to the official website to install rust. It should be at the time of writing this.[\[IR\]](#) After you download rust you should run "rustc" and see if the help prompt comes up. The extension for the language is .rs so you will be able to make a file "main.rs".

It is important to note Cargo and its relation to rust (cargo). It is pip for Python or gem for Ruby in that it is both the package manager and build system for Rust. Cargo is called when you run "rustc" and comes pre-installed when you download Rust.

The first program I will be going over is a basic "hello world". This is the code for the "hello world". As you can see in the code the function is denoted by fn.

```
fn main() {
    println!("hello World")
}
```

This is a concept of ownership with coding examples. In this example I will be showing how an immutable reference is borrowed. We take `Veci32` instead of just `Veci32` and pass in `v1` and `v2`. This is to call the type by reference rather than it owning the recourse it borrows the ownership. Since it is borrowing something it does not deallocate the recourse when it is out of the scope. Meaning after you call the function `foo()` we

can still use the original bindings of it. But references are immutable and the vectors that exist within `foo()` cannot be changed when we borrow it from outside of the scope. Here is the code example.

```
fn main() {
    // the point here is that an immutable reference is borrowed.
    fn sum_vec(v: &Vec<i32>) -> i32 {
        v.iter().fold(0, |a, &b| a + b)
    }
    // Borrow two vectors and sum them.
    // This kind of borrowing does not allow mutation through the borrowed reference.
    fn foo(v1: &Vec<i32>, v2: &Vec<i32>) -> i32 {
        // we do stuff with 'v1' and 'v2'.
        let s1 = sum_vec(v1);
        let s2 = sum_vec(v2);
        // this will return the answer
        s1 + s2
    }
    let v2 = vec![4, 5, 6];
    let v1 = vec![1, 2, 3];

    let answer = foo(&v1, &v2);
    println!("{}", answer);
}
```

2.3.1 Types

Very much like other programming languages rust has the basic types. [?]

Basic Types		
Name	Description	Example
Char	a single Unicode value that takes up 4 bytes	'a' 'h'
Boolean	Value that decides truth, two possible values	True or false
Integer	There are multiple integer data types in Rust. But the default integer type in Rust is i32	1,2,-1,100
Floating Point	There are two types for floating-point numbers f32(32 bit) and f64(64 bit)	2.0,3.0

2.3.2 Compound Types

There are also two compound types.

Compound Types		
Name	Description	Example
Tuples	Fixed data types that can contain multiple types within it. In this example there is a char, int, and float within this tuple.	('a', 1, 2.4)
Arrays	As said before arrays can only use one specific data type within them unlike tuples. Arrays within Rust are different from that in other languages. For example they have fixed lengths.	let a = [1, 2, 3, 4, 5];

2.4 Project deliverable 3

Rust is a programming language that is gaining popularity in the industry due to its focus on safety, performance, and concurrency. It is commonly used for building high-performance systems and is often a good choice for applications that require low-level control, such as operating systems, game engines, and embedded systems. Additionally, Rust has a strong ecosystem of libraries and tools, making it easy to develop and maintain complex projects. Many companies, such as Figma and Dropbox, have adopted Rust in their tech, and the language has received high praise from developers. Here are a few examples of how Rust has been used:

- Dropbox uses Rust for its file synchronization engine to improve performance and handle complex codebases
- Coursera uses Rust for its programming assignments feature because it is more secure than C
- Figma uses Rust to improve performance of its multiplayer syncing engine
- NPM uses Rust for its main service to improve performance and scalability
- Microsoft is experimenting with integrating Rust into its C/C++ codebases for improved memory safety.
- Facebook used Rust to rewrite the source control backend. Rust was also adopted for its safety
- Discord used Rust for some of their codebase on server-side and for client side. They said that rust had lots of advantages for their engineering team. Stating that its borrow checker and type system make it easy to refactor code

3 Conclusions

(approx 400 words)

In the conclusion, I want a critical reflection on the content of the course. Step back from the technical details. How does the course fit into the wider world of programming languages and software engineering?

References

- [PL] [Programming Languages 2022](#), Chapman University, 2022.
- [AR] “Announcing Rust 1.0: Rust Blog.” [The Rust Programming Language Blog](#),
- [IR] “Install Rust.” [Rust Programming Language](#)

- [RL] Rust-Lang. “[Rust/Releases.md at Master · Rust-Lang/Rust.](#)” GitHub, 4 Nov. 2022.
- [PDT] Rust-Lang. “<https://learning-rust.github.io/docs/primitive-data-types/>”.