Rust Setup

Lecture 1

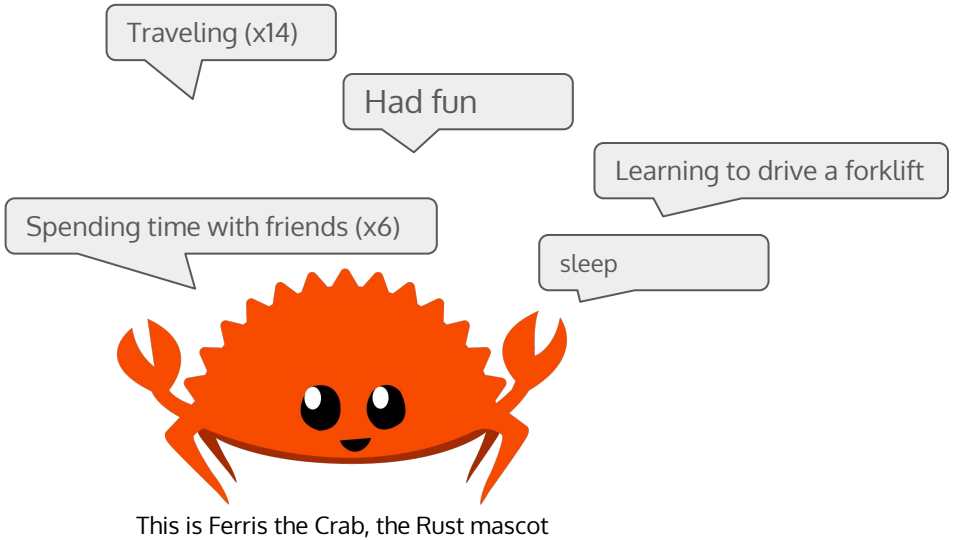
Slides: Neil Kaushikkar (Adapted from William Eustis)

Goals For Today

- Discuss more about the course
- Explore the Rust environment
- Hello, CS 128 Honors!

But First...

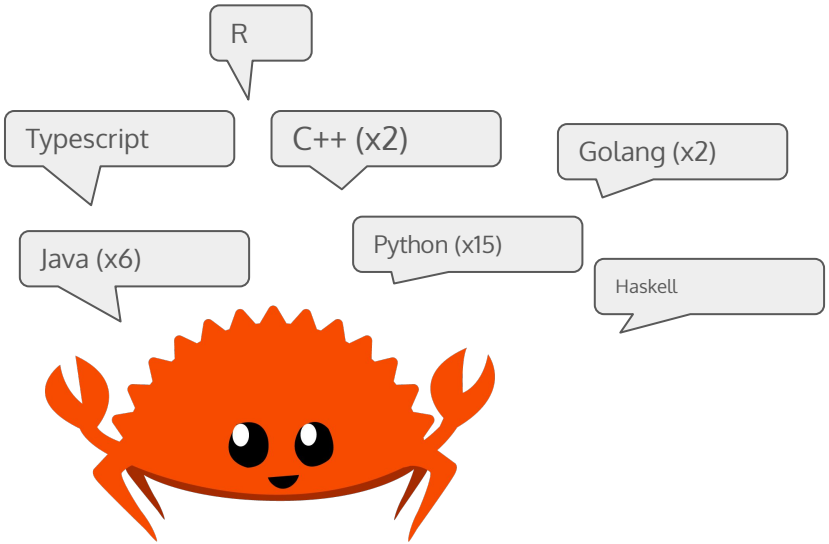
What did you do this summer?



This is Ferris the Crab, the Rust mascot

But First...

What is your favorite programming language to work with?



This is Ferris the Crab, the Rust mascot

But First...

What are your favorite pastas?

Spaghetti (x5)

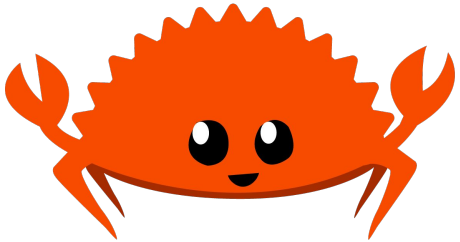
Tagliatelle

Penne (x5)

Rotini

calamarata

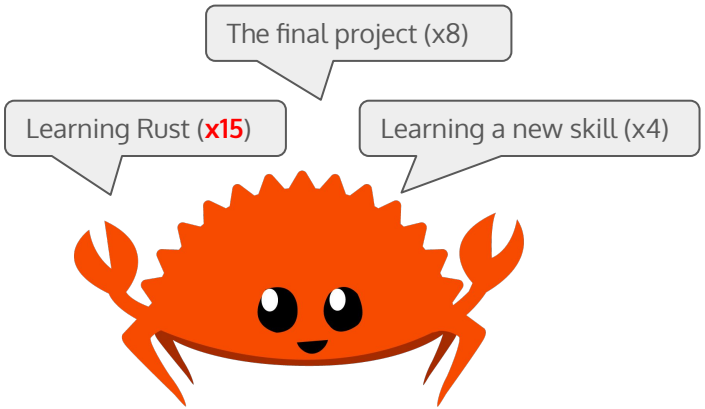
Farfalle



This is Ferris the Crab, the Rust mascot

But First...

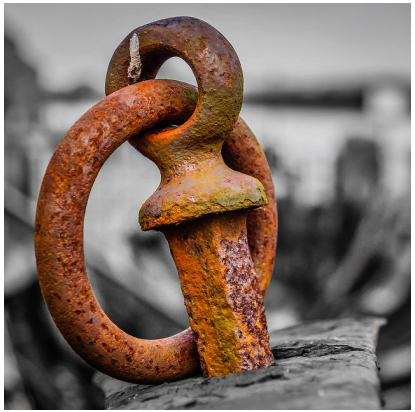
What are you most looking forward to in the course?



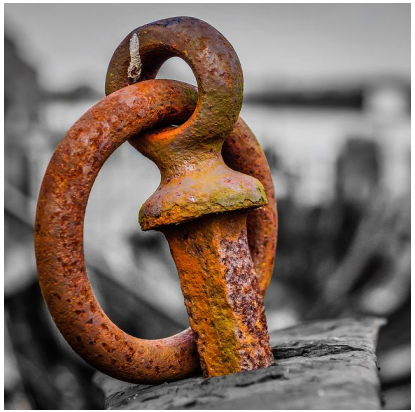
This is Ferris the Crab, the Rust mascot

So... What is Rust?

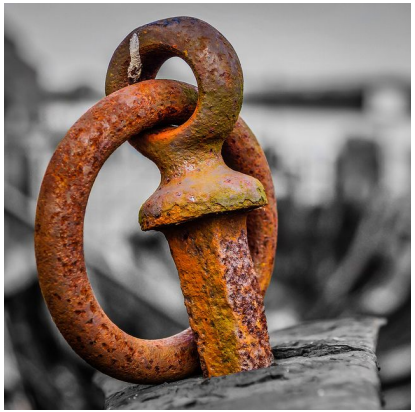
So... What is Rust?



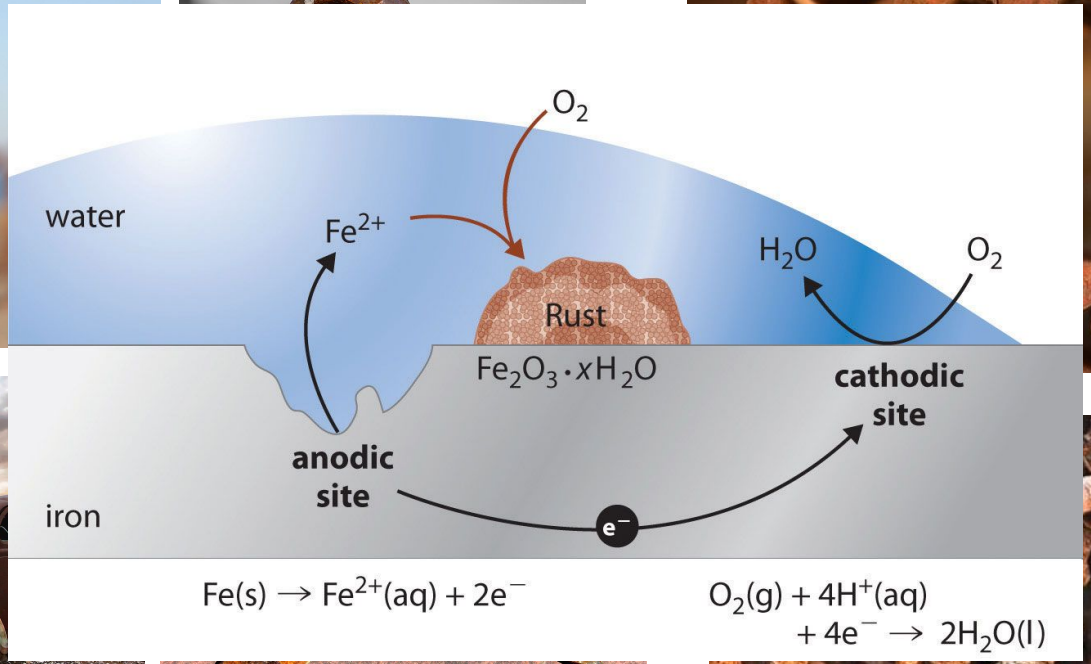
So... What is Rust?

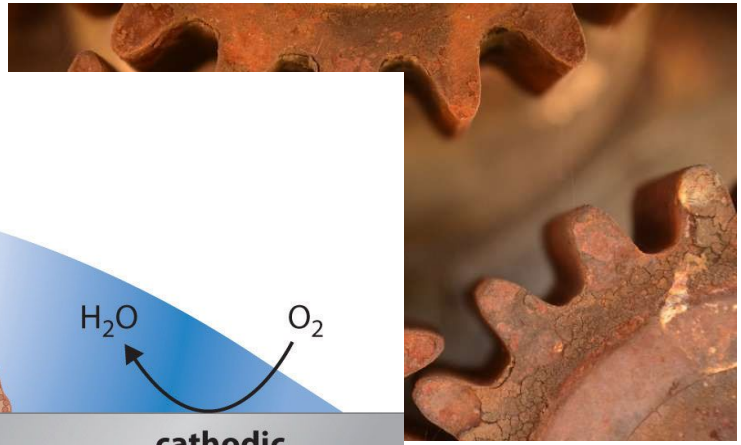


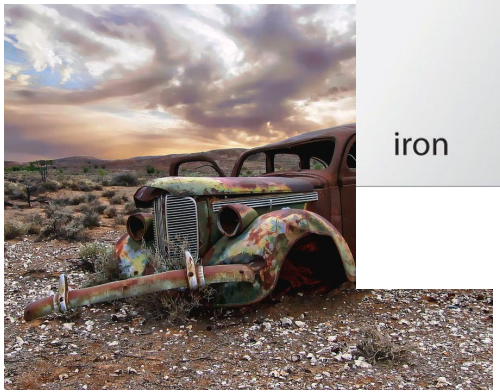
So... What is Rust?

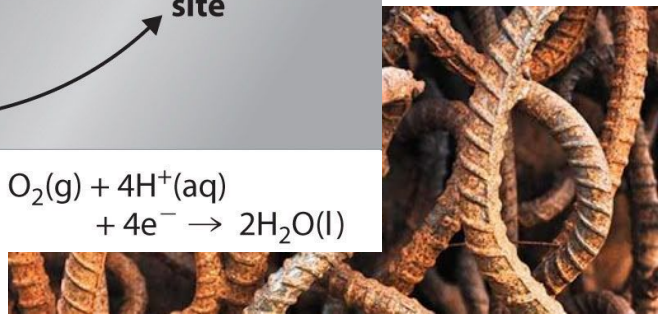


So... What is Rust?







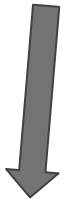


So... What is Rust?

- Rust is a **multi-paradigm system programming language**

So... What is Rust?

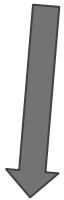
- Rust is a **multi-paradigm** **system programming** language



Programming Paradigms:
Categorizing programming languages based on their features

So... What is Rust?

- Rust is a **multi-paradigm** **system programming** language

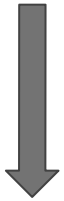


Programming Paradigms:
Categorizing programming languages based on their features

Multi-Paradigm:
Languages that support several programming paradigms and don't restrict the programmer to a single paradigm

So... What is Rust?

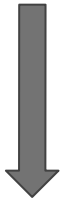
- Rust is a **multi-paradigm** **system programming** language



Systems Programming Language:
Systems programming aims to produce software and software platforms which provide services to other software, are performance constrained, or both

So... What is Rust?

- Rust is a **multi-paradigm** **system programming** language



Systems Programming Language:
Systems programming aims to produce software and software platforms which provide services to other software, are performance constrained, or both

System programming languages are designed to provide easy **access to the underlying hardware** and for **performance**

How will we learn Rust?

- We will often refer to the Rust textbook in our lectures
  - Google “Rust Textbook” - it’s free online
  - https://doc.rust-lang.org/book/

How will we learn Rust?

- Rust is already installed in your CS128 virtual machine
- Please test this whenever possible with **cargo -V**

```
vagrant@0d603b65c801:~$ cargo -V
cargo 1.57.0 (b2e52d7ca 2021-10-21)
vagrant@0d603b65c801:~$
```

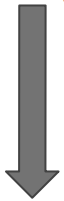
Cargo?

Cargo?

Cargo?

Cargo?

- Cargo is Rust's **build system** and **package manager**

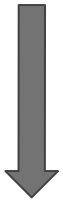


Build System:
Cargo **compiles*** your Rust project so that it can be executed by your computer

* Technically rustc is Rust's Compiler. Cargo acts as an abstraction layer that manages packages, handle metadata, and invokes rustc for you

Cargo?

- Cargo is Rust's **build system** and **package manager**



Package Manager:
Modern projects depend on external packages and libraries. A package manager handles those packages (and the packages that they might depend on)

Cargo?

- Cargo is Rust's **build system** and **package manager**
- How will we (primarily) use Cargo?
  - Compile and execute your code
  - Download and manage packages you use (final project!)

Hello, CS 128 Honors

Let's go through the process of creating some Rust code from scratch

1. Create a new project
2. Edit your code*
3. Compile
4. Run

* You should use whatever editor you want, but we recommend VSCode with the rust-analyzer extension

Seems Easy Enough?

Cargo Commands

- **cargo new** – creates a new Rust project
- **cargo build** – compile and build the code in your project
- **cargo run** – (build if not already built) & run the compiled code
- **cargo check** – check for syntax errors without compiling
- **cargo clean** – clear the project of all compiled sources

That’s All Folks!