CARGO CHEAT SHEET

RUST PACKAGE MANAGER

WHAT IS RUST AND WHY DO WE LOVE IT?

Rust was introduced in 2010 by Graydon Hoare of Mozilla Research. Rust is a low-level statically-typed multi-paradigm programming language that's focused on safety and performance, similar to C and C++, and is fast and memory-efficient with no garbage collections.

According to the StackOverflow surveys, Rust has been the most loved programming language for the last five years in a row!

What is Cargo?
Cargo is the Rust package manager. Cargo downloads your Rust package dependencies, compiles your packages, makes distributable packages, and uploads them to crates.io by default, the Rust community's package registry.

HOW TO INSTALL RUST AND CARGO

The easiest way to install Rust is with a tool called Rustup, which is a Rust installer and version management tool.

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

Rust updates are very frequent, so if you have installed Rustup some time ago, chances are your Rust version is out of date. To get the latest version of Rust run the following command:

```
$ rustup update
```

When you install Rustup you'll get the latest stable version of the Rust build tool and package manager, also known as Cargo.

HOW TO USE CARGO?

Cargo is a smart client that gives you the ability to build, run and test your projects. Here are some useful Cargo commands that take you through a typical workflow:

- Build your project:
  `$ cargo build`
- Run your project:
  `$ cargo run`
- Test your project:
  `$ cargo test`
- Build documentation for your project:
  `$ cargo doc`
- Publish a library to crates.io:
  `$ cargo publish`

CODING IN RUST

Import external crate

`extern crate rand;` //external dependency
`use rand::Rng; //bring Rng` trait which defines the methods into scope

Data Structures

`struct S {}` Define a struct with named fields.
`enum E {}` Define an enum
`const X: T = T();` Define constant
`static X: T = T();` Global variable static with a single memory location
`let x: T;` Allocate T bytes in stack bound as X assigned once not mutable

Control Flow

`while x {}` Loop , run while expression x is true.
`loop {}` Loop infinitely until break . Can yield value with break x .
`if x {} else {}` Conditional branch if expression is true.
`break` Break expression to exit a loop.
`continue` Continue expression to the next loop iteration of this loop.

SIMPLE HELLO WORLD EXAMPLE

Step 1: Create a new file called hello.rs, with the following:

```
fn main() {
 println!("Hello, world!");
}
```

Step 2: Compile it by running the following command:

```
$ rustc hello.rs
```

Run the binary that just created (hello) by running the following command:

```
$ ./hello
```

MOST DOWNLOADED CRATES

In Rust a library or executable program is called a crate. Crates are compiled using the Rust compiler, rustc. Here are the most downloaded crates*:

Rand - A Rust library for random number generation.

Syn - A parsing library for parsing a stream of Rust tokens into a syntax tree of Rust source code.

Libc - A library that provides all of the definitions necessary to easily interoperate with C code on each of the platforms that Rust supports.

Quote - A library that provides the quote! macro for turning Rust syntax tree data structures into tokens of source code.

Rand_core - Core traits and error types of the rand library, plus tools for implementing RNGs.

*Last updated on April 2021

CARGO & JFROG

Today JFrog Artifactory natively supports Cargo registries for the Rust programming language, providing full control of your deployment and resolution of Cargo packages.

A Cargo registry can be used to proxy remote Cargo resources and cache downloaded packages. Secure your packages using a **local Cargo** repository and **much more.**

Read more about Cargo repositories >