Rust

Matthias Colin

History

- 2006 - Personal project by Mozilla researcher **Graydon Hoare**
- 2009 - Sponsored by Mozilla
- 2015 - Version 1.0, 1st stable version
- 2021 - Rust Foundation by AWS, Google, Huawei, Microsoft, Mozilla
            foundation.rust-lang.org
- now - Version 1.70.0

- Website: www.rust-lang.org
- License: MIT and Apache 2.0
- Playground: https://play.rust-lang.org/

General

- Cross platform
- Paradigms: concurrent, functional, generic, imperative, structured
- Inspired by: Alef, C#, C++, Cyclone, Elm, Erlang, Haskell, Limbo, Newsqueak, OCaml, Ruby, Scheme, Standard ML, Swift
- Adopted by:
  - Amazon, Discord, Dropbox, Facebook (Meta), Google (Alphabet), and Microsoft.
  - Linux Kernel with C and Assembly

https://en.wikipedia.org/wiki/Rust_(programming_language)

Documentation

- Portail: https://www.rust-lang.org/learn
- Standard Library std
- Rust by example
- Rust book
- Cargo book
- Rust Editions Guide

Edition

- Rust 2015 = since Rust 1.0
- Rust 2018 = since Rust 1.31.0
  - Keywords: dyn (strict), async, await, try
- Rust 2021 = since Rust 1.56.0

Tools

- rustc: compiler
- cargo: packager

Primitive Types

- Integers:
  - Signed: i8, i16, i32, i64, i128: 12, -15, 128_255, 127i8, Oxff, 0o77, 0b1101, b'A'
  - Unsigned: u8, u16, u32, u64, u128: 12, 255u8
  - Pointer size: usize, isize
- Floats: f32, f64: 3.4, 3.4f32
- Text:
  - char (unicode): 'a','🦜', '\u{1F99C}'
  - str (slices)
- Boolean: bool: true, false
- References: &T, &mut T
- Pointers: *T, *mut T
- unit: ()
- never: !
- Arrays: [T;N]

Compound types

- Tuples: (u8, &str, f32): (255, "ok", 2.5)
- Structures
    - String
    - Vec
- Enumerations
    - Result<T,E>: Ok(v), Err(e) with v:T and e:E
    - Option<T>: Some(v) | None with v: T
- Traits

Print, format

- 2 traits: Display, Debug ({:?})
- macros:
  - println!, print! : print on stdout
  - eprintln!, eprint! : print on stderr
  - format! : output into string

Attributes

- #[derive(Debug)]
- #[allow(dead_code)]
- #[cfg(target_os = "linux")]
- #[cfg(not(target_os = "linux"))]

Collections

https://doc.rust-lang.org/std/collections/index.html

-

Async / await

https://rust-lang.github.io/async-book/