

RustoDo – Gestionnaire de tâches CLI en Rust

Découverte des bases de rust

https://docs.google.com/presentation/d/1f7TnAkq1s8R2WV5BpT-PXm3RFQBPE2wDWYxM5fUw/edit?usp=drive_link

Fonctionnalités principales

- add 'Faire les courses' → ajoute une tâche
- list → affiche toutes les tâches
- done 3 → marque une tâche comme terminée
- remove 3 → supprime une tâche
- Sauvegarde dans un fichier local (JSON, CSV, ou texte simple)



A screenshot of a terminal window titled "tasks.json [+]". The window displays a todo list organized by due date:

- Today**:
 - ✓ 2 create spotify playlists with truly the best 80's songs-
 - ✓ 3 create new screenshot for github-
- Tomorrow**:
 - 5 conjure the holy snek
- In 2 days**:
 - 4 remember the milk
- In 10 days**:
 - 1 finish firmware for custom keyboard

The terminal also shows a status bar at the bottom with commands like :h Help, :o Open, :d Done, :p Purge, and :r Remove, and a message "#5 ↵ [✓] File loaded".

Phase 1 : Base du CLI

- Parsing manuel des arguments avec std::env::args
- Gestion en mémoire via un Vec<String>
- Squelette d'application CLI minimal

Phase 2 : Structuration & persistance

- Struct Task { id, title, done }
- Sauvegarde dans un fichier JSON via serde_json
- Transition vers une gestion persistante

```
MyType { a: 1, b: 2, c: 3 }

MyType { a: 1, b: 2,
          ..Default::default() }

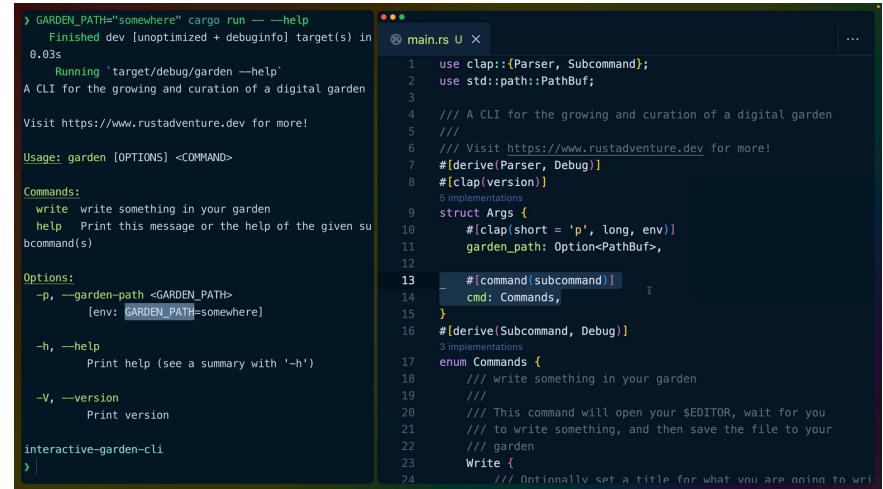
mytype!(a: 1, b: 2)

MyTypeBuilder::default()
    .a(1).b(2).build()

new_mytype(1, 2)
```

Phase 3 : Ergonomie & robustesse

- Utilisation de clap pour une interface ergonomique
- Gestion d'erreurs avec Result, thiserror
- Tests unitaires avec cargo test



```
> GARDEN_PATH="somewhere" cargo run -- --help
   Finished dev [unoptimized + debuginfo] target(s) in
0.03s
      Running `target/debug/garden --help`
A CLI for the growing and curation of a digital garden

Visit https://www.rustadventure.dev for more!

Usage: garden [OPTIONS] <COMMAND>

Commands:
  write  write something in your garden
  help   Print this message or the help of the given subcommand(s)

Options:
  -p, --garden-path <GARDEN_PATH>
                    [env: GARDEN_PATH=somewhere]

  -h, --help
                    Print help (see a summary with '-h')

  -V, --version
                    Print version

  interactive-garden-cli
|
```

```
1 use clap::{Parser, Subcommand};
2 use std::path::PathBuf;
3
4 /// A CLI for the growing and curation of a digital garden
5 /**
6  * Visit https://www.rustadventure.dev for more!
7 #[derive(Parser, Debug)]
8 #[clap(version)]
9 struct Args {
10     #[clap(short = 'p', long, env)]
11     garden_path: Option<PathBuf>; ...
12
13 #[command(subcommand)]
14 fn cmd: Commands,
15 }
16 #[derive(Subcommand, Debug)]
17 enum Commands {
18     Write {
19         /// write something in your garden
20         ...
21         /// This command will open your $EDITOR, wait for you
22         /// to write something, and then save the file to your
23         /// garden
24         Write {
25             /// Optionally set a title for what you are doing to wr...
```

Phase 4 : Fonctions avancées

- Dates d'échéance avec chrono
- Sous-commandes : todo add, todo list...
- Colorisation de la sortie avec colored
- Export CSV / Markdown (bonus)

chronotope/chrono

Date and time library for Rust



233
Contributors

481k
Used by

4k
Stars

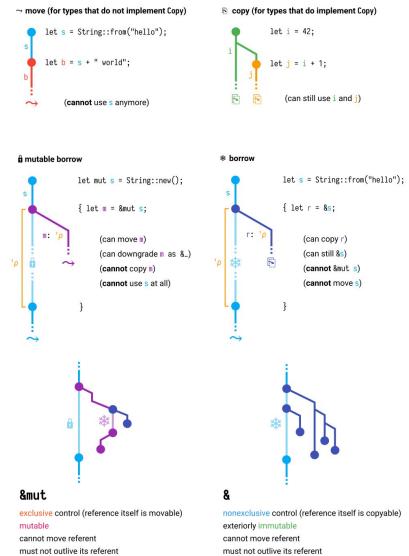
593
Forks



[Image Source](#)

Concepts Rust abordés

- Ownership & borrowing
- Structs et enums
- Gestion des erreurs (Result, Option)
- Modules, crates, I/O, fichiers
- Outils : cargo build/run/test



[Image Source](#)