



Elementary Systems Programming Project

01286120 Elementary Systems Programming

Software Engineering Program

Faculty of Engineering, KMITL

By

66010971 Chaiyanun Sakulsaowapakkul

RUST PROJECT: WINGMAN

The introduction

I would like to introduce my project from the third category, C3 Personal Data Management. I considered the name "Wingman" to mean someone who provides support and assistance to a friend in various situations. In the context of a to-do list, this name could suggest that this application is there to support and assist users in managing their tasks, just like a helpful friend or "wingman."

Overview

The Rust Task Manager is a command-line application that allows users to manage their to-do list by performing various tasks, including adding, viewing, editing, deleting, marking as complete, and exporting tasks to different formats.

Feature

The Rust Task Manager provides the following features:

- **Add Task:** Users can add a new task by specifying a title and a description for the task.
- **View Tasks:** Users can view the list of tasks along with their titles, descriptions, and completion status.
- **Edit Task:** Users can edit an existing task by providing the task number, a new title, and a new description.
- **Delete Task:** Users can delete a task by specifying its task number.
- **Mark as Complete:** Users can mark a task as complete by specifying its task number.
- **Export Tasks:** Users can export the list of tasks to a file in either JSON or YAML format. Users can specify the name of the output file and the format.

Command

- **'add'** subcommand

cargo run -- add --title "Task Title" --description "Task Description"

- **'view'** subcommand

cargo run -- view

- **'edit'** subcommand

cargo run -- edit --task_number 1 --title "Updated Title" --description "Updated Description"

- **'delete'** subcommand

cargo run -- delete --task_number 1

- **'mark-complete'** subcommand

cargo run -- mark-complete --task_number 1

- **'export'** subcommand

(export to **JSON**)

cargo run -- export --file_name "tasks" --format json

(export to **YAML**)

cargo run -- export --file_name "tasks" --format yaml

Example Usage

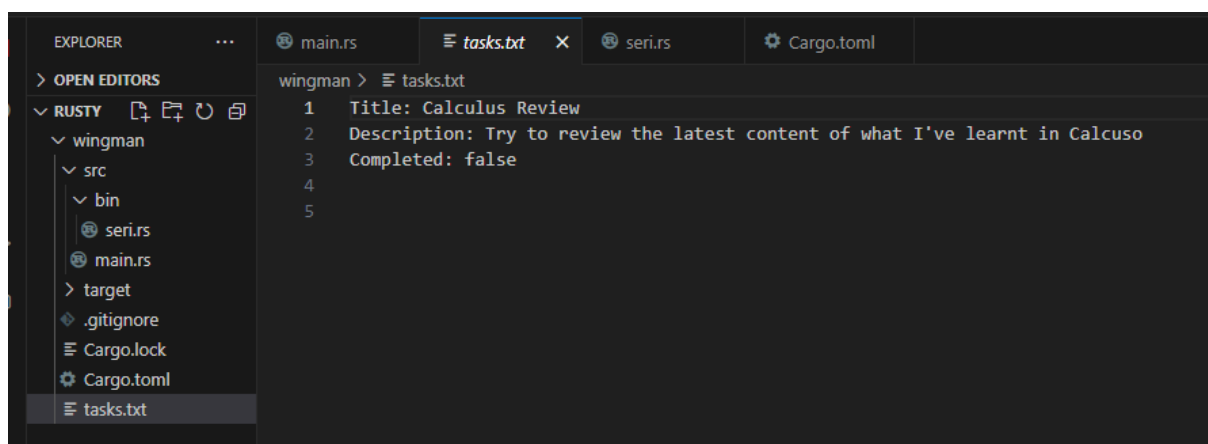
let's try adding a task! I will prompt

cargo run --bin wingman add "Calculus Review" "Try to review the latest content of what I've learnt in Calculus!"

```
PS D:\Programming_Files\rusty\wingman> cargo run --bin wingman add "Calculus Review" "Try to review the latest content of what I've learnt in Calculus"
Finished dev [unoptimized + debuginfo] target(s) in 0.04s
Running `target\debug\wingman.exe add "Calculus Review" "Try to review the latest content of what I've learnt in Calculus"`
Task added successfully.
PS D:\Programming_Files\rusty\wingman> |
```

Task added successfully.

Now the program will generate a "tasks.txt" file to store the information



Let me try using the option view

cargo run --bin wingman view

```
PS D:\Programming_Files\rusty\wingman> cargo run --bin wingman view
Finished dev [unoptimized + debuginfo] target(s) in 0.03s
Running `target\debug\wingman.exe view`
Viewing tasks:
Task 1: Title: Calculus Review, Description: Try to review the latest content of what I've learnt in Calculso, Completed: false
PS D:\Programming_Files\rusty\wingman> |
@ Connected to Discord
```

Viewing tasks:

Task 1: Title: Calculus Review, Description: Try to review the latest content of what I've learnt in Calculso, Completed: false

Oh no! looks like I've made a mistake, a typo in the <description>. I will edit it then.

cargo run --bin wingman edit 1 "Calculus Review" "Make sure that I ACTUALLY review it"

```
PS D:\Programming_Files\rusty\wingman> cargo run --bin wingman edit 1 "Calculus Review" "Make sure that I ACTUALLY review it"
Finished dev [unoptimized + debuginfo] target(s) in 0.03s
Running `target\debug\wingman.exe edit 1 "Calculus Review" "Make sure that I ACTUALLY review it"`
Task edited successfully.
PS D:\Programming_Files\rusty\wingman> 
```

Connected to Discord

Task edited successfully.

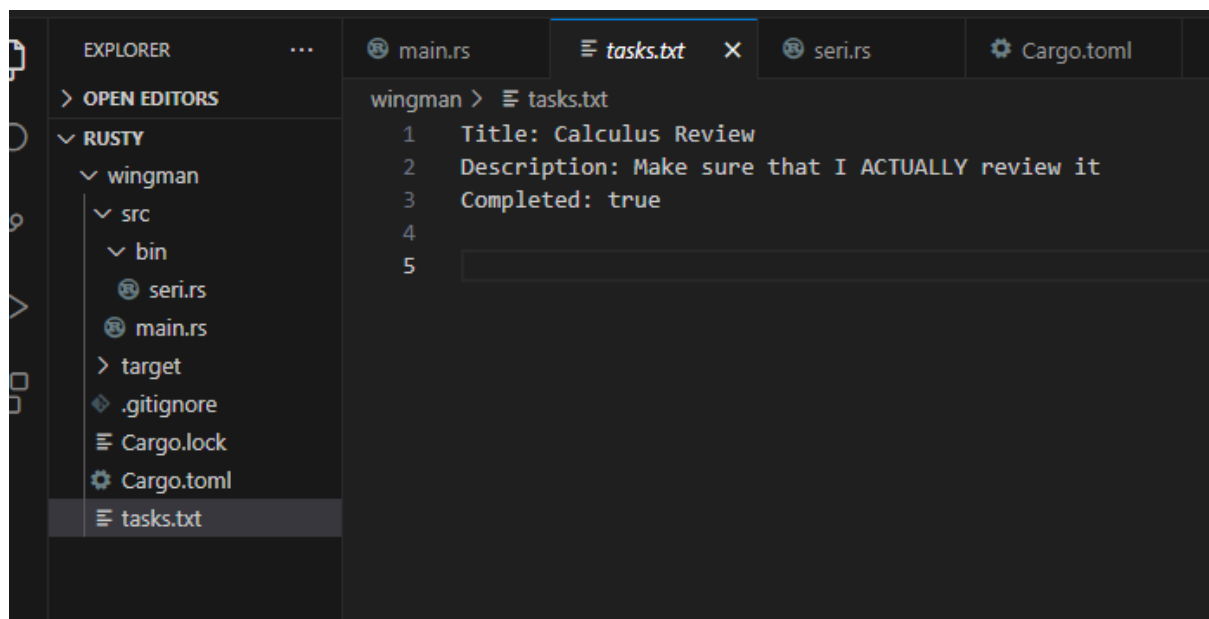
Alrighty! I just reviewed Calculus, now let's mark it as complete

cargo run --bin wingman mark-complete 1

```
PS D:\Programming_Files\rusty\wingman> cargo run --bin wingman mark-complete 1
Finished dev [unoptimized + debuginfo] target(s) in 0.03s
Running `target\debug\wingman.exe mark-complete 1`
Task marked as complete.
PS D:\Programming_Files\rusty\wingman> 
```

Connected to Discord

Task marked as complete.



Can add more tasks as you wish

```
1 Title: Calculus Review
2 Description: Make sure that I ACTUALLY review it
3 Completed: true
4
5 Title: Final Exam.....
6 Description: Prepare yourself for the final exam, first you need to understand that core
7 Completed: false
8
9
```

Kind of redundant, maybe I should delete the second task

cargo run --bin wingman delete 2

```
PS D:\Programming_Files\rusty\wingman> cargo run --bin wingman delete 2
Finished dev [unoptimized + debuginfo] target(s) in 0.03s
Running `target\debug\wingman.exe delete 2`
Task deleted successfully.
PS D:\Programming_Files\rusty\wingman>
```

Woopsy! Task 2 is gone

```
1 Title: Calculus Review
2 Description: Make sure that I ACTUALLY review it
3 Completed: true
4
5
```

Let's export it to JSON format

cargo run --bin wingman export thank_you json

```

PS D:\Programming Files\rusty\wingman> cargo run --bin wingman export thank_you json
    Finished dev [unoptimized + debuginfo] target(s) in 0.03s
    Running `target\debug\wingman.exe export thank_you json`
Tasks exported to JSON successfully.
PS D:\Programming Files\rusty\wingman>

```

Tasks exported to JSON successfully.

```

1  [
2
3      {
4          "title": "Calculus Review",
5          "description": "Make sure that I ACTUALLY review it",
6          "completed": true
7      },
8      {
9          "title": "Exercise regularly",
10         "description": "At least I think I want to exercise everyday for an hour.....",
11         "completed": false
12     }
13 ]

```

Let's do the same for YAML

cargo run --bin wingman export so_much yaml

```

PS D:\Programming Files\rusty\wingman> cargo run --bin wingman export so_much yaml
    Finished dev [unoptimized + debuginfo] target(s) in 0.03s
    Running `target\debug\wingman.exe export so_much yaml`
Tasks exported to YAML successfully.
PS D:\Programming Files\rusty\wingman>

```

Tasks exported to YAML successfully.

```

1  ---
2  - title: Calculus Review
3    description: Make sure that I ACTUALLY review it
4    completed: true
5  - title: Exercise regularly
6    description: At least I think I want to exercise everyday for an hour.....
7    completed: false
8

```

I also added a simple test case.

```
#[cfg(test)]

mod tests {

    use super::*;

    use std::fs;

    const TEST_FILE_NAME: &str = "test_tasks.txt";

    #[test]

    fn test_add_task() {

        let mut tasks = Vec::new();

        let title = "Test Task";

        let description = "This is a test task description.";

        add_task(&mut tasks, title, description);

        assert_eq!(tasks.len(), 1);

        assert_eq!(tasks[0].title, title);

        assert_eq!(tasks[0].description, description);

        assert!(tasks[0].completed);

        let _ = fs::remove_file(TEST_FILE_NAME);

    }

}
```



```
running 1 test
test tests::test_add_task ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

PS D:\Programming_Files\rusty\wingman> 
🌐 Connected to Discord
```

Dependencies

serde = { version = "1.0", features = ["derive"] }

serde_json = "1.0"

serde_yaml = "0.8"

clap = "2.33"

link to my github

<https://github.com/sharincy/Wingman>

I also have some prototype to Wingman, they are in

seri.rs and menu_base.rs

Both of them are the prototype of the current version, they are menu based (If interested, you can check them in the zip file I provided).