Week 03 Workshop

EXERCISE:

Workshop 3 - BuffeRS

PRE-WORKSHOP PREPARATION:

There are three things which may help prepare for this workshop:

- Watch the third week's lectures.
- Download and read through the starter code
- Complete the pre-workshop revision work.

For this week's pre-workshop work, you should:

- Fetch the workshop code, from this tar file
- Read each section of code. There are some comments, but you might like to ask questions on the Forum too.
- Edit the code so you can run the command: `open_file path/to/file.txt`, and read in the file at that path. When the buffer closes, save back to that file.
- Notice any missing features you want? Have a go at implementing them yourself.

This is not required, but will be very useful!

IN-WORKSHOP REVISION:

Your tutor will briefly revise the following:

- Mutable and Exclusive Borrows (see the `impl Buffer` comments to see an example of what will be talked about).
- The TermGame Library

If you have any questions, now is a great time to ask them!

In this workshop, we will be using Rust to build a basic text editor. This will also serve as practice for the assignment; in which we will use a small library called `termgame`; which is designed for creating games on the terminal.

The purpose of this workshop is to:

- Practice borrowing, mutability and lifetimes.
- Get a sneak peek at the library we will be using for assignment 1.
- Get experience designing a rust program.
- Have fun!

What we're doing

In small groups, you will be writing a text editing system. This system is slightly different to other systems you may be used to -- it revolves around a command line, which allows you to edit "buffers". In a buffer, you can see text and type it out. To exit a buffer, press "Esc".

After you have implemented some basic functionality, we will be implementing functions on buffers to do more advanced editing. Part of your goal should be to minimise your copying of strings -- pass references around where you can.

In the workshop

In the workshop, you will be asked to get into teams of 2 to 3 people. We recommend teams collaborate on one piece of code on one computer. If someone has already done the pre-workshop work, start with that code. If nobody has, get the starter code from this tar file. If more than two people have done the work, commence a duel to the death (or, discuss which code is better for which tasks, and use the most appropriate code/a combination of the two).

Teams can pick tasks to work on from the list below. Once a task is done, they should call over a tutor to have a chat about the task. Tutors will also be happy to provide help on rust issues (both related to getting code working, and design).

The Tasks

- The command `open buffer_name` should open a buffer, and call it `buffer_name`. Re-opening the same buffer should show the same text as the last time it was edited. Buffer names will never contain spaces.
- The command `search my interesting text` should go through each buffer, and print out every line with the matching text.

```
buffer_name:3 This is an example of my interesting text
buffer_name:7 my interesting text is awesome!
another_buffer_name:1 # my interesting text
```

- The command `copy_into buffer1 buffer2:3` should insert the text of `buffer1` after line 3 of `buffer2`
- The command `cut_into buffer1:3 buffer2:7` should remove line 3 of `buffer1`, and place it after line 7 of `buffer2`
- The command `buffer_from_command buffer_name [a command]` should fill the buffer called `buffer_name` with the output of the terminal command `[a command]`.

At the end, we'll come back together to review interesting bugs/errors, and interesting ways of completing the tasks.

COMP6991 24T1: Solving Modern Programming Problems with Rust is brought to you by
the School of Computer Science and Engineering
at the University of New South Wales, Sydney.
For all enquiries, please email the class account at cs6991@cse.unsw.edu.au
CRICOS Provider 00098G

Login as tutor