SYS2 Kernel - Tutorial

Copyright

Contents

Chapter 1

Environment

1.1 Operating system

Rust works on several operating systems, but the one we are using for the following lessons is "Linux x64", specifically on Ubuntu 22.04. You can work on other distros and architectures, it should work. But I am not going to help if you have a distro/architecture problem on another distro/architecture. Beside, the final project **must** work on the Ubuntu 22.04 x64 architecture.

1.1.1 The UNIX philosophy

- Make each program do one thing well;
- Make each program use text as data;
- Expect the output to each program to become the input to another.

How nice. We have an environment proposing smart and easy tools, all of them documented locally in the manpages.

1. Find a tutorial
2. Read it a bit
3. When you feel confident, do some exercises from the Piscine. Adjust them with how the programming language work.
4. If you want to continue to do some EPITA exercises, go back to instruction n°2, otherwise, continue
5. Recode some C functions not in the EPITA exercises list. Just *man* it if you need the specs
6. Not tired of recoding C functions? Go back to instructions n°5, otherwise, continue, but do not forget instruction n°2
7. Recode some libraries

   A string library (libstring)

   An IO library (libstream)

   A mathematical library

   Sets of data structures (FIFOs, LIFOs, ...)

   Sorters to those data structures

   Lexers

   Parsers

   ASTs

   ... and so on, even if these features already exist with the programming language
8. Not tired of recoding libraries? Go back to instruction n°7, otherwise, continue, but do not forget instruction n°2

9. You can now go for EPITA projects
    fnmatch
    evalexpr
    tinyprintf (if the language has variadic functions)
    minishell
    myreadiso
    minimake
    myfind
    malloc (if the language has C wrapping)
    myhttpd / spider
    opichat
    mybittorrent
    bistromathique (a.k.a libbistro)
    42sh
    K (just kidding)
10. Not tired of recoding EPITA projects? Go back to instruction n°9, otherwise, continue, but do not forget instruction n°2
11. Go for the shell builtins (although you have already made them with 42sh), just read the builtins(1) manpage
12. Go for some UNIX binaries
    ls
    sort
    uniq
    wc
    printf
    cat
    echo
    cut
    find
    rm
    rmdir
    mv
    cp
    dd
    kill
    ... and so on
13. And don't forget to train more and more:
    a shell interpreter (42sh but freer)
    a small compiler
    a pseudo netcat client and server to work with sockets
    requesting with a libcurl associated to your programming language
    forking
    threading
    daemoning
14. A personal project, but do not forget instruction n°2

```
[dependencies]
package = "*"
```

Chapter 2

LaTeX

Don't mind this chapter.Really, don't[1]
I'm just making notes for myself to write this document.

2.1 LaTeXsection

2.1.1 LaTeXsubsection

LaTeXsubsubsection

That's the lowest

Display a link with custom text

- item
    - subitem
        - subsubitem

i = 42 // Erreur

1. enum
    1. subenum
        1. subsubenum

```
#!/bin/sh

echo "Coucou"
```

rustc	The compiler
cargo	The environment tool

[1] It's not important