Emacs: Editor as a Platform

Settling the Editor War, once and for all.

$ whoami

- Juhun “RangHo” Lee

- Class of **2019**
- Dept. of **Computer Science and Engineering**
- Dept. of **English Language and Literature**

- Makes useless programs for living
- Massive **Linux** nerd

Let’s talk about **Emacs** today.

GitHub:	**@RangHo**
Twitter:	**@RangHo_777**
Instagram:	**@rangho_lee**

Editor Survey?

Programmer’s best friend

Integrated **D**evelopment **E**nvironment

Visual Studio **Xcode** **IntelliJ IDEA** **Android Studio**

Programmer's best friend...?

Integrated Development Environment

Which language do you use...

...for what purpose?

Game?

Back-end?

Visual Studio

Machine Learning?

Xcode

Low-level / IoT?

IntelliJ IDEA

Android Studio

Front-end?

Command line utility?

~~Programmer's best friend~~

Integrated **D**evelopment **E**nvironment

OBSOLETE!

		ntelliJ IDEA	**Android Studio**
	ve-C / Swift	**Java / Kotlin / C#**	**Java / Kotlin / C++**
ps	for **Apple devices**	for **JVM ecosystem**	for **Android devices**

What are the other options?

Sublime Text

Inventor of multi-select.
"Try it, for free, forever."

Fleet

New text editor from the folks at JetBrains.

VS Code

Text editor powered by trusty IntelliSense.

Notepad++

Superfast notepad replacement.

What are the other options?

Sublime

Inventor of multi

“Try it, f

f

VS Code

itor powered by

ntelliSense.

F

New text editor from
the folks at JetBrains.

pad++

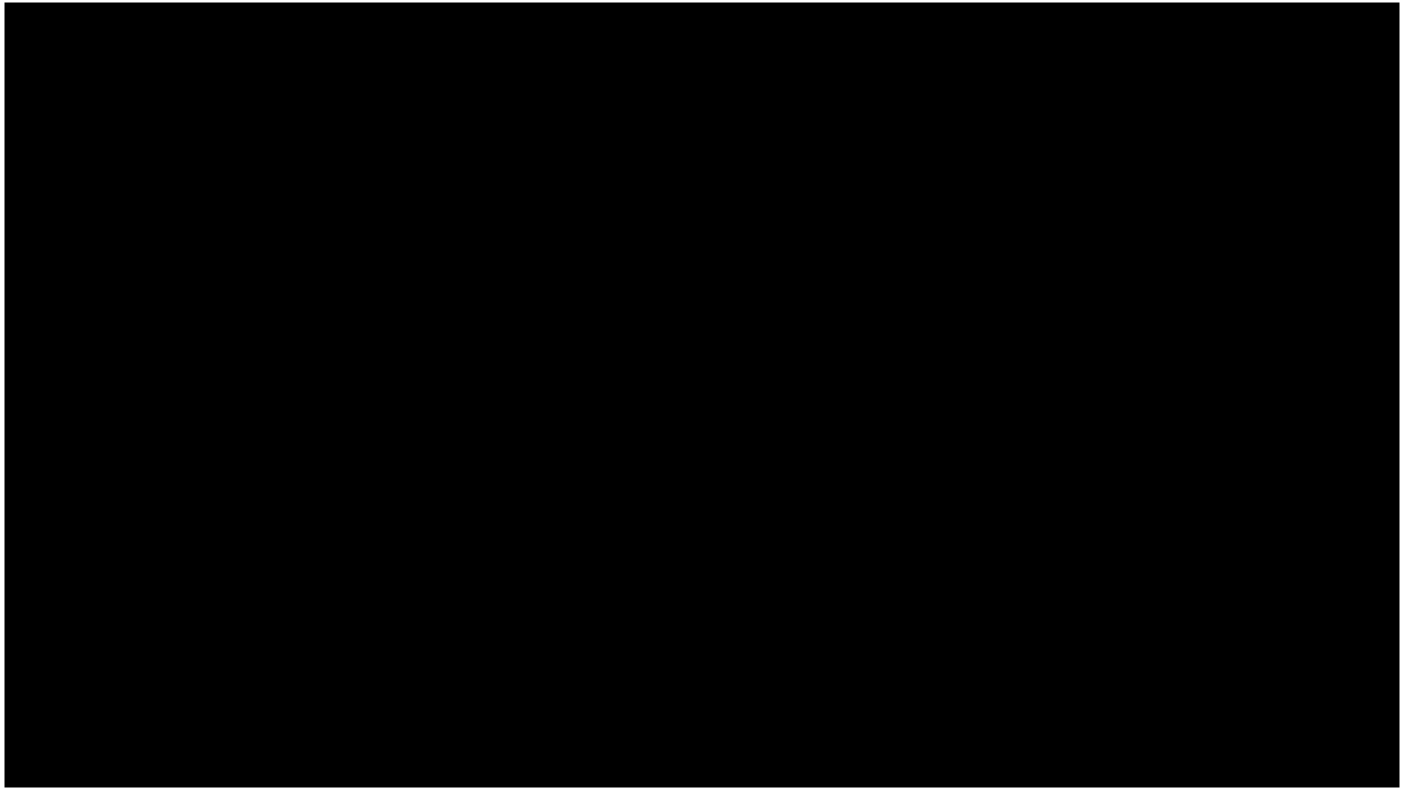
ast notepad

ment.

```
> ssh cspro.sogang.ac.kr

Welcome to Linux!

Now try to use your editor lmao
rangho@cspro $
```

“The Editor War”

“The Editor War”



"Emacs is **objectively** better than vim."

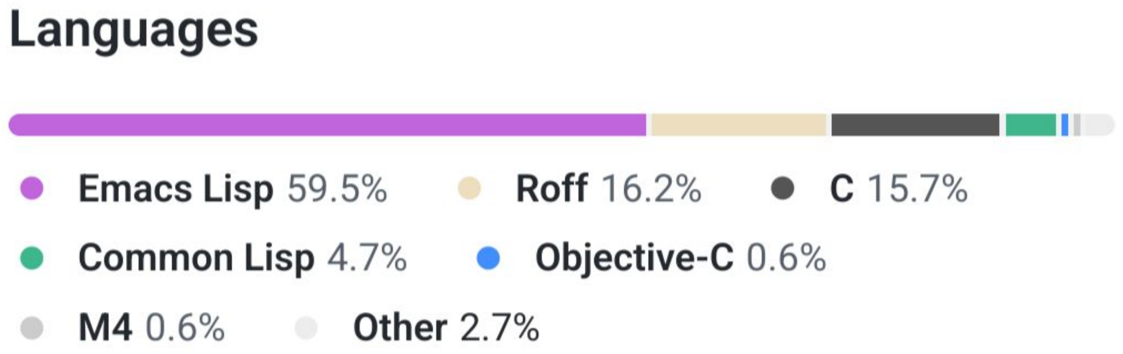
– this guy, 2022

What is Emacs?

- Part of **GNU** Project
- Both **GUI** and **CLI** available
- Built with **C** and **Emacs Lisp**
    - Lisp interpreter and display is in C
    - Everything else → **Emacs Lisp**

- <u>**Everything can be customized!**</u>

Customize, customize, customize!

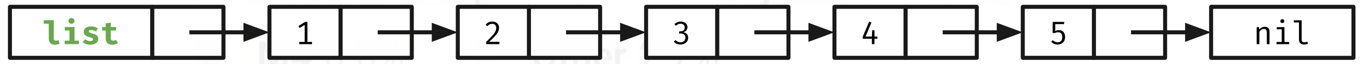


- Emacs is ***not*** a text editor
- Emacs is a **Lisp Interpreter** with **Lisp code**

Customize, customize, customize!

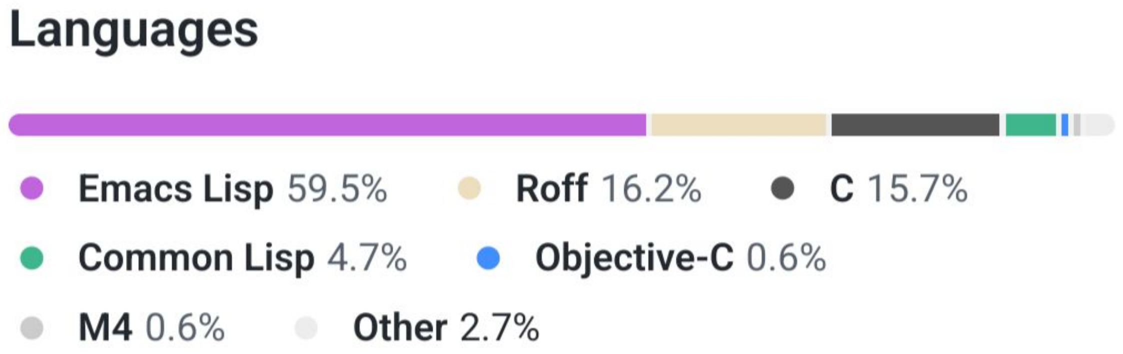
Lisp or **LISP** (**LIS**t **P**rocessor)

- Everything in the language is made of *lists*
  - `(list 1 2 3 4 5)` → `'(1 2 3 4 5)`



- Program can **modify itself** with ease
- Beginning of **REPL** (**R**ead → **E**valuate → **P**rint → **L**oop)

Customize, customize, customize!



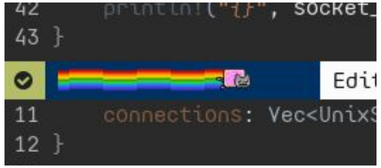
- Emacs is ***not*** a text editor
- Emacs is a **Lisp Interpreter** with **Lisp code**
- Lisp code ***evaluates to*** a text editor
  - ➜ Any aspect of the editor can be tweaked!

5,700+

Number of Emacs packages available for installation.

Lots and lots of packages

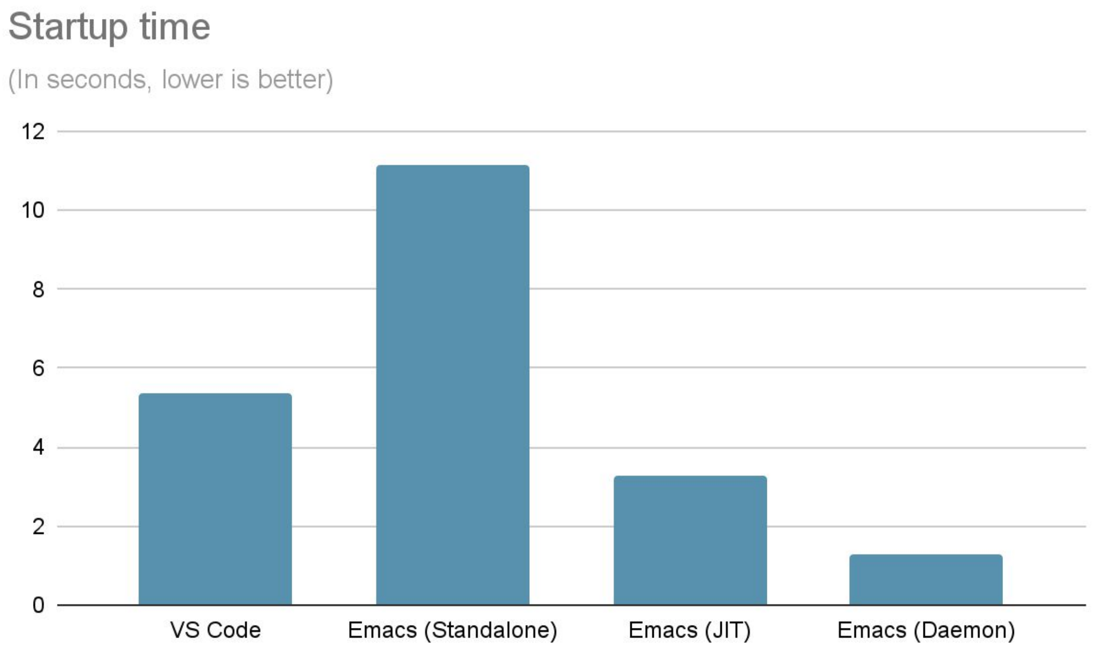
- **Major mode** vs. **Minor mode**
  - Major mode → How to interact with a text file?
  - Minor mode → What features to enable when editing?
- Major mode can be inherited for similar languages
  - Major mode for C#, Java, JavaScript inherit from C/C++ mode
- Minor modes enable many features
  - `evil-mode` brings Vi keybindings to Emacs
  - `nyan-mode` displays Nyan Cat, indicating the cursor position



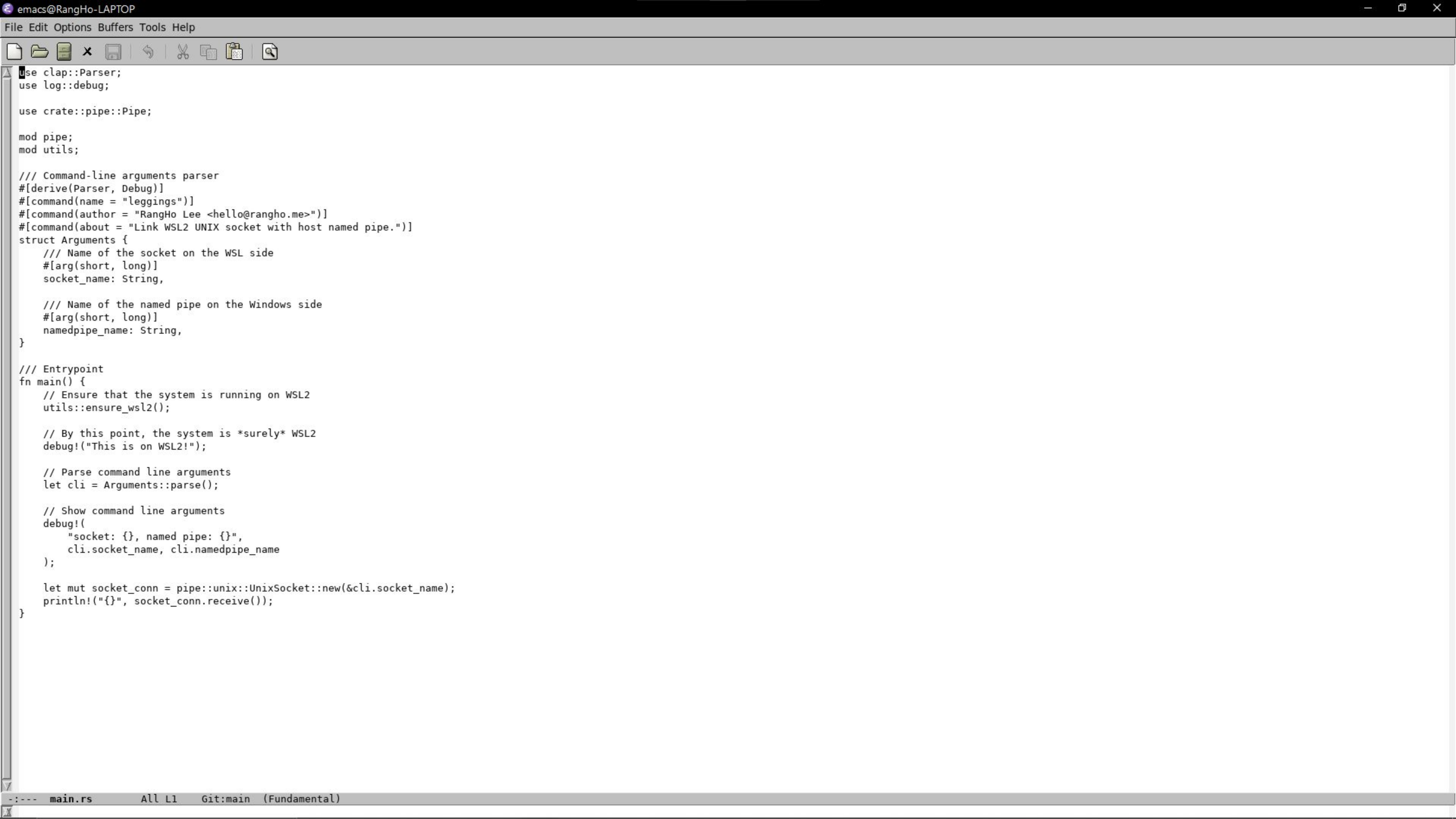
```
    println!("{}", socket_
}
```

```
    connections: Vec<UnixS
}
```

FAST!

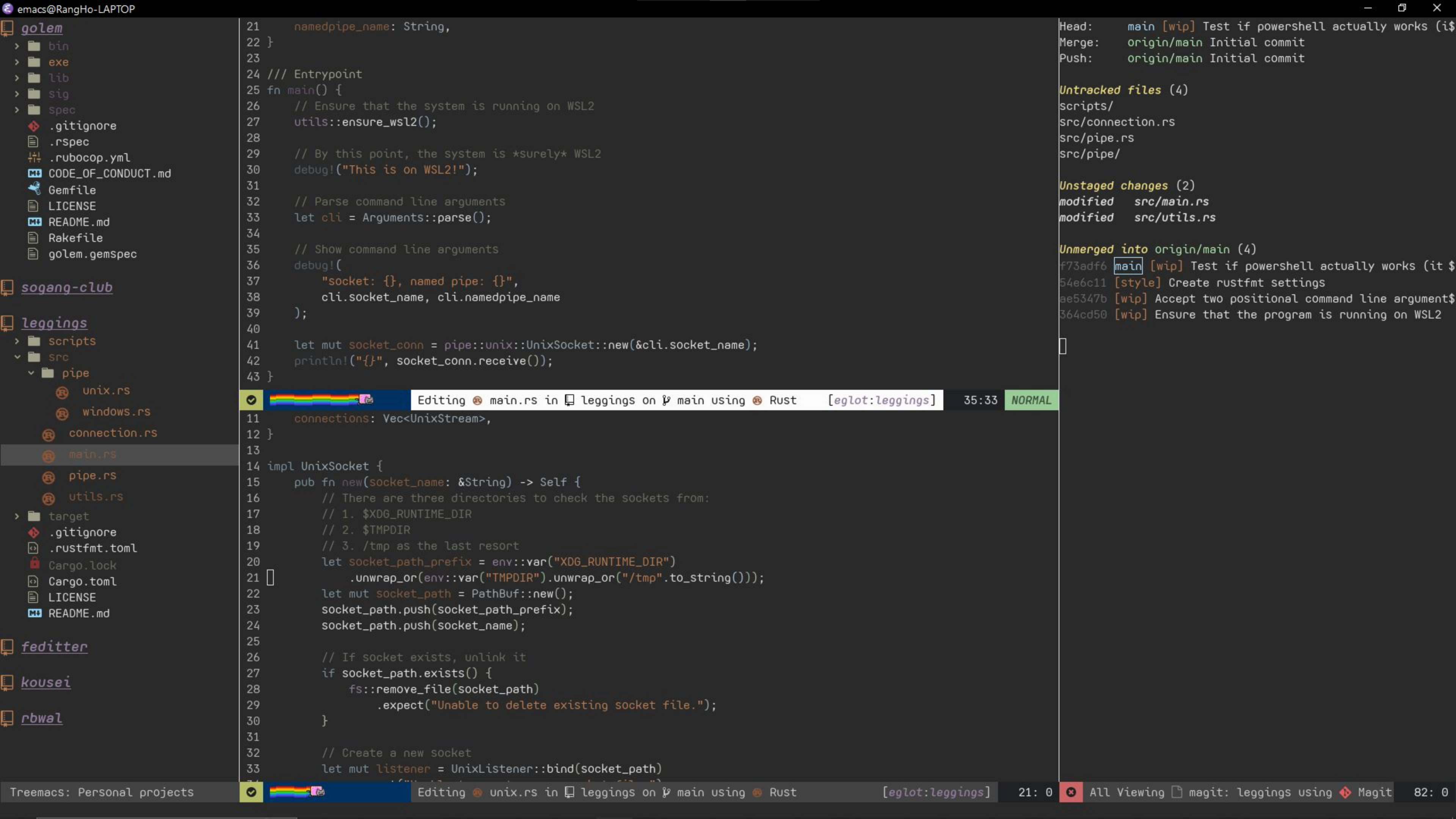


Startup time
(In seconds, lower is better)
12
10
8
6
4
2
0
VS Code
Emacs (Standalone)
Emacs (JIT)
Emacs (Daemon)



```rust
use clap::Parser;
use log::debug;

use crate::pipe::Pipe;

mod pipe;
mod utils;

/// Command-line arguments parser
#[derive(Parser, Debug)]
#[command(name = "leggings")]
#[command(author = "RangHo Lee <hello@rangho.me>")]
#[command(about = "Link WSL2 UNIX socket with host named pipe.")]
struct Arguments {
    /// Name of the socket on the WSL side
    #[arg(short, long)]
    socket_name: String,

    /// Name of the named pipe on the Windows side
    #[arg(short, long)]
    namedpipe_name: String,
}

/// Entrypoint
fn main() {
    // Ensure that the system is running on WSL2
    utils::ensure_wsl2();

    // By this point, the system is *surely* WSL2
    debug!("This is on WSL2!");

    // Parse command line arguments
    let cli = Arguments::parse();

    // Show command line arguments
    debug!(
        "socket: {}, named pipe: {}",
        cli.socket_name, cli.namedpipe_name
    );

    let mut socket_conn = pipe::unix::UnixSocket::new(&cli.socket_name);
    println!("{}", socket_conn.receive());
}
```

```
-:---  main.rs        All L1    Git:main  (Fundamental)
```



```
emacs@RangHo-LAPTOP
```

```
golem
 > bin
 > exe
 > lib
 > sig
 > spec
   .gitignore
   .rspec
   .rubocop.yml
   CODE_OF_CONDUCT.md
   Gemfile
   LICENSE
   README.md
   Rakefile
   golem.gemspec

sogang-club

leggings
 > scripts
 v src
   v pipe
       unix.rs
       windows.rs
     connection.rs
     main.rs
     pipe.rs
     utils.rs
 > target
   .gitignore
   .rustfmt.toml
   Cargo.lock
   Cargo.toml
   LICENSE
   README.md

feditter

kousei

rbwal
```

```rust
    namedpipe_name: String,
}

/// Entrypoint
fn main() {
    // Ensure that the system is running on WSL2
    utils::ensure_wsl2();

    // By this point, the system is *surely* WSL2
    debug!("This is on WSL2!");

    // Parse command line arguments
    let cli = Arguments::parse();

    // Show command line arguments
    debug!(
        "socket: {}, named pipe: {}",
        cli.socket_name, cli.namedpipe_name
    );

    let mut socket_conn = pipe::unix::UnixSocket::new(&cli.socket_name);
    println!("{}", socket_conn.receive());
}
```

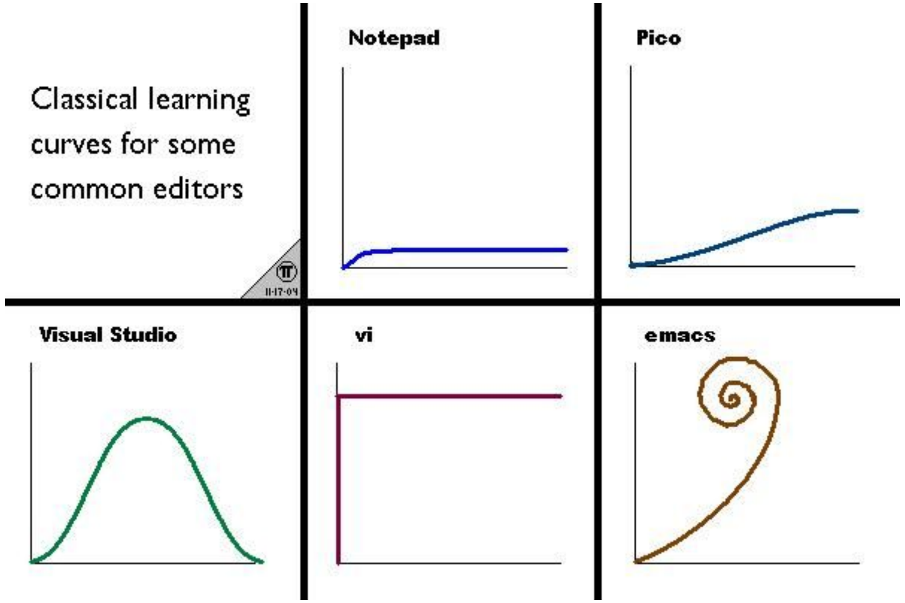
```
Editing main.rs in leggings on main using Rust    [eglot:leggings]    35:33  NORMAL
```

```rust
    connections: Vec<UnixStream>,
}

impl UnixSocket {
    pub fn new(socket_name: &String) -> Self {
        // There are three directories to check the sockets from:
        // 1. $XDG_RUNTIME_DIR
        // 2. $TMPDIR
        // 3. /tmp as the last resort
        let socket_path_prefix = env::var("XDG_RUNTIME_DIR")
            .unwrap_or(env::var("TMPDIR").unwrap_or("/tmp".to_string()));
        let mut socket_path = PathBuf::new();
        socket_path.push(socket_path_prefix);
        socket_path.push(socket_name);

        // If socket exists, unlink it
        if socket_path.exists() {
            fs::remove_file(socket_path)
                .expect("Unable to delete existing socket file.");
        }

        // Create a new socket
        let mut listener = UnixListener::bind(socket_path)
```

```
Head:     main [wip] Test if powershell actually works (i$
Merge:    origin/main Initial commit
Push:     origin/main Initial commit

Untracked files (4)
scripts/
src/connection.rs
src/pipe.rs
src/pipe/

Unstaged changes (2)
modified   src/main.rs
modified   src/utils.rs

Unmerged into origin/main (4)
f73adf6 main [wip] Test if powershell actually works (it $
54e6c11 [style] Create rustfmt settings
ae5347b [wip] Accept two positional command line argument$
364cd50 [wip] Ensure that the program is running on WSL2
```

```
Treemacs: Personal projects | Editing unix.rs in leggings on main using Rust    [eglot:leggings]    21: 0 | All Viewing magit: leggings using Magit    82: 0
```

Yep, cool. Now what?



Now this.

Emacs: the *good* part

- Make the editor for you, and you only
- Learn more about Lisp
- Think "functionally"
- Never touch the mouse again

Emacs: the *bad* part

- Customization is hard, like, ***hard***
- Default editor looks bad
- Overwhelmingly many features

Emacs is a shot worth giving.