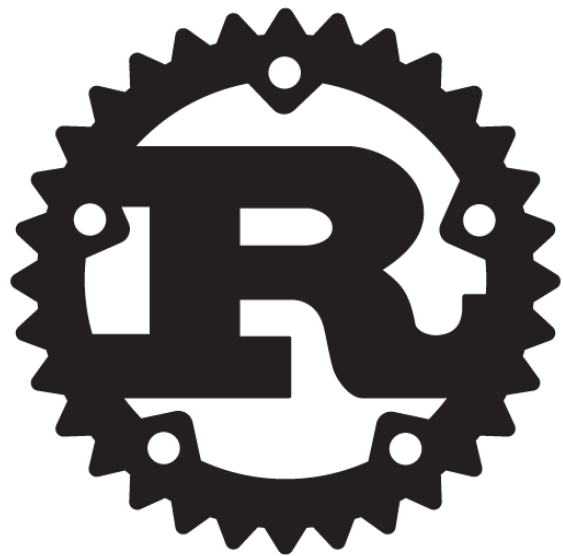


Why you should take a look at

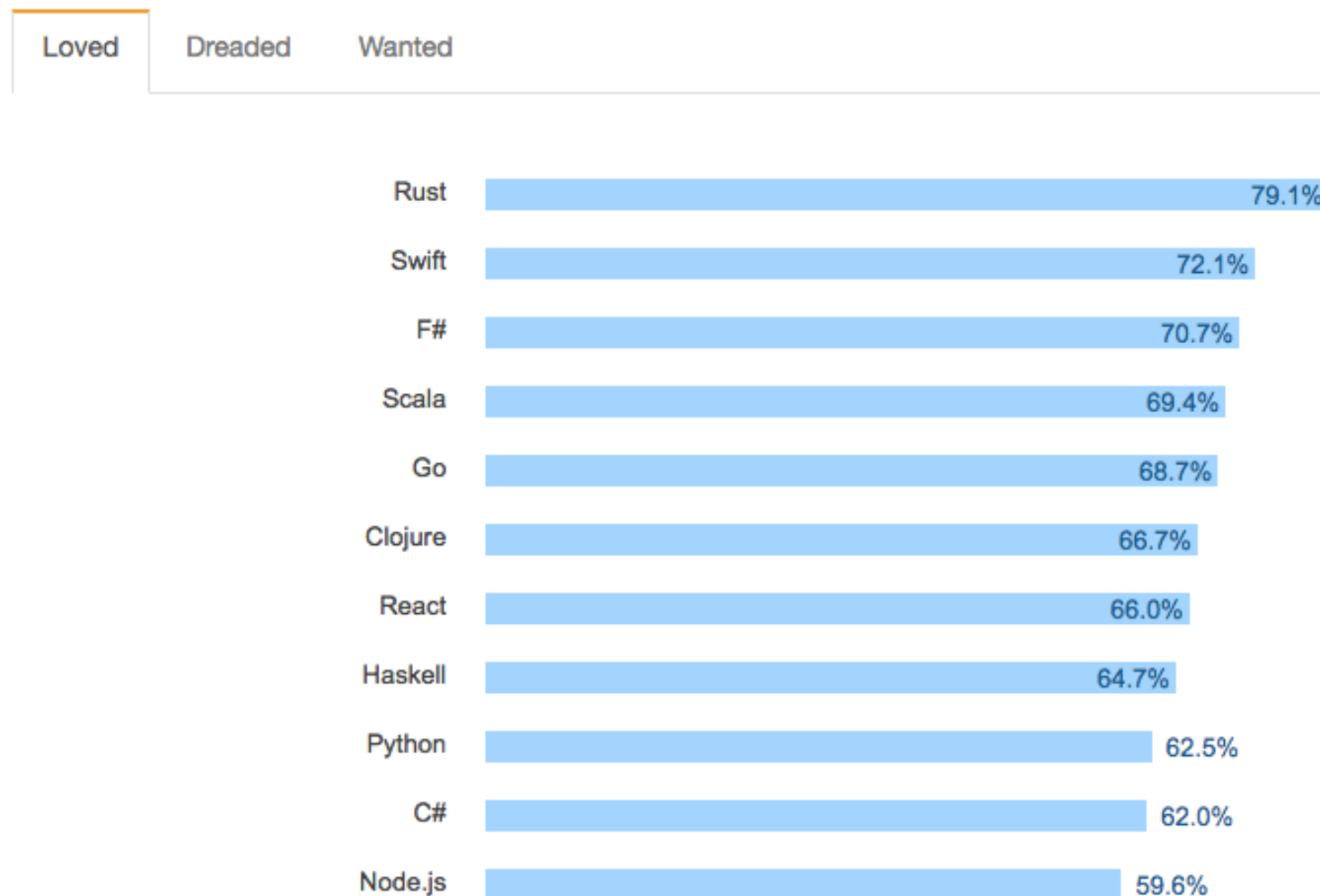




= ***Rust***, a modern, safe, fast, and
multi-core processor aware
programming language

Developers <3 Rust

II. Most Loved, Dreaded, and Wanted



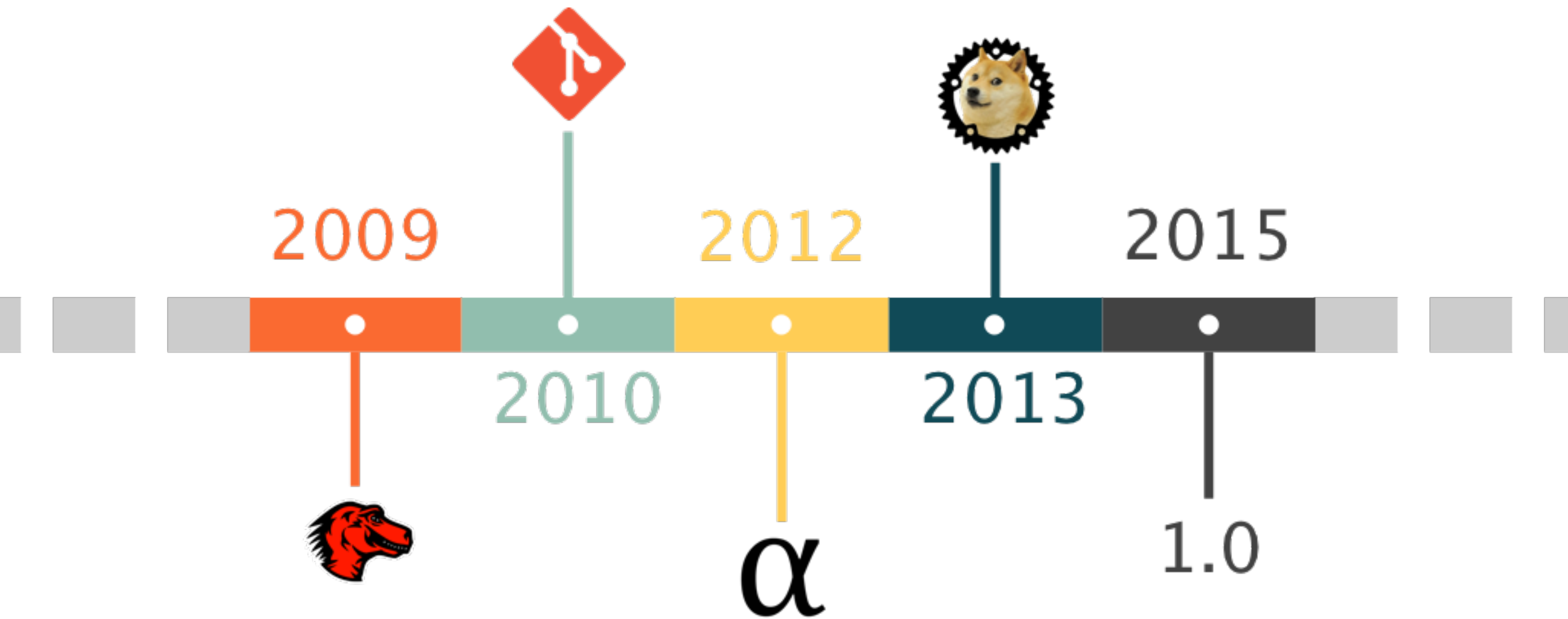
% of developers who are developing with the language or tech and have expressed interest in continuing to develop with it

<http://stackoverflow.com/research/developer-survey-2016#community>

I had your curiosity, but now...



Timeline



Promesses

Modern

- General-purpose
- Multi-paradigm
- Built-in packet manager
- ...

Safe

- Statically typed
- Type safe
- Memory safe
- Data safe
- Data races safe
- ...

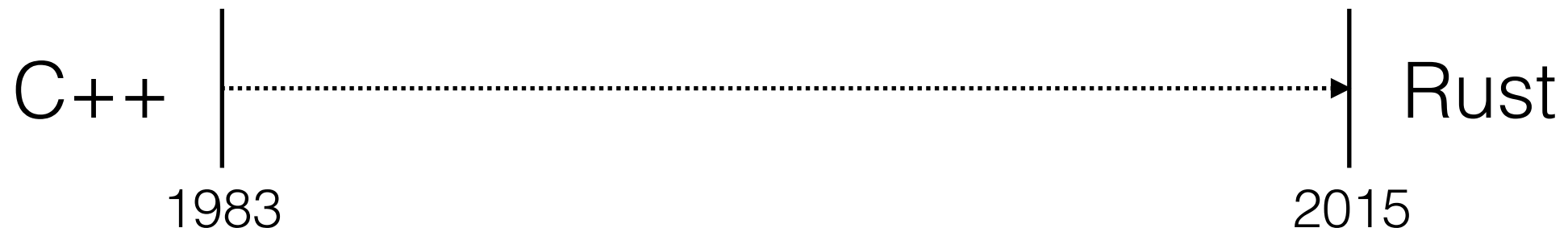
Fast

- LLVM back end
- No GC
- As fast as C++

Data race

- Concurrent
- Parallelism

Ambitious

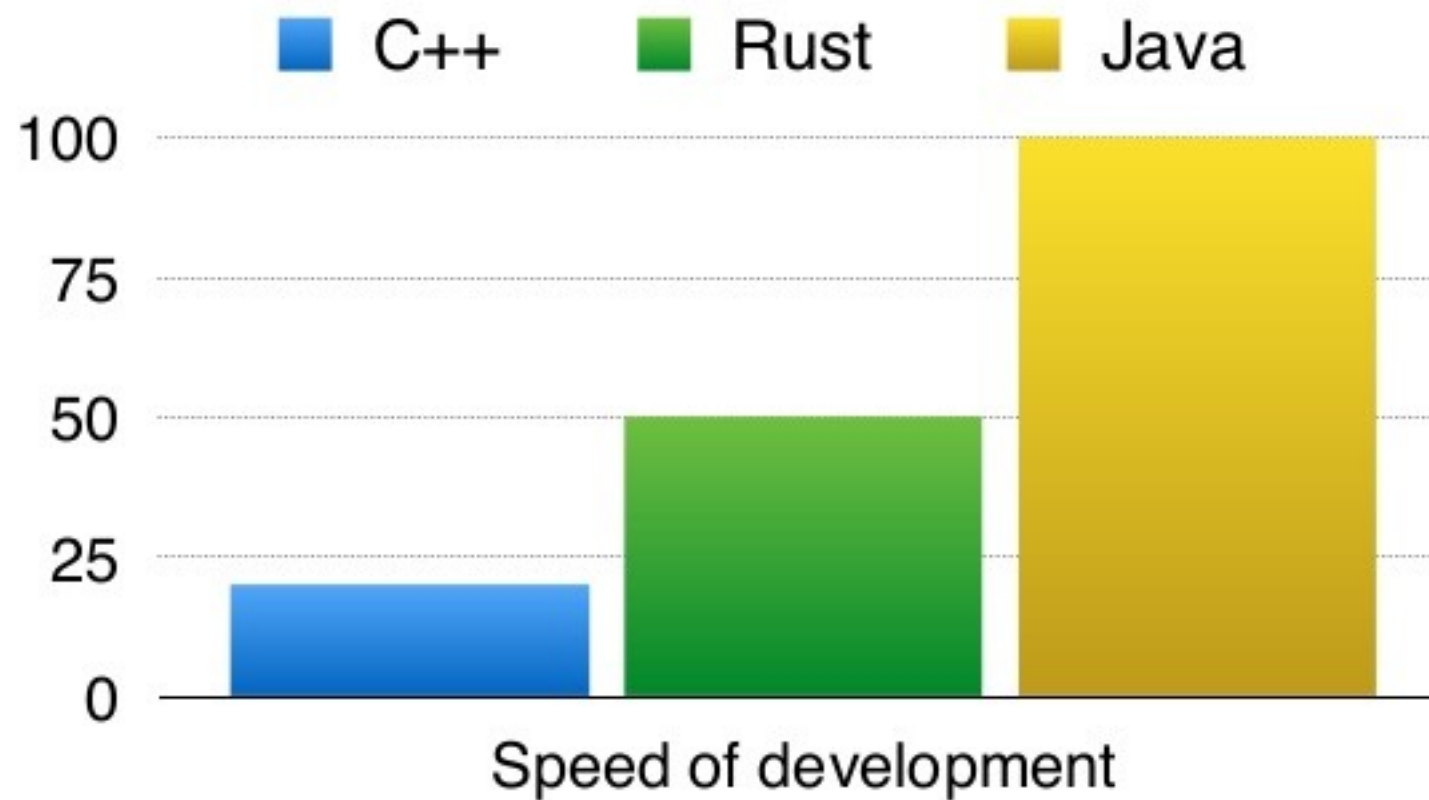


Inspired by popular technologies



Good development speed

Development speed



Cargo, the perfect companion

```
[antonin@MacBook-Air-de-antonin] - [~] - [Dim déc 11, 01:23]
[~] $ cargo
Rust's package manager

Usage:
  cargo <command> [<args>...]
  cargo [options]

Options:
  -h, --help            Display this message
  -V, --version          Print version info and exit
  --list                List installed commands
  --explain CODE         Run `rustc --explain CODE`
  -v, --verbose ...     Use verbose output
  -q, --quiet           No output printed to stdout
  --color WHEN          Coloring: auto, always, never
  --frozen              Require Cargo.lock and cache are up to date
  --locked              Require Cargo.lock is up to date

Some common cargo commands are (see all commands with --list):
  build      Compile the current project
  clean      Remove the target directory
  doc        Build this project's and its dependencies' documentation
  new        Create a new cargo project
  init       Create a new cargo project in an existing directory
  run        Build and execute src/main.rs
  test       Run the tests
  bench      Run the benchmarks
  update     Update dependencies listed in Cargo.lock
  search     Search registry for crates
  publish    Package and upload this project to the registry
  install    Install a Rust binary

See 'cargo help <command>' for more information on a specific command.
```

Sun 12/11: **7,098** crates in stock / **92,105,502** downloads

Cargo, the perfect companion

```
# The release profile, used for `cargo build --release`.
```

```
[profile.release]
```

```
opt-level = 3
```

```
debug = false
```

```
rpath = false
```

```
lto = false
```

```
debug-assertions = false
```

```
codegen-units = 1
```

```
panic = 'unwind'
```

```
# The testing profile, used for `cargo test`.
```

```
[profile.test]
```

```
opt-level = 0
```

```
debug = true
```

```
rpath = false
```

```
lto = false
```

```
debug-assertions = true
```

```
codegen-units = 1
```

```
panic = 'unwind'
```

```
# The benchmarking profile, used for `cargo bench`.
```

```
[profile.bench]
```

```
opt-level = 3
```

```
debug = false
```

```
rpath = false
```

```
lto = false
```

```
debug-assertions = false
```

```
codegen-units = 1
```

```
panic = 'unwind'
```

Rust, for web developers

Databases

Templating

APIs

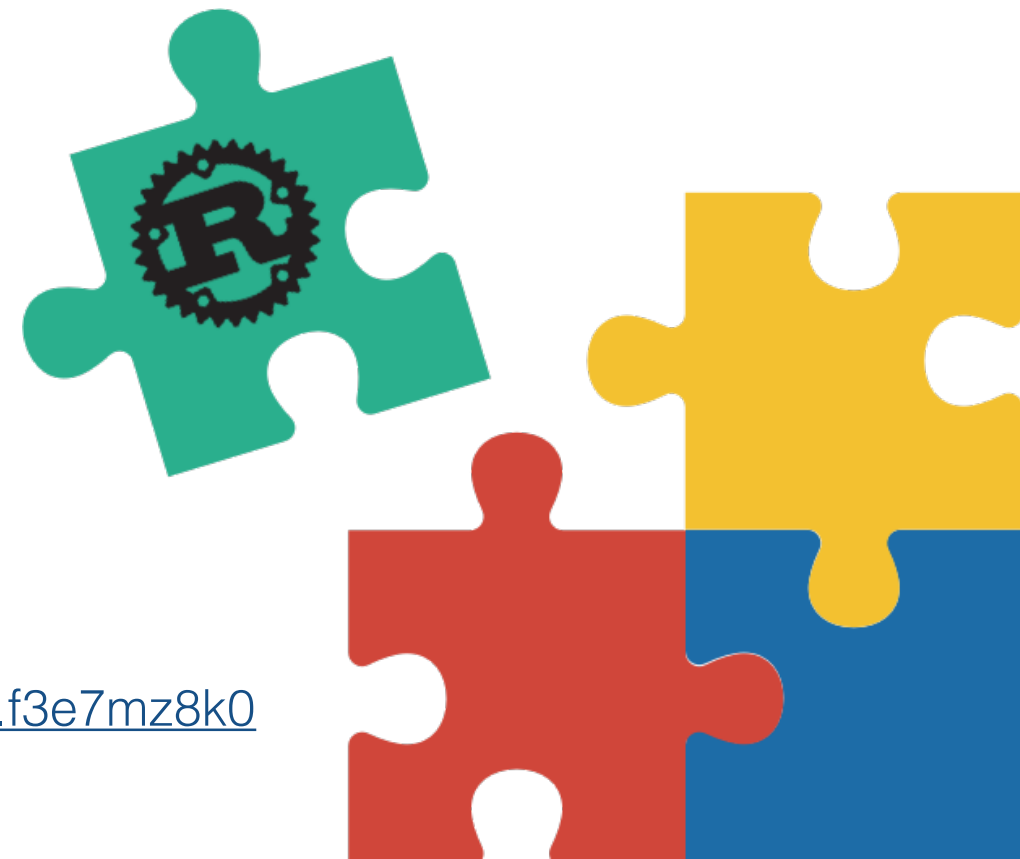
Client frameworks

Websocket

Server frameworks

and more...

Rust to Emscripten



<https://github.com/flosse/rust-web-framework-comparison>

<https://medium.com/mozilla-tech/rust-for-web-developers-1b0f4326e8b8#.f3e7mz8k0>

Rust, for web developers



Hans Koch

4 days ago

I really like Rust, and i use it for some of my web project whether via iron webframework or via neon-binding as nodejs module. The speed up at some sections of my code is enormous, and mutability helps fighting some problems with asynchronity.

Big thanks to the folks of Mozilla Research.



Code examples

```
fn main() {  
    println!("Hello World!");  
}
```

```
fn factorial_recursive (n: u64) -> u64 {  
    match n {  
        0 => 1,  
        _ => n * factorial_recursive(n-1)  
    }  
}  
  
fn factorial_iterative(n: u64) -> u64 {  
    (1..n+1).fold(1, |p, n| p*n)  
}  
  
fn main () {  
    for i in 1..10 {  
        println!("{}", factorial_recursive(i))  
    }  
    for i in 1..10 {  
        println!("{}", factorial_iterative(i))  
    }  
}
```


Code examples

```
use std::net::{TcpListener, TcpStream};
use std::io::{BufReader, BufRead, Write};
use std::thread;

fn main() {
    let listener = TcpListener::bind("127.0.0.1:12321").unwrap();
    println!("server is running on 127.0.0.1:12321 ...");

    for stream in listener.incoming() {
        let stream = stream.unwrap();
        thread::spawn(move || handle_client(stream));
    }
}

fn handle_client(stream: TcpStream) {
    let mut stream = BufReader::new(stream);
    loop {
        let mut buf = String::new();
        if stream.read_line(&mut buf).is_err() {
            break;
        }
        stream
            .get_ref()
            .write(buf.as_bytes())
            .unwrap();
    }
}
```

Code examples

```
fn as_str(data: &u32) -> &str {  
    // compute the string  
    let s = format!("{}", data);  
    &s  
}  
  
fn main() {  
    let x : u32 = 42;  
    let x_str = as_str(&x);  
    println!("Wow, {} is still {}!!!", x, x_str);  
}
```

Code examples

```
fn as_str(data: &u32) -> &str {  
    // compute the string  
    let s = format!("{}", data);  
    &s  
}  
  
fn main() {  
    let x : u32 = 42;  
    let x_str = as_str(&x);  
    println!("Wow, {} is still {}!!!", x, x_str);  
}
```

```
fn as_str(data: &u32) -> String {  
    // compute the string  
    let s = format!("{}", data);  
    s.to_string()  
}  
  
fn main() {  
    let x : u32 = 42;  
    let x_str = as_str(&x);  
    println!("Wow, {} is still {}!!!", x, x_str);  
}
```

Code examples

```
fn first_word<'a>(sentence: &'a str) -> &'a str {  
    let first_space = sentence.find(' ').unwrap_or(0);  
    let word = &sentence[..first_space];  
    return word;  
}  
  
fn main() {  
    println!("{}", first_word("Hello world from 'Dernier Cri'!"));  
}
```

Play with Rust @ <https://play.rust-lang.org/>

(Big) Rust projects

Alphabet



mozilla
Firefox[®]



Dropbox

OpenDNS

OpenDNS



Redox

(Big) Rust projects

... and more awesome projects!

<https://this-week-in-rust.org/>

So, welcome on board!



Do you want more ?

<https://www.rust-lang.org/fr/>

<https://www.codementor.io/rust/tutorial/steve-klabnik-rust-vs-c-go-ocaml-erlang>

<http://www.slideshare.net/yandex/rust-c>

<https://doc.rust-lang.org/stable/book/>

<http://rustbyexample.com/>

<http://thenewstack.io/safer-future-rust/>

<http://www.arewewebyet.org/>

[Please, click here!](#)