EXTENDING PYTHON WITH RUST

Richard Bownes

TALKING POINTS

Who I am

What I’m going to talk about

Why you should care

Learning Rust

Learning From Rust

Stuff to take away

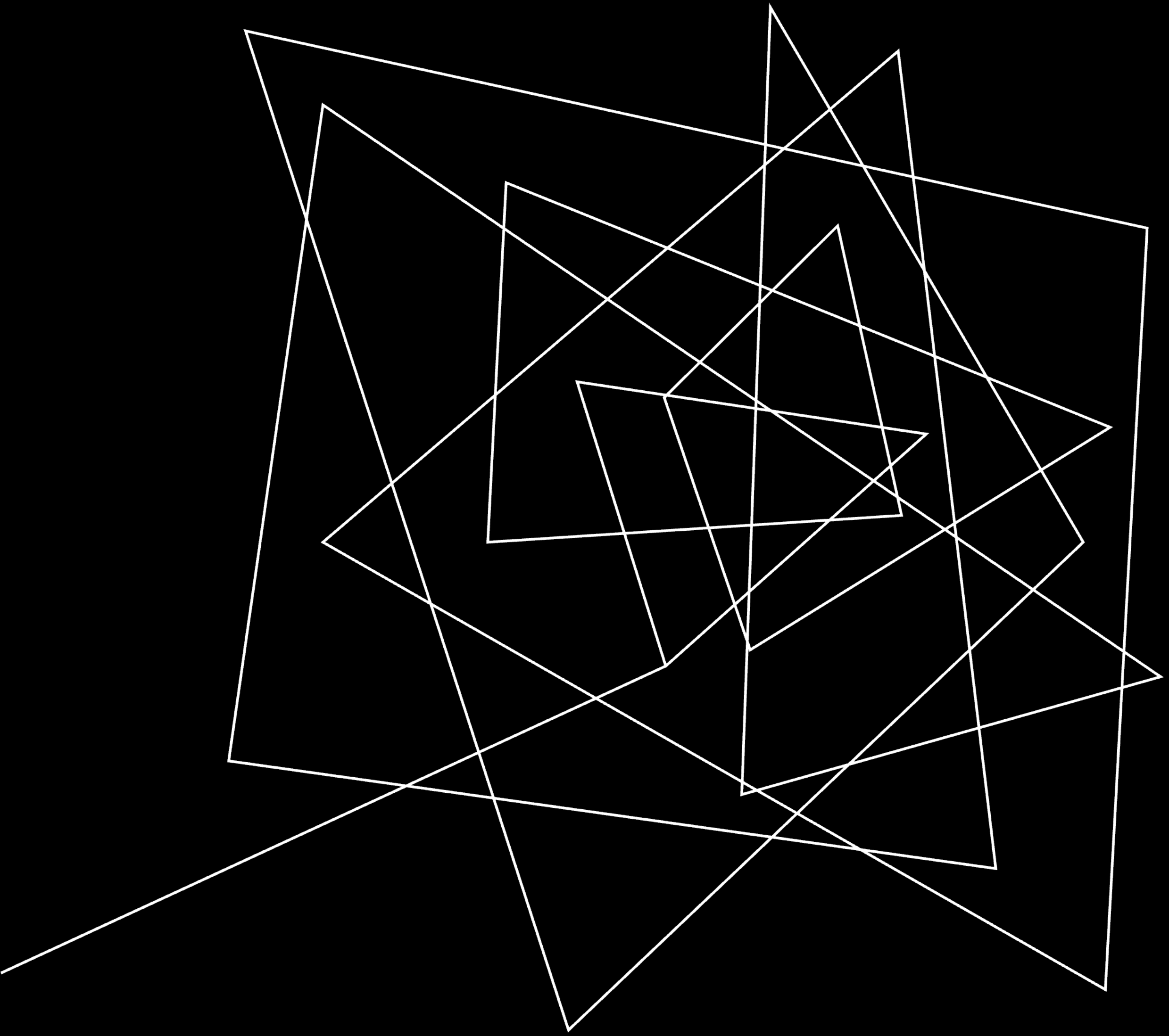
WHO I AM

- PhD in Machine Learning and Bioinformatics
- Data Science at the BBC Datalab
- NLP SAAS Start up
- Consultant
- OSS contributor

forecast

BBC datalab

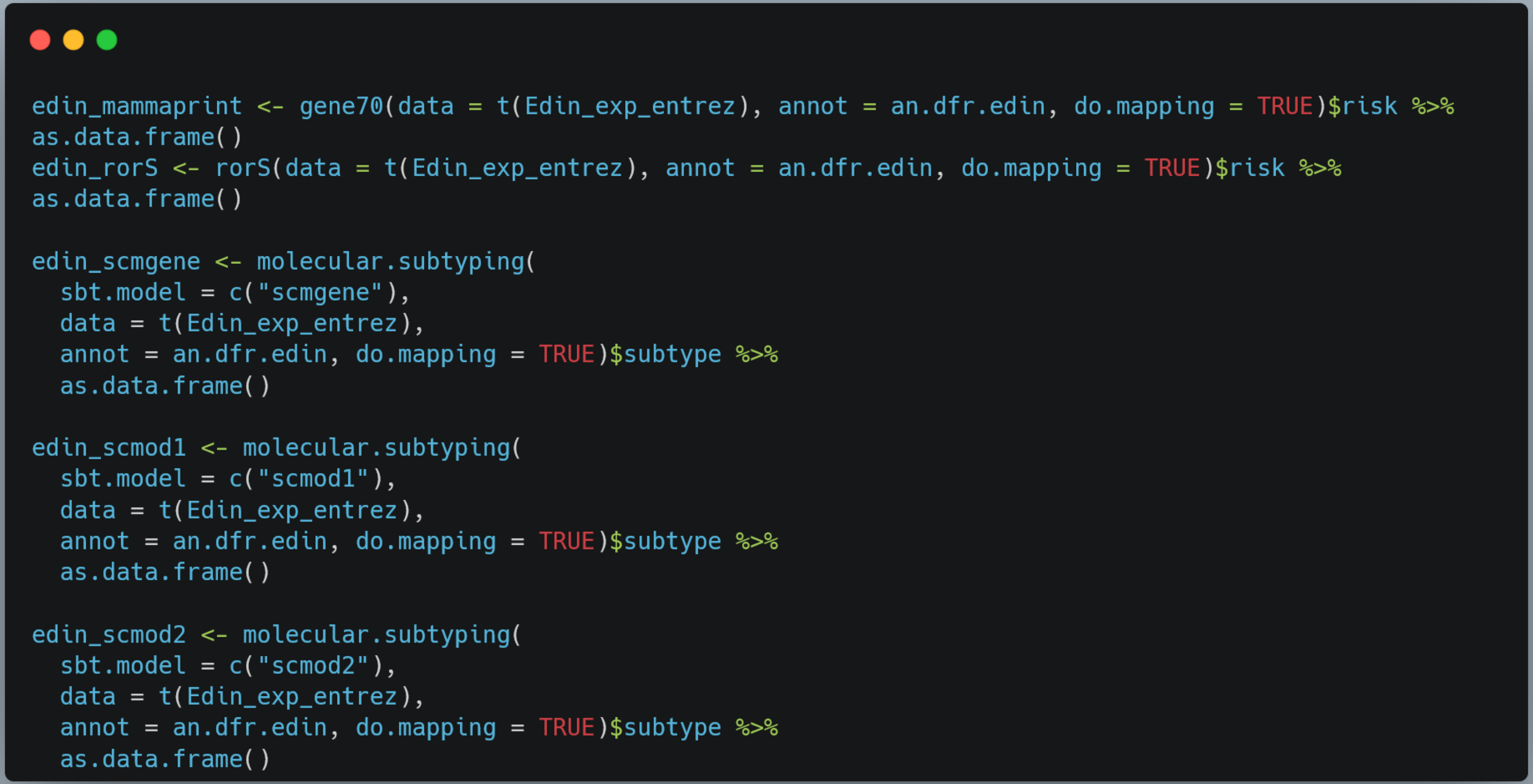
Hugging Face



WHY YOU SHOULD CARE

The level of entry

WHERE I STARTED



```r
edin_mammaprint <- gene70(data = t(Edin_exp_entrez), annot = an.dfr.edin, do.mapping = TRUE)$risk %>%
as.data.frame()
edin_rorS <- rorS(data = t(Edin_exp_entrez), annot = an.dfr.edin, do.mapping = TRUE)$risk %>%
as.data.frame()

edin_scmgene <- molecular.subtyping(
  sbt.model = c("scmgene"),
  data = t(Edin_exp_entrez),
  annot = an.dfr.edin, do.mapping = TRUE)$subtype %>%
  as.data.frame()

edin_scmod1 <- molecular.subtyping(
  sbt.model = c("scmod1"),
  data = t(Edin_exp_entrez),
  annot = an.dfr.edin, do.mapping = TRUE)$subtype %>%
  as.data.frame()

edin_scmod2 <- molecular.subtyping(
  sbt.model = c("scmod2"),
  data = t(Edin_exp_entrez),
  annot = an.dfr.edin, do.mapping = TRUE)$subtype %>%
  as.data.frame()
```

THINGS I WANTED TO DO BETTER

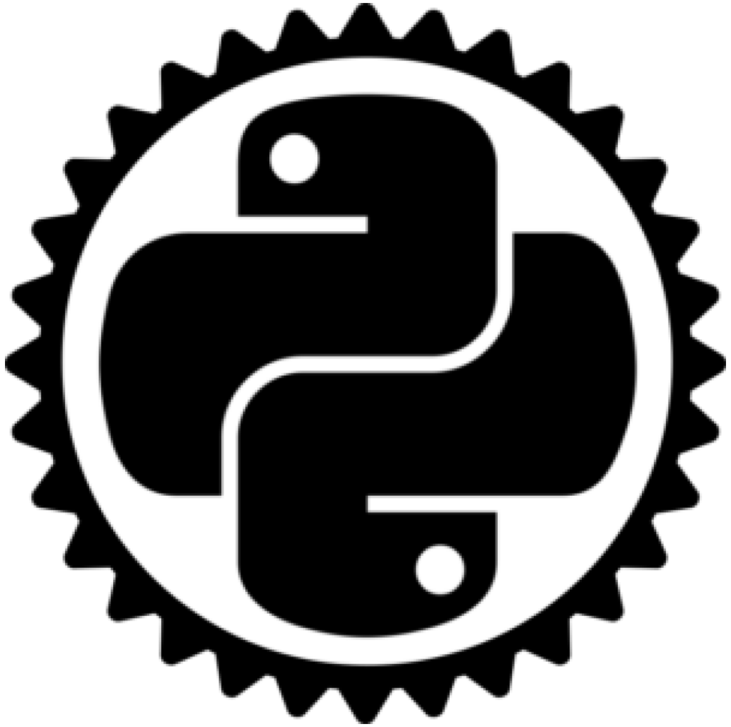
- Write faster code.
  - Overly indexed on optimisations.
  - Totally surplus to requirements.

- Write better code.
  - I knew I was sloppy, I had a lot of room to grow.

I HAD PROBLEMS TO SOLVE

Bioinformatics is underserved

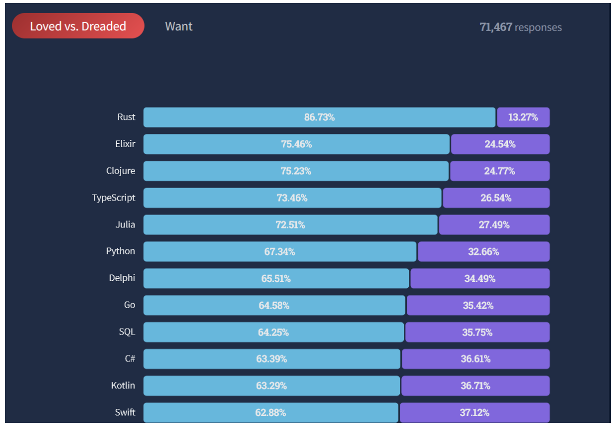
Lots of glue code between performant libraries



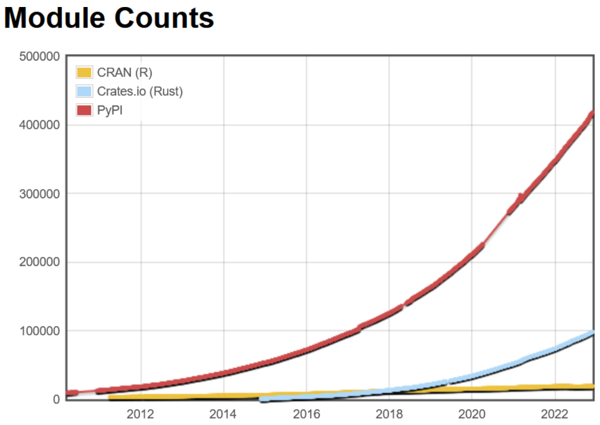
ENTER RUST

WHY RUST?

Well loved



Growing Community



Nice Features

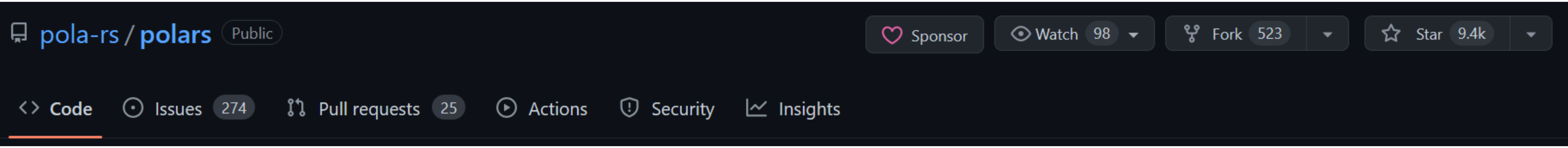
Guide Rails

Memory Safety

Fast

Tooling

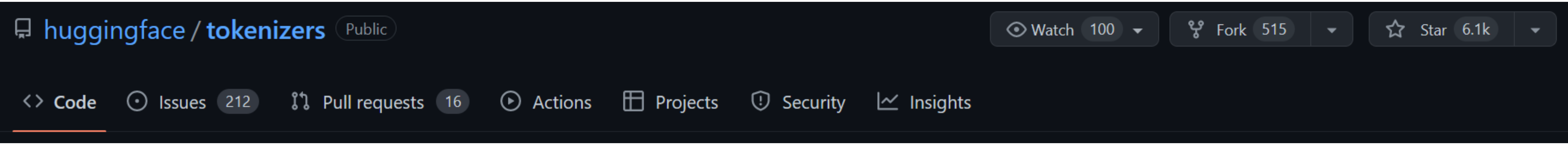
WHY RUST



pola-rs / polars Public

Sponsor | Watch 98 | Fork 523 | Star 9.4k

Code | Issues 274 | Pull requests 25 | Actions | Security | Insights



huggingface / tokenizers Public

Watch 100 | Fork 515 | Star 6.1k

Code | Issues 212 | Pull requests 16 | Actions | Projects | Security | Insights

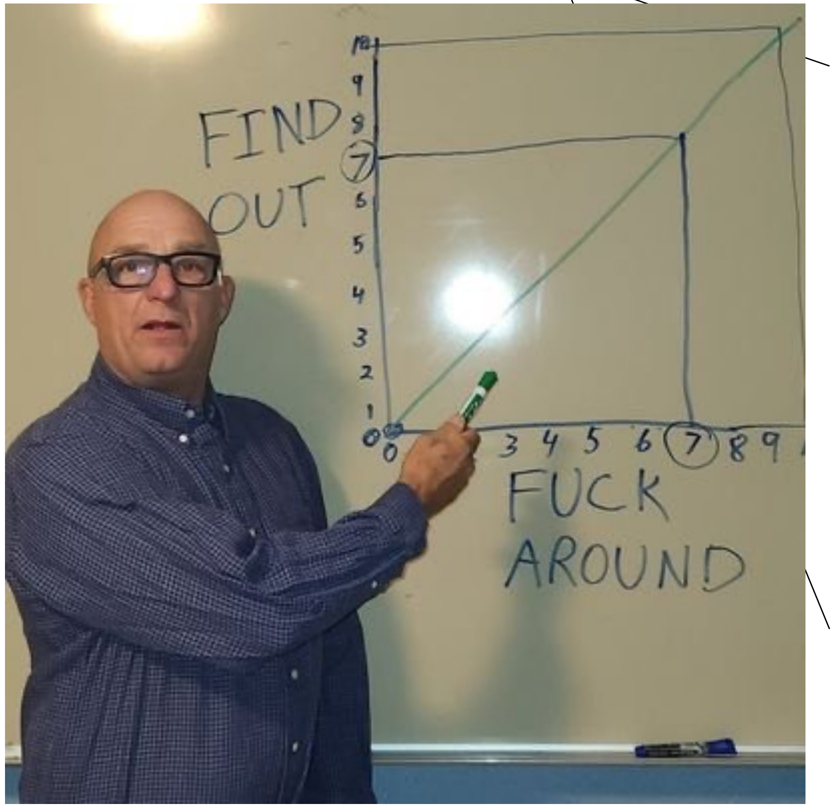
LEARNING RUST

Remarkably hard.

- https://doc.rust-lang.org/book/
- https://github.com/crazymykl/rust-koans
- Lets Get Rusty
- Jon Gjengset

IN WAY OVER MY HEAD

- I did not know how much my hand was held by dynamic typing.
- I don’t know types.
- I don’t handle errors.
- ?/Result<>/unwrap/panic!/Ok/Err
- scopes/ownership/borrowing/lifetimes
- Mutability and immutability



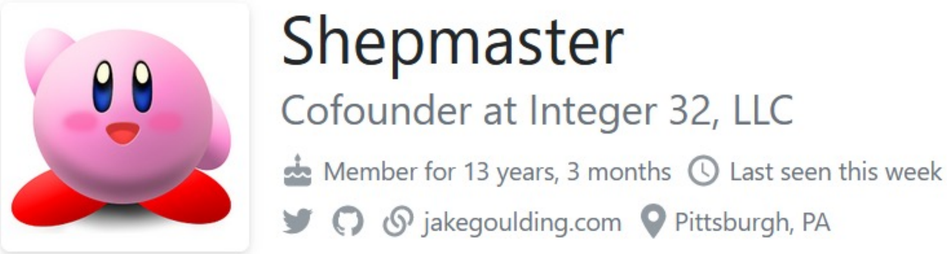
WHAT HELPED?

Just trying to build things not in The Book.

Unironically – Reddit.

The compiler.

This guy.



Shepmaster

Cofounder at Integer 32, LLC

Member for 13 years, 3 months · Last seen this week

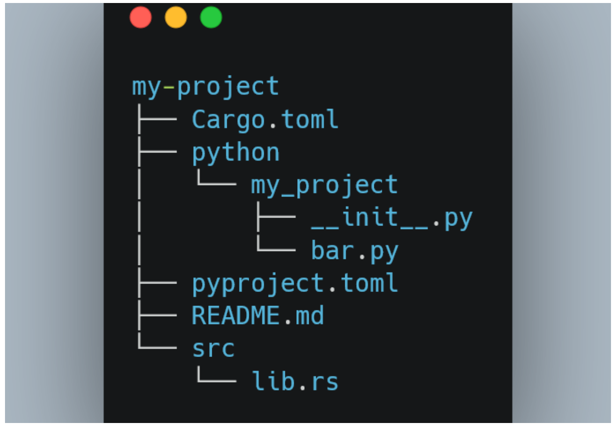
jakegoulding.com · Pittsburgh, PA

EXTENDING PYTHON

Pyo3 + Maturin

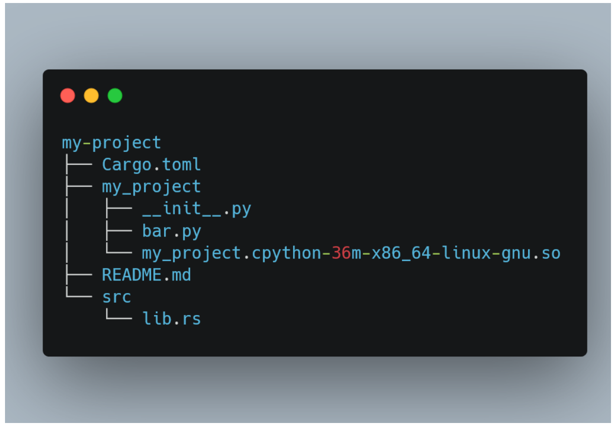
SUPER SIMPLE SET UP

Create a mixed directory



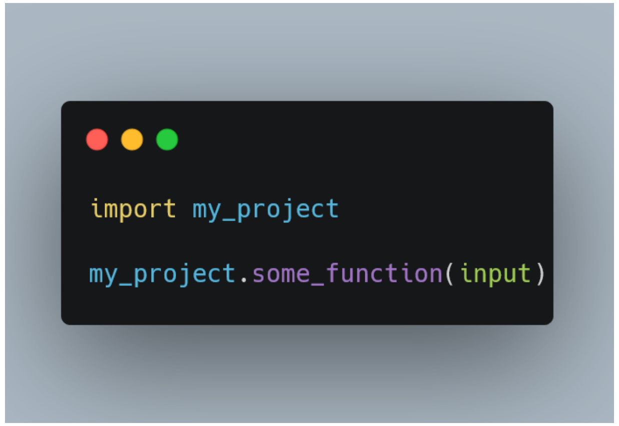
```
my-project
├── Cargo.toml
├── python
│   └── my_project
│       ├── __init__.py
│       └── bar.py
├── pyproject.toml
├── README.md
└── src
    └── lib.rs
```

Write Rust code and develop



```
my-project
├── Cargo.toml
├── my_project
│   ├── __init__.py
│   ├── bar.py
│   └── my_project.cpython-36m-x86_64-linux-gnu.so
├── README.md
└── src
    └── lib.rs
```

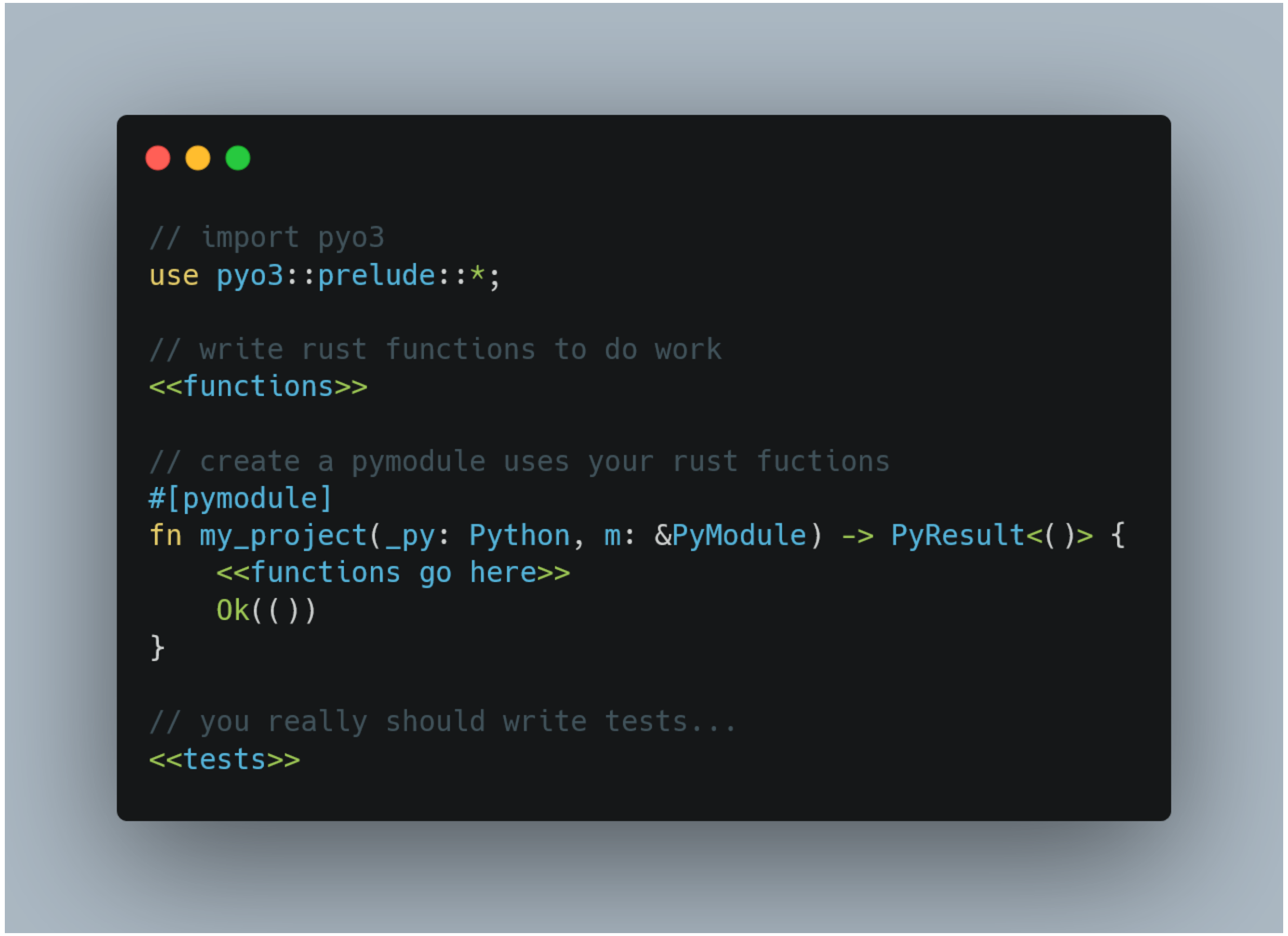
Import to python



```
import my_project

my_project.some_function(input)
```

BLUEPRINT FOR EXTENDING PYTHON



```rust
// import pyo3
use pyo3::prelude::*;

// write rust functions to do work
<<functions>>

// create a pymodule uses your rust fuctions
#[pymodule]
fn my_project(_py: Python, m: &PyModule) -> PyResult<()> {
    <<functions go here>>
    Ok(())
}

// you really should write tests...
<<tests>>
```

WHAT TO BUILD?

Faster non-vectorized stuff

- Large computations with no GPU supported libraries or memory limitations.
    - Chunked matrix cosine similarity
- Large lists in python I couldn’t find a good way to vectorise: hand off to rust.

Glue code replacements

- Wrangling or cleaning some data
- Better case matching handled out of python

LEARNING FROM RUST

Writing better code

- Type hints and annotation in python
- Better testing and code coverage
- More performant code
- Understanding (a little) what python is doing under the hood.

Being more flexible

- There isn’t a library for everything
- But you can do it yourself
- Most of Python is just pointing to other languages, join them

Learning Basics (last)

- Went back to the basics of what is provided by a language.
- Basic data structures.
- CRUD
- I/O
- Optimized queries
- Lazy vs. eager
- Complexity and scaling

SUMMARY

Learning Rust made me better at Rust. It also made me better at Python and my job. I think it can do the same for other people.

STUFF TO TAKE AWAY

There is no one sized fits all way to learn something.

Learning more languages gives you more tools and ways to think about problems.

Learning Rust or C or C++ as an extension of a more familiar language is a functional way of doing this.

THANK YOU

Richard Bownes

Richard.bownes@forecast.global

www.linkedin.com/in/rjbownes

www.github.com/rbownes