Give Rust a Chance

I’m Slater

- I invest in tools for data teams at Bain Capital Ventures.
- I started thinking about Rust through our investment in Polars.

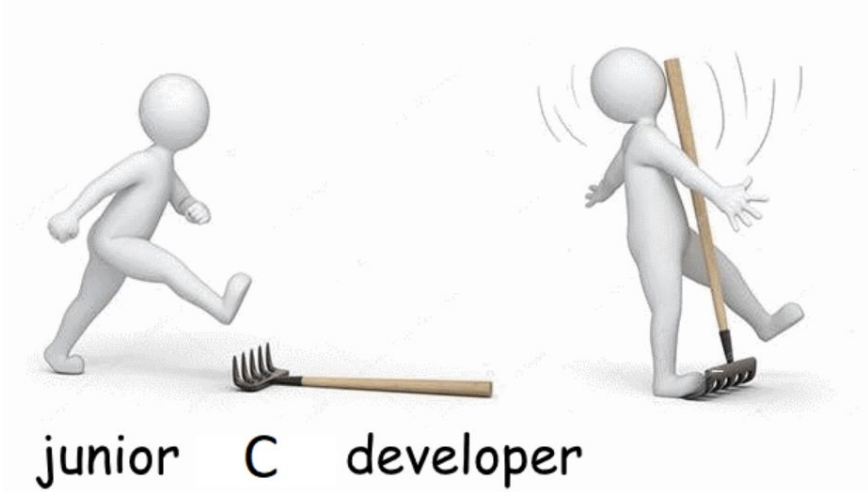
The Problem

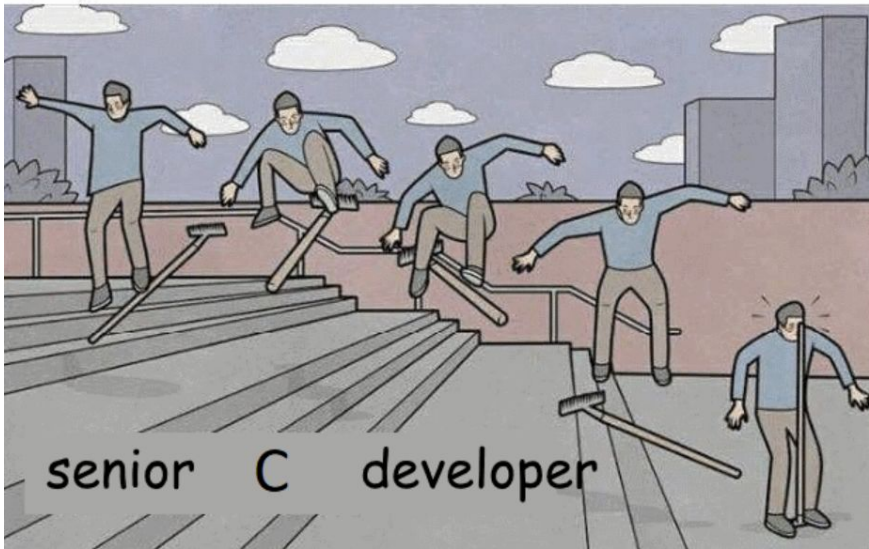
- Python is the most popular language for data science.
- Python is slow.

- Therefore: If you’re writing a high-performance library for data teams, you need the “frontend” to be in Python and the “backend” to be in some other language.

The Historical Solutions:

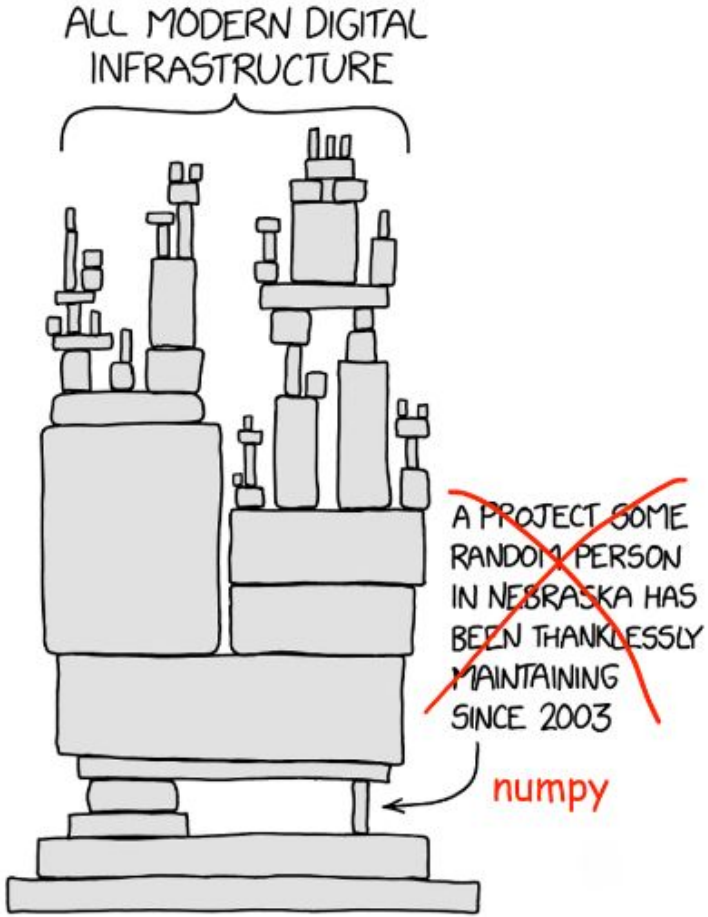
- “Just learn C”
- Yes, but:
  - Writing C is hard, and it’s easy to make subtle mistakes.
  - It’s not a fun language to dabble in part-time.





The Historical Solutions:

- Hope that somebody else learned C.
- Yes, but:
  - I like to write for loops. Vectorizing everything is a pain.
  - It's limiting to never be able to touch the internals.



Why is Rust the One True Solution?

Why is Rust the One True Solution?

Because you can write bad Rust.

Why is Rust the One True Solution?

Because you can write bad Rust.
And it’s fine.

Matrix Multiplication

Subtly wrong C

```c
int **multiplyMatrices(int mat1[][2], int mat2[][2], int row1, int col1, int col2) {
  int **result = (int **)malloc(row1 * sizeof(int *));

  for (int i = 0; i < row1; i++) {
    result[i] = (int *)malloc(col2 * sizeof(int));
  }

  for (int i = 0; i < row1; i++) {
    for (int j = 0; j < col2; j++) {
      for (int k = 0; k < col1; k++) {
        result[i][j] += mat1[i][k] * mat2[k][j];
      }
    }
  }

  return result;
}
```

Probably correct Rust

```rust
fn multiply_matrices(
  mat1: &[[i32; 2]],
  mat2: &[[i32; 2]],
  row1: usize,
  col1: usize,
  col2: usize) -> Vec<Vec<i32>> {
    let mut result = vec![vec![0; col2]; row1];

    for i in 0..row1 {
        for j in 0..col2 {
            for k in 0..col1 {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }

    result
}
```

Rust Stops You from Doing Bad Things

- Rust is memory safe.
- Rust has a strict compiler.
- Rust has nice concurrency primitives.
- Etc., etc.

This is Freeing

- It’s harder to make mistakes.
- It’s easier to find the mistakes you do make.
- And those mistakes are less costly.

- You should try it.

Thank You