PyO3: Python Loves Rust

Moshe Zadka – https://cobordism.com

Acknowledgement of Country

Belmont (in San Francisco Bay Area Peninsula)
Ancestral homeland of the Ramaytush Ohlone people

Rust: Intro

Rust: Intro

What

Rust: Intro

What
Why

Rust: Intro

What
Why
How

Rust: What?

Rust: What?

Low-level

Rust: What?

Low-level
Zero-cost abstractions

Rust: What?

Low-level
Zero-cost abstractions
Memory safe!

Rust: Why?

Rust: Why?

Performance

Rust: Why?

Performance
Safety

Rust: Why?

Performance
Safety
"Low-level parsing"

Toy Example: Counting

Toy Example: Counting

Check whether character appears more than X times

Toy Example: Counting

Check whether character appears more than X times
Optionally, reset counts on spaces/newlines

Toy Example: Counting

Check whether character appears more than X times
Optionally, reset counts on spaces/newlines
”Toy example”

Toy Example: Counting

Check whether character appears more than X times
Optionally, reset counts on spaces/newlines
"Toy example"
Just interesting enough

Rust example: Enum

```
enum Reset {
    NewlinesReset,
    SpacesReset,
    NoReset,
}
```

Rust example: Struct

```
struct Counter {
    what: char,
    min_number: u64,
    reset: Reset,
}
```

Rust example: Impl

```
impl Counter {
    fn has_count(
        &self,
        data: &str,
    ) -> bool {
        has_count(self, data.chars())
    }
```

Rust example: Loop

```
fn has_count(cntr: &Counter, chars: std::str::Chars
    let mut current_count : u64 = 0;
    for c in chars {
        if got_count(cntr, c, &mut current_count) {
            return true;
        }
    }
    false
}
```

Rust example: Counting

```
fn got_count(cntr: &Counter, c: char, current_count
    if *current_count >= cntr.min_number {
        return true;
    }
    maybe_reset(cntr, c, current_count);
    maybe_incr(cntr, c, current_count);
    false
}
```

Rust example: Reset

```
fn maybe_reset(cntr: &Counter, c: char, current_cou
    match (c, cntr.reset) {
        ('\n', Reset::NewlinesReset) | (' ', Reset:
            *current_count = 0;
        }
        _ => {}
    };
}
```

Rust example: Increment

```
fn maybe_incr(cntr: &Counter, c: char, current_coun
    if c == cntr.what {
        *current_count += 1;
    };
}
```

PyO3

Inline

PyO3

Inline
Modify together

PyO3 example: Include

```
use pyo3::prelude::*;
```

PyO3 example: Wrap enum

```
#[pyclass]
#[derive(Clone)]
#[derive(Copy)]
enum Reset {
    /* ... */
}
```

PyO3 example: Wrap struct

```
#[pyclass]
struct Counter {
    /* ... */
}
```

PyO3 example: Wrap impl

```
#[pymethods]
impl Counter {
    #[new]
    fn new(what: char, min_number: u64, reset: Rese
        Counter{what: what, min_number: min_number,
    }
    /* ... */
}
```

PyO3 example: Define module

```
#[pymodule]
fn counter(_py: Python, m: &PyModule) -> PyResult<(
    m.add_class::<Counter>()?;
    m.add_class::<Reset>()?;
    Ok(())
}
```

Maturin develop

```
(venv)$ maturin develop
```

Maturin build

```
(venv)$ maturin build
```

Python

Use!

Import

```
import counter
```

Constructor

```
cntr = counter.Counter('c', 3, counter.Reset.Newlin
```

Call

```
cntr.has_count(”hello–c–c–c–goodbye”)

True
```

Call

```
cntr.has_count("hello-c-c-\nc-goodbye")
```

```
False
```

Take-aways

Why?

Rust + Python

Easy!

Differences

Differences

Rust:

Differences

Rust: High-performance,

Differences

Rust: High-performance, safe,

Differences

Rust: High-performance, safe, learning curve,

Differences

Rust: High-performance, safe, learning curve, awkward prototyping

Differences

Rust: High-performance, safe, learning curve, awkward prototyping
Python:

Differences

Rust: High-performance, safe, learning curve, awkward prototyping
Python: Easy,

Differences

Rust: High-performance, safe, learning curve, awkward prototyping
Python: Easy, tight iteration,

Differences

Rust: High-performance, safe, learning curve, awkward prototyping
Python: Easy, tight iteration, Speed cap

Combined

Combined

Prototype in Python

Combined

Prototype in Python
Move perf bottlenecks to Rust

Combined

Prototype in Python
Move perf bottlenecks to Rust

Stronger together

Stronger together

Deployment

Stronger together

Deployment
Development

Stronger together

Deployment
Development
Enjoy!