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
I live in Belmont, in the San Francisco Bay Area Peninsula. I wish to acknowledge it as the ancestral homeland of the Ramaytush Ohlone people.

0.1 Short Intro to Rust

Rust: Intro

What
Why
How

0.1.1 What is Rust?

Rust: What?

Low-level
Zero-cost abstractions
Memory safe!

0.1.2 Why is Rust?

Rust: Why?

Performance
Safety
"Low-level parsing"

0.1.3 Counting characters

Toy Example: Counting

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

0.1.4 Enum

Rust example: Enum

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

0.1.5 Struct

Rust example: Struct

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

0.1.6 Implementation

Rust example: Impl

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

0.1.7 Function

Rust example: Loop

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

0.1.8 Counting

Rust example: Counting

```
fn got_count(cntr: &Counter, c: char, current_count: &mut u64) -> bool {
    if *current_count >= cntr.min_number {
        return true;
    }
    maybe_reset(cntr, c, current_count);
    maybe_incr(cntr, c, current_count);
    false
}
```

0.1.9 Reset

Rust example: Reset

```
fn maybe_reset(cntr: &Counter, c: char, current_count: &mut u64) -> () {
    match (c, cntr.reset) {
        ('\n', Reset::NewlinesReset) | (' ', Reset::SpacesReset)=> {
            *current_count = 0;
        }
        _ => {}
    };
}
```

0.1.10 Increment

Rust example: Increment

```
fn maybe_incr(cntr: &Counter, c: char, current_count: &mut u64) -> (){
    if c == cntr.what {
        *current_count += 1;
    };
}
```

0.2 PyO3

PyO3
Inline
Modify together

0.2.1 Include

PyO3 example: Include

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

0.2.2 Wrap enum

PyO3 example: Wrap enum

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

0.2.3 Wrap struct

PyO3 example: Wrap struct

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

0.2.4 Wrap impl

PyO3 example: Wrap impl

```
#[pymethods]
impl Counter {
    #[new]
    fn new(what: char, min_number: u64, reset: Reset) -> Self {
        Counter{what: what, min_number: min_number, reset: reset}
    }
    /* ... */
}
```

0.2.5 Define module

PyO3 example: Define module

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

0.2.6 Maturin develop

Maturin develop

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

0.2.7 Maturin develop

Maturin build

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

0.3 Python

Python

Use!

0.3.1 Import

Import

```
import counter
```

0.3.2 Construct

Constructor

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

0.3.3 Call

Call

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

```
True
```

0.3.4 Call

Call

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

```
False
```

0.4 Conclusion

Take-aways
Why?

0.4.1 Rust and Python is easy

Rust + Python
Easy!

0.4.2 Use each one for its purposes

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

Combined
Prototype in Python
Move perf bottlenecks to Rust