pyGRAZ

RustPython: a Python implementation in Rust

Lightning Talk
Lukas Prokop, 2019/03/05

http://lukas-prokop.at/talks/pygraz-rustpython

Rust programming language

- Since 2010, Mozilla Research
- focused on safety, especially safe concurrency
- “most loved programming language” in the Stack Overflow Developer Survey for 2016, 2017, and 2018

```
use std::fmt;
impl fmt::Display for Universe {
    fn fmt(&self, f: &mut fmt::Formatter) -> fmt::Result {
        for line in self.cells.as_slice().chunks(self.width as usize) {
            for &cell in line {
                let symbol = if cell == Cell::Dead { '◻' } else { '◼' };
                write!(f, "{}", symbol)?;
            }
            write!(f, "\n")?;
        }
        Ok(())
    }
}
```

Python 3.5

Reminder of Python 3.5 features:

- coroutines with async and await syntax
- a new matrix multiplication operator: a @ b
- additional unpacking generalizations
- Adding % formatting to bytes and bytearray
- RecursionError
- typing package
- zipapp package
- …

https://docs.python.org/3/whatsnew/3.5.html

RustPython

Python 3 interpreter written in Rust.

By Windel Bouwman and Shing Lyu.

- Repo at github.com/RustPython
- talk at FOSDEM 2018, 2019/02/03
- API documentation

Usecases via rspython:

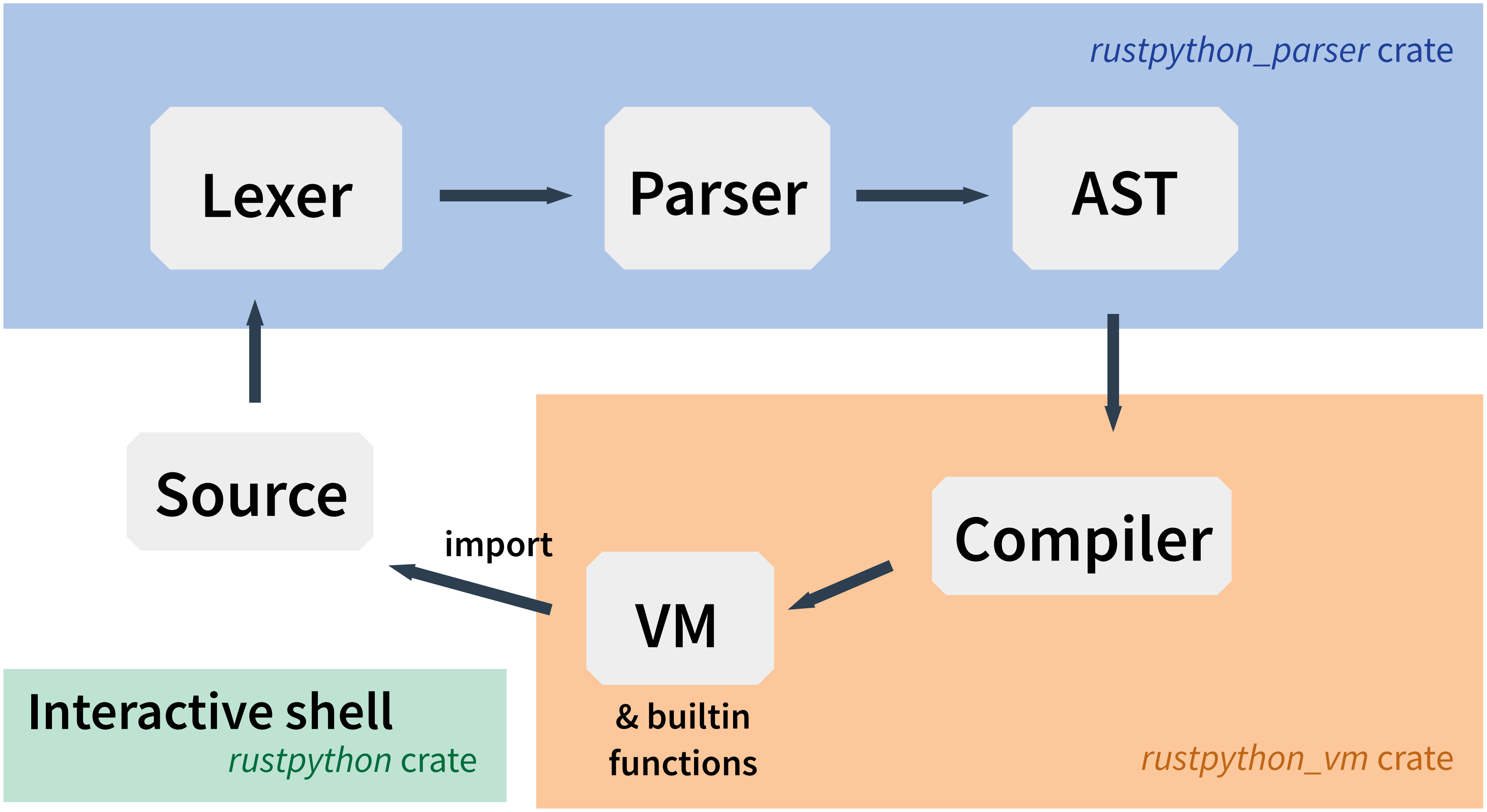
- Compile RsPython to webassembly and run python3 scripts in the browser
- Port python to Redox-os.
- Provide an alternative implementation next to CPython
- Combine rust with python in a more natural way

Goals

Goals of pythonvm-rust:

- Compatible with CPython 3.6's bytecode, in order to take advantage of FAT Python
- Support CPython's implementation of the standard library
- No crash, even when messing with code objects
- Bytecode optimizations at runtime
- Less bounded by the GIL than CPython

Design


rustpython_parser crate
Lexer
Parser
AST
Source
Compiler
import
VM
& builtin functions
rustpython_vm crate
Interactive shell
rustpython crate

Issue pytecode (typo intended)

```
>>> def f():
...     return '{var} = {value}'.format(
...         var='a', value=5
...     )
...
```

```
Python 3.2.3 [GCC 4.7.2] on linux2
>>> import dis
>>> dis.dis(f)
  1   0 LOAD_CONST      1 ('{var} = {value}')
      3 LOAD_ATTR       0 (format)
      6 LOAD_CONST      2 ('var')
      9 LOAD_CONST      3 ('a')
     12 LOAD_CONST      4 ('value')
     15 LOAD_CONST      5 (5)
     18 CALL_FUNCTION 512
     21 RETURN_VALUE
```

```
Python 3.6.7 [GCC 8.2.0] on linux
>>> import dis
>>> dis.dis(f)
  1   0 LOAD_CONST         1 ('{var} = {value}')
      2 LOAD_ATTR          0 (format)
      4 LOAD_CONST         2 ('a')
      6 LOAD_CONST         3 (5)
      8 LOAD_CONST         4 (('var', 'value'))
     10 CALL_FUNCTION_KW   2
     12 RETURN_VALUE
```

CPython and RustPython

How does rust help in the implementation?

- Arbitrary precision integers by BigInt
- Option<Value> to avoid null-pointer dereferences
- Reference counting implemented with Rc and RefCell

Challenge: the Python dict

- Rust has a HashMap type
- To implement Python the dict type, HashMap is tempting, but…
- Every python object can be a dict key, if it implements __hash__ and __eq__.
- Both these methods can raise an exception…
- HashMap does not permit for failing hashes…
- Now what? Own hash map implementation? 😖

Showcase

```
meisterluk@gardner ~/RustPython % cargo run demo.py
    Updating crates.io index
  Downloaded num-traits v0.2.6
  Downloaded num-rational v0.2.1
[...]
   Compiling lalrpop v0.15.2
   Compiling rustpython_parser v0.0.1 (/home/meisterluk/dev/RustPython/parser)
   Compiling rustpython_vm v0.1.0 (/home/meisterluk/dev/RustPython/vm)
   Compiling rustpython v0.0.1-pre-alpha.1 (/home/meisterluk/dev/RustPython)
    Finished dev [unoptimized + debuginfo] target(s) in 3m 31s
     Running `target/debug/rustpython demo.py`
Hello, RustPython!
meisterluk@gardner ~/RustPython % cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.18s
     Running `target/debug/rustpython`
Welcome to the magnificent Rust Python 0.0.1-pre-alpha.1 interpreter
No previous history.
>>>>> 2**200
1606938044258990275541962092341162602522202993782792835301376
>>>>> def f():
.....   return 3 + 5
>>>>> f()
8
```

Other interesting facts

Why?

- 14 CVEs for python 3.5:
- https://security-tracker.debian.org/tracker/source-package/python3.5

How?

- 8.2 MB RustPython executable on Linux
- 2.6 MB RustPython WebAssembly

Thanks!

Thanks!