FFI in Rust

Lessons from OpenSSL

Steven Fackler - sfackler

December 15, 2016

Basics

-sys Crates

Basics

What is FFI?

"A foreign function interface (FFI) is a mechanism by which a program written in one programming language can call routines or make use of services written in another."

Why FFI?

Lots of code has been written in languages that aren't Rust!

- Use Rust to accelerate portions of programs written in higher level languages. (jni, neon, ruru)
- Expose a C interface for your Rust library. (regex)
- Interact with libraries exposing C APIs. (openssl, curl, rusqlite)

Example

```
struct foo {
    long a;
    char *b;
};
int fizzbuzz(const struct foo *f);
```

```
#[repr(C)]
struct foo {
    a: c_long,
    b: *mut c_char,
}
extern {
    fn fizzbuzz(f: *const foo) -> c_int;
}
```

-sys Crates

-sys?

A '-sys' crate provides a direct transcription of a C API into Rust - think of them as Rust's version of header files.

Also responsible for linking to the C library.

Bindgen

Bindgen uses libclang to parse C headers and produce Rust definitions from them.

```
$ bindgen fizzbuzz.h
/* automatically generated by rust-bindgen */

#![allow(dead_code,
         non_camel_case_types,
         non_upper_case_globals,
         non_snake_case)]
#[repr(C)]
#[derive(Copy, Clone)]
#[derive(Debug)]
pub struct foo {
    pub a: ::std::os::raw::c_long,
...
```

Caveats

While Bindgen is fantastic to quickly get Rust bindings for a header, it has some drawbacks:

- libclang is a fairly heavyweight dependency.
- It produces definitions valid in the environment it's run.
- It can be a bit verbose, particularly when system headers are included.

rust-openssl does not use bindgen for its bindings.