Crates with Different Traits:

Learning more about Cargo build scripts

❤ Kevin Martin (S1’18) ❤

Background

Cargo is a tool that allows Rust projects to declare their various dependencies and ensure that you’ll always get a repeatable build. To accomplish this goal, Cargo does four things:

- Introduces two metadata files with various bits of project information.
- Fetches and builds your project’s dependencies.
- Invokes rustc or another build tool with the correct parameters to build your project.
- Introduces conventions to make working with Rust projects easier.

Source: https://doc.rust-lang.org/beta/cargo/guide/why-cargo-exists.html

Shameless Plug:

If this sounds interesting, come swing by the Rust Study Group!

Build Scripts (What? Why?)

What? The Rust file designated by the build command will be compiled and invoked before anything else is compiled in the package, allowing your Rust code to depend on the built or generated artifacts. (This is usually called `build.rs`)

Why? Some packages need to:

- Compile third-party non-Rust code, for example C libraries.
- Link to C libraries (which may need to be built from source).
- Others still need facilities for functionality such as code generation before building (think parser generators).

Build Scripts (Inputs)

When the build script is run, there are a number of inputs to the build script, all passed in the form of environment variables.

CARGO_MANIFEST_DIR - The directory containing the manifest for the package being built (the package containing the build script). Also note that this is the value of the current working directory of the build script when it starts.

OUT_DIR - the folder in which all output should be placed. This folder is inside the build directory for the package being built, and it is unique for the package in question.

Build Scripts (Outputs)

All the lines printed to stdout by a build script are written to a file like this:
`target/debug/build/<pkg>/output`

Any line that starts with cargo: is interpreted directly by Cargo. This line must be of the form
`cargo:key=value`

Build Scripts (Dependencies)

Build scripts are also allowed to have dependencies on other Cargo-based crates. Dependencies are declared through the build-dependencies section of the manifest.

The build script does not have access to the dependencies listed in the dependencies or dev-dependencies section (they’re not built yet!). All build dependencies will also not be available to the package itself unless explicitly stated as so.

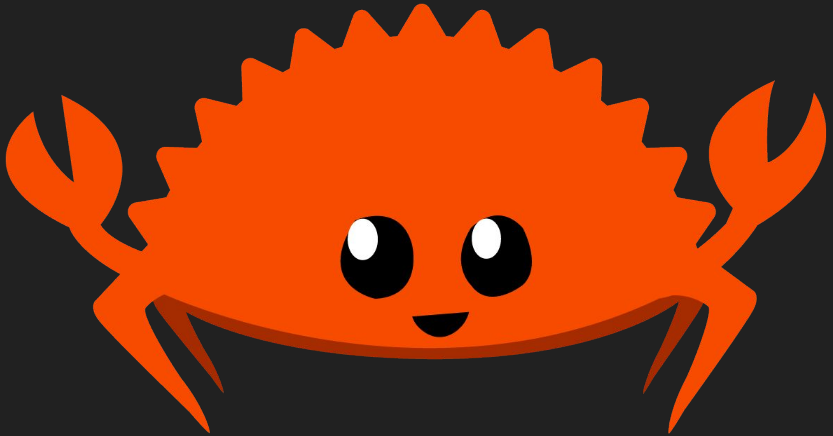
Takeaways:

Starting this project seemed conceptually intimidating at first.

- I don’t have a background working as a systems programmer!
- I’ve never worked with compilers before!
- I’ve only been writing Rust for ~3 months!

Takeaways:

“We believe Rust has potential as an enabling technology, empowering a people who would not traditionally do system programming to take it on with confidence.”



-Carol Nichols

Thanks!