Cross-Compile Rust, Targeting Windows from Linux

Tested using Ubuntu MATE 21.10 as a host system. Instructions might require some adjustment for your distro. Assumes Rust is already installed.

Make sure it's up-to-date:

```
rustup update
```

List currently installed toolchains:

```
rustup show
```

```
Default host: x86_64-unknown-linux-gnu
rustup home: /home/jimc/.rustup

stable-x86_64-unknown-linux-gnu (default)
rustc 1.56.0 (09c42c458 2021-10-18)
```

Your results may differ. Bottom line for me, though, is that I don't yet have a Windows toolchain installed. I installed a Windows target and toolchain with this:

```
rustup target add x86_64-pc-windows-gnu

rustup toolchain install stable-x86_64-pc-windows-gnu
```

Create a test project:

```
mkdir crossplat

cd crossplat

cargo init --vcs none
```

A simple main() is generated:

main.rs

```
fn main() {
    println!("Hello, world!");
}
```

Make sure it builds and runs:

```
cargo run
```

```
Hello, world!
```

Install your distribution’s MinGW C compiler:

```
sudo apt install gcc-mingw-w64-x86-64
```

Build the project, targeting Windows:

```
cargo build --target x86_64-pc-windows-gnu
```

Checking the file type of the generated .exe shows us that it’s a binary for Windows:

```
file target/x86_64-pc-windows-gnu/debug/crossplat.exe
```

Result:

```
target/x86_64-pc-windows-gnu/debug/crossplat.exe: PE32+ executable (console)
x86-64, for MS Windows
```

When you’re ready to deploy to your target system, build the release version with this:

```
cargo build --target x86_64-pc-windows-gnu --release
```

cross platform, rust