OpenGL

- LEARN OPENGL: https://github.com/bwasty/learn-opengl-rs?tab=readme-ov-file
    - Recommends gl+glfw. I'll take glow, almost the same thing but more rust-like.
- Glium: very high level abstraction, https://crates.io/crates/glium, no longer maintained
- Gl46: opengl4.6 bindings https://crates.io/crates/gl46
- Gfx, gfx-hal, pre-ll, wgpu?? https://github.com/gfx-rs/gfx
- Gl: https://crates.io/crates/gl unsafe bindings
- Glow: https://crates.io/crates/glow, unsafe bindings
- Glutin context provider?? https://crates.io/crates/glutin
- GLFW: window stuff?: https://crates.io/crates/glfw

> https://www.reddit.com/r/rust/comments/4nrldm/opengl_in_rust/
> Author of gl-rs here! Here it is the state of OpenGL in Rust at a glance:

gl-rs: An OpenGL function pointer loader for the Rust Programming Language. (Pretty much like GLEW)

gl_generator: Used to generate gl-rs, but has options to generate custom bindings if you want more extensions than are provided by default in gl-rs.

glutin: Like GLFW - but the build system is much easier to get up and running than glfw-rs, and most of it is written in Rust.

glium: Safe abstraction over OpenGL (built on top of gl_generator). Tied to Glutin for windowing.

gfx: Safe, common abstraction over OpenGL and other graphics APIs (OpenGL part also built on top of gl_generator). Not tied to a specific windowing library.

> https://www.reddit.com/r/rust/comments/11yexxk/opengl_crates_gl_vs_glow_vs_glium/

Hey, I'm working on inox2d which uses glow.

glow is really great as a Rusty wrapper for OpenGL calls (they're still all unsafe but at least they're more idiomatic to Rust). Basically all it does is to provide small wrapper functions that directly map to OpenGL functions, except they're in snake_case, are methods of a Context instead of being global, and sometimes have different arguments that cater better to Rust, like replacing pointer offsets with unsigned integers, data pointers with Rust slices, and stuff where 0 indicates absence to Option.