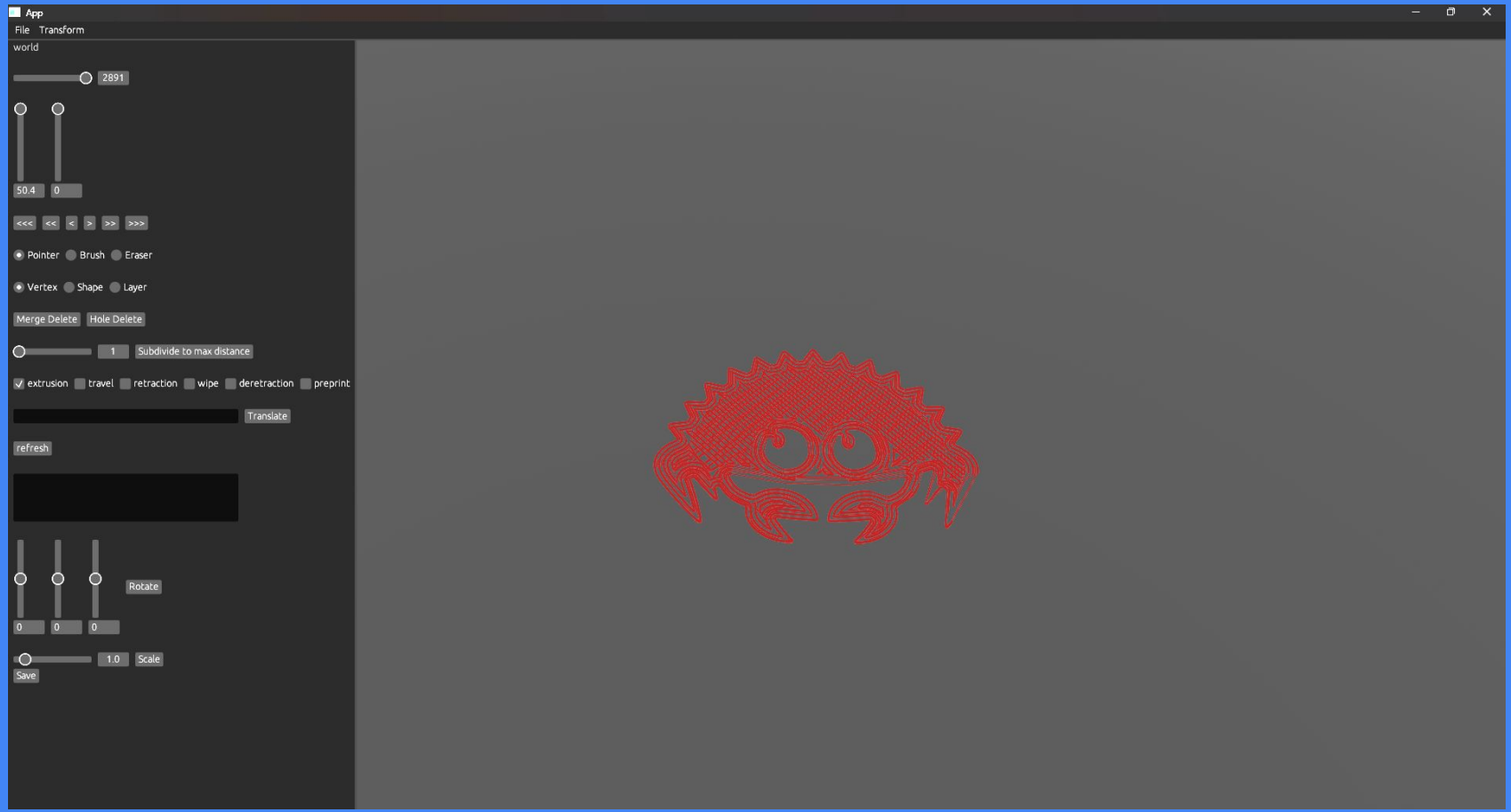
g-wiz: the visual .gcode editor

James Dietz

The App


App
File
Transform
world
2891
50.4
0
<<<
<<
<
>
>>
>>>
Pointer
Brush
Eraser
Vertex
Shape
Layer
Merge Delete
Hole Delete
1
Subdivide to max distance
extrusion
travel
retraction
wipe
deretraction
preprint
Translate
refresh
Rotate
0
0
0
1.0
Scale
Save

What is G-Code?

- G-Code is a programming language that controls CNC machines
- Generally consists of letter and number pairs called “Words”
- Firmware makers can and do whatever they want (single characters, strings, etc)
- Most common command is G1 which moves motors and sets feedrate

Example G-Code Snippets

```
M862.6 P"Input shaper" ; FW feature check
M115 U6.0.1+14848
G90 ; use absolute coordinates
M83 ; extruder relative mode
M104 S170 ; set extruder temp for bed leveling
M140 S60 ; set bed temp
M109 R170 ; wait for bed leveling temp
M190 S60 ; wait for bed temp
M569 S1 X Y ; set stealthchop for X Y
M204 T1250 ; set travel acceleration
G28 ; home all without mesh bed level
G29 ; mesh bed leveling
M104 S230 ; set extruder temp
G92 E0

G1 X0 Y-2 Z3 F2400

M109 S230 ; wait for extruder temp

; intro line
G1 X10 Z0.2 F1000
G1 X70 E8 F900
M73 P1 R1
M73 Q1 S1
G1 X140 E10 F700
M73 P6 R1
M73 Q5 S1
G92 E0

M569 S0 X Y ; set spreadcycle for X Y
M204 T4000 ; restore travel acceleration
M572 W0.06 ; set smooth time
M221 S95 ; set flow
G21 ; set units to millimeters
G90 ; use absolute coordinates
M83 ; use relative distances for extrusion
M572 S0.27
M107
;LAYER_CHANGE
;Z:0.2
;HEIGHT:0.2
G1 E-2.5 F4200
M73 P13 R1
M73 Q13 S1
```

```
G1 X93.413 Y98.514 E.00766
G1 F7124.667
G1 X93.204 Y98.797 E.01191
G1 F7096.05
G1 X92.791 Y99.949 E.04142
G1 F6671.905
G1 X91.995 Y99.058 E.04044
G1 X91.792 Y98.909 E.00852
M73 Q71 S0
G1 F6704.022
G1 X91.618 Y98.859 E.00613
G1 F6801.919
G1 X91.087 Y98.897 E.01802
G1 F7008.535
G1 X90.838 Y98.979 E.00887
G1 F7129.732
G1 X90.624 Y99.193 E.01024
G1 F7094.49
G1 X90.006 Y100.234 E.04098
G1 F6675.976
G1 X89.388 Y99.193 E.04098
G1 X89.174 Y98.979 E.01024
G1 F6720.421
G1 X88.982 Y98.906 E.00695
G1 F6837.908
G1 X88.482 Y98.854 E.01702
G1 F7038.58
G1 X88.215 Y98.911 E.00924
G1 F7148.086
G1 X88.016 Y99.058 E.00837
G1 F7083.788
G1 X87.22 Y99.949 E.04044
G1 F6667.655
```

What does a (visual) .gcode editor do?

1. Parses .gcode into intermediate structs
2. Renders visual representation
3. Processes transformations
4. Re-exports valid .gcode

Features

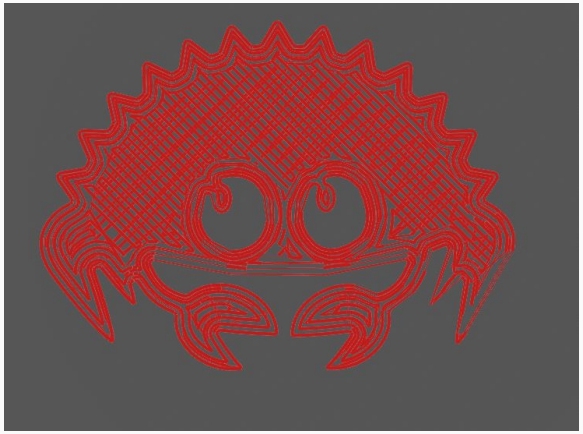
- Translate/Scale/Rotate commands
- Delete and merge extrusions
- Subdivide extrusion moves

Selected Example

->

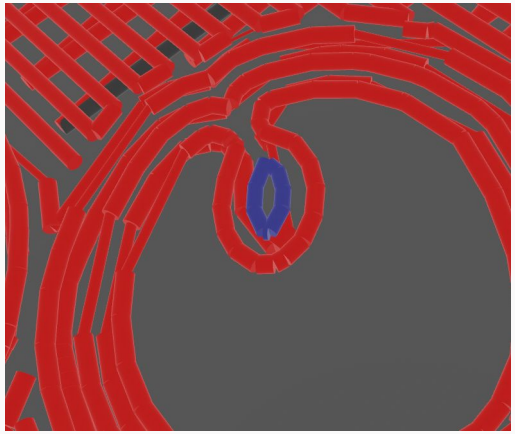
Selected Example

load



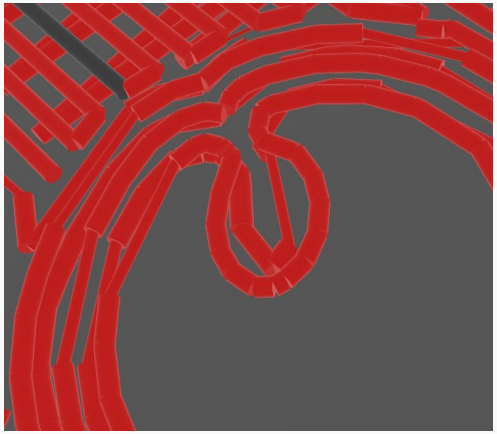
->

select



->

delete



save + print ->

Why Rust was/is nice?

- Rust is easy and rust works
- Multiplatform out of the box
- Convenient toolchain for setup and install
- Zero cost abstractions are ergonomic

Code Snippets

Parsed .gcode struct

```rust
#[derive(Clone, Debug, PartialEq)]
6 implementations
pub struct Parsed {
    pub lines: Vec<Id>, // keep track of line order
    pub vertices: HashMap<Id, Vertex>,
    pub instructions: HashMap<Id, Instruction>,
    pub shapes: Vec<Shape>,
    pub rel_xyz: bool,
    pub rel_e: bool,
    id_counter: Id,
}
```

G1 Commands Types

```rust
#[derive(Copy, Clone, Debug, PartialEq, Eq)]
5 implementations
pub enum Label {
    Uninitialized,
    Home,
    PrePrintMove,
    TravelMove,
    PlanarExtrustion,
    NonPlanarExtrusion,
    LiftZ,
    LowerZ,
    MysteryMove,
    Retraction,
    DeRetraction,
    Wipe,
    FeedrateChangeOnly,
}
```

Iterators are fun

```rust
pub fn parse_file(path: &str) -> Result<Vec<String>, Box<dyn std::error::Error>> {
    let out: Vec<String> = String::from_utf8(vec: std::fs::read(path)?)? String
        .lines() Lines
        .filter_map(|s: &str| {
        // ignore ';' comments
            let s: &str = s.split(';').next().unwrap();
            if s.is_empty() {
                None
            } else {
                Some(s.to_string())
            }
        }) impl Iterator<Item = String>
        .collect();
    Ok(out)
```

```rust
fn get_selections(mut s_query: Query<(&PickSelection, &Tag)>) -> HashSet<Id> {
    s_query Query<(&PickSelection, &Tag), …>
        .iter_mut() QueryIter<(&PickSelection, …), …>
        .filter_map(|(s: &PickSelection, t: &Tag)| if !s.is_selected { None } else { Some(t.id) })
        .collect()
}
```

Give it a try or chat about printing!

repo: https://github.com/mj10021/g-wiz

email: jamesthespeedy@gmail.com