Art with fantasy consoles, Part I: basic drawing

Last issue we talked a little bit about the role of *noise* in generative art. In this and the next several issues we'll be doing a series on generative art in the free fantasy computer TIC-80 (the name being a play off both the old computer from the 1980s, the VIC-20, and "TIny Computer"). "Fantasy computer" is a provocative but not a *clear* name. If you're familiar with emulators for old consoles or computers from the 80s like the Commodore 64 or the Apple][e, TIC-80 is like an emulator for a computer *that never existed*.

It has a limited number of colors it can use at a time, a limited number of sprites it can hold in memory, and restrictions on how the "hardware" of the system works.

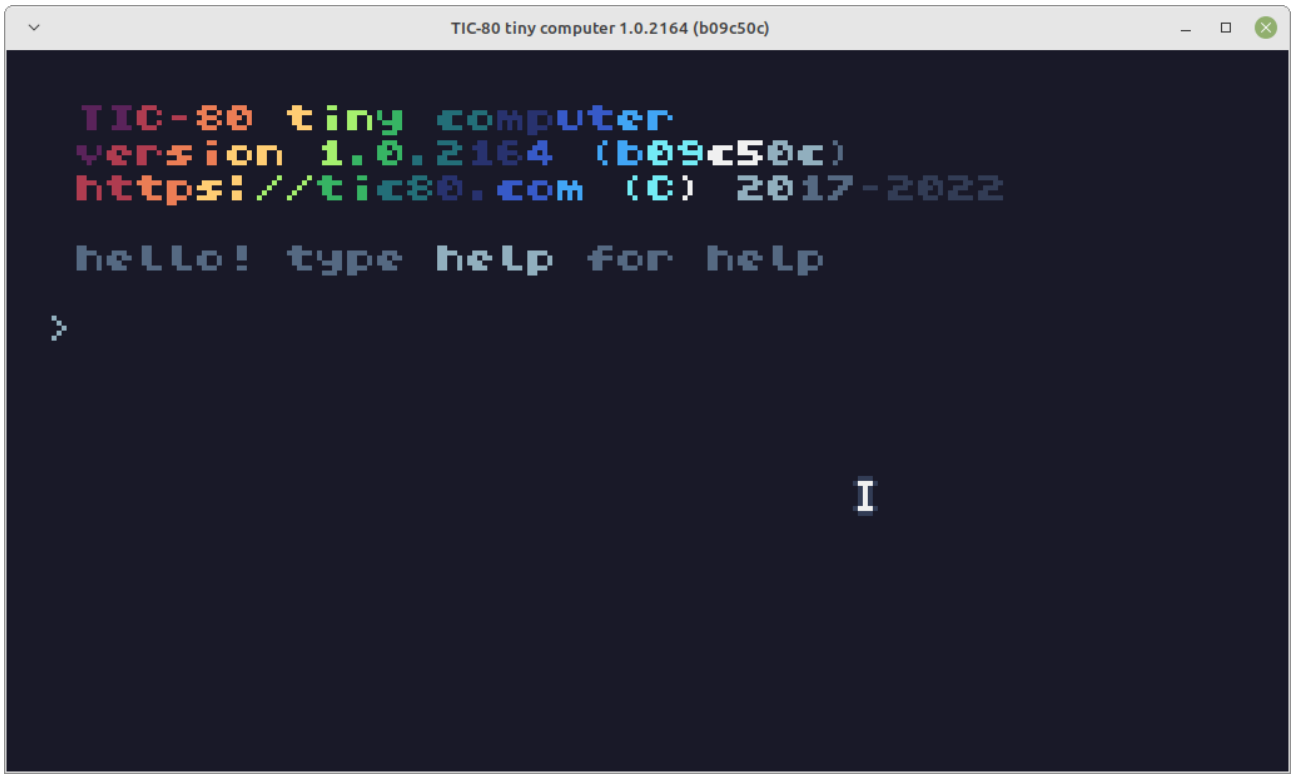
So why use something like this?

1. Because it's far simpler to program than something like Unity yet far more powerful than learning programs like Scratch
2. Because it can run on a huge variety of hardware, including single board computers like a raspberry pi, old phones, or even handheld consoles like the Nintendo 3DS

This time, we'll be covering how to load up and use TIC-80, and how to draw with it. After that, we'll be covering sprites and animation, then how to do some simple motion with physics.

The first thing to do is download the program from https://tic80.com/create. If you're reading this on a Chromebook you can use the in-browser version also on this same webpage.

When you load up TIC-80 for the first time, you'll see something like this! What you're looking at is



The next thing you'll want to do is something that will save you some time in the long-run: type

`menu` and then navigate to `OPTIONS` and hit enter. Next, hit the option that says `DEV MODE` so that it's on. Now hit `ESC` twice to exit back to the command line.

Now, if you type `demo` and hit enter at this command line you'll install a bunch of programs you can try out, like one of my favorites is one you can try by typing

```
load fire.tic
run
```

and you'll see a tiny little art demo of a fire

Hit `ESC` to exit the program and go to the code that made it! You're looking at the programming language *Lua*. You may have heard of Lua: it's the language used for making Roblox programs!

In order to start a new program, the very first thing you need to do is type `new` on the command line. Then you can hit the `ESC` key and enter the code editor.

You'll see something that looks like

Let's start by first highlighting all the text and ruthlessly erasing it.

Now, let's start again very simply and type the following on the screen:

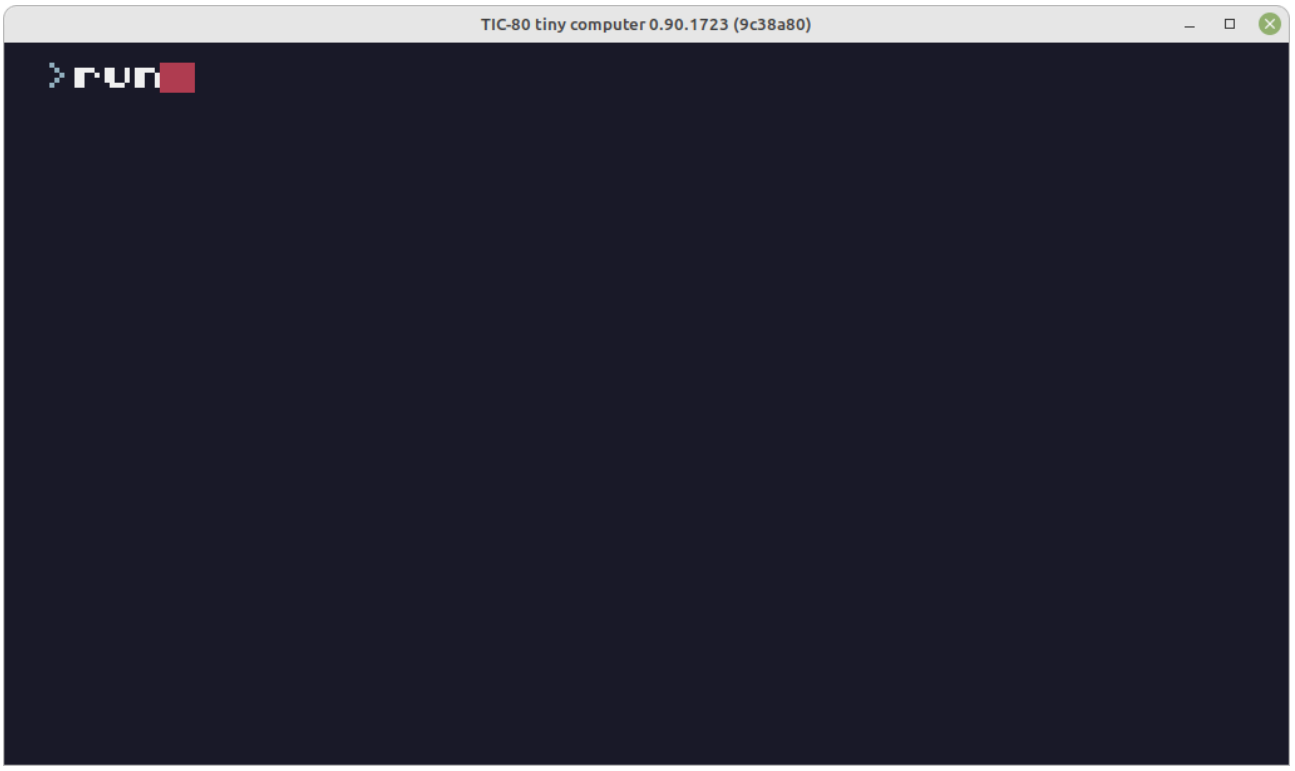
```
function TIC()
end
```

This is the minimum amount of code needed to run a program in TIC-80. Every TIC-80 program has to have a `TIC()` function which runs at 60 frames per second and contains all the code for drawing, moving things on the screen, everything needed to make your art project or game come to life.

Since there's nothing *in* this function, let's see what happens if we run it.

Hit `ESC` and then type `run` at the command line.

You'll see something like



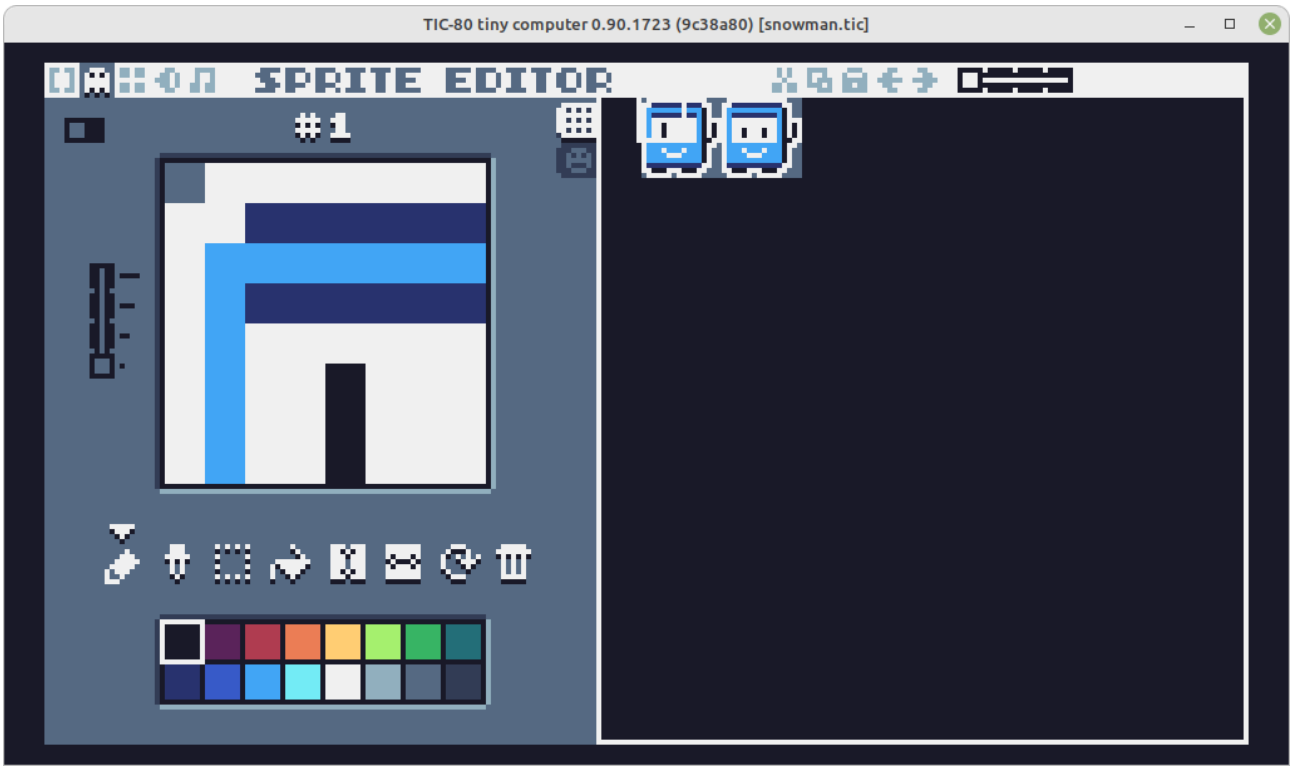
This is truly an *empty* program, there isn't even the code required to give you a black screen.

Now hit `ESC` to exit the program, such as it is.

So the next thing to do is just go ahead and add the following line so your code looks like this

```
function TIC()
   cls(0)
end
```

Now if you run it again you'll see a black screen. What have we done? We've called this function `cls` and given it a single argument, the number 0, which we write as `cls(0)`. Why 0? If you click on the tab with the little pacman ghost next to the `[]` you'll see



where you can both draw sprites and pick colors. As you can see, complete black is the first color!

From here, let's practice drawing on the screen with basic shapes!

Make your code look like

```
function TIC()
 cls(13)

 circ(120,100,22,12)
 circb(120,100,22,0)

 circ(120,80,18,12)
 circb(120,80,18,0)

 circ(120,65,13,12)
 circb(120,65,13,0)

 circ(115,60,2,0)
 circ(125,60,2,0)

 tri(118,63,122,63,120,70,3)

 line(105,80,85,70,0)
 line(135,80,155,70,0)
end
```

If you run the program now, you'll see a little snowman looking back at you!

Let's explain a couple of things before we end the article: coordinates in TIC-80 are like in more professional games engines and *not* like you might see in Scratch. In most art and game design programs `(0,0)` is the *upper-left* corner of the screen and in TIC-80 `(239,135)` is the lower-right corner of the screen.

The functions used above are pretty simple!

- `circ` takes the x and y position of the center, the radius, and the color
- `circb` is the same but only draws the outline
- `tri` takes the three points of the corners of the triangle and the color
- `line` takes the start and end of the line and the color

between these and `rect`, which takes the x and y of the corner and the width and height of the rectangle, you can draw almost anything.

And this is where we'll pick up next time as we start making procedurally generated art!

Happy hacking!