Tech (/category/tech.html) · 09 October 2020 · 22 min read

Down the ergonomic keyboard rabbit hole

425

For best results, listen to me read you this post instead! Just play the audio and the page will automatically scroll in sync.

This post is all about my recent obsession with ergonomic keyboards. There are technical elements to it, but mostly I'm just gonna tell you a story about my slow descent into madness. Hang on, and enjoy the ride.

Prologue

The story starts a few years ago, when I decided to buy myself a Koolertron Programmable Split Mechanical Keyboard, All 89 Keys Programmable Ergonomic Keypad, 8 Macro Keys… (https://www.amazon.co.uk/Koolertron-Programmable-Mechanical-Keyboard-Ergonomic/dp/B07PZT3Z25). If it wasn't obvious from the name, it was *very* Chinese, *very* badly documented, and *very* badly made. That didn't matter though, I was smitten.

Steven Waterman (/swaterman)

I'm a developer at Scott Logic, showing off the cool and tricky things I learn so that you don't have to go through as much pain as me!
Find me on **GitHub (https://github.com/stevenwaterman)** and **Twitter (https://twitter.com/SteWaterman)**.

Categories

Latest Articles (/index.html)

Resources (/category/resources.html)

Cloud (/category

I bought the split keyboard because I was obsessed with the idea of typing one-handed. If I could program the keyboard to use chorded keys (sending one character by pressing two buttons at once), then I could have the whole alphabet available on half a keyboard. At that point, I could use a keyboard and a mouse, or even a pen, at the same time. Think of the efficiency!

The keyboard came, and I started learning to use it. It was a bit of a pain, as the manual was mostly Chinese and the software only worked on Windows. I use the term *worked* loosely. It would break at random times, and had various undocumented limitations that I had to figure out through trial and error.

That wouldn't stop me though, I was smitten! I dutifully programmed the keyboard and started learning to type again. Over the course of a couple of days, I slowly worked my way up to 20 words per minute.

At some point, I realised it wasn't going to be as quick and easy as I hoped. I *also* realised that it was exam season, that I was a university student, and that I really had better things to do. Had my excitement just been subconscious procrastination?

Years later, the keyboard sat on a shelf, unused ever since. The answer seemed to be a yes.

/cloud.html)

Tech (/category/tech.html)

UX Design (/category/ux.html)

Delivery (/category/delivery.html)

Testing (/category/test.html)

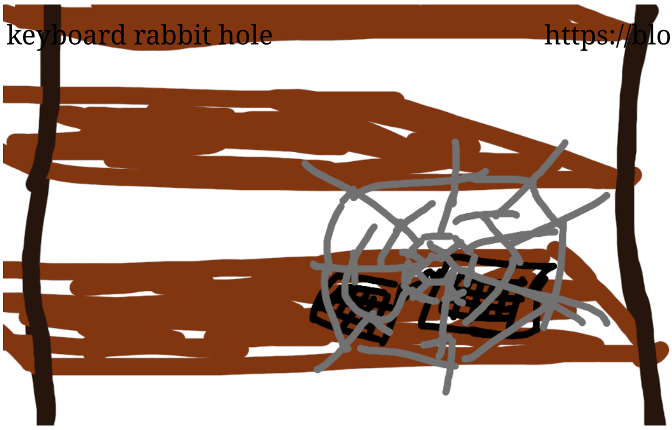
Data Engineering (/category/data-engineering.html)

Careers (/category/careers.html)

Diversity (/category/diversity.html)

Videos (/category/videos.html)

Back to all posts (/index.html)



Accidental Rediscovery

We jump forwards to March 2020, and I was working from home. Fed up with swapping my keyboard between my PC and company laptop, I went looking for a second one to use. As luck would have it, the only spare keyboard was the one I had abandoned years earlier.

There was no way I was using the chord-based setup though, so I went in search of a way to reset it. A few hours and one dodgy Google Drive link later, I had a copy of the software. I traipsed the keyboard upstairs to my wife's (Windows-based) PC, and reconfigured it to something resembling a standard QWERTY keyboard.

At this point, I couldn't care less about the ergonomics of my new keyboard. I just wanted it to work.

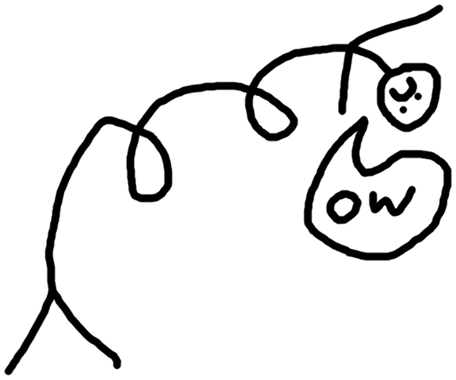
At first, it was a struggle. Learning any new keyboard is always a bit jarring, but the split layout was especially confusing. I had a bad habit of using my left hand for most keys, and I kept trying to press `Y` and `H` on the left keyboard. Both of those were on the right side, and what I was actually pressing was a macro key that tabbed out of the current window. That was a little annoying!

I tried to retrain my brain, but it was far too much effort. I gave up and just reprogrammed the keyboard to have `Y` and `H` on the left instead. Feeling like a genius, I started using the new layout. It felt much more natural, but I was still having trouble.

What was wrong? Turns out I press `Y` and `H` with my right hand too. Good job Steve, good typing skills you've got there. Up the stairs we went, reprogramming it to have two of each.

Finally, I wasn't making constant mistakes. The unfamiliar setup still caused problems though, and my typing speed definitely got worse. What's more, I ended each day with a sore back. Aren't split keyboards meant to be more ergonomic?

It turned out that the issue was just how unfamiliar it was. Everything was a bit more manual, and I was having to type more consciously. I couldn't rely on muscle memory, meaning I actually had to *think* about how to type. In the end, that just meant I was much more tense, hunching over and looking at the keys.

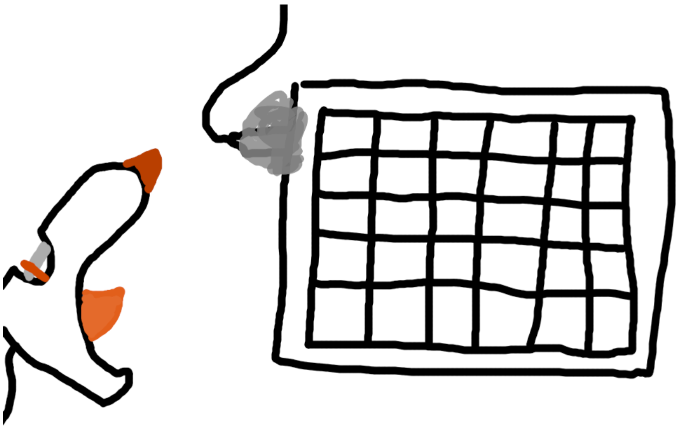


Over the next few months, I started to notice that I preferred typing on the split keyboard. The back pain was gone, and it just started to feel comfy. Going back to my old keyboard felt alien. Its keys felt too close together, and the bend in my wrists felt tense and uncomfortable. When the work project ended and I no longer needed two keyboards, I decided to keep using the split keyboard on my PC.

Getting Serious

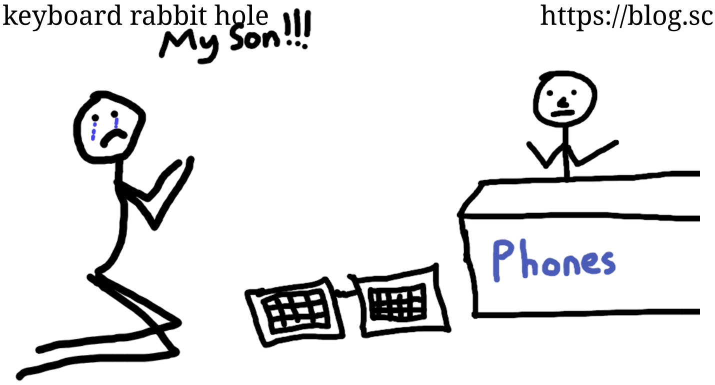
Things weren't all sunshine and rainbows though. The (lack of) build quality was starting to catch up with me. The micro-usb ports were coming loose, and it started to disconnect randomly. It was fine at first, and I just accepted that the cable needed to be in a specific position.

Over time, that position became more and more specific. At its worst, I once spent half an hour trying to get a connection. Exasperated, I held the cable in exactly the right place and pulled out the hot glue gun. Now the cable wasn't going anywhere, and I could use my keyboard in peace.



A couple of weeks later, the problem returned. The cable was now so fiddly that the slight flex in the hot glue was enough for it to disconnect. Drastic times call for drastic measures, so I started taking the other side of the keyboard apart. There were micro-usb ports on both sides, so I could just remove the port and solder a wire onto the pads directly.

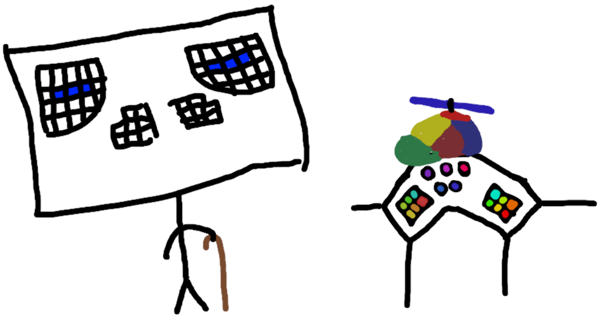
With the keyboard in pieces and a micro-usb port on my dining room table, I learned that micro-usb is really small. Probably could've guessed, given the name… I briefly considered taking my keyboard to a phone repair shop, but decided that I had too much self respect.



I tried just going back to my old keyboard, but it was too late. I was a split keyboard weirdo now, and I had to embrace it.

Suffering through the dodgy connection and the fact that I couldn't reprogram the right-hand side of my keyboard, I started looking for a replacement. I *could* just buy another of the same model, but that felt short-sighted. This one had broken - who's to say the next one wouldn't too?

I stumbled across the Kinesis Advantage 2 (https://kinesis-ergo.com/shop/advantage2/) but couldn't find it in stock anywhere. What's more, it just felt a bit old-fashioned. I'd seen the crazy stuff people had been doing with mechanical keyboards, surely things have moved on from that?



My suspicions were quickly confirmed when I found the Ergodox EZ (https://ergodox-ez.com/). It had a pretty compelling feature list:

- Completely configurable - both hardware and software
- Ortholinear layout, meaning the keys were arranged in neat columns without being offset left-right

- Adjustable legs to tilt it in any direction
- Completely open-source, even down to the PCBs
- Great build quality, reliable warranty, and an active DIY community with spare parts widely available

I was sold. It took me a while to stomach the price at nearly £300, but I worked my way through the configurator.

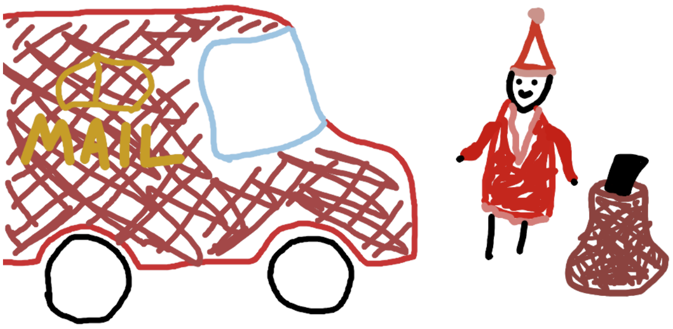
The black case (https://cdn.shopify.com/s/files/1/1152/3264/products/ab39140e3565f145aa488ee14e3bbd18_fdf03bc1-a62b-45f8-a10d-f62cc3e82ceb.jpg?v=1570475753) was an easy choice because I'm not flamboyant enough to pull off an all-white keyboard (https://cdn.shopify.com/s/files/1/1152/3264/products/e1fbe3297b62fb3a394029694f7d3d90_95ac7869-4eec-4249-a80b-b49b1e14e50e.jpg?v=1570475753). Blank keycaps weren't my first choice, but are nicer to use as they're sculpted for a specific location on the keyboard. Meanwhile, the printed keycaps are designed to be swapped around based on your configured layout. The adjustable legs seemed like a must, so I shelled out the extra for those. Trying to save at least a little money, I declined all of the RGB lights - both under the keys and the under-keyboard *mood lighting*.

When it came to keyswitches, I'm experienced enough to know I wanted Cherry MX Browns (https://www.cherrymx.de/en/mx-original/mx-brown.html), which are quiet but tactile. I bought an extra 20 Kailh Gold (https://ergodox-ez.com/pages/keyswitches) switches which are super clicky, to use for the macro buttons. Yes, that's right. The key switches are hot-swappable.

Then, I began long wait for delivery.

Learning to type again, again

I watched the tracker as my keyboard made its way over from Taiwan. With each city it passed throughout the next few weeks, I got more and more excited. Then it arrived. I immediately unboxed it and tried it out.



It was fine.

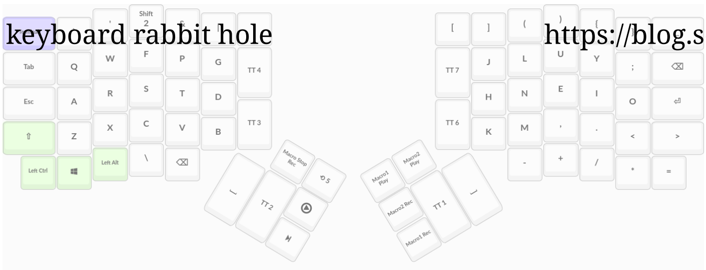
Like, it was a nice keyboard and everything. It was just different. I felt the same tension with this keyboard as I did when learning to use the first split keyboard. It was unfamiliar, felt strange, and made typing into a conscious activity.

I tried to get used to it, learning the default QWERTY layout, but it wasn't easy. At some point, I snapped.

> Screw this, if I have to relearn how to type I may as well learn a better layout.

I had a look around at the available options, especially Dvorak (https://en.wikipedia.org/wiki/Dvorak_keyboard_layout) and Colemak (https://en.wikipedia.org/wiki/Colemak). In the end, I chose Colemak, and honestly can't remember why. It probably just looked less scary, and the data supports that it's the more efficient option (https://colemak.com/Fast).

I loaded up Oryx (configure.ergodox-ez.com/), the Ergodox configuration tool, and started reprogramming my keyboard. It was basically just Colemak, but with the number keys in a separate layer to emphasise the symbols. My first attempt a custom keyboard layout looked like this (https://configure.ergodox-ez.com/ergodox-ez/layouts/Ov6O3/OYOD3/0):

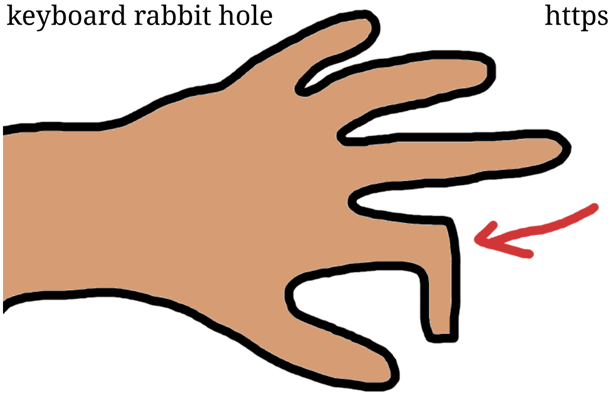


Armed with my new keyboard layout, I started my typing practice again. It was painful, but we worked through it. 4 hours of hard practice later, I had typed 228 snippets and the website decided I was a competent typist.

At least, I was competent with the letters `ENITRL`. Yes, I spent 4 hours typing words with those 6 letters before it decided I could handle `S`. The website was wrong - I couldn't handle `S`. My typing speed dropped from 45wpm to 20wpm.

Nevertheless, I stuck with it and slowly learned how to type `S`. After another 50 snippets, I was ready to move on. While we're here, just take a mental note that each of these snippets took between 30 seconds and two minutes to type. Every time I say 50 snippets, just mentally replace that with 50 minutes.

When I got to the letter `D` after almost 400 snippets total, I found it a lot harder than I expected. In Colemak, `D` requires you to move your left index finger out of its column and press the key to the right. It was the first time I had to do that, and It didn't come naturally.

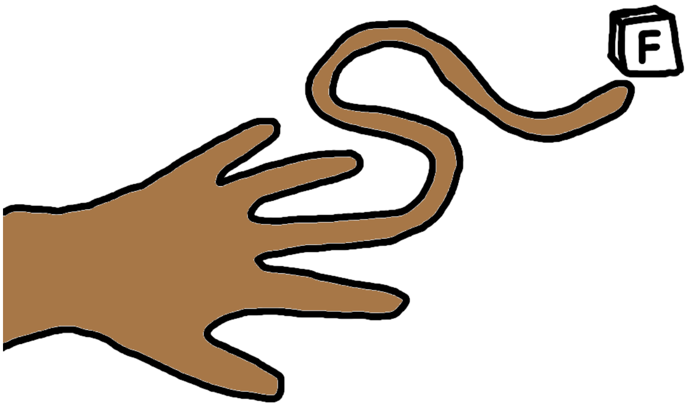


Later that day, I learned about the Workman keyboard layout (https://workmanlayout.org/). It's based around avoiding that horizontal movement I found so difficult. I briefly thought about swapping to the Workman layout completely, but wasn't willing to go through another 7 hours re-learning the home row.

Instead, I took the concepts they discussed and applied them to my keyboard layout, only moving the keys I hadn't learnt yet (and `D`). If moving my fingers sideways is hard, why don't we just… not? Could we build a keyboard layout around 8 columns, rather than the usual 10 columns?

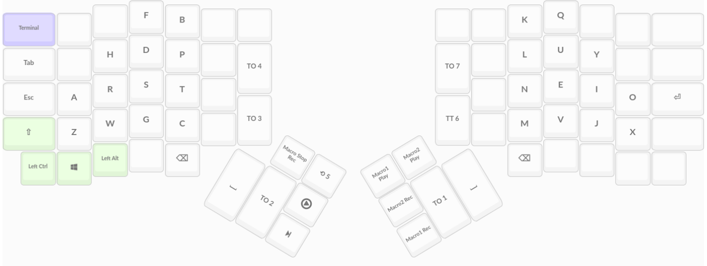
The obvious answer is no. There are only 3 rows of buttons, meaning the layout would only have space for 24 letters total. A strange thought hit me. I started playing around with my keyboard, trying to rank how easy it was to press each key.

Moving my index fingers towards the middle, like I was with `D`, is hard. Moving my little finger up, like you would to press Q on a standard layout, is also really hard. Pressing the number row with my index and middle fingers is easy. Why not just put letters there?



That seemed like a pretty crazy idea, but I'm not one to be held back by tradition or common practice. I loaded up Oryx and gave it a go. At this point I felt like a complete lunatic, but jumped right back into the typing practice. Re-learning the letter `D` was hard, but once I was there, it felt much easier.

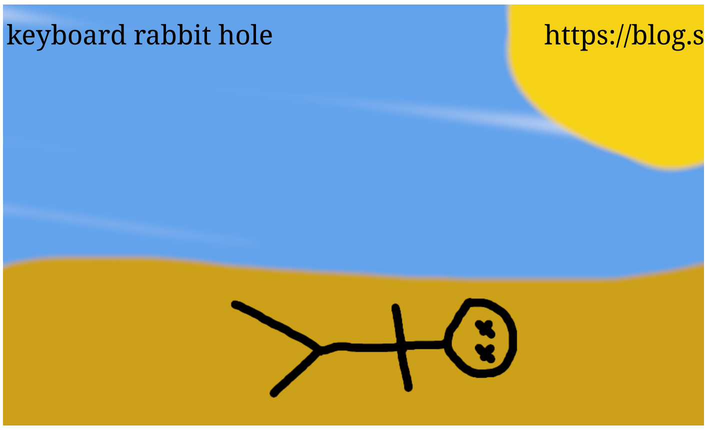
I continued through the training, dreading the day I'd unlock one of the number-row letters and have to give up on the layout. Roughly 600 snippets in, I unlocked `B` which was where `4` was on a normal layout. It was… fine? To everyone's surprise, I had no real problem with it. Moving my finger 2 keys up was much easier than 1 key sideways. I had settled on this (https://configure.ergodox-ez.com/ergodox-ez/layouts/5BvvY/LnPYL/0):



I kept going with that layout and had no real issues, unlocking the final 9 keys in the next 150 snippets. I had finally unlocked every letter. I was free!

I'm not going to say I was a fast typist at that point. I had hovered around 20wpm ever since unlocking the first letter. Every time I got a little faster, they threw more letters in and slowed me down again.

All in all, I spent almost 12 hours doing typing practice over a period of 7 days. It had consumed all my free time, and at least some of my life. I just wanted to be done.



I started using the Ergodox for programming, despite how slow I was. It was frustrating, I felt tense, and my back hurt. At times, it was wonderful. As I started to relax, I could see why people liked this keyboard so much.

Three months later, I was mostly happy with my keyboard layout (https://configure.ergodox-ez.com/ergodox-ez/layouts/5BvvY/x6PQL/0). All the symbols had their place, and I'd figured out how to make the best use of the layers. I was up to around 70wpm, which was enough for me to feel comfortable using it for almost everything. My current layout isn't much different, and we'll walk through that in depth at the end.

The biggest problem I had wasn't with the keyboard at all. It was the mouse. Every time I wanted to use the mouse, I had to take my hand off the keyboard and move it over to the mouse pad. Alternatively, I *could* use my keyboard as the mouse, but that wasn't precise enough. I could turn down the sensitivity, but then it wasn't fast enough to navigate across my monitors.

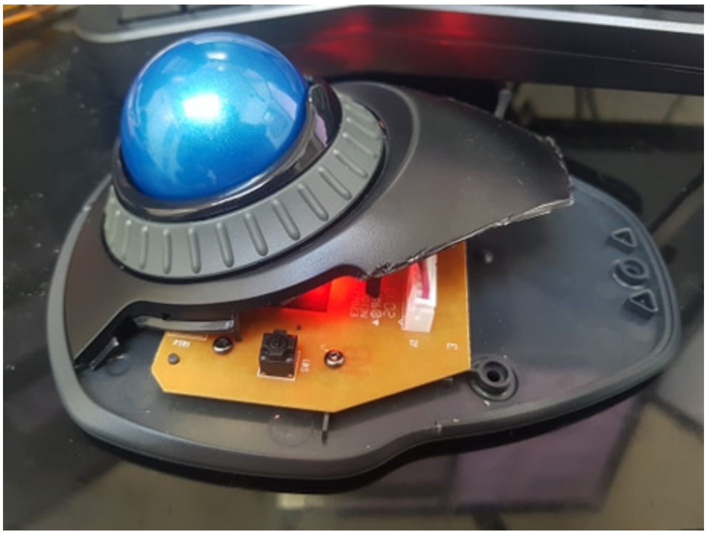
Instead, I picked the secret third option. I bought a trackball mouse (https://www.amazon.co.uk/gp/product/B002Q42S4E/) and used it as a wrist rest.

Going too far

It's become a theme at this point that I impulse-buy computer peripherals without thinking of the issues. Let's walk through some of the issues I had with my purchase.

The first issue was the size of the mouse. It was just too wide, and it wouldn't sit close enough to the keyboard. That meant I was using my wrist rather than the palm of my hand. It was uncomfortable, awkward, and hard to use. The solution? Get a screwdriver and take the cover off!

Great! Now it fits under the keyboard and is nicer to use. New issue - the trackball is all wobbly without the cover to reinforce it. The solution? Get a hacksaw, cut the cover up, screw it back into place.



Fantastic. Now it's more stable and I don't worry about breaking it when resting my hand there. One issue though - it'd really be a lot nicer if I could place the mouse sideways. That way, it would fit on my desk better and have the same sideways tilts the keyboard. I did that, but it's quite hard to use a sideways mouse. No worries, we're on Linux! Open the terminal, do a quick little `xinput --set-prop "Primax Kensington Eagle Trackball" 745 270` and boom! It's configured with a 270-degree rotation.

Surely there are no more issues now? Well, it turns out that resting your hand on a mouse while typing isn't very helpful. The mouse keeps moving and changing which window is focussed. No worries, we can fix that! Find an unused key combination, like `Alt + Win + Escape`, and add that to your keyboard layout. Listen for it in i3 (my window manager) and toggle the mouse off/on whenever it's pressed. Now I can turn the mouse on whenever I need it and turn it off afterwards.

But I don't like having to toggle the mouse on and off. I already have a layer dedicated to navigation - can't the mouse just turn on whenever we're in that layer? I searched every bit of Oryx, and there was no option for anything like that. Defeated, I just accepted there was no way to do it. I would just have to live with it like this forever. I accepted that, and I could move on.

The next day, I got fed up with accepting that, and reverted to grief stage 3 (https://en.wikipedia.org/wiki/Five_stages_of_grief): bargaining. There had to be another way. I took the step that I'd avoided for months and clicked `Download Source`. You see, the Ergodox is based on QMK (https://qmk.fm/), an open-source keyboard firmware. Oryx is really just a wrapper around that, generating and compiling layouts for you.

By avoiding the big shiny `Download Layout` button and clicking the scary one under that instead, you get the generated source code.

A short while later, I had a working development environment and flashed my keyboard manually for the first time. I had unlocked the true power of my keyboard! Armed with my *limited* knowledge of C, I stumbled through the process of customising my keyboard layout. I added a tiny bit of code into the `layer_state_set_user` function, which is called each time the layer changes. One *if* statement later, my keyboard would press the (unused) calculator button when it entered layer 3, and the home button when it left. In the i3 config, I listened for those keyboard events and turned the trackball on/off.

It was beautiful. No more toggle button - just go into the nav layer and the mouse starts working like magic. I was finally happy. Then I tried to use a web browser.

I was typing in a URL, and the browser autocompleted the end of it. I wanted to add something to the end of the URL, so went into the nav layer and tried to use the arrow keys. However, as soon as I went into the layer, the autocompleted text disappeared.

It turns out this is a limitation with X11 (https://en.wikipedia.org/wiki/X_Window_System), the windowing system at the core of most Linux desktops. Any time you listen for a key and react to it, X11 sends a `FocusOut` event, then runs the code, then sends a `FocusIn` event. That happens no matter what key you use (https://stackoverflow.com/questions/26489193/how-to-prevent-window-from-loosing-focus-when-receiving-a-grabbed-key-from-x11) - even media keys. In web browsers, that breaks autocomplete. In a lot of other software, it causes unfocus and focus animations to play. I rely a lot on autocomplete, so that wasn't gonna fly.

I did what anyone else would do in my position. I wrote a driver.

Well, kinda. One of the features of QMK is the raw HID mode (https://docs.qmk.fm/#/feature_rawhid). This lets you perform two-way communication between the keyboard and host, bypassing the drivers completely and only working with the raw bytes sent to and from the device.

I set up the Ergodox to broadcast on two interfaces, and listened to the second one with Python's hidapi (https://pypi.org/project/hidapi/). On the keyboard side, I replaced my previous layer-based key presses with this:

```
uint8_t msg[RAW_EPSIZE];
sprintf((char *)msg, "LAYER:%u", layer);
raw_hid_send(msg, RAW_EPSIZE);
```

Then, on the host side, I wrote:

```
while True:
  msg = h.read(8).decode()

  if msg.startswith("LAYER:"):
    layer = int(msg[6])
    if layer == 3:
      os.system("xinput --set-prop \"Primax
    else:
      os.system("xinput --set-prop \"Primax
```

There are no issues with windows unfocussing now! In fact, there's not even any key commands being sent. I made the script run on startup, and the mouse works reliably!

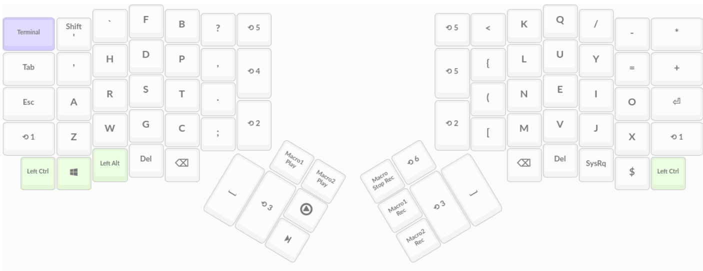
That's the end of the whole trackball saga. There are a few other things I want to talk about, but let's walk through my final setup and discuss those as they come up.

My Current Setup

That brings us to today, and my current keyboard layout (https://configure.ergodox-ez.com/ergodox-ez/layouts/5BvvY/ZKx5z/3). This section is an in-depth discussion of how it all works. If you're just here for the story, this button will skip to the conclusion:

Skip to Conclusion

By default, it looks like this:



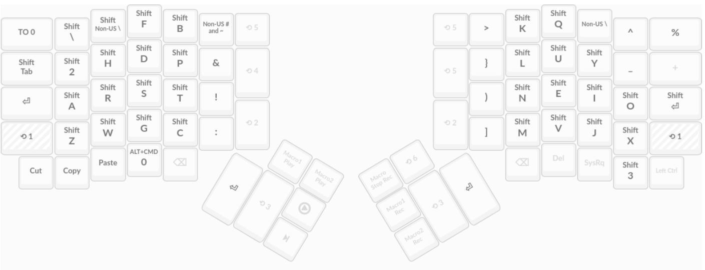
Other than the weird 8×4 lettering system, there are a few other things to note. Escape has been moved to the Caps Lock key, because who actually uses that? The Escape button is now used for a macro that opens a new terminal. My most common punctuation is in the middle columns and above my little fingers, where most other layouts have letters. I have two space bars, two backspaces, and two delete keys, all designed to be used with my thumbs. Also in the thumb clusters, I've got dynamic macro recording and the media keys.

The media keys are interesting because they didn't work at first. Something about my setup in Linux wasn't forwarding them properly to Spotify. Thankfully, I could configure i3 to work around the issue by manually listening for the keys and notifying Spotify:

```
bindsym XF86AudioPlay exec "dbus-send --pri
bindsym XF86AudioStop exec "dbus-send --pri
bindsym XF86AudioPrev exec "dbus-send --pri
bindsym XF86AudioNext exec "dbus-send --pri
```

Layer 1: Shift

By pressing either Shift key we move into layer 1, which contains the alternate variant of each key. In most cases, that's just the shifted version of the key from the default layer.



That's not always the case though. Sometimes, I decided to ignore common sense and bind a completely different key. For example, the default layer contains `[`. On a normal keyboard, pressing that while holding shift gives you `{`. I don't like that, so instead layer 0 contains all the open brackets and you get the closed version by holding shift. In other words, `Shift + [` gives you `]`.

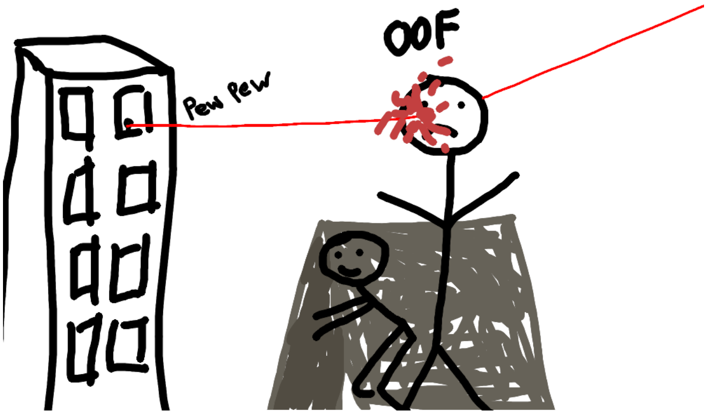
That goes for a lot of other keys too. Forward slash is on the default layer, hold shift and you get a backslash. The dollar (`$`) sign becomes a pound sign (`£`), and apostrophes (`'`) become speech marks (`"`). Rather than working from a normal keyboard, I worked from first principles, thinking about which keys are related semantically.

There's other interesting things on the shift layer, like the left side's bottom row. There, we have `Cut`, `Copy`, `Paste`, and `Programs`. The first three are just macros for `Ctrl + X/C/V`, while the last one launches i3's program menu, which is similar to the Windows key on that OS. You press it, type the name of a program, press enter, and it starts up.

My only problem with the Cut/Copy/Paste buttons is that the required shortcut changes constantly in Linux. In an IDE, `Ctrl + C` will copy, but in a terminal it interrupts the running process. A future improvement would be to send those events via raw HID, then handle the clipboard with shell commands from Python, allowing it to

Finally, you can see that the Caps Lock key is now bound to Enter. That allows me to press enter with just my left hand, for when I'm using my normal mouse and don't have my right hand on the keyboard.

The only problem with this setup was that the Shift key didn't work as a Shift key. Most of the time, programs don't listen for Shift directly. Games and image editors were the exception. I couldn't crouch or scroll horizontally!



For a while, I used a workaround where holding the Caps Lock key would act as Shift. That really wasn't ideal though. Once I had solved the trackball issues, I knew it was time to fix the Shift key once and for all. I had a workaround, but wasn't there a better way to do it?

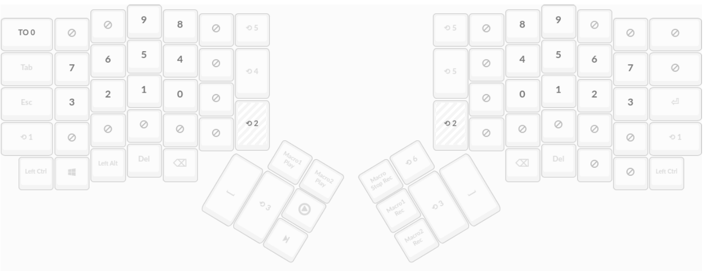
In the end, I did get it to work. First, I set up the keyboard so it sent a `Press Shift` event whenever we entered layer 1, and `Release Shift` whenever we left. That meant our Shift key now worked as normal, but it also meant that everything on the shift layer was pressed with Shift. For a lot of keys, like the alphabet, that wasn't an issue since the keyboard needed to hold Shift. For some keys like `]`, pressing shift would break it. In this case, it turns `]` into `}`.

For each of those keys that would break, I defined a macro. The macro sent `Release Shift`, then pressed the key, then sent `Press Shift`. As hacky as that is, I've actually had no issues with it so far.

That's it for layer 1, so let's move on to layer 2.

Layer 2: Numbers

This is the numbers layer, given that we booted them off the top row to make room for more letters.

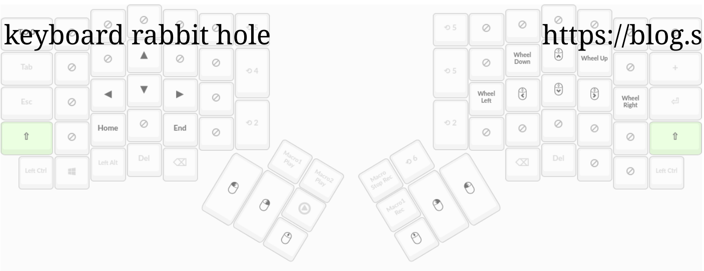


Rather than go for a simple number pad layout, I used a layout similar to the letters on the first layer. The home row contains the numbers `0123`, which are the most common (https://en.wikipedia.org/wiki/Benford%27s_law). The next row contains `4567`, and the top row contains `89` to be used by the index and middle fingers. The numbers appear on both sides of the keyboard, always laid out with the smallest ones in the middle. That means that the most common numbers are used by the strongest fingers.

At one point, I had all the number-related symbols on this layer too, like `+`. However, I wasn't really using them enough to commit it to muscle-memory. Instead, I just go back to the default layer and enter the symbols that way. It doesn't really seem to slow me down, but I'm still working on being fast with the number keys. Later down the line it might become an issue, and we can revisit it then.

Layer 3: Navigation

Layer 3 is entirely built around navigation - whether that means moving the text cursor or the mouse. You get there by pressing the button next to space in the thumb cluster - giving it pride of place.



First of all, being in layer 3 enables the trackball. I have the sensitivity set fairly low, which lets me move the mouse roughly half a screen using just the palm of my hand. If I want to make bigger mouse movements, I use the mouse keys on the right half of the layer. The sensitivity here is quite high, meaning it's useless for precise mouse movements but perfect for getting close enough. Once I'm close, I can use the trackball to position the mouse precisely.

The right half of the keyboard also contains keys for scrolling, both vertically and horizontally. Both thumb clusters contain left click, right click, and middle click.

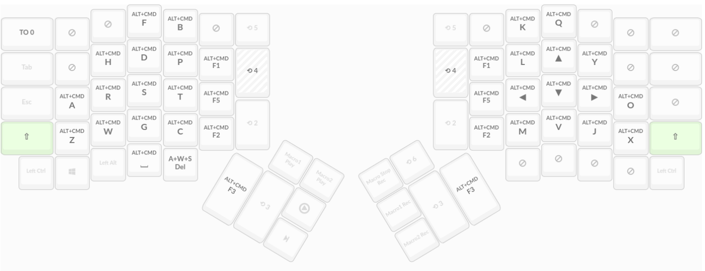
The left side of the layer is dedicated to navigating text. It may seem weird that I have to use both hands to move the mouse, and that I didn't just put the mouse keys on the left half. The problem with that is the trackball.

Whenever we're in this layer, the trackball is enabled. With the mouse keys on the left half, I have to move my hand to press the keys which causes the mouse to move around. The way it's set up now, I can press the mouse keys without moving my left hand. I can focus completely on the trackball, rather than trying to press the keys without moving too much.

The keys on the left side of the layout are fairly straightforward. We have the arrow keys, Home, and End. Home is placed under the left arrow, and End is under the right arrow. At some point I'll probably add PageDown under the down arrow, but I've recently moved the Home key from there and I keep pressing it by mistake.

Layer 4: Spaces

This layer is dedicated to i3, and moving windows around. It's bound to the button above layer 3, in the middle of the centermost columns.

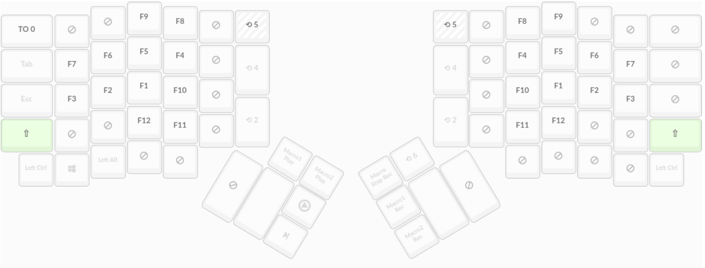


Pressing any letter brings that workspace into focus, and holding shift moves the currently selected window to that workspace. The right side of the keyboard has some arrow keys, which are used for moving focus or windows relatively. For example, the left arrow selects the window to the left of the currently selected window. If holding shift, it moves the window to the left instead.

The middle columns are used to swap between different window layouts. The space key toggles full-screen on the currently selected window, while the backspace key closes it.

Layer 5: Function

This layer contains the function keys, using the same layout as the number layer. It's a bit hard to reach the layer 5 key, which sits at the top of the center column, but it's rarely used.



I'm still getting used to this, and honestly I usually revert to my old, standard-layout QWERTY keyboard to press the F keys. We'll get there!

Other Functionality

All layer buttons are momentary - we stay in the layer as long as the key is held. The exception to that is if you press the Escape key (as in, the one in the top left) while holding the layer button. This locks you in the current layer. To get out, press the

There's also the debug layer, which is accessed through the right thumb cluster. By pressing that key and the mirrored one on the left side, you put the keyboard into its bootloader, allowing new firmware to be flashed. That's a godsend - because I can never find a paper clip to press the reset button!

You can see the current revision of my keyboard firmware on GitHub (https://github.com/stevenwaterman/keyboard).

Conclusion

Over the past few months, I have well and truly gone down the ergonomic keyboard rabbit hole. There wasn't one big event that thrust me into this subculture. It's been the culmination of me asking "couldn't this be better" for 3 months straight.

I don't expect you to immediately run out and spend money on a split keyboard, or change your layout. However, next time you're doing something and get annoyed at your keyboard, ask yourself if it could be better. If you experience wrist pain when using your keyboard, ask yourself if it could be better.

I've gone too far with my setup, but it's been fun. And yes, I am getting worse at QWERTY.

This has been a bit of a weird post, but I hope you enjoyed the ride! Please do get in touch on Twitter (https://twitter.com/SteWaterman) with any feedback, or to give suggestions about my setup!

Read more

View more by Steven Waterman (/swaterman)

View all Tech articles (/category/tech.html)