the codex

Life with Linux — A Zine

Typeset in LaTeX

Kenneth John Odle

2021
August

Contents

Impressum

FYI, this is made in LaTeXusing the report document class. It then gets exported to a letterhalf (5.5 in x 8.5 in) pdf, which then gets made into a booklet using Boomaga (`https://www.boomaga.org/`).

I'm pushing this to my own git server as I write this. You can find it here: `https://git.kjodle.net/kjodle/the-codex`. New issues will be pushed after they are complete.

The image on the front cover is courtesy JericoDelayah from the Wiki-Media Commons. The image is over here: `https://commons.wikimedia.org/wiki/File:4_RETAT_04_Linus_Torvalds.jpg`. You can also find a link to the Creative Commons CC BY-SA 3.0 license there, as well.

The image on the back cover is one that I highly agree with. We built it, it's ours, and we shouldn't be charged for using it.

You can just skip over all the diversions in you want. It's just how my mind works. (And yes, there will be politics in this. *You have been warned.*) Also, I use a lot of em-dashes and parentheses because that is also how my mind works. It's just one big long stream of consciousness up in here most days.

Chapter 1

The Early Salad Days

Boring, early life stuff when my world smelled like sweat and disinfectant and warm bologna. Feel free to skip this. I wish I could.

1.1 Calculators

Before computers were in my life, there were calculators.

These days, every kid has to have an expensive graphing calculator for middle school math. Specifically, it has to be a Texas Instruments graphing calculator, because the examples in the textbook are all described in terms of a Texas Instruments calculator.

I mean, *sure* you can get your kid that Casio, which has all the same features and all the same buttons and is an order of magnitude cheaper, but you spent all that money on an expensive pre-school, and all that money on expensive tutors, and do you really (he asked snottily) want to risk little Jimmy's chances of getting into Harvard because you were temporarily too cheap to buy the right calculator?

Oh my, a diversion already.

(A little off track here, but this begs two questions: 1) Why is it always a TI calculator that's required, and 2) Are we teaching kids to learn math or to learn how to operate a calculator? The answer to the first question is that Texas Instruments and the Major Textbook Pub-

lishers™ have colluded to produce expensive books that need to be replaced every two to three years [thereby costing the school district money] and that require expensive calculators[1] [thereby costing you as a parent money]. It's a racket, but that's capitalism for you. The answer to the second question is that we are teaching kids how to use calculators. Teaching them how to do actual math would require thought on both the parts of the teachers and the parts of the students, not to mention on the parts of parents and especially administrators, who would also be required to grow a spine—and learn how to use it. Again, education in the United States has become a racket, but that's capitalism for you.)

(Well, you can probably tell what my thoughts are on the dominant economic system on planet Earth. There *will* be more of that. If you're okay with that, I'm okay with that, too. If you're not okay with it and you want your money back, it's too late—I've already spent it.)

I have noticed that even little kids are required to bring little kid calculators to school with them in most of the local school districts. As I write this, the school supply buying season is coming to an end, but for the past six weeks every store was filled with school supply lists and yeah, you have to have a calculator to get into the second grade.

Ironically, the earliest calculators I can remember seeing (not getting my hands on, because they didn't belong to me) were Texas Instruments. I don't remember a lot about them, but an uncle had given a pair to two of my cousins. They took a *ton* of batteries, had red LEDs for outputs (meaning they glowed in the dark—you could use them in the dark if you memorized the keypad), and they were designed for students because they had a go-back-through-all-your-steps-function-to-see-where-you-done-screwed-up-boy function, which would be a useful feature on modern calculators to learn math, but again, we're not interested in kids actually learning how to think and do something as radical as math.

The other early calculator I remember was a Casio calculator and it was on a watch. A kid I knew for a short time had one, and even let me wear it for a while. (I wish I could remember his name, because this was

[1] A few years ago, I bought a scientific calculator at the **dollar store** and tested it against my very expensive TI-92. It was just as accurate as the more expensive calculator, and cheaper by two orders of magnitude. Did I mention that this is a racket?

a tremendous kindness on his part.) I swore that when I grew up, I would own one of these watches.

Well, I grew up and I didn't buy one of them, even though they are still available. I could just never justify spending the money on what is—let's face it, just a bit of full-frontal nerdity—when there were bills to pay. Nope, just could never bring myself to do it.

It's just me now, and my expenses are numerous but small, and a couple of years ago my local all-in-one-store had all their watches on sale for 40% off, including the name brand watches. I checked—it was in stock. At $25 bucks it was a lot, but on sale it was only $15. I could do this! So I picked it up and looked at it lovingly, thinking about all the good times we would have together as we went forth and explored the world one simple calculation at a time.

But there was a problem, a rather large problem, actually. The print on those buttons is tiny. and my eyes are bad. I couldn't actually read any of buttons. I use reading glasses when I'm reading or working on the computer, but I don't need them out in the wild. I could wear the watch with me everywhere, but unless I was at my desk, I wouldn't be able to actually use it.

Back on the shelf it went.

At this point, my only hope is that maybe my eyes will get so bad that I'll need bifocals all day, every day. When that happens, will this watch be on sale for so little money ever again? I highly doubt it.

1.2 Speaking of Watches, Timex Used to make Home Computers

My earliest memory of a computer in somebody's home is of being in an aunt's apartment, and she had a Timex Sinclair hooked up to her television.

I don't remember much about it, actually.[2] I do remember that I was not allowed to touch it.

[2]This aunt bought things not because she found them useful, but because other people didn't have them and she wanted to always have a status symbol to point to. I don't remember her actually doing anything *useful* with this computer.

This is where memory gets wonky, because I remember seeing this when I was about ten years old. But according to Wikipedia, the Timex Sinclair was released in 1982, when I would have been 14 years old. So it's entirely possible that my memory is losing track of *when* things happened, or it's possible that this aunt had some other home computer that for whatever reason my brain thinks is a Timex Sinclair. Who knows? I certainly don't, and I'll probably never find out for sure.

Chapter 2

What's to Like About Linux

I could go on and on here, but I'll try to keep it short. I can always come back to this.

2.1 Control... and an Opportunity

What I like—not love (love is about aesthetics for me when it comes to computers)—is that I'm in control.

Partly, that's the nature of open-source computing. If you want to know how something works, you can look at the source code. If you don't understand the source code, you can research how the source code works. You can ask questions. (Thank you, StackExchange!) You can do some more research and then learn how to ask *better* questions. There is always something to learn, and once you've learned everything there is to learn about a particular piece of software [1] you can fork it and start contributing to the project yourself.

Wondering how something in Windows works? So is everybody else. There is nothing more frustrating than googling a problem in Windoze, getting hundreds or thousands of results, and every result is just somebody else asking the same question.

And yeah, you can write code and create applications for Windows, and you can solve a lot of problems that way, but you can never make

[1] Which is never really true. What I really mean is that when you've learned everything *you* want to know about it.

Windows itself better. It is what it is, and if you don't like it, the feature that bugs you might be made better in the next release, or it might be made worse. It's a crap shoot, really.

For what it's worth, Mac OS X, even though it is based on Unix/Linux (I forget which—I dropped out of the Mac world at OS X version 4), is the same way. There *might* be an answer, there *might* be a solution, but you just *might* be on your own there, buddy.

But what I really, really like about Linux?

The command line.

I'll probably write about this some more later, but my experience with computers goes back way before Macintosh made the mouse popular (and alas, necessary). You turned on the computer, and there was just this dark screen with a blinking cursor. If you wanted to make the thing do something, you had to *know* something. With a GUI, you can guess. You can guess a lot, actually, and just poke around all you want because most GUIs come with an undo feature.

There is no "undo" on the command line.

I need to get that on a t-shirt.

Why? Because the command line is like real life. There is no undo button in real life. GUIs have made us lazy—lazy at thinking, lazy at figuring things out. Just do it, if you don't like it, just Ctrl-Z. Just throw that document away and leave it in the recycle bin. If you decide you want it later, you can just drag it on out of there.

2.2 The Unix Philosophy

The Unix Philosophy was originated by Ken Thompson (one of the creators of Unix, upon which Linux is based) and basically says that each program should do one thing and do it well. (There is more to it than this; if you are interested, you can always google it.[2]

This runs counter to physical life, where everything has to be a Swiss army watch. Watch any ad for a new kitchen gadget and this device does

[2]Searching for something on the internet is *always* an option these days, and so many people seem to be unable to do just that. Honestly, this is the kind of stuff that gets my underwear in a twist.

Question: "Where can I find *X*?" Answer: The same place I would find it: At the other end of a google search.

Better question: "Which is the **best** source for *X*? Ah, *now* we have the basis for a discussion.

everything except walk the dog and take out the trash. If it *actually* did all those things and did them well, I would be happy to own one and more than happy to pay a couple of hundred dollars for it.

Unfortunately, it seems that it's impossible to build a device that will do a large number of things really, really well. I like to cook and so my parents inevitably give me a cooking-related gift every Christmas and every birthday. One year, I received a mandoline-type device—if you've spent any time watching the Home Shopping Network[3] I'm sure you've seen them. It's basically a plastic tray with different cutting inserts, a handle to hang on to the food item, and a box that snaps on to the bottom to hold whatever you are slicing.

I absolutely *love* this thing for slicing potatoes, and since I spend each autumn and winter making scalloped potatoes or au gratin potatoes, it sees a lot of use during those months when the days are short. It does a fantastic job slicing potatoes into a uniform thickness and does it far more quickly than I can do it with a knife.

It also includes inserts to make waffle slices (if you rotate the potato 90 degrees on each pass, you're supposed to be able to make waffle fries), inserts for dicing onions, and so forth. But here's the thing: as great as it is at slicing potatoes (and also carrots, which have the same general hardness as potatoes), it does a terrible job at slicing those things. Basically, the thin and the thick slicing inserts work well for potatoes and carrots, and all the other inserts don't work at all for them, and any other vegetable just doesn't get cut or gets crushed because you have to hold onto it so hard.

I don't know how much my parents spent on this thing, but if it's anything north of $20, that's a lot of money for something I can already do fairly easily (and actually enjoy doing) with a sharp knife. Don't get me wrong—I love the thing (even though it's a bit of a pain to clean), but if we had spent at least as much time and money engineering the thing as we did marketing it, we might have concluded that it would probably be better to just encourage people to buy decent knives and then teach them how to sharpen them and use them properly.

(Also, I'm not picking on the Home Shopping Network[4] because everybody does this. Monty Python, all those years ago, even had a skit about this, which you can find if you google "simpsons individual stringettes".

[3]Which is now called "HSN". Apparently, we are two busy to pronounce those two extra syllables. Modern life may be difficult, but I don't think the energy I save from not pronouncing those two syllables are going to give me enough energy to overcome it.

[4]Okay, "HSN".

Of course Monty Python was making fun of this tendency and 50 years later we just accept it as a part of life.

Chapter 3

Coda

3.1 What I Learned About LaTeX While Creating This Issue

I'm still a relative newbie to LaTeX, so there's always something to learn. Here's a running list of what I've learned so far:

1. You might think you want the **book** document class, but you probably will find the **report** class just as handy.
2. You want links[1]? Use the **hyperref** package.
3. The **kpfonts** package has beautiful fonts.
4. Footnotes are easy! (Seriously, footnotes in LaTeXhave got to be the easiest footnotes I've ever managed.)
5. Use the **fancyhdr** package to get more granular control over your headers and footers.
6. You can use the **geometry** package to make a document have a paper size of half letter.
7. You can make your top margin larger by using `\addtolength {\topmargin}{0.5in}` but there is not a similar parameter for the

[1] Yeah, I know these are irrelevant in a paper document.

bottom margin. Instead, you need to make the text box shorter by using `\addtolength{\textheight}{-1in}`.

8. Want to show inline code without executing it? Use `verb` following by two pipes. Place your code between the pipes. (I had to use two of those in #7, because that code just went right off the edge of the page when I only used one.)

9. Need a little space between elements? Just insert `\,` (that is, a backslash followed by a comma).

Like I said, I'm still a newb and I may be completely wrong or off base on some of these things, in which case, I'll make a note of that in a future issue [2]

If you are interested, there is a link in the Impressum to the git repo for this publication where you can check out the source code.

[2] Always assuming that there *will* be another issue.