Why using the terminal still makes sense nowadays

Marcin Pastudzki

May 27, 2019

Every now and then people see me doing things (be it at work or at home) in terminal and they ask why on Earth do you do it to yourself? You've got plenty of fine graphical interfaces for everything, where things are easier and faster to do. Why do you memorize arcane shell commands instead of just clicking your way through simple everyday tasks?

And quite honestly, my personal reason for this is: out of habit. In the past I worked a lot on remote machines through SSH, where you do not really have a choice, terminal is the only option. In addition to that I did a lot of shell scripting, which got me well acquainted with ways of the terminal. And I got used to it so much back then, that even nowadays, when the kind of work I do for living and my daily routines changed a lot, it's still much more natural to me to continue using the terminal.

Frankly, learning some GUI at this point does not make a lot of sense to me, because GUIs tend to change ways of displaying things over time, which makes you constanly relearn what you already know. It's not that way with the terminal, which is virtually the same tool as it was back in the early days of computing. Of course, terminal emulators and shells[1] evolve over time too, but those changes are much less substantial, beacuse commands are still carried by the text you type and arrangement of buttons, inputs and graphical menus does not matter.

However, if you aksed me whether or not it's desireable for a software developer (or any person doing a lot of work on a computer for that matter) to put an effort into purpouseful learning to work in the console just for its own sake, without additional stimuli that I had, I would still say: yes, it is. Actually I believe that the terminal is particularly well suited tool for programmers and other people who write some kind of computer-interpreted code at work. However, after some getting used to it it's also more convenient for everyone.

First of all I'd like to sort out a common misconception about the terminal. People tend to think it's more difficult compared to any graphical tool. That is simply not true. The difference lies not in the difficulty, but in so called learning curve. Learning curve is a function relating time and effort put into learning a new ability to its usefulness or power. In other words, terminal *is not* harder, it only *appears* harder to an untrained observer.

With most GUI tools the learning curve is flat[2]: it takes little effort to learn the tool, but its

[1] I'm going to use the terms *terminal*, *console* and *shell* interchangably, beacuse from the point of view of this article they're basically the same. It's worth noting, however, that technically speaking they're very different things. A *terminal* (or a console) is a mechanical device from a time when a personal computer was a luxury even undreamt of, consisting of a keyboard and a monitor, which served as an access point to a computer mainframe. These are nowadays replaced by software *terminal emulators*, which replicate their behaviour on modern computers. That software is what most people mean when they say "*terminal*" nowadays. A *shell* on the other hand is another program running inside the *terminal*, providing a language with which user can issue commands to the system.

[2] This does not really concern complex and specialised graphical software like image processors, text processors, spreadsheets and so on. It's mostly about simple programs that help you manage your system like file managers, package mangagers, configuration wizards and the like. However, you might be surprised by how complex tasks can be handled efficiently by exquisite command-line programs like LaTeX. If there is a solid terminal-based alternative to your favourite software, you should definitely at least give it a try.

capabilities are also quite limitted. User's power does not grow with time they spend using them. Unfortunately, power only comes with hard work. It's quite the opposite with the terminal: it's harder at first, but user's power grows immensely over time. Actually the more complex task you need to accomplish, the easier it is in terminal compared to any GUI tool. Suppose you'd like to find all files bigger than a certain size in a particular directory (and its subdirectories) and rename them *en masse* according to some rule or move them to a common directory. You can do it with a single command in any shell. And with graphical file managers? It's cumbersome to say the least and usually takes time proportional to the number of files.

The power of terminal should also be apparent to anyone who tried to look for help with their computer on the Internet. For instance if you asked a question on any Linux forum about some configuration issue you have with your system, you can almost certainly expect to receive an answer in the form of a shell command you should perform or a change to a configuration file you should modify. The reason is not that forums users are wicked computer gurus trying to scare ordinary people off from computers; it's just a lot easier to copy-paste a shell command than to explain how to find a specific button in a graphical interface. Especially if you consider that the person who asked might use different graphical tool than the responder. GUIs tend to differ a lot visually but a shell is a shell, no matter what terminal emulator you use. In other words the shell language might serve not only to communicate with the computer, but also with other people who happen to understand it.

Some people say that error messages of shell programs are often cryptic and more difficult to understand. The truth is that console programs tend to be simpler and focused on a single, simple task. If that fails, the program tells you that directly and without pleasantries. You can think of a shell as a vast workshop. All the simple tools you might possibly need are there somewhere, but they're not really organized anyhow. On the the other hand, graphical utilities are like little bundles of specialized tools. Because they are bundled together, they can work together without user knowing it. With the shell the user selects his tools by himself and chains them himself at will to best meet his purpose. With this additional responsibility comes also additional power, because now you've got litterally infinite number of compound tools at your disposal, not only what bundle's author thought might be useful. This is also why error messages are more scary: because they now come from individual tools which know nothing about each other, and not from the bundle as a whole. There's no "friendly" interface between you and the tools anymore. You're exposed to them directly. And unless you wish to remain blissfully ignorant about how your system works, this is a good thing, because it allows you to learn a lot through experimentation.

Once you get sufficiently familiar with the shell, it also becomes more convenient to use. Notice that to use a terminal you only need two hands placed on a keyboard and you scarcely move them at all while typing. On the other hand with most graphical tools you also need a mouse and you find yourself constantly moving your hand from the keyboard to the mouse and back again. Once you think about it it's not very convenient.

Understanding the shell at an expert level enables the user to take one more step and automate any long and cumbersome operation so that it can be perfomed by the computer on its own with little to no supervision. This is called shell scripting, an ability of crucial importance to systen admininistrators and other people responsible for smooth running of computer systems. With scripting the shell becomes a real programming language and using it, you'll quickly find out, that certain things are a lot simpler in it than in most general purpose programming languages. In fact shell scripting can be a great introduction to more serious computer programming.

Your system comes with a lot of simple little tools which are only really accessible through terminal. Without it, you're really condemned to using bundles prepared by other people, who don't really know or understand your specific needs. Terminal is certainly a good first step to really taking your computer into your own hands.