Emacs

/run/media/lijunjie/资料/Learning Video/writing-running-fixing-code/01_writing-code/04_tools/09_emacs_instructions.html

When you are programming, your *editor* —the program you use to actually write your code—is key to your productivity. One of the most important aspects of the editor in your productivity is *not* getting in the way of the flow of ideas from your brain through your keyboard, and into your source code. While this point may seem odd or unimportant to novice programmers, as you become more experienced, you will start finding yourself "In The Zone" as you program. When you are "In The Zone," your mental focus is fully dedicated to your programming, and you have your mind full of your plans and designs for the program. Here, disrupting your focus at all can take you out of The Zone, destroying your mental focus and losing your train of thought.

Programmer's editors (such as Emacs and Vim) are designed so that you can do everything you need without taking your hands off the keyboard. The ability to do everything on the keyboard is crucial to staying in The Zone—pausing to grab the mouse and stumble through menus looking for what you need can be enough to disrupt your focus. Instead, programmers using tools like Emacs and Vim have the keyboard shortcuts for anything they do regularly in *muscle memory* —they can perform the activities without consciously thinking about how, leaving their full focus to their programming tasks.

Besides enabling you to stay In The Zone by not having the distraction of the mouse, programming editors provide a variety of other productivity enhancing features. One such feature is *syntax highlighting* —coloring the words that you type based on how they fit into the syntax of the language (*e.g,* keywords, type names, the name of a function being declared, string literals, integer constants, etc...). Syntax highlighting helps make it easier for you to read the code—for example, you can easily distinguish what text is inside a string literal. It also helps you avoid, identify and correct mistakes—such as accidentally using a keyword (that you are not familiar with) as a variable name, mismatching (or improperly escaping) quotation marks in string literals, or misspelling keywords.

Another feature is automatic indentation according to the structure of the language. Programmers generally consider indenting code according its nesting level (how many {} it is inside of) to be a key element of proper style and formatting. Not only does this help with readability, but having the editor automatically indent based on the actual nesting level (as opposed to what the programmer *thinks* it is) can help identify and correct errors earlier.

A third important feature of programming editors is their ability to interact with the other tools used by the programmer—the debugger, the compilation/build tools, and revision control systems. For example, in Emacs, you can compile from within the editor, and if the compilation results to errors, jump directly to each of them within the editor.

Likewise, when using the debugger *gdb* , Emacs understands how to interact with *gdb* and will display the current execution point in your code, and let you send commands to gdb about a particular line of code you are viewing.

We will also note that programming editors typically have a lot of flexibility in what they are useful for. The authors of this book use Emacs for pretty much *everything* that they write, including this book itself! As with programming, staying In The Zone is crucial to productivity when writing text.

While both Emacs and Vim are appropriate editors for professional programmers, we will focus on Emacs here. We note that if you use (or want to learn) Vim, that is great, but the instructors use and know Emacs. We recommend having a basic familiarity with both editors (in case you ever need to use a system where one is installed, but not the other). However, we note that it virtually impossible to be an expert in both of them. Remember that you want to train your muscle memory so that you can perform editor commands by instinct, without thinking about them. Most people cannot do this for two completely different sets of editor commands.

****IMPORTANT****:** When using programs like emacs, be sure that if you suspend a process with ctrl-z that you **always** return to the session you started with "fg". Do NOT repeatedly start emacs sessions. This will cause your PPE to get shutdown, locking you out of the assignments until the next day.