# Colemak modifications for Atom

I’m becoming more and more interested on efficiency and how to learn and create content faster and faster, so I started to get involved in the use of programs and computer techniques for this. I found a high pay off on identifying better way of doing things in a programmatically way (initially in excel) and I got hooked to it. I think these are things that should be taught in school way earlier, instead of a lot of useless knowledge that I don’t remember now and I don’t think I will bother in learning again, but let’s not get off topic on the inefficacies of current learning systems.

My context puts me in laboring a lot with computers for research and content creation (I’m an aeronautical engineer), where I found the need to be able to work smarter not harder… at a desk. But how do you create content in an office job? You type it in a computer keyboard right? So I started to learn two things at once, one was about touch typing and the other one about efficient touch typing, that is where I came across with Colemak, which in short, is a keyboard layout designed to be more efficient than Qwerty (ever wondered why we use it?). Well really I started with Dvorak, but since I’m a Spanish native speaker, not being able to easily input latin characters was not practical for me, so I started to learn Colemak layout.  
I must say it was hard at first, even though many keys are not switched, I had to really commit to only use Colemak and start to create content way slower that before. I would sometimes just type the more common word patterns in a sheet, lines and lines of rubbish text to create muscle memory, until I got the grasp of it.

Moving on, I decided to learn how to program and got involved in the CLI world, where you can find some powerful tools for data analytics and with the capacity to create some specialized content in an efficient way. That is where I came across with vim (cloud 9 ide actually encourages you to use it). Vim is basically a command line editor that optimizes the navigation and text editing by using the keyboard keys for a lot of functions. Of course, it was created a long time ago, when Qwerty was basically the only input system, thus the functions are designed for a layout that fits weird in other layouts. At this point I got involved in the vimrc files modifications, which involved another bit of a learning curve, but soon I had a working vim environment fitted to Colemak.

Nevertheless, my job functions and projects where not involved too much in the command line environment, but still created content on text editors, where my favorite was Atom; mainly due to its UI, but then I decided to seize it’s ‘hackable’ capability to create an environment suited for what I wanted, efficient work.

# Setting up Atom for Colemak

The first interaction I had with vim for Colemak came from a GitHub repository I found belonging to Jooize, that based on the original vim navigation (h, j, k, l keys), but really I thought that that set could be improved, thus we can define a more efficient and intuitive set. Using the Colemak layout creator (Shai Coleman) makes sense, and even though he states that his main text editor is now Sublime Text (worth looking into).

The first step was basically installing the vim-mode-plus package and then modify the keymap file to override the key functions to fit Shai’s Colemak modifications for vim.

I opted for commenting out some of the functions included in the vim-mode-plus package but I think that is not necessary actually.