

Keys for movement in editors are most often adapted to fit with as efficient usage with QWERTY as possible, and so they will most certainly need to be remapped if you change key layout and want the *optimal* placement of everything, which you strive. E.g. in Vim, the HJKL buttons are used for a reason with QWERTY, and would most probably need to be remapped back to the same placement after the key map has been modified.

What I mean is that it won't help you much to track movement and editing keys and use it as a basis for a new layout, since they are **easily reconfigurable** (in any editor worth its salt, and since we are talking about a programmer's layout, we are most likely talking about Vim or Emacs), shouldn't interfere with the placement of the literal keys and have already been optimized (again: we are not talking about Notepad).

You are trying to solve a problem which is an **inefficient way to productivity**, especially **for a programmer**, ***imho***. There would be a much greater effect in simply learning more about the tools (once again: probably Vim/Emacs). You will find that less and less time is spent actually writing characters when programming, and more (but more efficient) time is spent on auto-completion, auto-tagging, auto-indenting, quick function definition look-ups, etc. The keys to do all this **are already adapted** to allow efficiency, and the big speed boost comes simply with familiarity. Thus I argue that **a different keyboard layout is right on comparably destructive for productivity**, since you already have many *years* of QWERTY exercise. If the same analytical training time was spent on QWERTY as people who switch layouts spend on Dvorak, they would also notice a speed boost. Speed comes with explicit training.

If you were a copywriter/translator/author/etc., someone who actually spends his time doing work with the *literal* meaning of the keys, then a different layout might be of help. For a programmer, the best tip is usually to at least get an English keyboard layout, since programming idioms have been shaped by these and their key placement (on my local key layout, `@$[]{}~` are all behind AltGr which is quite sub-optimal).

tldr: Dvorak/Colemak/[the next "best thing since sliced bread"] (arguably) solves a problem only for those who enter a lot of flowing text *in a specific language* (most often English). For programming, the needed keys have not been subject to the same restriction as literal language, and thus it has already been optimized for its purpose (which is *not* just "write as fast as you can"; it builds more on logical operations. See Vim). I believe that the time spent in learning alternate layouts and the confusion that most certainly occurs time and time again is definitely not worth the effort in most cases (not just your own confusion; others who sit down at the same terminal you last used will throw things at you), very much including the programmer's.

[share](#) [improve this answer](#)

answered Apr 11 '12 at 6:31



Daniel Andersson