Future keyboard standards: some requirements, ideas, challenges

Markus G. Kuhn

Computer Laboratory, University of Cambridge,
15 JJ Thomson Avenue, Cambridge CB3 0FD, United Kingdom
http://www.cl.cam.ac.uk/~mgk25/

Current European keyboard standards originated in an environment that no longer exists. Many original design choices are no longer justified:

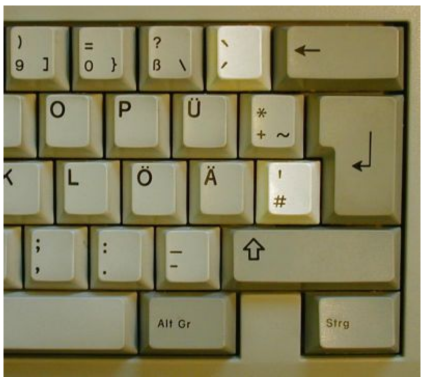
- Christopher Sholes' 140-year old QWERTY layout of the English alphabet was optimized to reduce jamming in the very first typewriters, intentionally slowing down users. Its basic layout received only trivial modifications in national adoptions:
  - German, Czech, Romanian: swapped Z↔Y
  - French, Belgian: swapped Q↔A, swapped W↔Z, moved M to the right of L
  - Some precomposed characters were added to the right of the basic alphabet
  - Small variations in location of punctuation characters
- The diagonal 15° arrangement of keys was entirely motivated by mechanical design considerations and the resulting lack of left-right symmetry cannot be justified anatomically.
- Many historic national keyboard standards emerged by accident (i.e., copied the layout of one prominent national mechanical typewriter manufacturer), rather than as the result of documented systematic research into user needs. In particular, within Europe, the existing diversity of keyboard layouts is much higher than what can be justified by character repertoire requirements, character frequency of different languages, or other actual cultural needs.

PC keyboards introduced many more features that quickly turned into historic ballast and today merely serve to puzzle users:

- The UK PC keyboard (derived from the 48-key version of BS 4822) features the non-ASCII characters BROKEN BAR (¦) and LOGICAL NOT (¬). They were solely motivated by their appearance in EBCDIC, and satisfy no contemporary user need whatsoever.
- The German PC keyboard (derived from DIN 2137-2:1988) features a key oddly labelled "Strg", which is now customarily known in German-speaking regions as the "String" key. Only a small fraction of users understands that this mystery label was originally meant to abbreviate the word "Steuerung", a rather unfortunate attempt to translate the English "Control", abbreviating it similar to "Ctrl". It is now a prominent failed attempt to localize a keyboard label and merely added a touch of confusion and out-of-this-world mystique to an everyday object.
- PC keyboards are about 25% wider than necessary, to accommodate a numeric block that hardly anyone (outside accounting offices) ever uses. It pushes the location of the mouse inconveniently far to the right. Numpad-free keyboards are perceived as a big improvement by most who try them.
- The numeric block's layout is oddly incompatible with the dominant standard on telephones and ATM PIN pads, which all have the "1" in the top-right position. For inexplicable reasons, keyboard standards placed the "1" into the second row from the bottom.
- PC keyboards feature mysterious function keys labelled "PrtScn", "SysRq", "Pause", "Break", "ScrLk" that have long ago lost their original function, are unsupported by current operating systems, and – if at all – then rarely for their original purpose. The same could be said – to a lesser degree – about the "Home" and "End" keys and the distinction between "Backspace" and "Delete"; while these are still commonly used, their meaning is far from clear from their standard label.

On the other hand, the by far most commonly used application-independent functions are entirely missing on standard keyboards. In particular “undo”, “redo”, “copy”, “cut”, “paste”, “find”, “find next”, “save”, “help”, “quit” still have to be accessed either via awkward vendor-dependent Ctrl keystrokes, or via slow and tedious mouse interaction. These very important functions have been present on proprietary keyboard layouts (e.g., Sun) for more than two decades, but were never offered by any of the formal national or international keyboard standards.

Some keyboard layouts, and in particular the behaviour of the keyboard drivers, actively *encourage* incorrect use of characters, such as the well-known apostrophe problem with German and Dutch keyboards. On these two keyboards, a non-spacing key for acute (´) and grave (`) accent is positioned more prominently than the (straight) ASCII apostrophe (see photo of German keyboard).



Dutch and German keyboards are very frequently used to input English text (e.g., manuals and correspondence for international customers and colleagues), where the apostrophe is common in front of the letter “s”. If the non-spacing key is mistakenly pressed to write an apostrophe before an “s”, the drivers of Microsoft Windows (and other platforms) will output the spacing accent character followed by the “s”, as in

- It´s awful.
- It`s horrible.

rather than the desired

- It's sort of ok.
- It’s wonderful.

A simple change in software could fix this. The only possible way to enter the (very rarely needed) spacing acute and grave accents on these keyboards should be by following the non-spacing accent key with the space key. This simple measure will discourage accidental use of spacing accents as apostrophes and quotation marks.

Another problem key is Caps Lock. We need it as little as we need a bold or italics key. Word processing functions to capitalize selected characters are far more convenient, and most of the time Caps Lock is pressed by accident. Users tend to be most grateful for the ability to just deactivate Caps Lock permanently or reassign some useful function to it.

Manufacturers of mechanical typewriters greatly simplified the repertoire of punctuation characters that are clearly distinguished in both handwriting and typography. In particular, the quotation marks “ ” and ‘ ’ were unified to symmetric variants " and ', and similarly the hyphen (-), minus (−), en dash (–) and em dash (—) were all unified into a single “hyphen-minus” character. While this was long ago fixed in coded character sets (PostScript, MacRoman, Windows CP 1252, Unicode/UCS), the keyboard standards still have not caught up. As a result, documents are now commonly written with a hyphen instead of a minus (+/- instead of correct +/−). Some manufacturers (e.g., Apple) who care particularly about the desktop-publishing market have fixed this in well-known proprietary extensions to keyboard standards. Others have added heuristics (e.g., Microsoft Word’s “smart quote” function) to individual applications that guess what the actually desired character might have been, automatically substitute it, and often get it badly wrong (e.g., Congress ‘99 instead of Congress ’99).

Additional characters are available via special “insert symbol” functions that present character repertoires for mouse selection. These are typically very primitive copies of Unicode glyph charts that provide no semantic information about characters and thereby actively encourage the incorrect selection of characters. A typical example is the misuse of the Spanish ordinal indicator instead of a degree signs (ºC instead of °C), which are hardly distinguishable in some fonts and grossly different in others. The Unicode code charts were never intended as a tool for interactive character selection. They hardly make any attempt to group similar characters together and have their own historic baggage. For example, the obvious semantic pair + and − are located over 8000 code positions apart, and the Unicode charts are full of legacy compatibility characters that are not meant to be used in freshly entered text. But users receive no guidance whatsoever on character selection by these input methods.

My suggestions and recommendations for next-generation keyboard standards:

A) Dramatically reduce the number of different European keyboard layouts
Recognize that most of the existing diversity is a historic accident and not justified by actual different cultural needs. Recognize that many keyboards are already used multilingually. Recognize that the same keyboard is already often used by people from many different countries (Internet cafés, hotel computers, academic exchanges, reuse of older computers in developing countries). Ideally there should only be a single keyboard layout for all European, American (and possibly even African) languages that use the Latin alphabet. Dedicated keys for precomposed characters are an anachronism from the days of mechanical typewriters. A main reason for precomposed keys was that mechanical non-spacing keys did not work well on both lowercase and capital letters. Likewise, the minor swapping of QWAZM across Europe is hardly justified by the far more substantial differences between character frequency in different languages, and neither do the minor changes in the position in punctuation characters reflect actual cultural needs. All the current QWERTY-derived keyboards are suboptimal anyway. Besides user inertia and training, there are few practical reasons for having more than one single Latin keyboard layout for all European languages. As that includes, apart from the EU, at least also North and South America, such standardization work should attempt to include the relevant transatlantic industries and standards bodies as well. With regard to keyboards, their linguistic needs are likely to be nearly identical.

B) Make aggressive use of shift chords, keystroke sequences and non-spacing keys
The existing standards use merely level 2 and 3 shift keys to extend the repertoire and use separate keys for a few precomposed accented characters. These restrictions were motivated by the constraints of mechanical typewriters, which cannot easily translate multiple keystrokes into arbitrary characters, and which cannot easily adjust the height of a non-spacing accent to the base character. Microprocessors have made all this long obsolete. We can now easily handle all diacritical marks conveniently via non-spacing keys, we can use a "Compose" key to signal that a character will be entered by a short mnemonic basic character sequence (e.g. Compose + c + o for ©), and we can address with n shift keys 2^n different levels of keyboard mappings, if the shift keys are only positioned such that they can conveniently be pressed as "cords". Use all of these. Keys for precomposed characters are a comparatively inefficient means to reduce keystrokes and improve performance. If that were the aim, having separate keys for the most common digrams (e.g. English: th, he, in, en, an, re, ...) or trigrams (e.g. English: the, ing, and, her, ere, ...) would save far more keystrokes, but would actually introduce *real* language dependency.

C) Define an integrated input method rather than just a keyboard layout
Even the smaller European subsets of Unicode contain far too many characters to ever fit onto a traditional keyboard map that could make everyone happy. In the future, the keyboard can only be one component in a collection of integrated standardized input methods that assist the user in quickly finding the correct rarely used character (e.g., for foreign names), and that help at the same time advanced users to quickly learn the keystroke sequences that allow an experienced user to enter every needed specialist character solely from the keyboard. Character selection code charts will no doubt play an important role. Pressing a "Compose" or "Symbol" key without following it within (say) 500 ms with a keystroke sequence that selects a character should bring up on the display a *standardized* selection of character-selection charts. These can then be

navigated with a *standardized* set of keystrokes, which the user can then learn and use in the future directly on the keyboard, without needing any more the guidance from selection charts. These charts must be carefully designed, to group characters in semantically meaningful ways, and should rather not be influenced too much by a character's (often pretty random) position in the Unicode table. About a dozen such selection charts may be necessary to cover all users needs (including non-Latin scripts and scientific notations).

D) Attack the problem at different levels simultaneously
There are conservative and radical ways in which a next-generation keyboard standard can be developed:

- **Modest maintenance:** Simply add the most urgently needed characters (curly quotes, minus, en dash, degree sign, micro sign, etc.) to level 3 of the traditional British BS4822 keyboard and its equivalent in other countries, without breaking backwards compatibility in any way.
- **Ambitious evolution:** Design a modern single standard Latin keyboard for all European Latin-script languages that is an integrated element of a comprehensive input-method concept (on-screen selection charts, compose keystroke tables, aggressive use of non-spacing keys, gentle modernization of function keys, etc.), but do retain a reasonable level backwards compatibility with historic practice and user customization (QWERTY, 15° arrangement, full ASCII coverage, access to all existing keys, etc.).
- **Radical revolution:** Throw away all historic ballast and go for an ergonomic orthogonal grid, Dvorak-style layout, complete redesign of function keys, etc.), something that no ISO member has ever tried.

I believe, there is great merit in attempting all three of these simultaneously, and then let the market decide, which solution people ultimately prefer. While the "modest maintenance" approach will be useful very quickly, it would be a shame to restrict the effort to that level. Having just three keyboard standards that clearly differ in how radically they break with the past makes far more sense than having 40 different national standards whose differences and number hardly anyone can justify other than by historic accident.

E) Keep the keyboard small
Today, standard keyboards are often used on tiny mobile devices (web phones, sub-notebooks, PDAs), where space is much more expensive than it was on 1920s typewriters or 1970s video terminals. Any additional keys should be justified by measured improvements in user performance rather than by being political status symbols or marketing fetishes. Avoid "German keys", "Irish keys", "Windows key", "Apple key", "Email key", "My Music key".

F) Do not be intimidated by the seeming diversity of existing national keyboard standards
Similar to the situation with domestic 230 V power plugs, the current diversity of national keyboard standards is more an artefact of historic accidents combined with the constraints of mechanical typewriters rather than real different functional and cultural needs. There is no doubt that *every* next generation keyboard input system should be well suited to enter *every* character and symbol commonly used in Europe. However, that does not mean that every character needs to have its own key or even its own visible label on a key, which was the only available solution on mechanical typewriters. I strongly believe, that today the "cultural" requirements of an accountant, a BlackBerry user, a C++ programmer, and a scientific author, who all work in the *same* language for the *same* company in the *same* building, are *far* more different than those of a German secondary-education pupil learning French, compared to a same-age French pupil learning German. Nevertheless, the former group is currently forced to use the same national keyboard standard, whereas the second group is forced to use minor and fairly arbitrary national permutations of the original Sholes layout.

G) Repertoire requirements and extensibility
Past CEN work on European Unicode subsets (MES) has shown how very difficult it is for a committee to agree on one single out of the 2^{3000} possible subsets of the about 3000 different Unicode characters that could potentially be relevant to users of modern European languages (i.e., what is left of Unicode if you exclude all

non-European scripts, historic scripts, and compatibility characters). As a result, any future keyboard standard will, just like Unicode, unavoidably become a moving target that will have to be extended multiple times in coming years. This extensibility is only feasibly if the vast majority of these well over 600 characters is mapped onto mnemonic keystroke sequences that are documented in standardized, convenient, on-screen selection menus, rather than keyboard engravings. While the most important of these mappings will have to be standardized, in the interest of efficiency for non-specialists, such a standard should also define a vendor-independent interface by which individual users or specialist user communities can easily add their own custom keystroke-sequence mappings. When checking language coverage, do not forget IPA and mathematics (at least the ISO 31-11 repertoire) as languages commonly used in secondary education.