

Keyboard layout design and special characters

Jukka K. Korpela, jkorpela@cs.tut.fi, <http://www.cs.tut.fi/~jkorpela/>

January 2008

Distribution of this memo is unlimited.

Summary

It is currently cumbersome to enter commonly needed special characters. Usually program-specific methods are used. This memo suggests an approach to keyboard layout design that makes it possible to support a few dozens of special characters on commonly used keyboards. It is based on making normal letter keys act as dead keys when used with the AltGr key, so that the dead key converts the next keystroke to a symbol, e.g. so that AltGr+C C produces ¢, AltGr+C Y produces ¥ etc. (AltGr+C would here be understood as “Currency”), whereas AltGr+S C might be defined to produce © etc. This allows a somewhat mnemonic and intuitive system to be used. On the other hand, this approach requires several AltGr+letter combinations to be free for such usage, i.e. not assigned to printable characters.

Definition of “special character”

The concept of special character is relative to what one considers as “normal” characters. Perhaps most intuitively in general, a special character is any character that is not a letter, a digit, or a commonly used punctuation mark. In this sense, even “+” and “@” are special, though mostly uninteresting for the topic, since there is usually some simple way to type them using common keyboard layouts.

Even letters of foreign alphabets can be regarded as special characters, particularly when they are used in special meanings, such as the Greek letters as used in science. When typing text dominantly in Latin letters, it is reasonable to switch to a Greek keyboard setting when Greek words in Greek letters are to be typed; but for occurrences of α and Ω in scientific texts, some faster method (such as AltGr+G A to produce α) would be preferable.

Problem description

Common keyboard layouts support a rather limited repertoire of special characters. Less common settings like Microsoft’s US International contain some additional characters from the Latin-1 Supplement, but usually in an unintuitive way, but extended settings usually deal with additional letters mostly.

Although various methods for entering arbitrary UCS characters (or at least a large subset of UCS) are available, they tend to be program-specific and not very convenient for purely keyboard-based input. This means that users run into troubles when they need to enter some special characters frequently. Keyboard shortcuts can often be defined, but they are even more program-specific and they depend on particular settings.

Among the multitude of UCS characters, only a few dozens of special characters are *frequently* needed in most kinds of text, including popular science. In highly specialized application areas, special needs arise. However, the vast majority of users writing in an alphabetic script could type

their text properly (using the most appropriate UCS characters) with ease, if two essential requirements are met:

- There are convenient ways to enter all the letters of the script (including letters with diacritics) and the commonly used punctuation marks.
- There are ways to enter the few dozens of often-needed special characters.

The first requirement can be satisfied using approaches such as dead keys for diacritics and the AltGr key (or equivalent) to enter letters outside the basic repertoire of a keyboard (such as the additional Latin letters ß, þ and œ when using a common US English keyboard). The second requirement is what we consider here.

Special nonalphabetic layout?

It would be possible to design a keyboard layout specifically for nonalphabetic characters, so that e.g. pressing C would produce ©, L would produce £, etc. It would be used in conjunction with a normal keyboard layout. Typically, on Windows systems, you can switch between two keyboard layouts just by pressing Alt+enter. Yet, even with such simplicity, typing a special character in the midst of normal text would mean something like Alt+enter C Alt+enter, i.e. a total of five keystrokes.

Moreover, when several keyboard layouts have been activated, switching between them is usually not as easy as switching between only two layouts. People who need special characters may well need to use other layouts as well, such as Greek layout or phonetic alphabet layout. Typically, a combination of three keys would be needed to select one of several layouts, implying a total of seven keystrokes for entering a single special character in the middle of normal text.

The apparently simple AltGr method

Many keyboard layouts use the AltGr key or its equivalent to turn normal keys into special character keys, e.g. making AltGr+E produce the euro sign € and AltGr+M produce the micro sign μ. Some of such assignments are widely deployed in national keyboard layouts, possibly with added engravings for them. In such situations, it would be unwise to try to use such combinations in a layout that is meant to act as a possible replacement for the national layout, in the sense that it would be a user-selectable option for the basic layout and, eventually, perhaps made even the default layout later. In a specialized layout, the situation is different. For example, in a layout for typing mathematical symbols, users do not expect AltGr+E to produce the euro sign but probably a mathematical symbol somehow resembling the letter E, such as \mathbb{E} or \in .

On a normal PC keyboard, the AltGr method consists of having the AltGr key pressed down while pressing some normal key. The AltGr key is located on the right of the space bar, and on many physical keyboards, it is labeled just as “Alt” and called “right Alt.” In the US keyboard layout, for example, this key acts just as another Alt key, but on such keyboards, you can still use keyboard layouts that assign the AltGr (alternate graphic) functionality to the “right Alt.”

On other keyboards, the method may involve using a key or keys with other names.

Extending a common keyboard layout with new AltGr assignments is possible, of course, and quite feasible if one intends to add a few characters only. If you decide that you just need © and ®, you can probably assign AltGr+C and AltGr+R to them, and users would see this as natural and easy. However, if you later decide that you also need a combination for ¢ (or %) or ℝ, you’re in trouble. Using the AltGr key together with the Shift key you could support additional characters, but it would not be obvious at all which characters are produced using AltGr+Shift+letter as opposite to

those created by just AltGr+letter. Moreover, this would still keep the number to supported special characters relatively small.

Using the Ctrl, Alt, and other special keys is one possibility but a fairly problematic one, since such key combinations are very often used for program-specific control functions. This can be serious, since the keyboard layout cannot usually be overridden by program-level assignments. The AltGr key, though equivalent to Ctrl+Alt, is usually (though not always) “free” for use in a keyboard layout.

The “symbol escape” approach

As an alternative approach, we can make combinations of AltGr and a normal key act as dead keys that modify the meaning of the next keystroke. To take a simple example, AltGr+G might be assigned so that it changes the next letter to a Greek letter, according to some simple scheme corresponding to common transliterations, e.g. AltGr+G A produces alpha (α), AltGr+G B produces beta (β), etc. Similarly, AltGr+M might be a “symbol escape” for mathematical characters, AltGr+A for arrows, etc.

Intuitively, in such an approach, combinations like AltGr+G, AltGr+M, or AltGr+A could be understood as prefix commands, e.g. “switch to Greek letters,” “switch to math symbols,” or “switch to arrows.” These prefix commands would thus not be comparable to switching between keyboard layouts but affect the next keystroke only, or possibly a combination using e.g. the Shift key and a normal key. For simplicity, and to avoid confusion, it might be better to use simple combinations only, not utilizing the Shift key. However, the Shift key might be used for the second letter key when it helps to create more natural associations.

It would not always be possible to define natural, intuitive, easy-to-remember combinations. Not all symbol collections can be associated with letters as naturally as AltGr+G might associate with Greek (to people speaking a language where the word for Greek begins with G). More importantly, not all special characters can be associated with letters, even though many of them are originally based on letters or have a shape corresponding to a letter.

Yet, this approach allows more intuitive key combinations than the alternatives, and it can be used to support hundreds of special characters using just AltGr+X Y combinations, where X and Y are keys for printable characters in the normal keyboard—letter keys, digit keys (of the normal keyboard), or punctuation keys. Even if we exclude a few of the combinations because some AltGr+X combinations are considered as assigned to special characters or because we wish to assign them that way due to the high frequency of some special characters, we would still have about 30×30 combinations. The effective number would be smaller, since we would wish to group the special characters in some natural way, and many “slots” in each group would then be unassigned. (Arbitrary assignments could be included, though, if users are expected to memorize them or use lookup tables, though such input might not be any easier than the use of input methods on technologies other than keyboard layout, such as input by Unicode number.)

Implications

To increase the potential of the “symbol escape” approach, the keyboard layout should reserve as many AltGr+something combinations as possible for use in the manner outlined above. This goal, however, needs to be weighed against national and other special needs. For example, in European countries where it is common to use AltGr+E for the euro sign, it would not be feasible to change its meaning.

In the Finnish multilingual keyboard layout, most AltGr+something combinations are unassigned, so it could be enhanced into a layout that has its current functionality and a fairly large collection of special characters. It uses e.g. AltGr+S for ß, the sharp s, and AltGr+D for ð, the letter eth, but it has a special technique for producing letters such as ł (l with stroke) and ø (o with stroke).

In a keyboard layout where special characters are regarded as more important than multilingual text input, even the added Latin letters such as ß and ð could be generated using the “symbol escape” approach. Thus, e.g. AltGr+X S and AltGr+X D might produce ß and ð, leaving AltGr+S and AltGr+D free for other use.

Due to the variation of current keyboard layouts as well as varying cultural needs, layouts based on the “symbol escape” principle would thus need to be different for different languages, countries, and environments. However, they could still be based on the same principles and be compatible whenever feasible. This would make it easier to users to move from one environment to another.

Pan-European repertoire

It is thus desirable to outline a set of special characters that should, when reasonable, be supported by keyboard layouts. For definiteness, we will focus on European usage and assume that the main character repertoire consists of Latin letters and other Ascii characters.

The question is: Which special characters are frequent enough in such a context so that the keyboard should support them, and how could they be classified into sets. Each set would then be accessible using a particular AltGr+something prefix, though the letter (or other) key used in the prefix might be different in different layouts. Such variation might depend on linguistic differences (e.g., “A” for “Arrows” works for English but not for most other languages) as well as the assignment of AltGr+something combinations for other purposes.

The MES-2 repertoire can be used as a basis for defining the repertoire, but it was partly based on other, more general considerations. MES-2 lacks some characters that are relatively often needed even in newspapers, brochures, and other common texts, such as the diameter sign \varnothing . On the other hand, it contains some mostly historical characters as well as box-drawing characters that are rarely used nowadays. Moreover, there are some very natural extensions. For example, assuming that an AltGr+something prefix is used for producing common superscript characters such as ² and ³, using e.g. AltGr+P 2 and AltGr+P 3, it would be strange if the method were not applicable to producing other superscript digits as well, even though they are much less commonly used.

The following is just a *sketchy* classification, intended to illustrate the nature of the problem rather than suggest a particular solution:

- Greek letters: α , β , γ etc. (due to their common use as symbols; only basic Greek without diacritics would be needed)
- Letterlike symbols: © ® % № ℒ
- Currency symbols: ¢ £ ¤ ¥ € ₧ ₨
- Superscripts: ^{a o 1 2 3 n}
- Logical: ¬ ∇ ∃ ∈ ∉ ∧ ∨ ∩ ∪ ⊂ ⊃
- Mathematical: ± − × ¼ ½ ¾ ⅛ ⅜ ⅝ ⅞ ‰ Δ ∏ ∑ ∙ √ ∞ ∟ ∫ ≈ ≐ ≠ ≡ ≤ ≥ ⊕ ⊗
- Physical: ° μ Ω
- Arrows: ← ↑ → ↓ ↔ ↕ ↘ (and double arrows?)
- Other symbols: · ¶ † ‡ • ’ ” ∆ ∟ ☺ ☹ ☼ ♀ ♂ ♠ ♣ ♥ ♦

Currency symbols could be classified as letterlike symbols, too, but they can be separated for practical reasons.

The following characters in Latin-1 Supplement are probably so infrequent in actual use that they need not be supported: ¡ ¨ ´ ¸ ÷. They can, when required, usually be typed using commonly available other methods (such as Alt+0nnn on Windows).

IPA characters are probably best handled using a separate IPA layout (or a more general phonetic character layout), partly because of the large number of characters needed, partly because IPA characters are usually used in succession, not as isolated characters in the midst of other text.

The lists above do not contain all MES-2 characters. There might be some characters outside MES-2 that should be supported. But roughly speaking, the classification above illustrates the nature of the assignment problem. For Mathematical and Other symbols, there might be some clashes between natural assignments. For example, when entering mathematical characters, the key to be used after the “symbol escape” could be M for minus and S for sum, but both plus-minus, per mille, and product would want to use the P key. Such problems should be solvable, however; we could associate plus minus with +, per mille with % (or the key in which % resides), and product with Shift+P, so that P would even be free for some other use!