

To: UTC
 From: Vacek Nules (forwarded by Debbie Anderson)
 Date: 29 June 2023
 Subject: Hungarian FORINT SIGN - Response

Note: Below is correspondence from the author of the Hungarian FORINT SIGN proposal ([L2/23-060R](#)). The author is responding to the report from discussion at UTC #175 that some members recommended users should use either F-ZWJ-t (<0046, 200D, 0074>) or Ft (<0046, 0074>).

I believe the following summary gives sufficient answer to the two questions posed in L2/23-083: *“What is the attestation of legacy implementations using the symbol as an atomic character? What is the communication process?”*

In Hungary a law mandates that all PCs must be sold with a preinstalled operating system (there are other countries with similar laws as far as I know). FreeDOS is usually the OS of choice for computers sold without an OEM Windows license (although, predictably, the majority of these computers get Windows installed at home anyway). Hence, and also owing to its still ongoing development, I centered my research around FreeDOS, likely the most prevalent source of forint signs destined to be interchanged with other systems.

Note: It had come to my knowledge that the maintainers of FreeDOS have renumbered, somewhat confusingly, what I refer to as code page 57781 in L2/23-060R “code page 3845”. This number represented the version of CWI–2 **without** the forint sign before 2005. It seems that “code page 779” is also sometimes used as an alias for CWI–2 on other, different systems, not related to FreeDOS. Sources differ on which code page numbers denote which exact variations.

CWI–2, the Hungarian code page commonly used in the 1980s and early 1990s, was, and due to continuing use of legacy business (e.g. accounting, billing, ticketing, POS) systems, remains popular in Hungary, because unlike code page 852, which was used under DOS for Central European languages using Latin script, it both preserves graphic characters and remains readable when misinterpreted as code page 437, character set of the original IBM PC. It had several subtle variants, some of which omitted the forint sign completely, while others shuffled it around (see Wikipedia). Since FreeDOS uses codepoint 0x9F for the forint sign, I regard this assignment as authoritative. Older variations in “homebrew” MS-DOS applications often used 0x9E or 0xA8. Unlike MS-DOS, FreeDOS natively supports CWI–2 so it is preferable against code page 852 for the aforementioned reasons.

FreeDOS does, by design, **not** support Unicode per se (owing to its aim of compatibility with MS-DOS), using a broad selection of 8-bit national codepages instead. However, there is a clearly defined long-term goal for a rudimentary level of Unicode support for at least some applications related to the FreeDOS project for the purpose of interchange, in the form of mapping tables between the national codepages and Unicode. These mapping tables, again by design, map one character in the national codepage to one character in Unicode, as evidenced by this example of such a table for a help file viewer on FreeDOS: <https://gitlab.com/FreeDOS/base/htmlhelp/-/blob/unstable/SOURCE/HTMLHELP/CP/CP437.H>. They have no pretense for mapping a 8-bit character to a character sequence, let alone one containing a zero-width formatting character.

Correspondence with Eric Auer, a developer associated with the FreeDOS project, revealed around 30 different pieces of open source, FreeDOS-related software make use of mapping tables like these.

Compilation of these mapping tables are a slow but ongoing effort in the community, and Eric seems to support and encourage the idea of adding the forint sign in Unicode to enable the de-facto Hungarian code page, hereby "code page 3845", to be mapped correctly and completely in these applications without relying on a private-use character. He writes, *"FreeDOS has started to include apps with Unicode support, but as long as there is not standard Unicode code point for the Forint sign, only preliminary support based on a temporary Unicode assignment could be added to individual apps. With an official mapping, more apps, both for DOS and other, more widespread operating systems, could support the Forint sign."*

The following is a message from another user on "Discussion and general questions about FreeDOS." [<freedos-user@lists.sourceforge.net>](mailto:freedos-user@lists.sourceforge.net) list, dated Jun 13, 2023, 10:44 AM:

Hi,

To be fair this seems like a very good idea mainly because any barriers to FreeDOS use and acceptance should be removed and this seems like a reasonably easy thing to do as it already is there.

The latter part of the above message is hard to understand but the first part makes it clear that it has a supporting stance.

This would also make a BMP codepoint (i.e. something along the lines of the originally proposed U+20C1) more desirable, again because FreeDOS is designed to be as much backwards-compatible with the "original", 16-bit MS-DOS, as possible. As far as I've checked, existing mapping tables use a 16-bit word for the Unicode target codepoint and no FreeDOS code page includes any character from the SMP (ideographic planes are obviously out of the question). I also find U+1CCFA an awkward codepoint, located in an empty space between two blocks of outlined alphanumerics which was clearly reserved only to make the codepoints of the outlined digits have the respective last digits. As the forint sign *is* a currency symbol I don't think the code point of U+20C1 is problematic in any way. There are already a few currency symbols there which are considered "legacy."

The forint is expected to be in circulation for at least a decade. Online sources which I consulted during June 2023 concur that even 2033 is a highball estimate for Euro adoption because of a soaring inflation and economic indicators suggest an even later date.

A similar, although perhaps less important, point could be made on similar interchange situations occurring between other, more legacy, systems like MS-DOS with a CWI-based custom font loaded and modern Unicode-supporting platforms. These systems lack the capacity to handle multi-byte sequences or zero width characters, being designed with correspondence between one byte, one character cell and one character taken into account.

For example, this file: <https://github.com/dosbox-staging/dosbox-staging/blob/608ca8fb7869cc5e8c618c0e6204d15c2981cc92/contrib/resources/mapping/MAIN.TXT> contains mappings from national codepages to Unicode. It has the pre-2005 version of code page 3845 without the forint sign, but contains Polish code pages with the złoty sign, its mapping looking like this:

```
0x9b 0xffffd #replacement for PLN SYMBOL, symbol not available in Unicode
```

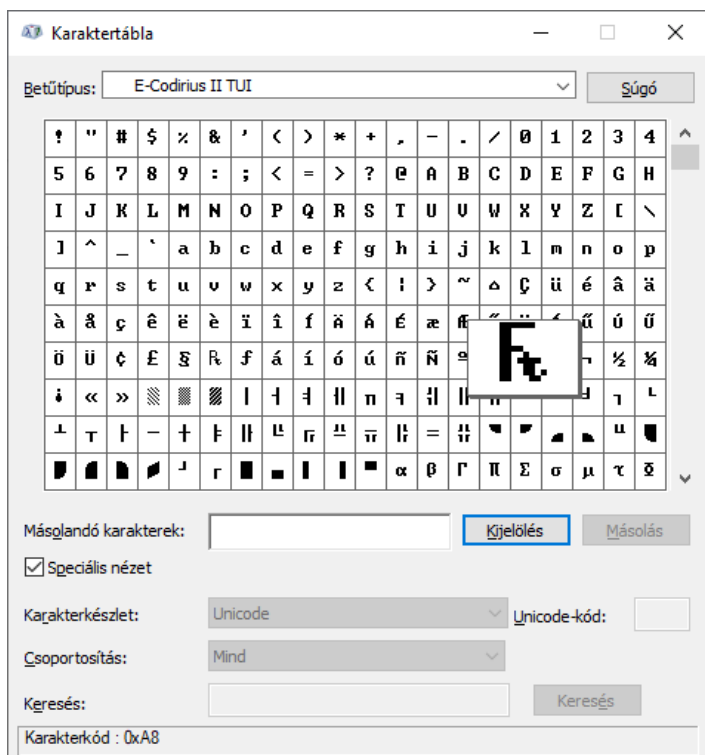
in other words, mapping the one-byte codepoint 0x9B to U+FFFD, for lack of a better atomic character, instead of a sequence of 007A, 0142 or 007A, 200D, 0142. This is because of the aforementioned character encoding model and would affect the forint sign as well had it been included in this mapping table.

In particular, running a (Free)DOS session using a CWI-family codepage in a windowed console on Windows or Linux would need a typeface including a forint sign to prevent mojibake. There are bitmap fonts already existing for the purpose of running Hungarian DOS software in a window, but they use OEM encoding and do not have Unicode mappings. However, OSes dropping support for bitmap fonts creates a need for new, Unicode-aware vector typefaces for console windows.

<https://clasqm.github.io/freedos-repo/imgs/mined.png>

This image shows the DOS text editor Mined, embraced by the FreeDOS community, running inside a console window. It has partial Unicode support, displaying the European characters supported in the currently loaded 8-bit codepage (probably CP 850 based on the available selection) and replacing unsupported ones with either suitable alternatives or universal currency signs, being a prime example for an application which utilizes the aforementioned mapping tables.

This image shows a screenshot of Windows' built-in Character Map utility, showing a custom font included with later versions of the POS software *E-Codirius II*, whose modified DOS font is shown on image 3.2 of L2/23-060R, a modified version of *Terminal* with an identical character set (complete with the custom block elements which I do **not** intend to ever propose to Unicode) to facilitate usage of the aforementioned application on Windows, in a windowed console. This font, like *Terminal*, does not have a Unicode mapping; however, a replacement in a modern format would need one (custom block elements should be private-use characters but the forint sign should not). The Ft sign is highlighted on the screenshot.



The rationale thus laid out also applies to the Polish złoty sign (minus the law on preinstalled operating systems, possibly – I’m not familiar with Polish law), which FreeDOS includes in its Polish codepage, inherited from Mazovia, a local codepage analogous to Hungarian CWI. If this proposal is accepted, the złoty sign should be proposed next, which I’ll gladly do. (The previous złoty sign proposal, 22-092, was deficient in sources and badly worded, contributing to its rejection.)

Adding an annotation like this could be useful in clarifying the forint sign’s intended use:

U+XXXX  FORINT SIGN

- for compatibility with legacy Hungarian code pages
- in running text U+0046 F U+0074 t is preferred

A nuance about the forint sign which I did not previously report is that sometimes the middle horizontal bars of the letters ‘F’ and ‘t’ fuse into one bar, as seen on images 2.1 and 3.2 of L2/23-060R. This is both induced by the low screen resolution of legacy devices and a peculiarity also observed in Hungarian handwriting which could sometimes be seen on price tags and signs in shops as well. Another annotation should be included for the FORINT SIGN entry which explicitly defines this as an acceptable alternative glyph shape. Such a glyph can be found at codepoint U+0047 in the reference font I’ve previously submitted.