L2/03-155

From: Markus Kuhn <Markus.Kuhn@cl.cam.ac.uk>
Date: 2003-05-19 14:42:35 -0700
Subject: Unicode interpretation of SOFT HYPHEN breaks ISO 8859-1 compatibility

I believe the recent "clarification" of the semantics of the SOFT HYPHEN (U+00AD) character in Unicode 4.0 had an unfortunate outcome. In particular, changing its class from Pd to Cf in UnicodeData.txt breaks backwards compatibility with how this character was widely used in ISO 8859-1 terminals for the past 15 years and causes now headaches with the designers of VT100-style terminal emulators with ISO 8859-1 and UTF-8 support.

Modern text processing systems make a clear distinction between an unformatted content data stream (e.g., a Word, TeX or HTML file) and a formatted presentation datastream (e.g., a PDF, Postscript, or PCL file). This distinction is today given for granted by people who grew up with Word and similar content-data-stream editors. Unicode with its character/glyph model is clearly targeted primarily for use in content data streams, which still have to undergo line-breaking before display. However, ISO 8859-1 was written in about 1984, when internationalization was all about adding diacritical characters for European countries, which preserved the 1:1 character glyph mapping of traditional ASCII usage. At that time, in most electronic communication, content and data presentation streams were highly similar, often identical, and commonly substituted for each other. Converting presentation data streams back into plain-text content files was common practice in the mid 1980s, when ISO 8859-1 was written. It is still very common practice for any Unix users who uses the cut&paste function to extract plain text from an xterm screen in order to copy it into a text editor.

As Unicode claims for U+0000 to U+00FF to be compatible with ISO 8859-1, it should also respect the intended and de-facto use of ISO 8859-1 characters and should not change their semantics over a decade later.

As discussed in detail for example on

  http://www.cs.tut.fi/~jkorpela/shy.html

the ISO 8859-1 standard defines, in section 6.3.3 the SOFT HYPHEN as "[a] graphic character that is imaged by a graphic symbol identical with, or similar to, that representing hyphen".

The ISO 8859-1 standard uses unfortunately only the rather unclear words "for use when a line break has been established within a word" as the complete definition of the intended usage of this character. This clearly falls short completely of setting up a document processing model and defining unambiguously what role SOFT HYPHEN plays it its various phases and functions. However, to people who routinely convert formatted presentation data streams (such as formatted ISO 8859-1 text sent to a terminal or printer) back into unformatted plain text in word processors, the meaning was obvious, as it addressed a practical need (see the above URL). Hyphens that were inserted by the line-breaking algorithm at the end of a line into the presentation data stream needed to be distinguished in formatted text from hyphens that were already present in the unformatted content data stream. Only this way could they be removed automatically when converting formatted presentation data back into unformatted plaintext that will be reformatted later. Text processing practitioners are well familiar with the problems that hyphenation hyphens can show up accidentally within lines when formatted text is imported into another application.

The definition "graphic character that is imaged by a graphic symbol identical with, or similar to, that representing hyphen" made it clear to users familiar with the above mentioned problem that the SOFT HYPHEN is just an alternative of the normal graphical character HYPHEN, for use when a hyphen is inserted by a line formatting routine. For an output device such as a printer or terminal, it can then be treated exactly like the graphical character HYPHEN. It was even placed 128 code positions above the hyphen, to limit the visual damage caused when the most-significant bit in a byte was accidentally stripped of by a text transfer.

Many years later cam the HTML specification. It's authors deal exclusively with an unformatted content data stream and therefore didn't support what appears to be the original motivation for inserting the SOFT HYPHEN in ~1984 into the ANSI draft 8-bit character set proposal that later evolved into ECMA-94 and ISO 8859-1.

Tim Berners-Lee, the original author of HTML 2 [RFC 1866], still wisely decided to leave the matter with:

> NOTE - Use of the non-breaking space and soft hyphen indicator characters is discouraged because support for them is not widely deployed.
>
> http://www.ietf.org/rfc/rfc1866.txt

Unfortunately by HTML 4, this had mutated into a complete reinterpretation of the purpose of the SOFT HYPHEN, as it had been used over the past decase in the communication with output devices:

> 9.3.3 Hyphenation
>
> In HTML, there are two types of hyphens: the plain hyphen and the soft hyphen. The plain hyphen should be interpreted by a user agent as just another character. The soft hyphen tells the user agent where a line break can occur.
>
> Those browsers that interpret soft hyphens must observe the following semantics: If a line is broken at a soft hyphen, a hyphen character must be displayed at the end of the first line. If a line is not broken at a soft hyphen, the user agent must not display a hyphen character. For operations such as searching and sorting, the soft hyphen should always be ignored.
>
> In HTML, the plain hyphen is represented by the "-" character (&#45; or &#x2D;). The soft hyphen is represented by the character entity reference &shy; (&#173; or &#xAD;)
>
> http://www.w3.org/TR/html4/struct/text.html#h-9.3.3

This HTML 4 reinterpretation is essentially the semantics that Unicode then adopted as well.

Nevertheless, there is a vast number of VT100 terminal emulators, printers, and similar 8-bit output devices out there that treat the SOFT HYPHEN as a full graphical character, as had been suggested by ISO 8859-1 and by the old application need to distinguish between content and hyphenation hyphens in formatted presentation data streams.

A number of applications use the SOFT HYPHEN as a graphical character in presentation data streams, and that is how terminal emulators such as xtermn as well as printing software have used it for well over a decade. A popular example is the gnroff command that does the formatting behind the "man" manual page tool on Linux systems.

I have a keen interest in the design of terminal emulator applications and I maintain a function definition

  http://www.cl.cam.ac.uk/~mgk25/ucs/wcwidth.c

It is used today by a number of UTF-8 terminal applications to decide, by how many character cell positions the cursor will advance if the Unicode character provided as an argument is sent to the terminal. The rules for generating its semantics from Unicode tables are very simple and include the rule

- Other format characters (general category code Cf in the Unicode database) and ZERO WIDTH SPACE (U+200B) have a column width of 0.

With the change of SOFT HYPHEN from general category code Pd to Cf in the Unicode 4.0 database, this causes now terminal behaviour to change from wcwidth(0x00ad) = 1 to wcwidth(0x00ad) = 0. In other words, what used to be a spacing graphical character in accordance with ISO 8859-1 that always advances the cursor by one cell after printing the glyph of a hyphen is not an ignoreable and usually invisible format character.

In this sense, Unicode 4.0 breaks with the well-established tradition of interpreting the SOFT HYPHEN as a graphical character in output devices.

It would have been nice, if Unicode hadn't done that. Unicode could instead have chosen to add a new ignorable formatting character for marking possible hyphenation points in documents, which could be called for instance HYPHENATION POINT. A formatting function can then either discard a HYPHENATION POINT (if it ended up inside a formatted line), or convert it into the graphical SOFT HYPHEN character, where the hyphenation point ended up at the end of a line in the presentation data stream. This would have preserved backwards compatibility with the zillions of ISO 8859-1 output devices out there that treat SOFT HYPHEN as a graphical character.

What shall I now do as the implementor of an ISO 8859-1 terminal emulator when I receive a SOFT HYPHEN?

Will the next edition of ISO 8859 be changed, to remove the definition of the SOFT HYPHEN as a graphical character?

Or, my preferred outcome, do you agree that all this SOFT HYPHEN = Cf revision was probably a mistake and we should undo everything quickly in the next revision?

Markus

--
Markus Kuhn, Computer Laboratory, University of Cambridge
http://www.cl.cam.ac.uk/~mgk25/ || CB3 0FD, Great Britain