Javascript porting of Markus Kuhn's wcwidth() implementation

The following explanation comes from the original C implementation:

This is an implementation of wcwidth() and wcswidth() (defined in IEEE Std 1002.1-2001) for Unicode.

http://www.opengroup.org/onlinepubs/007904975/functions/wcwidth.html
http://www.opengroup.org/onlinepubs/007904975/functions/wcswidth.html

In fixed-width output devices, Latin characters all occupy a single "cell" position of equal width, whereas ideographic CJK characters occupy two such cells. Interoperability between terminal-line applications and (teletype-style) character terminals using the UTF-8 encoding requires agreement on which character should advance the cursor by how many cell positions. No established formal standards exist at present on which Unicode character shall occupy how many cell positions on character terminals. These routines are a first attempt of defining such behavior based on simple rules applied to data provided by the Unicode Consortium.

For some graphical characters, the Unicode standard explicitly defines a character-cell width via the definition of the East Asian FullWidth (F), Wide (W), Half-width (H), and Narrow (Na) classes. In all these cases, there is no ambiguity about which width a terminal shall use. For characters in the East Asian Ambiguous (A) class, the width choice depends purely on a preference of backward compatibility with either historic CJK or Western practice. Choosing single-width for these characters is easy to justify as the appropriate long-term solution, as the CJK practice of displaying these characters as double-width comes from historic implementation simplicity (8-bit encoded characters were displayed single-width and 16-bit ones double-width, even for Greek, Cyrillic, etc.) and not any typographic considerations.

Much less clear is the choice of width for the Not East Asian (Neutral) class. Existing practice does not dictate a width for any of these characters. It would nevertheless make sense typographically to allocate two character cells to characters such as for instance EM SPACE or VOLUME INTEGRAL, which cannot be represented adequately with a single-width glyph. The following routines at present merely assign a single-cell width to all neutral characters, in the interest of simplicity. This is not entirely satisfactory and should be reconsidered before establishing a formal standard in this area. At the moment, the decision which Not East Asian (Neutral) characters should be represented by double-width glyphs cannot yet be answered by applying a simple rule from the Unicode database content. Setting up a proper standard for the behavior of UTF-8 character terminals will require a careful analysis not only of each Unicode character, but also of each presentation form, something the author of these routines has avoided to do so far.

http://www.unicode.org/unicode/reports/tr11/

Markus Kuhn – 2007-05-26 (Unicode 5.0)

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