NAME

Unicode − the Universal Character Set

DESCRIPTION

The international standard **ISO 10646** defines the **Universal Character Set (UCS)**. UCS contains all characters of all other character set standards. It also guarantees **round-trip compatibility**, i.e., conversion tables can be built such that no information is lost when a string is converted from any other encoding to UCS and back.

UCS contains the characters required to represent practically all known languages. This includes not only the Latin, Greek, Cyrillic, Hebrew, Arabic, Armenian, and Georgian scripts, but also Chinese, Japanese and Korean Han ideographs as well as scripts such as Hiragana, Katakana, Hangul, Devanagari, Bengali, Gurmukhi, Gujarati, Oriya, Tamil, Telugu, Kannada, Malayalam, Thai, Lao, Khmer, Bopomofo, Tibetan, Runic, Ethiopic, Canadian Syllabics, Cherokee, Mongolian, Ogham, Myanmar, Sinhala, Thaana, Yi, and others. For scripts not yet covered, research on how to best encode them for computer usage is still going on and they will be added eventually. This might eventually include not only Hieroglyphs and various historic Indo-European languages, but even some selected artistic scripts such as Tengwar, Cirth, and Klingon. UCS also covers a large number of graphical, typographical, mathematical and scientific symbols, including those provided by TeX, Postscript, APL, MS-DOS, MS-Windows, Macintosh, OCR fonts, as well as many word processing and publishing systems, and more are being added.

The UCS standard (ISO 10646) describes a *31-bit character set architecture* consisting of 128 24-bit *groups*, each divided into 256 16-bit *planes* made up of 256 8-bit *rows* with 256 *column* positions, one for each character. Part 1 of the standard (**ISO 10646-1**) defines the first 65534 code positions (0x0000 to 0xfffd), which form the *Basic Multilingual Plane (BMP)*, that is plane 0 in group 0. Part 2 of the standard (**ISO 10646-2**) adds characters to group 0 outside the BMP in several *supplementary planes* in the range 0x10000 to 0x10ffff. There are no plans to add characters beyond 0x10ffff to the standard, therefore of the entire code space, only a small fraction of group 0 will ever be actually used in the foreseeable future. The BMP contains all characters found in the commonly used other character sets. The supplemental planes added by ISO 10646-2 cover only more exotic characters for special scientific, dictionary printing, publishing industry, higher-level protocol and enthusiast needs.

The representation of each UCS character as a 2-byte word is referred to as the **UCS-2** form (only for BMP characters), whereas **UCS-4** is the representation of each character by a 4-byte word. In addition, there exist two encoding forms **UTF-8** for backwards compatibility with ASCII processing software and **UTF-16** for the backwards compatible handling of non-BMP characters up to 0x10ffff by UCS-2 software.

The UCS characters 0x0000 to 0x007f are identical to those of the classic **US-ASCII** character set and the characters in the range 0x0000 to 0x00ff are identical to those in **ISO 8859-1 Latin-1**.

Combining Characters

Some code points in **UCS** have been assigned to *combining characters*. These are similar to the non-spacing accent keys on a typewriter. A combining character just adds an accent to the previous character. The most important accented characters have codes of their own in UCS, however, the combining character mechanism allows us to add accents and other diacritical marks to any character. The combining characters always follow the character which they modify. For example, the German character Umlaut-A ("Latin capital letter A with diaeresis") can either be represented by the precomposed UCS code 0x00c4, or alternatively as the combination of a normal "Latin capital letter A" followed by a "combining diaeresis": 0x0041 0x0308.

Combining characters are essential for instance for encoding the Thai script or for mathematical typesetting and users of the International Phonetic Alphabet.

Implementation Levels

As not all systems are expected to support advanced mechanisms like combining characters, ISO 10646-1 specifies the following three *implementation levels* of UCS:

Level 1 Combining characters and **Hangul Jamo** (a variant encoding of the Korean script, where a Hangul syllable glyph is coded as a triplet or pair of vovel/consonant codes) are not supported.

Level 2 In addition to level 1, combining characters are now allowed for some languages where they are essential (e.g., Thai, Lao, Hebrew, Arabic, Devanagari, Malayalam, etc.).

Level 3 All **UCS** characters are supported.

The **Unicode 3.0 Standard** published by the **Unicode Consortium** contains exactly the **UCS Basic Multilingual Plane** at implementation level 3, as described in ISO 10646-1:2000. **Unicode 3.1** added the supplemental planes of ISO 10646-2. The Unicode standard and technical reports published by the Unicode Consortium provide much additional information on the semantics and recommended usages of various characters. They provide guidelines and algorithms for editing, sorting, comparing, normalizing, converting and displaying Unicode strings.

Unicode Under Linux

Under GNU/Linux, the C type *wchar_t* is a signed 32-bit integer type. Its values are always interpreted by the C library as **UCS** code values (in all locales), a convention that is signaled by the GNU C library to applications by defining the constant **__STDC_ISO_10646__** as specified in the ISO C99 standard.

UCS/Unicode can be used just like ASCII in input/output streams, terminal communication, plaintext files, filenames, and environment variables in the ASCII compatible **UTF-8** multi-byte encoding. To signal the use of UTF-8 as the character encoding to all applications, a suitable *locale* has to be selected via environment variables (e.g., "LANG=en_GB.UTF-8").

The **nl_langinfo(CODESET)** function returns the name of the selected encoding. Library functions such as **wctomb**(3) and **mbsrtowcs**(3) can be used to transform the internal *wchar_t* characters and strings into the system character encoding and back and **wcwidth**(3) tells, how many positions (0–2) the cursor is advanced by the output of a character.

Under Linux, in general only the BMP at implementation level 1 should be used at the moment. Up to two combining characters per base character for certain scripts (in particular Thai) are also supported by some UTF-8 terminal emulators and ISO 10646 fonts (level 2), but in general precomposed characters should be preferred where available (Unicode calls this **Normalization Form C**).

Private Area

In the **BMP**, the range 0xe000 to 0xf8ff will never be assigned to any characters by the standard and is reserved for private usage. For the Linux community, this private area has been subdivided further into the range 0xe000 to 0xefff which can be used individually by any end-user and the Linux zone in the range 0xf000 to 0xf8ff where extensions are coordinated among all Linux users. The registry of the characters assigned to the Linux zone is currently maintained by H. Peter Anvin <Peter.Anvin@linux.org>.

Literature

* Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane. International Standard ISO/IEC 10646-1, International Organization for Standardization, Geneva, 2000.

  This is the official specification of **UCS**. Available as a PDF file on CD-ROM from http://www.iso.ch/.

* The Unicode Standard, Version 3.0. The Unicode Consortium, Addison-Wesley, Reading, MA, 2000, ISBN 0-201-61633-5.

* S. Harbison, G. Steele. C: A Reference Manual. Fourth edition, Prentice Hall, Englewood Cliffs, 1995, ISBN 0-13-326224-3.

  A good reference book about the C programming language. The fourth edition covers the 1994 Amendment 1 to the ISO C90 standard, which adds a large number of new C library functions for handling wide and multi-byte character encodings, but it does not yet cover ISO C99, which improved wide and multi-byte character support even further.

* Unicode Technical Reports.
  http://www.unicode.org/unicode/reports/

* Markus Kuhn: UTF-8 and Unicode FAQ for Unix/Linux.
http://www.cl.cam.ac.uk/˜mgk25/unicode.html

Provides subscription information for the *linux-utf8* mailing list, which is the best place to look for advice on using Unicode under Linux.

* Bruno Haible: Unicode HOWTO.
ftp://ftp.ilog.fr/pub/Users/haible/utf8/Unicode-HOWTO.html

BUGS

When this man page was last revised, the GNU C Library support for **UTF-8** locales was mature and XFree86 support was in an advanced state, but work on making applications (most notably editors) suitable for use in **UTF-8** locales was still fully in progress. Current general **UCS** support under Linux usually provides for CJK double-width characters and sometimes even simple overstriking combining characters, but usually does not include support for scripts with right-to-left writing direction or ligature substitution requirements such as Hebrew, Arabic, or the Indic scripts. These scripts are currently only supported in certain GUI applications (HTML viewers, word processors) with sophisticated text rendering engines.

SEE ALSO

setlocale(3), **charsets**(7), **utf-8**(7)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.