Create account Log in

Article Talk Read Edit View history Search

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
العربية
Български
Català
Čeština
Deutsch
Ελληνικά
Español
Esperanto
فارسی
Français
Gaeilge
한국어
Italiano
עברית
Lingála
Magyar
Bahasa Melayu
Монгол
မြန်မာဘာသာ
Nederlands
日本語
Polski
Português
Русский
Simple English
Slovenčina
کوردیی ناوەندی
Türkçe
Українська
中文

Character (computing)

From Wikipedia, the free encyclopedia

This article contains special characters. Without proper rendering support, you may see question marks, boxes, or other symbols.

In computer and machine-based telecommunications terminology, a **character** is a unit of information that roughly corresponds to a grapheme, grapheme-like unit, or symbol, such as in an alphabet or syllabary in the written form of a natural language.[1]

Examples of characters include letters, numerical digits, common punctuation marks (such as "." or "-"), and whitespace. The concept also includes control characters, which do not correspond to symbols in a particular natural language, but rather to other bits of information used to process text in one or more languages. Examples of control characters include carriage return or tab, as well as instructions to printers or other devices that display or otherwise process text.

Characters are typically combined into strings.

Contents [hide]

Character encoding [edit]

Main article: Character encoding

Computers and communication equipment represent characters using a character encoding that assigns each character to something — an integer quantity represented by a sequence of digits, typically — that can be stored or transmitted through a network. Two examples of usual encodings are ASCII and the UTF-8 encoding for Unicode. While most character encodings map characters to numbers and/or bit sequences, Morse code instead represents characters using a series of electrical impulses of varying length.

Terminology [edit]

Historically, the term *character* has been widely used by industry professionals to refer to an *encoded character*, often as defined by the programming language or API. Likewise, *character set* has been widely used to refer to a specific repertoire of characters that have been mapped to specific bit sequences or numerical codes. The term glyph is used to describe a particular visual appearance of a character. Many computer fonts consist of glyphs that are indexed by the numerical code of the corresponding character.

With the advent and widespread acceptance of Unicode[2] and bit-agnostic *coded character sets*,[*clarification needed*] a character is increasingly being seen as a unit of information, independent of any particular visual manifestation. The ISO/IEC 10646 (Unicode) International Standard defines *character*, or *abstract character* as "a member of a set of elements used for the organisation, control, or representation of data". Unicode's definition supplements this with explanatory notes that encourage the reader to differentiate between characters, graphemes, and glyphs, among other things. Such differentiation is an instance of the wider theme of the separation of presentation and content.

For example, the Hebrew letter aleph ("א") is often used by mathematicians to denote certain kinds of infinity, but it is also used in ordinary Hebrew text. In Unicode, these two uses are considered different characters, and have two different Unicode numerical identifiers ("code points"), though they may be rendered identically. Conversely, the Chinese logogram for water ("水") may have a slightly different appearance in Japanese texts than it does in Chinese texts, and local typefaces may reflect this. But nonetheless in Unicode they are considered the same character, and share the same code point.

Edit links

The Unicode standard also differentiates between these abstract characters and *coded characters* or *encoded characters* that have been paired with numeric codes that facilitate their representation in computers.

Combining character [edit]

The combining character is also addressed by Unicode. For instance, Unicode allocates a code point to each of i, ["](combining trema) and ï (U+00ef). This makes it possible to code the middle character of the word naïve both as a single code point 'ï' or as a combination of the character i with diacritic (") (U+0069 LATIN SMALL LETTER I + U+0308 COMBINING DIAERESIS).

Both are considered canonically equivalent by the Unicode standard.

char [edit]

A `char` in the C programming language is a data type with the size of exactly one byte,[3] which in turn is defined to be large enough to contain any member of the basic execution character set and UTF-8 code units.[4] This implies a minimum size of 8 bits. The exact number of bits can be checked via `CHAR_BIT` macro. By far the most common size is 8 bits, and the POSIX standard *requires* it to be 8 bits.[5]

Since Unicode requires at least 21 bits to store a single code point, it is usually impossible to store one inside a single `char`; instead a variable-length encoding such as UTF-8 must be used. Unfortunately, the fact that a character was historically stored in a single byte led to the two terms being used interchangeably in most documentation. This often makes the documentation confusing or misleading when multibyte encodings such as UTF-8 are used, and has led to inefficient and incorrect implementations of string manipulation functions. Modern POSIX documentation attempts to fix this, defining "character" as a sequence of one or more bytes representing a single graphic symbol or control code, and attempts to use "byte" when referring to char data.[6] However it defines *Character Array* as an array of elements of type char.[7]

Unicode can also be stored in strings made up of code units that are larger than `char`. These are called wide characters. The original C type was called `wchar_t`. Due to some platforms defining `wchar_t` as 16 bits and others defining it as 32 bits, recent versions have added `char16_t`, `char32_t`. Even then the objects being stored might not be "characters", for instance the variable-length UTF-16 is often stored in arrays of `char16_t`.

Other languages also have a `char` type. Some such as C++ use 8 bits like C. Others such as Java use 16 bits for `char`, in order to represent UTF-16 values.

Word character [edit]

A "word" character has special meaning in some aspects of computing. A "word character" within ASCII typically means a letter of the alphabet A-Z (upper or lower case), the digits 0 to 9, and the underscore.[8][9]

It might be dependent on localization and encoding in use. If $ or | are not a word character, 'é' (in French) or 'æ' or 'я' (in Russian) or 'ά' (in Greek) are, as used in words such as fédération, Αγορά, or Примечания.

See also [edit]

- Character literal
- Fill character
- Combining character
- Universal Character Set characters
- Homoglyph

References [edit]

1. ^ http://www.merriam-webster.com/dictionary/character
2. ^ Davis, Mark (2008-05-05). "Moving to Unicode 5.1". *Google Blog*. Retrieved 2008-09-28.
3. ^ *ISO/IEC 14882:2011*. § 5.3.3 *Sizeof*.
4. ^ *ISO/IEC 14882:2011*. § 1.7 *The C++ memory model*.
5. ^ http://pubs.opengroup.org/onlinepubs/009695399/basedefs/limits.h.html
6. ^ http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html#tag_03_87
7. ^ http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html#tag_03_88
8. ^ Regexp Tutorial - Character Classes or Character Sets
9. ^ See also the `[:word:]` regular expression character class

External links [edit]

- Characters: A Brief Introduction by The Linux Information Project (LINFO)
- ISO/IEC TR 15285:1998 summarizes the ISO/IEC's character model, focusing on terminology definitions and differentiating between characters and glyphs

v· t· e	Data types [hide]
Uninterpreted	Bit · Byte · Trit · Tryte · Word
Numeric	Bignum · Complex · Decimal · Fixed point · Floating point (Double precision · Extended precision · Half precision · Minifloat · Octuple precision · Quadruple precision · Single precision) · Integer (signedness) · Interval · Rational
Text	**Character** · String (null-terminated)
Pointer	Address (physical · virtual) · Reference
Composite	Algebraic data type (generalized) · Array · Associative array · Class · Dependent · Equality · Inductive · List · Object (metaobject) · Option type · Product · Record · Set · Union (tagged)
Other	Boolean · Bottom type · Collection · Enumerated type · Exception · Function type · Opaque data type · Recursive data type · Semaphore · Stream · Top type · Type class · Unit type · Void
Related topics	Abstract data type · Data structure · Generic · Kind (metaclass) · Parametric polymorphism · Primitive data type · Protocol (interface) · Subtyping · Type constructor · Type conversion · Type system

Categories: Character encoding | Data types | Digital typography | Primitive types

This page was last modified on 4 March 2015, at 03:43.