# What is Unicode?

## Definition

**Unicode** is a universal standard for character encoding that allows providing characters from almost all written languages of the world: Latin, Cyrillic, Greek, and Arabic letters, Chinese hieroglyphs, punctuation marks, numbers, special symbols (mathematical, technical, and ideograms), etc.

## Standard’s History

The harbinger of Unicode was a single-byte (8-bit) standard, which included many encoding variants for a different range of supported languages. In this standard, the border among characters, their representation in the computer’s memory, and the display on the screen was rather arbitrary.

Because of using this standard, there were several problems: displaying documents in the wrong encoding, limited character set, converting one encoding to another, and duplicating fonts. First, each encoding represented its own character set. For example, to transfer files between devices running different operating systems, user had either to have additional fonts or apply a converter program. Second, one could work only with 256 characters at a time. In fact, first 128 positions were reserved for Latin and control characters, and remaining symbols were declared for a national alphabet and pseudo-graphics. Third, conversion from one encoding to another was possible only with partial losses when missing characters were replaced with graphically similar ones. In addition, fonts with a certain alphabetical writing system were tied to a specific encoding, and hieroglyphic writing was not presented in principle. To sum up, in 1991, the solution of *Unicode Consortium* was to create a single 16-bit standard where two bytes were used to encode each character.

As shown, after the need arose to develop a universal standard, a new type of encoding Unicodewas created. It allowed achieving the impossible: to create a tool, which supports a huge number of different characters.

## Basic Principles

The main idea of the Unicode philosophy is a clear distinguish among characters, their representation in the computer, and their display on the output device.

The standard consists of two basic sections: a *universal character set* *(UCS)* and a *Unicode transformation format* *(UTF)*. On one hand, UCS defines a unique correspondence of symbols to codes — code space elements, representing non-negative integers. As an illustration, each Unicode symbol in the set (for instance, 1 — “A”; 2 — “B”; 3 — “C”) has its position, and the character code in the set has a so-called code point, defined by the hexadecimal number and the prefix “U +” (in particular, U+0041 is the Latin letter “A”). However, for various reasons, final character codes in the computer may not always correspond to their code positions in the set, so one should distinguish the code point and the computer code. On other hand, UTF defines a machine representation of a UCS code sequence. To demonstrate, the Latin letter “A” has number 0065 (Dec) and is written with one byte in UTF-8. Overall, UCS reports that each letter corresponds to a certain number, and a specific UTF gives an accurate representation of this number in the computer.

The Unicode code space consists of 1,114,112 code positions (for UTF-16) in the range from 0 to 10FFFF, but only about 137 thousand positions (12%) are occupied. The entire space is divided into 17 planes of 2^16 (65,536) characters; nevertheless only six of them are involved now. The zero (basic) multilingual plane includes symbols of the most common scripts. The first (supplementary) plane is primarily used for historical writings. The second (supplementary ideographic) plane is applied for rarely used hieroglyphs. The third (tertiary ideographic) plane is reserved for archaic Chinese hieroglyphs. Planes 15 and 16 (supplementary private) are allocated for exclusive use, i.e. for the internal needs of various applications. To indicate Unicode characters, a record of the form “U+xxxx” (for codes 0...FFFF), or “U+xxxxx” (for codes 10000...FFFFF), or “U+xxxxxx” (for codes 100000...10FFFF) is used where xxx are hexadecimal numbers. In general, the Unicode code space allows encoding symbols of world’s main writing scripts.

All things considered, the Unicode standard is the basis for storing textual information in many modern computer systems. It consists of the universal character set and the Unicode transformation format. Herewith, any transformation format allows encoding the whole space of Unicode code positions, and conversion among various such encodings is carried out without loss of information.

## Differences between UTF-8 and UTF-16

Unicode has several main forms of representation: UTF-8, UTF-16, and UTF-32. This study compares UTF-8 and UTF-16. Both of these transformation formats are used to encode characters, but each has its pluses and minuses.

Advantages of UTF-8 are following.

* UTF-8 is backward compatible with seven-bit ASCII encoding (American Standard Code for Information Interchange). The ASCII character set has a fixed width and uses only one byte (for example, for Latin and service characters, punctuation marks, and digits). In UTF-8, symbols of other scripts occupy two or three bytes, and hieroglyphic characters can hold four bytes.
* When one encodes a file, which uses only ASCII characters in the UTF-8, the resulting file is identical to an ASCII-encoded file.
* UTF-8 does not contain null bytes, so it allows using strings with a null symbol and provides backward compatibility.
* UTF-8 is a byte-oriented format and, therefore, has no problem with byte networks or files. In addition, there is no byte order problem: the most significant byte is placed before the byte with the lowest address (Big-Endian) or after its (Little Endian).
* UTF-8 better restores errors, which damage parts of a file or stream, so it can still decode the next intact byte.
* UTF-8 is useful during the work with a large amount of monolingual information.

Disadvantages of UTF-8 are next.

* UTF-8 allows encoding only a small part of the Unicode spectrum.
* Although the byte order does not matter, sometimes UTF-8 still has such specification. In these cases, the byte order notifies that the text is encoded in UTF-8. However, this violates compatibility with ASCII software even if the text contains only ASCII characters (for example, Microsoft Notepad software).
* Many common symbols have different lengths, so it slows code point indexing and the calculation of the number of codes.

Advantages of UTF-16 are presented below.

* Basic Multilingual Plane, including Latin, Cyrillic, most Chinese hieroglyphs, etc. (65,536 positions), can be recorded by two bytes. This accelerates indexing and calculation of the number of code points in case if the text does not contain additional characters.
* Even if the text has additional characters, they are still represented by pairs of 16-bit values. The total length is divided by two and allows the use of the 16-bit character data type “char” as a primitive string component.
* UTF-16 is often used to store string information in applications because of the simplicity of this transformation format and its prevalence in the encoding of major writing systems (for instance, inside Java and Windows OS).

Disadvantages of UTF-16 are indicated in the following list.

* Outdated software, which does not support Unicode, will not be able to open the UTF-16 file. Encoding a file with only ASCII characters is not possible using UTF-16 because each symbol will have two bytes.
* When using UTF-16 as a variable-length encoding, indexing characters as code points is a time-consuming task. In turn, using UTF-16 as a fixed-length encoding works in many common scenarios (alphabets), but one needs to correctly handle “surrogate pairs” (a first element of the pair is from the U+D800...U+DBFF field, and a second element is from U+DC00...U+DFFF).
* UTF-16 contains many null bytes in ASCII strings, that is, there are no null strings and a lot of lost memory.
* UTF-16 encoding is not byte-oriented, so it must set the byte order for such networks.
* UTF-16 restores errors byte corruption, but some bytes are lost. A lost byte can mix following byte combinations and distort the final result.

After all, both UTF-8 and UTF-16 transformation formats are variable-length encodings. First of all, a UTF-8 character can occupy a minimum of 8 bits whereas the symbol length in UTF-16 begins with 16 bits. Second, UTF-8 encoding is ASCII-compatible while UTF-16 is ASCII-incompatible. Third, a UTF-8 encoded file tends to be smaller than a UTF-16 encoded file. Next, UTF-8 is byte-oriented whereas UTF-16 is not. UTF-8 also restores errors better than UTF-16. In general, UTF-16 is usually better suited for in-memory representation because the byte order problem “Big Endian/Little Endian” (BE/LE) does not matter here, and indexing is faster (with proper surrogate pair processing). On other hand, UTF-8 encoding is very good for text files and network protocols because there is no BE/LE problem and there is zero completion and ASCII-compatibility.

## Advantages and Disadvantages of Unicode

The universal Unicode character-encoding standard has its pros and cons.

Advantages of the Unicode standard include the following features. Firstly, due to the huge stock of characters for encrypting of various characters, it became possible to encode almost all existing alphabets. Secondly, the problem of converting different encodings has disappeared. Thirdly, Elvish letters and duplication of fonts were excluded. In addition, a useful feature of the standard is normalization, that is, Unicode does not spend computer resources regularly checking the same character, which may have similar spelling in different alphabets. For this purpose, a special algorithm takes out similar characters separately by a column and refers to them. Indeed, these positive features allowed Unicode to take a leading position in the world.

Unicode problems lie in some non-universal coding principles. First, letters of different alphabets, which are identical in spelling, are same symbols (for example, Russian “a” and Serbian “a”). Second, switching between horizontal and vertical writing orientation is not provided in Unicode (for example, for hieroglyphic scripts). Third, Unicode provides for the possibility of different styles (italics) of same character, depending on the language. Conversion from lowercase letters to capital letters also depends on the language. In addition, some disadvantages relate to the capabilities of text handlers. In detail, the performance of all string-processing programs is reduced when using Unicode instead of single-byte encodings. In short, these shortcomings are not critical and can be neglected in most cases.

As shown above, despite the fact that Unicode has some disadvantages, the standard continues to hold a leading position in the world. This was made possible largely because this standard has become easily implemented and widespread.

## Conclusion

This paper addressed various aspects of the Unicode character-encoding standard, which was an attempt to create a single set of symbols for basic writing scripts of the world. The main idea is a clear distinguish among characters, their representation in the computer, and their display on the output device. The standard consists of the universal character set and the Unicode transformation format and includes 1,114,112 code positions. The most popular encodings are 8-bit UTF-8 and 16-bit UTF-16. UTF-16 is usually better for storing string information in applications, and UTF-8 is very good for text files and network protocols. Due to their features, these Unicode transformation formats have become mainstream in many computer systems. In the future, Unicode code space is planned to be replenished with new characters, letters and signs.