**Character representation**

*A character is a letter, digit, punctuation mark, or other symbol that you might type in on your keyboard.*

Computers store characters as numbers – each letter or other symbol has a unique number (for example, A might be represented by the number 65, B by 66 etc). A character string such as "Hello world!" is represented by a list of numbers.

What is a ***character set*** is a list of all the characters which are recognized by a computer's hardware and software. It also defines which number is used to represent each character. In order to transfer text data between different computer systems, we need to use a standard character set that all computers understand.

There are two main popular character sets which are very widely used

* ASCII (American Standard Code for Information Interchange), pronounced “ASK ee”)
* UNICODE (universal character encoding)

ASCII, stands for American Standard Code for Information Interchange. It's an 8-bit character code where every single bit represents a unique character. It represents a set of the main characters used in English language text. These are:

* The upper-case letters A to Z
* The lower-case letters a to z
* The digits 0 to 9
* A selection of over 30 punctuation symbols including full stop, comma, brackets, quotes and more
* The space character

ASCII was first developed in the 1960s, but relatively few people used it at the time. An alternative scheme, called EBCDIC, was more popular.

The first IBM PCs, created in the early 1980s, used ASCII, and its popularity grew from that.

ASCII character set usually uses 8 bits (1 byte) to store each character. The 8th bit is used as a check digit, meaning that only 7 bits are available to store each character.

|  |  |
| --- | --- |
| Lower case letters of the alphabet | 26 |
| Upper case letters of the alphabet | 26 |
| Number symbols | 10 |
| Punctuation marks and white space | 33 |
| Control codes such as carriage return and line feed | 32 |
| Total set | 127 |

This gives ASCII the ability to store a total of 2^7 = 128 different values. Therefore, the ASCII character set goes from binary 0 to binary 127 (1111111). The character set is broken down like this

To make up the 128 characters the code 000 means NUL. This means it does not do or show nothing.

The first 32 characters in the ASCII table (**NUL** through to **US**) are called control characters. Character 127 (**DEL**) is also a control character. **CR** (carriage return) and **LF** (line feed), which are used to mark the end of each line in a text file, and the **TAB** character which is used to align text. Most of the other control characters are no longer really used very often.

Characters 32 to 126 are used for printable characters:

* 48 to 57 are used for the digits zero to nine (in hex these are values 0x30 to 0x39)
* 65 to 90 are the upper-case letters (hex 0x41 to 0x5A)
* 97 to 122 are the lower-case letters (hex 0x61 to 0x7A)
* The gaps are filled with punctuation characters

Limitations of the ASCII Code

* Can only represent 127 or 255 characters
* Can only accept letters in the English language

## ASCII conversion

Converting a text string to ASCII values is a simple matter of looking up each character in the table. For example, to convert "Hello" to ASCII, do the following:

* Look up **H** in the table, it has a decimal value 72
* Look up **e** in the table, it has a decimal value 115
* and so on, to the end of the string

Take care to look up the upper or lower case letter correctly. You should find that the ASCII values for the string "Hello" are:

[72, 115, 108, 108, 111]

Converting ASCII codes back to a text string is just as easy - look up the letter corresponding to each number. For example, to convert:

[119, 111, 114, 108, 100]

Do the following

* Look up decimal 119 in the table, it corresponds to the letter **w**
* Look up decimal 114 in the table, it corresponds to the letter **o**
* etc

## Using ASCII codes in a program

Most programming languages allow you to convert between ASCII values and text characters.

example in Python 3 you can use the chr() function to convert a value to text., this code prints **B** (the character with ASCII value 66):

c = chr(66)

print(c)

This code prints 69 (the ASCII value of character 'D'):

c = ord('D')

print(c)

Most languages have similar functions.

# Extended ASCII

computers usually store ASCII text as one character per byte. A byte can store values 0 to 255, but ASCII codes only use values 0 to 127. This means that the codes 128 to 255 are unused.

Extended ASCII makes us of these unused codes to define extra characters.

UNICODE

ASCII is quite limited, it only allows for the English alphabet, numerals and a few punctuation symbols. Only 5% of the world's population speak English as a first language. Schemes such as Extended ASCII provided limited support for other languages and alphabets, but with the increasing importance of computers in society, and in particular the growth of the web, a better solution is needed.

Unicode is a universal character encoding standard. It is designed to support characters from all languages around the world.

## Unicode structure

Unicode characters are each given a value between 0 and 65536 (0x0000 to 0xFFFF in hexadecimal). This range is divided into blocks, where each block contains characters for a particular alphabet.

* Value 0 to 127 form the Basic Latin block. These characters exactly match the ASCII character set.
* Value 128 to 255 form the Latin 1 block. These characters exactly match the default Extended ASCII character set.
* These are followed by further blocks for other alphabets: Greek, Cyrillic, Hebrew, Arabic and many more.

As Unicode was being developed, it became apparent that 65536 code points was not enough to cover every character known to man. Unicode was extended to allow extra "planes" with even more characters. However, the extra planes contain ancient or obscure characters which are rarely used.

Advantages

* Unicode is used universally by all browsers and mobile devices.
* Supported by most programming languages such as Java, XML, Javascript, C++, etc.
* Unicode can represent over 1 million characters.
* It’s open source, free to use
* It supports emoji characters.

Disadvantage

* More bits are needed for non-ASCII characters.