Text file

A **text file** (sometimes spelled **textfile**; an old alternative name is **flatfile**) is a kind of computer file that is structured as a sequence of lines of electronic text. A text file exists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special characters, known as an end-of-file (EOF) marker, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows and Unix-like systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the file size in bytes. Most text files need to have end-of-line delimiters, which are done in a few different ways depending on operating system. Some operating systems with record-orientated file systems may not use new line delimiters and will primarily store text files with lines separated as fixed or variable length records.

Text file

Filename extension	`.txt`
Internet media type	`text/plain`
Type code	TEXT
Uniform Type Identifier (UTI)	public.plain-text
UTI conformation	public.text
Type of format	Document file format, Generic container format

"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[1]

Data storage

Because of their simplicity, text files are commonly used for storage of information. They avoid some of the problems encountered with other file formats, such as endianness, padding bytes, or differences in the number of bytes in a machine word. Further, when data corruption occurs in a text file, it is often easier to recover and continue processing the remaining contents. A disadvantage of text files is that they usually have a low entropy, meaning that the information occupies more storage than is strictly necessary.

A simple text file may need no additional metadata (other than knowledge of its character set) to assist the reader in interpretation. A text file may contain no data at all, which is a case of zero-byte file.

A stylized iconic depiction of a CSV-formatted **text file**.

Encoding

The ASCII character set is the most common compatible subset of character sets for English-language text files, and is generally assumed to be the default file format in many situations. It covers American English, but for the British pound sign, the euro sign, or characters used outside English, a richer character set must be used. In many systems, this is chosen based on the default locale setting on the computer it is read on. Prior to UTF-8, this was traditionally single-byte encodings (such as ISO-8859-1 through ISO-8859-16) for European languages and wide character encodings for Asian languages.

Because encodings necessarily have only a limited repertoire of characters, often very small, many are only usable to represent text in a limited subset of human languages. Unicode is an attempt to create a common standard for representing all known languages, and most known character sets are subsets of the very large Unicode character set. Although there are multiple character encodings available for Unicode, the most common is UTF-8, which has the advantage of being backwards-compatible with ASCII; that is, every ASCII text file is also a UTF-8 text file with identical meaning. UTF-8 also has the advantage that it is easily auto-detectable. Thus, a common operating mode of UTF-8 capable software, when opening files of unknown encoding, is to try UTF-8 first and fall back to a locale dependent legacy encoding when it definitely is not UTF-8.

Formats

On most operating systems, the name *text file* refers to a file format that allows only plain text content with very little formatting (e.g., no **bold** or *italic* types). Such files can be viewed and edited on text terminals or in simple text editors. Text files usually have the MIME type `text/plain`, usually with additional information indicating an encoding.

Microsoft Windows text files

MS-DOS and Microsoft Windows use a common text file format, with each line of text separated by a two-character combination: carriage return (CR) and line feed (LF). It is common for the last line of text *not* to be terminated with a CR-LF marker, and many text editors (including Notepad) do not automatically insert one on the last line.

On Microsoft Windows operating systems, a file is regarded as a text file if the suffix of the name of the file (the "filename extension") is `.txt`. However, many other suffixes are used for text files with specific purposes. For example, source code for computer programs is usually kept in text files that have file name suffixes indicating the programming language in which the source is written.

Most Microsoft Windows text files use ANSI, OEM, Unicode or UTF-8 encoding. What Microsoft Windows terminology calls "ANSI encodings" are usually single-byte ISO/IEC 8859 encodings (i.e. ANSI in the Microsoft Notepad menus is really "System Code Page", non-Unicode, legacy encoding), except for in locales such as Chinese, Japanese and Korean that require double-byte character sets. ANSI encodings were traditionally used as default system locales within Microsoft Windows, before the transition to Unicode. By contrast, OEM encodings, also known as DOS code pages, were defined by IBM for use in the original IBM PC text mode display system. They typically include graphical and line-drawing characters common in DOS applications. "Unicode"-encoded Microsoft Windows text files contain text in UTF-16 Unicode Transformation Format. Such files normally begin with byte order mark (BOM), which communicates the endianness of the file content. Although UTF-8 does not suffer from endianness problems, many Microsoft Windows programs (i.e. Notepad) prepend the contents of UTF-8-encoded files with BOM,[2] to differentiate UTF-8 encoding from other 8-bit encodings.[3]

Unix text files

On Unix-like operating systems, text files format is precisely described: POSIX defines a text file as a file that contains characters organized into zero or more lines,[4] where lines are sequences of zero or more non-newline characters plus a terminating newline character,[5] normally LF.

Additionally, POSIX defines a **printable file** as a text file whose characters are printable or space or backspace according to regional rules. This excludes most control characters, which are not printable.[6]

Apple Macintosh text files

Prior to the advent of macOS, the classic Mac OS system regarded the content of a file (the data fork) to be a text file when its resource fork indicated that the type of the file was "TEXT".[7] Lines of classic Mac OS text files are terminated with CR characters.[8]

Being a Unix-like system, macOS uses Unix format for text files.[8] Uniform Type Identifier (UTI) used for text files in macOS is "public.plain-text"; additional, more specific UTIs are: "public.utf8-plain-text" for utf-8-encoded text, "public.utf16-external-plain-text" and "public.utf16-plain-text" for utf-16-encoded text and "com.apple.traditional-mac-plain-text" for classic Mac OS text files.[7]

Rendering

When opened by a text editor, human-readable content is presented to the user. This often consists of the file's plain text visible to the user. Depending on the application, control codes may be rendered either as literal instructions acted upon by the editor, or as visible escape characters that can be edited as plain text. Though there may be plain text in a text file, control characters within the file (especially the end-of-file character) can render the plain text unseen by a particular method.

See also

- ASCII
- EBCDIC
- Filename extension
- List of file formats
- Newline
- Syntax highlighting
- Text editor
- Unicode

Notes and references

1. Lewis, John (2006). *Computer Science Illuminated* (https://archive.org/details/computersciencei00nell). Jones and Bartlett. ISBN 0-7637-4149-3.
2. "Using Byte Order Marks" (https://docs.microsoft.com/en-gb/windows/win32/intl/using-byte-order-marks). *Internationalization for Windows Applications*. Microsoft. Jan 7, 2021. Archived (https://web.archive.org/web/20230221224807/https://learn.microsoft.com/en-gb/windows/win32/intl/using-byte-order-marks) from the original on Feb 21, 2023. Retrieved 2022-04-21.
3. Freytag, Asmus (2015-12-18). "FAQ – UTF-8, UTF-16, UTF-32 & BOM" (https://www.unicode.org/faq/utf_bom.html#BOM). The Unicode Consortium. Retrieved 2016-05-30. "Yes, UTF-8 can contain a BOM. However, it makes *no* difference as to the endianness of the byte stream. UTF-8 always has the same byte order. An initial BOM is only used as a signature — an indication that an otherwise unmarked text file is in UTF-8. Note that some recipients of UTF-8 encoded data do not expect a BOM. Where UTF-8 is used *transparently* in 8-bit environments, the use of a BOM will interfere with any protocol or file format that expects

specific ASCII characters at the beginning, such as the use of "#!" of at the beginning of Unix shell scripts."

4. "3.403 Text File" (http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html#tag_03_403). *IEEE Std 1003.1, 2017 Edition*. IEEE Computer Society. Retrieved 2019-03-01.
5. "3.206 Line" (http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html#tag_03_206). *IEEE Std 1003.1, 2013 Edition*. IEEE Computer Society. Retrieved 2015-12-15.
6. "3.284 Printable File" (http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html#tag_03_284). *IEEE Std 1003.1, 2013 Edition*. IEEE Computer Society. Retrieved 2015-12-15.
7. "System-Declared Uniform Type Identifiers" (https://developer.apple.com/library/prerelease/content/documentation/Miscellaneous/Reference/UTIRef/Articles/System-DeclaredUniformTypeIdentifiers.html). *Guides and Sample Code*. Apple Inc. 2009-11-17. Retrieved 2016-09-12.
8. "Designing Scripts for Cross-Platform Deployment" (https://developer.apple.com/library/mac/documentation/OpenSource/Conceptual/ShellScripting/PortingScriptstoMacOSX/PortingScriptstoMacOSX.html). *Mac Developer Library*. Apple Inc. 2014-03-10. Retrieved 2016-09-12.

External links

- Power of Plain Text on C2 wiki

-