

The file Command

The file command attempts to classify each filesystem object (i.e., file, directory or link) that is provided to it as an argument (i.e., input). Thus, it can usually provide immediate information as to whether some specified object is, for example, a GIF89a image file, a directory, a GNU tar archive, ASCII English text, a symbolic link, an HTML document, an empty file, bzip2 compressed data, an ELF 32-bit LSB executable, etc.

file accomplishes this by probing each object with three types of tests until one succeeds. The first is a filesystem test, which uses the stat system call to obtain information from the object's inode (which contains information about a file). A system call is a request in a Unix-like operating system for a service performed by the kernel (i.e., the core of the operating system).

The second test checks to see if there is a magic number, which is a number embedded at or near the beginning of many types of files that indicates the file format (i.e., the type of file).

In the event that the first two tests fail to determine the type of a file, language tests are employed to determine if it is plain text (i.e., composed entirely of human-readable characters), and, if so, what type of plain text, such as HTML (hypertext markup language) or source code (i.e., the original version of a program as written by a human). In this situation, file also attempts to determine the natural language (e.g., English, Turkish or Japanese) that is used in the file.

A simplified version of file's syntax is

```
file [option(s)] object_name(s)
```

file has several options, but it is most commonly used without any of them. For example, information about a file named file1 that is located in the in the current directory (i.e., the directory in which the user is currently working) could be obtained by merely typing the following and pressing the RETURN key:

```
file file1
```

Information about the types of all of the files in the current directory can be obtained by using the star wildcard to represent every object in that directory as follows:

```
file *
```

Likewise, information about all of the files in another directory can be obtained by using that directory as an argument and following it immediately by a forward slash and the star wildcard. For example, the following classifies all of the objects in the /boot directory:

```
file /boot/*
```

The square brackets wildcard can be used together with the star wildcard to show the file types for only those objects whose names begin with specified letters or with a specified range of letters. For example, the following would show only those objects in the current directory whose names begin with letters a through g:

```
file [a-g]*
```

The -k option tells file to not stop at the first successful test, but to keep going; this can result in the reporting of additional information about some filesystem objects. The -b (i.e., brief) option tells file to not prepend filenames to output lines, which can be useful when compiling statistics about file types. The -v option returns information about the version of file that is installed.