

filefilter — Convert text or binary patterns in a file

Syntax	Description	Options	Remarks and examples
Stored results	Reference	Also see	

Syntax

```
filefilter oldfile newfile ,
    { from(oldpattern) to(newpattern) | ascii2ebcdic | ebcdic2ascii } [options]
```

where *oldpattern* and *newpattern* for ASCII characters are

"string" or *string*

string := [*char*[*char*[*char*[...]]]]

char := *regchar* | *code*

regchar := ASCII 32–91, 93–128, 161–255; excludes ‘\’

<i>code</i>	:= \BS	backslash
	\r	carriage return
	\n	newline
	\t	tab
	\M	Classic Mac EOL, or \r
	\W	Windows EOL, or \r\n
	\U	Unix or Mac OS X EOL, or \n
	\LQ	left single quote, ‘
	\RQ	right single quote, ’
	\Q	double quote, ”
	\\$	dollar sign, \$
	\###d	3-digit [0–9] decimal ASCII
	\##h	2-digit [0–9, A–F] hexadecimal ASCII

<i>options</i>	Description
----------------	-------------

* <u>f</u> rom(<i>oldpattern</i>)	find <i>oldpattern</i> to be replaced
* <u>t</u> o(<i>newpattern</i>)	use <i>newpattern</i> to replace occurrences of from()
* ascii2ebcdic	convert file from ASCII to EBCDIC
* ebcdic2ascii	convert file from EBCDIC to ASCII
<u>r</u> eplace	replace <i>newfile</i> if it already exists

* Both from(*oldpattern*) and to(*newpattern*) are required, or ascii2ebcdic or ebcdic2ascii is required.

Description

filefilter reads an input file, searching for *oldpattern*. Whenever a matching pattern is found, it is replaced with *newpattern*. All resulting data, whether matching or nonmatching, are then written to the new file.

Because of the buffering design of `filefilter`, arbitrarily large files can be converted quickly. `filefilter` is also useful when traditional editors cannot edit a file, such as when unprintable ASCII characters are involved. In fact, converting end-of-line characters between Mac OS X, Classic Mac, Windows, and Unix is convenient with the EOL codes.

Unicode is not directly supported at this time, but you can attempt to operate on a Unicode file by breaking a 2-byte character into the corresponding two-character ASCII representation. However, this goes beyond the original design of the command and is technically unsupported. If you attempt to use `filefilter` in this manner, you might encounter problems with variable-length encoded Unicode.

Although it is not mandatory, you may want to use quotes to delimit a pattern, protecting the pattern from Stata's parsing routines. A pattern that contains blanks must be in quotes.

Options

`from(oldpattern)` specifies the pattern to be found and replaced. It is required unless `ascii2ebcdic` or `ebcdic2ascii` is specified.

`to(newpattern)` specifies the pattern used to replace occurrences of `from()`. It is required unless `ascii2ebcdic` or `ebcdic2ascii` is specified.

`ascii2ebcdic` specifies that characters in the file be converted from ASCII coding to EBCDIC coding. `from()`, `to()`, and `ebcdic2ascii` are not allowed with `ascii2ebcdic`.

`ebcdic2ascii` specifies that characters in the file be converted from EBCDIC coding to ASCII coding. `from()`, `to()`, and `ascii2ebcdic` are not allowed with `ebcdic2ascii`.

`replace` specifies that *newfile* be replaced if it already exists.

Remarks and examples

[stata.com](http://www.stata.com)

Convert Classic Mac-style EOL characters to Windows-style

```
. filefilter macfile.txt winfile.txt, from(\M) to(\W) replace
```

Convert left quote (‘) characters to the string “left quote”

```
. filefilter auto1.csv auto2.csv, from(\LQ) to("left quote")
```

Convert the character with hexadecimal code 60 to the string “left quote”

```
. filefilter auto1.csv auto2.csv, from(\60h) to("left quote")
```

Convert the character with decimal code 96 to the string “left quote”

```
. filefilter auto1.csv auto2.csv, from(\096d) to("left quote")
```

Convert strings beginning with hexadecimal code 6B followed by “Text” followed by decimal character 100 followed by “Text” to an empty string (remove them from the file)

```
. filefilter file1.txt file2.txt, from("\6BhText\100dText") to("")
```

Convert file from EBCDIC to ASCII encoding

```
. filefilter ebcdicfile.txt asciifile.txt, ebcdic2ascii
```

Stored results

filefilter stores the following in `r()`:

Scalars

<code>r(occurrences)</code>	number of <i>oldpattern</i> found
<code>r(bytes_from)</code>	# of bytes represented by <i>oldpattern</i>
<code>r(bytes_to)</code>	# of bytes represented by <i>newpattern</i>

Reference

Riley, A. R. 2008. Stata tip 60: Making fast and easy changes to files with filefilter. *Stata Journal* 8: 290–292.

Also see

- [P] [file](#) — Read and write ASCII text and binary files
- [D] [changeol](#) — Convert end-of-line characters of text file
- [D] [hexdump](#) — Display hexadecimal report on file