For example, `-COE` and `-C6` will both turn on UTF-8-ness on both STDOUT and STDERR. Repeating letters is just redundant, not cumulative nor toggling.

The `io` options mean that any subsequent open() (or similar I/O operations) will have the `:utf8` PerlIO layer implicitly applied to them, in other words, UTF-8 is expected from any input stream, and UTF-8 is produced to any output stream. This is just the default, with explicit layers in open() and with binmode() one can manipulate streams as usual.

`-C` on its own (not followed by any number or option list), or the empty string `""` for the `PERL_UNICODE` environment variable, has the same effect as `-CSDL`. In other words, the standard I/O handles and the default `open()` layer are UTF-8-fied **but** only if the locale environment variables indicate a UTF-8 locale. This behaviour follows the *implicit* (and problematic) UTF-8 behaviour of Perl 5.8.0.

You can use `-C0` (or `"0"` for `PERL_UNICODE`) to explicitly disable all the above Unicode features.

The read-only magic variable `${^UNICODE}` reflects the numeric value of this setting. This is variable is set during Perl startup and is thereafter read-only. If you want runtime effects, use the three-arg open() (see open in *perlfunc*), the two-arg binmode() (see binmode in *perlfunc*), and the open pragma (see *open*).

(In Perls earlier than 5.8.1 the `-C` switch was a Win32-only switch that enabled the use of Unicode-aware "wide system call" Win32 APIs. This feature was practically unused, however, and the command line switch was therefore "recycled".)

-c

causes Perl to check the syntax of the program and then exit without executing it. Actually, it *will* execute `BEGIN`, `CHECK`, and `use` blocks, because these are considered as occurring outside the execution of your program. `INIT` and `END` blocks, however, will be skipped.

-d

runs the program under the Perl debugger. See *perldebug*.

-d:*foo[=bar,baz]*

runs the program under the control of a debugging, profiling, or tracing module installed as Devel::foo. E.g., **-d:DProf** executes the program using the Devel::DProf profiler. As with the **-M** flag, options may be passed to the Devel::foo package where they will be received and interpreted by the Devel::foo::import routine. The comma-separated list of options must follow a = character. See *perldebug*.

-D*letters*

-D*number*

sets debugging flags. To watch how it executes your program, use **-Dtls**. (This works only if debugging is compiled into your Perl.) Another nice value is **-Dx**, which lists your compiled syntax tree. And **-Dr** displays compiled regular expressions; the format of the output is explained in *perldebguts*.

As an alternative, specify a number instead of list of letters (e.g., **-D14** is equivalent to **-Dtls**):

```
        1  p  Tokenizing and parsing
        2  s  Stack snapshots
                with v, displays all stacks
        4  l  Context (loop) stack processing
        8  t  Trace execution
       16  o  Method and overloading resolution
       32  c  String/numeric conversions
       64  P  Print profiling info, preprocessor command for -P, source file input state
      128  m  Memory allocation
      256  f  Format processing
      512  r  Regular expression parsing and execution
     1024  x  Syntax tree dump
     2048  u  Tainting checks
     4096     (Obsolete, previously used for LEAKTEST)
     8192  H  Hash dump -- usurps values()
```