PERL - SPECIAL VARIABLES

There are some variables which have a predefined and special meaning in Perl. They are the variables that use punctuation characters after the usual variable indicator $, @, or, such as $_ (explained below).

Most of the special variables have an english like long name, e.g., Operating System Error variable !canbewrittenasOS_ERROR. But if you are going to use english like names, then you would have to put one line **use English;** at the top of your program file. This guides the interpreter to pickup exact meaning of the variable.

The most commonly used special variable is $_, which contains the default input and pattern-searching string. For example, in the following lines −

```
#!/usr/bin/perl

foreach ('hickory','dickory','doc') {
   print $_;
   print "\n";
}
```

When executed, this will produce the following result −

```
hickory
dickory
doc
```

Again, let's check the same example without using $_ variable explicitly −

```
#!/usr/bin/perl

foreach ('hickory','dickory','doc') {
   print;
   print "\n";
}
```

When executed, this will also produce the following result −

```
hickory
dickory
doc
```

The first time the loop is executed, "hickory" is printed. The second time around, "dickory" is printed, and the third time, "doc" is printed. That's because in each iteration of the loop, the current string is placed in ,andisusedbydefaultbyprint.HerearetheplaceswherePerlwillassume_ even if you don't specify it −

- Various unary functions, including functions like ord and int, as well as the all file tests −f, −d except for -t, which defaults to STDIN.
- Various list functions like print and unlink.
- The pattern-matching operations m//, s///, and tr/// when used without an =~ operator.
- The default iterator variable in a foreach loop if no other variable is supplied.
- The implicit iterator variable in the grep and map functions.
- The default place to put an input record when a line-input operation's result is tested by itself as the sole criterion of a while test i.e. , . Note that outside of a while test, this will not happen.

Special Variable Types

Based on the usage and nature of special variables we can categorize them in the following categories −

- Global Scalar Special Variables.
- Global Array Special Variables.
- Global Hash Special Variables.

- Global Special Filehandles.
- Global Special Constants.
- Regular Expression Special Variables.
- Filehandle Special Variables.

Global Scalar Special Variables

Here is the list of all the scalar special variables. We have listed corresponding english like names along with the symbolic names.

$_	The default input and pattern-searching space.
$ARG	
$.	The current input line number of the last filehandle that was read. An explicit close on the filehandle resets the line number.
$NR	
$/	The input record separator; newline by default. If set to the null string, it treats blank lines as delimiters.
$RS	
$,	The output field separator for the print operator.
$OFS	
$\	The output record separator for the print operator.
$ORS	
$"	Like "$," except that it applies to list values interpolated into a double-quoted string (or similar interpreted string). Default is a space.
$LIST_SEPARATOR	
$;	The subscript separator for multidimensional array emulation. Default is "\034".
$SUBSCRIPT_SEPARATOR	
$^L	What a format outputs to perform a formfeed. Default is "\f".
$FORMAT_FORMFEED	
$:	The current set of characters after which a string may be broken to fill continuation fields starting with ^ in a format. Default is "\n"".
$FORMAT_LINE_BREAK_CHARACTERS	
$^A	The current value of the write accumulator for format lines.
$ACCUMULATOR	
$#	Contains the output format for printed numbers *deprecated*.
$OFMT	
$?	The status returned by the last pipe close, backtick ‘‘ command, or system operator.
$CHILD_ERROR	
$!	If used in a numeric context, yields the current value of the errno variable, identifying the last system call error. If used in a string context, yields the corresponding system error string.
$OS_ERROR or ERRNO	
$@	The Perl syntax error message from the last eval command.
$EVAL_ERROR	
$$	The pid of the Perl process running this script.
$PROCESS_ID or PID	
$<	The real user ID *uid* of this process.
$REAL_USER_ID or UID	

$>	The effective user ID of this process.
$EFFECTIVE_USER_ID or $EUID	
$(	The real group ID *gid* of this process.
$REAL_GROUP_ID or $GID	
$)	The effective gid of this process.
$EFFECTIVE_GROUP_ID or $EGID	
$0	Contains the name of the file containing the Perl script being executed.
$PROGRAM_NAME	
$[	The index of the first element in an array and of the first character in a substring. Default is 0.
$]	Returns the version plus patchlevel divided by 1000.
$PERL_VERSION	
$^D	The current value of the debugging flags.
$DEBUGGING	
$^E	Extended error message on some platforms.
$EXTENDED_OS_ERROR	
$^F	The maximum system file descriptor, ordinarily 2.
$SYSTEM_FD_MAX	
$^H	Contains internal compiler hints enabled by certain pragmatic modules.
$^I $INPLACE_EDIT	The current value of the inplace-edit extension. Use undef to disable inplace editing.
$^M	The contents of $M can be used as an emergency memory pool in case Perl dies with an out-of-memory error. Use of $M requires a special compilation of Perl. See the INSTALL document for more information.
$^O $OSNAME	Contains the name of the operating system that the current Perl binary was compiled for.
$^P	The internal flag that the debugger clears so that it doesn't debug itself.
$PERLDB	
$^T	The time at which the script began running, in seconds since the epoch.
$BASETIME	
$^W	The current value of the warning switch, either true or false.
$WARNING	
$^X	The name that the Perl binary itself was executed as.
$EXECUTABLE_NAME	
$ARGV	Contains the name of the current file when reading from <ARGV>.

Global Array Special Variables

@ARGV	The array containing the command-line arguments intended for the script.

@INC	The array containing the list of places to look for Perl scripts to be evaluated by the do, require, or use constructs.
@F	The array into which the input lines are split when the -a command-line switch is given.

Global Hash Special Variables

%INC	The hash containing entries for the filename of each file that has been included via do or require.
%ENV	The hash containing your current environment.
%SIG	The hash used to set signal handlers for various signals.

Global Special Filehandles

ARGV	The special filehandle that iterates over command line filenames in @ARGV. Usually written as the null filehandle in <>.
STDERR	The special filehandle for standard error in any package.
STDIN	The special filehandle for standard input in any package.
STDOUT	The special filehandle for standard output in any package.
DATA	The special filehandle that refers to anything following the __END__ token in the file containing the script. Or, the special filehandle for anything following the __DATA__ token in a required file, as long as you're reading data in the same package __DATA__ was found in.
_ *underscore*	The special filehandle used to cache the information from the last stat, lstat, or file test operator.

Global Special Constants

__END__	Indicates the logical end of your program. Any following text is ignored, but may be read via the DATA filehandle.
__FILE__	Represents the filename at the point in your program where it's used. Not interpolated into strings.
__LINE__	Represents the current line number. Not interpolated into strings.
__PACKAGE__	Represents the current package name at compile time, or undefined if there is no current package. Not interpolated into strings.

Regular Expression Special Variables

$digit	Contains the text matched by the corresponding set of parentheses in the last pattern matched. For example, $1 matches whatever was contained in the first set of parentheses in the previous regular expression.
$&	The string matched by the last successful pattern match.
$MATCH	
$`	The string preceding whatever was matched by the last successful pattern match.
$PREMATCH	
$'	The string following whatever was matched by the last successful pattern match.
$POSTMATCH	
$+	The last bracket matched by the last search pattern. This is useful if you don't know which of a set of alternative patterns was matched. For example: /Version: . * |Revision: . * / && $*rev* = $+;
$LAST_PAREN_MATCH	

Filehandle Special Variables

$|	If set to nonzero, forces an fflush3 after every write or print on the currently selected output channel.
$OUTPUT_AUTOFLUSH	
$%	The current page number of the currently selected output channel.
$FORMAT_PAGE_NUMBER	
$=	The current page length *printablelines* of the currently selected output channel. Default is 60.
$FORMAT_LINES_PER_PAGE	
$-	The number of lines left on the page of the currently selected output channel.
$FORMAT_LINES_LEFT	
$~	The name of the current report format for the currently selected output channel. Default is the name of the filehandle.
$FORMAT_NAME	

$^ $FORMAT_TOP_NAME	The name of the current top-of-page format for the currently selected output channel. Default is the name of the filehandle with _TOP appended.