# Perl File/Dir Operations

## File Test Operators

| | |
|---|---|
| **-r -w -x** | File is readable/writable/executable by effective uid/gid<br>next unless -f $file;<br>if(-r “test.pl”) { print “readable\n” } |
| **-R -W -X** | File is readable/writable/executable by real uid/gid. |
| **-o -O** | File is owned by effective/real uid. |
| **-e -z** | File exists/has zero size. |
| **-s** | File exists and has non-zero size. Returns the size. |
| **-f -d** | File is a plain file/a directory. |
| **-l -S -p** | File is a symbolic link/a socket/a named pipe (FIFO). |
| **-b -c** | File is a block/character special file. |
| **-u -g -k** | File has setuid/setgid/sticky bit set. |
| **-t** | Tests if filehandle (STDIN by default) is opened to a tty |
| **-T -B** | File is a text/non-text (binary) file.<br>-T and -B return true on a null file, or a file at EOF when testing a |
| **-M -A -C** | File modification / access / inode-change time.<br>Measured in days. Value returned reflects the file age at the time the<br>search on $^T in the Special Variables<br>next unless -M $file > .5; # older then 12 hours |

## File Operations

| | |
|---|---|
| **chmod** LIST | Changes the permi<br>The first element of the |
| **chown** LIST | Changes the owner a<br>The first two elements of the |
| **truncate** FILE, SIZE | Truncates FILE to SIZE. FIL |
| **link** OLDFILE, NEWFILE | Creates a new filenar |
| **lstat** FILE | Like stat, but does not |
| **mkdir** DIR, MODE | Creates a directory with gi |
| **readlink** EXPR | Returns the va |
| **rename** OLDNAME, NEWNAME | Changes t |

| | |
|---|---|
| **rmdir** NAME | Deletes the directory if |
| **stat** FILE | Returns a<br>(0: $dev, 1: $ino, 2: $m<br>6: $rdev, 7: $size, 8:<br>11: $blks<br>FILE can be a filehandle, an<br>or _ to refer to the last file test operation |
| **symlink** OLDFILE, NEWFILE | Creates a new filename symb |
| **unlink LIST** | Deletes |
| **utime LIST** | Changes the acces<br>The first two elements of the list must |

## The following filename conventions apply when opening a file.

**“FILE”** open FILE for input. also “<FILE”.
**“>FILE”** open FILE for output, creating it if necessary.
open(LOG,”>$access_log”) || die “Can’t open User Access Log: $!\n”;
**“>>FILE”** open FILE in append mode. open(LOG,”>>$access_log”);
**“+<FILE”** open FILE with read/write access (file must exist).
**“+>FILE”** open FILE with read/write access (file truncated).
**“|CMD”** opens a pipe **to** command CMD; forks if CMD is.
open(LOG,”| ausgabe”);
**“CMD|”** opens a pipe **from** command CMD; forks if CMD is.
open(LOG,”eingabe |”);
open(DATUM,”date |”);
open(WORT,”| wc > wort.dat”); # wc : counter