NAME

NDBM_File - Tied access to ndbm files

SYNOPSIS

```
 use Fcntl;   # For O_RDWR, O_CREAT, etc.
 use NDBM_File;

 tie(%h, ’NDBM_File’, ’filename’, O_RDWR|O_CREAT, 0666)
   or die "Couldn’t tie NDBM file ’filename’: $!; aborting";

 # Now read and change the hash
 $h{newkey} = newvalue;
 print $h{oldkey};
 ...

 untie %h;
```

DESCRIPTION

"NDBM_File" establishes a connection between a Perl hash variable and a file in NDBM_File format;. You can manipulate the data in the file just as if it were in a Perl hash, but when your program exits, the data will remain in the file, to be used the next time your program runs.

Use "NDBM_File" with the Perl built-in "tie" function to establish the connection between the variable and the file. The arguments to "tie" should be:

1. The hash variable you want to tie.

2. The string "NDBM_File". (Ths tells Perl to use the "NDBM_File" package to perform the functions of the hash.)

3. The name of the file you want to tie to the hash.

4. Flags. Use one of:

   "O_RDONLY"
   Read-only access to the data in the file.

   "O_WRONLY"
   Write-only access to the data in the file.

"O_RDWR"
Both read and write access.

If you want to create the file if it does not exist, add "O_CREAT" to any of these, as in the example. If you omit "O_CREAT" and the file does not already exist, the "tie" call will fail.

5. The default permissions to use if a new file is created. The actual permissions will be modified by the user's umask, so you should probably use 0666 here. (See "umask" in perlfunc.)

DIAGNOSTICS

On failure, the "tie" call returns an undefined value and probably sets $! to contain the reason the file could not be tied.

"ndbm store returned -1, errno 22, key "..." at ..."

This warning is emitted when you try to store a key or a value that is too long. It means that the change was not recorded in the database. See BUGS AND WARNINGS below.

SECURITY AND PORTABILITY

Do not accept NDBM files from untrusted sources.

On modern Linux systems these are typically GDBM files, which are not portable across platforms.

The GDBM documentation doesn't imply that files from untrusted sources can be safely used with "libgdbm".

Systems that don't use GDBM compatibilty for ndbm support will be using a platform specific library, possibly inherited from BSD systems, where it may or may not be safe to use an untrusted file.

A maliciously crafted file might cause perl to crash or even expose a security vulnerability.

BUGS AND WARNINGS

There are a number of limits on the size of the data that you can store in the NDBM file. The most important is that the length of a key, plus the length of its associated value, may not exceed 1008 bytes.

See "tie" in perlfunc, perldbmfilter, Fcntl