



Reading from Files

How to read from a file in Perl

Page Resource

[CGI Resources](#) | [Perl Tutorials](#)

[Home](#)/[CGI](#)/[Perl](#)/File Reading

Reading from a file in Perl gives us a way to create "flat text file" databases. These can be useful, and are fairly easy to set up. The downside is if you need to deal with file permissions, and the possibility of data being overwritten. However, as we get into it further we will discuss those issues in more detail. First, let's take a look at how to get started.

To Begin: Create a File

Our first step is to create a file so we have something to read. Suppose we want to store a few pro wrestler's names and some other data about them, like their crowd reaction and favorite moves. For this, we could put each wrestler on a line, and separate the wrestler's information using a separator character (delimiter). One that is often used for separation is the pipe symbol (|). We will use it here to separate our data. Here is what we want to store:

Wrestler Name	Crowd Reaction	Favorite Move
The Rock	Cheer	Rock Bottom
Triple H	Boo	Pedigree
Stone Cold	Cheer	Stone Cold Stunner

Now, we can take this data and put it in a file in a similar way. We won't use the headings, just the wrestlers and their information:

```
The Rock|Cheer|Rock Bottom
Triple H|Boo|Pedigree
Stone Cold|Cheer|Stone Cold Stunner
```

Each wrestler has a new line for his information, and the information on each line is separated with the pipe symbol. Remember to be sure the new line is started after the last entry (hit "enter" right after the last character but don't put anything on the new line). This is so Perl sees a "\n" character at the end of each line. When we chop the lines after reading them in, this will keep the last character from being chopped instead. Just be sure there is no new data (even a space) on the new line though, or it will read it as a new line of information.

Once it is ready, we can save it as some type of text file. We can use lots of extensions, such as .txt, .dat, or other things. However, if someone stumbles onto the file in their browser, they can easily read the contents. One thing that helps a little is to give it the same extension as your executable cgi scripts. This way, the server tries to execute the file if it is called from a browser, and should return a permission error or an internal server error. If your server executes files with the .cgi extension (ask your host, some use .pl or others instead), then save the file with that extension, like:

```
wrestldata.cgi
```

Once it is saved, be sure the file has the permissions set so it is readable (755 should be OK here, if you plan to write to it you may want to use 777, see the [CHMOD page](#) for more). Once that is done, we need to make a script which will use it. For ease of writing and of having the right location for the file, we will assume the data file and script will be in the same directory. If you choose to use separate directories, be sure to make those changes.

Opening the File

Within our script, we will want to read the data into our script. In order to do so, we must first open the file. We do this with a command like this:

```
open(HANDLE, "FileName/Location");
```

The HANDLE above is something you will use to reference the file when you read from it and when you close it. The FileName/Location is the actual location of the file. Since we will have them in the same directory, we can just use the filename. If you have it in another directory, use the server path to the file. Here is how we can open our file:

```
open(DAT, "wrestledata.cgi");
```

Of course, you may want to assign the filename to a variable, so you could change it later more easily if you need to:

```
$data_file="wrestledata.cgi";
open(DAT, $data_file);
```

One last bit on the opening of the file. You may want to have an option to show an error if the file cannot be opened. So, we can add the "die" option to print the error to standard output. What we will do is use the open command, give the "or" option (two pipe symbols) and use the "die" routine as the option:

```
$data_file="wrestledata.cgi";
open(DAT, $data_file) || die("Could not open file!");
```

Reading the File

Now we are able to read from the open file. The easiest way to do this is to just assign the contents of the file to an array:

```
$data_file="wrestledata.cgi";
open(DAT, $data_file) || die("Could not open file!");
@raw_data=<DAT>;
```

This will take everything from the file and toss it into the @raw_data array. Notice the use of the DAT handle for reading, with the < and > around it. We can then use the array to grab the information later, so that we can go ahead and close the file.

Close the File!

We have to be sure to remember to close the file when we are done with it, so we close it with the close command:

```
close(DAT);
```

Again, the DAT handle is used to reference the file and close it. So now we have:

```
$data_file="wrestledata.cgi";
open(DAT, $data_file) || die("Could not open file!");
@raw_data=<DAT>;
close(DAT);
```

This is enough to read in the data, but if we want to make use of it we will want to pull it out of the array and do something with it. For that, we should head on to the second part of this tutorial.

Well, let's go on to: [Reading Files 2: Using the Data](#).

Partners

[CoolHomepages](#) | [Web Design Library](#) | [Website Content](#)

The tutorials and articles on these pages are © 1997-2005 by John Pollock and may not be reposted without written permission from the author, and may not be reprinted for profit.



[Previous](#)

By: [John Pollock](#)



[Next](#)

[Main Page](#) | [HTML](#) | [JavaScript](#) | [Graphics](#) | [DHTML/Style Sheets](#) | [ASP/PHP](#)
[PutWeb/FTP](#) | [CGI/Perl](#) | [Promotion](#) | [Java](#) | [Design Articles](#)
[Support Forums](#) | [Site Search](#) | [FAQs](#) | [Privacy](#) | [Contact](#)

Copyright © 1997-2005 [The Web Design Resource](#). All rights reserved.