Navigation IO in Unix / Linux

Most Unix system commands receive input from your terminal and send the output back to your terminal. A command usually reads its input from a location called standard input, which happens to your terminal by default.

Most Unix system commands receive input from your terminal and send the output back to your terminal. A command usually reads its input from a location called standard input, which happens to your terminal by default. In a similar way, a command usually outputs to standard output (standard output), which also happens to your terminal by default.

Navigate the output in Unix / Linux

Output from a command that tends to wait for standard output can easily be directed to a file. This ability is known as re-navigating the output.

If the> file symbol is assigned to any command, it usually writes its output to standard output, the output of that command will be written to the file instead of your terminal.

We check the **who** command, which will redirect the entire output of the command in files users:

```
$ who > users
```

Notice that no output appears on the terminal. That's because the output has been redirected from the default standard output device (terminal) into a specific file. If you check the file users then that file will have full content:

```
$ cat users oko tty01 Sep 12 07 : 30 ai tty15 Sep 12 13 : 32 ruth tty21 Sep 12
```

If a command with output is redirected to a file and the file already contains some data, that data will be lost. Consider the following example:

```
$ echo line 1 > users $ cat users line 1 $
```

You can use the >> operator to assign the output to an existing file as follows:

```
$ echo line 2 >> users $ cat users line 1 line 2 $
```

Redirecting input in Unix / Linux

Similar to the output of the command, the command input can also redirect from a file. When the larger character> is used for input navigation, the smaller character

Commands that often receive their input from standard input can have input from a file according to the user's operation. For example, to calculate the number of lines in a users file, you can run the following command:

```
$ wc - l users 2 users $
```

Here it will produce 2 lines. You can calculate the number of lines in the file by redirecting the standard input of wc command from users.

```
$ wc - l  users 2 $
```

Remember, there is a difference in output by 2 patterns of wc command. In the first case, the name of the users file is listed with the line number, but in the second case it is not.

In the first case, wc knows that it is reading its input from users. In the second case, it only knows that it is its input from the standard input so it does not display the file name.

Here document in Unix / Linux

Here a document is used to redirect input into an interactive shell script or program.

We can run an interactive program in a shell script without the user's manipulation providing the required input for the program or the interactive shell script.

The general template for this document is:

```
command delimiter document delimiter
```

Here, the shell interprets the operator as an instruction to read the input until it finds a line containing the specified limit. All input that is above the line containing the limit is then given as the standard input of the command.

This limit tells the shell that here the document is done. Without it, the shell continues to read the input forever. The limit must be a single word that does not contain spaces or tabs.

Below is the input of the **wc -l** command to calculate the total number of lines.

```
$ wc - l  EOF This is a simple lookup program cho good ( and sai ) restaurants
```

You can use here document to print multiple lines using your script as follows:

```
#! / bin / sh cat  EOF This is a simple lookup program cho good ( và sai ) res
```

This code will produce the following result:

```
  ?ây là m?t Lookup ??n ch??
ng trình này cho good ( và sai ) restaurants trong Cape Town .
```

The following script runs an area with the Micro Text Editor and stores the input into the test.txt file.

```
  #! / bin / sh filename = test .  txt vi $ filename  EndOfCommands i This file
???c t?o ???c t? ??ng t? m?t script này ^ [ ZZ EndOfCommands
```

If you run this script with vim that works similarly, then you will see the output like this:

```
  $ sh test .  sh Vim : C?nh báo : ??u vào không ph?i t? m?t dòng $
```

After running the script, you will see the following output added to the test.txt file.

```
  $ cat test .  txt t?p tin này ???c t?o ???c t? ??ng t? m?
t $ shell script
```

Remove the output in Unix / Linux

Sometimes you will need to run a command, but you do not want the output displayed on the screen. In such cases, you can remove the output by redirecting it into the file / dev / null:

```
  $ command > / dev / null
```

Here, command is the name of the command you want to run. File / dev / null is a special file that automatically removes all its input.

To remove both the output of a command and the faulty output of the command, you use standard redirection to redirect STDERR to STDOUT.

```
  $ command > / dev / null 2 > & 1
```

Here 2 represents STDERR and 1 represents STDOUT. You can display a message on STDERR by redirecting STDIN to STDERR as follows:

```
  $ echo message 1 > & 2
```

The commands are redirected in Unix / Linux

Below is a complete list of commands that you can use to redirect:

Command Descriptionpgm> file The output of pgm is directed to the file. pgm pgm >> file The output of pgm is assigned to the file. n> file Output from the stream with the n sign being directed to the file. n >> file Output from stream with n assigned to file. n> & m Merging output from stream n with stream m. n & m Merging input from stream n with stream m. standard input tag comes from here via the next tag at the beginning of the line. | Get output from a program or a process and send it to another program .
Note that the 0 file signature is usually the standard input (STDIN), 1 is usually the standard output (STDOUT), and 2 is usually the standard error output (STAERR).

According to Tutorialspoint

Previous article: Techniques cited in Unix / Linux

Next lesson: Shell functions

guides. Thank you for reading and for following us regularly.