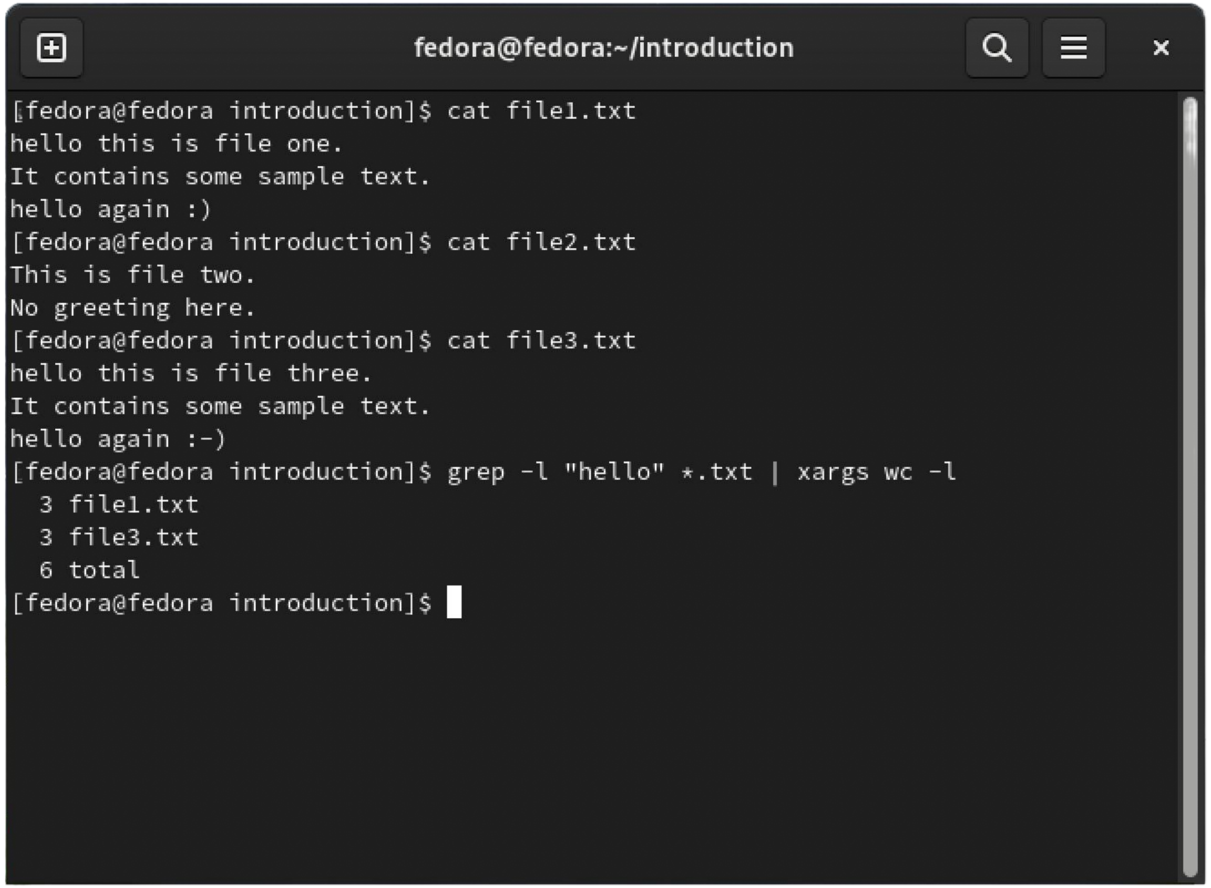
Red Hat

Pipes, Grep, Xargs

Command line chaining

Introduction to command line chaining



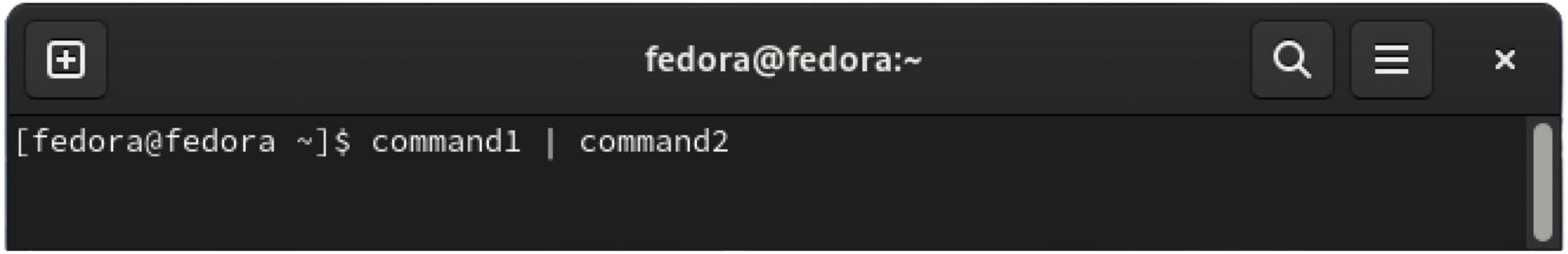
```
[fedora@fedora introduction]$ cat file1.txt
hello this is file one.
It contains some sample text.
hello again :)
[fedora@fedora introduction]$ cat file2.txt
This is file two.
No greeting here.
[fedora@fedora introduction]$ cat file3.txt
hello this is file three.
It contains some sample text.
hello again :-)
[fedora@fedora introduction]$ grep -l "hello" *.txt | xargs wc -l
  3 file1.txt
  3 file3.txt
  6 total
[fedora@fedora introduction]$ 
```

- Introduction to command line chaining
- Understanding Pipes
- Using grep for searching
- Introduction to xargs

What is command line chaining ?

What is command line chaining?

Command line chaining allows us to link multiple commands together so that they work in sequence as part of one single operation. This allows us to use the output of one command as the input for another.

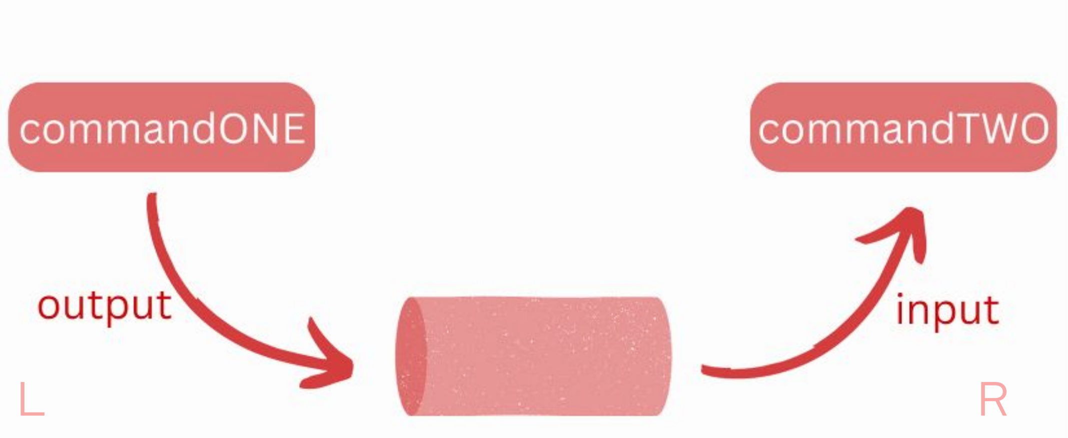


```
[fedora@fedora ~]$ command1 | command2
```

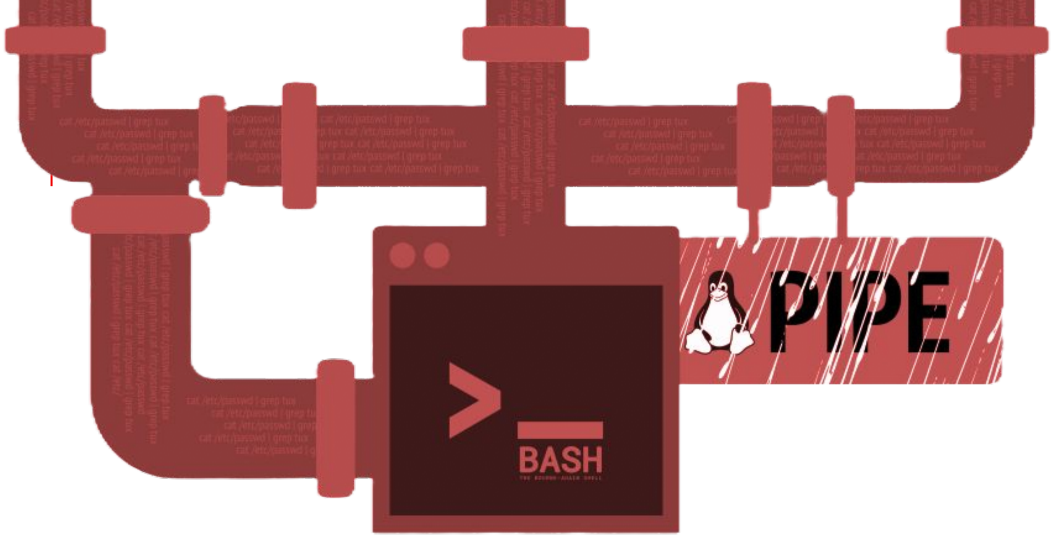
In the above command1 run first and it output is passed into command2.

Understanding pipes

What is a pipe?



- A pipe is represented by (|)
- It take the standard output on the command on the left and uses it as a standard input for the command on the right.

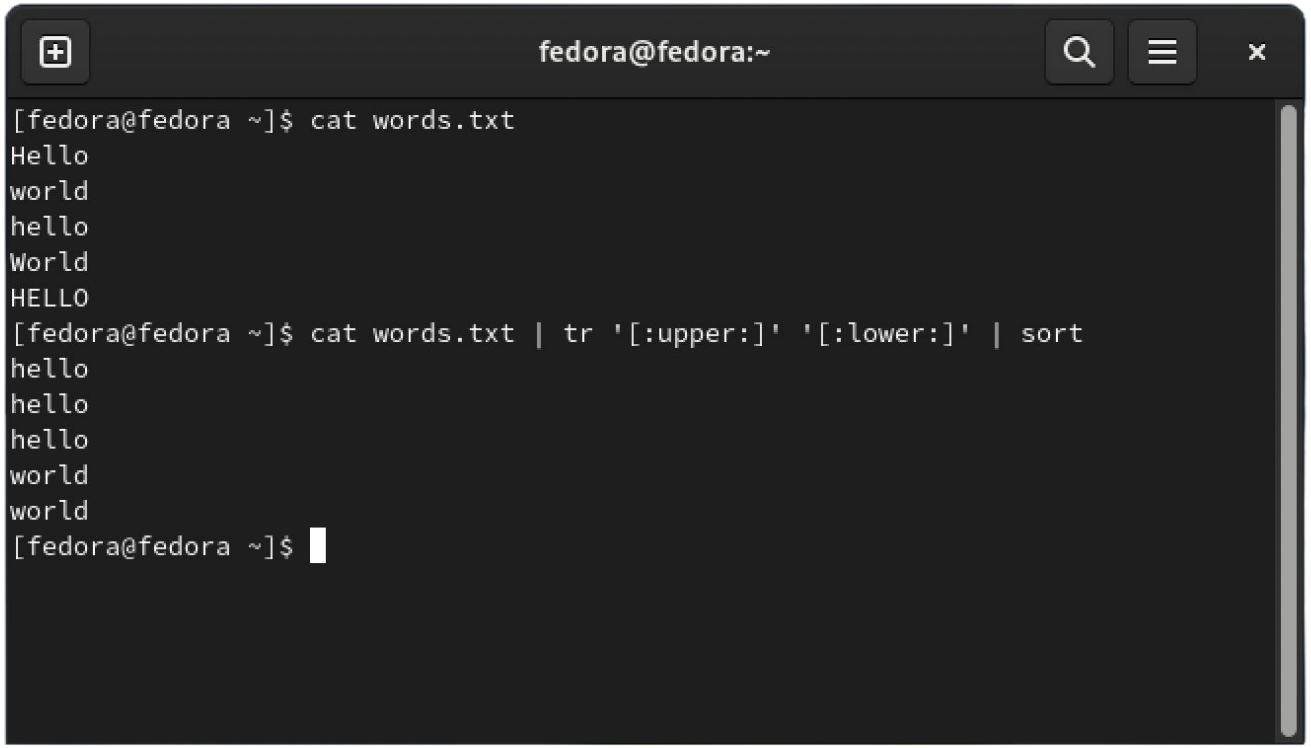


Why use a pipe?

- Combine simple commands to make more complex operations.
- Processes data step by step without the use of writing to a file.
- It easily redirects and filters output between commands.

Example of a pipe (|)

Example of a pipe (|)



```
[fedora@fedora ~]$ cat words.txt
Hello
world
hello
World
HELLO
[fedora@fedora ~]$ cat words.txt | tr '[:upper:]' '[:lower:]' | sort
hello
hello
hello
world
world
[fedora@fedora ~]$ 
```

In this example we have a text file containing “hello” and “world” on separate lines. We have chained three commands using two pipes to read the content, make the text lowercase and then sort the text.

```
cat words.txt | tr '[:upper:]' '[:lower:]' | sort
```

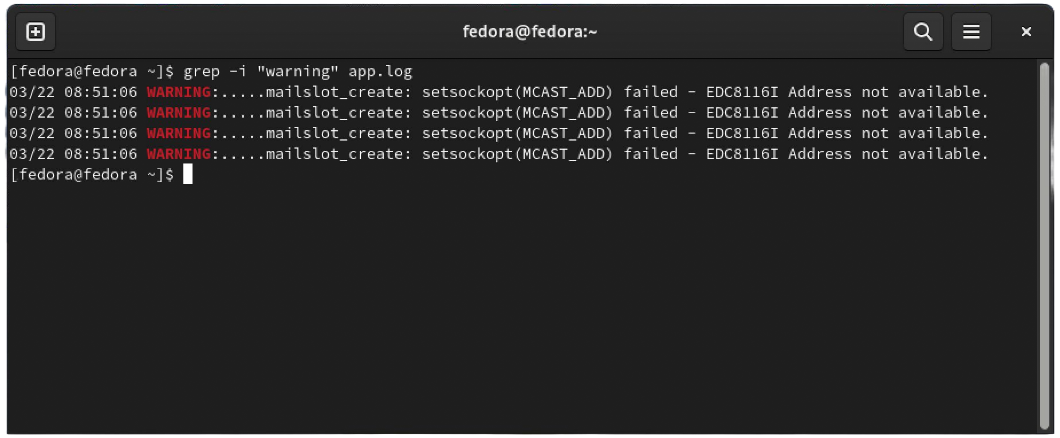
The first command, “cat”, outputs the content of the “words.txt”.

The output from the “cat” is piped meaning all of the file content is passed directly to the next command. The “tr” command with “[:upper]” and “[:lower:]” arguments means translate uppercase letters into lowercase. Meaning it reads the text from the standard input (cat) and transforms every uppercase letter to lower.

Finally the pipe now passes the lowercase text to the “sort” command which arranges the lines in alphabetical order and output the results to the terminal.

Understanding greps

What is grep?



```
[fedora@fedora ~]$ grep -i "warning" app.log
03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
[fedora@fedora ~]$
```

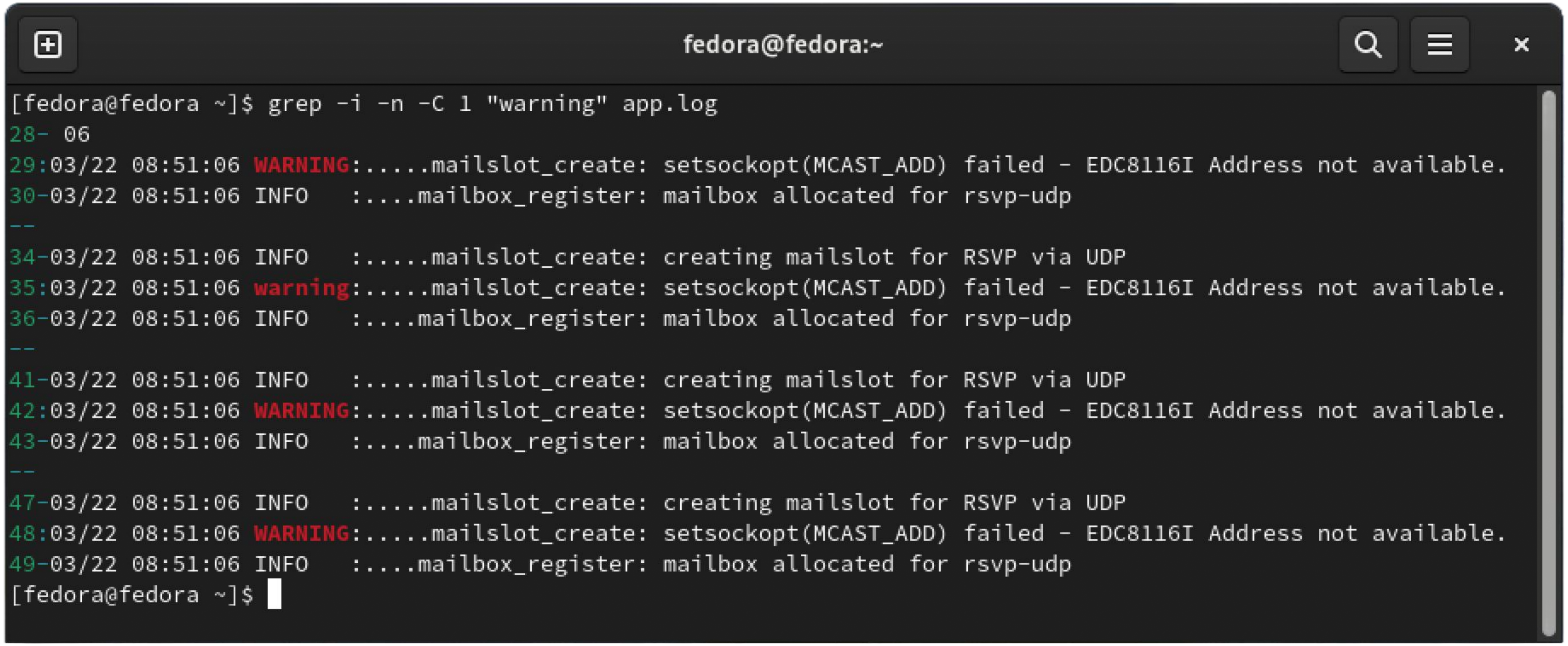
Grep stands for “global regular expression print”. It is a command line utility that searches for patterns in files and outputs the lines that match the given pattern. Grep allow for both simple string searches and regular expression to filter and find what you're looking for.

Why use grep?

- Grep can quickly scan large amounts of text and find specific patterns.
- It handles both simple and complex searches
- Grep works seamlessly with pipes making it ideal for scripting.
- Grep has multiple optional arguments allowing you to control the search behaviour even more.

Example of grep

Example of grep



```
[fedora@fedora ~]$ grep -i -n -C 1 "warning" app.log
28- 06
29:03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
30-03/22 08:51:06 INFO   :....mailbox_register: mailbox allocated for rsvp-udp
--
34-03/22 08:51:06 INFO   :.....mailslot_create: creating mailslot for RSVP via UDP
35:03/22 08:51:06 warning:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
36-03/22 08:51:06 INFO   :....mailbox_register: mailbox allocated for rsvp-udp
--
41-03/22 08:51:06 INFO   :.....mailslot_create: creating mailslot for RSVP via UDP
42:03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
43-03/22 08:51:06 INFO   :....mailbox_register: mailbox allocated for rsvp-udp
--
47-03/22 08:51:06 INFO   :.....mailslot_create: creating mailslot for RSVP via UDP
48:03/22 08:51:06 WARNING:.....mailslot_create: setsockopt(MCAST_ADD) failed - EDC8116I Address not available.
49-03/22 08:51:06 INFO   :....mailbox_register: mailbox allocated for rsvp-udp
[fedora@fedora ~]$
```

The above grep command searches in “app.log” for a “warning” message which is not case sensitive e.g. WARNING, Warning. For each match it shows one line of context before and after along with the line number.

```
grep -i -n -C 1 "warning" app.log
```

Here is a breakdown of each part of the argument:

- **Grep:** Is the command that searches through text for a specified pattern.
- **-i:** Make the search non case sensitive.
- **-n:** Outputs the line number.
- **-C 1:** Shows one line of context before and after the matching line.
- **“Warning”:** The search pattern.
- **app.log:** The file in which to search for the pattern.

Understanding Xargs

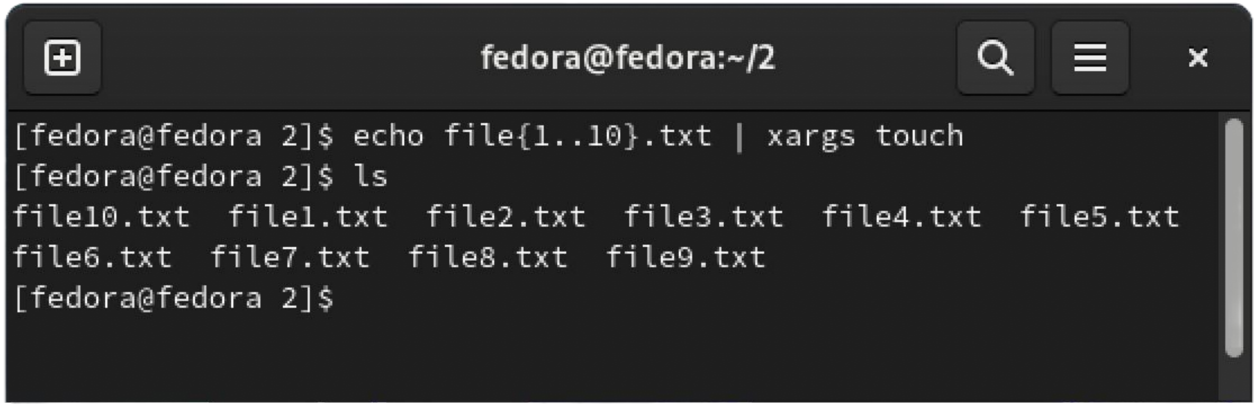
What is a xargs?

Xargs is a command line utility that take in standard input and uses that as arguments for another command. In other words it takes a list of items and builds and executes a new command using those items as parameters.

Red Hat

Practical Understanding

Imagine you have a list of text files you want to create. Instead of manually creating each text file in a file manager. You could generate a list of file names using “echo”and pipe that list into “xargs” with the command touch.



```
[fedora@fedora 2]$ echo file{1..10}.txt | xargs touch
[fedora@fedora 2]$ ls
file10.txt  file1.txt  file2.txt  file3.txt  file4.txt  file5.txt
file6.txt  file7.txt  file8.txt  file9.txt
[fedora@fedora 2]$
```

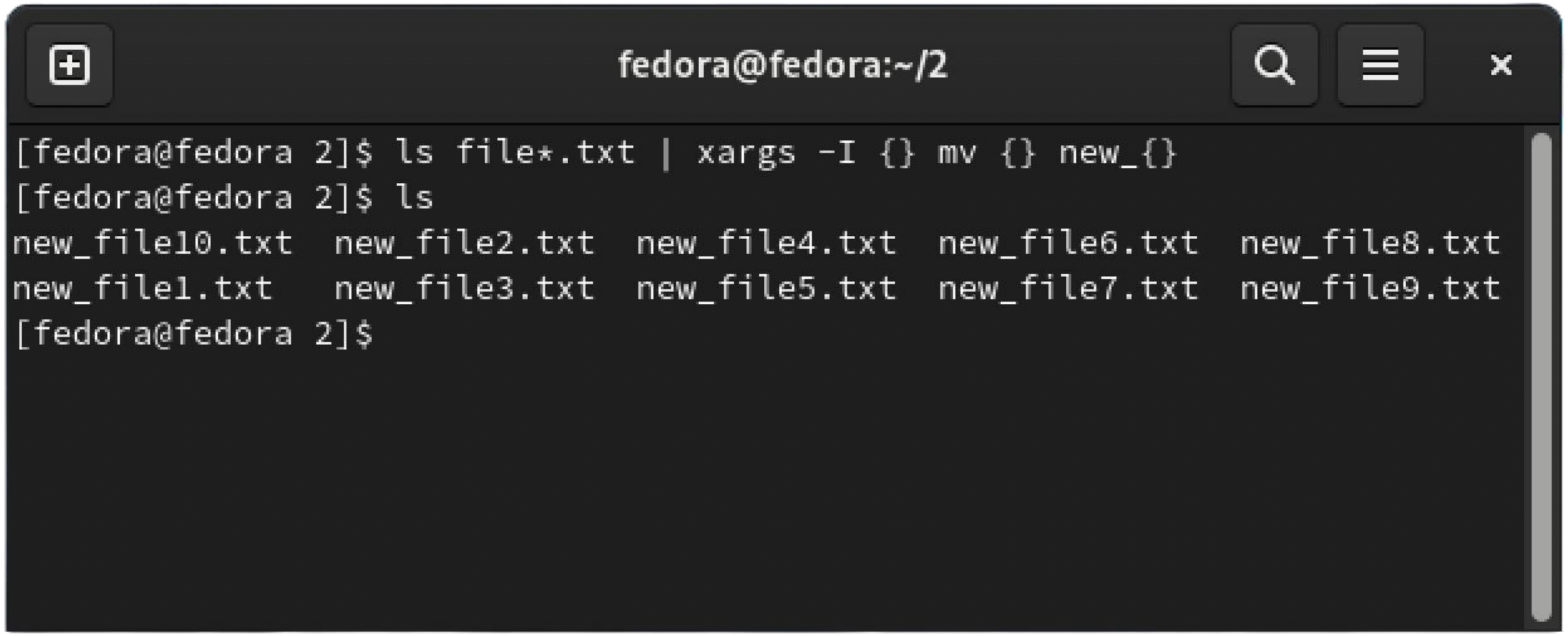
You can also replace “touch” with other commands depending on what you want to do e.g. “rm”

Why use xargs?

- Xargs can manage a large list of items without exceeding the command line length limit.
- It allows you to execute a command on multiple items at once, reducing the need for repetitive command entries.
- You can easily swap out the command used by xargs e.g. “touch”, “rm”, “mv” or “chmod” allowing you to perform various operations on the same set of items.

Example of xargs

Example of xargs



```
[fedora@fedora 2]$ ls file*.txt | xargs -I {} mv {} new_{}
[fedora@fedora 2]$ ls
new_file10.txt  new_file2.txt  new_file4.txt  new_file6.txt  new_file8.txt
new_file1.txt   new_file3.txt  new_file5.txt  new_file7.txt  new_file9.txt
[fedora@fedora 2]$
```

The command begins by listing all the files in the current directory that matches the pattern “file*.txt”. It then passes the list of filenames to “xargs” which processes each line of input individually. The “-I” option tells “xargs” to replace {} with each file name. For each file the command “mv {} new_{} renames the file by adding the prefix “new_” to its original name.

ls file*.txt | xargs -I {} mv {} new_{}

Here is a breakdown of each part of the argument:

- **Ls file*.txt:** List all the files in the current directory that matches the pattern “file*.txt”
- **Xargs: -I {}:** Processes each line of input individuality. The “-I {}” option tells the xargs to replace every instance of {} in the command with each file name.
- **Mv {} new_{}:** For each file name this command renames them by adding the prefix “new_{} to it. Example file1.txt becomes new_file1.txt

Recap

- Introduction to command line chaining
- Understanding Pipes
- Using grep for searching
- Introduction to xargs

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat