

# Redirection and Pipes

Spoken Tutorial Project  
National Mission on Education through ICT  
<http://spoken-tutorial.org>

Script  
Anirban Roy Choudhury  
Narration  
Abhijit Sunil

6 September 2014



# Pre-requisites :

- ▶ I am using Linux OS.



# Pre-requisites :

- ▶ I am using Linux OS.
- ▶ You should know how to get started with the Linux OS.



# Pre-requisites :

- ▶ I am using Linux OS.
- ▶ You should know how to get started with the Linux OS.
- ▶ You should have some basic idea about commands.



# Pre-requisites :

- ▶ I am using Linux OS.
- ▶ You should know how to get started with the Linux OS.
- ▶ You should have some basic idea about commands.
- ▶ If not, please refer to spoken tutorials on <http://spoken-tutorial.org>



# About the Linux commands

- ▶ **Linux is case sensitive.**



# About the Linux commands

- ▶ **Linux is case sensitive.**
- ▶ **All the commands used here are in lowercase, unless mentioned otherwise.**



# Linux commands on Terminal

- ▶ Most of the work that we do in Linux is through a **terminal**.



# Linux commands on Terminal

- ▶ Most of the work that we do in Linux is through a **terminal**.
- ▶ **When we have to execute a command, we generally type through the keyboard.**



# Preliminaries



# Preliminaries

- ▶ **Two important concepts:**  
**Stream.**  
**File descriptor.**



# Stream

## Stream:

- ▶ A Linux shell, like **Bash**, receives input and sends output as sequences or streams of characters.



# Stream

## Stream:

- ▶ A Linux shell, like **Bash**, receives input and sends output as sequences or streams of characters.
- ▶ Each character is independent of the one before it and the one after it.



# Stream

## Stream:

- ▶ A Linux shell, like **Bash**, receives input and sends output as sequences or streams of characters.
- ▶ Each character is independent of the one before it and the one after it.
- ▶ **Accessed using file IO techniques.**



# Stream

## Stream:

- ▶ A Linux shell, like **Bash**, receives input and sends output as sequences or streams of characters.
- ▶ Each character is independent of the one before it and the one after it.
- ▶ Accessed using file IO techniques.
- ▶ **Actual stream of characters may come from or go to a file/keyboard/window.**



# File Descriptor

## File Descriptor -

- ▶ **Every open file of a process is associated with an integer number.**



# File Descriptor

## File Descriptor -

- ▶ Every open file of a process is associated with an integer number.
- ▶ This numeric value is known as the **file descriptor**.



# The three standard streams



# The three standard streams

## stdin

- ▶ It is the standard input stream.



# The three standard streams

## stdin

- ▶ It is the standard input stream.
- ▶ Provides input to commands.



# The three standard streams

## stdin

- ▶ It is the standard input stream.
- ▶ Provides input to commands.
- ▶ It has file descriptor **0**.



# The three standard streams(Contd)

## stdout

- ▶ It is the standard output stream.



# The three standard streams(Contd)

## stdout

- ▶ It is the standard output stream.
- ▶ Displays output from commands.



# The three standard streams(Contd)

## stdout

- ▶ It is the standard output stream.
- ▶ Displays output from commands.
- ▶ It has file descriptor **1**.



# The three standard streams(Contd)

## stderr

- ▶ It is the standard error stream.



# The three standard streams(Contd)

## stderr

- ▶ It is the standard error stream.
- ▶ Displays error output from commands.



# The three standard streams(Contd)

## stderr

- ▶ It is the standard error stream.
- ▶ Displays error output from commands.
- ▶ It has file descriptor **2**.



# More about Streams

- ▶ **Input streams provide input to programs.**



# More about Streams

- ▶ **Input streams provide input to programs.**
- ▶ **By default, it takes from terminal key-strokes.**



# More about Streams

- ▶ **Input streams provide input to programs.**
- ▶ **By default, it takes from terminal key-strokes.**
- ▶ **Output streams print text characters, by default, on the terminal.**



# About the Terminal

- ▶ **The terminal was originally an ASCII typewriter or display terminal.**



# About the Terminal

- ▶ The terminal was originally an ASCII typewriter or display terminal.
- ▶ Now, it is more often a text window on a graphical desktop.



# Redirection

- ▶ We have seen that the 3 streams are connected to some files, by default.



# Redirection

- ▶ We have seen that the 3 streams are connected to some files, by default.
- ▶ But in Linux, we can change this default behaviour.



# Redirection

- ▶ We have seen that the 3 streams are connected to some files, by default.
- ▶ But in Linux, we can change this default behaviour.
- ▶ We can connect these 3 streams to other files.



# Redirection

- ▶ We have seen that the 3 streams are connected to some files, by default.
- ▶ But in Linux, we can change this default behaviour.
- ▶ We can connect these 3 streams to other files.
- ▶ This process is called **Redirection**.



# Redirection– Two ways



# Redirection– Two ways

**n**>

- ▶ Redirects output from file descriptor **n** to a file.



# Redirection– Two ways

**n**>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.



# Redirection– Two ways

**n**>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.
- ▶ If destination file does not exist - it is created.



# Redirection– Two ways

**n**>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.
- ▶ If destination file does not exist - it is created.
- ▶ If destination file exists - the existing contents are lost without any warning.



# Redirection(contd)

**n>>**

- ▶ Redirects output from file descriptor **n** to a file.



# Redirection(contd)

**n**>>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.



# Redirection(contd)

**n**>>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.
- ▶ If destination file does not exist - it is created.



# Redirection(contd)

**n**>>

- ▶ Redirects output from file descriptor **n** to a file.
- ▶ Must have **write** authority to the file.
- ▶ If destination file does not exist - it is created.
- ▶ If destination file exists - the output is appended to the existing file.



# Redirection(contd)

- ▶ **Redirect stdout** `> or >>`  
**(same as)** `1 > or 1 >>`



# Redirection(contd)

- ▶ **Redirect stdout** `> or >>`  
(same as) `1 > or 1 >>`
- ▶ **Redirect stderr** `2 > or 2 >>`



# Redirection(contd)

- ▶ By default, **wc** writes its output to the **stdout**.



# Redirection(contd)

- ▶ By default, **wc** writes its output to the **stdout**.
- ▶ **stdout** is, by default, connected to the **terminal**.



# Redirection(contd)

- ▶ By default, **wc** writes its output to the **stdout**.
- ▶ **stdout** is, by default, connected to the terminal.
- ▶ Hence we see the output in the terminal.



# Redirection(contd)

- ▶ By default, **wc** writes its output to the **stdout**.
- ▶ **stdout** is, by default, connected to the terminal.
- ▶ Hence we see the output in the terminal.
- ▶ If we redirect **stdout** to a file, then the output from **wc** command will be written to that file.



# Redirection(contd)

- ▶ **Redirecting the standard error is done similarly.**



# Redirection(contd)

- ▶ Redirecting the standard error is done similarly.
- ▶ In this case, we need to mention the file descriptor number of the standard error before `>` or `>>`.



# Pipes

- ▶ **Manipulate and connect the different streams simultaneously.**



# Pipes

- ▶ Manipulate and connect the different streams simultaneously.
- ▶ This process is called **Pipelining**.



# Pipes

- ▶ Manipulate and connect the different streams simultaneously.
- ▶ This process is called **Pipelining**.
- ▶ **Pipes are used to create a chain of commands.**



# Pipes

- ▶ Manipulate and connect the different streams simultaneously.
- ▶ This process is called **Pipelining**.
- ▶ Pipes are used to create a chain of commands.
- ▶ **Connects the output of one command to the input of the next command.**



# Pipes

- ▶ It looks like  
`command1 | command2 -option |`  
`command3 -option1 -option2 | command4`



# Acknowledgement

- ▶ Spoken Tutorial Project is a part of Talk to a Teacher Project.
- ▶ Supported by the National Mission on Education through ICT, MHRD, Government of India.
- ▶ More information available at <http://spoken-tutorial.org/NMEICT-Intro>

