Shell Scripting Tutorial

Shell Scripting is an open-source programming language designed to be run by the Unix/Linux shell. Shell Scripting is a program to write a series of commands for the shell to execute. It can combine lengthy and repetitive sequences of commands into a single and simple script that can be stored and executed anytime which, reduces programming efforts.

This Shell Scripting tutorial helps to learn a basic understanding of the Linux/Unix shell scripting program to advanced concepts of Shell Scripting. This Shell Script tutorial designed for beginners and professionals who want to learn What is Shell Scripting? How shell scripting works, types of shell, and more.

What is Shell?

Shell is a UNIX term for an interface between a user and an operating system service. Shell provides users with an interface and accepts human-readable commands into the system and executes those commands which can run automatically and give the program's output in a shell script.

An Operating is made of many components, but its two prime components are -

- Kernel
- Shell

A Kernel is at the nucleus of a computer. It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.

A shell in a Linux operating system takes input from you in the form of commands, processes it, and then gives an output. It is the interface through which a user works on the programs, commands, and scripts. A shell is accessed by a terminal which runs it.

When you run the terminal, the Shell issues **a command prompt (usually $),** where you can type your input, which is then executed when you hit the Enter key. The output or the result is thereafter displayed on the terminal.

The Shell wraps around the delicate interior of an Operating system protecting it from accidental damage. Hence the name **Shell**.

This Unix/Linux Shell Script tutorial helps understand shell scripting basics to advanced levels.

Types of Shell

There are two main shells in Linux:

1. The **Bourne Shell**: The prompt for this shell is $ and its derivatives are listed below:

- POSIX shell also is known as sh
- Korn Shell also knew as sh
- **B**ourne **A**gain **SH**ell also knew as bash (most popular)

2. The C shell: The prompt for this shell is %, and its subcategories are:

- C shell also is known as csh
- Tops C shell also is known as tcsh

We will discuss bash shell based shell scripting in this tutorial.

How to Write Shell Script in Linux/Unix

Shell Scripts are written using text editors. On your Linux system, open a text editor program, open a new file to begin typing a shell script or shell programming, then give the shell permission to execute your shell script and put your script at the location from where the shell can find it.

Let us understand the steps in creating a Shell Script:

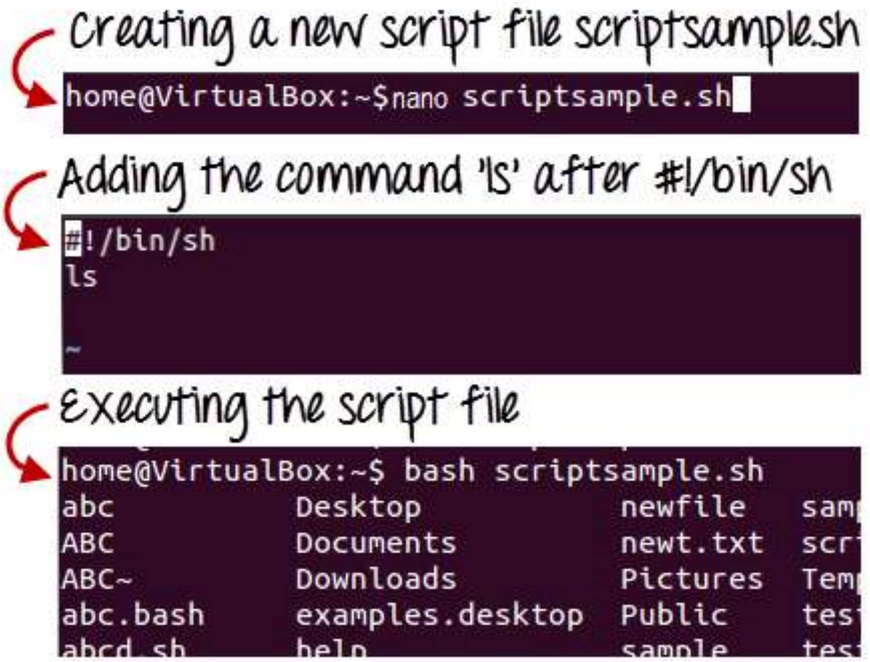
1. **Create a file using** a **nano** editor(or any other editor). Name script file with **extension .sh**
2. **Start** the script with **#! /bin/sh**
3. Write some code.
4. Save the script file as filename.sh
5. For **executing** the script type **bash filename.sh**

"#!" is an operator called shebang which directs the script to the interpreter location. So, if we use"#! /bin/sh" the script gets directed to the bourne-shell.

Let's create a small script -

```
#!/bin/sh
ls
```

Let's see the steps to create Shell Script Programs in Linux/Unix -



Creating a new script file scriptsample.sh

```
home@VirtualBox:~$nano scriptsample.sh
```

Adding the command 'ls' after #!/bin/sh

```
#!/bin/sh
ls

~
```

Executing the script file

```
home@VirtualBox:~$ bash scriptsample.sh
abc           Desktop           newfile   samp
ABC           Documents         newt.txt  scri
ABC~          Downloads         Pictures  Temp
abc.bash      examples.desktop  Public    tes
abcd.sh       help              sample    tes
```

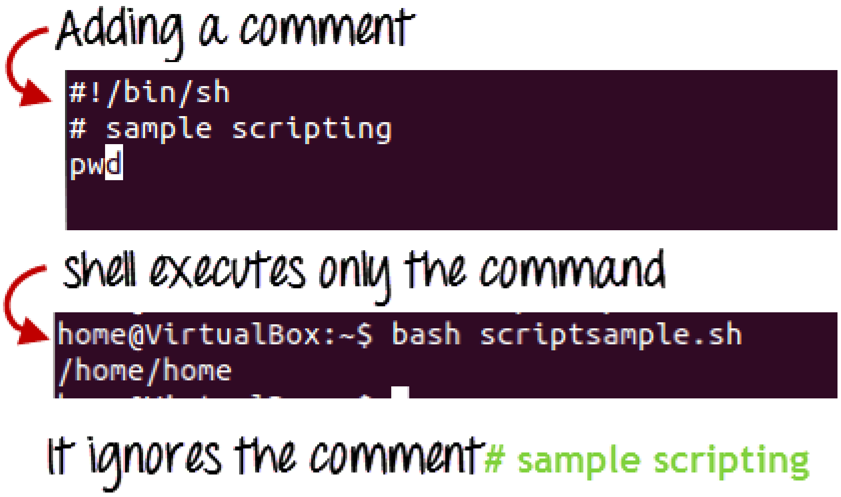
Command 'ls' is executed when we execute the scrip sample.sh file.

Adding shell comments

Commenting is important in any program. In Shell programming, the syntax to add a comment is

```
#comment
```

Let understand this with an example.



Adding a comment

```
#!/bin/sh
# sample scripting
pwd
```

Shell executes only the command

```
home@VirtualBox:~$ bash scriptsample.sh
/home/home
```

It ignores the comment# sample scripting

What are Shell Variables?

As discussed earlier, Variables store data in the form of characters and numbers. Similarly, Shell variables are used to store information and they can by the shell only.

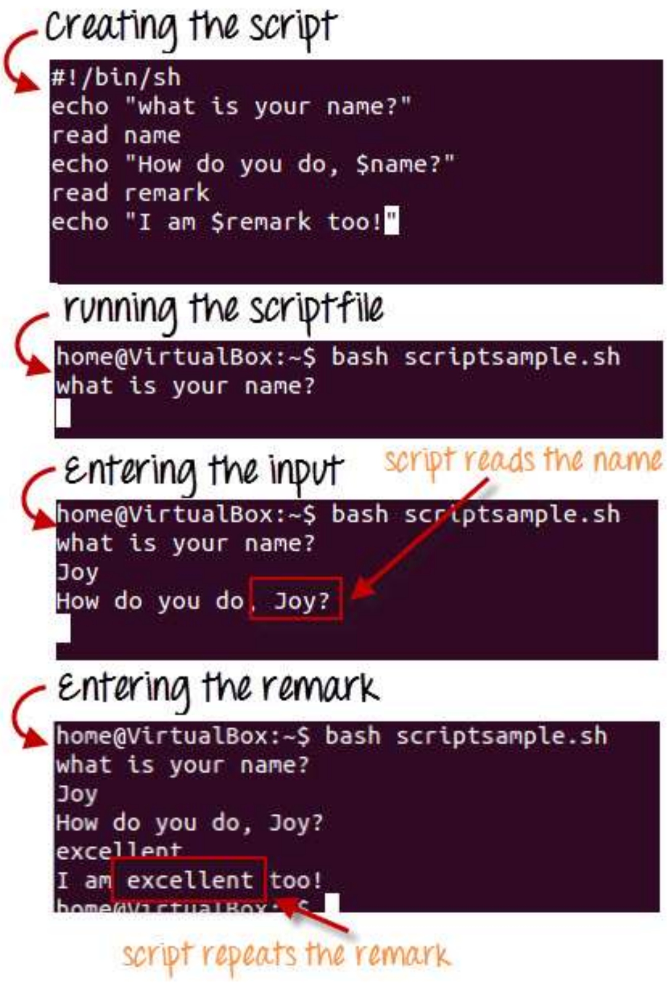
For example, the following creates a shell variable and then prints it:

```
variable ="Hello"
echo $variable
```

Below is a small script which will use a variable.

```
#!/bin/sh
echo "what is your name?"
read name
echo "How do you do, $name?"
read remark
echo "I am $remark too!"
```

Let's understand, the steps to create and execute the script



```
#!/bin/sh
echo "what is your name?"
read name
echo "How do you do, $name?"
read remark
echo "I am $remark too!"
```

```
home@VirtualBox:~$ bash scriptsample.sh
what is your name?
```

```
home@VirtualBox:~$ bash scriptsample.sh
what is your name?
Joy
How do you do, Joy?
```

```
home@VirtualBox:~$ bash scriptsample.sh
what is your name?
Joy
How do you do, Joy?
excellent
I am excellent too!
home@VirtualBox:~$
```

As you see, the program picked the value of the variable 'name' as Joy and 'remark' as excellent.

This is a simple script. You can develop advanced scripts which contain conditional statements, loops, and functions. Shell scripting will make your life easy and Linux administration a breeze.

Summary:

- Kernel is the nucleus of the operating systems, and it communicates between hardware and software
- Shell is a program which interprets user commands through CLI like Terminal
- The Bourne shell and the C shell are the most used shells in Linux
- Linux Shell scripting is writing a series of command for the shell to execute
- Shell variables store the value of a string or a number for the shell to read
- Shell scripting in Linux can help you create complex programs containing conditional statements, loops, and functions
- Basic Shell Scripting Commands in Linux: cat, more, less, head, tail, mkdir, cp, mv, rm, touch, grep, sort, wc, cut and, more.