01.02.2020

Welcome Everyone!!

Introduction to Shell Scripting

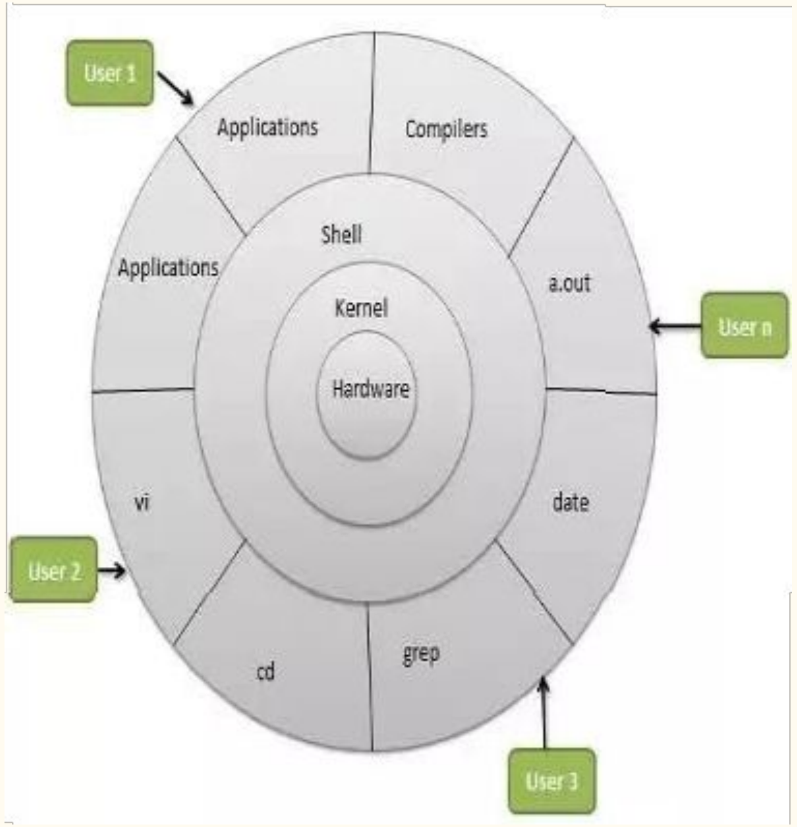
What is a Shell?

An Operating is made of many components,

but its two prime components are -

1.)Kernel

2.)Shell



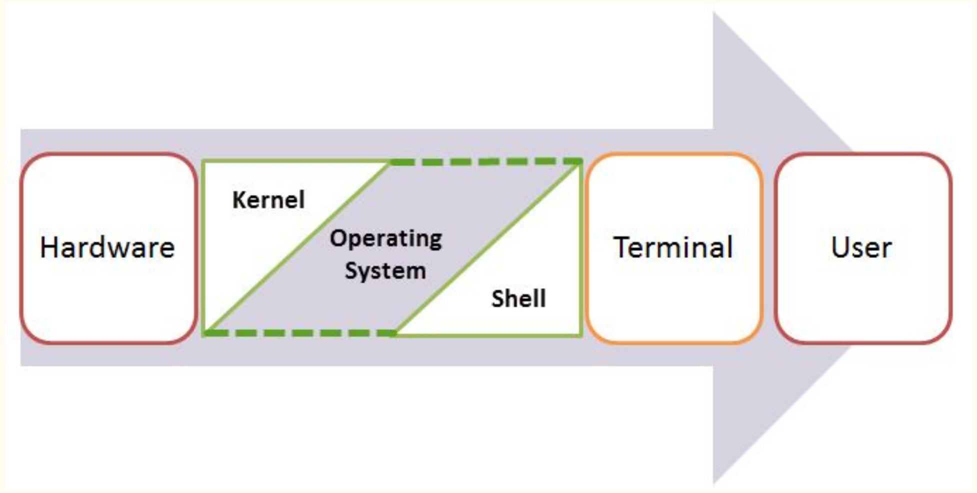
So what is kernel?

Kernel :- The Nucleus of an Computer.

Innermost part of the OS.

Makes the Communication between the hardware and software possible.

The Shell is the Outermost part.



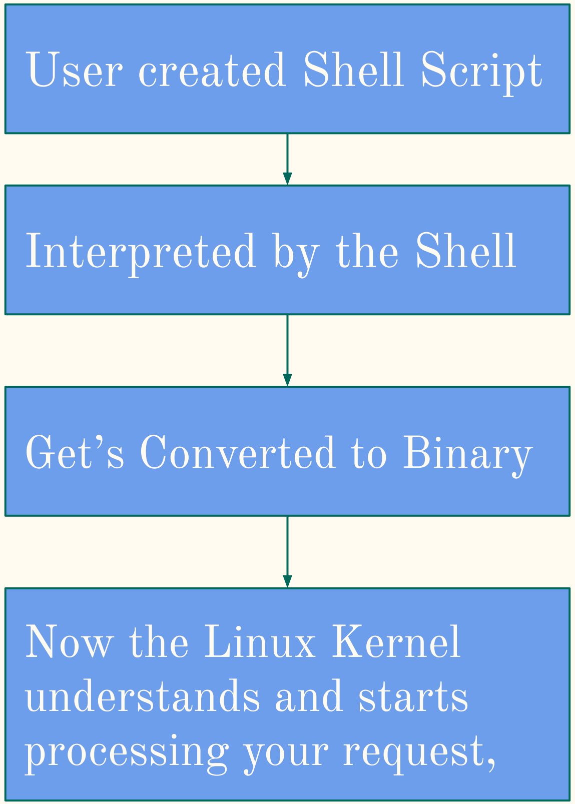
Shell

- It is a Command–Line Interpreter.
- It forms the interface between the Kernel and the User.

Shell Scripting

- Shell scripting is writing a series of command for the shell to execute.
- It can combine lengthy and repetitive sequences of commands into a single and simple script.

Basic Workflow


User created Shell Script
Interpreted by the Shell
Get's Converted to Binary
Now the Linux Kernel understands and starts processing your request,

Why should you use it?

You can use shell scripts to **automate administrative tasks.**

Encapsulate complex configuration details.

Get at the full power of the operating system.

The **ability to combine commands** allows you to **create new commands.**

Kinds of shells

- Bourne Shell > `prompt: $`
- Its derivatives -
  - Bash Shell (BASH)
  - Friendly interactive shell (FISH)
- C Shell > `prompt: %`
- Its derivatives -
  - Z Shell (ZSH)

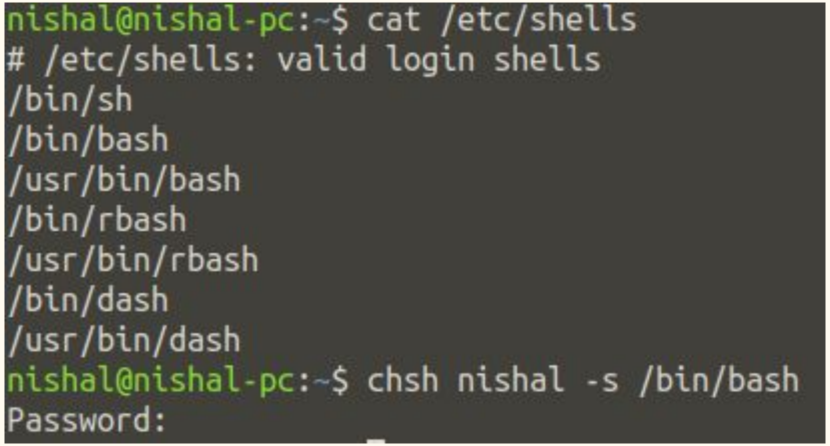
Changing Your Default Shell

To find all available shells in your system type following command:

```
~$ cat /etc/shells
```

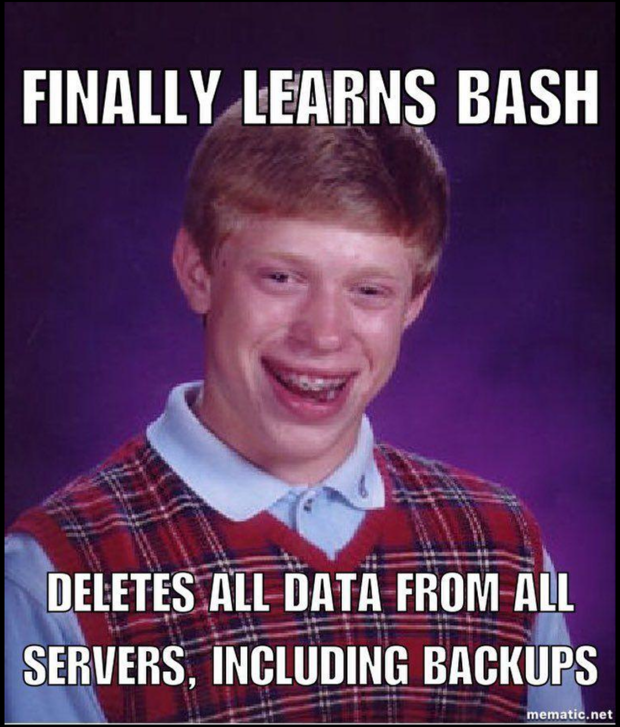
The basic Syntax to change your shell :

```
~$ chsh username -s new_default_shell
```



```
nishal@nishal-pc:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/bin/dash
/usr/bin/dash
nishal@nishal-pc:~$ chsh nishal -s /bin/bash
Password:
```

Time to code!


FINALLY LEARNS BASH
DELETES ALL DATA FROM ALL SERVERS, INCLUDING BACKUPS
mematic.net

Hello

!

Shell code

```
#!/bin/bash
echo "Hello World"
```

C code

```
#include <stdio.h>

int main() {

  printf("Hello, World!");

  return 0;

}
```

Executing your shell script

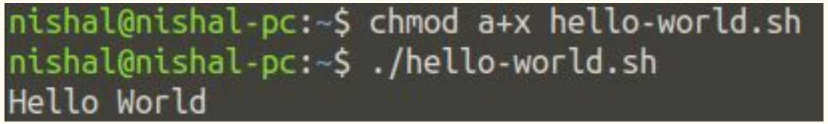
Save your file as hello-world.sh

```
~$ chmod a+x hello-world.sh
```

You can run this using any one of the below two commands.

```
~$ bash hello-world.sh
```

```
~$ ./hello-world.sh
```



```
nishal@nishal-pc:~$ chmod a+x hello-world.sh
nishal@nishal-pc:~$ ./hello-world.sh
Hello World
```

Comments?

Shell code

```
#!/bin/bash

# Single Line Comment

: '
This is a Multi Line
comment
'
```

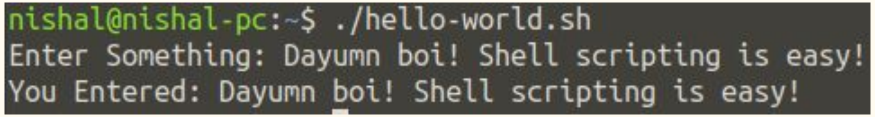
C code

```
#include <stdio.h>

int main() {

  // Single Line Comment

  /*

      Multi-Line

      Comment

  */

}
```

Reading Input

Shell code

```
#!/bin/bash

echo -n "Enter Something:"
read something

echo "You Entered: $something"
```



```
nishal@nishal-pc:~$ ./hello-world.sh
Enter Something: Dayumn boi! Shell scripting is easy!
You Entered: Dayumn boi! Shell scripting is easy!
```

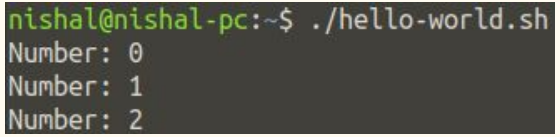
C code

```
#include <stdio.h>

int main() {

     int something;

     scanf(“%d”,&something);

     printf(“%d”,something);

     return 0;

}
```

While Loops

Shell code

```
#!/bin/bash
i=0

while [ $i -le 2 ]
do
echo Number: $i
((i++))
done
```



```
nishal@nishal-pc:~$ ./hello-world.sh
Number: 0
Number: 1
Number: 2
```

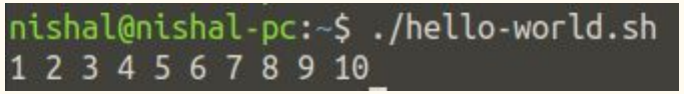
C code

```
#include <stdio.h>

int main() {

      int i=0;

      while(i<=2){

            printf(“Number: %d”,i);

            i++;

      }

}
```

For Loops

Shell code

```
#!/bin/bash

for((counter=1; counter<=10; counter++ ))
do
echo -n "$counter "
done
```



```
nishal@nishal-pc:~$ ./hello-world.sh
1 2 3 4 5 6 7 8 9 10
```

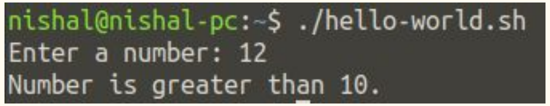
C code

```
#include <stdio.h>

int main() {

     int counter;

     for(counter=1;counter<=10;counter++){

          printf(“%d ”,counter);

     }

     return 0;

}
```

Conditionals - if’s

Shell code

```
#!/bin/bash

echo -n "Enter a number: "
read num

if [[ $num -gt 10 ]]
then
echo "Number is greater than 10."
elif [[ $num -eq 10 ]]
then
echo "Number is equal to 10."
else
echo "Number is less than 10."
fi
```



```
nishal@nishal-pc:~$ ./hello-world.sh
Enter a number: 12
Number is greater than 10.
```

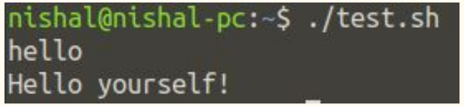
C code

```
int num;

scanf(“%d”,&num);

if(num>10){

        printf(“Number is greater than 10.”);

}else if(num==10){

        printf(“Number is equal to 10.”);

}else{

        printf(“Number is less than 10.”);

}
```

Switch

Shell code

```
#!/bin/bash

read INPUT_STRING
case $INPUT_STRING in
     hello)
          echo "Hello yourself!"
          ;;
     bye)
          echo "See you again!"
          ;;
     *)
          echo "Sorry, I don't understand"
          ;;
esac
```



```
nishal@nishal-pc:~$ ./test.sh
hello
Hello yourself!
```

C code

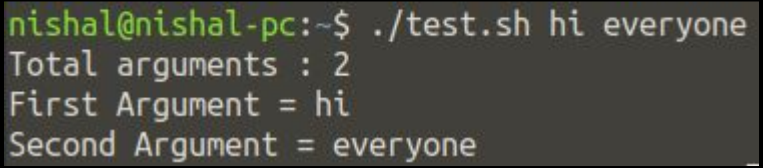
```
int num;
scanf("%d",&num);
switch(num){
     case 0:
          printf("Hello yourself!");
          break;
     case 1:
          printf("See you again!");
          break;
     }
     default:
          printf("Sorry, I don't understand");
          break;
}
```

Command Line arguments

Shell code

```
#!/bin/bash
echo "Total arguments : $#"
echo "First Argument = $1"
echo "Second Argument = $2"
```

Output



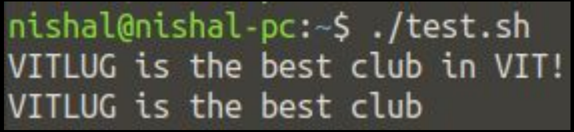
```
nishal@nishal-pc:~$ ./test.sh hi everyone
Total arguments : 2
First Argument = hi
Second Argument = everyone
```

String operations

Shell code

```
#!/bin/bash

# Concatenation
string1="VIT"
string2="LUG"
ccstring=$string1$string2
echo "$ccstring is the best club in
$string1!"

# Slicing
str="VITLUG is the best club in VIT!"
subStr=${str:0:23}
echo $subStr
```

Output



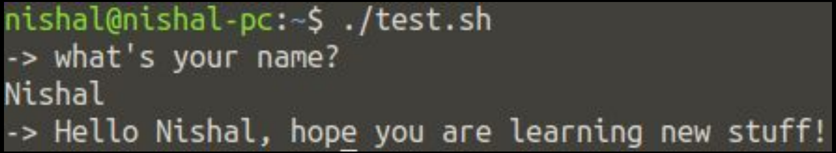
```
nishal@nishal-pc:~$ ./test.sh
VITLUG is the best club in VIT!
VITLUG is the best club
```

Functions

Shell code

```
#!/bin/bash

function Greet() {

str="Hello $name, hope you are learning
new stuff!"
echo $str
}

echo "-> what's your name?"
read name

val=$(Greet)
echo -e "-> $val"
```

Output



```
nishal@nishal-pc:~$ ./test.sh
-> what's your name?
Nishal
-> Hello Nishal, hope you are learning new stuff!
```

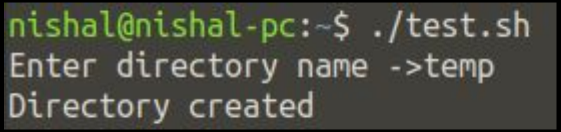
We are done with the “basic stuff”...

Creating Directories from Bash Scripts

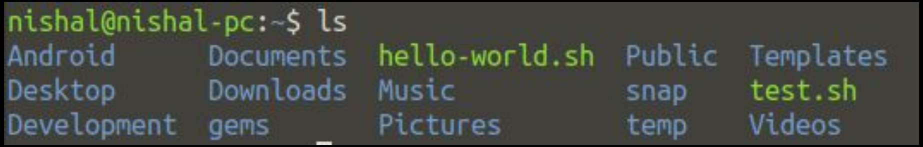
Shell code

```
#!/bin/bash
echo -n "Enter directory name ->"
read dir
if [ -d "$dir" ]
then
echo "Directory exists"
else
`mkdir $dir`
echo "Directory created"
fi
```

Output



```
nishal@nishal-pc:~$ ./test.sh
Enter directory name ->temp
Directory created
```



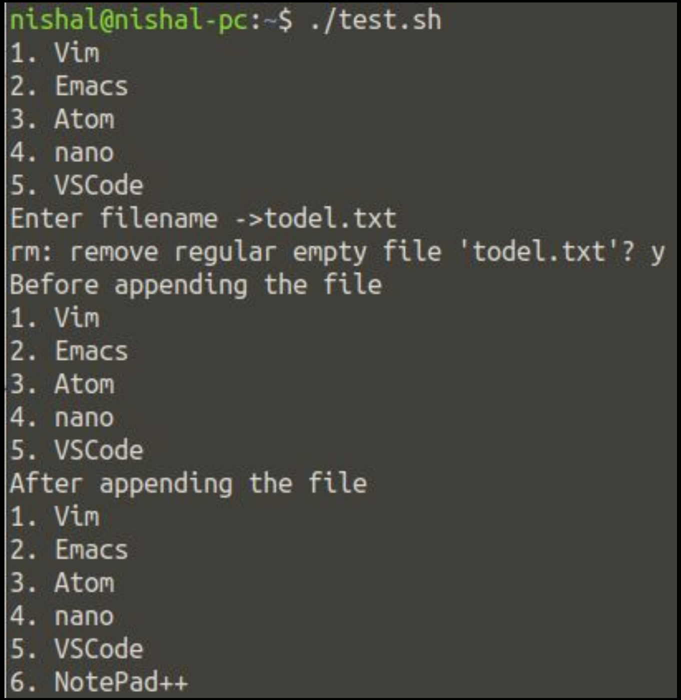
```
nishal@nishal-pc:~$ ls
Android      Documents  hello-world.sh  Public  Templates
Desktop      Downloads  Music           snap    test.sh
Development  gems       Pictures        temp    Videos
```

File Operations

Shell code

```bash
#!/bin/bash

# Reading a file
file='editors.txt'
while read line; do
echo $line
done < $file

# Deleting a file
echo -n "Enter filename ->"
read name
rm -i $name

# Appending to a file
echo "Before appending the file"
cat editors.txt
echo "6. NotePad++" >> editors.txt
echo "After appending the file"
cat editors.txt
```

Output



```
nishal@nishal-pc:~$ ./test.sh
1. Vim
2. Emacs
3. Atom
4. nano
5. VSCode
Enter filename ->todel.txt
rm: remove regular empty file 'todel.txt'? y
Before appending the file
1. Vim
2. Emacs
3. Atom
4. nano
5. VSCode
After appending the file
1. Vim
2. Emacs
3. Atom
4. nano
5. VSCode
6. NotePad++
```

Sleep & Wait

Shell code

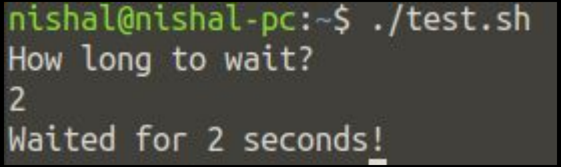
The sleep command allows your shell script to pause between instructions.

```
#!/bin/bash
echo "How long to wait?"
read time
sleep $time
echo "Waited for $time seconds!"
```

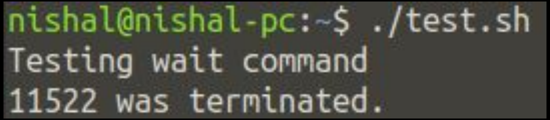
The wait command is used for pausing system processes from Linux bash scripts.

```
#!/bin/bash
echo "Testing wait command"
sleep 5 &
pid=$!
kill $pid
wait $pid
echo $pid was terminated.
```

Output



```
nishal@nishal-pc:~$ ./test.sh
How long to wait?
2
Waited for 2 seconds!
```



```
nishal@nishal-pc:~$ ./test.sh
Testing wait command
11522 was terminated.
```

Simple Applications

Send Mails from Shell Scripts

```
#!/bin/bash
recipient=”admin@example.com”
subject=”Greetings”
message=”Welcome to VITLUG SESSION 1”
`mail -s $subject $recipient <<< $message`
```

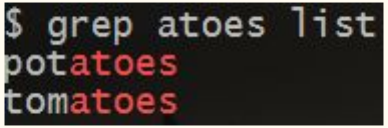
Adding Batch Extensions

```bash
#!/bin/bash
dir=$1
for file in `ls $1/*`
do
mv $file $file.txt
done
```

Removing Duplicate Lines from Files

```
#! /bin/sh
echo -n "Enter Filename-> "
read filename
if [ -f "$filename" ]; then
sort $filename | uniq | tee sorted.txt
else
echo "No $filename in $pwd...try again"
fi
exit 0
```

Tip - **grep** is another very important command which is used to search for a particular text within the file and generate the output for you related to the pattern being matched.



```
$ grep atoes list
potatoes
tomatoes
```

System Maintenance

```
#!/bin/bash

echo -e "\n$(date "+%d-%m-%Y --- %T") --- Starting work\n"

apt-get update
apt-get -y upgrade

apt-get -y autoremove
apt-get autoclean

echo -e "\n$(date "+%T") \t Script Terminated"
```

Thank You