Embedded Systems

Week 4: Linux Fundamentals III

Fenerbahçe University

Professor & TAs

Prof: Dr. Vecdi Emre Levent

Office: 311

Email: emre.levent@fbu.edu.tr

TA: Arş. Gör. Uğur Özbalkan

Office: 311

Email: ugur.ozbalkan@fbu.edu.tr

Fundamental Linux – Bash Scripts

Bash Scripts

Bash is a command language interpreter. It is widely available on various operating systems and is a default command interpreter on most GNU/Linux systems.

To view current shell type:

echo $SHELL

```
emre@emre-VirtualBox:~/Documents$ echo $SHELL
/bin/bash
```

Fundamental Linux – Bash Scripts

Bash Scripts

Create a file named test.sh, helloworld

test.sh

```
#!/bin/bash

echo "Hello World"
```

```
emre@emre-VirtualBox:~/Documents$ sh test.sh
Hello World
```

or

```
emre@emre-VirtualBox:~/Documents$ chmod u+x test.sh
emre@emre-VirtualBox:~/Documents$ ./test.sh
Hello World
```

Fundamental Linux – Bash Scripts

Bash Scripts

Variables and printing

test.sh

```
greeting="Welcome"
user=$(whoami)
day=$(date +%A)

echo "$greeting back $user! Today is $day, which is the best day of the entire week!"
echo "Your Bash shell version is: $BASH_VERSION. Enjoy!"
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Welcome back emre! Today is Cumartesi, which is the best day of the entire week!
Your Bash shell version is: 4.4.20(1)-release. Enjoy!
```

Fundamental Linux – Bash Scripts

Bash Scripts

Read user input

test.sh

```
#!/bin/sh

echo "What is your name?"
read PERSON
echo "Hello, $PERSON"
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
What is your name?
deneme
Hello, deneme
```

Fundamental Linux – Bash Scripts

Bash Scripts

int and string variables

test.sh

```
#!/bin/sh

year=2012
comp_name=jtp

echo $year
echo $comp_name
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
2012
jtp
```

Fundamental Linux – Bash Scripts

Bash Scripts

echo variables

test.sh

```
#!/bin/sh

echo $HOME # Home Directory
echo $PWD # current working directory
echo $BASH # Bash shell name
echo $BASH_VERSION # Bash shell Version
echo $LOGNAME # Name of the Login User
echo $OSTYPE # Type of OS
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
/home/emre
/home/emre/Documents
/bin/sh
4.4.20(1)-release
emre
linux-gnu
```

Fundamental Linux – Bash Scripts

Bash Scripts

Getting arguments

test.sh

```
#!/bin/sh

args=("$@")

echo ${args[0]} ${args[1]} ${args[2]} ${args[3]}
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh testa b c d
testa b c d
```

Fundamental Linux – Bash Scripts

Bash Scripts

Calling a command, example ls

test.sh

```
#!/bin/sh

lsResult=$(ls)

echo "My files are:" $lsResult
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
My files are: test.sh
```

Fundamental Linux – Bash Scripts

Bash Scripts

Printing date

test.sh

```
#!/bin/bash

d=`date +%m-%d-%Y`

echo "Date in format MM-DD-YYYY"
echo $d #MM-DD-YYYY
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Date in format MM-DD-YYYY
01-22-2022
```

Fundamental Linux – Bash Scripts

Bash Scripts

Sleep

test.sh

```
#!/bin/bash

date +"%H:%M:%S"
echo "wait for 9 seconds"
sleep 9s
date +"%H:%M:%S"
echo "Task Completed"
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
19:22:21
wait for 9 seconds
19:22:30
Task Completed
```

Fundamental Linux – Bash Scripts

Bash Scripts

There are 11 arithmetic operators supported by bash shell

Operator	Description	Examples
+	Addition, measures addition of numbers (operands)	$((10 + 3)), result=13
-	Substraction, measures subtraction of second operand from first	$((10 - 3)), result=7
*	Multiplication, measures the multiplication of operands.	$((10 * 3)), result=30
/	Division, measures the division of first operand by second operand and and return quotient.	$((10 / 3)), result=3
**	Exponentiation, measures the result of second operand raised to the power of first operand.	$((10 ** 3)), result=1000
%	Modulo, measures remainder when the first operand is divided by second operand.	$((10 % 3)), result=1

Fundamental Linux – Bash Scripts

Bash Scripts

There are 11 arithmetic operators supported by bash shell

Operator	Description	Examples
+=	Increment Variable by Constant- used to increment the value of first operand by the constant provided.	x=10 let "x += 3" echo $x result=13
-=	Decrement Variable by Constant- used to decrement the value of first operand by the constant provided.	x=10 let "x -= 3" echo $x result=7
*=	Multiply Variable by Constant- used to multiply the value of the first operand by the constant provided.	x=10 let "x *= 3" echo $x result=30
/=	Divide Variable by Constant- used to calculate the value of (variable / constant) and store the result back to variable.	x=10 let "10 /= 3" echo $x result=3
%=	Remainder of Dividing Variable by Constant- used to calculate the value of (variable % constant) and store the result back to variable.	x=10 let "10 %= 3" echo $x result=1

Fundamental Linux – Bash Scripts

Bash Scripts

Arithmetic operators

test.sh

```
#!/bin/sh

Num1=10
Num2=3
((Sum=Num1+Num2))
echo "Sum = $Sum"
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Sum = 13
```

Fundamental Linux – Bash Scripts

Bash Scripts

Arithmetic operators

test.sh

```
#!/bin/sh

x=8
y=2
echo "x=8, y=2"
echo "Addition of x & y"
echo $(( $x + $y ))
echo "Subtraction of x & y"
echo $(( $x - $y ))
echo "Multiplication of x & y"
echo $(( $x * $y ))
echo "Division of x by y"
echo $(( $x / $y ))
```

test.sh (cont)

```
echo "Exponentiation of x,y"
echo $(( $x ** $y ))
echo "Modular Division of x,y"
echo $(( $x % $y ))
echo "Incrementing x by 5, then x= "
(( x += 5 ))
echo $x
echo "Decrementing x by 5, then x= "
(( x -= 5 ))
echo $x
echo "Multiply of x by 5, then x="
(( x *= 5 ))
echo $x
echo "Dividing x by 5, x= "
(( x /= 5 ))
echo $x
echo "Remainder of Dividing x by 5, x="
(( x %= 5 ))
echo $x
```

```
x=8, y=2
Addition of x & y
10
Subtraction of x & y
6
Multiplication of x & y
16
Division of x by y
4
Exponentiation of x,y
64
Modular Division of x,y
0
Incrementing x by 5, then x=
13
Decrementing x by 5, then x=
8
Multiply of x by 5, then x=
40
Dividing x by 5, x=
8
Remainder of Dividing x by 5, x=
3
```

Fundamental Linux – Bash Scripts

Bash Scripts

if statements

test.sh

```
#!/bin/bash

read -p " Enter number : " number

if [ $number -gt 125 ]
then
echo "Value is greater than 125"
fi
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
 Enter number : 130
Value is greater than 125
```

Fundamental Linux – Bash Scripts

Bash Scripts

if statements

test.sh

```
#!/bin/bash

# TRUE && TRUE
if [ 8 -gt 6 ] && [ 10 -eq 10 ];
then
echo "Conditions are true"
fi

# TRUE && FALSE
if [ "mylife" == "mylife" ] && [ 3 -gt 10 ];
then
echo "Conditions are false"
fi
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Conditions are true
```

Fundamental Linux – Bash Scripts

Bash Scripts

Nested if statements

test.sh

```
#!/bin/bash

if [ $1 -gt 50 ]
then
  echo "Number is greater than 50."

  if (( $1 % 2 == 0 ))
  then
    echo "and it is an even number."
  fi
fi
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh 52
Number is greater than 50.
and it is an even number.
```

Fundamental Linux – Bash Scripts

Bash Scripts

if else statements

test.sh

```
#!/bin/bash

#when the condition is true
if [ 10 -gt 3 ];
then
  echo "10 is greater than 3."
else
  echo "10 is not greater than 3."
fi

#when the condition is false
if [ 3 -gt 10 ];
then
  echo "3 is greater than 10."
else
  echo "3 is not greater than 10."
fi
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
10 is greater than 3.
3 is not greater than 10.
```

Fundamental Linux – Bash Scripts

Bash Scripts

Elseif statements

test.sh

```
#!/bin/bash

read -p "Enter a number of quantity:" num

if [ $num -gt 100 ];
then
echo "Eligible for 10% discount"
elif [ $num -lt 100 ];
then
echo "Eligible for 5% discount"
else
echo "Lucky Draw Winner"
echo "Eligible to get the item for free"
fi
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Enter a number of quantity:100
Lucky Draw Winner
Eligible to get the item for free
```

Fundamental Linux – Bash Scripts

Bash Scripts

Case statements

test.sh

```
#!/bin/bash

echo "Which Operating System are you using?"
echo "Windows, Android, Chrome, Linux, Others?"
read -p "Type your OS Name:" OS

case $OS in
   Windows|windows)
      echo "That's common. You should try something new."
      echo
      ;;
   Android|android)
      echo "This is my favorite. It has lots of applications."
      echo
      ;; *)
      echo "Sounds interesting. I will try that."
      echo
      ;;
esac
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Which Operating System are you using?
Windows, Android, Chrome, Linux, Others?
Type your OS Name:windows
Sounds interesting. I will try that.
```

Fundamental Linux – Bash Scripts

Bash Scripts

For loops

test.sh

```
#!/bin/bash

for num in {1..10}
do
echo $num
done

echo "Series of numbers from 1 to 10."
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
1
2
3
4
5
6
7
8
9
10
Series of numbers from 1 to 10.
```

Fundamental Linux – Bash Scripts

Bash Scripts

For loops

test.sh

```
#!/bin/bash

for num in {1..10..2}
do
echo $num
done
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
1
3
5
7
9
```

Fundamental Linux – Bash Scripts

Bash Scripts

For loops

test.sh

```
#!/bin/bash

#For Loop to Read a Range with Decrement

for num in {10..0..3}
do
echo $num
done
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
10
7
4
1
```

Fundamental Linux – Bash Scripts

Bash Scripts

For loops

test.sh

```
#!/bin/bash

for table in {2..100..2}
do
echo $table
if [ $table == 20 ]; then
break
fi
done
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
2
4
6
8
10
12
14
16
18
20
```

Fundamental Linux – Bash Scripts

Bash Scripts

While

test.sh

```
#!/bin/bash
#While loop example in C style

i=1
while((i <= 10))
do
echo $i;
((i=i+1))
done
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
1
2
3
4
5
6
7
8
9
10
```

Fundamental Linux – Bash Scripts

Bash Scripts

Functions

test.sh

```
#!/bin/bash

function denemeFnc {
   echo 'Merhaba'
}

denemeFnc

echo 'Test';
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Merhaba
Test
```

Fundamental Linux – Bash Scripts

Bash Scripts

Functions

test.sh

```
#!/bin/bash
function_arguments()
   {
   echo $1
   echo $2
   echo $3
   echo $4
   echo $5
   }

#Calling function_arguments
function_arguments
"Test""ABC""XTZ""XXX""Deneme."
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
TestABCXTZXXXDeneme.
```

Fundamental Linux – Bash Scripts

Bash Scripts

File read

test.sh

```
#!/bin/bash

value=`cat deneme.txt`
echo "$value"
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
Merhaba
Test
```

Fundamental Linux – Bash Scripts

Bash Scripts

File write

test.sh

```
#!/bin/bash
output=output_file.txt
ls > $output
```

```
emre@emre-VirtualBox:~/Documents$ ./test.sh
emre@emre-VirtualBox:~/Documents$ ls
deneme.txt  output_file.txt  test.sh
emre@emre-VirtualBox:~/Documents$ more output_file.txt
deneme.txt
output_file.txt
test.sh
```