Bash scripting

03 May 2021

00:08

The GNU Bourne-Again shell(bash) is powerfull work envirment and scripting engin.

**1.1 Intro to bash Scripting**

Begin with simple "hello world" script

-->cat ./hello-world.sh

-->#!/bin/bash

-->echo "HELLO WORLD"

(#!=is commonly known as the shebang. And is ignored by the bash interpreter.)

(#=is used to add a comment, so all text that follows it is ignored.

**1.2 Variables**

Variables are named places to temporarily store data. We can set a variable, which assign a value to it, or read a variable which will "expand" or "resolve" it to its stored value.

The easiest method is to set variables directly with a simple **name-value** declaration.

-->name=bob (set a simple variable)

To run a variable use **$** character before variable name.

-->echo $name

You can call many variable at a same time

-->age=5

-->echo $name $age

To run script

-->#!/bin/bash

-->NAME="bob"

-->echo "My name is $NAME"

-->#!/bin/bash

-->NAME="bob"

-->SPORT="bash"

-->echo "the most popular sport is ${SPORT}ball"

Variable how to be start = (start with string)

**1.3 User Input & Comments**

User input

-->#!/bin/bash

-->read -p "enter your name: " NAME

-->echo "your name is $NAME"

-p=(promt) (NAME=variable)

#user Input

-->#!/bin/bash

-->echo "enter your full name: "

-->read FNAME LNAME

-->echo "your first name is $FNAME LNAME $

-->#!/bin/bash

-->echo "enter your full name and age: "

-->read FNAME AGE

-->echo "your name is $FNAME and you are $LNAME years old "



Also do like this

**1.4 IF & if/else**

Conditional statements are use for decision making when you are creating script or program.

**If Syntex**

-->#!/bin/bash

-->if [condition];

-->then

--> condition

-->fi

**Else syntex**

-->#!/bin/bash

-->if [condition];

-->then

--> condition

-->else

--> condition

-->fi

**IF statement**

-->#!/bin/bash

-->NAME="bob"

-->if [ "$NAME" = "bob" ];

-->then

--> echo "access granted"

-->fi

**If else statement**

-->#!/bin/bash

-->read -p "enter your username: " username

-->if [ "$username" = "bob" ];

-->then

--> echo "access granted"

-->else

--> echo "access denied"

-->fi

Also write like this

-->#!/bin/bash

-->echo "enter your username"

-->read NAME

-->if [ "NAME" = "elliot" ];

-->then

--> echo "welcome to the jungle"

-->else

--> echo "username not valid"

-->fi

**Script =Test file is exit or not**

-->man test (test= command)

**Check directory exist or not**

-->#!/bin/bash

-->if [ -d /usr/share/python ]; (-d=use for directory)

-->then

--> echo "yes it does"

-->else

--> echo "no it does not"

-->fi

**Check file is exist or not**

-->#!/bin/bash

-->if [ -e /etc/passwd ]; (-d=use for directory)

-->then

--> echo "yes it does"

-->else

--> echo "no it does not"

-->fi

**1.5 for loops**

Syntex

For VARIBALENAME in $(); do

commands

done

**Simple script**

-->#!/bin/bash

-->for NAMES in $(cat list.txt); do (make a file list.txt and enter some names on it)

--> echo "the names are: $NAMES"

-->done

**1.6 Ping Sweep script**

-->#!/bin/bash

-->echo "enter the subnet"

-->read SUBNET

-->for IP in $(seq 1 254); do (sequence= use characters, keywords and numbering)

--> ping -c 2 $SUBNET.$IP

-->done

**1.7 Creating a Password Generator**

We will use tool or library called **openssl**

-->#!/bin/bash

-->echo "this is a simple password generator"

-->echo "enter the length of the password: "

-->Read PASS\_LENTH

-->for p in $(seq 1); (seq1= the no off the password it will give)

-->Do

--> openssl rand --bash64 48 | cut -c1-$PASS\_LENTH

-->done

(openssl rand = to generate random characters)

(--bash 64 48 = base64 have 48 characters just like alphabet have 26)

( cut -c1 = cutting from the first column or the first letters all the way to the length of the password which user assigned)

You can add seq no

--> for p in $(seq 1 5); (it will give you 5 password)

**1.8 Functions**

Function is a peace of code or section of a code that’s perform a specific task.

#syntex

-->#!/bin/bash

-->function functionname() {

#all code write here

-->}

#test that shodow file exist ot not)

-->Function test\_shadow() {

-->If [ -e /etc/shadow ];

-->then

--> Echo "yes it is"

-->else

--> Echo "fine done not exist"

-->fi

-->}

-->test\_shadow (you need to type fucntion name outside to execute the function)

You can also create another function in same file

-->Function test\_shadow() {

-->If [ -e /etc/shadow ];

-->then

--> Echo "yes it is"

-->else

--> Echo "fine done not exist"

-->fi

-->}

-->Function test\_passwd() {

-->If [ -e /etc/passwd ];

-->then

--> Echo "yes it is"

-->else

--> Echo "fine done not exist"

-->fi

-->}

-->test\_shadow

-->test\_passwd

We can call function within a another function

Type **test\_passwd** in **fi and }** between in shadow function

**1.9 file Encrypter / decrypter**

-->#!/bin/bash

-->echo "This is a simple file encrypter/decrypter"

-->echo "Please choose what you want to do"

-->choice="Encrypt Decrypt"

-->select option in $choice; do

--> if [ $REPLY = 1 ];

-->then

--> echo "Please enter the filename you want to encrypt"

--> read file;

--> gpg -c $file (gpg=GNU Privacy Guard tools)

--> echo "The file has been encrypted"

-->else

--> echo "Please enter the filename you want to decrypt"

--> read file2;

--> gpg -d $file2

--> echo "The file has been decrypted"

-->fi

-->done