Arrays in shell programming

1. Declaring an array and assigning values

#! /bin/bash

Unix[0]='Debian'

Unix[1]='Red hat'

Unix[2]='Ubuntu'

Unix[3]='Suse'

echo ${Unix[1]}

$./arraymanip.sh

2. #/bin/sh

a[0]=12

a[1]=34

a[2]=55

echo ${a[2]}

3 . another method of creating an array

arr=(one two three)

echo ${arr[0]

4. Initializing array during declaration itself . Instead of initializing an each element of an array separately, you can declare and initialize an array by specifying the list of elements (separated by white space) with in a curly braces.

Syntax:

declare -a arrayname=(element1 element2 element3)

#!/bin/sh

declare -a arname=('Unix' 'Red hat' 'Suse')

echo ${arname[0]}

echo ${arname[1]}

declare -a declares an array and all the elements in the parentheses are the elements of an array.

5 . Print the whole bash array

#!/bin/sh

declare -a arname=('Unix' 'Red hat' 'Suse')

echo ${arname[@]}

6. Length of the bash array

We can get the length of an array using the special parameter called $#.

${#arrayname[@]} gives you the length of the array.

7 . #/bin/sh

declare -a arname=('Unix' 'Red hat' 'Suse')

echo ${arname[@]}

echo ${#arname[@]} # gives the length of the array

echo ${#arname} # gives the number of characters in the first element in the array

echo ${#arname[1]}

### 8. Length of the nth Element in an Array

${#arrayname[n]} should give the length of the nth element in an array.

#! /bin/bash

Unix[0]='Debian'

Unix[1]='Red hat'

Unix[2]='Ubuntu'

Unix[3]='Suse'

echo ${#Unix[3]} # length of the element located at index 3 i.e Suse

9. Extraction of offset and length for an array

This example shows the way to extract 2 elements starting from the position 3 from an array called Unix

declare -a arname=('Unix' 'Red hat' 'Suse' 'Debian' 'Ubuntu')

echo ${arname[@]}

echo ${#arname[@]} # gives the length of the array

echo ${#arname} # gives the number of characters in the first element in the array

echo ${#arname[1]}

echo ${arname[@]:3:2}

output: 5

4

7

Debian Ubuntu

10 . Extraction of offset and length for a particular element in array

To extract only first four elements from an array element . For example, Ubuntu which is located at the second index of an array, you can use offset and length for a particular element of an array.

Example extracts the first four characters from the 2nd indexed element of an array

declare -a arname=('Unix' 'Red hat' 'Suse' 'Debian' 'Ubuntu')

echo ${arname[@]}

echo ${#arname[@]} # gives the length of the array

echo ${#arname} # gives the number of characters in the first element in the array

echo ${#arname[1]}

echo ${arname[@]:3:2}

echo ${arname[2]:0:3}

### 11. Search and Replace in an array elements

The following example, searches for Ubuntu in an array elements, and replace the same with the word ‘SCO Unix’.

#!/bin/bash

Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora' 'UTS' 'OpenLinux');

echo ${Unix[@]/Ubuntu/SCO Unix}

$./arraymanip.sh

Debian Red hat SCO Unix Suse Fedora UTS OpenLinux

In this example, it replaces the element in the 2nd index ‘Ubuntu’ with ‘SCO Unix’. But this example will not permanently replace the array content.

12 . Add an element to an existing array

#/bin/sh

declare -a arname=('Unix' 'Red hat' 'Suse' 'Debian' 'Ubuntu' 'Fedora' 'UTS')

arname=("${arname[@]}" "AIX" "HP-UNIX")

echo ${arname[7]}

In the array called Unix, the elements ‘AIX’ and ‘HP-UX’ are added in 7th and 8th index respectively .

13. Remove an element from the array

unset is used to remove an element from an array.unset will have the same effect as assigning null to an element

declare -a arname=('Unix' 'Red hat' 'Suse' 'Debian' 'Ubuntu' 'Fedora' 'UTS')

arname=("${arname[@]}" "AIX" "HP-UNIX")

echo ${arname[7]}

unset arname[2]

echo ${arname[2]}

The above script will just print null which is the value available in the 2nd index. The following example shows one of the way to remove an element completely from an array.

14. Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora' 'UTS' 'OpenLinux');

pos=3

Unix=(${Unix[@]:0:$pos} ${Unix[@]:$(($pos + 1))})

echo ${Unix[@]}

$./arraymanip.sh

Debian Red hat Ubuntu Fedora UTS OpenLinux

In this example, ${Unix[@]:0:$pos} will give you 3 elements starting from 0th index i.e 0,1,2 and ${Unix[@]:4} will give the elements from 4th index to the last index.

### 15. Remove Bash Array Elements using Patterns

In the search condition you can give the patterns, and stores the remaining element to an another array as shown below.

#!/bin/bash

declare -a Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora');

declare -a patter=( ${Unix[@]/Red\*/} )

echo ${patter[@]}

$ ./arraymanip.sh

Debian Ubuntu Suse Fedora

### 16 . Copying an Array

Expand the array elements and store that into a new array as shown below.

#!/bin/bash

Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora' 'UTS' 'OpenLinux');

Linux=("${Unix[@]}")

echo ${Linux[@]}

$ ./arraymanip.sh

Debian Red hat Ubuntu Fedora UTS Open. Linux

### 17. Concatenation of two Bash Arrays

Expand the elements of the two arrays and assign it to the new array.

#!/bin/bash

Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora' 'UTS' 'OpenLinux');

Shell=('bash' 'csh' 'jsh' 'rsh' 'ksh' 'rc' 'tcsh');

UnixShell=("${Unix[@]}" "${Shell[@]}")

echo ${UnixShell[@]}

echo ${#UnixShell[@]}

$ ./arraymanip.sh

Debian Red hat Ubuntu Suse Fedora UTS OpenLinux bash csh jsh rsh ksh rc tcsh

14

It prints the array which has the elements of the both the array ‘Unix’ and ‘Shell’, and number of elements of the new array is 14

### 18. Deleting an Entire Array

unset is used to delete an entire array.

$cat arraymanip.sh

#!/bin/bash

Unix=('Debian' 'Red hat' 'Ubuntu' 'Suse' 'Fedora' 'UTS' 'OpenLinux');

Shell=('bash' 'csh' 'jsh' 'rsh' 'ksh' 'rc' 'tcsh');

UnixShell=("${Unix[@]}" "${Shell[@]}")

unset UnixShell

echo ${#UnixShell[@]}

$ ./arraymanip.sh

0

After unset an array, its length would be zero as shown above.

### 19. Load Content of a File into an Array

You can load the content of the file line by line into an array.

#Example file

$ cat logfile

Welcome

to

thegeekstuff

Linux

Unix

$ cat loadcontent.sh

#!/bin/bash

filecontent=( `cat "logfile" `)

for t in "${filecontent[@]}"

do

echo $t

done

echo "Read file content!"

$ ./loadcontent.sh

Welcome

to

thegeekstuff

Linux

Unix

Read file content!

In the above example, each index of an array element has printed through for loop.

20. #!/bin/bash

array1[0]=one

array1[1]=1

echo ${array1[0]}

echo ${array1[1]}

array2=( one two three )

echo ${array2[0]}

echo ${array2[2]}

array3=( [9]=nine [11]=11 )

echo ${array3[9]}

echo ${array3[11]}

read -a array4

for i in "${array4[@]}"

do

echo $i

done

exit 0