

```

1
2 -----
3 12-26-2020
4 https://youtu.be/RhOhupxTz4M
5 -----
6 shell script
7 -----
8
9 [root@script01 ~]# ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab
10
11 you can literally have unlimeted command running from CLI
12 this kind of commands is considered shell script
13
14 -----
15 so you these command and put in a file, it bomes a bash script
16
17 -----
18 Bash Script
19 -----
20 #!                #< --- shebang
21 /bin/bash         #< --- location of bash command
22
23
24 #!/bin/bash       #< --- you have to have this to become script
25
26 -----
27
28 #!/bin/bash
29
30
31 ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab;
32
33 -----
34 Method of executing the script file
35 -----
36
37 [root@script01 ~]# ./script
38
39 -----
40 Permission
41 -----
42 The script file must have execute permission
43
44 -----
45
46 [root@script01 ~]# ls -l
47 total 4
48 -rw-r--r--. 1 root root 78 Dec 26 13:45 script
49 [root@script01 ~]# ./script
50 bash: ./script: Permission denied
51
52 [root@script01 ~]# chmod 755 script      #< ---execute permission added
53
54 [root@script01 ~]# ls -l
55 total 4
56 -rwxr-xr-x. 1 root root 78 Dec 26 13:45 script
57
58 -----
59 execute the script
60 -----
61
62
63 [root@script01 ~]# ./script #< --- All the commands in the file will run
64
65 -----
66 if you want to redirec the conent of the out to a file
67
68 [root@script01 ~]# ./script >> output.txt
69

```

```

70 output.txt will hold the output from the script
71
72 -----
73 Script Identification
74 -----
75 The script itself does not need any extension, but generally its
76 good idea to have one
77
78 Exmaples: .sh .scr .scrp
79
80
81 [root@script01 ~]# ls -l
82 total 12
83 -rw-r--r--. 1 root root 2096 Dec 26 13:57 output.txt
84 -rwxr-xr-x. 1 root root 78 Dec 26 13:45 script #< ---
85 -rwxr-xr-x. 1 root root 39 Dec 26 14:00 template.scr #< ---
86
87 -----
88 comment out
89 -----
90 # hash is used for comments or disabling specific line in script
91
92 #!/bin/bash
93
94
95 #ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab; #< --- blind
96 ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab;
97 #ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab; #< --- blind
98
99 -----
100
101
102 #!/bin/bash
103
104 echo "This is lsblk command" #< --- This command will run
105
106 lsblk #< --- This command will run
107
108 echo "df -h is commented out" #< --- This command will run
109
110 #df -h #< --- This command will not run
111
112 -----
113 basic.sh
114 -----
115
116 #!/bin/bash
117
118 #This is basic script with clean output
119
120 echo "-----This is output of ls -l-----"
121
122 ls -l
123
124 echo "-----This is output of pwd-----"
125
126 pwd
127
128 echo "-----This is output of hostname-----"
129
130 hostname
131
132 echo "-----This is output of lsblk-----"
133
134 lsblk
135
136 echo "-----This is output of df -h-----"
137
138 df -h

```

```

139
140 echo "-----This is output of cat /etc/fstab-----"
141
142 -----
143 commandlocation.scr
144 -----
145
146 #!/bin/bash
147
148 #This is a commandlocation.scr
149
150
151 df -h
152 echo
153
154 echo "this is ls -l `ls -l` " #< ---use backtick ` if you have to use
155                               the command middle of the line
156
157
158 -----
159 Arithmetic Operators
160 -----
161
162 + addition
163 - subtraction
164 * multiplication #< ---when using inside script use \*
165 / division
166 -----
167 expr             #< --- expr is built-in command, stands for expression
168 -----
169
170 [root@script01 ~]# expr 2 + 2
171 4
172 [root@script01 ~]# expr 2 - 2
173 0
174 [root@script01 ~]# expr 2 / 2
175 1
176 [root@script01 ~]# expr 2 \* 2
177 4
178
179 -----
180 booblean operators #< --- comparision ture or false
181 -----
182
183 -eq == equal to
184 -ne != not equal to
185 -gt > greater than
186 -lt < less than
187 -le <= less than or equal to
188 -ge >= greater than or equal to
189
190 || means "or" - two pipes - this is key above "Enter"
191
192 && mesn "and"
193
194 $? - to check the exit code
195
196 -----
197 read - this is built in command, provides chance to user for input
198 -----
199
200
201
202 [root@script01 ~]# read
203 zafar #< --- input is sitting memory
204
205 There is no way to recall whats in memory
206
207 [root@script01 ~]# read name1 #< ---you need to give a place holder [variable], name1

```

```

208 is variable
209 zafar                                #< --- now its sitting memory
210 [root@script01 ~]# echo $name1 #< --- echo command is used to read from memory
211 zafar
212
213 -----
214 another example
215 -----
216
217
218 [root@script01 ~]# read name1
219 This is a linux course
220 [root@script01 ~]# echo $name1
221 This is a linux course
222
223
224 -----
225 another example
226 -----
227 [root@script01 ~]# echo $name1
228 This is a linux course
229 [root@script01 ~]# read name2
230 haroon
231 [root@script01 ~]# read name3
232 burhan
233 [root@script01 ~]# read name4
234 adil
235 [root@script01 ~]# echo $name1 $name2 $name3 $name4 $name2 #< ---extracted from memory
236 This is a linux course haroon burhan adil haroon
237
238 -----
239
240 [root@script01 ~]# read num1
241 45
242 [root@script01 ~]# read num2
243 55
244 [root@script01 ~]# echo $num1 $num2
245 45 55
246 [root@script01 ~]# expr $num1 + $num2
247 100
248 [root@script01 ~]# expr $num1 - $num2
249 -10
250 [root@script01 ~]# expr $num1 \* $num2
251 2475
252 [root@script01 ~]# expr $num1 / $num2
253 0
254
255 -----
256
257 [root@script01 ~]# read num1
258 45
259 [root@script01 ~]# read num2
260 55
261 [root@script01 ~]# echo $num1 $num2
262 45 55
263 [root@script01 ~]# expr $num1 + $num2
264 100
265 [root@script01 ~]# expr $num1 - $num2
266 -10
267 [root@script01 ~]# expr $num1 \* $num2
268 2475
269 [root@script01 ~]# expr $num1 / $num2
270 0
271
272 -----
273
274 [root@script01 ~]# echo $name1 $num2 $name2 $num1 $name3 $name4 $name2
275 This is a linux course 55 haroon 45 burhan adil haroon

```

```

276
277 -----
278 cal.sc
279 -----
280
281 #!/bin/bash
282
283 #This is a simple calculator
284
285 echo "Enter a first number"
286
287 read num1
288
289 echo "Enter a second number"
290
291 read num2
292
293 total=`expr $num1 + $num2`    #< --- total is place holder [variable - made up]
294
295 echo "The sum of $num1 and $num2 is : $total" #<
296
297 -----
298 in reality a script is extracted and executed on CLI by the system
299 -----
300
301 [root@script01 ~]# echo "Enter a first number"
302 Enter a first number
303 [root@script01 ~]# read num1
304 45
305 [root@script01 ~]# echo "Enter a second number"
306 Enter a second number
307 [root@script01 ~]# read num2
308 90
309 [root@script01 ~]# total=`expr $num1 + $num2`
310 [root@script01 ~]# echo "The sum of $num1 and $num2 is : $total"
311 The sum of 45 and 90 is : 135
312 [root@script01 ~]# echo "The sum of $num1 and $num2"
313 The sum of 45 and 90
314 [root@script01 ~]# echo $total
315 135
316
317 -----
318 you get same result running at CLI
319 -----
320
321 [root@script01 ~]# echo "Enter a first number";read num1;echo "Enter a second
322 number";read num2;total=`expr $num1 + $num2`;echo "The sum of $num1 and $num2 is :
323 $total"
324 Enter a first number
325 30
326 Enter a second number
327 40
328 The sum of 30 and 40 is : 70
329
330 -----
331 variable - place holder
332 -----
333
334 What is variable
335
336 Variable is a placeholder for a input, commands etc...,
337 variable are defined by person writing a script
338 -----
339 Two types of Variable - static and dynamic
340 -----
341
342 Static variable
343 -----
344

```

```
343 stvariable.st
344 -----
345
346 [root@script01 ~]# chicago=illinois
347 [root@script01 ~]# newyork=ny
348 [root@script01 ~]# echo chicago
349 chicago
350 [root@script01 ~]# echo $chicago
351 illinois
352 [root@script01 ~]# echo $newyork
353 ny
354
355
356
357 #!/bin/bash
358
359 #This is a static vairable exmaple
360
361 chicago=illinois
362 newyork=ny
363
364
365 echo "$chicago is in midwest"
366 echo
367 echo "$newyork is in eastcoast"
368
369 -----
370 [root@script01 ~]# ./stvariable.st
371 illinois is in midwest
372
373 ny is in eastcoast
374
375 -----
376
377 Dynamic Variable
378 -----
379
380 dyvariable.dy
381 -----
382
383 #!/bin/bash
384
385 #This is a dynamic [changes] variable
386
387 echo "Enter a place in midwest"
388
389 read midcity
390
391 echo "Enter a place in east coast"
392
393 read eastcity
394
395 echo
396 echo "$midcity is in midwest"
397 echo
398 echo "$eastcity is in east coast"
399
400 -----
401
402 [root@script01 ~]# ./dyvariable.dy
403 Enter a place in midwest
404 st louis
405 Enter a place in east coast
406 new jersey
407
408 st louis is in midwest
409
410 new jersey is in east coast
411
```

```
412 [root@script01 ~]# ./dyvariable.dy
413 Enter a place in midwest
414 springfield
415 Enter a place in east coast
416 boston
417
418 springfield is in midwest
419
420 boston is in east coast
421
422 -----
423 Using Alias inside the script
424 -----
425
426 aliaszafar.scr
427 -----
428
429
430 #!/bin/bash
431
432 #This is a exmample of using alias
433
434 shopt -s expand_aliases #< ---type exactly the way it is, it will set and unset once
the script is done
435
436 alias haroon="ls;pwd;echo;hostname;echo;lsblk;echo;df -h;echo;cat /etc/fstab;"
437
438 haroon
439
440 -----
441 12-27-2020
442
443 -----
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
```