

- > Какво е това нещо bash
- За какво може да се използва
- > Как да разберем кога нещо не е за bash



Ресурси от където можете да започнете

- http://www.tldp.org/HOWTO/
 Bash-Prog-Intro-HOWTO.html
- --- man bash ---
- Advanced bash scripting
 http://www.tldp.org/LDP/abs/html/



Прост скрипт на bash

```
#!/bin/bash
STR="Hello World!"
echo $STR
```



Какво ни е нужно за да си напишем хубав скрипт

- ясно дефиниране на задачата
- познаване на командите които ще ползвате за да се свърши задачата
- свободно време :)



Видове скриптове

- > one line ps aux|grep root|awk '{print \$2}'
- multiline
 string='2'
 let string++
 echo \$string



Variables

- export, typeset, declare, readonly
- local, global

```
function x {
  local USER='root'
}
```



Variables deference

- > \$a
- file_no\$a
- \$ \$afile_no does not work :)
- > \${a}file_no this is the way to do it



File descriptors(std=standard):
stdin - uses pipe or <
stdout - uses >
stderr - uses 2>



Какво можете да правите с тези file descriptor-и:

```
1. redirect stdout to a file
   ( ls - l > ls - l.txt )
2. redirect stderr to a file
   ( grep da * 2> grep-errors.txt )
3. redirect stdout to a stderr
   ( grep da * 1>&2 )
4. redirect stderr to a stdout
   (grep * 2 > \&1)
5. redirect stderr and stdout to a file
   ( rm -f $(find / -name core) &> /dev/null )
```



Pipes

Is -I | grep aha

ps aux | grep userx | awk '{print \$2}'

cat filename | sed 's/krava/mlqko/g'



```
#!/bin/bash
tar -cfz /usr/local/backup/my-backup-today.tgz \
    /home/me
```

```
#!/bin/bash
dir='/usr/local/backup'
file=$dir/my-backup-$(date+%Y%m%d).tgz
tar -cfz $file /home/me
```



debuging

най дорият начин: #!/bin/bash -x

за големи скриптове echo или read



Контролни структори

if-then-elif-then-elif-then-else select-in case-in for-do while-do until-do



if-then-else

```
if [ "$1" == " ]; then
    echo "Usage: $0 variable"
else
    make_me_stop
fi

if ( ! ps ax | grep kuku > /dev/null ); then
    echo "KUKU not found!"
fi
```



Difference in parenthesis

- () executes commands in new shell
- { } executes commands in the current shell
- (()) arithmetic expressions
- [] basic integer arithmetic, basic string comparison and file attributes checks
- [[]] basic integer arithmetic, regular expressions and file attributes checks



```
if-then-elif-then-else
if [ "$1" == 'weekly' ]; then
     table list=$(<tables.weekly)
elif [ "$1" == 'daily' ]; then
     table list=$(<tables.daily)
else
     table list=$(<tables.hourly)
tables="
for table in $table list; do tables="$tables $table"; done
mysqldump --compact -e -t -q -R --single-transaction blog db
$tables
```



case-in

```
case "$1" in
   'start')
     make_me_stop ;;
   'stop')
     make_me_start ;;
   *)
     echo "Usage: $0 variable";;
esac
```



```
for-do
for var in `seq 1 10`; do
echo -n "$var "
done
Ето какво ще изведе това:
1 2 3 4 5 6 7 8 9 10
```

for file in /bin/*; do echo \$file done Това е еквивалентно на ls -1A /bin



while-do

```
count=0;
while(true); do
    let count++;
    if [ "$count" == "10" ]; then
        exit 0;
    fi
done
```



while-do

```
count=0;
while(true); do
    let count++;
    if [ "$count" == "10" ]; then
        exit 0;
    fi
done
```



functions

```
function fixme {
    echo $0 $1 $2 $3
}
```

Как викаме функция # ./fixme a j k ./fixme a j k #



Bash regular expressions

```
#!/bin/bash
regex=$1
if [[ $1 =~ $regex ]]; then
  echo "$2 matched"
else
  echo "$2 NOT matched"
fi
# ./regex search searchstring
searchstring matched
```



Bash substitutions

```
#!/bin/bash
for i in /dir/*; do
echo ${i/.*/}
done
```

\$ Is *.pl *.txt
collect-plans.pl cpu-abuse.pl landing.txt ap_write.txt
\$./script
collect-plans
cpu-abuse
landing
ap write



Bash substitutions

Inside \${ ... }

Action taken

name:number:number

#name

name#pattern

name##pattern

Substring starting character,

length

Return the length of the

string

Remove (shortest)

front-anchored pattern

Remove (longest)

front-anchored pattern



Bash substitutions

Inside \${ ... }

name%pattern

name%%pattern

name/pattern/string name//pattern/string

Action taken

Remove (shortest)
rear-anchored pattern
Remove (longest)
rear-anchored pattern
Replace first occurrence
Replace all occurrences



Bash substitutions

hackman@terion:~\$ a='123456'

hackman@terion:~\$ echo \${a:3:5} 456

hackman@terion:~\$ echo \${a:1:4} 2345

hackman@terion:~\$ echo \${#a} 6



Bash substitutions (front)

hackman@terion:~\$ a='grizzly.yuhu.biz'

hackman@terion:~\$ echo \${a#*.} yuhu.biz

hackman@terion:~\$ echo \${a##*.} biz



Bash substitutions (rear)

hackman@terion:~\$ a='grizzly.yuhu.biz'

hackman@terion:~\$ echo \${a%.*} grizzly.yuhu

hackman@terion:~\$ echo \${a%%.*} grizzly



Bash substitutions (regexp)

hackman@terion:~\$ a='pate.pate.patence.txt'

hackman@terion:~\$ echo \${a/pate/kate} kate.pate.patence.txt

hackman@terion:~\$ echo \${a//pate/kate} kate.kate.katence.txt



Bash arrays

```
#!/bin/bash
a=(x y z)
echo ${a[0]}
echo ${a[1]}
echo ${a[2]}
# ./array
x
y
z
```

```
#!/bin/bash
a=(x y z)
echo ${#a[*]}
# ./array
3
```



Parameter expansion

```
$ function am() { for i in "$*"; do echo \"$i\"; done }
$ am jo ji ko
"jo ji ko"

$ function am() { for i in "$@"; do echo \"$i\"; done }
$ am jo ji ko
"jo"
"ji"
"ko"
```



Bash arrays

```
#!/bin/bash
a=(x y z)
for i in "${a[*]}"; do
# ./array
x y z
```

```
#!/bin/bash
a=(x y z)
for i in "${a[@]}"; do
# ./array
X
Y
Z
```



Bash special variables

- \$0 - the name of the script or shell
- \$# - number of parameters/arguments
- \$! - PID of the last executed background command
- exit status of the last executed command
- \$? \$* \$@ \$-- expands to a single quoted word
- expands to separate words
- current bash flags/options
- IFS - internal field separator



Bash arrays

```
hackman@gamelon:~$ ip a I eth0|grep 'inet '
inet 10.2.0.3/24 brd 10.2.0.255 scope global eth0
hackman@gamelon:~$ ip=($(ip a I eth0|grep 'inet '))
hackman@gamelon:~$ echo ${ip[0]}
inet
hackman@gamelon:~$ echo ${ip[1]}
10.2.0.3/24
```



Command execution

```
ip=( $( ip a I eth0|grep 'inet ' ) )
```

VS.

ip=(` ip a l eth0|grep 'inet ' `)



