Bash Scripting

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

Bash Scripting

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/

Bash Scripting

Прост скрипт на **bash**

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

Bash Scripting

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 :)

Bash Scripting

Видове скриптове

- one line
  ps aux|grep root|awk '{print $2}'

- multiline
  string='2'
  let string++
  echo $string

Bash Scripting

Variables

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

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

Bash Scripting

File descriptors(std=standard):

- **stdin – uses pipe or <**
- **stdout – uses >**
- **stderr – uses 2>**

Bash Scripting

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)

Bash Scripting

Pipes

```
ls -l | grep aha

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

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

Bash Scripting

```
#!/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
```

Bash Scripting

debuging

най дорият начин:
#!/bin/bash -x

за големи скриптове
echo или read

Bash Scripting

Контролни структори

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

Bash Scripting

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
```

Bash Scripting

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

Bash Scripting

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)
fi

tables=''
for table in $table_list; do tables="$tables $table"; done
mysqldump --compact -e -t -q -R --single-transaction blog_db
$tables
```

Bash Scripting

case-in

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

Bash Scripting

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

Bash Scripting

while-do

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

Bash Scripting

functions

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

Как викаме функция

```
# ./fixme a j k
./fixme a j k
#
```

Bash Scripting

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 Scripting

Bash substitutions

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

$ ls *.pl *.txt
collect-plans.pl  cpu-abuse.pl  landing.txt  ap_write.txt
$ ./script
collect-plans
cpu-abuse
landing
ap_write
```

Bash Scripting

Bash substitutions

Inside ${ ... }	Action taken
name:number:number	Substring starting character, length
#name	Return the length of the string
name#pattern	Remove (shortest) front-anchored pattern
name##pattern	Remove (longest) front-anchored pattern

Bash Scripting

Bash substitutions

Inside ${ ... }	Action taken
name%pattern	Remove (shortest) rear-anchored pattern
name%%pattern	Remove (longest) rear-anchored pattern
name/pattern/string	Replace first occurrence
name//pattern/string	Replace all occurrences

Bash Scripting

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 Scripting

Bash substitutions (front)

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

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

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

Bash Scripting

Bash substitutions (rear)

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

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

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

Bash Scripting

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 Scripting

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
```

Bash Scripting

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 Scripting

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 Scripting

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 Scripting

Bash arrays

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

Bash Scripting

? ? ? ? ? ?

? ? ? ?

? ? ? ? ?

? ? ? Въпроси ? ?

?

? ? ? ? ? ?

? ? ? ? ? ? ?

? ? ? ? ? ?

Мариян Маринов(HackMan) mm@yuhu.biz