Big Script – tips and general requirements

Operating Systems,

Katarzyna Łukasiewicz, Michał Wróbel, 2020

Big script- requirements

1. Each script **should have a header**. It's worth adding a license information (not necesseraly opensource), so the users would know how they can use (or modify) the script.

2. Code should have **comments**.

3. Script should be **immune** to various "unwanted" scenarios of usage

4. Each script should have at least **two options**:
   -h – short help
   -v – version and author's info.

Header example

```
# Author            : Name ( email )
# Created On        : date
# Last Modified By  : Name ( email )
# Last Modified On  : date
# Version           :
#
# Description       :
#
#
# Licensed under GPL (see /usr/share/common-licenses/GPL for more
# details or contact the Free Software Foundation for a copy)
```

getopts

```
while getopts hvf:q OPT; do
    case $OPT in
        h) help;;
        v) version;;
        f) FILE=$OPTARG;;
        q) echo ”Text”
           exit;;
        *) echo ”Unknown option”;;
    esac
done
```

The colon (:) means that the preceeding option has an additional argument.

Functions

- Code can be arranged into functions.

```
name(){
    # CODE
}
# CODE
name
# CODE
```

- All the variables in bash are global. You may pass the variables though, if you want to:

```
name(){
    echo $1
}
NAME=”Ala”
name $NAME
```

- Then, you can access them by using $1 $2..., just like script parameters.

Configuration

The unchangeable values (like names of files, catalogs etc) should be kept in variables at the beginning of the script:

```
KATALOG=~/ala
FILE=$KATALOG/makota
#CODE
cat $FILE | ....
```

Or you can create a config file, like skrypt.rc:

```
KATALOG=”~/katalog”
TIME=10
ACTION=”delete”
```

Then you can import these variables into the script file:

```
. skrypt.rc
```