Linux — TD3

Thomas PRÉVOST, Margot TAILLANTOU-CANDAU (CSN24)

1. En mode interactif

```
#!/bin/bash

function update_indexes(){
        echo "updating indexes"
        sudo apt-get update
}

function update_soft(){
        echo "updating softwares"
        sudo apt-get upgrade
}

function save_config(){
        echo "Creating a backup of current config"
        journalctl -u ssh >> /root/logs/journalctl_$(date +%y%m%d).log
        journalctl -u nginx >> /root/logs/journalctl_$(date +%y%m%d).log
}

function save_state(){
        echo "saving state"
        top >> /root/logs/state_$(date +%y%m%d).log
        free >> /root/logs/state_$(date +%y%m%d).log
        df >> /root/logs/state_$(date +%y%m%d).log
        ps >> /root/logs/state_$(date +%y%m%d).log
}


Choices=(1 "Mettre à jour l'index"
           2 "Mettre à jour les logiciels"
           3 "Sauvegarder la config système"
           4 "Créer un rapport système")


opt=$(dialog --clear --backtitle "backtitle" --title "Utilitaire admin" --
menu "chose :" 15 40 4 "${Choices[@]}" 2>&1 >/dev/tty)

clear

case $opt in
        1)
                update_indexes
```

```
                ;;
        2)
                update_soft
                ;;
        3)
                save_config
                ;;
        4)
                save_state
                ;;
esac
```

2. En lot

```
#!/usr/bin/env bash

set -Eeuo pipefail
trap cleanup SIGINT SIGTERM ERR EXIT

script_dir=$(cd "$(dirname "${BASH_SOURCE[0]}")" &>/dev/null && pwd -P)

update(){
        sudo apt-get update
}

upgrade(){
        sudo apt-get upgrade
}

config_sys(){
        sudo journalctl -u ssh >> /root/logs/journalctl_$(date +%y%m%d).log
        sudo journalctl nginx >> /root/logs/journalctl_$(date +%y%m%d).log
}

sys_state(){
        sudo free >> /root/logs/state_$(date +%y%m%d).log
        sudo ps >> /root/logs/state_$(date +%y%m%d).log
        sudo top >> /root/logs/state_$(date +%y%m%d).log
        sudo df >> /root/logs/state_$(date +%y%m%d).log
}

all(){
        update
        upgrade
        config_sys
        sys_state
}
```

```
usage() {
  cat <<EOF
Usage: $(basename "${BASH_SOURCE[0]}") [-h] [-v] [-f] -p param_value arg1
[arg2...]

Script description here.

Available options:

-h, --help      Print this help and exit
-v, --verbose   Print script debug info
--upgrade       Mise à jour des logiciels
--update        Mise à jour de l'index des dépôts logiciel
--config        Sauvegarde de la config système
--state         Création d'un rapport sur l'état du système
--all           Les exécute tous à la suite
EOF
  exit
}

cleanup() {
  trap - SIGINT SIGTERM ERR EXIT
  # script cleanup here
}

setup_colors() {
  if [[ -t 2 ]] && [[ -z "${NO_COLOR-}" ]] && [[ "${TERM-}" != "dumb" ]];
then
    NOFORMAT='\033[0m' RED='\033[0;31m' GREEN='\033[0;32m'
ORANGE='\033[0;33m' BLUE='\033[0;34m' PURPLE='\033[0;35m' CYAN='\033[0;36m'
YELLOW='\033[1;33m'
  else
    NOFORMAT='' RED='' GREEN='' ORANGE='' BLUE='' PURPLE='' CYAN=''
YELLOW=''
  fi
}

msg() {
  echo >&2 -e "${1-}"
}

die() {
  local msg=$1
  local code=${2-1} # default exit status 1
  msg "$msg"
  exit "$code"
}

parse_params() {
  # default values of variables set from params
  flag=0
```

```
  param=''

  while :; do
    case "${1-}" in
    -h | --help) usage ;;
    -v | --verbose) set -x ;;
    --no-color) NO_COLOR=1 ;;
    -f | --flag) flag=1 ;; # example flag
    --upgrade) upgrade ;;
    --update) update ;;
    --state) sys_state ;;
    --config) config_sys ;;
    --visual) ./script.sh ;;
        --all) all ;;
    -?*) die "Unknown option: $1" ;;
    *) break ;;
    esac
    shift
  done

  args=("$@")

  # check required params and arguments
  [[ -z "${param-}" ]] && die "Missing required parameter: param"
  [[ ${#args[@]} -eq 0 ]] && die "Missing script arguments"

  return 0
}

parse_params "$@"
setup_colors

# script logic here

msg "${RED}Read parameters:${NOFORMAT}"
msg "- flag: ${flag}"
msg "- param: ${param}"
msg "- arguments: ${args[*]-}"
```