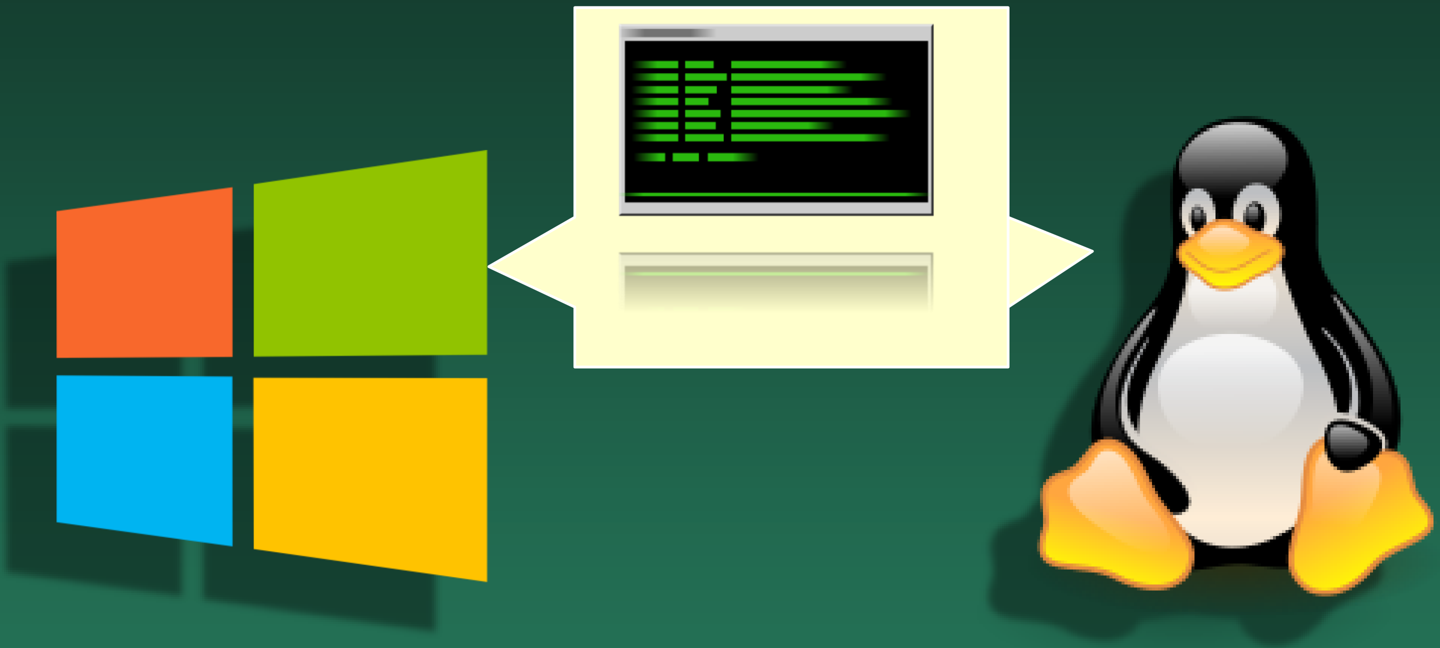
System shells



BASH syntax - 7

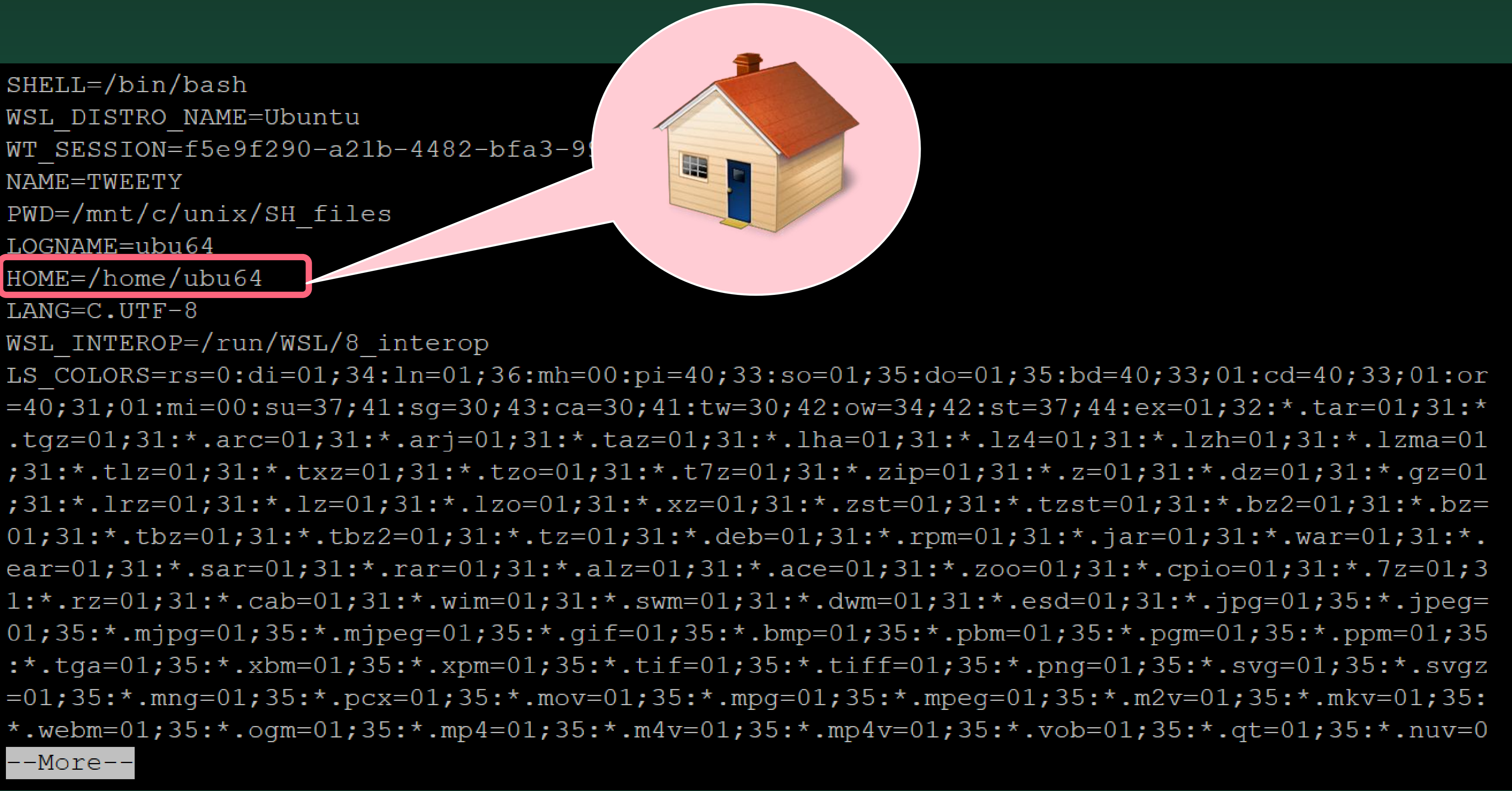
```
 8    <DIR>          workspace
               2 295 zaza.txt
ichier(s)           123 383 octets
p(s)  118 519 529 472 octets libres

AN>cd \unix


cteur C s'appelle Tweety
 volume est 5E35-93A3



05/12/202   34             2 768 goal.o
17/01/2023  11:53          2 568 lab1.o
10/11/2022  10:07          1 472 lab11.o
01/09/2022  16:31          2 416 lab2.o
02/09/2022  10:58          1 968 lab3.o
20/01/2023  12:31          2 880 lateP ndin
20/01/2023  11:51          1 976 main
20/01/2023  12:16          1       meas     e.
09/11/2022  12:59                 ar5.o
09/11/2022  15:00                 ar6.
04/01/2023  12:09
              11 fichier(s)        26 176 octets
               0 Rép(s)  118 617  4 096 octets libres

C:\unix>
C:\unix>
C:\unix>
```

2 - BASH Vars

Variables, scripts

BASH variables list

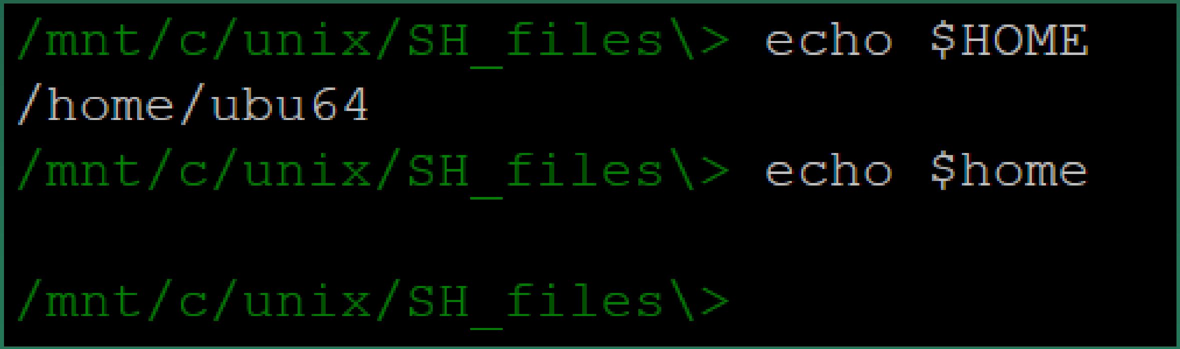
- Type **printenv | MORE**



```
SHELL=/bin/bash
WSL_DISTRO_NAME=Ubuntu
WT_SESSION=f5e9f290-a21b-4482-bfa3-9
NAME=TWEETY
PWD=/mnt/c/unix/SH_files
LOGNAME=ubu64
HOME=/home/ubu64
LANG=C.UTF-8
WSL_INTEROP=/run/WSL/8_interop
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or
=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*
.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01
;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01
;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=
01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.
ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;3
1:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=
01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35
:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz
=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:
*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=0
--More--
```

Home sweet Home

- Type `echo HOME`
  - Expected HOME string ☹
- Type `echo $HOME`
  - Content of the HOMEPATH variable ☺
- Type `echo $home`
  - Remember BASH IS case sensitive



```
/mnt/c/unix/SH_files\> echo $HOME
/home/ubu64
/mnt/c/unix/SH_files\> echo $home

/mnt/c/unix/SH_files\>
```

Some other important variables

- `echo $PATH`
  - Folders BASH will search for files to execute
  - `nano batch.sh` works even if `nano` is not in the current folder
- `echo $LOGNAME` = who is talking?
- `echo $SHELL` = what is the engine?
- `echo $PWD` = it is also a variable!
- `echo $_` = name of the running command
- `echo $RANDOM` = pseudo-random positive integer

Create your variables

- Create your own variables
  - `VAR1=this_is_my_variable`
    - !! No separators before & after `=`
  - Idem. `VAR2="this is my variable"`
  - Idem. `VAR3='this is my variable’`
  - No partial setting

    `myv1=this is` → will not set anything

- Remove variables
  - `unset VAR1`

Use variables

- Variables and special characters
  - `VAR1="my variable"`
  - `$VAR1` → `my variable`
  - `"$VAR1"` → `my variable`
  - `'$VAR1'` → `$VAR1`
  - `${VAR1}` → `my variable`
  - `{$VAR1}` → `{my variable}`
- Variable substitution
  - `VAR2="VAR1"`
  - `${!VAR2}` → `my variable`
  - `${VAR3:-"none"}` → `none`
  - `${VAR3:-$VAR1}` → `my variable`

Variable operations

String variable length	`${#VAR}`
Substring substitution	`${VAR/pattern/replace-string}`
Prefix remove to first pattern	`${VAR#*delete-pattern}`
Prefix remove to last pattern	`${VAR##*delete-pattern}`
Suffix remove from last pattern	`${VAR%delete-Pattern*}`
Suffix remove from first pattern	`${VAR%%delete-pattern*}`
Reduction to end	`${VAR:offset}`
Reduction of length	`${VAR:offset:length}`
Upper case	`${VAR^^}`
Lower case	`${VAR,,}`

Arithmetic expressions

- Type **`TEN=10`**
- Type **`echo $TEN`**
  - Displays `10`
- Type **`ELEVEN=$(($TEN+1))`**

Double parenthesis

- Type **`echo $ELEVEN`**
  - Variable `%ELEVEN%` is `11`

String variable!!

- Type **`ELEVEN=$TEN+1`**
- Type **`echo $ELEVEN`**
  - Variable `$ELEVEN` is `(10+1)`

Calculus & numeric bases

- Default is decimal
  - Type `TEN=10`
  - **`echo $(($TEN +1))`** Displays `11`
- Hexadecimal with **`0x`** prefix
  - Type `TEN=0x10`
  - **`echo $(($TEN +1))`** Displays `17` *(…the Hex 11)*
- Octal with **`0`** prefix
  - Type `TEN=010`
  - **`echo $(($TEN +1))`** Displays `9` *(…the Oct 11)*
  - Beware `08` and `09` are not allowed in `$((…))`

Declare & use Arrays

◆ Declare	`declare -a TABL=(AAA BB CCC)`	
◆ Use index	`echo ${TABL[1]}`	→ BB
• `echo ${TABL[@]:1:2}`		→ BB CCC
◆ Serialize	`echo ${TABL[@]}`	→ AAA BB CCC
◆ Cardinal	`echo ${#TABL[@]}`	→ 3
◆ Element size	`echo ${#TABL[1]}`	→ 2
◆ Clear element	`unset ${TABL[1]}`	

Brace expansion

- Special brace expression
  - `{a..z}` → `a b c d e f ... x y z`
  - `{0..1}` → `0 1 2 3 4 5 6 7 8 9`
  - `{$from..$to}`

- Array construction
  - `ALPHA=({A..Z})`
  - `${#ALPHA[@]}` → `26`

- String generation
  - `DIGIT=({0..9})`
  - `TYPO=(${ALPHA[@]} ${DIGIT[@]})`
  - `PASS=$(shuf -n8 -er ${TYPO[@]} | paste -sd "")`

Query the user

- Type `read USERINPUT`
  - Will wait for user input until `<ENTER>`
- Type `echo $USERINPUT`
  - Will display user input
- Type `read -p "Please type..." USERINPUT`
  - Displays a message before waiting

```
read -p "Continue? (Y/N): " confirm \
&& [[ $confirm == [yY] || $confirm == [yY][eE][sS] ]] \
&& echo OK! || echo NOK…
```