Using UNIX/Linux system

Krzysztof Kluza

08.11.2021

Introduction to Bash shell

1. In the shell, we can take advantage of environment variables.
   Using `printenv` check the existing environment variables.
   Check the meaning and the values of the following variables:

   - `HOME`
   - `SHELL`
   - `SHLVL`
   - `PATH`
   - `PWD`
   - `USER`

2. Using other shells like `csh` or `tcsh` find out what can you check using the following commands:

   - `printenv SHELL`
   - `echo $SHELL`
   - `echo $0`

3. Define your own environment variable (e.g. MYVAR and set its value to MYVALUE). How to set and unset such a variable?

4. Using the `PS1` variable, change the prompt in the terminal, e.g.:
   `export PS1='\u@\h:\w$'` Check other possibilities of setting the prompt, e.g. set the prompt to:

   - `I am user in working directory>`
   - `date,time$`

   where underlined text should be changed according to the current environment.

5. Give an example of brace expansion and how it can be used in bash.

6. Using `export` (or not), it is easy to test the inheritance of the bash prompt:

   ```
   bash
   MYVAR=somevalue
   printenv MYVAR
   echo $MYVAR
   bash
   printenv MYVAR
   echo $MYVAR
   exit
   export MYVAR
   printenv MYVAR
   echo $MYVAR
   bash
   printenv MYVAR
   echo $MYVAR
   exit
   exit
   ```

7. Compare the behaviour of the following commands:

   ```
   echo $SHELL
   echo "$SHELL"
   echo '$SHELL'
   echo \$SHELL
   echo \\$SHELL
   echo \$$SHELL
   echo "my system: uname"
   echo "my system: 'uname'"
   echo "my system: `uname`"
   echo "ls -l"
   echo 'ls -l'
   echo `ls -l`
   ```

8. There is a fixed execution order of startup files. Add in each startup file an `echo` command with some additional information e.g. `echo alpha`, `echo beta`, etc. and determine the order of startup files when started `bash` as an interactive login shell, a login shell, or when started as an interactive shell (but not a login shell), e.g. `bash` or `bash --login`.

   Add the `clear` command when closing the shell. You can test the behaviour when you call a specific file using the `source` command.