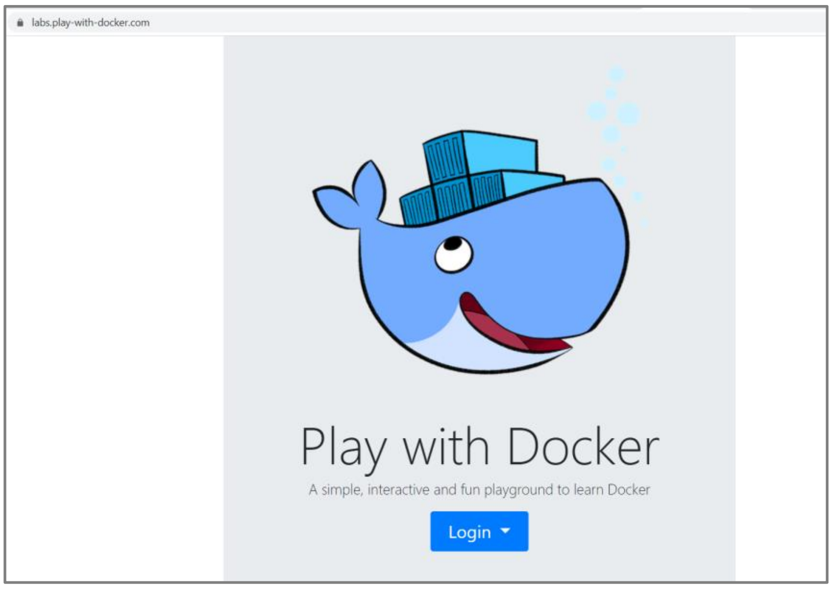
Lab: Linux and Linux Shell

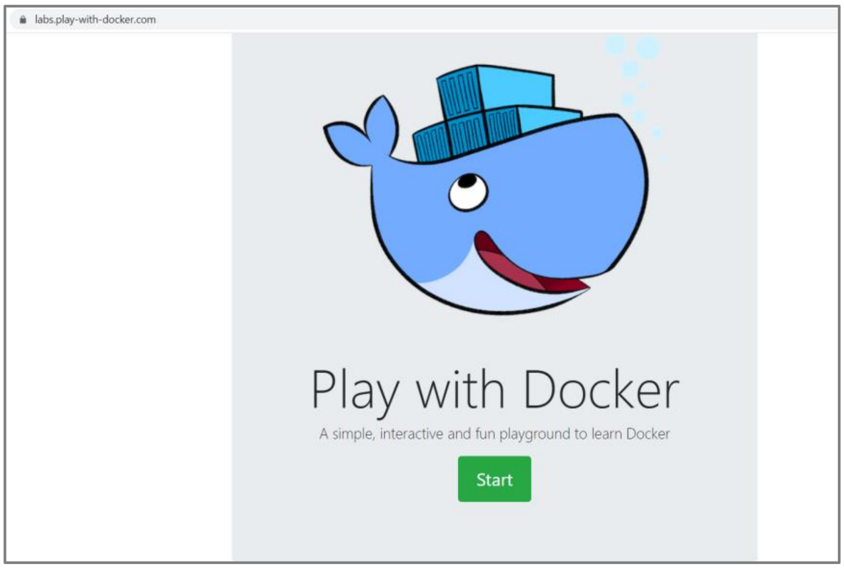
Lab for the "Containers and Clouds" course @ SoftUni

1. Configure Connectivity of the VM

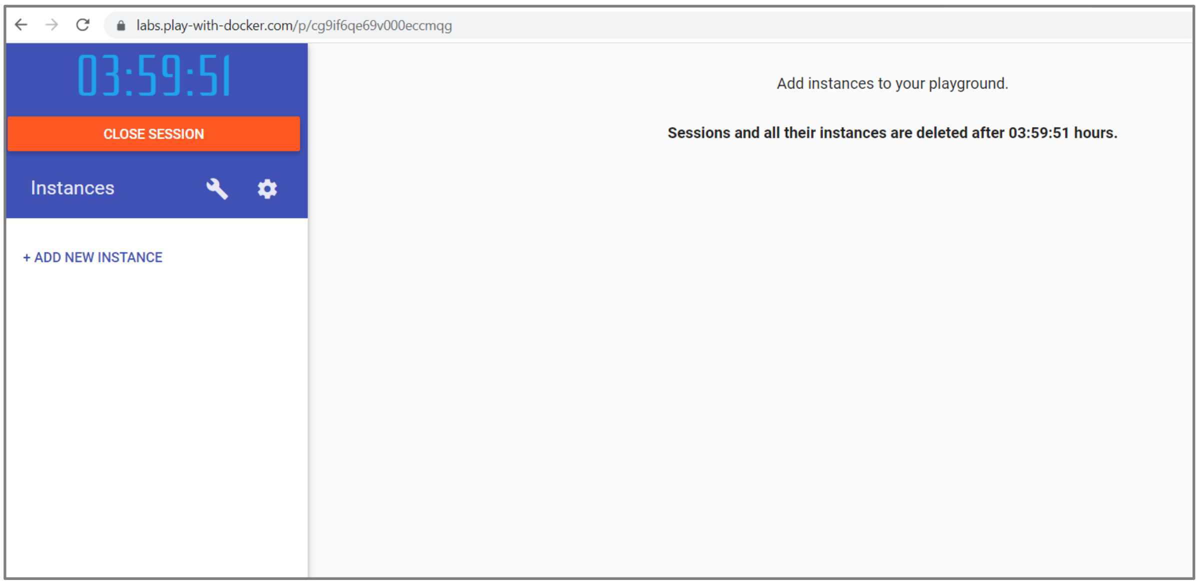
The first step is to open the link – https://labs.play-with-docker.com/.


labs.play-with-docker.com
Play with Docker
A simple, interactive and fun playground to learn Docker
Login

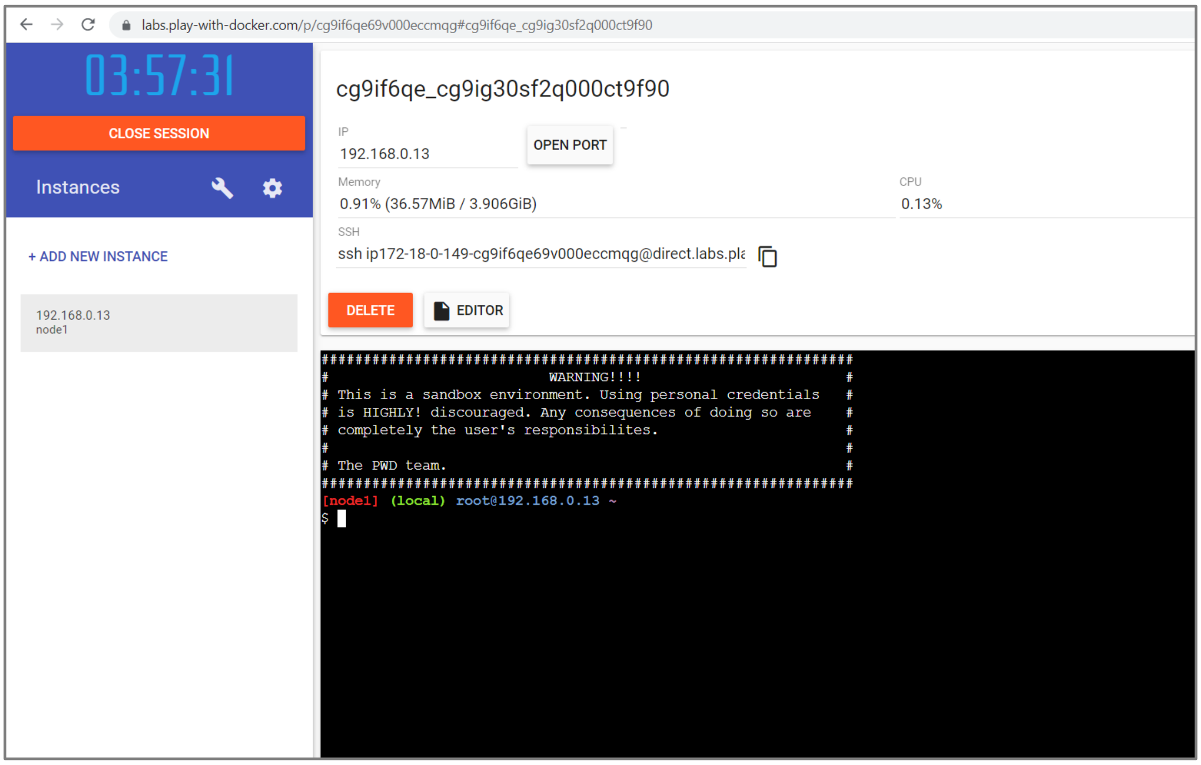
Then press the **`[Login]`** button and click on "**`docker`**". A new dialog box opens, which is for **docker registration**. If you don't have a registration click on **`[Sign Up]`**. You have to create an account with a username, password and email. Then sign in to your account. It takes you to a page to select the plan you want – click on "**Continue to Free**" (Personal plan). Log in to the email you registered with and **confirm your account**. You may need to reload the page until you see this:


labs.play-with-docker.com
Play with Docker
A simple, interactive and fun playground to learn Docker
Start

Click on the **`[Start]`** button.


labs.play-with-docker.com/p/cg9if6qe69v000eccmqg
03:59:51
CLOSE SESSION
Instances
+ ADD NEW INSTANCE
Add instances to your playground.
Sessions and all their instances are deleted after 03:59:51 hours.

You should see this. Click on `[+ add new instance]`.

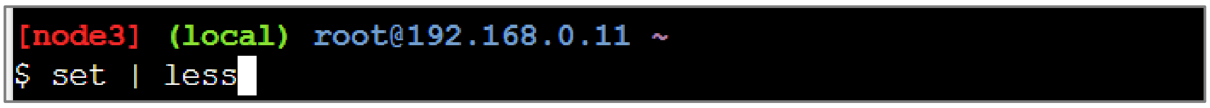

labs.play-with-docker.com/p/cg9if6qe69v000eccmqg#cg9if6qe_cg9ig30sf2q000ct9f90
03:57:31
CLOSE SESSION
Instances
+ ADD NEW INSTANCE
192.168.0.13
node1
cg9if6qe_cg9ig30sf2q000ct9f90
IP
192.168.0.13
OPEN PORT
Memory
0.91% (36.57MiB / 3.906GiB)
CPU
0.13%
SSH
ssh ip172-18-0-149-cg9if6qe69v000eccmqg@direct.labs.pla
DELETE
EDITOR

```
###################################################################
#                          WARNING!!!!                             #
# This is a sandbox environment. Using personal credentials        #
# is HIGHLY! discouraged. Any consequences of doing so are         #
# completely the user's responsibilites.                           #
#                                                                  #
# The PWD team.                                                    #
###################################################################
[node1] (local) root@192.168.0.13 ~
$ 
```

We are ready for work!

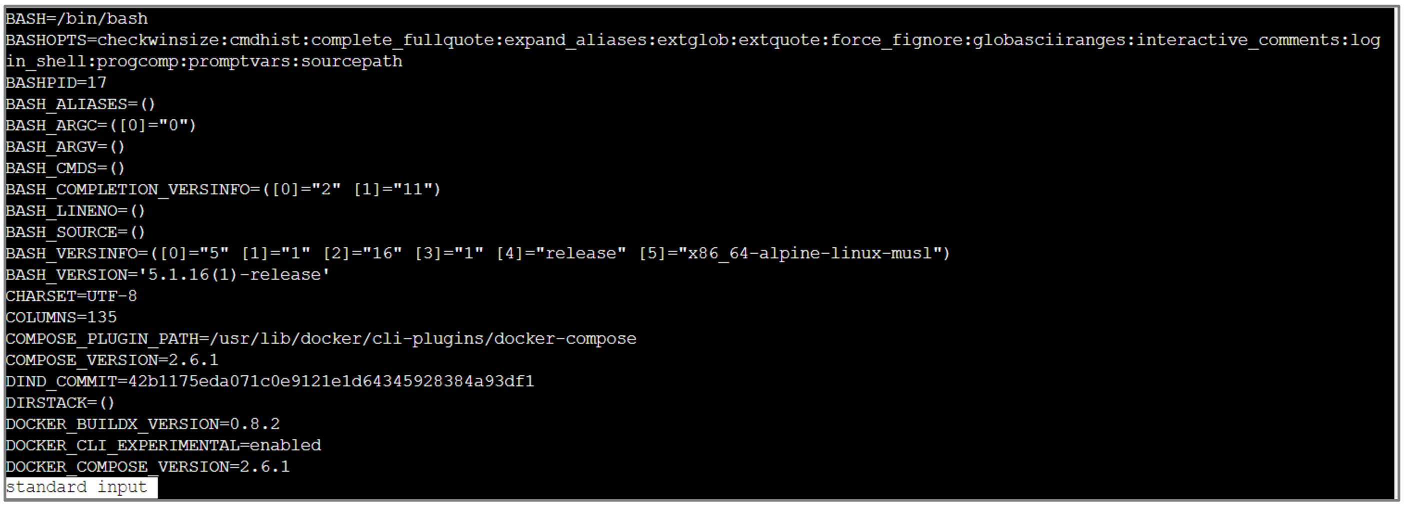
2. Getting to Know Environment Variables

Let's start with the **environment exploration process**. Try out the **set** command and look at its result:



```
[node3] (local) root@192.168.0.11 ~
$ set | less
```

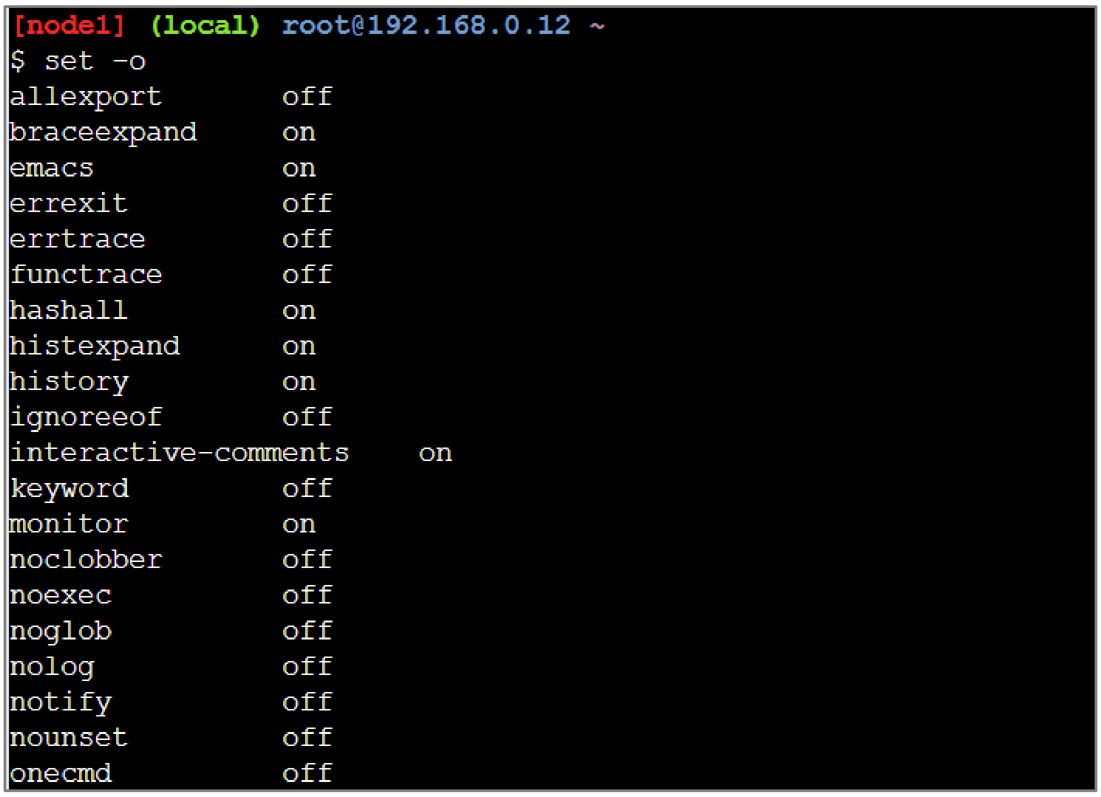
↓


```
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_fignore:globasciiranges:interactive_comments:log
in_shell:progcomp:promptvars:sourcepath
BASHPID=17
BASH_ALIASES=()
BASH_ARGC=([0]="0")
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_VERSINFO=([0]="2" [1]="11")
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="5" [1]="1" [2]="16" [3]="1" [4]="release" [5]="x86_64-alpine-linux-musl")
BASH_VERSION='5.1.16(1)-release'
CHARSET=UTF-8
COLUMNS=135
COMPOSE_PLUGIN_PATH=/usr/lib/docker/cli-plugins/docker-compose
COMPOSE_VERSION=2.6.1
DIND_COMMIT=42b1175eda071c0e9121e1d64345928384a93df1
DIRSTACK=()
DOCKER_BUILDX_VERSION=0.8.2
DOCKER_CLI_EXPERIMENTAL=enabled
DOCKER_COMPOSE_VERSION=2.6.1
standard input
```

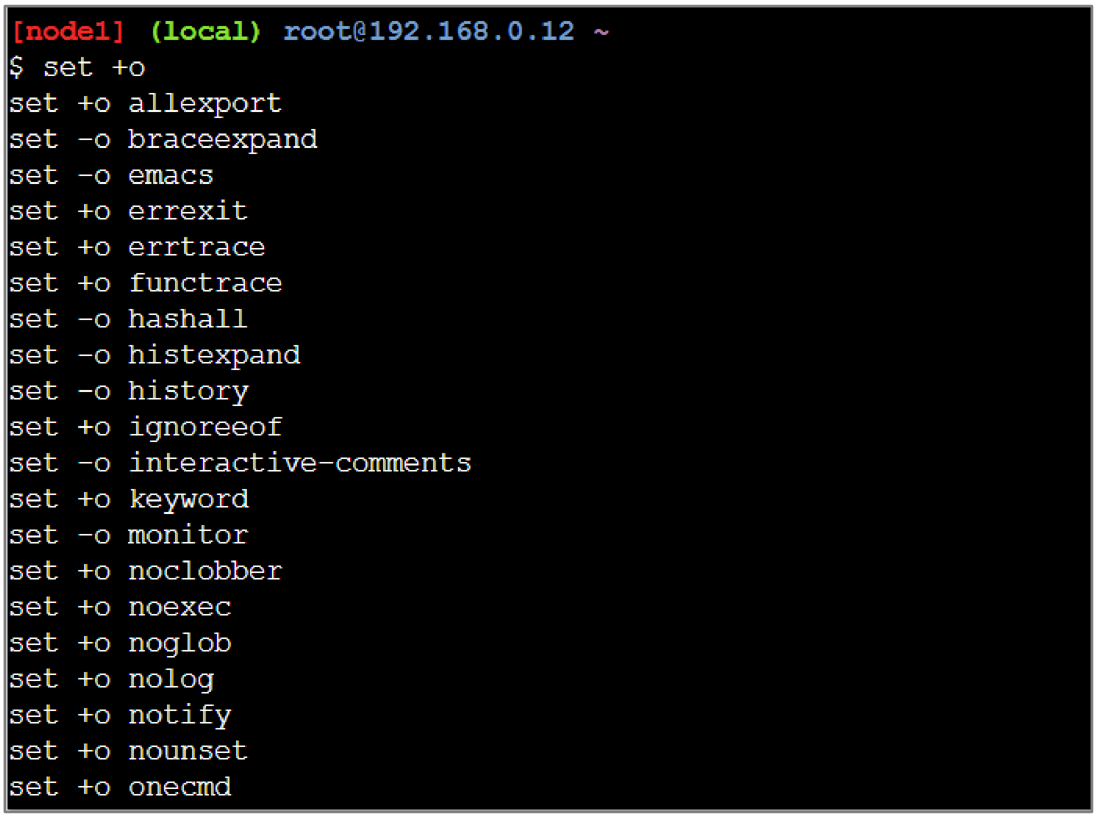
When executed **without parameters**, **`set`** returns **information about the environment** – **variables** and **functions**. Again, the information will differ from distribution to distribution and between versions of distribution.

We added **`| less`** to the **`set`** command to **display the command output**, **one page at a time**. It is **optional**, but it's a good idea to use it when the **output is longer**. You can go to the **next pages** with **`[Enter]`** or **quit** with **`[q]`**.

The **`set`** command can be used to **modify the parameters** that are driving the environment. In order to see what **parameters** are there, we can execute:

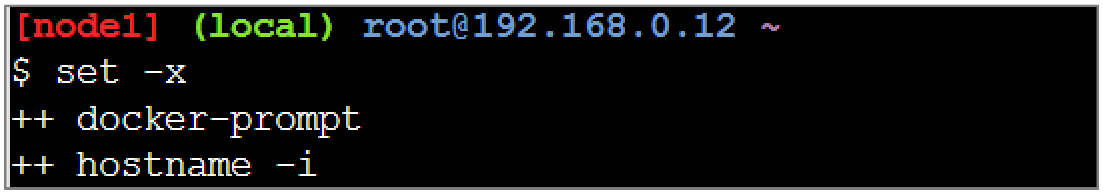

```
[node1] (local) root@192.168.0.12 ~
$ set -o
allexport       off
braceexpand     on
emacs           on
errexit         off
errtrace        off
functrace       off
hashall         on
histexpand      on
history         on
ignoreeof       off
interactive-comments    on
keyword         off
monitor         on
noclobber       off
noexec          off
noglob          off
nolog           off
notify          off
nounset         off
onecmd          off
```

We can **see the same information** but prepared for **re-use** with:



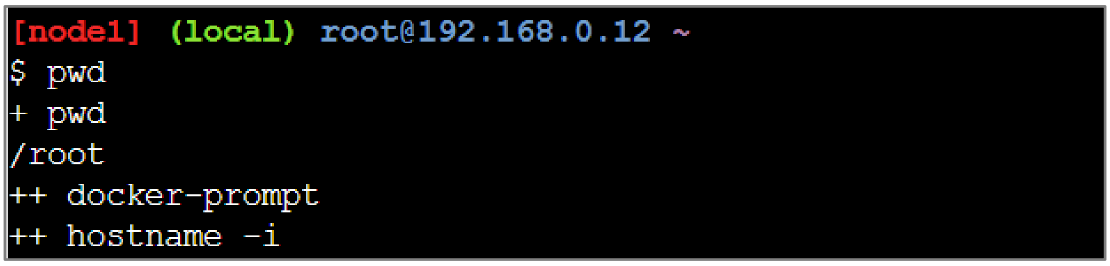
```
[node1] (local) root@192.168.0.12 ~
$ set +o
set +o allexport
set -o braceexpand
set -o emacs
set +o errexit
set +o errtrace
set +o functrace
set -o hashall
set -o histexpand
set -o history
set +o ignoreeof
set -o interactive-comments
set +o keyword
set -o monitor
set +o noclobber
set +o noexec
set +o noglob
set +o nolog
set +o notify
set +o nounset
set +o onecmd
```

Let's **change the flag** that **controls the amount of information** shown during commands execution:

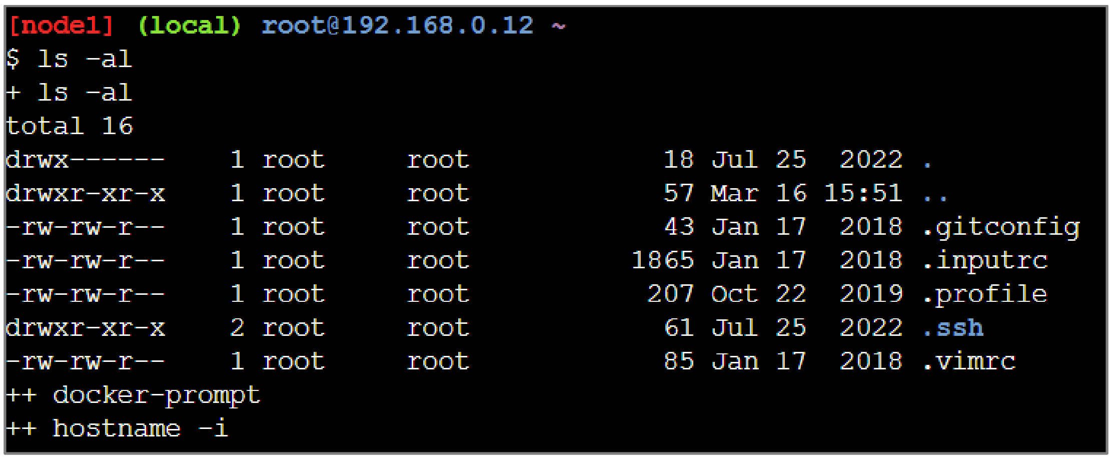


```
[node1] (local) root@192.168.0.12 ~
$ set -x
++ docker-prompt
++ hostname -i
```

You should note that the minus ("**-**") is used to **turn on a flag**, while the plus ("**+**") is used to **deactivate a flag**. Now, we can execute few more commands:



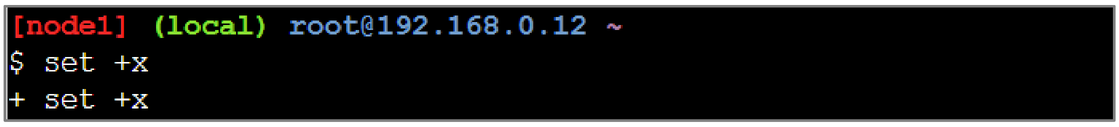
```
[node1] (local) root@192.168.0.12 ~
$ pwd
+ pwd
/root
++ docker-prompt
++ hostname -i
```



```
[node1] (local) root@192.168.0.12 ~
$ ls -al
+ ls -al
total 16
drwx------    1 root     root            18 Jul 25  2022 .
drwxr-xr-x    1 root     root            57 Mar 16 15:51 ..
-rw-rw-r--    1 root     root            43 Jan 17  2018 .gitconfig
-rw-rw-r--    1 root     root          1865 Jan 17  2018 .inputrc
-rw-rw-r--    1 root     root           207 Oct 22  2019 .profile
drwxr-xr-x    2 root     root            61 Jul 25  2022 .ssh
-rw-rw-r--    1 root     root            85 Jan 17  2018 .vimrc
++ docker-prompt
++ hostname -i
```

We can see that immediately **after each command** there is **information aboutse what is going to be executed**. This way we can see that instead `ls`, in fact something else is being executed – its **alias** or the statement it specifies.

It is time to **deactivate** the **`xtrace` mode**:



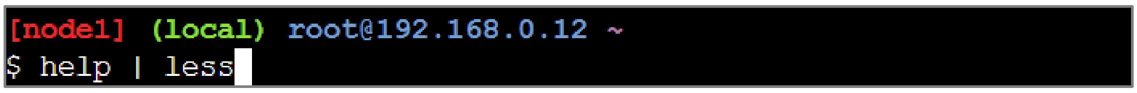
```
[node1] (local) root@192.168.0.12 ~
$ set +x
+ set +x
```

Now you will not be able to see what commands are executed.

3. Getting Help

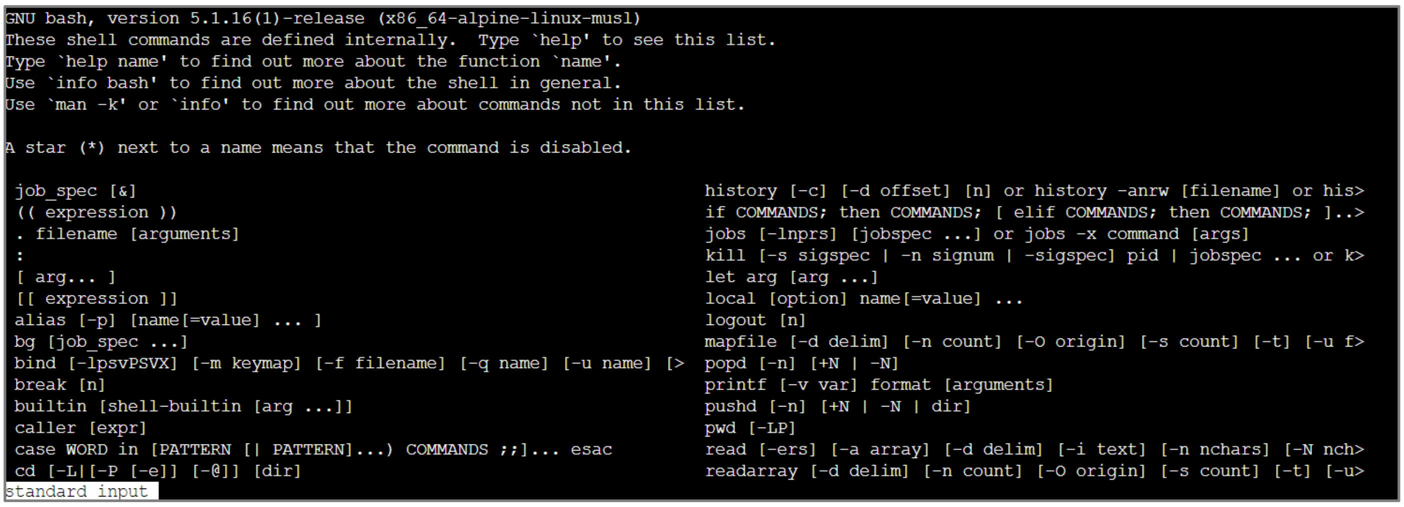
Now it is time to explore ways of **getting help** about different commands.

For the **shell built-in commands**, we can use the **help** command. If we execute it **without parameters**, it will return **all built-in commands**:



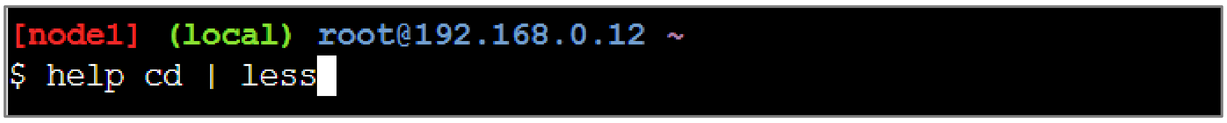
```
[node1] (local) root@192.168.0.12 ~
$ help | less
```

↓



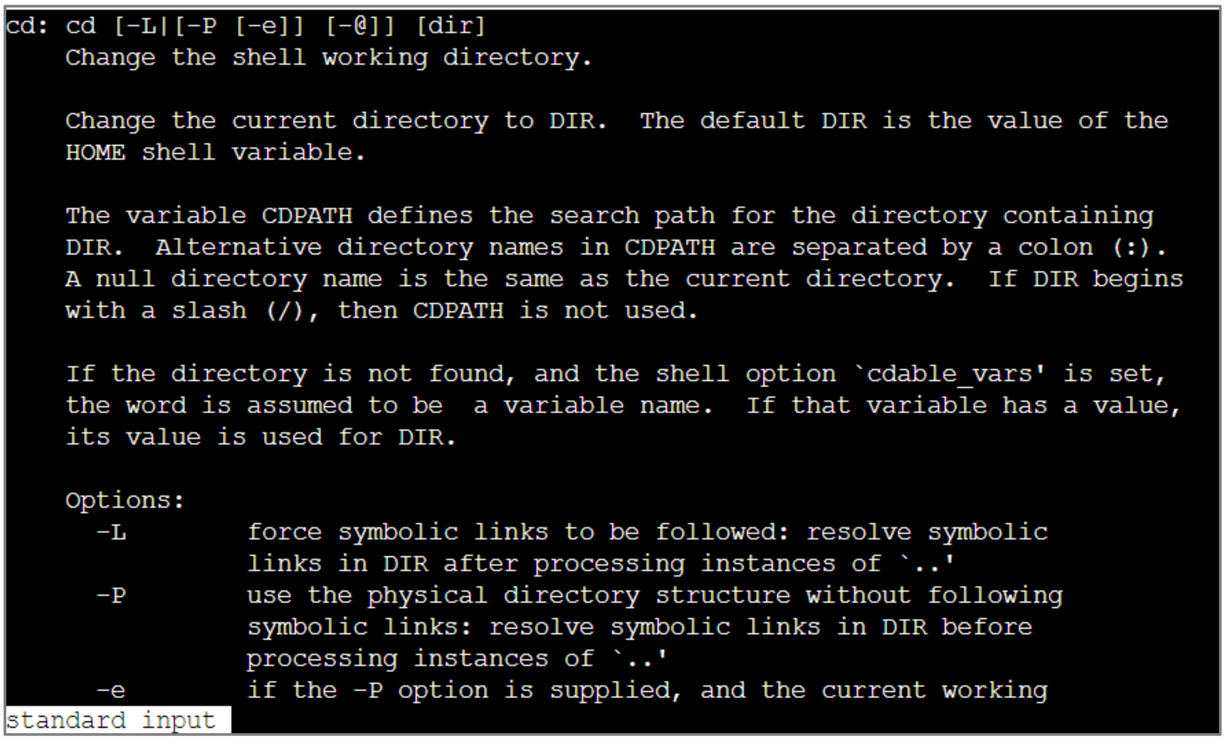
```
GNU bash, version 5.1.16(1)-release (x86_64-alpine-linux-musl)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                                                       history [-c] [-d offset] [n] or history -anrw [filename] or his>
 (( expression ))                                                   if COMMANDS; then COMMANDS; [ elif COMMANDS; then COMMANDS; ]..>
 . filename [arguments]                                             jobs [-lnprs] [jobspec ...] or jobs -x command [args]
 :                                                                  kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or k>
 [ arg... ]                                                         let arg [arg ...]
 [[ expression ]]                                                   local [option] name[=value] ...
 alias [-p] [name[=value] ... ]                                     logout [n]
 bg [job_spec ...]                                                  mapfile [-d delim] [-n count] [-O origin] [-s count] [-t] [-u f>
 bind [-lpsvPSVX] [-m keymap] [-f filename] [-q name] [-u name] [>  popd [-n] [+N | -N]
 break [n]                                                          printf [-v var] format [arguments]
 builtin [shell-builtin [arg ...]]                                  pushd [-n] [+N | -N | dir]
 caller [expr]                                                      pwd [-LP]
 case WORD in [PATTERN [| PATTERN]...) COMMANDS ;;]... esac         read [-ers] [-a array] [-d delim] [-i text] [-n nchars] [-N nch>
 cd [-L|[-P [-e]] [-@]] [dir]                                       readarray [-d delim] [-n count] [-O origin] [-s count] [-t] [-u>
standard input
```

In order to **ask for a command**, we must execute it like:



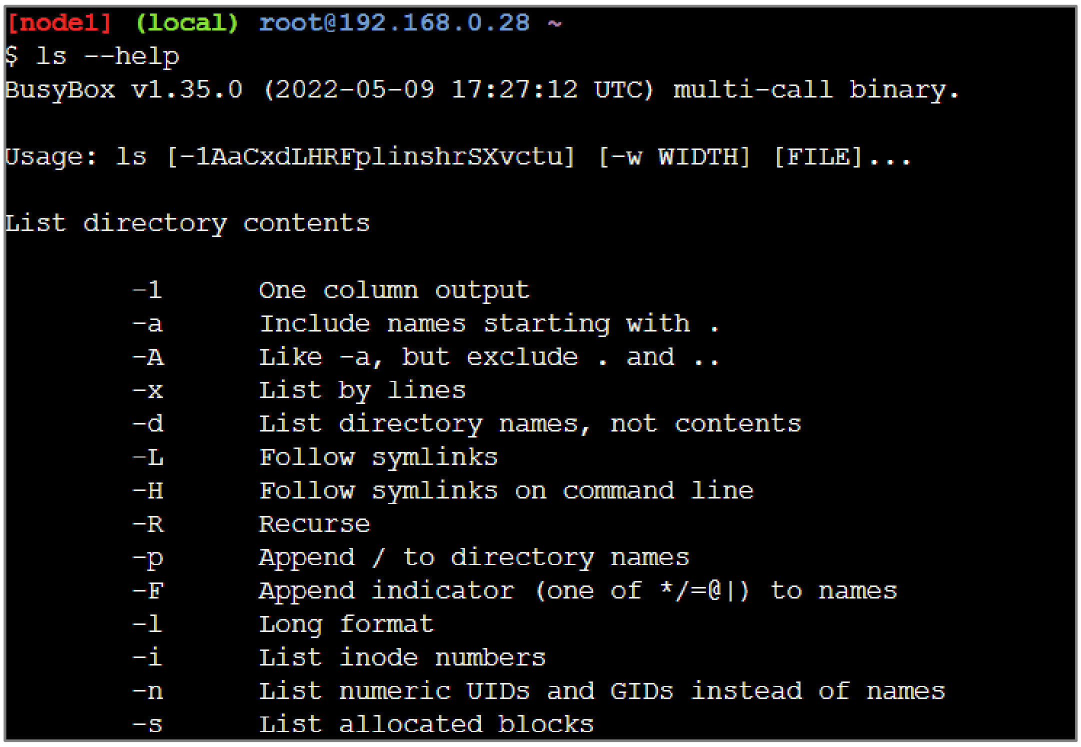
```
[node1] (local) root@192.168.0.12 ~
$ help cd | less
```

↓



```
cd: cd [-L|[-P [-e]] [-@]] [dir]
    Change the shell working directory.

    Change the current directory to DIR.  The default DIR is the value of the
    HOME shell variable.

    The variable CDPATH defines the search path for the directory containing
    DIR.  Alternative directory names in CDPATH are separated by a colon (:).
    A null directory name is the same as the current directory.  If DIR begins
    with a slash (/), then CDPATH is not used.

    If the directory is not found, and the shell option `cdable_vars' is set,
    the word is assumed to be  a variable name.  If that variable has a value,
    its value is used for DIR.

    Options:
      -L        force symbolic links to be followed: resolve symbolic
                links in DIR after processing instances of `..'
      -P        use the physical directory structure without following
                symbolic links: resolve symbolic links in DIR before
                processing instances of `..'
      -e        if the -P option is supplied, and the current working
standard input
```

Most **external commands** offer integrated help. The ways to **ask for this information** vary, but typically we can use:



```
[node1] (local) root@192.168.0.28 ~
$ ls --help
BusyBox v1.35.0 (2022-05-09 17:27:12 UTC) multi-call binary.

Usage: ls [-1AaCxdLHRFplinshrSXvctu] [-w WIDTH] [FILE]...

List directory contents

        -1      One column output
        -a      Include names starting with .
        -A      Like -a, but exclude . and ..
        -x      List by lines
        -d      List directory names, not contents
        -L      Follow symlinks
        -H      Follow symlinks on command line
        -R      Recurse
        -p      Append / to directory names
        -F      Append indicator (one of */=@|) to names
        -l      Long format
        -i      List inode numbers
        -n      List numeric UIDs and GIDs instead of names
        -s      List allocated blocks
```

When using **man**, it is good to know the following **set of keys**:

- Key `[h]` shows **help information**
- Key `[q]` either **exits the help mode** or **the man**
- If we want to **search for something** from the current cursor position onwards, then we can press `[/]`, then enter the string and then press the `[Enter]` key. Once in this mode, we can press `[n]` key to **move forward** over the matches, or the `[N]` key to **move backward**
- Searching backwards works the same, but it is initiated with the `[?]` key

A **quick search** in **man** can be done on the command line with:

```
lsauser@softuni:~$ man -k passwd
chgpasswd (8)        - update group passwords in batch mode
chpasswd (8)         - update passwords in batch mode
gpasswd (1)          - administer /etc/group and /etc/gshadow
grub-mkpasswd-pbkdf2 (1) - generate hashed password for GRUB
openssl-passwd (1ssl) - compute password hashes
pam_localuser (8)    - require users to be listed in /etc/passwd
passwd (1)           - change user password
passwd (1ssl)        - compute password hashes
passwd (5)           - the password file
update-passwd (8)    - safely update /etc/passwd, /etc/shadow and /etc/group
```

Similar effect can be achieved by using the **apropos** command:

```
lsauser@softuni:~$ apropos passwd
chgpasswd (8)        - update group passwords in batch mode
chpasswd (8)         - update passwords in batch mode
gpasswd (1)          - administer /etc/group and /etc/gshadow
grub-mkpasswd-pbkdf2 (1) - generate hashed password for GRUB
openssl-passwd (1ssl) - compute password hashes
pam_localuser (8)    - require users to be listed in /etc/passwd
passwd (1)           - change user password
passwd (1ssl)        - compute password hashes
passwd (5)           - the password file
update-passwd (8)    - safely update /etc/passwd, /etc/shadow and /etc/group
```

Beside **man**, usually there is an alternative and modern help system available – **info**. We can use it the same way:

```
lsauser@softuni:~$ info ls
```

↓

```
Next: dir invocation,  Up: Directory listing

10.1 'ls': List directory contents
==================================

The 'ls' program lists information about files (of any type, including
directories).  Options and file arguments can be intermixed arbitrarily,
as usual.

   For non-option command-line arguments that are directories, by
default 'ls' lists the contents of directories, not recursively, and
omitting files with names beginning with '.'.  For other non-option
arguments, by default 'ls' lists just the file name.  If no non-option
argument is specified, 'ls' operates on the current directory, acting as
-----Info: (coreutils)ls invocation, 57 lines --Top----------------------------
Welcome to Info version 6.7.  Type H for help, h for tutorial.
```

For **help** inside the tool, we can press the **[h]** key. The help screen can be **closed** with the **[x]** key. No matter where we are in the tool, we can **exit** with the **[q]** key.