Linux Plus for AWS and DevOps

CLARUSWAY
WAY TO REINVENT YOURSELF

Linux Environment Variables

CLARUSWAY
WAY TO REINVENT YOURSELF

Table of Contents

1

What are Environment variables?

Variables

Variables are used to store for data values

What are Environment variables?

An environment variable is a **dynamic-named** value that can **affect the way running processes will behave** on a computer.

Environment variables can be created, edited, saved, and deleted and give information about the system behavior.

Environment variables allow you to **customize how the system works** and the behavior of the applications on the system.

What are Environment variables?

A list of **all specified environment variables** can be viewed entering the **env** command.

There is **nothing special** about variable names, but, by convention, environment variables should have **UPPER CASE** names.

2

Common Environment Variables

Common Environment Variables

Variable	Description
PATH	This variable contains a **colon (:)-separated list of directories** in which your system looks for executable files.
USER	The username
HOME	**Default path** to the **user's home** directory
EDITOR	Path to the program which edits the content of files
UID	User's unique ID
TERM	Default terminal emulator
SHELL	Shell being used by the user
LANG	The current locales settings.

Common Commands

Command	Description
env	The **env** command is a shell command used to display and manipulate environment variables. It is **used to list down** environment variables.
printenv	The command prints all or the specified environment variables.
set	The command sets or unsets shell variables. When used without an argument it will print a list of all variables including environment and shell variables, and shell functions.
unset	The command deletes shell and environment variables.
export	The command sets environment variables.

3

Accessing Variable

Common Commands

Command	Description
echo $VARIABLE	To display value of a variable
env	Displays all environment variables
VARIABLE_NAME=variable_value	Create a new shell variable
unset	Remove a variable
export Variable=value	To set value of an environment variable

Accessing Variable

printenv or echo

Display Path Environment Variable.

```
clarusway@DESKTOP-UN6T2ES:~$ printenv USER
clarusway
clarusway@DESKTOP-UN6T2ES:~$ printenv HOME
/home/clarusway
clarusway@DESKTOP-UN6T2ES:~$ printenv UID
clarusway@DESKTOP-UN6T2ES:~$ echo $TERM
xterm-256color
clarusway@DESKTOP-UN6T2ES:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```


Define a New Variable

Define a new variable

```
clarusway@DESKTOP-UN6T2ES:~$ NEWVARIABLE=newvalue
clarusway@DESKTOP-UN6T2ES:~$ echo $NEWVARIABLE
newvalue
clarusway@DESKTOP-UN6T2ES:~$
```

Remove a Variable

unset Remove a variable from the system.

```
clarusway@DESKTOP-UN6T2ES:~$ unset NEWVARIABLE
clarusway@DESKTOP-UN6T2ES:~$ echo $NEWVARIABLE

clarusway@DESKTOP-UN6T2ES:~$
```


Kahoot!

Exercise

Create a variable named **MYVAR** with the value of **“my value”**
Print value of the **MYVAR** variable to the screen
Assign **“new value”** to the **MYVAR** variable
Print value of the **MYVAR** variable to the screen
Delete **MYVAR** variable
Print value of the **MYVAR** variable to the screen

4 The PATH Variable

The PATH Variable

When we want the system to execute a command, we almost **never need to give the full path** to that command.

For instance, we know that the ls command is in the /bin directory (you can check with), yet we don't need to enter the /bin/ls command for the computer to list the content of the current directory.

This is maintained by the PATH environment variable. This variable **lists all directories** in the system **where executable files** can be **found**.

The PATH Variable

printenv — Display Path Environment Variable.

```
clarusway@DESKTOP-UN6T2ES:~$ printenv PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

In this example, the /usr/local/sbin, /usr/local/bin, /usr/sbin, /usr/bin, /sbin and /bin directories are subsequently searched for the required program. The search will be stopped as soon as a match is found, even if not all directories in the path have been

The PATH Variable

export Add a New Directory to the Path.

```
clarusway@DESKTOP-UN6T2ES:~$ export PATH=$PATH:/games/awesome
clarusway@DESKTOP-UN6T2ES:~$ _
```

Let's say you want to run that file called fun. You learned from running the find command that it's in a directory called /games/awesome. However, /games/awesome is not in your path, and you don't want to type the full path just to run the game. So you can add it to PATH variable with export command.

5 Quoting with Variables

Quoting

Quoting is used to disable special treatment of certain characters and words, as well as to prevent parameter expansion and preserve what is quoted.

The bash shell knows rare, important characters. For example, $var is used to extend the value of the element.

```
echo "$PATH"
echo "$PS1"
```


Quoting

Double Quotes

- The double quote ("quote") protects everything enclosed between two double quote marks except $, ', " and \.

```
echo "$SHELL"
echo "/etc/*.conf"
```

Single Quotes

- The single quote ('quote') protects everything enclosed between two single quote marks.

```
echo '$SHELL'
echo '/etc/*.conf'
```

Backslash

- Use the backslash to change the special meaning of the characters or to escape special characters within the text such as quotation marks.

```
echo "Path is \$PATH"
```

```
root@DESKTOP-4QQ1S5L:~# var="These are quotes(\)"
root@DESKTOP-4QQ1S5L:~# echo $var
These are quotes(\)
root@DESKTOP-4QQ1S5L:~# var='These are quotes(")'
root@DESKTOP-4QQ1S5L:~# echo $var
These are quotes(")
root@DESKTOP-4QQ1S5L:~# var="These are quotes(")"
-bash: syntax error near unexpected token `)'
root@DESKTOP-4QQ1S5L:~# var="The VAR1 variable is $VAR1"
root@DESKTOP-4QQ1S5L:~# echo $var
The VAR1 variable is
root@DESKTOP-4QQ1S5L:~#
```

6 sudo Command

CLARUSWAY
WAY TO REINVENT YOURSELF

sudo Command

- The sudo (**superuser do**) command gives **some admin privileges** to non-admin users.
- When you put sudo in front of any command in terminal, that command **runs with elevated privileges**.
- If you're not sure whether you're using sudo or su, look at the trailing character on the command line. If it's a pound sign (#), you're logged in as root.

CLARUSWAY
WAY TO REINVENT YOURSELF

sudo Command

Commands	Meaning
sudo -l	List available commands.
sudo command	Run command as root.
sudo -u root command	Run command as root.
sudo -u user command	Run command as user.
sudo su	Switch to the superuser account.
sudo su -	Switch to the superuser account with root's environment.
sudo su - username	Switch to the username's account with the username's environment.
sudo -s	Start a shell as root
sudo -u root -s	Same as above.
sudo -u user -s	Start a shell as user.

THANKS!

Any questions?