**User Environment**

**Environment Variables**

**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.

A lot of programs want to know about the kind of terminal you are using; this information is stored in the TERM variable, the shell you are using is stored in the SHELL variable, and so on.

A list of all specified environment variables can be viewed entering the printenv command. There is nothing special about variable names, but, by convention, environment variables should have UPPER CASE names.

The environment variables are managed by the shell. Unlike regular shell variables, environment variables are inherited by any program you start, including another shell. New processes are assigned a copy of these variables, which they can read, modify, and pass on in turn to their own child processes.

**💡Tips:**Variables can be classified into two main categories, environment variables, and shell variables.

* Environment variables are variables that are available system-wide and are inherited by all spawned child processes and shells.
* Shell variables are variables that apply only to the current shell instance.

**💡Tips:**

$LANG environment variable stores the value of the language that the user understands. This value is read by an application such that a Turkish user is shown a Turkish interface while an American user is shown an English interface.

**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. |

Commands that allow you to list and set environment variables in Linux

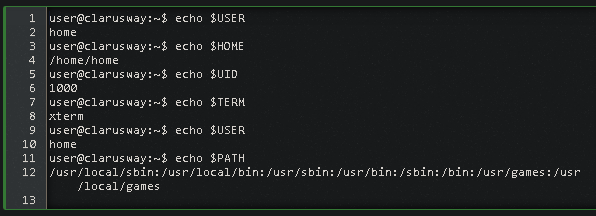
| **Command** | **Description** |
| --- | --- |
| env | The command allows you to run another program in a custom environment without modifying the current one. When used without an argument it will print a list of the current environment variables. |
| printenv | The command prints all or the specified 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. |

**Accessing Variable**

Variables are- Case Sensitive. Make sure that you type the variable name in the right letter case otherwise you may not get the desired results.

**💡Tips:**

Another important character interpreted by the shell is the dollar sign $. The shell will look for an environment variable named like the string following the dollar sign and replace it with the value of the variable (or with nothing if the variable does not exist). These are some examples using $HOSTNAME, $USER, $UID, $SHELL, and $HOME.



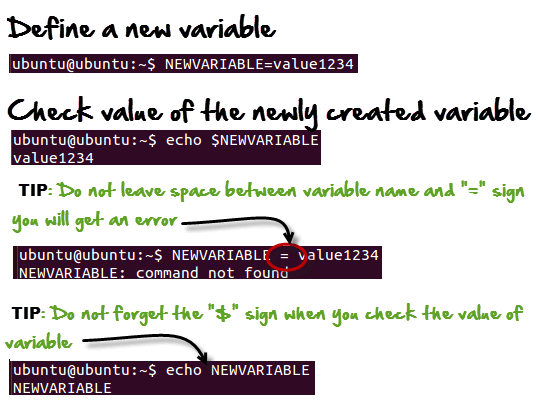
**💡Tips:**

Variables are case-sensitive and usually they are created in upper case.

You can create your own user-defined variable, with the syntax: VARIABLE\_NAME= variable\_value

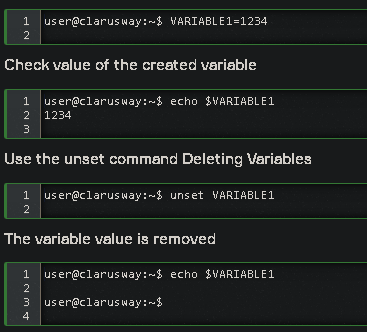
**💡Tips:**

This example creates the variable $NEWVARIABLE and sets its value. It then uses echo to verify the value.



You can be used to remove a Variable from the system - with unset variable name

**Define a new variable**



Environment variables control software actions in your Operating System.

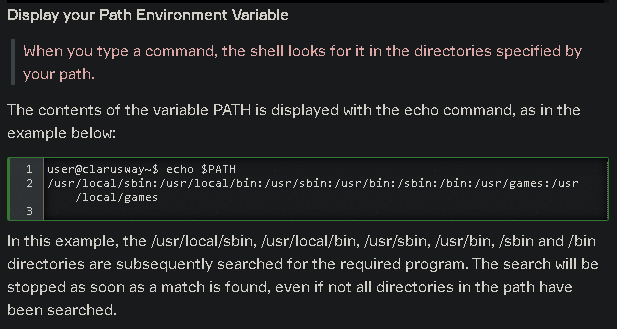
| **Command** | **Description** |
| --- | --- |
| echo $VARIABLE | To display value of a variable |
| env | Displays all environment variables |
| VARIABLE\_NAME= variable\_value | Create a new variable |
| echo $VARIABLE | To display value of a variable |
| unset | Remove a variable |
| export Variable=value | To set value of an environment variable |

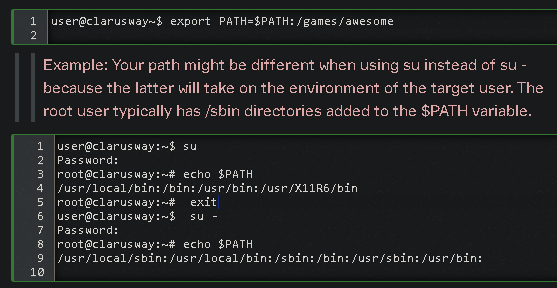
**The PATH**

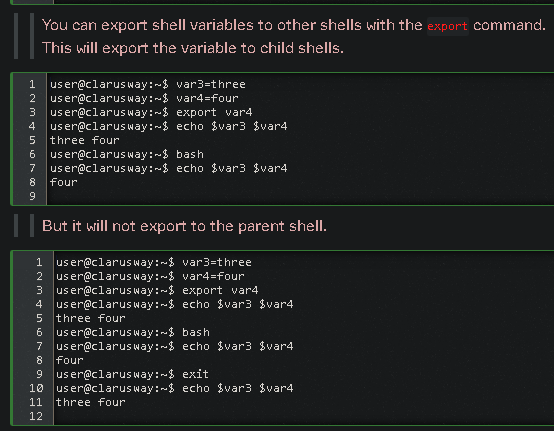
When we want the system to execute a command, we almost never need to give the full path to that command. For instance, we know that the ls command is in the /bin directory (you can check with which -a ls), yet we don't need to enter the /bin/ls command for the computer to list the content of the current directory.

This is maintained by the PATH environment variable. This variable lists all directories in the system where executable files can be found. So obviously the path includes several directories that contain bin somewhere in their names.

Video : <https://youtu.be/rJMFxIbDe-g>







**Tips:**

export k=1 and k=1:

* export makes the variable available to subprocesses.
* That is, if you spawn a new process from your script, the variable k won't be available to that subprocess unless you export it. Note that if you change this variable in the subprocess that change won't be visible in the parent process.

**Q:** What is the difference between path and directory?  
**A:** A directory is a "folder", a place where you can put files or other directories. It is a container for filesystem objects. A path is a string that specify how to reach a filesystem object

 - Interview Q&A

Complementary Lesson about Linux Environment :

<https://youtu.be/pjh9rU9h22Q>