Environment Variables

Environment Variables:

1. Environment variables are quantities that have specific values which may be utilized by the command shell, such as **bash**, or other utilities and applications.
2. Some environment variables are given **preset** values by the system (which can usually be overridden), while others are set directly by the user, either at the command line or within **startup** and other scripts.
3. There are a number of ways to view the values of currently set environment variables; one can type **set, env, or export**.
4. Depending on the state of your system, set may print out many more lines than the other two methods.

```
File  Edit  View  Search  Terminal  Help
c7:/tmp>env | head -5
XDG_VTNR=1
XDG_SESSION_ID=1
HOSTNAME=c7
IMSETTINGS_INTEGRATE_DESKTOP=yes
TERM=xterm-256color
c7:/tmp>
c7:/tmp>set | head -5
ABRT_DEBUG_LOG=/dev/null
BASH=/usr/bin/bash
BASHOPTS=checkwinsize:cmdhist:expand_aliases:extglob:extquote:force_fignore:histapp
end:interactive_comments:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
c7:/tmp>
c7:/tmp>export | head -5
declare -x BITS="64"
declare -x CCACHE_COMPRESS="1"
declare -x CCACHE_DIR="/tmp/.ccache"
declare -x CCACHE_TEMPDIR="/tmp/.ccache"
declare -x CCACHE_UMASK="002"
c7:/tmp>
```

Environment Variables

Setting Environment Variables:

1. By default, variables created within a script are only available to the current shell.
2. **child processes (sub-shells)** will not have access to values that have been set or modified.
3. Allowing child processes to see the values requires use of the export command.
4. You can also set environment variables to be fed as a one shot to a command as in the below. which feeds the values of the **SDIRS** and **KROOT** environment variables to the command make modules_install.
   SDIRS=s_0* KROOT=/lib/modules/$(uname -r)/build make modules_install

Task	Command
Show the value of a specific variable	`echo $SHELL`
Export a new variable value	`export VARIABLE=value` (or `VARIABLE=value; export VARIABLE`)
Add a variable permanently	Edit `~/.bashrc` and add the line `export VARIABLE=value` Type `source ~/.bashrc` or just `. ~/.bashrc` (*dot* `~/.bashrc`); or just start a new shell by typing `bash`

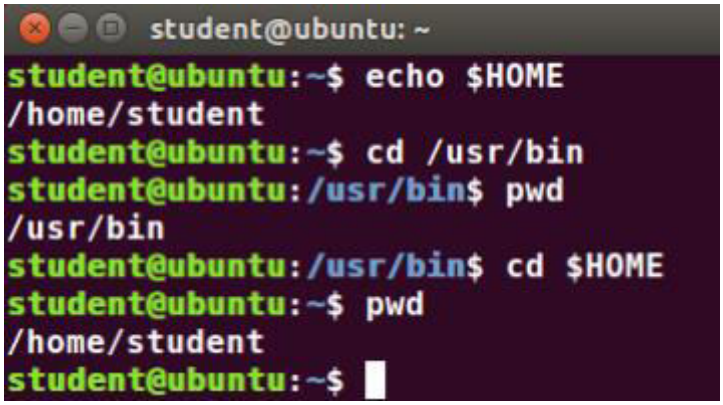
The HOME Variable:

1. **HOME** is an environment variable that represents the home (or login) directory of the user.
2. **cd** without arguments will change the current working directory to the value of **HOME**.
3. Note the tilde character (**~**) is often used as an abbreviation for **$HOME**.
4. Thus, **cd $HOME** and **cd ~** are completely equivalent statements.

Command	Explanation

`$ echo $HOME`
`/home/me`
`$ cd /bin`
	Show the value of the `HOME` environment variable, then change directory (`cd`) to `/bin`.

`$ pwd` `/bin`	Where are we? Use print (or present) working directory (`pwd`) to find out. As expected, `/bin`.
`$ cd`	Change directory without an argument...
`$ pwd` `/home/me`	...takes us back to `HOME`, as you can now see.



The PATH Variable:

1. **PATH** is an ordered list of directories (the path) which is scanned when a command is given to find the appropriate program or script to run.
2. Each directory in the path is separated by colons (**:**).
3. A null (empty) directory name (or **./**) indicates the current directory at any given time.
   - :path1:path2
   - path1::path2
4. In the above example :path1:path2, there is a null directory before the first colon (:). Similarly, for path1::path2 there is a null directory between path1 and path2.
5. To prefix a private bin directory to your path:
   - **export PATH=$HOME/bin:$PATH**
   - **echo $PATH**
     /home/student/bin:/usr/local/bin:/usr/bin:/bin/usr

The SHELL Variable:

1. The environment variable **SHELL** points to the user's default command shell (the program that is handling whatever you type in a command window, usually bash) and contains the full pathname to the shell:
   $ echo $SHELL
   /bin/bash
   $

The PS1 Variable and the Command Line Prompt:

1. Prompt Statement (**PS**) is used to customize your prompt string in your terminal windows to display the information you want.
2. **PS1** is the primary prompt variable which controls what your command line prompt looks like. The following special characters can be included in **PS1**. They must be surrounded in single quotes when they are used, as in the following example:
   \u - User name
   \h - Host name
   \w - Current working directory
   \! - History number of this command
   \d – Date
   $ echo $PS1
   $
   $ export PS1='\u@\h:\w$ '
   student@example.com:~$ # new prompt
3. To revert the above changes
   student@example.com:~$ export PS1='$ '
   $
4. An even better practice would be to save the old prompt first and then restore, as in:
   $ OLD_PS1=$PS1
5. change the prompt, and eventually change it back with:
   $ PS1=$OLD_PS1
   $

```
student@openSUSE:
File  Edit  View  Search  Terminal  Help
student@openSUSE:~> echo $PS1
\[$(ppwd)\]\u@\h:\w>
student@openSUSE:~> OLDPS1=$PS1
student@openSUSE:~> PS1='\u@\h:\d\w>'
student@openSUSE:Wed Dec 14~>cd /tmp
student@openSUSE:Wed Dec 14/tmp>PS1=$OLDPS1
student@openSUSE:/tmp>
```

The PS1 Variable and the Command Line Prompt