– General Linux 2 –
Customise and Use the Shell Environment []

(Linux Professional Institute Certification)

a


```
   .~.
   /V\     by: geoffrey robertson
  // \\        geoffrey@zip.com.au
  @._.@
```


$Id: gl2.109.1.slides.tex,v 1.2 2003/08/29 14:36:14 waratah Exp $

List of Slides

Shells, Scripting, Programming & Compiling

2.109.1 **Customise and use the shell environment []**

2.109.2 Customise or Write Simple Scrips []

Customise and Use the Shell Environment

Objective

Candidate should be able to customise shell environments to meet users' needs. This objective includes setting environment variables (e.g. PATH) at login or when spawning a new shell. It also includes writing bash functions for frequently used sequences of commands.

Customise and Use the Shell Environment

Key files, terms, and utilities

```
~/.bash_profile
~/.bash_login
~/.profile
~/.bashrc
~/.bash_logout
~/.inputrc
function (Bash built-in command)
export
env
set (Bash built-in command)
unset (Bash built-in command)
```


Bash Configuration Files

- When a user logs in to a `bash` shell the following configuration files are usually executed:

Bash Configuration Files

- When a user logs in to a `bash` shell the following configuration files are usually executed:

  `/etc/profile` System wide profile, common to all users and shells

Bash Configuration Files

- When a user logs in to a `bash` shell the following configuration files are usually executed:

  `/etc/profile` System wide profile, common to all users and shells

  `~/.bash_profile` Executed after `/etc/profile` at login

Bash Configuration Files

- When a user logs in to a `bash` shell the following configuration files are usually executed:

  `/etc/profile` System wide profile, common to all users and shells

  `~/.bash_profile` Executed after `/etc/profile` at login

  `~/.bashrc` Executed after `/.bash_profile` at login

Bash Configuration Files

- When a user logs in to a `bash` shell the following configuration files are usually executed:

  `/etc/profile` System wide profile, common to all users and shells

  `~/.bash_profile` Executed after `/etc/profile` at login

  `~/.bashrc` Executed after `/.bash_profile` at login

- Note `/.bashrc` is executed when any new `bash` shell is spawned

Bash Aliases

Bash Aliases

-

Bash Functions

- Functions work similarly to aliases but allow more complex constructions.

Bash Functions

- Functions work similarly to aliases but allow more complex constructions.
- They have the following syntax:

  `$ [ function ] NAME() { COMMAND_LIST;} ↩`

Bash Functions

- Functions work similarly to aliases but allow more complex constructions.
- They have the following syntax:

  `$ [ function ] NAME() { COMMAND_LIST;} ↩`

- Where

  `function` Optional tag

  `NAME()` The name of the function

  `COMMAND_LIST` The body of the function

Bash Functions

- Functions work similarly to aliases but allow more complex constructions.
- They have the following syntax:

  `$ [ function ] NAME() { COMMAND_LIST;} ↩`

- Where

  `function` Optional tag

  `NAME()` The name of the function

  `COMMAND_LIST` The body of the function

- Functions may be stored in `~/.bashrc`

Bash Functions

Function Example

- This simple function prints the current working directory and the list of files in it:

```
$ function look() { pwd; ls;} ↩
```


Bash Functions

Function Example

- This simple function prints the current working directory and the list of files in it:

```
$ function look() { pwd; ls;} ↩
```

- This function would be used like this:

```
$ look ↩
/home/geoffrey/lpic/general-linux-2/notes
CVS     _whizzy_gl2.notes.fmt
_whizzy_gl2.notes.pag
```


Bash Functions

Valid Function Definitions

Bash Functions

Valid Function Definitions

- `$ function look() { pwd; ls;}`

Bash Functions

Valid Function Definitions

- `$ function look() { pwd; ls;}`
- `$ function look { pwd; ls; }`

Bash Functions

Valid Function Definitions

- `$ function look() { pwd; ls;}`
- `$ function look { pwd; ls; }`
- `$ look() { pwd; ls;}`

Bash Functions

Valid Function Definitions

- `$ function look() { pwd; ls;}`
- `$ function look { pwd; ls; }`
- `$ look() { pwd; ls;}`
-
```
$ look()
> {
> pwd;
> ls;
> }
```


Bash Functions

Invalid Function Definitions

Bash Functions

Invalid Function Definitions

- `$ function look()  pwd; ls;`

Bash Functions

Invalid Function Definitions

- `$ function look()  pwd; ls;`
- `$ look() { pwd; ls }`

Bash Functions

Invalid Function Definitions

- `$ function look()  pwd; ls;`
- `$ look() { pwd; ls }`
- `$ function look() {pwd; ls;}`

Bash Functions

Example from Jeffrey Dean's Nutshell Book

Bash Functions

Example from Jeffrey Dean's Nutshell Book

- A function that uses a command line argument:

```
$ laps () { ↩
> ls -l $1
> ps aux | grep '/usr/bin/basename $1'
> }
```


Bash Functions

Example from Jeffrey Dean's Nutshell Book

- A function that uses a command line argument:

```
$ laps () { ↩
> ls -l $1
> ps aux | grep '/usr/bin/basename $1'
> }
```

- Use the `laps()` function:

```
$ laps /usr/sbin/sshd ↩
-rwxr-xr-x    1 root     root       276200 Jun 29 01:28 /usr/sbin/sshd
root       255  0.0  0.3  2792 1216 ?        S    Aug31   0:00 /usr/sbin/sshd
geoffrey  1187  0.0  0.1  1332  424 pts/1    R    14:39   0:00 grep sshd
```