Shell Variables

A shell variable is a character string in a shell that stores some value. It could be an integer, filename, string, or some shell command itself. Basically, it is a pointer to the actual data stored in memory.

Rules for variable definition

A variable name could contain any alphabet (a-z, A-Z), any digits (0-9), and an underscore (_). However, a variable name must start with an alphabet or underscore. It can never start with a number.

Valid Variable Names

Xyz
_x
XY_2Z

Invalid variables name:

```
2_AN
!ABD
$ABC
```

Eg:

Num=5

Name=”Unix”

Accessing variable

Variable data could be accessed by appending the variable name with ‘$’

Variable Types

three main types of variables:

1) Local Variable:

Variables which are specific to the current instance of shell. They are basically used within the shell, but not available for the program or other shells that are started from within the current shell.

For example:

`name=Roy`

In this case the local variable is (name) with the value of Roy. Local variables is temporary storage of data within a shell script.

2) Environment Variable:

These variables are commonly used to configure the behaviour script and programs that are run by shell. Environment variables are only created once, after which they can be used by any user.

For example:

`export PATH=/usr/local/bin:$PATH` would add `/usr/local/bin` to the beginning of the shell's search path for executable programs.

3) Shell Variables:

Variables that are set by shell itself and help shell to work with functions correctly. It contains both, which means it has both, some variables are Environment variable, and some are Local Variables.

For example:

`$PWD` = Stores working directory

`$HOME` = Stores user's home directory

`$SHELL` = Stores the path to the shell program that is being used.

Metacharacters/*Quoting Mechanisms in Unix*

These are the special characters that are first interpreted by the shell before passing the same to the command. They are also known as shell wildcards.

- > used for Output Redirection.

- **>>** used for Output Redirection to append.
- **<** Input redirection.
- **<<** used for input redirection and is also known as here document.
- * Match any number of characters, Substitution wildcard for zero or more characters
- **?** Match one character, Substitution wildcard for 1 character
- **[]** Match range of characters, Substitution wildcard for any character between brackets
- **`cmd`** Replace cmd with the command to execute and will execute that, Substitution wildcard for command execution
- **$(cmd)** Replace cmd with the command to execute and will execute that, Substitution wildcard for command execution
- **|** Pipe is a Redirection to send the output of one command/program/process to another command/program/process for further processing.
- **;** Command separator is used to execute 2 or more commands with one statement.
- **||** OR conditional execution of the commands.
- **&&** AND conditional execution of the commands.
- **()** Groups the command in to one output stream.
- **&** executes command in the background and will display the assigned Pid.
- **#** to comment something.
- **$** To expand the value of a variable.
- \ used to escape the interpretation of a character or to prevent that.