Scripting variables

Dr Peadar Grant

March 6, 2024

Contents

1 Shell Variables

- Variables are used to store values in a script.
- You can use shell variables when in Bash interactively too!
- Can be used more than once and can be reassigned a value.
- BASH has two main types of Variables:

Environment variables that contain information that the system and programs access regularly

User-defined variables defined by users

2 Environment Variables

- Examples of Environment Variables include `$HOME`, `$HOSTNAME`, `$PWD`, `$SHELL`
- To view the available Environment Variables, use the `printenv` command
- To view environment variable value, use `echo` (which outputs to screen):

  ```
  echo $HOSTNAME
  ```

- You can also use the `set` and `env` commands to view the environment variables and their values

3 PATH Environment Variable

- The PATH variable is one of the most important variables in the BASH shell
- Allows users to execute commands by typing the command name alone
- If a command is located within a directory that is listed in the PATH variable, You can type the name of the command on the command line to execute it
- To view contents of PATH variable:
  - Interactively: `$PATH`
  - Or in script `echo $PATH`

4 User Defined Variables

- Variable Names are important and should reflect the name of the value where possible.
- No need to declare a variable, just assign a value to it (same as Python)
- To assign a value to a variable use the =equal sign:

```
college=dkit
```

- Here we have created a **variable** called **college** and assigned (**=)**) the value **dkit** to it
- To use a variable value, use the $ sign in front of the variable:

```
echo ``The college name is'' \$college
```

4.1 Variable names

- Variables can contain alphanumeric characters, the dash character, or the underscore character.
- They must not start with a number.
- Better not to use UPPERCASE.
- Decide on appropriate naming convention i.e. surName or firstName or fname and stick with this convention.
- To display the values of the variables, put the **$** sign infront of the variable.

```
fname=Mary
sname=Jones
echo "Hello $fname $sname"
```

4.2 Assigning Values to Variables

3 ways to do this:

Direct assignment (like we just saw i.e. college=dkit)

User Input Prompt similar to `input`

Positional Parameters (arguments)

5 User Input

- Shell scripts may require input from a user.
- This input can be stored in a variable to be used later.
- The `read` command is used to take input from standard input and place it in a variable.

```
echo 'enter your name: '
read fname
echo ”your name is $fname”
```

6 Positional Parameters (Arguments)

- When we want to *specify values on the command line* , we use **positional parameters or arguments** .
- Positional Parameters are a series of *special variables ($0. . . $9)* that contain the contents of the values specified on the command line.
- So for example if you want to read in the first name and second name of a person, *you would pass the values while running the script.*
- So if we write a script to echo greetings to a user and we want to pass the values for the user at run time, we replace the variable value with `$1`, `$2`, etc i.e.
- *echo "Welcome" $1 $2*
- Then run the script and pass the values for $1 and $2 when running the script.
- *./script.sh liz ␣␣ frances ␣␣*

7 Echo

The echo command is used to add descriptions and spaces to script output:

```
college=dkit
echo "the college name is $college
```