**Script Files**

Table of Contents

[Common Commands 3](#_Toc83133029)

[Special Variables 4](#_Toc83133030)

**Shell scripts** are computer programs that are designed to run on the **UNIX shell**. Essentially, these are just files with a **.sh extension** that hold the same commands that we run on the terminal. When the files are executed, the commands run.

Every shell script file must begin with the following line:

*#!/bin/sh*

BASH

This tells the computer that this is a shell script file. Similarly, we could also use this line instead:

*#!/bin/bash*

BASH

This tells the computer that this is a **bash script**. There are multiple types of shell script files, of which bash scripts are one. They are also probably the most famous.

## Common Commands

|  |  |  |
| --- | --- | --- |
| **Description** | **Command** | **Notes** |
| **Display** text. | echo "Hello World!" |  |
| Store data in a **variable**. | someVar="Some Value" | There should be **no space** on either side of the assignment operator. |
| **Use** a variable. | echo "Value: $someValue" |  |
| Read user **input** into a variable. | read anotherVar |  |
| Make a variable **read-only**. | readonly someVar |  |
| **Delete** a variable. | unset someVar |  |

## Special Variables

There are some special variables that can be used which give us special data:

|  |  |
| --- | --- |
| **Variable** | **Description** |
| $$ | **Process ID** of the shell script’s process. |
| $0 | The script’s **file name**. |
| $1, $2,… | Specific **arguments** provided during execution |
| $# | **Number of arguments** provided during execution. |
| $@ | Arguments should be treated as **array**. Each argument is inside **double quotes**. |
| $\* | All arguments are provided inside a **single double quote pair**, separated by spaces. They should be treated as an **array**. |
| $? | **Exit status** of the **last command**. |
| $! | **Process number** of the **last background command**. |

The part about arguments provided during execution may be a little confusing, so an example should help. Suppose we have a number of strings provided as arguments inside a single double quote pair when executing the script file through the terminal. As stated above, these can be accessed as an **array** using the **$\*** variable.

for str in $\*  
do  
 echo$str  
done

BASH