



Bash Scripting Part 2

P. TenHoopen - WMLUG



Coloring Echo Output

You can colorize the output from the Echo command by using ANSI escape sequences.

For example,

```
echo -e "This word is \e[0;32mgreen\e[0m."
```

outputs This word is **green**.



Coloring Echo Output

The `-e` option lets Echo interpret the ANSI escape sequences which have a beginning and ending tag.

The `\e[0;32m` tells Echo to start coloring text in green.

The `\e[0m` tells Echo to stop coloring and revert back to normal output.



Coloring Echo Output

Beginning sequence:

| | |
|-----------------|---------------------------------|
| <code>\e</code> | Escape character |
| <code>[</code> | Beginning tag |
| <code>0</code> | Foreground attribute |
| <code>;</code> | Attribute separator |
| <code>32</code> | Background attribute (32=green) |
| <code>m</code> | Ending tag |



Coloring Echo Output

Foreground attributes:

- 0 Normal
- 1 Bold
- 4 Underlined
- 5 Blinking
- 7 Inverted

There are no background attributes.



Coloring Echo Output

Foreground colors:

| Color Code | 00 | 01 |
|------------|-----------|-------------|
| 30 | Black | Gray |
| 31 | Red | Light Red |
| 32 | Green | Light Green |
| 33 | Brown | Yellow |
| 34 | Blue | Light Blue |
| 35 | Magenta | Pink |
| 36 | Cyan | Light Cyan |
| 37 | White | Light Gray |



Coloring Echo Output

Ending sequence:

| | |
|-----------------|----------------------|
| <code>\e</code> | Escape character |
| <code>[</code> | Beginning tag |
| <code>0</code> | Foreground attribute |
| <code>m</code> | Ending tag |



Coloring Echo Output

To change the foreground and background color, just add the attribute for the background color after the foreground color.

For example:

```
echo -e "This is \e[0;31m\e[42mred on green\e[0m."
```



Exit Status

The `exit` command can be used to exit a script.

Commands return an exit status on completion. This is also referred to as the exit code or return status. An exit status of 0 indicates success while a non-zero value indicates failure.

The `$?` variable stores the exit code of the last command executed and can be used to check for errors.



IF Statement

The `IF` statement checks for a true condition and, if it has one, it performs the commands listed.

The formats are:

```
if condition-true
then command 1
    command 2
fi
```

```
if condition-true; then
    command 1
    command 2
fi
```



IF Statement

Some other forms:

```
if condition-true
then command 1
else
    command 2
fi
```

```
if condition-1-true
then command 1
else if condition-2-true
    command 2
else
    command 3
fi
```



IF Statement

Some examples:

```
if grep -q Linux MyFile.txt
then echo "The file contains the word Linux"
fi
```

```
if grep -q Linux MyFile.txt
then echo "The file contains the word Linux"
else echo "The file does not contain the word Linux"
fi
```



Test Command

The test command is a Bash command that tests file types and compares strings.

Example:

```
if test "5" = "7"
then echo "5 equals 7"
else
    echo "5 does not equal 7"
fi
```



Test Command

Using the `[]` brackets surrounding a condition in an `if` statement invokes the `test` command.

Therefore, this is equivalent to the above code:

```
if [ "5" = "7" ]
then echo "5 equals 7"
else
    echo "5 does not equal 7"
fi
```



Test Command Uses

Testing if a file exists:

```
if [ -e file.txt ]
```

Testing if the current user has write privileges to a file:

```
if [ -w file.txt ]
```

Testing whether a string is null:

```
if [ -n "$var" ]
```



Putting it all together

```
#!/bin/bash

# Script for demonstrating echo output coloring, if statement, and test command

# clear the terminal window
clear

# demo echo color coding

echo "Greetings, $USER"
echo ""
echo -e "and \e[1;32m\e[41mHappy Holidays\e[0m too!"
echo ""
echo -e "I hope you had a \e[5mspooky\e[0m \e[0;33mHalloween\e[0m!"
echo ""

# demo the if statement and test command

echo "Testing for the existence of file textfile1.txt..."
echo ""

if test -e textfile1.txt; then
    echo "The file textfile1.txt exists."
    echo ""
else
    echo -e "\e[0;31mThe file textfile1.txt doesn't exist.\e[0m"
    echo ""
fi

echo "Now, testing to see if the text files in the current directory contain the word demo..."
echo ""

if grep -q demo *.txt 2> /dev/null; then
    echo "There are text files in the current directory ($PWD) that contain the word \"demo\""
else
    echo "Sorry, there are no text files in the current directory ($PWD) that contain the word \"demo\""
fi

echo ""
echo "Testing to see if you can modify /var/log/messages..."
echo ""

SUCCESS="\e[0;32m"
ERROR="\e[0;31m"
NORMAL="\e[0m"

if [ -w /var/log/messages ]; then
    echo -e "$SUCCESS Congratulations, you have write privileges to /var/log/messages.$NORMAL"
else
    echo -e "$ERROR Sorry, you don't have write privileges to /var/log/messages.$NORMAL"
fi

echo ""
```