Bash If Else – Syntax & Examples

Bash If Else Statement

Bash If Else : If-else statement is used for conditional branching of program (script) execution between two paths.

An expression is associated with the if statement. If the expression evaluates to true, statements of if block are executed. If the expression evaluates to false, statements of else block are executed.

In this tutorial, we will go through Syntax and usage of **if-else** statement with examples.

Bash If Else is kind of an extension to Bash If statement.

Syntax of Bash If Else

Syntax of **If Else** statement in Bash Shell Scripting is as given below :

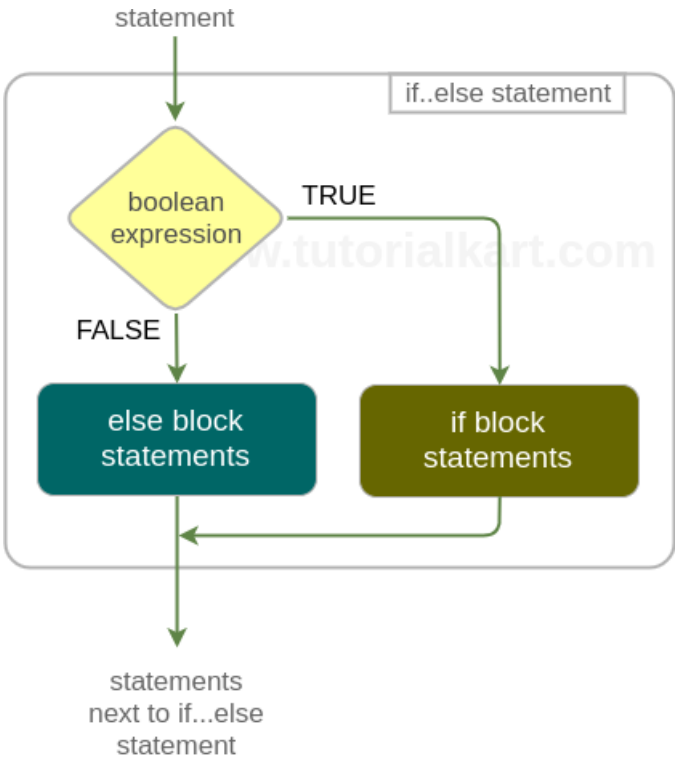
```
if <expression>; then
    <commands>
else
    <other_commands>
fi
```

where

Code	Description
<expression>	Set of one or more conditions joined using conditional operators.
<commands>	Set of commands to be run when the <condition> is true.
<other_commands>	Set of commands to be run when the <condition> is false.

Observe that `if` , `then` , `else` and `fi` are keywords. The semi-colon (;) after the conditional expression is must.

statements
previous to
if...else



Example 1 : Bash If Else Statement

In this example, we shall look into two scenarios where in first if-else statement, expression evaluates to true and in the second if-else statement, expression evaluates to false.

Bash Shell Script

```
#!/bin/bash

# if condition is true
if [ "hello" == "hello" ];
then
    echo hello == hello : is true condition
else
    echo hello == hello : is false condition
fi

# if condition is false
if [ "hello" == "bye" ];
then
    echo hello == bye : is true condition
else
    echo hello == bye : is false condition
fi
```

Output

```
~$ ./bash-if-else-example
hello == hello : is true condition
hello == bye : is false condition
```

In the first if-else expression, condition is true and hence the statements in the if block are executed. Whereas in the second if-else expression, condition is false and hence the statements in the else block are executed.

Example 2 : Bash If Else – Multiple Conditions

In this example, we shall look into a scenario where if expression has multiple conditions. Multiple conditions in an expression are joined together using bash logical operators.

Bash Shell Script

```
#!/bin/bash

# FALSE && TRUE || FALSE || TRUE evaluates to TRUE
if [[ 8 -eq 11 && "hello" == "hello" || 1 -eq 3 ]];
then
    echo "True condition"
else
    echo "False condition"
fi
```

Output

```
~$ ./bash-if-else-multiple-conditions-example
False condition
```

The expression after if keyword evaluated to false. Therefore program execution skipped if block statements and executed else block statements.

Bash If Else Statement in One Line

If Else statement along with commands in if and else blocks could be written in a single line.

To write complete if else statement in a single line, follow the syntax that is already mentioned with the below mentioned edits :

- End the statements in if and else blocks with a semi-colon (;).
- Append all the statements with a space as delimiter.

In the following example, we will write an if else statement in a single line.

Bash Shell Script

```
#!/bin/bash

if [ "hello" == "hello" ]; then echo "statement 1"; echo "statement 2"; else echo "state
```

```
if [ "hello" == "hi" ]; then echo "statement 5"; echo "statement 6"; else echo "statemen
```

Output

```
~$ ./bash-if-else-single-line
statement 1
statement 2
statement 7
statement 8
```

Bash Nested If Else

If Else can be nested in another if else. Following is an example for the same.

Bash Shell Script

```
#!/bin/bash

if [ "hello" == "hello" ]; then
    if [ "hello" == "hi" ]; then
        echo "statement 1";
    else
        echo "statement 2";
    fi
else
    echo "statement 3";
fi
```

According to the expressions, `statement 2` should be echoed to the console. Let us see the output.

Output

```
statement 2
```

if, if-else and else-if statements could be nested in each other.

Conclusion

In this Bash Tutorial, we have learnt about the syntax and usage of **bash if else** statement with example bash scripts.

Functions

- Bash Function
- Bash Override Buitlin Commands

Arrays

- Bash Array

Files

- Bash Write to File
- Bash Read File
- Bash Read File line by line
- Bash If File Exists
- Bash If File is Directory
- Bash If File is Readable

Bash Others

- Bash Check if variable is set