Bash If Else

Many times, you need to execute a certain set of commands based on condition and skip others under different conditions. In that situation, use Bash `if else` statement.

Whenever you run a command, it returns exit status code. This exit status code of the command is used by the `if` statement to execute a block of statements.

In Bash, zero(0) is considered true and non-zero value is considered false. If you want to check condition in `if else` then use Bash test command or square brackets([]). There are 4 different forms of if statements in Bash-

1. Simple if statement
2. if-then-else statement
3. else if ladder
4. Nested if statements

Bash if then statement

```
if command
then
   commands
fi
```

Another syntax of Bash if then statement

```
if command; then
   commands
fi
```

If the command runs successfully, it will return 0 as the exit status code which is considered true. And all the commands written between `then` and `fi` are executed. If the exit status of the command is non-zero, then the commands mentioned between `then` and `fi` are skipped.

Bash if-then-else statement

```
if command
then
   commands
else
   commands
fi
```

If the command runs successfully and returns an exit status code of zero, then the commands listed between `then` and `else` will be executed. And if the command returns a non-zero exit status code, then the commands written in the `else` section is executed.

Bash else if ladder

```
if command1
then
   commands
elif command2
then
   commands
elif command3
then
   commands
fi
```

Bash executes `if` statements sequentially and any command present in the `if` statement returns a zero exit status code, a block of commands written just after the immediate `then` will be executed and all other statements are skipped.

Bash Nested if statement

```
if command1
then
   commands
else
   if command2
   then
      commands
   else
      commands
   fi
fi
```

You can write `if` statement inside `else` block and `if` block and thus nested any number of `if` statement.

> **Note:** Instead of using exit status code of a command Bash provides testcommand that enables you to evaluate any condition.