LUA - IF...ELSE STATEMENT

http://www.tutorialspoint.com/lua/if_else_statement_in_lua.htm

An **if** statement can be followed by an optional **else** statement, which executes when the Boolean expression is false.

Syntax

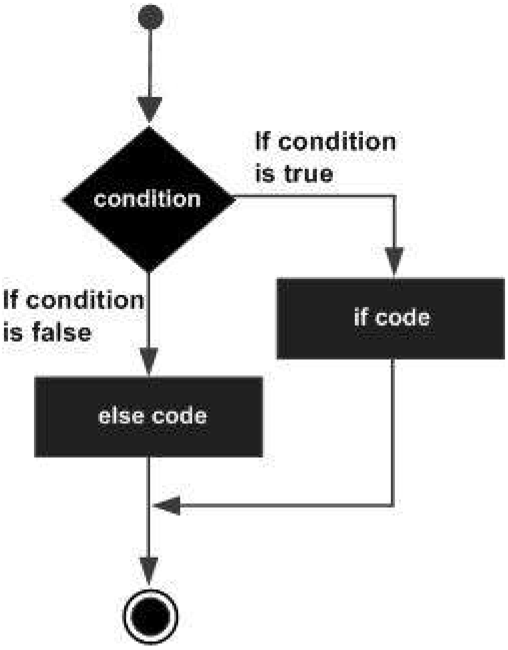
The syntax of an **if...else** statement in Lua programming language is −

```
if(boolean_expression)
then
   --[ statement(s) will execute if the boolean expression is true --]
else
   --[ statement(s) will execute if the boolean expression is false --]
end
```

If the Boolean expression evaluates to **true**, then the **if block** of code will be executed, otherwise **else block** of code will be executed.

Lua programming language assumes any combination of Boolean **true** and **non-nil** values as **true**, and if it is either Boolean **false** or **nil**, then it is assumed as **false** value. It is to be noted that in Lua, zero will be considered as true.

Flow Diagram



Example

```
--[ local variable definition --]
a = 100;

--[ check the boolean condition --]

if( a < 20 )
then
   --[ if condition is true then print the following --]
   print("a is less than 20" )
else
   --[ if condition is false then print the following --]
   print("a is not less than 20" )
end
```

```
print("value of a is :", a)
```

When you build and run the above code, it produces the following result.

```
a is not less than 20
value of a is : 100
```

The if...else if...else Statement

An **if** statement can be followed by an optional **else if...else** statement, which is very useful to test various conditions using single if...else if statement.

While using if , else if , else statements, there are a few points to keep in mind −

- An *if* can have zero or one else's and it must come after any else if's.
- An *if* can have zero to many else if's and they must come before the else.
- Once an *else if* succeeds, none of the remaining else if's or else's will be tested.

Syntax

The syntax of an **if...else if...else** statement in Lua programming language is −

```
if(boolean_expression 1)
then
   --[ Executes when the boolean expression 1 is true --]

else if( boolean_expression 2)
   --[ Executes when the boolean expression 2 is true --]

else if( boolean_expression 3)
   --[ Executes when the boolean expression 3 is true --]
else
   --[ executes when the none of the above condition is true --]
end
```

Example

```
--[ local variable definition --]
a = 100

--[ check the boolean condition --]

if( a == 10 )
then
   --[ if condition is true then print the following --]
   print("Value of a is 10" )
elseif( a == 20 )
then
   --[ if else if condition is true --]
   print("Value of a is 20" )
elseif( a == 30 )
then
   --[ if else if condition is true  --]
   print("Value of a is 30" )
else
   --[ if none of the conditions is true --]
   print("None of the values is matching" )
end
print("Exact value of a is: ", a )
```

When you build and run the above code, it produces the following result.

```
None of the values is matching
Exact value of a is: 100
```