synthiam.com

If Elseif Else Goto & Return Commands

Another one which was written and hidden within a different topic and should warrant it's own tutorial post, the IF (etc.) commands.
Quote:
If (Value Condition Value)
IF condition can support multiple comparisons and functions.
Condition tree must be closed with an ENDIF
An if command requires a condition, this is the part in brackets after it. Think of it logically, it's the same as you think when you decide IF you want to do something or not. So, for instance, you go to...

Last Updated: 9/17/2015

Step 1

Another one which was written and hidden within a different topic and should warrant it's own tutorial post, the IF (etc.) commands.
Quote:
If (Value Condition Value)
IF condition can support multiple comparisons and functions.
Condition tree must be closed with an ENDIF
An if command requires a condition, this is the part in brackets after it. Think of it logically, it's the same as you think when you decide IF you want to do something or not. So, for instance, you go to the supermarket, you walk past the apples and need to decide IF you need any. You check how many you have at home (not sure how, I didn't think this through well but we will carry on as it's unimportant), how many you need and decide.

```
IF($ApplesAtHome < $ApplesNeeded)
Buy some apples
EndIf
```

This says, if you have less than the amount of apples you need at home then you need to buy some apples. Otherwise do nothing. Basically, whatever is between the IF and the EndIf lines is only executed IF the condition is met or true. Otherwise it is ignored.
ElseIf works in a similar way but is part of a bigger condition. Say you wanted to know IF you needed to buy a lot apples or needed to buy just a few apples.

```
IF($ApplesAtHome = 0)
Buy lots of apples
ElseIf($ApplesAtHome < $ApplesNeeded)
Buy a few apples
EndIf
```

You could change it so it works out how many apples to buy with some arithmetic functions but I will save that for another time. One function at a time is easiest to learn.
You can have as many ElseIfs as you need. It's only the part between the first ElseIf and the next ElseIf line or EndIf (if it's last) that is executed, the rest is ignored.
Now say only needed apples provided you had no bananas, this is where the AND and OR conditions come in to play. Again, think logically. If I have no bananas and have no apples then I need some apples...

```
IF($BananasAtHome = 0 and $ApplesAtHome < $ApplesNeeded)
Buy apples
EndIf
```

Only if both parts of the condition are met will the command between the IF and EndIf be executed.
Or is another condition that can be used. For instance, you now look at the pears, but you only need to buy pears IF you have no apples or have no bananas.

```
IF($apples = 0 OR $bananas = 0)
Buy some pears
EndIf
```

All of the AND, OR, <, >, =, <=, >= and <> conditions can be used in all IF statements and all ElseIf statements. You can also mix and match the ANDs and ORs. Phrase the condition correctly, in the order you would phrase it to a person or in your head...

```
IF($apples = 0 OR $bananas = 0 AND $pears < 5)
Buy some pears
ElseIf($apples = 0 AND $bananas = 0 or $Apples = 0 AND $pears < 2)
Buy a mango
EndIf
```

In this example, if you have no bananas or apples and less than 5 pears you will buy some pears. If you have no apples and no bananas, or no apples and less than 2 pears you will buy a mango. Otherwise you buy nothing.
Else is slightly different, else doesn't have a condition. Think of the last example and when you buy nothing. Else fills in this gap. Rather than buying nothing, or executing nothing, if no condition is met something else can be executed or bought.

```
IF($apples = 0 OR $bananas = 0 AND $pears < 5)
Buy some pears
ElseIf($apples = 0 AND $bananas = 0 or $Apples = 0 AND $pears < 2)
Buy a mango
Else
You have too much fruit, buy beer
EndIf
```

In this example, if you have no bananas or apples and less than 5 pears you will buy some pears. If you have no apples and no bananas, or no apples and less than 2 pears you will buy a mango. Otherwise you buy beer.
Goto and Return
Quote:
Goto(label)
Goto a specific :Label location
Quote:
Return()
Return from a Goto()
If you jump to a position of code with a Goto(), the Return statement will allow you to return back to that piece of code following the last Goto()
If you attempt to Return() with an empty stack, nothing will happen. The script will ignore the Return() statement.
The Goto command is one I use a lot as part of IF statements. Rather than re-writing the same code over and over I set these up as smaller "sub-routines" and call them with Gotos, returning back once run with the Return command.
Following on the same examples. Your wife tells you to GOTO the shop

```
```

Goto(shop)

```

So you go to the shop. Or in the script, jump to the label called shop.

```
:shop
```

You then carry on following whatever instructions are after that label. Let's say it's the above IFs. You keep following line by line ignoring anything that any IF conditions are not met.
At the end of it you need to RETURN home. So a return command is used.

```
Return()
```

This command will go back the last Goto command which was executed, and then you carry on following instructions line by line.
So, an example with line numbers;

```

1 Print("Hello world")
2 If($day = "Monday")
3   Goto(monday)
4 ElseIf($day = "Friday")
5   Goto(friday)
6 ElseIf($day = "Saturday" or $day = "Sunday")
7   Goto(weekend)
8 Else
9   Goto(end)
10 EndIf
11 Halt()
12 :monday
13 Print("It is Monday, Boo!")
14 Return()
15 :friday
16 Print("It is Friday, the weekend is almost here!")
17 Return()
18 :weekend
19 Print("It is the weekend, yay!")
20 Return()
21 :end
22 Print("Today isnt important - terminating")
23 Halt()

```

In this example code, if it's a Monday it will execute the lines of code in this order;
1, 2, 3, 12, 13, 14, 10, 11
If it's a Friday it will execute in this order;
```

1, 2, 4, 5, 15, 16, 17, 10, 11
If it's a weekend;
1, 2, 4, 6, 7, 18, 19, 20, 10, 11
If it's any other day;
1, 2, 4, 6, 8, 9, 21, 22, 23
Take a look at my Ping Roam script for some IF tree examples which use Gotos and Returns (and in some cases need Gotos in order to return). They are well documented so should prove to be useful examples.
As always, any questions, comments, suggestions etc. are more than welcome.