if

[Control Structure]

Description

The if statement checks for a condition and executes the proceeding statement or set of statements if the condition is 'true'.

Syntax

if (condition) {

//statement(s)

}

Parameters

condition: a boolean expression (i.e., can be true or false).

Example Code

The brackets may be omitted after an if statement. If this is done, the next line (defined by the semicolon) becomes the only conditional statement.

if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)

digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LEDpin, HIGH);}

if (x > 120) {

digitalWrite(LEDpin1, HIGH);

digitalWrite(LEDpin2, HIGH);

}

// all are correct

Notes and Warnings

The statements being evaluated inside the parentheses require the use of one or more operators shown below.

**Comparison Operators:**

x == y (x is equal to y)

x != y (x is not equal to y)

x < y (x is less than y)

x > y (x is greater than y)

x <= y (x is less than or equal to y)

x >= y (x is greater than or equal to y)

Beware of accidentally using the single equal sign (e.g. if (x = 10) ). The single equal sign is the assignment operator, and sets x to 10 (puts the value 10 into the variable x). Instead use the double equal sign (e.g. if (x == 10) ), which is the comparison operator, and tests *whether* x is equal to 10 or not. The latter statement is only true if x equals 10, but the former statement will always be true.

This is because C++ evaluates the statement if (x=10) as follows: 10 is assigned to x (remember that the single equal sign is the ([assignment operator](http://arduino.cc/en/Reference/Assignment))), so x now contains 10. Then the 'if' conditional evaluates 10, which always evaluates to TRUE, since any non-zero number evaluates to TRUE. Consequently, if (x = 10) will always evaluate to TRUE, which is not the desired result when using an 'if' statement. Additionally, the variable x will be set to 10, which is also not a desired action.