Cheatography

RTTMbot Cheat Sheet

by lucas_su via cheatography.com/116479/cs/21767/

Control structures

```
if(condition)
{
// if condition is TRUE, do something here
}
else
{
// otherwise, do this*
}
while (condition)
{
// while condition is true, do this
}
break; // break a loop
continue; // continue to next iteration
switch (variable)
{
case 1:
    // if case == variable, do this
    break;
case 2:
    // if case == variable, do this
    break;
}
for(initialization; condition; increment)
{
// do this
}
/* The 'for' statement is used to repeat
a block of statements enclosed in curly
braces. An increment counter is usually
used to increment and terminate the loop.
/*
```

RTT Import

To give access to functions to control the MBot, import the RTT library.

```
#include "rttimport.h"
```

The following functions are available after including the RTT library.

```
drive(int speed);
// drive forward with speed speed, which may range from 1-100
leftWheel(int speed);
// move left wheel with speed speed
rightWheel(int speed);
// move right wheel with speed speed
stopDrive();
// stop moving
lineSensor.readSensors()
/* read both sensors
returns 0x00 if both left and right sensors are over a black line
returns 0x01 if left sensor is over a black line but the right sensor is not
returns 0x02 if left sensor is not over a black line but the right sensor is
returns 0x03 if neither left sensor nor the right sensor is over a black line */
lineSensor.readSensorLeft()
// read state of left sensor
// returns 0 if over black line, returns 1 if not over black line
lineSensor.readSensorRight()
// read state of right sensor
// returns 0 if over black line, returns 1 if not over black line
```

Operators

```
Mathematical Operators
= // assignment
+ // addition
- // subtraction
// multiplication*
/ // division
% // modulus
Logical Operators
== // boolean equal to
!= // not equal to
< // less than
> // greater than
<= // less than or equal to
>= // greater than or equal to
&& // Boolean AND
|| // Boolean OR
! // Boolean NOT
```

Time

```
delay(time_ms);
/ Pauses the program for the amount of time*
(in milliseconds). */
delayMicroseconds(time_us);
/ Pauses the program for the amount of time*
(in microseconds). */
millis();
/ Returns the number of millis-econds since*
the board began running the current program.
max: 4,294,967,295 */
micros();
/ Returns the number of micros-econds since*
the board began running the current program.
max: 4,294,967,295 */
```