Motor Control Functions...

It's time to make Wink go! Controlling the motors is super easy. The motors can be controlled by a few different functions. GoForward(), GoBackward(), BeStill(), and Motors(). You will use Motors() most of the time as it is most powerful. Let's talk about Motors() first, then we'll cover the other functions in a few minutes.

```
Motors(left,right);    //the Motors() function needs a left motor speed and a right
                       //motor speed. The values can range from 0 to 255. Positive
                       //numbers make motors go forward. Negative numbers make motors
                       //go backward.
```

Let's look at a real example and see what happens.

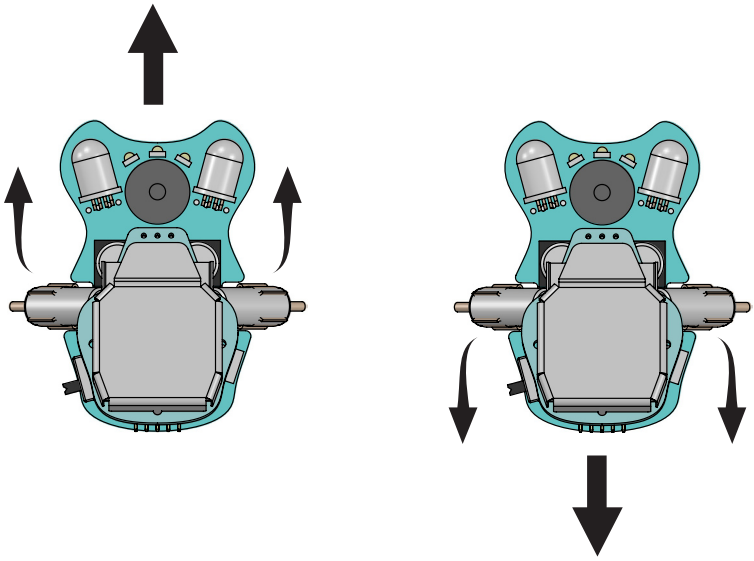
```
void loop(){
  Motors(150,150);          //Both motors forward at 150 speed
  delay(500);               //wait 1/2 second
  Motors(-150,-150);        //Both motors backward at 150 speed
  delay(500);               //wait 1/2 second then loop
}
```

Wink_Ch03Motors_Ex01

Both left and right motors run forward at speed 150 for 1/2 second, then they both go backward at speed 150 for 1/2 second. This loops over and over.

Now lets discuss what is happening. Wink moves whenever his motors turn. If the left and right both turn in the forward direction, then Wink will move forward.

If both motors turn in the reverse direction, then Wink will move backward.



Making Wink turn...

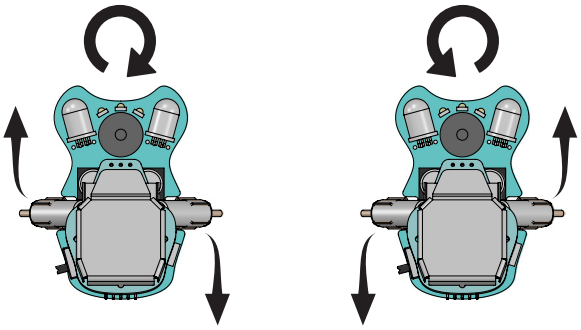
So we see what happens when both motors turn the same speed. Wink goes in a mostly straight direction. If we run the motors in oppsite directions, Wink will spin in his own footprint.

```
void loop(){
  Motors(250,-250);        //Spin to the right
  delay(3000);             //wait 3 seconds
  Motors(-250,250);        //Spin to the left
  delay(3000);             //wait 3 seconds
}
```

Wink_Ch03Motors_Ex02

The positive left number moves the left side forward and the negative right number moves the right side backward.

The opposite is also true.



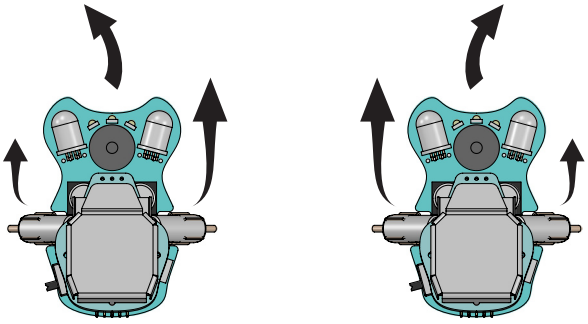
With this next example, be careful that Wink doesn't jump off your table. If we drive both motors the same direction, but at different speeds, Wink will drive forward in an arc movement. He can drive in a backward in an arc movement as well by using two different negative numbers.

```
void loop(){
  Motors(200,100);         //Arc to the right
  delay(3000);             //wait 3 seconds
  Motors(100,200);         //Spin to the left
  delay(3000);             //wait 3 seconds
}
```

Wink_Ch03Motors_Ex03

If both numbers are positive but different values, then one side will drive faster than the other, so Wink will drive in an arc.

Run this code and see what happens. Be careful wink doesn't jump off your table.



Other simple motor functions...

We mentioned you can also use GoForward(), GoBackward(), and BeStill() to control the motors. These functions do exactly what you would expect. Have a look at this example.

```
void loop(){
  GoForward(150,0);        //Arc to the right
  delay(2000);             //wait 2 seconds
  GoBackward(150,40);      //Spin to the left
  delay(2000);             //wait 2 seconds
  BeStill();               //make both motors stop
  delay(2000);             //wait 2 seconds
}
```

Wink_Ch03Motors_Ex04

Go forward at speed 150, with 0 turn for 2 seconds. Then go backward at speed 150 with a 40 turn for 2 seconds, then be still for 2 seconds before looping.

GoForward() and GoBackward() each take two "arguments". These are the two numbers you pass into the function to tell it how to behave. The first number is the speed, and the second number is a turn strength. The turn strength introduces a speed difference between the motors which will case Wink to curve. If you pass a positive number for turn strength, then Wink will turn to the right (also called "clockwise"). If you pass a negative number for turn strength, Wink will turn to the left (also called "counter-clockwise").

You can also make Wink rotate in his own footprint by using SpinRight() and SpinLeft(). Pass either of these functions a speet between 0 and 255 to set the spin speed. The Spin functions run the motors in opposite directions.

```
void loop(){
  SpinRight(200);          //spin right at speed 200
  delay(2000);             //wait 2 seconds
  SpinLeft(200);           //spin left at speed 200
  delay(2000);             //wait 2 seconds
}
```

Wink_Ch03Motors_Ex05

Use the Spin functions to make Wink turn in his own footprint. This is done by running the motors in opposite directions.

A few more notes on motors...

If you do a bit of experementing, you will likely realize that even if you set both motors to the same speed in the same direction, Wink doesn't travel in an exactly straight line. He will likely tend to curve off to one side or the other. This is becasue mechanical systems are never perfect. One motor will always produce a bit more power than the other, and one motor will always get a bit more traction on the surface - so one side will always tend to run a little stronger than the other. This can be adjusted in various ways as you'll see in upcoming chapters.