```
#include <CPutil.h>


// Simple sketch to just test a motor

// Define our pins
const int leftMotorPin1 = 7;
const int leftMotorPin2 = 6;
const int rightMotorPin1 = 5;
const int rightMotorPin2 = 4;
const int button = 10;

void setup()
{
  // put your setup code here, to run once:
  Serial.begin(9600);

  setupMessage(__FILE__, "Simple Motor Test sketch");
  delay(500);

  motorSetup();
}




void loop()
{

  drivingControl();
}

void drivingControl()
{
  enum {START, FORWARD, RIGHT, LEFT, BACKWARDS, SPIN, STOP};
  static int state = START;
  int returnValue = false;

  switch(state)
  {
    case START:
    if (button.wasPushed() == true)
    {
      robotForward(250, 3000);
      state = FORWARD;
      returnValue = false;
    }
    break;

    case FORWARD:
    if (returnValue == false)
    {
      robotRight(2000);
      returnValue = true;
      state = RIGHT;
    }
    else
    {
      robotLeft(3000);
      returnValue = false;
      state = LEFT;
    }
    break;

    case RIGHT:
    if (returnValue == true)
    {
      robotForward(250, 5000);
      state = FORWARD;
    }
    break;

    case LEFT:
      robotBackwards(250, 1000);
      state = BACKWARDS;
    break;

    case BACKWARDS:
      robotSpin(500);
      state = SPIN;
    break;

    case SPIN:
      robotStop();
      state = STOP;
    break;

    case STOP:
      state = START;
    break;
```

```
  }
}

void motorTest()
{

 while (robotSpin(1000))
 {
 }
  delay(1000);
}

void motorSetup()
{

   // Initalize the pins for output
   pinMode(leftMotorPin1, OUTPUT);
   pinMode(leftMotorPin2, OUTPUT);
   pinMode(rightMotorPin1, OUTPUT);
   pinMode(rightMotorPin2, OUTPUT);

    // Stop the motor
   analogWrite(leftMotorPin1, 0);
   analogWrite(leftMotorPin2, 0);
   analogWrite(rightMotorPin1, 0);
   analogWrite(rightMotorPin2, 0);

   Button button(button);
   robotStop();

}

void motorControl(int pin1, int pin2, int aSpeed)
{
   analogWrite(pin1, aSpeed);
   analogWrite(pin2, 0);
}

void leftMotorForward(int aSpeed)
{
  motorControl(leftMotorPin1, leftMotorPin2, aSpeed);
}

void rightMotorForward(int aSpeed)
{
  motorControl(rightMotorPin1, rightMotorPin2, aSpeed);
}

void leftMotorBackward(int aSpeed)
{
  motorControl(leftMotorPin2, leftMotorPin1, aSpeed);
}

void rightMotorBackward(int aSpeed)
{
  motorControl(rightMotorPin2, rightMotorPin1, aSpeed);
}

void leftMotorStop()
{
  motorControl(leftMotorPin1, leftMotorPin2, 0);
}

void rightMotorStop()
{
  motorControl(rightMotorPin1, rightMotorPin2, 0);
}

void leftTurn(int aSpeed)
{
  rightMotorForward(aSpeed);
  leftMotorStop();
}

void rightTurn(int aSpeed)
{
  leftMotorForward(aSpeed);
  rightMotorStop();
}

void robotStop()
{
  leftMotorStop();
  rightMotorStop();
}

int robotForward(int aSpeed, int driveTime)
{
  enum {FORWARD, STOP};
  static int state = FORWARD;
  static MSTimer timer;
  int returnValue = false;

  switch(state)
  {
    case FORWARD:
      timer.set(driveTime);
      returnValue = false;
      state = STOP;
    break;

    case STOP:
    if (timer.done())
    {
      robotStop();
      returnValue = true;
      state = FORWARD;
    }
   else
   {
    leftMotorForward(aSpeed);
    rightMotorForward(aSpeed);
   }
   break;
  }
  return returnValue;
}

int robotBackward(int aSpeed, int driveTime)
{
  enum {FORWARD, STOP};
  static int state = FORWARD;
  static MSTimer timer;
  int returnValue = false;

  switch(state)
  {
    case FORWARD:
      timer.set(driveTime);
      returnValue = false;
      state = STOP;
    break;

    case STOP:
    if (timer.done())
    {
      robotStop();
      returnValue = true;
      state = FORWARD;
    }
   else
   {
    leftMotorBackward(aSpeed);
    rightMotorBackward(aSpeed);
   }
   break;
  }
  return returnValue;
}

int robotLeft(int turnTime)
{
  enum {FORWARD, STOP};
  static int state = FORWARD;
  static MSTimer timer;
  int returnValue = false;

  switch(state)
  {
    case FORWARD:
      timer.set(turnTime);
      returnValue = false;
      state = STOP;
    break;

    case STOP:
    if (timer.done())
    {
      robotStop();
      returnValue = true;
      state = FORWARD;
    }
   else
   {
    leftTurn(250);
   }
   break;
  }
  return returnValue;
}
```

```
int robotRight(int turnTime)
{
  enum {FORWARD, STOP};
  static int state = FORWARD;
  static MSTimer timer;
  int returnValue = false;

  switch(state)
  {
    case FORWARD:
      timer.set(turnTime);
      returnValue = false;
      state = STOP;
    break;

    case STOP:
    if (timer.done())
    {
      robotStop();
      returnValue = true;
      state = FORWARD;
    }
   else
   {
    rightTurn(250);
   }
   break;
  }
  return returnValue;
}
int robotSpin(int spinTime)
{
  enum {FORWARD, STOP};
  static int state = FORWARD;
  static MSTimer timer;
  int returnValue = false;

  switch(state)
  {
    case FORWARD:
      timer.set(spinTime);
      returnValue = false;
      state = STOP;
    break;
    case STOP:
    if (timer.done())
    {
      robotStop();
      returnValue = true;
      state = FORWARD;
    }
   else
   {
    rightMotorForward(250);
    leftMotorBackward(250);
   }
   break;
  }
  return returnValue;
}
```