sample.c

```
// NeoPixel Ring simple sketch (c) 2013 Shae Erisson
// released under the GPLv3 license to match the rest of the AdaFruit NeoPixel library

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h>
#endif

//#define ANALOGMODE

// Which pin on the Arduino is connected to the NeoPixels?
// On a Trinket or Gemma we suggest changing this to 1
#define PIN            0

// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS      1
// Modul State
#define IDLE_MODE  1
#define MOVE_MODE  2
#define REVERSE_MODE  3
#define MAX_POWER 254



// When we setup the NeoPixel library, we tell it how many pixels, and which pin to use to send signals.
// Note that for older NeoPixel strips you might need to change the third parameter--see the strandtest
// example for more information on possible values.
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

int delayval = 5; // delay for half a second
#ifdef ANALOGMODE
int SenserPin = 2;
# else
int SenserPin = 4;
#endif
int RightPin = 1;
int LeftPin = 3;

void openDrainOut(int pin_no,int value)
{
  if(value== HIGH){
    //入力にする
    pinMode(pin_no,INPUT_PULLUP);
  }
  else {
    pinMode(pin_no,OUTPUT);
    digitalWrite(pin_no,LOW);
  }
}



void setup() {
  // This is for Trinket 5V 16MHz, you can remove these three lines if you are not using a Trinket
#if defined (__AVR_ATtiny85__)
  if (F_CPU == 8000000) clock_prescale_set(clock_div_1);
#endif
  // End of trinket special code
    pinMode(SenserPin, INPUT_PULLUP);      // sets the digital pin as output

    pinMode(RightPin, INPUT_PULLUP);
    pinMode(LeftPin, INPUT_PULLUP);

  pixels.begin(); // This initializes the NeoPixel library.
}

void loop() {
  unsigned int r,g,b;
  int state;
  int sub_mode;
  int start_device;
  // For a set of NeoPixels the first NeoPixel is 0, second is 1, all the way up to the count of pixels minus one.
  r =0;
  g =0;
  b =0;
  delay(100);

   pixels.setPixelColor(0, pixels.Color(r,g,b)); // Moderately bright green color.

    pixels.show(); // This sends the updated pixel color to the hardware.

  delay(100);
```

sample.c

```
  sub_mode =0;
  start_device =0;
  state = IDLE_MODE;
  for(;;){
    switch(state){
      case IDLE_MODE:
 //     pixels.setPixelColor(0, pixels.Color(r,0,0)); // Moderately bright green color.
 //     pixels.show(); // This sends the updated pixel color to the hardware.
      if( sub_mode ==0){
        b++;
        if(b >= MAX_POWER){
          sub_mode =1;
        }
      }
      else {
        b --;
//        g++;
        if( b ==0){
          sub_mode =0;
        }
      }
      break;
      case MOVE_MODE:
      if(sub_mode == 0){
        b = MAX_POWER;
        g =0;
        r =0;
//        pixels.setPixelColor(0, pixels.Color(r,g,b)); // Moderately bright green color.
//        pixels.show(); // This sends the updated pixel color to the hardware.
        sub_mode =1;
      }
      else if(sub_mode == 1){
        b--;
        r++;
        if(r == (MAX_POWER/2)){
          //ここで ioをドライブする
          if(start_device == SenserPin){
            openDrainOut(LeftPin,LOW);
            openDrainOut(RightPin,LOW);
          }
          else if(start_device== RightPin){
            openDrainOut(LeftPin,LOW);
          }
          else {
            openDrainOut(RightPin,LOW);
          }
        }

        if(r == MAX_POWER){
          sub_mode  =2;
        }
      }
      else if(sub_mode == 2){
        r--;
        g++;
        if(g == MAX_POWER){
          sub_mode = 3;
        }
      }
      else if(sub_mode == 3){
        //なにもしない
      }
      break;
      case REVERSE_MODE:
      if(sub_mode == 3 || sub_mode == 2){// 緑から変化
        if(g >r){
          if(g >0){
            g--;

          }
          if(r > 0){
            r--;
          }
          if(b != MAX_POWER){
            b++;
          }
          if(g ==(MAX_POWER/2)){

            //ここで ioをドライブする
            openDrainOut(LeftPin,HIGH);
            openDrainOut(RightPin,HIGH);
```

```
          }
          if(g == 0){
            sub_mode =1;
            state = IDLE_MODE;
          }
        }
        else if(r >= g){
          if(r >0){
            r--;
          }
          if(g != 0){
            g--;
          }
          if(b != MAX_POWER){
            b++;
          }
          if(r ==0){
            sub_mode =1;
            //ここで ioをドライブする
            openDrainOut(LeftPin,HIGH);
            openDrainOut(RightPin,HIGH);
            state = IDLE_MODE;
          }

        }
      }
      else if(sub_mode == 1){// から変化
        if( r > 0){
          r--;
        }
        b++;
        if(r ==0){
          sub_mode =0;
          //ここで ioをドライブする
            openDrainOut(LeftPin,HIGH);
            openDrainOut(RightPin,HIGH);
            state = IDLE_MODE;
         }
      }


      break;
    }
    // センサー状態確認
    # ifdef ANALOGMODE
    if(analogRead(SenserPin)<=100){//さわられた
    # else
    if(digitalRead(SenserPin)==0){//さわられた
    # endif
      if(state == IDLE_MODE){
        state = MOVE_MODE;
        sub_mode =0;
        start_device = SenserPin;
      }
    }
    # ifdef ANALOGMODE
    else if(analogRead(SenserPin)>=900){//はなされた
    # else
    else if(digitalRead(SenserPin)==1){//はなされた
    # endif
      if(state ==MOVE_MODE){
        if(start_device == SenserPin){
            state = REVERSE_MODE;

        }
        else if(start_device == RightPin){
          if(digitalRead(RightPin)==HIGH){
              state = REVERSE_MODE;
          }
        }
        else {
          if(digitalRead(LeftPin)==HIGH){
              state = REVERSE_MODE;
          }
        }
      }
      else if(state ==IDLE_MODE){
        if(digitalRead(RightPin)==LOW){
          state = MOVE_MODE;
          sub_mode =0;
          start_device = RightPin;
         }
```

sample.c

```
         else if(digitalRead(LeftPin)==LOW){
          state = MOVE_MODE;
          sub_mode =0;
          start_device = LeftPin;
          }
       }
     }
     // pixels.Color takes RGB values, from 0,0,0 up to 255,255,255
      pixels.setPixelColor(0, pixels.Color(r,g,b)); // Moderately bright green color.
     pixels.show(); // This sends the updated pixel color to the hardware.

     if(state ==IDLE_MODE){
        delayval = 5;
     }
     else {
       delayval = 3;
     }
     delay(delayval); // Delay for a period of time (in milliseconds).
//     pixels.setPixelColor(0, pixels.Color(0,0,0)); // Moderately bright green color.
//     pixels.show(); // This sends the updated pixel color to the hardware.

//     delay(1); // Delay for a period of time (in milliseconds).

  }
}
```