

```

/* Schneemann-Pingpong      by J.Reinert   für Arduino Nano V3
* letzte Änderung 22.03.2019 10:15
* Der eingebettete 'trinket special code' stammt aus den Demo-Programmen der Adafruit_Neopixel
Library
*
* Dieses Programm steuert zwei Schneemänner in der Art, dass sie sich Schneebälle zuwerfen.
* Dafür wird eine Schneeball-Flugbahn aus 2812b-LEDs mit mindestens 6 Schneebällen benötigt und
* vier Arme (zwei je Schneemann), die je zwei Positionen einnehmen können.
*
* Anschlussbelegung:
* D2 Linker Schneemann Arm unten (SML_D)
* D3 Linker Schneemann Arm oben (SML_U)
* D4 Rechter Schneemann Arm oben (SMR_U)
* D5 Rechter Schneemann Arm unten (SMR_D)
* D6 Datenausgang WS2812b für Schneebälle
*
*
* Konstanten im Programm:
* MAXPIXEL      Bestimmt die Anzahl der angeschlossenen Schneebälle (mindestens 6)
* SPEED_DELAY   anfängliche Ballgeschwindigkeit. Je kleiner der Wert, um so schneller wird der
Effekt
* MINWAIT       Minimale Wartezeit in Millisekunden bis zum nächsten Effekt
* MAXWAIT       Maximale Wartezeit in Millisekunden bis zum nächsten Effekt
*
Die Werte zwischen MINWAIT und MAXWAIT werden per Zufallsgenerator
*
berechnet

```

```
*
*
* Einzubindende Bibliothek: Adafruit_NeoPixel
*
*/
```

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

```

/-----
// Hardwareabhängige Werte. Bitte anpassen!
#define MAXPIXEL 17          // Anzahl Schneebälle. Sollte ein ungerader Wert >4 sein
#define SPEED_DELAY 150      // Je kleiner der Wert, um so schneller fliegt der Ball
#define MINWAIT 500          // minimale Wartezeit (ms)
#define MAXWAIT 2500         // maximale Wartezeit (ms)
/-----
```

```
// Variablen, Konstanten und Ports
```

```
#define SML_D 2              // Linker Schneemann Arm unten
#define SML_U 3              // Linker Schneemann Arm oben
#define SMR_U 4              // Rechter Schneemann Arm oben
```

```

#define SMR_D 5                // Rechter Schneemann Arm untern
#define DATAPIN 13             // Daten-Ausgang zu Pixel-LEDs
#define ANALOG_IN A0           // Analog-Eingang zur Initialisierung des Zufallsgenerators


// Nimmt berechnete Zufallszahl auf; Wird evtl nicht benötigt
long WaitRandom;


// Library-Parameter setzen
Adafruit_NeoPixel strip = Adafruit_NeoPixel(MAXPIXEL+8, DATAPIN, NEO_RGB + NEO_KHZ800);


/*-----
   Programm-Setup
   -----
*/
void setup() {
//  Serial.begin(9600); // kann später entfernt werden

#ifdef (__AVR_ATtiny85__)
  if (F_CPU == 16000000) clock_prescale_set(clock_div_1);
#endif
// End of trinket special code


pinMode(SML_U,OUTPUT); // Linker Schneemann Arm oben
pinMode(SML_D,OUTPUT); // Linker Schneemann Arm unten

```

```

pinMode(SMR_D,OUTPUT); // Rechter Schneemann Arm unten
pinMode(SMR_U,OUTPUT); // Rechter Schneemann Arm oben

// Zufallsgenerator initialisieren
randomSeed(analogRead(ANALOG_IN));

// Alle LEDs ausschalten und Datenausgabe starten
strip.begin();
strip.show();
strip.setBrightness(255);
Show_Arms();
}

/*-----
  Hauptprogramm-Schleife
-----*/
void loop() {
// DEMO! Von hier aus sollte später in verschiedene Spielarten verzweigt werden
delay(random(MINWAIT,MAXWAIT));
SMR2L(0);
delay(random(MINWAIT,MAXWAIT));
SML2R(0);

}

```

```

/*-----
   Ball von rechts nach links mit oder ohne Fading
   -----
*/
void SMR2L(bool fade) {
    // Acht LEDs nacheinander ab Adresse 0 mit unterschiedlichen Helligkeiten einschalten
    (0-->MAXPIXEL-1)
    int j;
    strip.setPixelColor(0,255,255,255);
    strip.show();
    delay(SPEED_DELAY);
    for(j=0;j<MAXPIXEL; j++) {
        if (fade) {
            if(j==2) {strip.setPixelColor(0, 0, 0, 0);} // hört mit schwarz auf
            if(j==1) {strip.setPixelColor(0, 24, 24, 24);}
            if(j==0) {strip.setPixelColor(0,255,255,255);} // Fängt mit weiß an
        }
        else {
            if(j==1) {strip.setPixelColor(0, 0, 0, 0);} // hört mit schwarz auf
            if(j==0) {strip.setPixelColor(0,255,255,255);} // Fängt mit weiß an
        }
        strip.show();
        Show_Arms();
        delay(SPEED_DELAY);
    }
}

```

```

    Scroll_R2L();
}
if (fade) {
    strip.setPixelColor(MAXPIXEL-1,255,255,255);
    strip.show();
    Show_Arms();
    delay(SPEED_DELAY);
}
}

/*-----
   Wurf von links nach rechts mit oder ohne fading
   -----*/

void SML2R(bool fade) {
    // Acht LEDs nacheinander ab Adresse MAXPIXEL-1 mit unterschiedlichen Helligkeiten
    einschalten (MAXPIXEL-1 --> 0)
    strip.setPixelColor(MAXPIXEL-1,255,255,255);
    strip.show();
    delay(SPEED_DELAY);
    for(int j=MAXPIXEL-1; j>=0; j--) {
        if (fade) {
            if(j==(MAXPIXEL-3)) {strip.setPixelColor(MAXPIXEL-1, 0, 0, 0);} // hört mit schwarz
auf
            if(j==(MAXPIXEL-2)) {strip.setPixelColor(MAXPIXEL-1, 24, 24, 24);}
            if(j==(MAXPIXEL-1)) {strip.setPixelColor(MAXPIXEL-1,255,255,255);} // Fängt mit weiß an

```

```

    }
    else {
        if(j==(MAXPIXEL-2)) {strip.setPixelColor(MAXPIXEL-1, 0, 0, 0);} // hört mit schwarz
auf
        if(j==(MAXPIXEL-1)) {strip.setPixelColor(MAXPIXEL-1,255,255,255);} // Fängt mit weiß an
    }
    strip.show(); // Ergebnis anzeigen
    Show_Arms();
    delay(SPEED_DELAY);
    Scroll_L2R();
}
if (fade) {
    strip.setPixelColor(0,255,255,255);
    strip.show();
    Show_Arms();
    delay(SPEED_DELAY);
}
}

```

```

/*-----
 *  Arme der Schneemänner steuern
 *  wenn der zweite oder dritte bzw zweitletzte oder driitletzte Ball leuchtet, muss
 *  der Arm zum Fangen oder Werfen gehoben werden
*-----*/
void Show_Arms() {
    if (strip.getPixelColor(1)>0xEEEEEE || strip.getPixelColor(2)>0xEEEEEE ) // Rechter

```

Schneemann

```
//if (strip.getPixelColor(1)>0 || strip.getPixelColor(2)>0 ) // Rechter Schneemann
```

```
{  
    digitalWrite(SMR_U, HIGH);  
    digitalWrite(SMR_D, LOW);  
}
```

else

```
{  
    digitalWrite(SMR_U, LOW);  
    digitalWrite(SMR_D, HIGH);  
}
```

```
if (strip.getPixelColor(MAXPIXEL-2)>0xEEEEEE || strip.getPixelColor(MAXPIXEL-3)>0xEEEEEE) //  
linker Schneemann
```

```
//if (strip.getPixelColor(MAXPIXEL-2)>0 || strip.getPixelColor(MAXPIXEL-3)>0) // linker  
Schneemann
```

```
{  
    digitalWrite(SML_U, HIGH);  
    digitalWrite(SML_D, LOW);  
}
```

else

```
{  
    digitalWrite(SML_U, LOW);  
    digitalWrite(SML_D, HIGH);  
}
```

```
}
```

```

/*-----
 * Alle LEDs in Kette oder Streifen um eine Position nach links schieben
 *-----
 */
void Scroll_R2L() {
    for(int i=MAXPIXEL-1; i>0; i--) {                // Alle Pixel bearbeiten
        strip.setPixelColor(i,strip.getPixelColor(i-1));    // Pixel eine Position nach
links kopieren
    }
}

/*-----
 * Alle LEDs in Kette oder Streifen um eine Position nach rechts schieben
 *-----
 */
void Scroll_L2R() {
    for(int i=0; i<(MAXPIXEL-1); i++) {                // Alle Pixel bearbeiten
        strip.setPixelColor(i,strip.getPixelColor(i+1));    // Pixel eine Position nach
unten kopieren
    }
}

/*-----
 * Alle LEDs in Kette ausschalten
 *-----

```

```
*/  
void CLR() {  
    for(int i=0; i<(MAXPIXEL-1); i++) {  
        strip.setPixelColor(i,0);  
    }  
}
```

// Alle Pixel bearbeiten
// Pixeldaten auf NULL