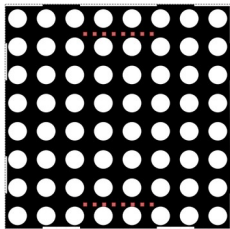


Die RGB-Matrix soll ein zufälliges Pixelmuster anzeigen.

So sieht es aus:



Die RGB-Matrix besteht aus miteinander verbundenen RGB-LEDs. Jede besitzt einen eigenen Controller und kann einzeln angesteuert werden. Die RGB-Matrix benötigt nur einen digitalen Eingang.

RGB ist eine Mischung der Farben Rot, Grün und Blau. Jede Farbe kann von 0 bis 255 gesetzt werden, die Werte werden durch Kommata getrennt.

Beispiele:

163, 0, 93

226, 176, 50

255, 255, 0

255, 228, 225

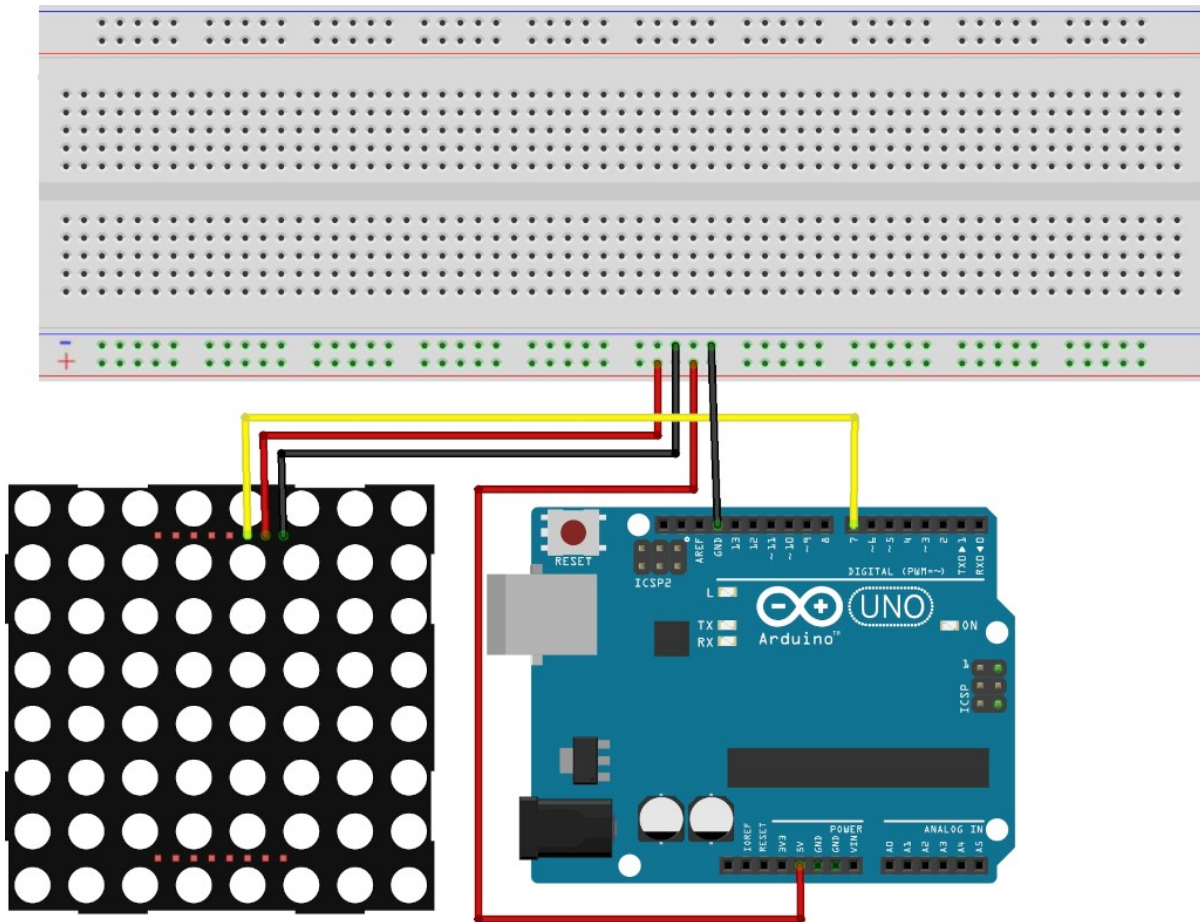


[Weitere Informationen](#) (externer Link)

Benötigte Bauteile:

- ➡ RGB-Matrix 8x8
- ➡ Leitungsdrähte

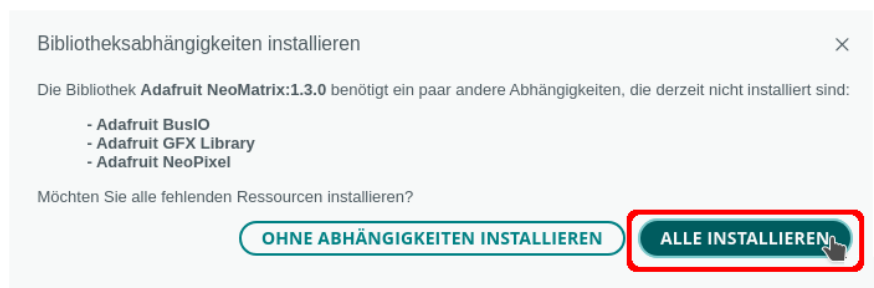
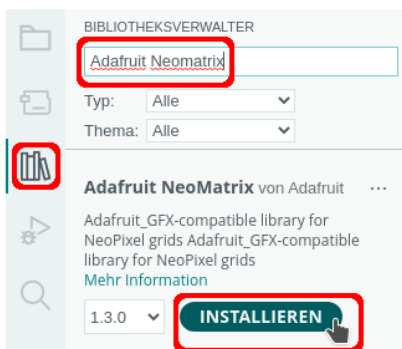
Baue die Schaltung auf.



fritzing

Benötigte Bibliotheken:

Sketch → Bibliothek einbinden → Bibliotheken verwalten



Ob die Ausrichtung der RGB-Matrix korrekt ist, kannst du mit diesem Programm feststellen:

```
# include <Adafruit_NeoMatrix.h>

# define RGBMatrixPin 13

// RGBMatrix -> Name der RGB-Matrix
/*
  die wichtigsten Parameter:
  Parameter 1 = Breite der Matrix (8)
  Parameter 2 = Höhe der Matrix (8)
  Parameter 3 = Name des Daten-Pins (RGBMatrixPin)
*/
Adafruit_NeoMatrix RGBMatrix =   Adafruit_NeoMatrix(8, 8, RGBMatrixPin,
                                                    NEO_MATRIX_TOP + NEO_MATRIX_RIGHT +
                                                    NEO_MATRIX_COLUMNS + NEO_MATRIX_PROGRESSIVE,
                                                    NEO_GRB + NEO_KHZ800);

void setup()
{
  RGBMatrix.setBrightness(10);

  // RGBMatrix starten
  RGBMatrix.begin();
}

void loop()
{
  RGBMatrix.clear();

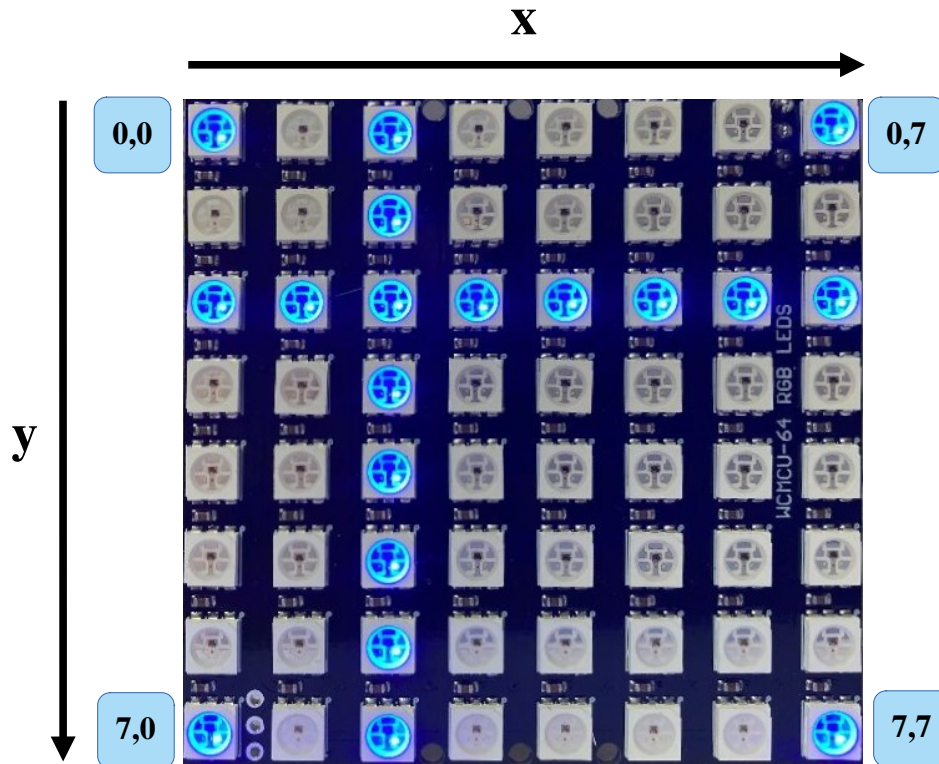
  // horizontale Linie
  RGBMatrix.drawFastHLine(0, 2, 8, RGBMatrix.Color(0, 0, 255));

  // vertikale Linie
  RGBMatrix.drawFastVLine(2, 0, 8, RGBMatrix.Color(0, 0, 255));

  /*
    leuchtende LEDs in den Ecken
    oben links
    oben rechts
    unten links
    unten rechts
  */
  RGBMatrix.drawPixel(0, 0, RGBMatrix.Color(0, 0, 255));
  RGBMatrix.drawPixel(7, 0, RGBMatrix.Color(0, 0, 255));
  RGBMatrix.drawPixel(0, 7, RGBMatrix.Color(0, 0, 255));
  RGBMatrix.drawPixel(7, 7, RGBMatrix.Color(0, 0, 255));

  RGBMatrix.show();
}
```

So muss es aussehen:



Funktionen der Bibliothek Adafruit_NeoMatrix (Auswahl)

Schlüsselwort	Aktion	Parameter
RGB-Matrix starten	<code>begin();</code>	
Bildschirm ausrichten	<code>setRotation(Richtung);</code>	Richtung = 0 → nicht drehen Richtung = 1 → 90° drehen Richtung = 2 → 180° drehen
Bildschirmhelligkeit setzen	<code>setBrightness(Parameter)</code>	0 = aus, 255 = größte Helligkeit
Bildschirmhintergrund füllen	<code>fillScreen(Farbe)</code>	Farben definieren: # define Rot RGBMatrix.Color(255,0,0) # define Gruen RGBMatrix.Color(0,255,0) # define Blau RGBMatrix.Color(0,0,255) # define Magenta RGBMatrix.Color(139,0,139) # define Pink RGBMatrix.Color(255,20,147) # define Weiss RGBMatrix.Color(255,255,255) # define Gelb RGBMatrix.Color(255,255,0)
Linie zeichnen	<code>drawLine(StartX, StartY, EndeX, EndeY, Farbe);</code>	
horizontale Linie zeichnen	<code>drawFastHLine(StartX, StartY, Länge, Farbe);</code>	
vertikale Linie zeichnen	<code>drawFastVLine(StartX, StartY, Länge, Farbe);</code>	
abgerundetes Rechteck zeichnen	<code>drawRoundRect(StartX, StartY, Breite, Höhe, Eckenradius, Farbe);</code>	
ausgefülltes Rechteck zeichnen	<code>fill.Rect(StartX, StartY, Breite, Höhe, Farbe);</code>	

Schlüsselwort	Aktion	Parameter
Kreis zeichnen	drawCircle(MittelpunktX, MittelpunktY, Radius, Farbe);	
Ausgefüllten Kreis zeichnen	fillCircle(MittelpunktX, MittelpunktY, Radius, Füllfarbe);	
Einzelnes Zeichen schreiben	drawChar(0, 1, Zeichen, Rot, Hintergrund, Textgröße);	drawChar(0, 1, 'Z', Rot, Weiss, 1);
Cursor setzen	setCursor(x, y);	
Textgröße setzen	setTextSize(Textgröße);	Textgröße: 1 – 4 bei einer Matrix nur 1 möglich
Textfarbe bestimmen	setTextColor(Farbe);	
Text schreiben	print("Text"); println("Text");	

Beispiel Lauflicht:

```
# include <Adafruit_NeoMatrix.h>

// Startposition links oben
int Spalte = 0;

# define RGBMatrixPin 13

// RGBMatrix -> Name der RGB-Matrix
/*
die wichtigsten Parameter:
Parameter 1 = Breite der Matrix (8)
Parameter 2 = Höhe der Matrix (8)
Parameter 3 = Name des Daten-Pins (RGBMatrixPin)
*/
Adafruit_NeoMatrix RGBMatrix = Adafruit_NeoMatrix(8, 8, RGBMatrixPin,
                                                    NEO_MATRIX_TOP + NEO_MATRIX_RIGHT +
                                                    NEO_MATRIX_COLUMNS + NEO_MATRIX_PROGRESSIVE,
                                                    NEO_GRB + NEO_KHZ800);

void setup()
{
  // Helligkeit setzen
  RGBMatrix.setBrightness(10);

  // RGBMatrix starten
  RGBMatrix.begin();
}
```

```
void loop()
{
  RGBMatrix.clear();

  // von oben nach unten
  for (int i = 0; i <= 7; i++)
  {
    RGBMatrix.drawPixel(Spalte, i, RGBMatrix.Color(0, 0, 255));
    delay(100);
    RGBMatrix.show();
  }

  RGBMatrix.clear();

  // eine Spalte nach rechts
  Spalte++;

  // von unten nach oben
  for (int i = 7; i >= 0; i--)
  {
    RGBMatrix.drawPixel(Spalte, i, RGBMatrix.Color(0, 0, 255));
    delay(100);
    RGBMatrix.show();
  }

  // solange das Ende (Spalte = 7) nicht erreicht ist
  // -> eine Spalte hinzufügen
  if (Spalte < 6) Spalte++;

  // Ende erreicht, Spalte wieder auf 0 setzen
  else Spalte = 0;
}
```

Beispiel mit den Grafikmethoden:



```
// die Bibliotheken Adafruit_GFX.h und Adafruit_NeoPixel.h werden durch
// Adafruit_NeoMatrix.h eingebunden
# include <Adafruit_NeoMatrix.h>

# define RGBMatrixPin 13

// RGBMatrix -> Name der RGB-Matrix
/*
  die wichtigsten Parameter:
  Parameter 1 = Breite der Matrix (8)
  Parameter 2 = Höhe der Matrix (8)
  Parameter 3 = Name des Daten-Pins (RGBMatrixPin)
*/
```

```
Adafruit_NeoMatrix RGBMatrix = Adafruit_NeoMatrix(8, 8, RGBMatrixPin,
                                                    NEO_MATRIX_TOP + NEO_MATRIX_RIGHT +
                                                    NEO_MATRIX_COLUMNS + NEO_MATRIX_PROGRESSIVE,
                                                    NEO_GRB + NEO_KHZ800);

// Farben definieren
# define Rot      RGBMatrix.Color(255,0,0)
# define Gruen    RGBMatrix.Color(0,255,0)
# define Blau     RGBMatrix.Color(0,0,255)
# define Magenta  RGBMatrix.Color(139,0,139)
# define Pink     RGBMatrix.Color(255,20,147)
# define Weiss    RGBMatrix.Color(255,255,255)
# define Gelb     RGBMatrix.Color(255,255,0)

void setup()
{
    // Helligkeit setzen
    RGBMatrix.setBrightness(10);

    // RGBMatrix starten
    RGBMatrix.begin();
}

void loop()
{
    RGBMatrix.clear();
    char Text[7] = {'A', 'r', 'd', 'u', 'i', 'n', 'o'};

    for (int i = 0; i < sizeof(Text); i++)
    {
        RGBMatrix.drawChar(0, 1, Text[i], Rot, 1, 1);
        RGBMatrix.show();
        delay(500);
        RGBMatrix.clear();
    }
    delay(1000);
    RGBMatrix.clear();

    // Bildschirm mit Farbe füllen
    RGBMatrix.fillScreen(Blau);
    RGBMatrix.show();
    delay(500);
    RGBMatrix.clear();

    RGBMatrix.fillScreen(Gelb);
    RGBMatrix.show();
    delay(500);
    RGBMatrix.clear();

    RGBMatrix.fillScreen(Rot);
    RGBMatrix.show();
    delay(500);

    RGBMatrix.clear();
    RGBMatrix.fillScreen(Gruen);
```

```
RGBMatrix.show();
delay(500);
RGBMatrix.clear();

RGBMatrix.fillScreen(Magenta);
RGBMatrix.show();
delay(500);
RGBMatrix.clear();

// einzelnes Zeichen schreiben
RGBMatrix.setCursor(0, 1);
RGBMatrix.setTextColor(Pink);
RGBMatrix.setTextSize(1);
RGBMatrix.print('Z');
RGBMatrix.show();
delay(500);
RGBMatrix.clear();

// Linie zeichnen
for (int i = 0; i <= 7; i++)
{
    RGBMatrix.drawLine(i, 0, i, 8, Gelb);
    RGBMatrix.show();
    delay(200);
}
delay(500);
RGBMatrix.clear();

// horizontale Linie zeichnen
for (int i = 1; i < 9; i++)
{
    RGBMatrix.drawFastHLine(0, i, i, Gruen);
    RGBMatrix.show();
    delay(200);
}
delay(500);
RGBMatrix.clear();

// vertikale Linie zeichnen
for (int i = 1; i < 9; i++)
{
    RGBMatrix.drawFastVLine(i, 0, i, Magenta);
    RGBMatrix.show();
    delay(100);
}
delay(500);
RGBMatrix.clear();

// Kreis zeichnen
RGBMatrix.drawCircle(4, 4, 3, Weiss);
RGBMatrix.show();
delay(500);
RGBMatrix.clear();

// ausgefüllten Kreis zeichnen
RGBMatrix.fillCircle(4, 4, 3, Blau);
```



```
RGBMatrix.show();
delay(500);

// Rechtecke zeichnen
RGBMatrix.clear();
for (int i = 1; i < 9; i++)
{
    RGBMatrix.drawRect(0, 0, i, i, RGBMatrix.Color(0, 255, 255));
    RGBMatrix.show();
    delay(200);
}
delay(500);
RGBMatrix.clear();
}
```

Das eigentliche Programm: LEDs mit Zufallsfarben:

```
# include <Adafruit_NeoMatrix.h>

# define RGBMatrixPin 13

// Anzahl der leuchtenden LEDs
// da die Position zufällig bestimmt wird, können sie sich auch überlagern
# define AnzahlLED 64

// RGBMatrix -> Name der RGB-Matrix
/*
  die wichtigsten Parameter:
  Parameter 1 = Breite der Matrix (8)
  Parameter 2 = Höhe der Matrix (8)
  Parameter 3 = Name des Daten-Pins (RGBMatrixPin)
*/
Adafruit_NeoMatrix RGBMatrix = Adafruit_NeoMatrix(8, 8, RGBMatrixPin,
                                                    NEO_MATRIX_TOP + NEO_MATRIX_RIGHT +
                                                    NEO_MATRIX_COLUMNS + NEO_MATRIX_PROGRESSIVE,
                                                    NEO_GRB + NEO_KHZ800);

# define Zufallsfarbe RGBMatrix.Color(random(1, 255), random(1, 255), random(1, 255))

void setup()
{
    RGBMatrix.setBrightness(10);

    // NeoPixel Bibliothek initialisieren
    RGBMatrix.begin();

    // Zufallsgenerator starten
    randomSeed(analogRead(0));
}
```

```
void loop()
{
  RGBMatrix.clear();

  int Minimum = 0;
  int Maximum = 8;

  for (int i = 0; i < AnzahlLED; i++)
  {
    // Zufallsposition der Pixel
    int Spalte = random(Minimum, Maximum);
    int Zeile = random(Minimum, Maximum);

    RGBMatrix.drawPixel(Spalte, Zeile, ZufallsFarbe);
  }
  RGBMatrix.show();
  delay(1000);
  RGBMatrix.clear();
}
```



Erstelle mit den grafischen Methoden auch eigene Muster oder einen eigenen Text.