```
/**************************************************************************/
/*!
This is a demo for the Adafruit MCP9808 breakout
----> http://www.adafruit.com/products/1782
Adafruit invests time and resources providing this open source code,
please support Adafruit and open-source hardware by purchasing
products from Adafruit!
*/
/**************************************************************************/
#include <Adafruit_NeoPixel.h>
#include <Wire.h>
#include "Adafruit_MCP9808.h"
// Create the MCP9808 temperature sensor object
Adafruit_MCP9808 tempsensor = Adafruit_MCP9808();
// Which pin on the Arduino is connected to the NeoPixels?
// On a Trinket or Gemma we suggest changing this to 1:
#define LED_PIN    6
// How many NeoPixels are attached to the Arduino?
#define LED_COUNT 60
// Declare our NeoPixel strip object:
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRBW + NEO_KHZ800);
// Argument 1 = Number of pixels in NeoPixel strip
// Argument 2 = Arduino pin number (most are valid)
// Argument 3 = Pixel type flags, add together as needed:
//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)
//   NEO_KHZ400  400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811 drivers)
//   NEO_GRB     Pixels are wired for GRB bitstream (most NeoPixel products)
//   NEO_RGB     Pixels are wired for RGB bitstream (v1 FLORA pixels, not v2)
//   NEO_RGBW    Pixels are wired for RGBW bitstream (NeoPixel RGBW products)
// setup() function -- runs once at startup --------------------------------
void setup() {
  strip.begin();           // INITIALIZE NeoPixel strip object (REQUIRED)
  strip.show();            // Turn OFF all pixels ASAP
  strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)

  Serial.begin(9600);
  while (!Serial); //waits for serial terminal to be open, necessary in newer arduino boards.
  Serial.println("MCP9808 demo");

  // Make sure the sensor is found, you can also pass in a different i2c
  // address with tempsensor.begin(0x19) for example, also can be left in blank for default address use
  // Also there is a table with all addres possible for this sensor, you can connect multiple sensors
  // to the same i2c bus, just configure each sensor with a different address and define multiple objects for that
  //  A2 A1 A0 address
  //  0  0  0   0x18  this is the default address
  //  0  0  1   0x19
  //  0  1  0   0x1A
  //  0  1  1   0x1B
  //  1  0  0   0x1C
  //  1  0  1   0x1D
  //  1  1  0   0x1E
  //  1  1  1   0x1F
  if (!tempsensor.begin(0x18)) {
    Serial.println("Couldn't find MCP9808! Check your connections and verify the address is correct.");
    while (1);
  }

   Serial.println("Found MCP9808!");
  tempsensor.setResolution(3); // sets the resolution mode of reading, the modes are defined in the table bellow:
  // Mode Resolution SampleTime
  //  0    0.5°C       30 ms
  //  1    0.25°C      65 ms
  //  2    0.125°C     130 ms
  //  3    0.0625°C    250 ms
}
void loop() {
  Serial.println("wake up MCP9808.... "); // wake up MCP9808 - power consumption ~200 mikro Ampere
  tempsensor.wake();   // wake up, ready to read!
  // Read and print out the temperature, also shows the resolution mode used for reading.
  Serial.print("Resolution in mode: ");
  Serial.println (tempsensor.getResolution());
  float c = tempsensor.readTempC();
  float f = tempsensor.readTempF();
  Serial.print("Temp: ");
  Serial.print(c, 4); Serial.print("*C\t and ");
  Serial.print(f, 4); Serial.println("*F.");

  delay(200);
  Serial.println("Shutdown MCP9808.... ");
  tempsensor.shutdown_wake(1); // shutdown MSP9808 - power consumption ~0.1 mikro Ampere, stops temperature sampling
  Serial.println("");
```

```
  delay(200);
if (c <= 24.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(0,   0,   0), 50); // light turned off
  }
if (c > 24.75 && c <= 25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 10); // white
  }
if (c > 25 && c <= 25.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 20); // white
  }
if (c > 25.25 && c <= 25.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 30); // white
  }
if (c > 25.5 && c <= 25.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 40); // white
  }
if (c > 25.75 && c <= 26){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 50); // white
  }
if (c > 26 && c <= 26.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   204,   204), 50); // red1
  }
 if (c > 26.25 && c <= 26.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   153,   153), 50); // red2
  }
 if (c > 26.5 && c <= 26.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   102,   102), 50); // red3
  }
if (c > 26.75 && c <= 27){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   51,   51), 50); // red4
  }
if (c > 27 && c <= 27.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   51,   153), 50); // rose4
  }
if (c > 27.25 && c <= 27.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   102,  178), 50); // rose3
  }
if (c > 27.5 && c <= 27.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   153,   204), 50); // rose2
  }
if (c > 27.75 && c <= 28){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   204,   229), 50); // rose1
  }
if (c > 28 && c <= 28.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   204,   255), 50); // pink1
  }
if (c > 28.25 && c <= 28.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   153,   255), 50); // pink2
  }
if (c > 28.5 && c <= 28.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   102,   255), 50); // pink3
   }
if (c > 28.75 && c <= 29){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   51,   255), 50); // pink4
  }
if (c > 29 && c <= 29.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   51,   255), 50); // purple4
  }
if (c > 29.25 && c <= 29.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(178,   102,   255), 50); // purple3
  }
```

```
if (c > 29.5 && c <= 29.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   153,   255), 50); // purple2
  }
if (c > 29.75 && c <= 30){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(229,   204,   255), 50); // purple1
  }
if (c > 30 && c <= 30.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   204,   255), 50); // phthalo blue1
  }
if (c > 30.25 && c <= 30.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   153,   255), 50); // phthalo blue2
  }
if (c > 30.5 && c <= 30.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(102,   102,   255), 50); // phthalo blue3
  }
if (c > 30.75 && c <= 31){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(51,   51,   255), 50); // phthalo blue4
  }
if (c > 31.25 && c <= 31.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(51,   153,   255), 50); // acid blue4
  }
if (c > 31.5 && c <= 31.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(102,   178,   255), 50); // acid blue3
  }
if (c > 31.75 && c <= 32){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   204,   255), 50); // acid blue2
  }
if (c > 32 && c <= 32.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   229,   255), 50); // acid blue1
  }
if (c > 32.25 && c <= 32.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   255,   255), 50); // sky blue1
  }
if (c > 32.5 && c <= 32.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   255,   255), 50); // sky blue2
  }
if (c > 32.75 && c <= 33){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(102,   255,   255), 50); // sky blue3
  }
if (c > 33.25 && c <= 33.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(51,   255,   255), 50); // sky blue4
  }
if (c > 33.5 && c <= 33.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(51,   255,   153), 50); // pastel green4
  }
if (c > 33.75 && c <= 34){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(102,   255,   178), 50); // pastel green3
  }
if (c > 34 && c <= 34.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   255,   204), 50); // pastel green2
  }
if (c > 34.25 && c <= 34.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(104,   255,   229), 50); // pastel green1
  }
if (c > 34.5 && c <= 34.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   255,   204), 50); // grass green1
  }
if (c > 34.75 && c <= 35){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   255,   153), 50); // grass green2
  }
if (c > 35 && c <= 35.25){
```

```
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(102,   255,   102), 50); // grass green3
  }
 if (c > 35.25 && c <= 35.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(51,   255,   51), 50); // grass green4
  }
 if (c > 35.5 && c <= 35.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(153,   255,   51), 50); // olive4
  }
 if (c > 35.75 && c <= 36){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(178,   255,   102), 50); // olive3
  }
 if (c > 36 && c <= 36.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(204,   255,   153), 50); // olive2
  }
 if (c > 36.25 && c <= 36.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(229,   255,   204), 50); // olive1
  }
 if (c > 36.5 && c <= 36.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   204), 50); // yellow1
  }
 if (c > 36.75 && c <= 37){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   153), 50); // yellow2
  }
 if (c > 37 && c <= 37.25){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   102), 50); // yellow3
  }
 if (c > 37.25 && c <= 37.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   51), 50); // yellow4
  }
 if (c > 37.5 && c <= 37.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   153,   51), 50); // orange4
  }
 if (c > 37.75 && c <= 38){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   178,   102), 50); // orange3
  }
 if (c > 38.25 && c <= 38.5){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   204,   153), 50); // orange2
  }
 if (c > 38.5 && c <= 38.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   229,   204), 50); // orange1
  }
 if (c > 38.75){
    strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
    colorWipe(strip.Color(255,   255,   255), 50); // white
  }
 // Some functions of our own for creating animated effects -----------------
 // Fill strip pixels one after another with a color. Strip is NOT cleared
 // first; anything there will be covered pixel by pixel. Pass in color
 // (as a single 'packed' 32-bit value, which you can get by calling
 // strip.Color(red, green, blue) as shown in the loop() function above),
 // and a delay time (in milliseconds) between pixels.
 void colorWipe(uint32_t color, int wait) {
   for(int i=0; i<strip.numPixels(); i++) { // For each pixel in strip...
    strip.setPixelColor(i, color);         //  Set pixel's color (in RAM)

   }
   strip.show();                          //  Update strip to match
    delay(wait);                           //  Pause for a moment
 }
 // Theater-marquee-style chasing lights. Pass in a color (32-bit value,
 // a la strip.Color(r,g,b) as mentioned above), and a delay time (in ms)
 // between frames.
 void theaterChase(uint32_t color, int wait) {
   for(int a=0; a<10; a++) {  // Repeat 10 times...
    for(int b=0; b<3; b++) {  //  'b' counts from 0 to 2...
      strip.clear();         //   Set all pixels in RAM to 0 (off)
     // 'c' counts up from 'b' to end of strip in steps of 3...
```

```
      for(int c=b; c<strip.numPixels(); c += 3) {
        strip.setPixelColor(c, color); // Set pixel 'c' to value 'color'
      }
      strip.show(); // Update strip with new contents
      delay(wait);  // Pause for a moment
    }
  }
}
// Rainbow cycle along whole strip. Pass delay time (in ms) between frames.
void rainbow(int wait) {
  // Hue of first pixel runs 5 complete loops through the color wheel.
  // Color wheel has a range of 65536 but it's OK if we roll over, so
  // just count from 0 to 5*65536. Adding 256 to firstPixelHue each time
  // means we'll make 5*65536/256 = 1280 passes through this loop:
  for(long firstPixelHue = 0; firstPixelHue < 5*65536; firstPixelHue += 256) {
    // strip.rainbow() can take a single argument (first pixel hue) or
    // optionally a few extras: number of rainbow repetitions (default 1),
    // saturation and value (brightness) (both 0-255, similar to the
    // ColorHSV() function, default 255), and a true/false flag for whether
    // to apply gamma correction to provide 'truer' colors (default true).
    strip.rainbow(firstPixelHue);
    // Above line is equivalent to:
    // strip.rainbow(firstPixelHue, 1, 255, 255, true);
    strip.show(); // Update strip with new contents
    delay(wait);  // Pause for a moment
  }
}
// Rainbow-enhanced theater marquee. Pass delay time (in ms) between frames.
void theaterChaseRainbow(int wait) {
  int firstPixelHue = 0;     // First pixel starts at red (hue 0)
  for(int a=0; a<30; a++) {  // Repeat 30 times...
    for(int b=0; b<3; b++) { //  'b' counts from 0 to 2...
      strip.clear();         //   Set all pixels in RAM to 0 (off)
      // 'c' counts up from 'b' to end of strip in increments of 3...
      for(int c=b; c<strip.numPixels(); c += 3) {
        // hue of pixel 'c' is offset by an amount to make one full
        // revolution of the color wheel (range 65536) along the length
        // of the strip (strip.numPixels() steps):
        int      hue   = firstPixelHue + c * 65536L / strip.numPixels();
        uint32_t color = strip.gamma32(strip.ColorHSV(hue)); // hue -> RGB
        strip.setPixelColor(c, color); // Set pixel 'c' to value 'color'
      }
      strip.show();              // Update strip with new contents
      delay(wait);               // Pause for a moment
      firstPixelHue += 65536 / 90; // One cycle of color wheel over 90 frames
    }
  }
}
```