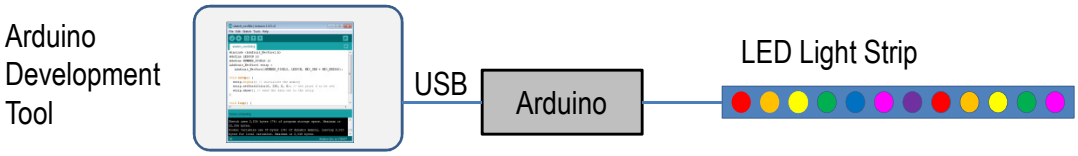
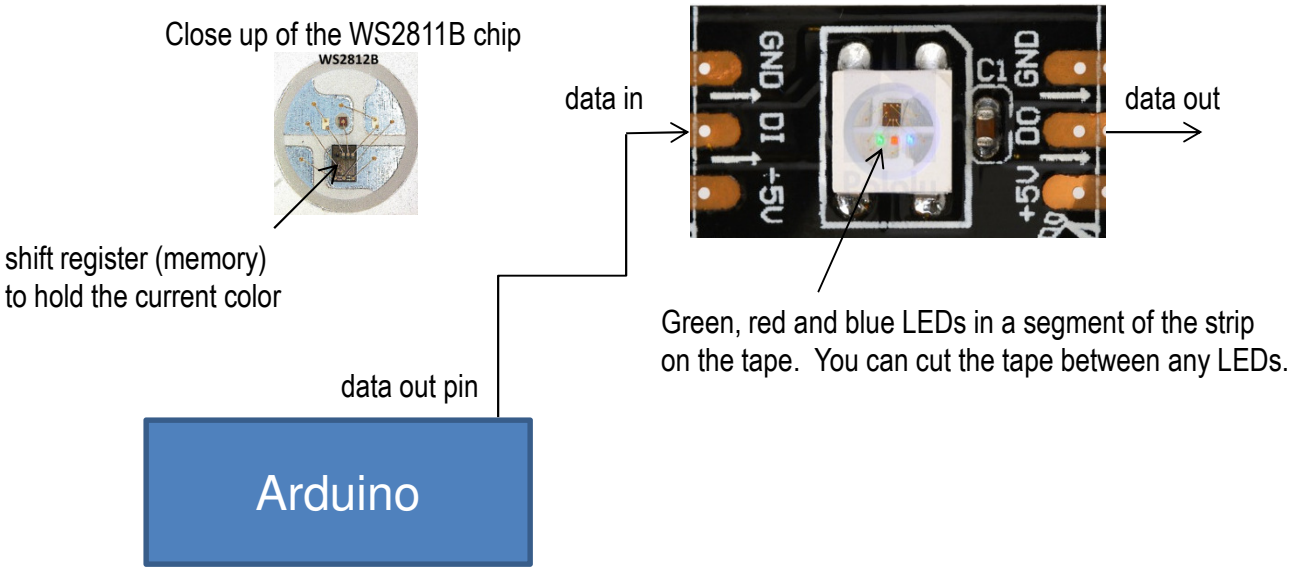
Moving Rainbow Labs

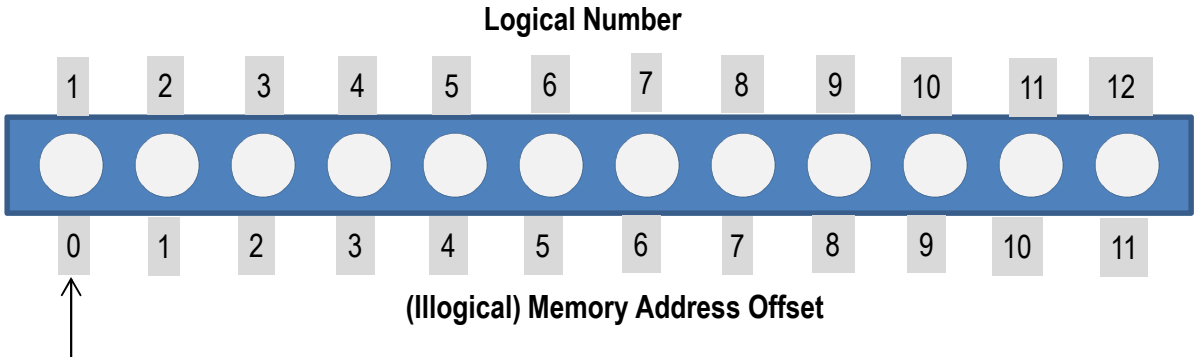
Welcome to the moving rainbow labs. The goal of these labs is to learn the basics of programming by learning to turn each of these LEDs on in different patterns. To use these labs connect the USB of the Moving Rainbow kit to your computer and load the labs into the Arduino development system.



The Moving Rainbow Kit is based on a strip of 12 "addressable" Red, Green and Blue (RGB) Light Emitting Diodes (LEDs). We only need a single data signal (and power and ground) to connect the entire strip.



We will use a software library provided by Adafruit called the "NeoPixel" library to turn lights on and off. Our LED strip has 12 pixels. However we start numbering at "0" to show how computer science people are usually off-by-one digit when they access memory locations!



The first pixel has an address of "0", the second an address called "1" etc.

Moving Rainbow Labs

Background – the NeoPixel Library and Examples

Our fist task is to learn about using the Adafruit NeoPixel library.

The software is free and is available on github here: http://github.com/adafruit/Adafruit_NeoPixel

To use the library you will need to add the library to your Arduino Library area such as.

Window: C:\Users\[name]\Documents\Arduino\libraries\Adafruit_NeoPiixel

Mac: /users/[name]/Documents/Arduino/library/Adafruit_NeoPixel

We must add an "include" line in our program to call the library.

We must set up what **data pin** will be connected to our LED strip (Important!)

We must configure the strip with the right number of pixels and initialize the library data in our setup()

This value might have to be changed for your kit.

Most of our kits have 12 pixels, so you can keep this the same

header

setup

loop

```
#include <Adafruit_NeoPixel.h>
#define LEDPIN 10 // connect the Data from the strip to this pin on the Arduino
#define NUMBER_PIEXELS 12 // the number of pixels in your LED strip
Adafruit_NeoPixel strip =
        Adafruit_NeoPixel(NUMBER_PIEXELS, LEDPIN, NEO_GRB + NEO_KHZ800);

void setup() {
  strip.begin(); // initialize all the variables
}

void loop() {
   // put your code here…
   strip.show(); // This sends the data out the data port to the strip
}
```

Notes:

The header will be the same for most of our programs. Remember to change the LED PIN!

The setup runs only once. It must initialize the variables with the strip.begin(); function

Each time we change a color or value we MUST add a strip.show()! This updates the strip.

Moving Rainbow Labs

Lab 1: set the first pixel to be red, green and blue

Goal: set the color of the first pixel to be red, then green, then blue. We will use the following function:

strip.setPixelColor(pixel-address, red, green-value, blue-value);

Where the color values are 0 for off and 255 for very bright.

Sample Program #1

```
#include <Adafruit_NeoPixel.h>
#define LEDPIN 10 // connect the Data from the strip to this pin on the Arduino
#define NUMBER_PIEXELS 12 // the number of pixels in your LED strip
Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUMBER_PIEXELS, LEDPIN,
NEO_GRB + NEO_KHZ800);

void setup() {
  strip.begin(); // initialize all the variables
  strip.show(); // Initialize all pixels in the strip to be off
}

void loop() {
  strip.setPixelColor(0, 255, 0, 0); // set pixel 0 to be red
  strip.show();
  delay(1000);
  strip.setPixelColor(0, 255, 0, 0); // set pixel 0 to be green
  strip.show();
  delay(1000);
  strip.setPixelColor(0, 255, 0, 0); // set pixel 0 to be blue
  strip.show();
  delay(1000);
}
```

More to Explore

Explore – change the function delay(1000) to be shorter or longer.

Question: What if you change the address from "0" to be "1".

Can you make pixel 1 red, pixel 2 green and pixel 3 blue?

Lab 2: set each pixel in the strip to a color

Goal: set the color of each of the pixels to a color you choose. You can change the red, green and blue values to be whatever value you would like. For example purple is red and blue with green set to be 0.

strip.setPixelColor(11, 255, 0, 255); // set pixel 0 to be purple

```
#include <Adafruit_NeoPixel.h>
#define LEDPIN 10 // connect the Data from the strip to this pin on the Arduino
#define NUMBER_PIEXELS 12 // the number of pixels in your LED strip
Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUMBER_PIEXELS, LEDPIN,
NEO_GRB + NEO_KHZ800);

void setup() {
  strip.begin(); // initialize all the variables
  strip.setPixelColor(0, 255, 0, 0); // set the first pixel to be red
  strip.setPixelColor(1, 0, 255, 0); // set second pixel green
  strip.setPixelColor(2, 0, 0, 255); // set the third pixel to be blue
  strip.setPixelColor(3, 128, 128, 0); // set fourth pixel to be olive?
  strip.setPixelColor(4, 255, 165, 0); // set fifth pixel to be orange
  strip.setPixelColor(5, 255, 255, 0); // set sixth pixel to be yellow
  strip.setPixelColor(6, 255, 0, 255); // set seventh pixel to be Fuchsia/Magenta
  strip.setPixelColor(7, 0, 255, 255); // set eighth pixel to be Aqua/Cyan
  strip.setPixelColor(8, 255, 105, 180); // set ninth pixel to be hot pink
  strip.setPixelColor(9, 255, 255, 255); // set tenth pixel to be white
  strip.setPixelColor(10, 10, 10, 10); // set eleventh pixel to be light gray
  strip.setPixelColor(11, 1, 1, 1); // set twelfth and last pixel to be almost off
  strip.show(); // Send to the strip}

void loop() {
}
```

More To Explore

Do a websearch for "Web Colors" or go to this Wikipedia page: http://en.wikipedia.org/wiki/Web_colors

Can you find the RGB values for yellow, orange and olive?

Which colors are similar to what you see on the web page? Which ones are not even close? Why do you think there is such a wide variation?

Remember that full on is 255 and half on is 128 so

Red	#FF0000	100%	0%	0%
Maroon	#800000	50%	0%	0%
Yellow	#FFFF00	100%	100%	0%
Olive	#808000	50%	50%	0%
Lime	#00FF00	0%	100%	0%
Green	#008000	0%	50%	0%
Aqua	#00FFFF	0%	100%	100%
Teal	#008080	0%	50%	50%
Blue	#0000FF	0%	0%	100%
Navy	#000080	0%	0%	50%
Fuchsia	#FF00FF	100%	0%	100%
Purple	#800080	50%	0%	50%

Moving Rainbow Labs

Lab 3: Color Wheel and for loops

Goal: use a for loop to set the color of each pixel in the LED strip.

We will use the Wheel() function that will turn a number from 0 to 255 into a specific color. We first need to calculate how far into the color we want to go for each pixel. This will be 1/12th of 255 since we have 25 pixels.

Our for loop starts at 0 and goes up to 11.

i++ means add one to I

Don't worry about how the Wheel() function works for now. We just need to use it in our program.

```
// header is not shown here…
int colorIncrment = 255/NUMBER_PIEXELS;  // 1/12th of the way into the color wheel
void setup() {
  strip.begin(); // initialize the strip
  for (int i=0; i<=11; i++) {
    strip.setPixelColor(i, Wheel(colorIncrment * i & 255));
  }
  strip.show();
}

void loop() {} // nothing to do here yet

// Input a value 0 to 255 to get a color value.
uint32_t Wheel(byte WheelPos) {
  WheelPos = 255 - WheelPos;
  if(WheelPos < 85) {
   return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else if(WheelPos < 170) {
    WheelPos -= 85;
   return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  } else {
   WheelPos -= 170;
   return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  }
}
```

Explore

Note that twe are using the "BINARY AND" function to only look at the lower 8 bits of the color. You can remove it and it will still work, however it is a good practice to get rid of the other bits when the function expect only a single byte of data.

The Wheel function is very handy for cycling through each of the colors.

Lab 4: Move a pixel!

Goal: use the main loop() to make a single red dot move from one end of the strip to the other

Our for loop starts at 0 and goes up to 11. For each time through the loop we will turn a pixel on, wait a 1/0th of a second and turn that pixel off.

We use the delay function which has an parameter the number of milliseconds to wait. 1 millisecond = 1/1,000 of a second.

```
// header is not shown here…
void setup() {
  strip.begin(); // initialize the strip
}

void loop() {
  for (int i=0; i<=11; i++) {
    strip.setPixelColor(i, 255, 0, 0); // turn the "i"th pixel on
    strip.show();
    delay(100); // wait 1/10th of a second or 100 milliseconds
    strip.setPixelColor(i, 0, 0, 0); // turn the "i"th pixel off
  }
}
```

Explore

Can you change the color of the pixel?

Can you make the pixel go faster or slower by changing the delay time?

Can you put another color in the line that turns the pixel off?

Can you change the direction of the motion? Try starting at 11, going to 0 and doing i--.

Note – the name of the "i"th pixel is sometimes called the "index" pixel. It is the one you are working on.

Lab 5: Move a group three pixels

In this lab we will use the main loop() to make a group of three pixels appear to move through the strip

We will have the leading and trailing pixels be dim, and the center pixel be brighter. We will not only draw the three pixels in a loop but we will also have to erase them. Note how we use the index, the index + 1 and the index + 2. Note that we are only drawing to the 10th pixel here (i-9).

```
void loop() {
  for (int i=0; i<=9; i++) {
    strip.setPixelColor(i,   10, 0, 0); // turn the index on dim
    strip.setPixelColor(i+1, 100, 0, 0); // turn the index + 1 bright
    strip.setPixelColor(i+2, 10, 0, 0); // turn the index + 2 dim
    strip.show();
    delay(100); // wait 1/10th of a second
    strip.setPixelColor(i, 0, 0, 0);
    strip.setPixelColor(i+1, 0, 0, 0);
    strip.setPixelColor(i+2, 0, 0, 0);
  }
}
```

Explore

Can you make the group move back in the other direction?

What happens if you change the brightness levels of the first and last pixel?

Lab 6: Fade in to location

Goal: make a single pixel look like it is moving behind a grid by slowing fading it in to a location and then out. We do this by slowly ramping up the brightness from 150 to 200 and then back down. Compare this to the version that does not slowly turn on the brightness.

```
int delayTime = 20;

void setup() {
  strip.begin(); // initialize the strip
}

void loop() {
  for (int i=0; i<=11; i++) {
    strip.setPixelColor(i,   150, 0, 0);
    strip.show();
    delay(delayTime);
    strip.setPixelColor(i,   175, 0, 0);
    strip.show();
    delay(delayTime);
    strip.setPixelColor(i,   200, 0, 0);
    strip.show();
    delay(delayTime);
    strip.setPixelColor(i,   175, 0, 0);
    strip.show();
    delay(delayTime);
    strip.setPixelColor(i,   150, 0, 0);
    strip.show();
    delay(delayTime);
    strip.setPixelColor(i, 0, 0, 0);
  }
}
```

More to Explore

- Does the light look like it is flickering on and off?
- How many steps of dimming do you need to you need to make it look smooth?
- Can you put the fade in and fade out in a new loop within the motion loop?

Moving Rainbow Labs

Lab 7: Theater-style running lights (chase)

We now learn now to call a specific function to draw a specific pattern. In this case we will use a function in the NeoPixel library called theaterChase(). The function takes two parameters, the color and the time to spend on each draw. We use a function called strip.Color() that turns an RGB value into a single integer that gets passed to the function.

```
int delayTime = 100;  // 1/10th of a second for each postion

void setup() {
  strip.begin(); // initialize the strip
}

void loop() { // 4th of July pattern for red, white and blue
  theaterChase(strip.Color(127, 0, 0), delayTime); // red
  theaterChase(strip.Color(127, 127, 127), delayTime); // white
  theaterChase(strip.Color(0, 0, 127), delayTime); // blue
}

// Theatre-style crawling lights.
void theaterChase(uint32_t c, uint8_t wait) {
  for (int j=0; j<10; j++) {  //do 10 cycles of chasing
    for (int q=0; q < 3; q++) {
      for (int i=0; i < strip.numPixels(); i=i+3) {
        strip.setPixelColor(i+q, c);    //turn every third pixel on
      }
      strip.show();
      delay(wait);
      for (int i=0; i < strip.numPixels(); i=i+3) {
        strip.setPixelColor(i+q, 0);        //turn every third pixel off
      }
    }
  }
}
```

More to Explore

- Does the light look like it is flickering on and off?
- How many steps of dimming do you need to you need to make it look smooth?
- Can you put the fade in and fade out in a new loop within the motion loop?

Lab 8: Rainbow effect library calls

Next we show how to combine our knowledge of functions to call three separate functions. In the example code below we show a loop that contains three functions that alternate. The rainbow() function is a static rainbow where all the colors are the same. The rainbowCycle() slides each pixel through the colors. The theaterChaseRainbow() cycles through each of the colors in a chase pattern.

```
int delayTime = 25;

void loop() {
  rainbow(delayTime); // have all the colors be the same but cycle through all of them
  rainbowCycle(delayTime); // the rainbow patterns slides through as though it is moving
  theaterChaseRainbow(delayTime * 2);
}

void rainbow(uint8_t wait) {
  uint16_t i, j;
  for(j=0; j<256; j++) {
    for(i=0; i<strip.numPixels(); i++) {
      strip.setPixelColor(i, Wheel((i+j) & 255));
    }
    strip.show();
    delay(wait);
  }
}

// Slightly different, this makes the rainbow equally distributed throughout
void rainbowCycle(uint8_t wait) {
  uint16_t i, j;
  for(j=0; j<256*5; j++) { // 5 cycles of all colors on wheel
    for(i=0; i< strip.numPixels(); i++) {
      strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) + j) & 255));
    }
    strip.show();
    delay(wait);
  }
}
```

More to Explore

- Try writing your own function. Add it to the cycle of patterns.

Lab 9: Moving rainbow

In this example we are using a 7 element moving rainbow. Each time we call it id draws the rainbow starting at the index and going in 7 elements. It turns the other elements off. This program calculates the index offset first and then does a modulo function using the number of pixels in the strip as the divisor. The remainder Is then the actual index that will get used. This has the effect of cycling through the strip.

```
int delayTime = 100;

void setup() {
  strip.begin(); // initialize the strip
}

void loop() {
  for (int i=0; i<11; i++) {
    rainbow7(i, delayTime); // starting at i, draw the 7 color rainbow
  }
}

void rainbow7(uint16_t i, uint16_t wait) {
    int np = strip.numPixels();  // we use the modulo function with this
    strip.setPixelColor(i     % np, 25, 0, 25); // violet
    strip.setPixelColor((i+1) % np, 255, 0, 255); // indigo
    strip.setPixelColor((i+2) % np, 0, 0, 150); // blue
    strip.setPixelColor((i+3) % np, 0, 150, 0); // green
    strip.setPixelColor((i+4) % np, 255, 255, 0); // yellow
    strip.setPixelColor((i+5) % np, 110, 70, 0); // orange
    strip.setPixelColor((i+6) % np, 150, 0, 0); // red
    strip.setPixelColor((i+10) % np, 0, 0, 0); // turn the second to the last one off
    strip.setPixelColor((i+11) % np, 0, 0, 0); // turn the last one off
    strip.show();
    delay(wait);
}
```

More to Explore

- Note that we are using the function strip.numPixels(); to get the exact number of pixels in the strip.
- Can you make an 8 or 9 segment rainbow? What colors would you include?
- Can you replace the lines in the rainbow7 function with a for loop that uses Wheel()?

Lab 10: Random numbers and candle flicker

Sometimes people like a bit of variation in their patterns. The random flicker of a candle

```
void setup() {
  strip.begin();
}

void loop() {
 candle();
}

// simulate a flickering candle which is mostly yellow with a bit or orange thrown in but no blue
void candle() {
   uint8_t green; // brightness of the green
   uint8_t red;  // add a bit for red
   for(uint8_t i=0; i<100; i++) {
     green = 50 + random(155);
     red = green + random(50);
     strip.setPixelColor(random(strip.numPixels() - 1), red, green, 0);
     strip.show();
     delay(5);
  }
}
```

More to Explore

- Note that we are using the function strip.numPixels(); to get the exact number of pixels in the strip.
- Can you make an 8 or 9 segment rainbow? What colors would you include?
- Can you replace the lines in the rainbow7 function with a for loop that uses Wheel()?