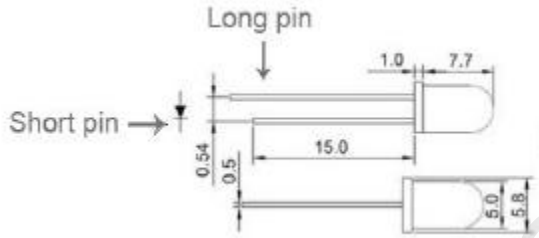
LED blink

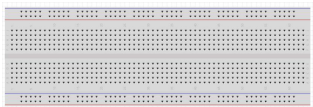
Overview

This example shows the simplest thing you can do with an Arduino to see physical output: it blinks an LED.

Specification



Hardware required

Material diagram	Material name	Number
	220/330Ω resistor	1
	LED	1
	USB Cable	1
	UNO R3	1
	Breadboard	1
	Jumper wires	Several

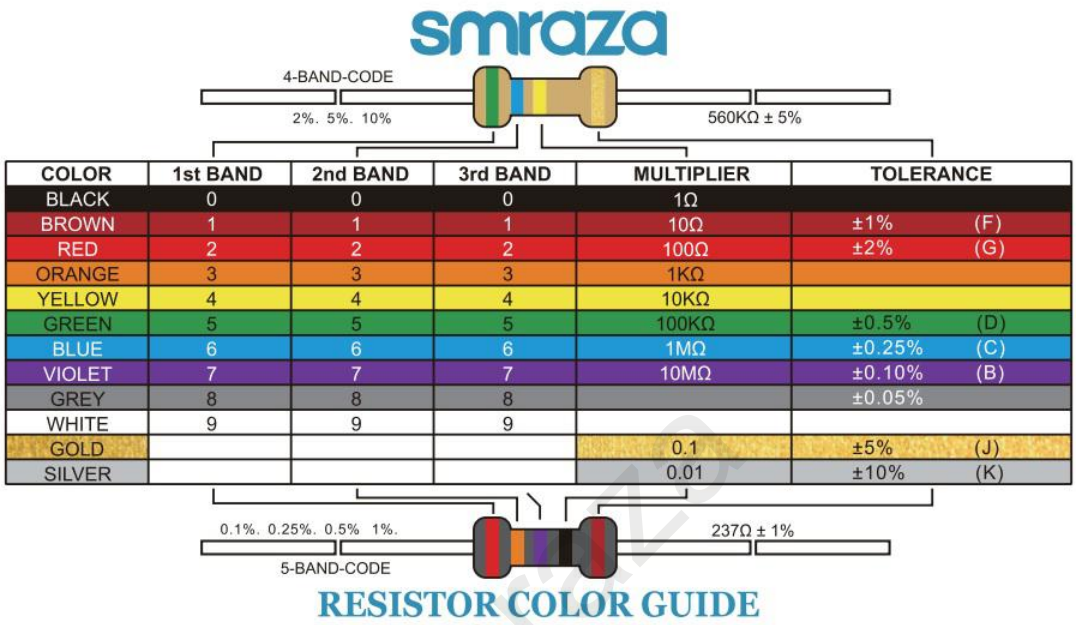
Component Introduction

LED:

In this lesson, you will use perhaps the most common of all LEDs: a 5mm red LED. 5mm refers to the diameter of the LED. Other common sizes are 3mm and 10mm. You cannot directly connect an LED to a battery or voltage source because 1) the LED has a positive and a negative lead and will not light if placed the wrong way and 2) an LED must be used with a resistor to limit or 'choke' the amount of current flowing through it; otherwise, it will burn out!

If you do not use a resistor with an LED, then it may well be destroyed almost immediately, as too much current will flow through, heating it and destroying the 'junction' where the light is produced.

RESISTORS:



COLOR	1st BAND	2nd BAND	3rd BAND	MULTIPLIER	TOLERANCE	
BLACK	0	0	0	1Ω		
BROWN	1	1	1	10Ω	±1%	(F)
RED	2	2	2	100Ω	±2%	(G)
ORANGE	3	3	3	1KΩ		
YELLOW	4	4	4	10KΩ		
GREEN	5	5	5	100KΩ	±0.5%	(D)
BLUE	6	6	6	1MΩ	±0.25%	(C)
VIOLET	7	7	7	10MΩ	±0.10%	(B)
GREY	8	8	8		±0.05%	
WHITE	9	9	9			
GOLD				0.1	±5%	(J)
SILVER				0.01	±10%	(K)

RESISTOR COLOR GUIDE

As the name suggests, resistors resist the flow of electricity. The higher the value of the resistor, the more it resists and the less electrical current will flow through it.

We are going to use this to control how much electricity flows through the LED and therefore, how brightly it shines.

The unit of resistance is called the Ohm, which is usually shortened to Ω the Greek letter Omega. Because an Ohm is a low value of resistance, we also denote the values of resistors in kΩ (1,000 Ω) and MΩ (1,000,000 Ω).These are called kilo-ohms and mega-ohms.

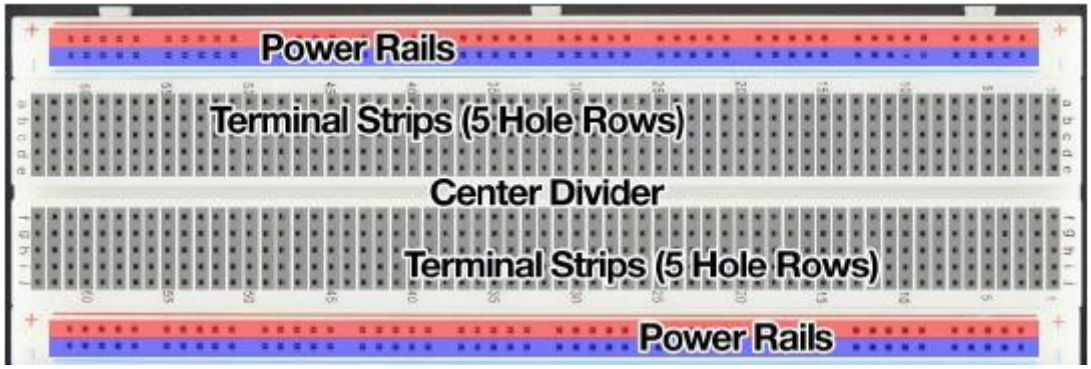
The resistor color code has three colored stripes and then a gold stripe at one end.

Unlike LEDs, resistors do not have a positive and negative lead. They can be connected either way around.

If you find this approach method too complicated, you can read the color ring flag on our resistors directly to determine its resistance value. Or you may use a digital multimeter instead.

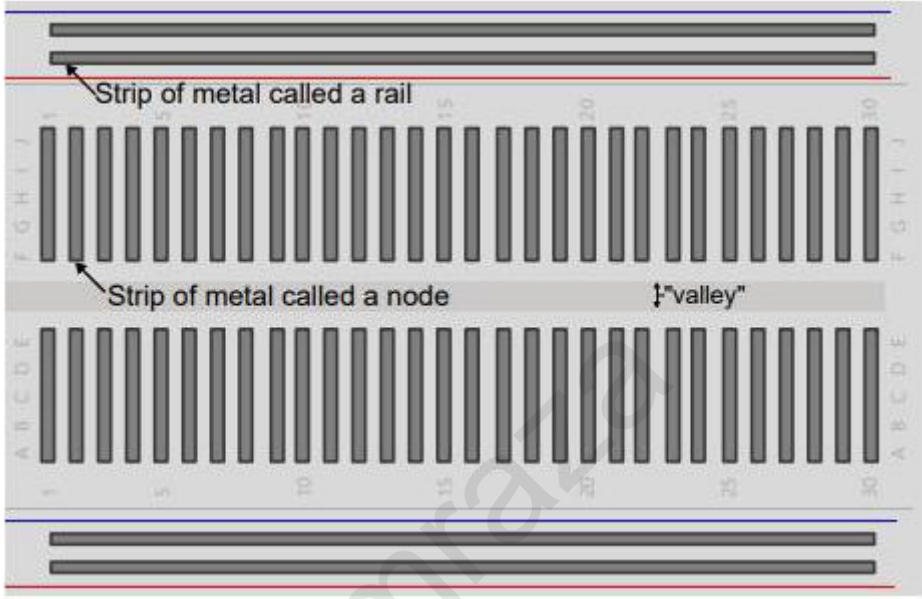
BREADBOARD:

A breadboard enables you to prototype circuits quickly, without having to solder the connections. Below is an example.
Just about every breadboard is made of three sections: Two sets of very long power rails and the large middle section that is full of those 5-hole-long terminal strips.



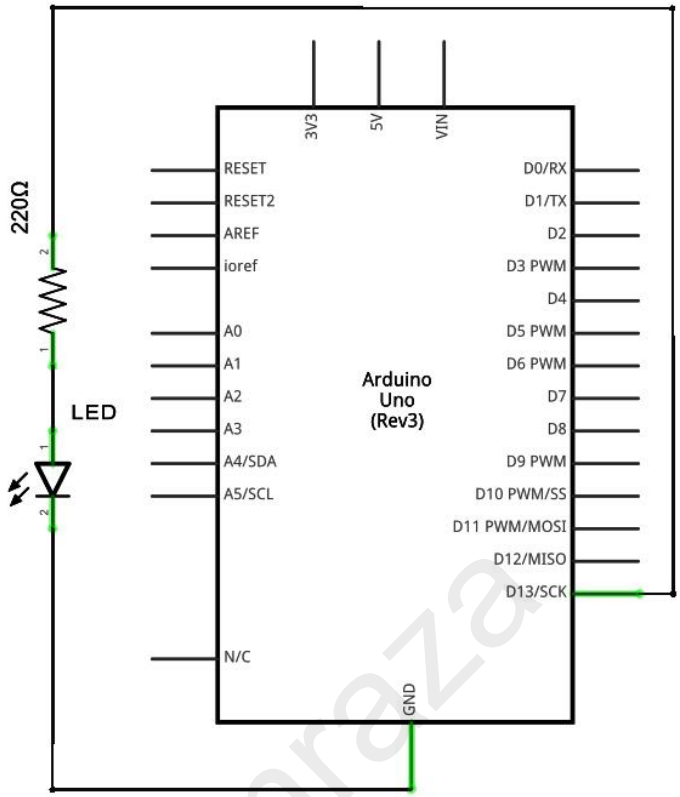
You put the components (buttons, LEDs, resistors, integrated circuits, etc)
in the middle section, with each pin connected to the rows terminal strip. The power rails are long columns used to distribute the power and ground connections along the entire circuit.
As you build circuits you'll quickly find that each part usually needs a connection to power or ground, so having a lot of power/ground pins available will be very handy. To help you keep track of which rail is ground and which is power, there's a red (+) and blue (-) stripe down the sides of the rails. Just make super-sure you connect positive to (+) and ground to (-) or you're gonna have a bad time!

Figure the next depict, using dark gray bars, the underlying electrical connections of the nodes and rails. There are various configurations of breadboards, with different numbers of nodes, rails, etc. But, the underlying electrical connections should always be fairly obvious, provided you understand the configuration described here. If you would like to read a bit more about breadboards, another discussion can be found via the box on the right.

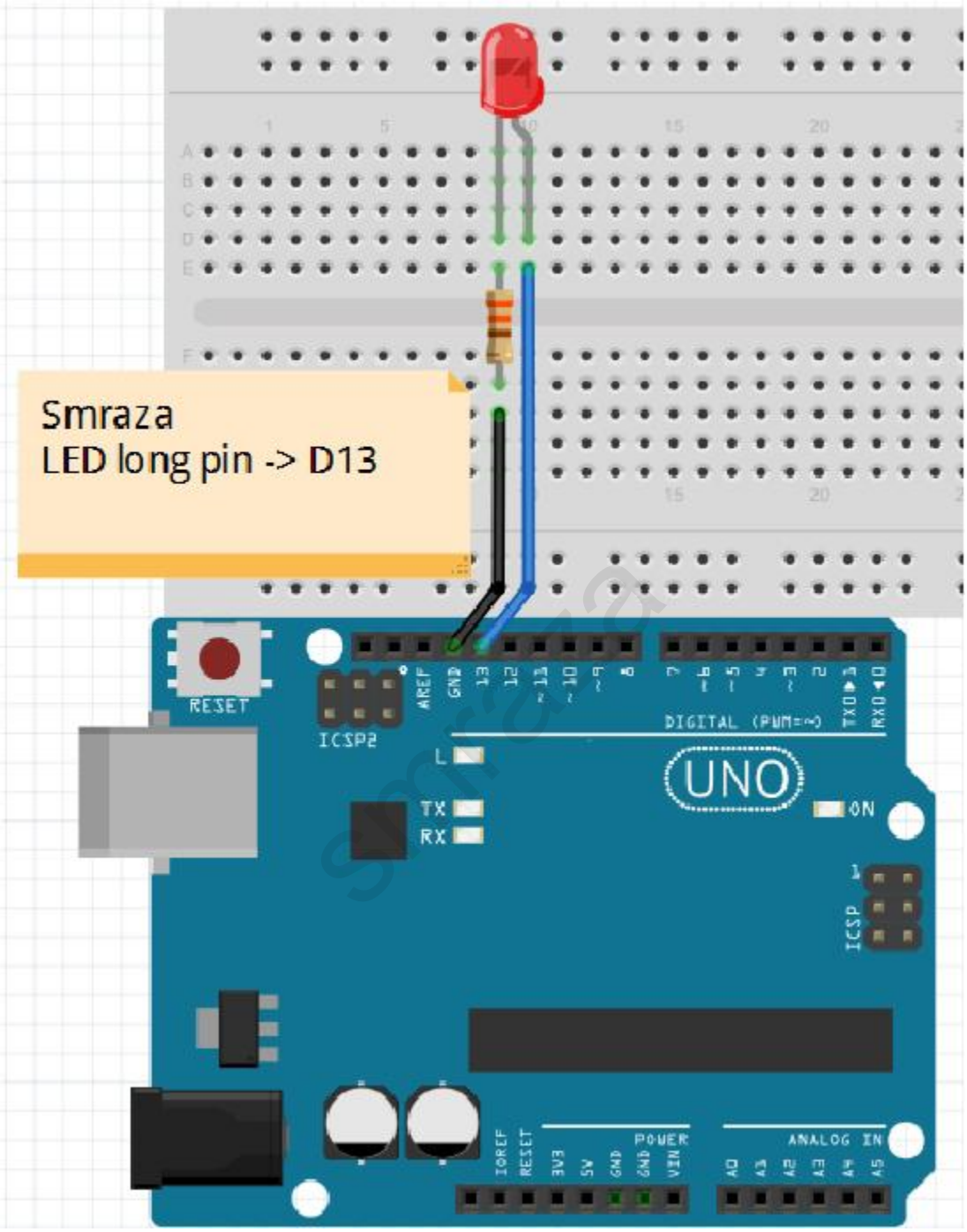


Connection

Schematic



Connection diagram



Note：The longest LED of the pin is connected to the digital signal port 13(D13).

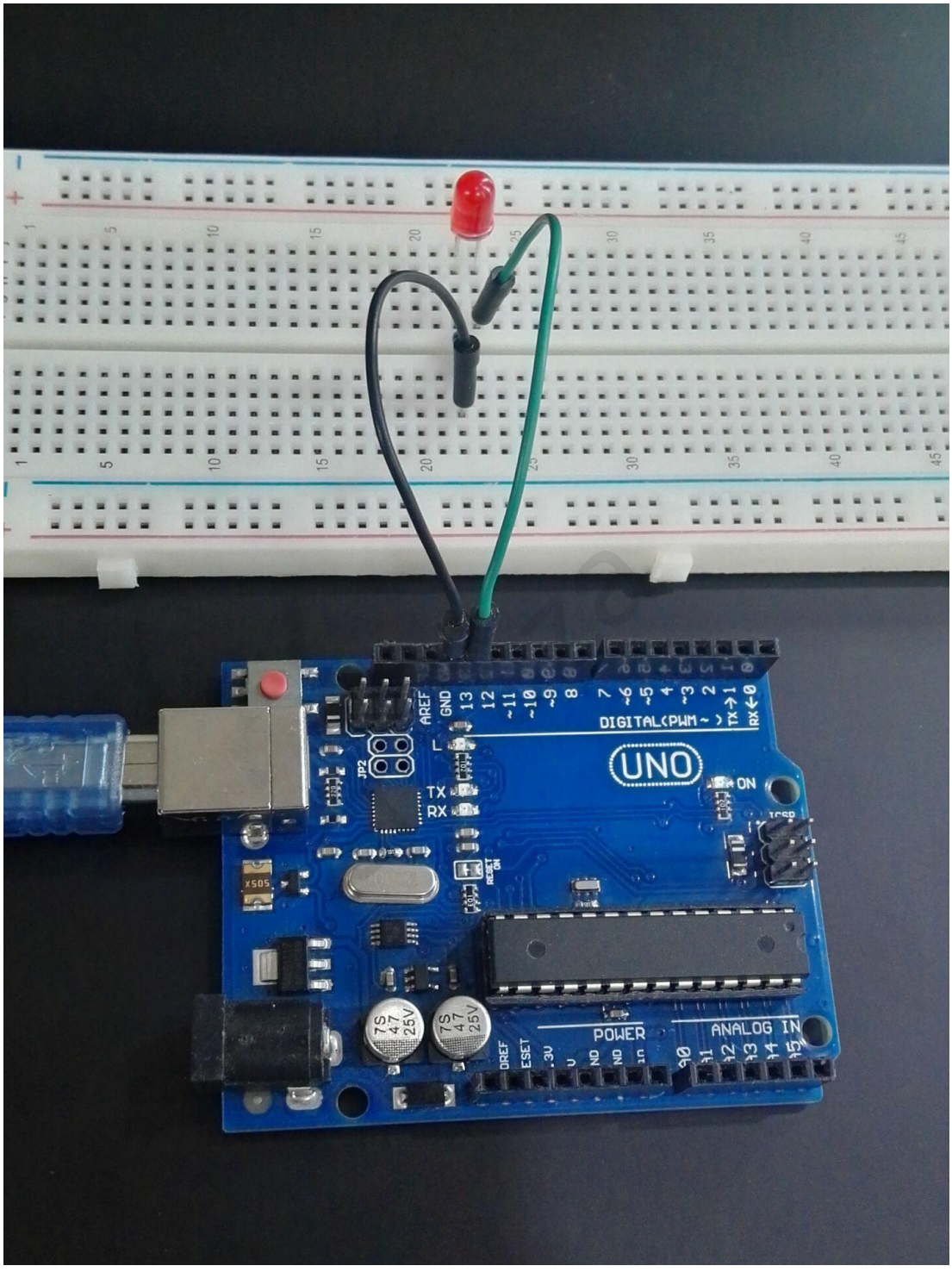
Sample code

Note: sample code under the **Sample code** folder

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;
// the setup routine runs once when you press reset:
void setup()
{
      // initialize the digital pin as an output.
      pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()
{
      digitalWrite(led, HIGH);
      delay(1000);                  // wait for a second
      digitalWrite(led, LOW);
      delay(1000);
}
```

Example picture



Language reference

Tips：click on the following name to jump to the web page.
If you fail to open, use the Adobe reader to open this document.

int
setup()
pinMode()
OUTPUT
loop()
HIGH
LOW
digitalWrite()
digitalRead()
delay()
; (semicolon)
{} (curly braces)
= (assign)
// (comment)

Application effect

Turns on an LED on for one second, then off for one second, repeatedly.