Arduino flicker Candle

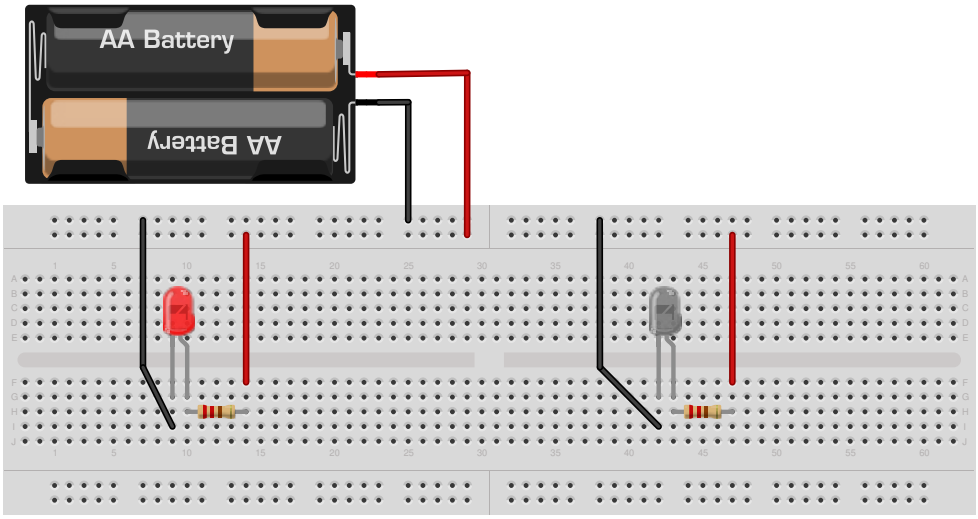
An Arduino controlled LED that can be turned on and off and flickers like a candle.
Enable native scrolling

Getting the Arduino IDE

Please download the Arduino software from their website https://www.arduino.cc/
(You have to click "Just download")

- Windows: installer or .zip
- Mac: .zip
- Linux: {apt, yum, pacman} install arduino

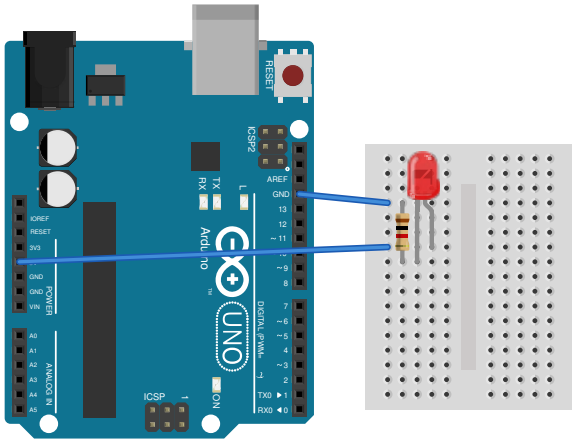
Internal connections on a breadboard



- Long horizontal power supply connections. Split into left an right half.
- Shorter vertical groups of five pins.

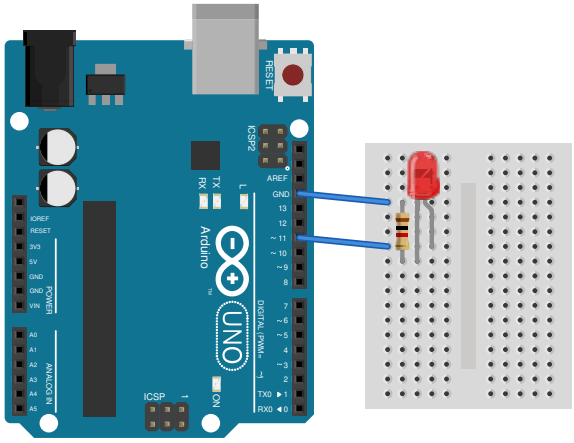
Connecting an LED

Put together the following circuit on your breadboards:



If the LED does not light you should try swapping its leads around.

Controlling the LED (hardware)



Remove the connection to the Arduino 5V pin and attach it to the Arduino pin 11 instead.

Controlling the LED (software)

Open the Arduino IDE, configure your Board and enter the following program:

```
1  void setup() {
2    pinMode(11, OUTPUT);
3  }
4
5  void loop() {
6    digitalWrite(11, HIGH);
7    delay(500);
8    digitalWrite(11, LOW);
9    delay(500);
10 }
```

Controlling the LED

After uploading the program, the LED should blink with a frequency of 1Hz.

Blinking faster

The statement `delay(500)` in the uploaded program determines how long the microcontroller should wait between the LED state changes.

In the case above it waits 500ms.

Lets see what happens if we decrease both delays by a factor of 100.

Blinking faster

The Human eye does no longer perceive the LED as blinking.

Instead the LED seems to be lit less brightly than before.

By varying the proportion between the LED beeing on and the LED beeing of how bright the LED seems to be lit.

PWM

The Process of varying the on/off proportion to change the perceived brightness is called pulse width modulation (PWM).

The Microcontroller used on the Arduino can also do PWM in hardware.

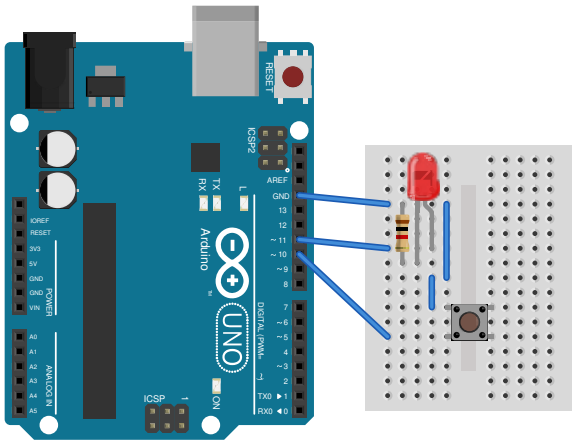
The corresponding function is rather missleadingly called `analogWrite(pin, value)`.

Hardware PWM

The following program will cycle through different led brightnesses.

```
1  void setup() {
2    pinMode(11, OUTPUT);
3  }
4
5  void loop() {
6    analogWrite(11, 0); delay(1000);
7    analogWrite(11, 63); delay(1000);
8    analogWrite(11, 127); delay(1000);
9    analogWrite(11, 191); delay(1000);
10   analogWrite(11, 255); delay(1000);
11 }
```


Connecting a button



- Connect one pin of a button to one Arduino GND pin.
- Connect the other pin on the same side of the button to Arduino pin 10.

Using the button

```
1  void setup() {
2    pinMode(11, OUTPUT);
3    pinMode(10, INPUT_PULLUP);
4  }
5
6  void loop() {
7    if (digitalRead(10) == true) {
8      analogWrite(11, 255);
9    }
10   else {
11     analogWrite(11, 5);
12   }
13   delay(100);
14 }
```

Not !

The LED gets dimmer when the button is pressed! It was supposed to be the other way around!

Executing `pinMode(10, INPUT_PULLUP)` connects an internal resistor to pin 10 pulling the voltage up (`digitalRead(10) == 1`).

When the button is pressed the button shorts pin 10 to 0V (GND) (`digitalRead(10) == 0`).

To invert the return value of `digitalRead` use `if (!digitalRead(10)) {`.

Using the button (correctly)

```
1  void setup() {
2    pinMode(11, OUTPUT);
3    pinMode(10, INPUT_PULLUP);
4  }
5
6  void loop() {
7    if (!digitalRead(10)) {
8      analogWrite(11, 255);
9    }
10   else {
11     analogWrite(11, 5);
12   }
13   delay(100);
14 }
```


Toggling between states

```
1  boolean is_on= true;
2  void loop() {
3    if (!digitalRead(10)) {
4      is_on= !is_on;
5    }
6    if (is_on) {
7      analogWrite(11, 255);
8    }
9    else {
10     analogWrite(11, 5);
11   }
12 }
```

...

Toggling more slowly

The LEDs toggle between on and off while the button is pressed.

We only want it to toggle once each time the button is pressed.

⇒ We need to track the last button state.

...

Randomness

To simulate the random flickering of an actual candle we need a source of randomness.

The Arduino environement supplies the `random(min, max)` function for this case.

Randomness

In the program below `rand_var` is randomly set to either 10, 11, 12, 13 or 14* each time `loop` is executed.

```
1  void setup() {
2  }
3
4  void loop() {
5    int rand_var= random(10, 15);
6  }
```

* The lower bound is inclusive. The upper bound is exclusive.

Combining the functions

You can now try to combine the `analogWrite`, `analogRead`, `random` and `delay` functions to create a nice looking Candle animation.

Working example

```
1  if (is_on) {
2    analogWrite(11, random(10, 255));
3    delay(random(20, 120));
4  }
5  else {
6    analogWrite(11, 0);
7    delay(100);
8  }
```

...

Appendix

What happens when we click the *Upload* button?

Compiling

Firstly the textual representation of the program is translated into a representation the microcontroller understands.

```
00000128 <main>:
128:  87 b3   in      r24, 0x17       ; 23
12a:  88 61   ori     r24, 0x18       ; 24
12c:  87 bb   out     0x17, r24       ; 23
12e:  30 d1   rcall   .+608           ; 0x390 <uart_init>
```


Uploading (Microcontroller)

Then the Computer instructs the microcontroller on the Arduino to restart by setting the special *RESET* pin to 0V.

Upon startup a special program on the microcontroller is executed.

This program is called the *bootloader*. It flashes the LED on the Arduino board and waits for programming instructions.

Uploading (Computer)

The Computer then starts sending the compiled program to the Arduino.

Once the upload is successfull the uploaded program is started.