Qualcomm
thinkabit lab

What Will You Learn?

Learn to wire a circuit, upload code to a microcontroller to make an LED Blink, and modify that code to change how it blinks.

Why Should You Learn This?

Blink is one of Arduino's basic example programs. It is a great way to begin to understand the programming language. You should complete the "LED circuit" activity before beginning to understand how an LED circuit works. During this activity, you will upload an LED Blink example program and then watch what happens when you change the code.

Here's What You'll Need:
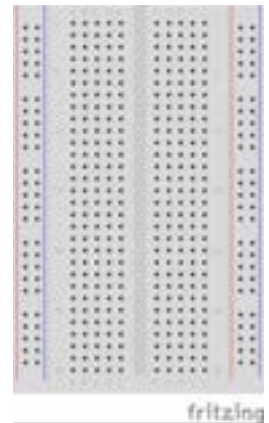
Arduino Microcontroller

LED

Notice that one of the wires of the LED is slightly longer than the other, the long leg is the positive side and the short leg is the negative side. This will be important when you plug in your LED.

Breadboard

Red Wire

Black Wire

A to B USB Cable

You will power the Arduino from a computer through the USB cord.

Resistor


Use a range of 100 – 400 ohms

Safety First

Never allow the red and black wires to touch together while they are connected to a powered microcontroller, as that creates a short circuit. A short circuit can potentially cause the wires and/or the Arduino board to get hot enough to burn the skin. In addition, in the event of a short circuit the Arduino board can potentially catch on fire.
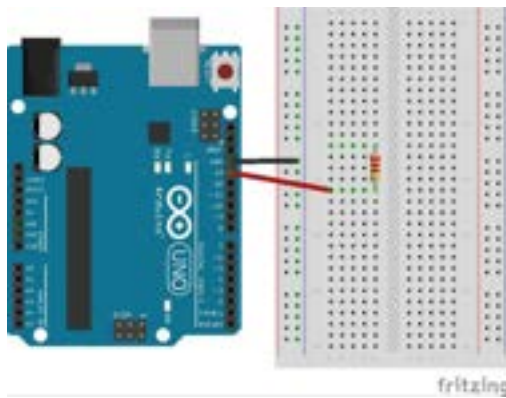
Let's Get Connected

1. Start by connecting the black wire from GND to the negative rail of the breadboard.
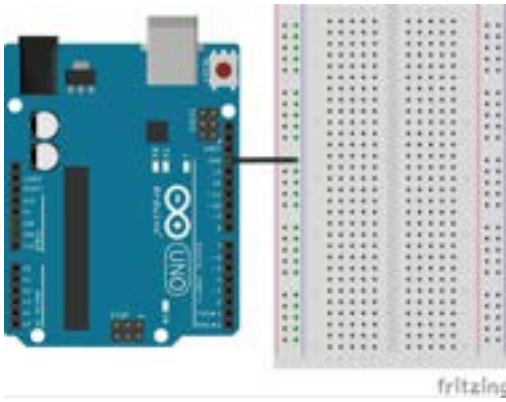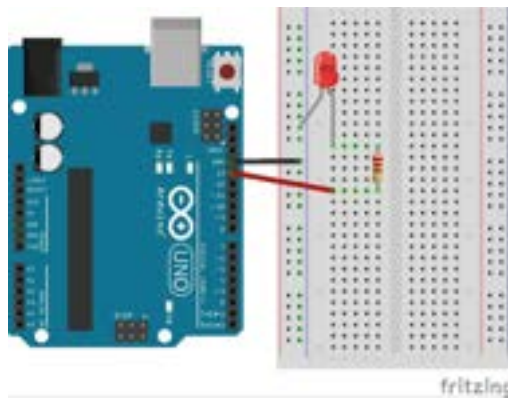

2. Next, connect the red wire from pin 13 to any row on the breadboard (shown using row 16).
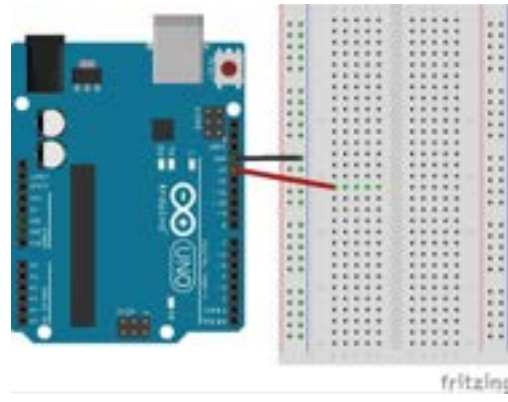
3. Let's add the resistor. Use it to connect row 16 (where the red wire is) to any other row on the breadboard (shown using row 12).

4. Lastly, place the LED. The long leg (positive) will go in the row with the resistor (row 12) and the short leg (negative) will go in the negative rail.

Time to Upload Code

You can think of sending code to the microcontroller like sending a text message. You open a texting app (in this case you'll use the Arduino software), choose someone to send the message to (select the correct communication portal, or COM#, for the Arduino board), compose a message (write some code), then hit send (upload the program). This activity will walk you through each of these steps.

1. First, you need to open the app that you're going to use to send out your "text." We'll use the Arduino programming software on our computer.


Note that the image above uses Windows to run Arduino. Other operating systems may have instructions that look slightly different, but the concept is the same.

2. Next, you need to identify in your phone who you are texting. In the program, this means selecting the correct communication port by going to the drop down menu and choosing the port that says Arduino Uno above it. If the board name does not appear, please choose the port with the highest COM#. Please ensure your microcontroller is plugged into the computer prior to choosing the communication port.
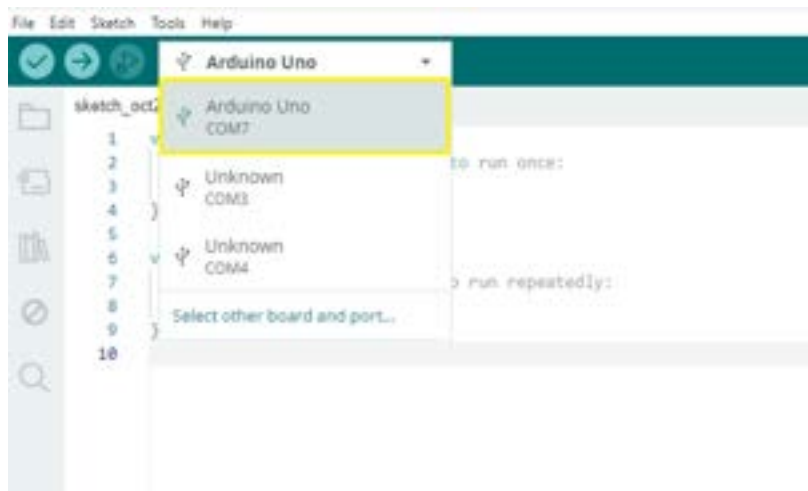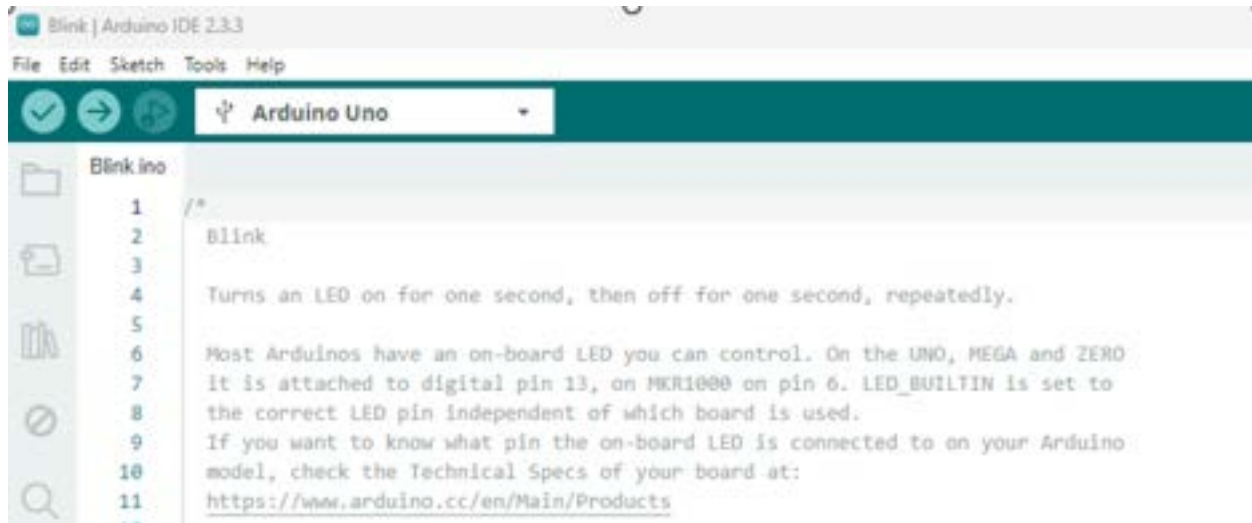
3. Next, you need to type in your message. In this activity, you are going to use one of Arduino's built in example programs for your message. Please open the "Blink" example code by going to File -   Examples -   01. Basics -   Blink.
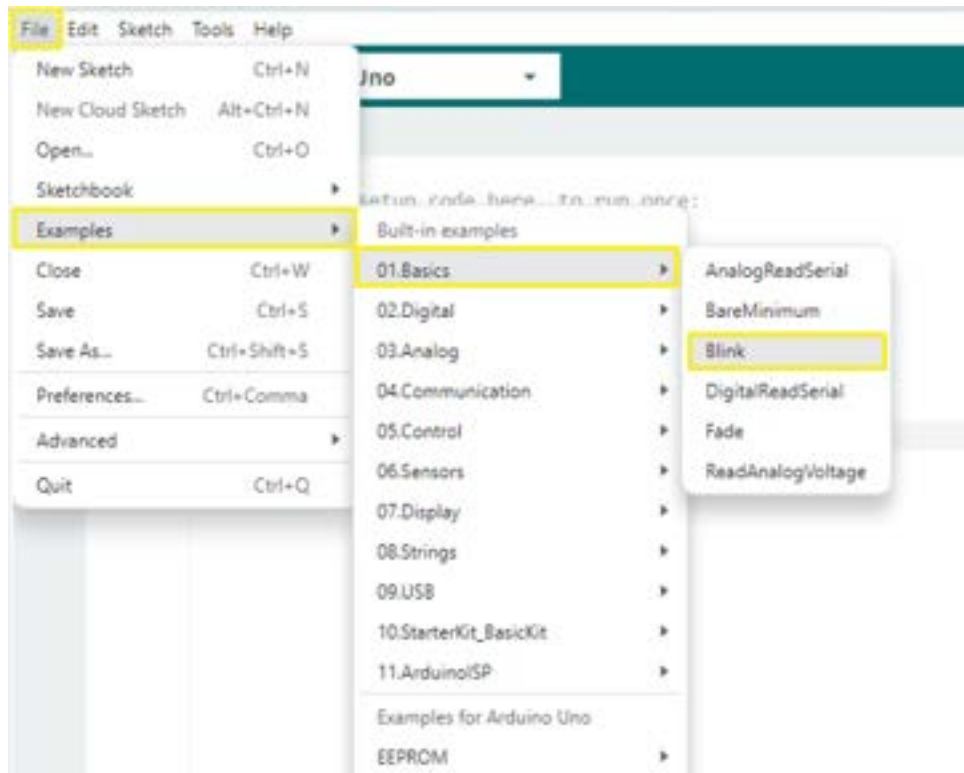
4. Finally, you need to "send" your text message. In this programming language, this is called "uploading". It is done by clicking the arrow button in the top left corner of your screen. Once this code uploads, your LED should begin to blink.

5. Let's look through each section of the code so that you understand what is happening. The first part of the program is grey, which means that it is made up of "comments." These comments do not get uploaded to the microcontroller, they are just for humans to read. You can use them when you want to share your code with someone else or if you want to remember what certain parts of the code do.

6. The first bit of real code is "**void setup**()**{**". This sets up your board. The code we write within the setup function lets it know what pins you are using and how each pin will act. There is one line of code in this setup, "**pinMode**(LED_BUILTIN, OUTPUT);", which programs the microcontroller to set pin 13 (our pin that is specifically built to power an LED and is referred to as our LED_BUILTIN) to output mode. Output mode means it can send power. As a result, your LED will be plugged into pin 13.

7. The next section of this code is "**void loop**()". This is the action part of the code and this section will repeat forever until you disconnect the power, reset the board, or if you coded for it to leave the loop. There are 4 lines of code in this loop and they will repeat, line 1, line 2, line 3, line 4, line 1, line 2, line 3, line 4, line 1, etc.

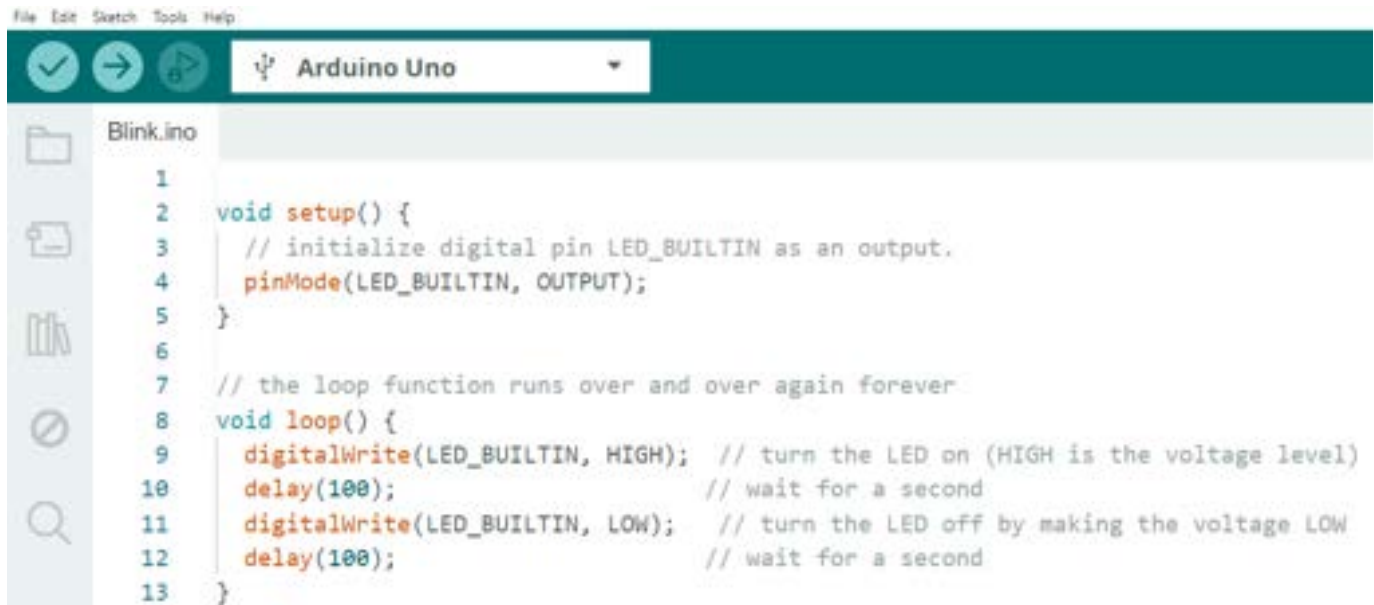   a. The first line of code, "**digitalWrite**(LED_BUILTIN, HIGH);" sets pin 13 (our LED_BUILTIN) to a high voltage and turns the LED on.
   b. The next line, "**delay**(1000);" programs the microcontroller to wait for 1000 milliseconds (or 1 second) before reading the next line of code. Since you haven't set the LED to off yet, the LED will simply stay on.
   c. The next line of code, "**digitalWrite**(LED_BUILTIN, LOW);" will set pin 13 to low voltage, making the LED turn off.
   d. Lastly, "**delay**(1000);" programs the microcontroller to wait for 1 second again before reading the next line of code. So, the LED turns on, waits for 1 second, then turns off and waits for 1 second. This makes the light blink.

```
Blink | Arduino IDE 2.3.3
File  Edit  Sketch  Tools  Help

        Arduino Uno

Blink.ino
    1
    2
    3    // the setup function runs once when you press reset or power the board
    4    void setup() {
    5      // initialize digital pin LED_BUILTIN as an output.
    6      pinMode(LED_BUILTIN, OUTPUT);
    7    }
    8
    9    // the loop function runs over and over again forever
   10    void loop() {
   11      digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
   12      delay(1000);                       // wait for a second
   13      digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
   14      delay(1000);                       // wait for a second
   15    }
   16
```

8. Now that you have a basic understanding of how this code works, how could you change the code to make the light blink faster?

You can change both delays to a smaller value to make the light blink faster. Don't forget to upload once you have changed the code.


```
File  Edit  Sketch  Tools  Help

     Arduino Uno          ▼

Blink.ino
   1
   2   void setup() {
   3     // initialize digital pin LED_BUILTIN as an output.
   4     pinMode(LED_BUILTIN, OUTPUT);
   5   }
   6
   7   // the loop function runs over and over again forever
   8   void loop() {
   9     digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  10     delay(100);                        // wait for a second
  11     digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  12     delay(100);                        // wait for a second
  13   }
```

Notice that the comments still read "wait for a second," but remember comments don't affect the code. It is good practice to have comments reflect the code, so you may choose to modify them to be correct.

Tips for Success

- My light is not blinking: Be sure that all your wires are plugged into the right places on the breadboard and microcontroller (check the diagram in step 4 of "Let's Get Connected"). Pin 13 can be tricky to find sometimes and you may be one off to the left or to the right. Try moving the red wire around to find 13.

- Double check to see if your LED is plugged in the right way. The LED won't work if it's plugged in backwards, so try flipping the direction.

- If you're sure the wire is in pin 13 and it still is not blinking, make sure you uploaded your code to your Arduino. At the bottom of the screen it should read "done uploading." If not, try uploading again.

- When I upload, I get an error. Make sure you are connected to the correct Port and that the code is the original example.

Are You Ready for Some Challenges?

1. Did you know that all the lights in the room you're currently in are blinking? Your eyes just aren't fast enough to see the blinking. This is how flat screen TV's and smart phone screens work, but don't believe everything you read! Prove to yourself that you can make the LED blink so fast that you can't see it blink anymore. Make sure both delays in your program are the same for this challenge.

2. Now that you know that a blinking light could appear solid, try to find the threshold value of visible blinking. In other words, try to find the largest value that you can put into both delays and you can't see the light blink, but if you add 1 to that value you would see it flicker. Hint: it's greater than 1 and less than 50.

3. Make your LED Blink on and off for random amounts of time for the delay. "Random" is a function in Arduino that will select a random value within a given range. So if you write "**random(0,10)**" inside the parentheses of "**delay()**", it will pick a random number between 0 and 10. For this challenge, it is recommended to use a range of 20 to 1000, but you can always try different values as well.

Challenge Solutions

1. Any value less than 10 will work for this challenge. For example:

```
File  Edit  Sketch  Tools  Help

Arduino Uno

Blink.ino
1
2   void setup() {
3     // initialize digital pin LED_BUILTIN as an output.
4     pinMode(LED_BUILTIN, OUTPUT);
5   }
6
7   // the loop function runs over and over again forever
8   void loop() {
9     digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
10    delay(100);                       // wait for a second
11    digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
12    delay(10);                        // wait for a second
13  }
```

2. The threshold is usually at 10 (like the example above), sometimes it is 9, 11, or 12 because there is slight variability in either the color of the light, the value of the resistor, or maybe the human.

3. The random function goes inside of the delay.

```
File  Edit  Sketch  Tools  Help

Arduino Uno

Blink.ino
1
2   void setup() {
3     // initialize digital pin LED_BUILTIN as an output.
4     pinMode(LED_BUILTIN, OUTPUT);
5   }
6
7   // the loop function runs over and over again forever
8   void loop() {
9     digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
10    delay(random(20-1000));           // wait for a random amount of time
11    digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
12    delay(random(2-1000));            // wait for a random amount of time
13  }
```