Programmed Control with Arduino

Activity 1: My First LED (Continuous On)

The first stop on our journey is to understand how energy flows through our Arduino Uno board and to learn how to issue very basic instructions. Today you will succeed in turning on a light-emitting diode (LED) by programming your first lines of code in C++.

Learning Objectives

By the end of this session, you will be able to:

- Identify the fundamental structure of an Arduino program (`setup` and `loop`).
- Configure a digital pin as an output channel using `pinMode`.
- Control the flow of electrical current using the `digitalWrite` command.
- Correctly connect an LED and a protective resistor on a breadboard.

Required Components (TinkerCad)

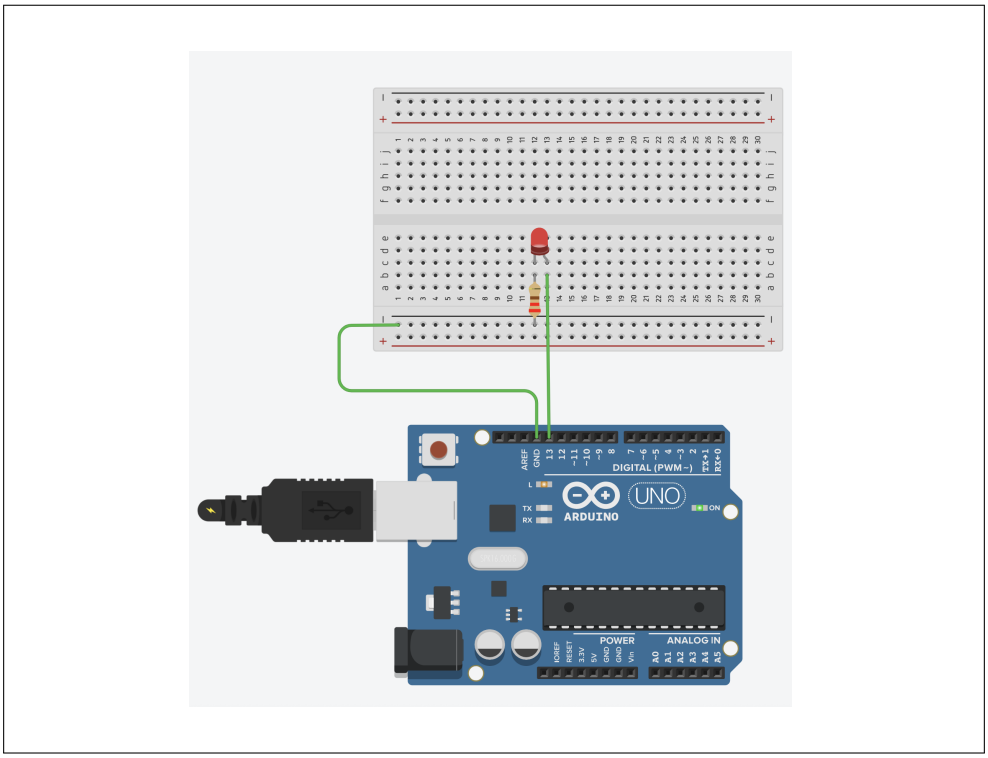
For this activity, you only need to drag the following components onto your screen:

- 1 Arduino Uno board.
- 1 Small breadboard.
- 1 LED (you can choose whichever color you like best).
- 1 $220\,\Omega$ resistor *(Important! If you do not include the resistor, the LED could burn out due to excessive current)*.
- Virtual jumper wires.

Breadboard Assembly Diagram

To connect the components safely in TinkerCad, carefully follow these guidelines:

Initial configuration of the Terminal object



Base Code: The Scaffolding

Open the code editor in TinkerCad, select **Text** mode (clear the default code if necessary), and write the following program. Read the green comments carefully to understand what each line does.

```
// Activity 1: Continuous on of an LED
// This function runs once when the Arduino powers up or resets
void setup() {
  // Configures Digital Pin 13 as a current OUTPUT
  pinMode(13, OUTPUT);
}

// This function runs continuously in an infinite loop
void loop() {
  // Sends a HIGH state (HIGH / 5 Volts) to Pin 13 to turn on the LED
  digitalWrite(13, HIGH);
}
```

Listing 1: Code to turn on an LED continuously.

How does the code work?

- `setup` : This is the preparation block. Here, we tell Arduino which pins we will use and how they are going to behave (as inputs or outputs of information).

- `loop` : This is the main engine of the program. When it finishes executing the last instruction in this block, it immediately restarts from the first one in an endless cycle.
- `pinMode` : Defines whether the selected pin works by sending energy (`OUTPUT`) or receiving it ( ).
- `digitalWrite` : Allows electrical voltage to be applied to the specified pin. If we send `HIGH`, the pin will provide 5V. If we send `LOW`, it will remain at 0V (no current).

Activity 1 Challenge

Now that you have verified that the circuit works in the simulator, demonstrate what you have learned by solving this challenge:

1. Modify the connections in TinkerCad so that the LED does not use pin 13, but rather **digital pin 8**.
2. Make the necessary changes in the code so that the program continues to turn on the LED continuously.
3. *Question for reflection:* If you wanted to turn the LED completely off using the program, which instruction would you have to modify in the `loop` function and with what parameter?

Try changing the loop limit to 100 and you'll see how fast your computer counts!