DOCUMENTATION

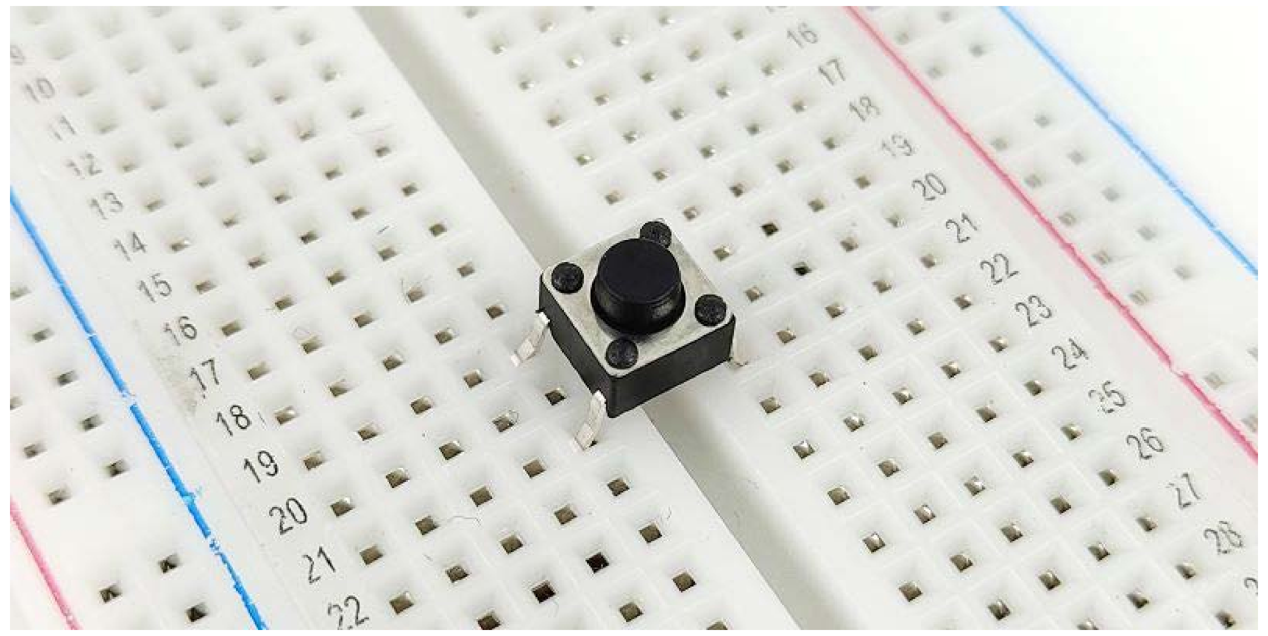
ACTIVITY

ASSESSMENT

USEFUL LINKS

BUTTON TRIGGER

Button Edge Detection



DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

CONTINUOUS READING

VS

STATE CHANGE
(EDGE DETECTION)

DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

CONTINUOUS READING

The following code turns an LED on/off using a push button.

- wiring link
- download code link

Run the code and observe:

- Does the LED turn on/off when you press the button?

Now open the Serial Monitor.

- What data is being printed?
- Is it printing once per press, or many times?

... WHY IS THIS A PROBLEM?

Arduino IDE + Serial Monitor screenshot

DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

CONTINUOUS READING

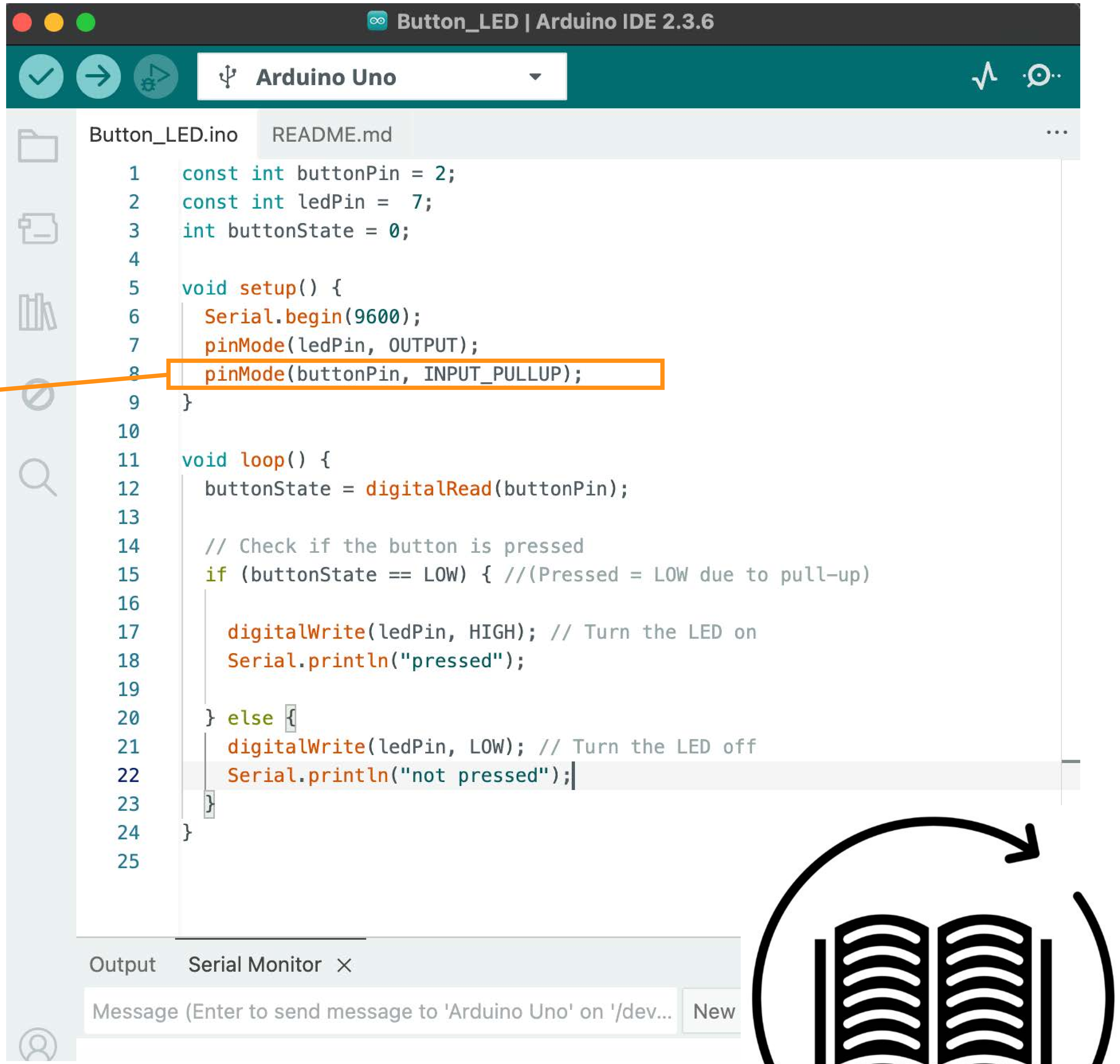
In this code, we turn the LED on/off by directly following the data coming from the button — digitalRead(buttonPin).

As a result, we get a flood of signals, happening multiple times per second (on every loop) — which you can see in the Serial Monitor

This behavior can be problematic in more complex projects, for example when triggering sounds, motors, or other actions that should only happen once per press.

SOLUTION?

Edge Detection!


Button_LED | Arduino IDE 2.3.6
Arduino Uno
Button_LED.ino
README.md
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on '/dev...
New

```
const int buttonPin = 2;
const int ledPin =  7;
int buttonState = 0;

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  buttonState = digitalRead(buttonPin);

  // Check if the button is pressed
  if (buttonState == LOW) { //(Pressed = LOW due to pull-up)

    digitalWrite(ledPin, HIGH); // Turn the LED on
    Serial.println("pressed");

  } else {
    digitalWrite(ledPin, LOW); // Turn the LED off
    Serial.println("not pressed");
  }
}
```

STATE CHANGE
(EDGE DETECTION)

WHY IT WORKS?

Edge Detection means your code responds only when the button changes state —
instead of constantly reacting to the raw signal coming from the button.

This avoids repeated signals and lets your code react once per press or release.

- Download Code link
- Diagram link

Run the code, then open your Serial Monitor.
How is the button performing now?

DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

HOW IT WORKS?

To achieve Edge Detection we compare the current state of the button to its previous state:

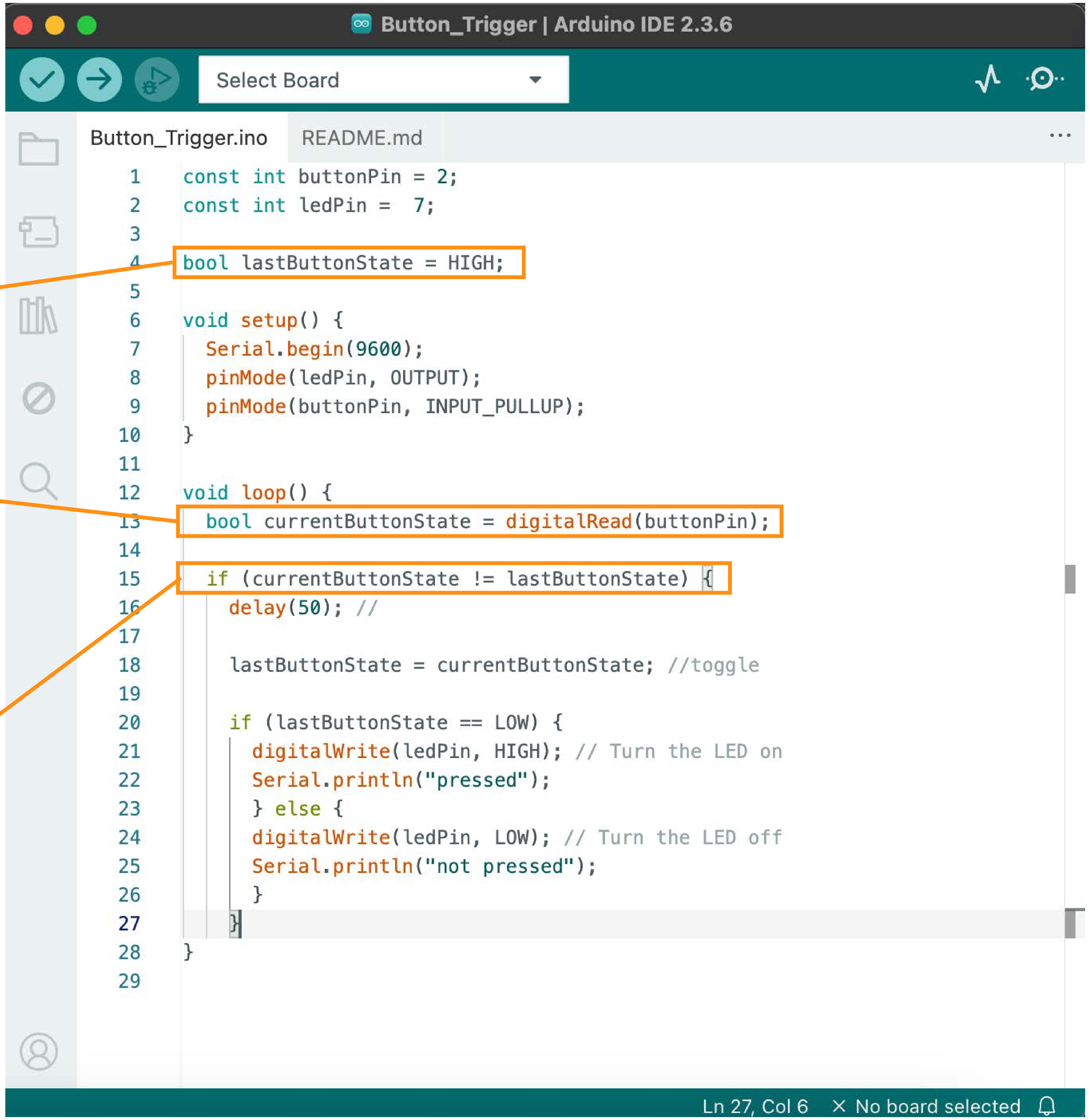
bool lastButtonState;
bool currentButtonState;

Our currentButtonState is constantly reading signals from the button.

However, lastButtonState is only updated when a change in state is detected — from HIGH to LOW or LOW to HIGH.

If the two values are different:
if (currentButtonState != lastButtonState)

...it means the button was just pressed or released. This change can triggers single actions — like playing a sound or starting a motor.



```
const int buttonPin = 2;
const int ledPin =  7;

bool lastButtonState = HIGH;

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  bool currentButtonState = digitalRead(buttonPin);

  if (currentButtonState != lastButtonState) {
    delay(50); //

    lastButtonState = currentButtonState; //toggle

    if (lastButtonState == LOW) {
      digitalWrite(ledPin, HIGH); // Turn the LED on
      Serial.println("pressed");
      } else {
      digitalWrite(ledPin, LOW); // Turn the LED off
      Serial.println("not pressed");
      }
    }
}

```

DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

CHALLENGE:

1. Add a second button to your circuit.

2. Modify your code, so:
   a) When Button-A is pressed:
      - The LED turns ON
      - The Serial Monitor prints “Hello”

   b) When Button-B is pressed:
      - The LED turns OFF
      - The Serial Monitor prints “Good Bye”

3. Upload a video (max 10 seconds) showing your achievements.

VIDEO UPLOAD

DOCUMENTATION

ACTIVITY

ASSESSMENT

USEFUL LINKS

FURTHER LEARNING

- Edge Detection Tutorial by Arduino Oficial Website