

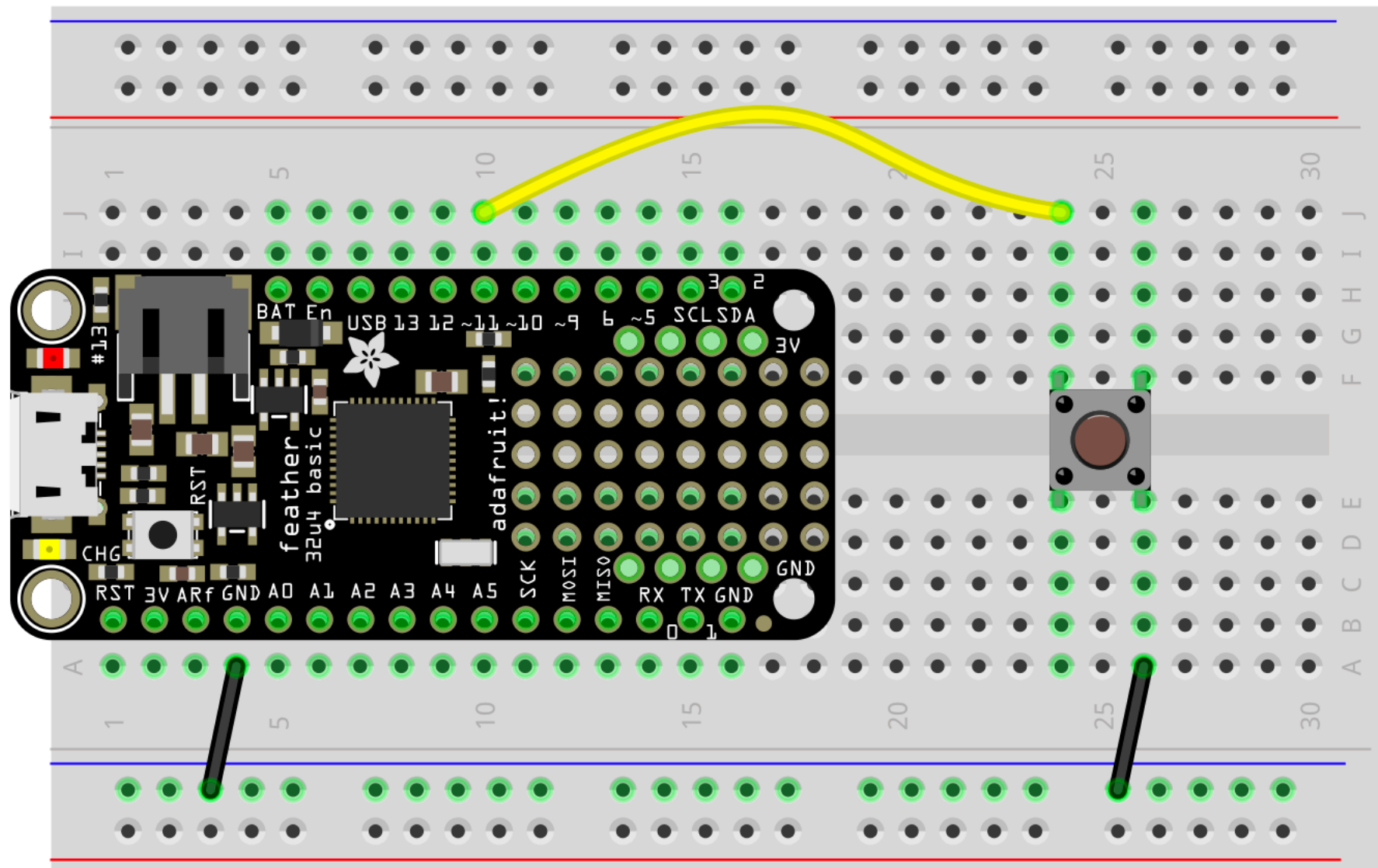
ARDUINO

+

PROCESSING

ARDUINO HID -> COMPUTER

BUTTON INPUT



TRANSLATE BUTTON PRESS INTO KEYBOARD PRESS

```
keyboard | Arduino 1.8.16

keyboard
1 #include "Keyboard.h"
2
3 const int BUTTON_PIN = 11;
4
5 void setup() {
6   pinMode(BUTTON_PIN, INPUT_PULLUP);
7
8   // initialize control as a keyboard
9   Keyboard.begin();
10 }
11
12 void loop() {
13   // read the pushbutton:
14   int val = digitalRead(BUTTON_PIN);
15
16   if (val == LOW) {
17     Keyboard.print("x");
18     delay(100);
19   }
20 }

Done uploading.
done in 0.019 seconds
CPU reset.

17 Adafruit Feather M0 Express, Small (-Os) (standard), Arduino, Off on /dev/cu.usbmodem14201
```

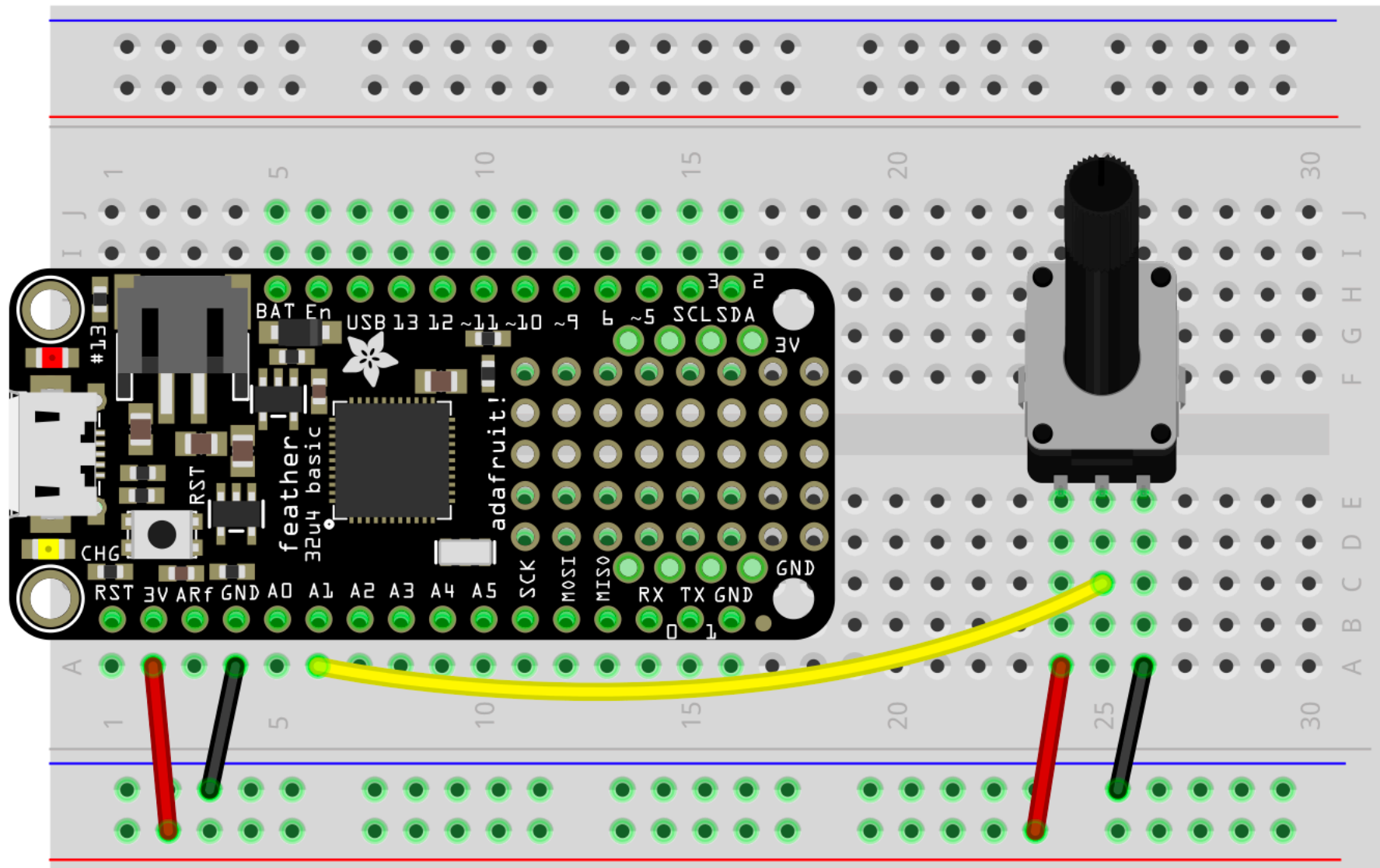
The screenshot shows a code editor window titled "sketch.js — keyboard". The Explorer panel on the left shows a folder named "KEYBOARD" containing files: "libraries", "index.html", "jsconfig.json", "sketch.js", and "style.css". The "sketch.js" file is selected. The main editor area shows the following JavaScript code:

```
1  let bgColor = 0;
2
3  function setup() {
4    createCanvas(windowWidth, windowHeight);
5  }
6
7  function draw() {
8    background(bgColor);
9  }
10
11 function keyPressed() {
12   bgColor = color(random(0, 255), random(0, 255), random(0, 255));
13 }
14
```

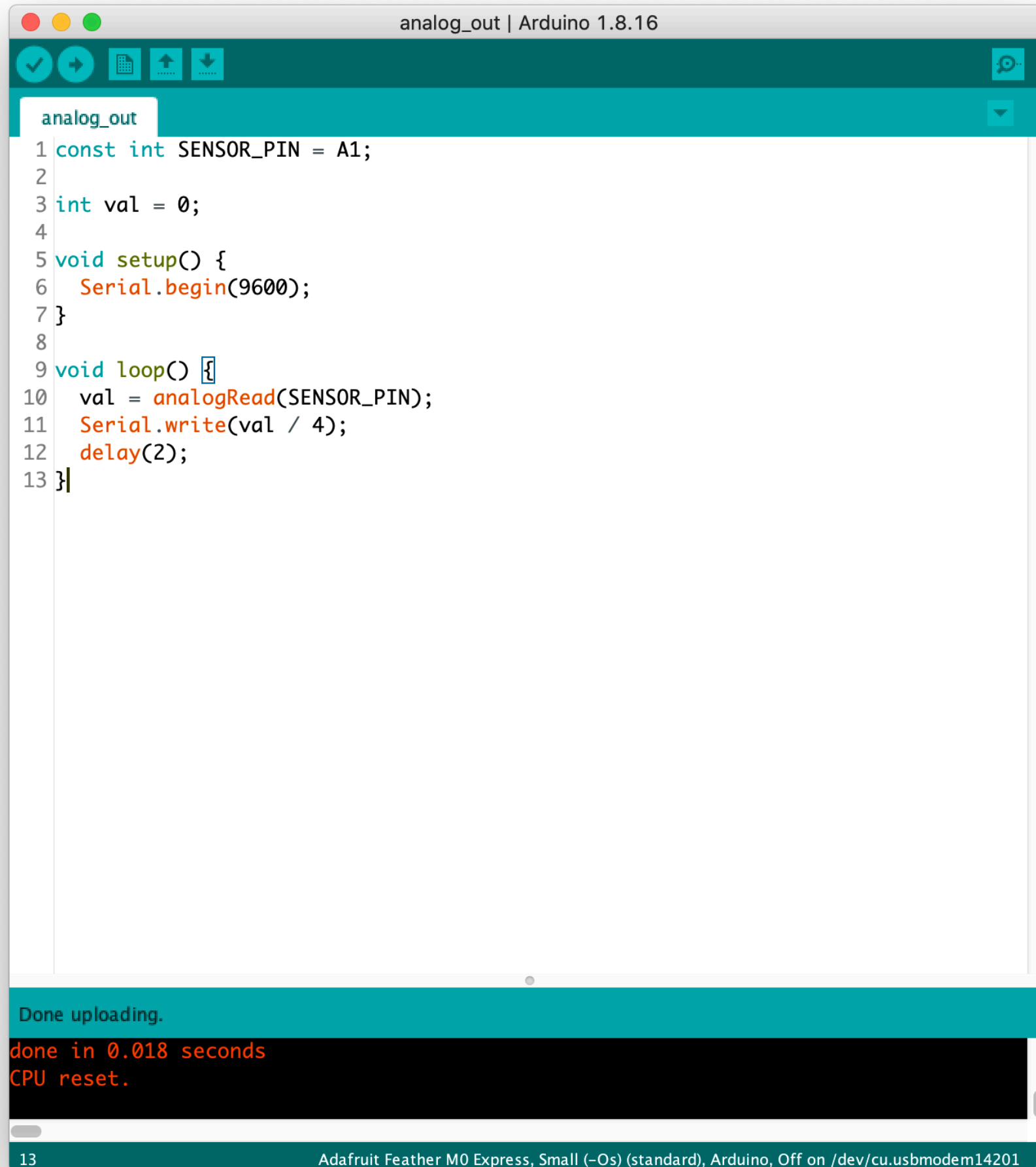
The bottom status bar shows the following information: "main*", "0 ↓ 1 ↑", "0 0", "Spaces: 2", "UTF-8", "LF", "{ }", "JavaScript", "Indents: 0", "Port : 5500", "✓ Spell", "✓ Prettier", and a user icon.

RESPOND TO KEYBOARD PRESSES

POTENTIOMETER (OR OTHER ANALOG SENSOR) INPUT



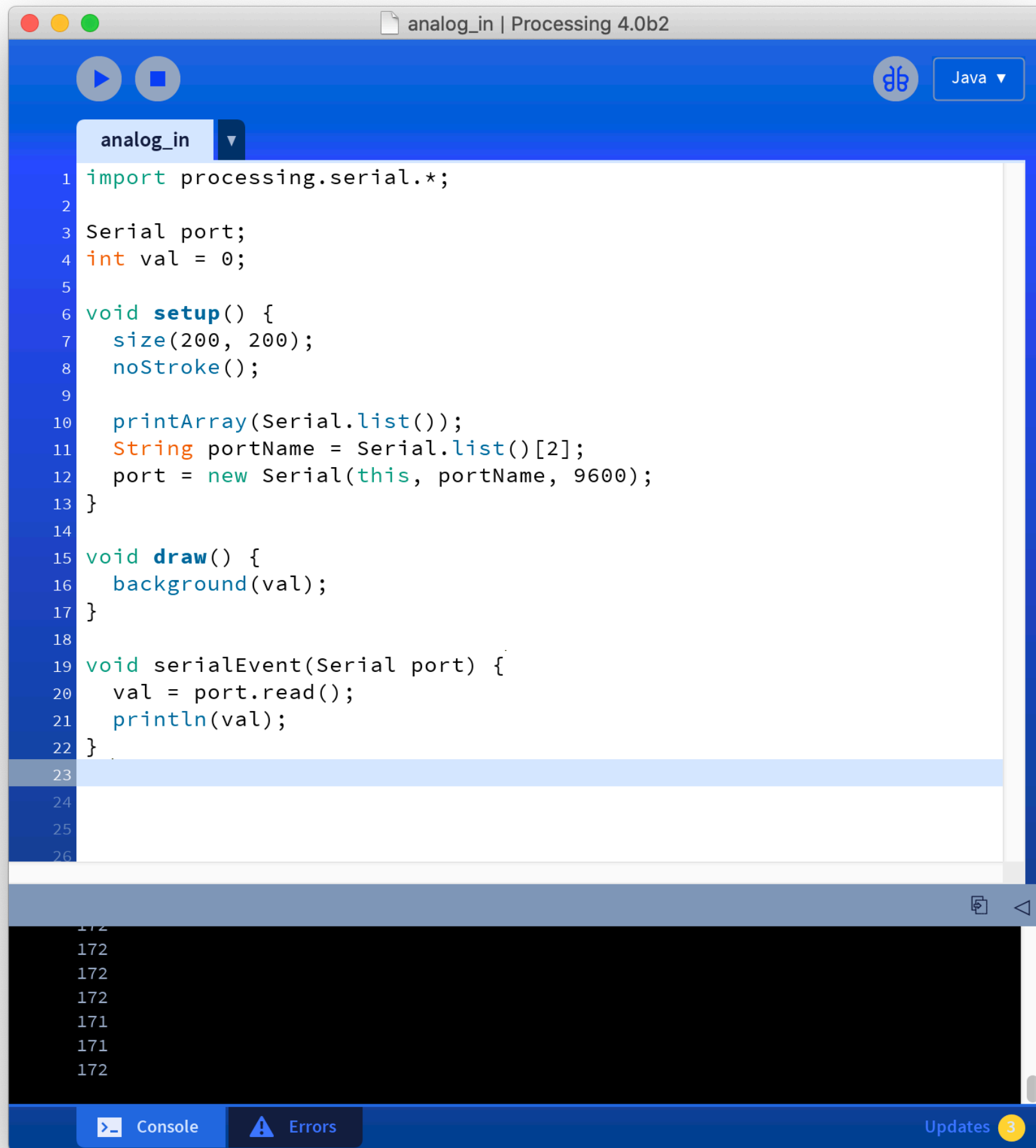
WRITE VALUES VIA SERIAL WITH ARDUINO



The screenshot shows the Arduino IDE interface. The title bar reads 'analog_out | Arduino 1.8.16'. The menu bar includes 'File', 'Edit', 'Tools', and 'Help'. The toolbar contains icons for checking, running, uploading, and downloading. The main editor area shows the following code:

```
1 const int SENSOR_PIN = A1;
2
3 int val = 0;
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  val = analogRead(SENSOR_PIN);
11  Serial.write(val / 4);
12  delay(2);
13 }
```

Below the editor is a status bar showing 'Done uploading.' and 'done in 0.018 seconds CPU reset.' The bottom status bar indicates the board is 'Adafruit Feather M0 Express, Small (-Os) (standard), Arduino, Off on /dev/cu.usbmodem14201'.



analog_in | Processing 4.0b2

analog_in

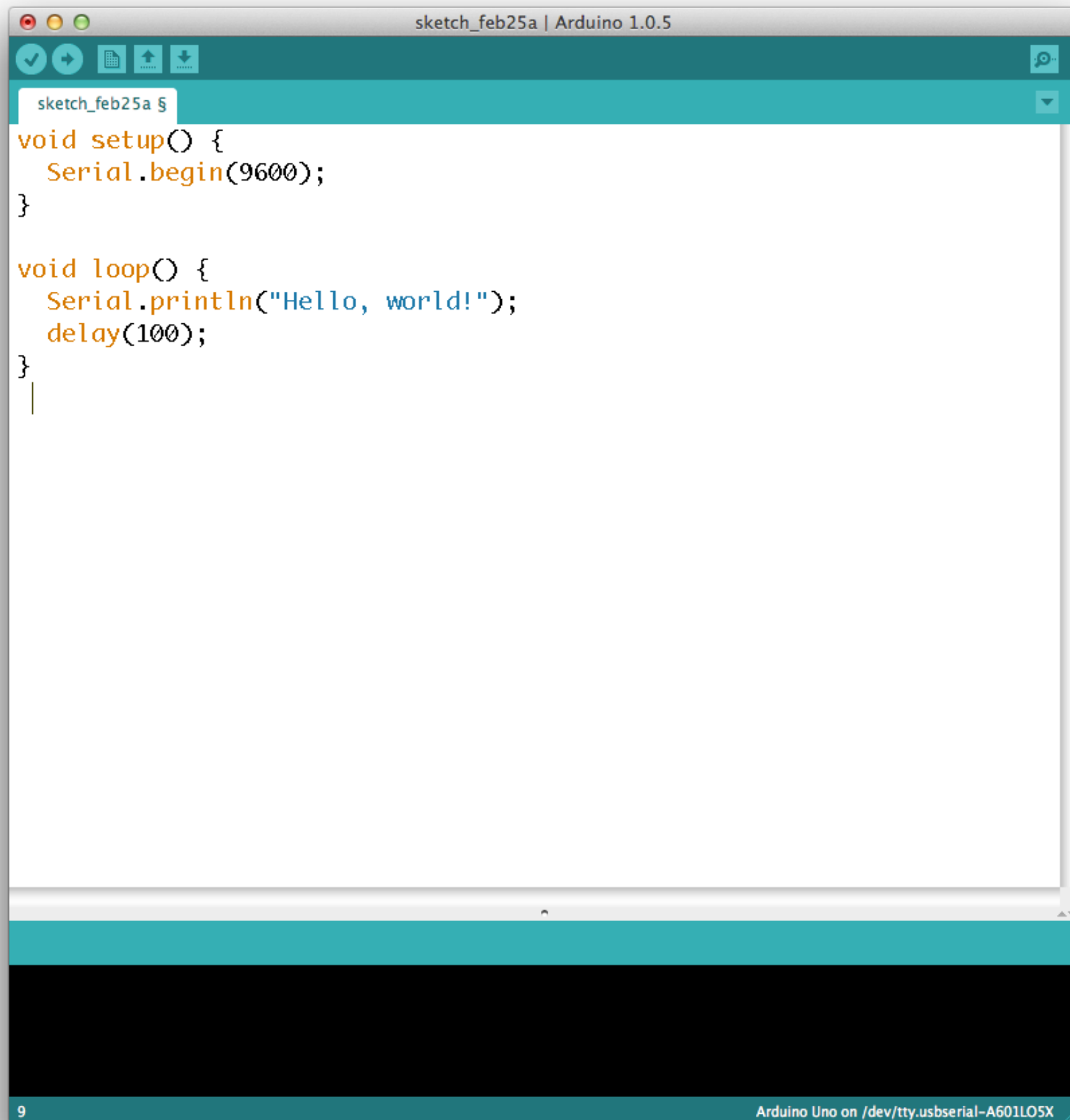
```
1 import processing.serial.*;
2
3 Serial port;
4 int val = 0;
5
6 void setup() {
7   size(200, 200);
8   noStroke();
9
10  printArray(Serial.list());
11  String portName = Serial.list()[2];
12  port = new Serial(this, portName, 9600);
13 }
14
15 void draw() {
16   background(val);
17 }
18
19 void serialEvent(Serial port) {
20   val = port.read();
21   println(val);
22 }
23
24
25
26
```

172
172
172
172
171
171
172

Console Errors Updates 3

READ VALUES VIA SERIAL WITH PROCESSING

ARDUINO SERIAL -> PROCESSING



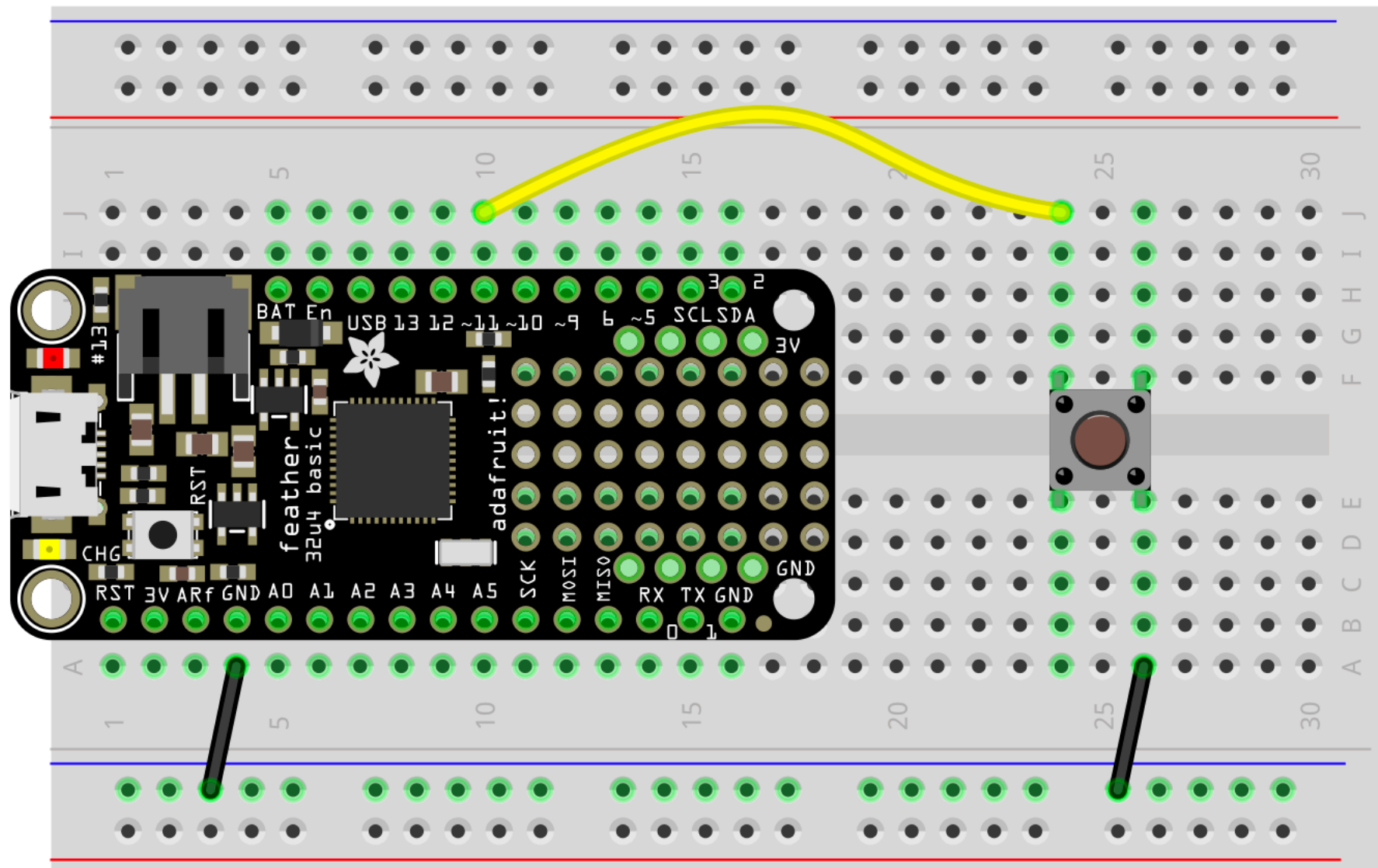
The image shows a screenshot of the Arduino IDE interface. The title bar at the top reads "sketch_feb25a | Arduino 1.0.5". The main text area contains the following C++ code:

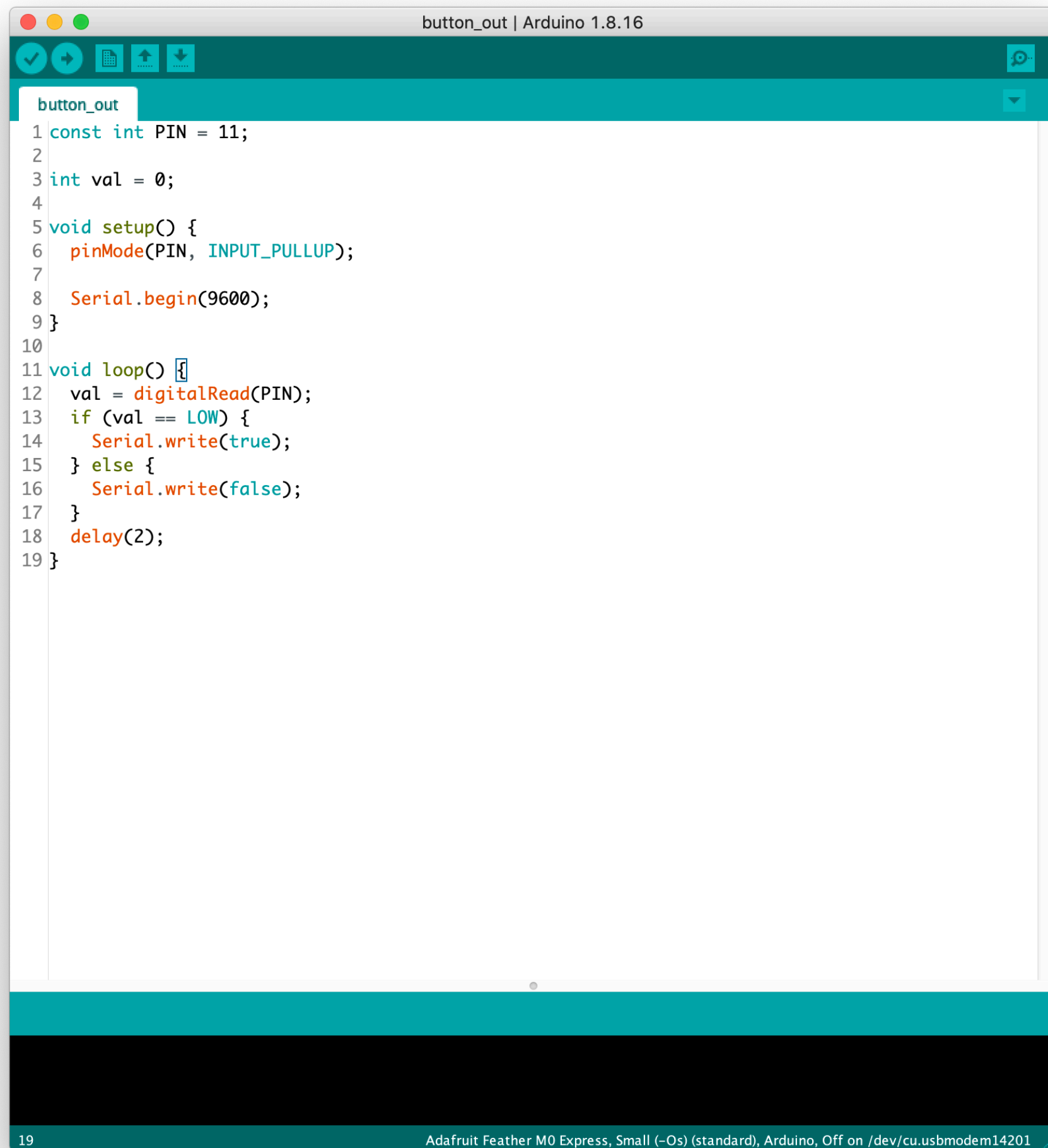
```
void setup() {  
  Serial.begin(9600);  
}  
  
void loop() {  
  Serial.println("Hello, world!");  
  delay(100);  
}
```

The code is written in a monospaced font with syntax highlighting: keywords like `void` and `loop` are in orange, `Serial` is in blue, and string literals like `"Hello, world!"` are in blue. The IDE has a teal header bar with icons for checking, compiling, uploading, and saving. At the bottom, there is a black console area and a status bar that says "9" on the left and "Arduino Uno on /dev/tty.usbserial-A601LO5X" on the right.

PRINT MESSAGE VIA SERIAL

BUTTON INPUT

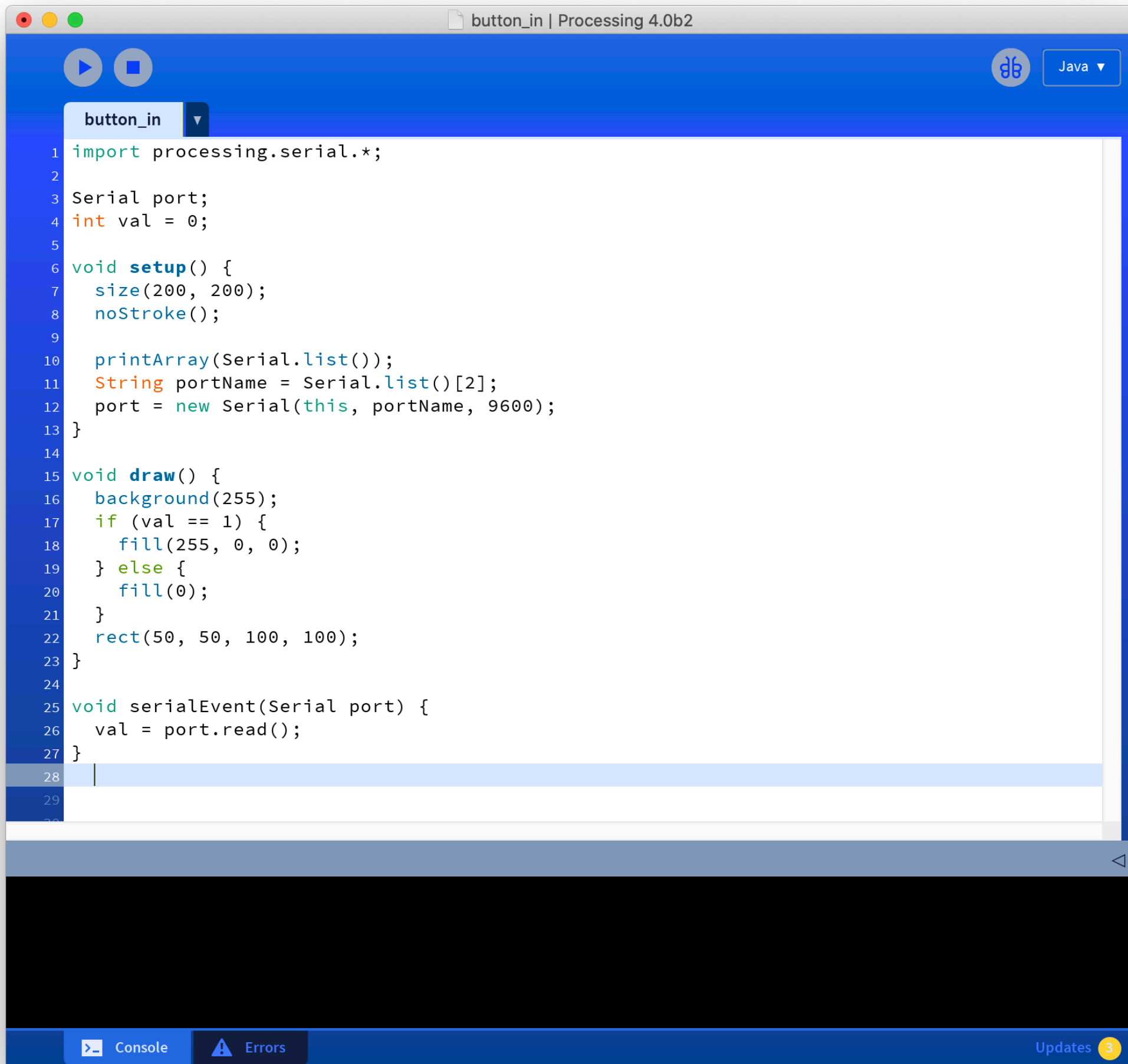


A screenshot of the Arduino IDE interface. The window title is 'button_out | Arduino 1.8.16'. The code editor shows a sketch named 'button_out' with the following code:

```
1 const int PIN = 11;
2
3 int val = 0;
4
5 void setup() {
6   pinMode(PIN, INPUT_PULLUP);
7
8   Serial.begin(9600);
9 }
10
11 void loop() {
12   val = digitalRead(PIN);
13   if (val == LOW) {
14     Serial.write(true);
15   } else {
16     Serial.write(false);
17   }
18   delay(2);
19 }
```

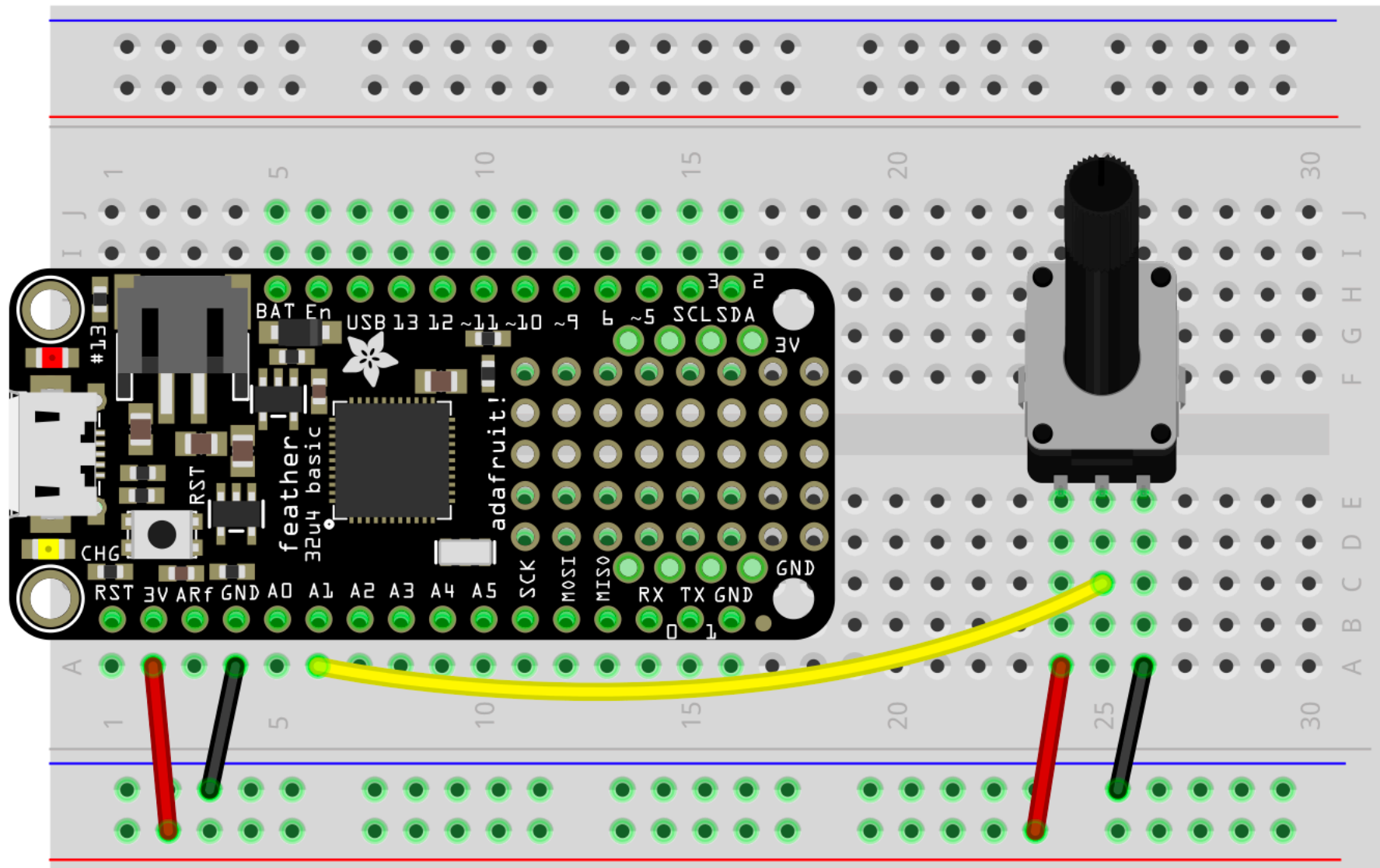
The bottom status bar shows '19' on the left and 'Adafruit Feather M0 Express, Small (-Os) (standard), Arduino, Off on /dev/cu.usbmodem14201' on the right.

WRITE VALUES VIA SERIAL WITH ARDUINO

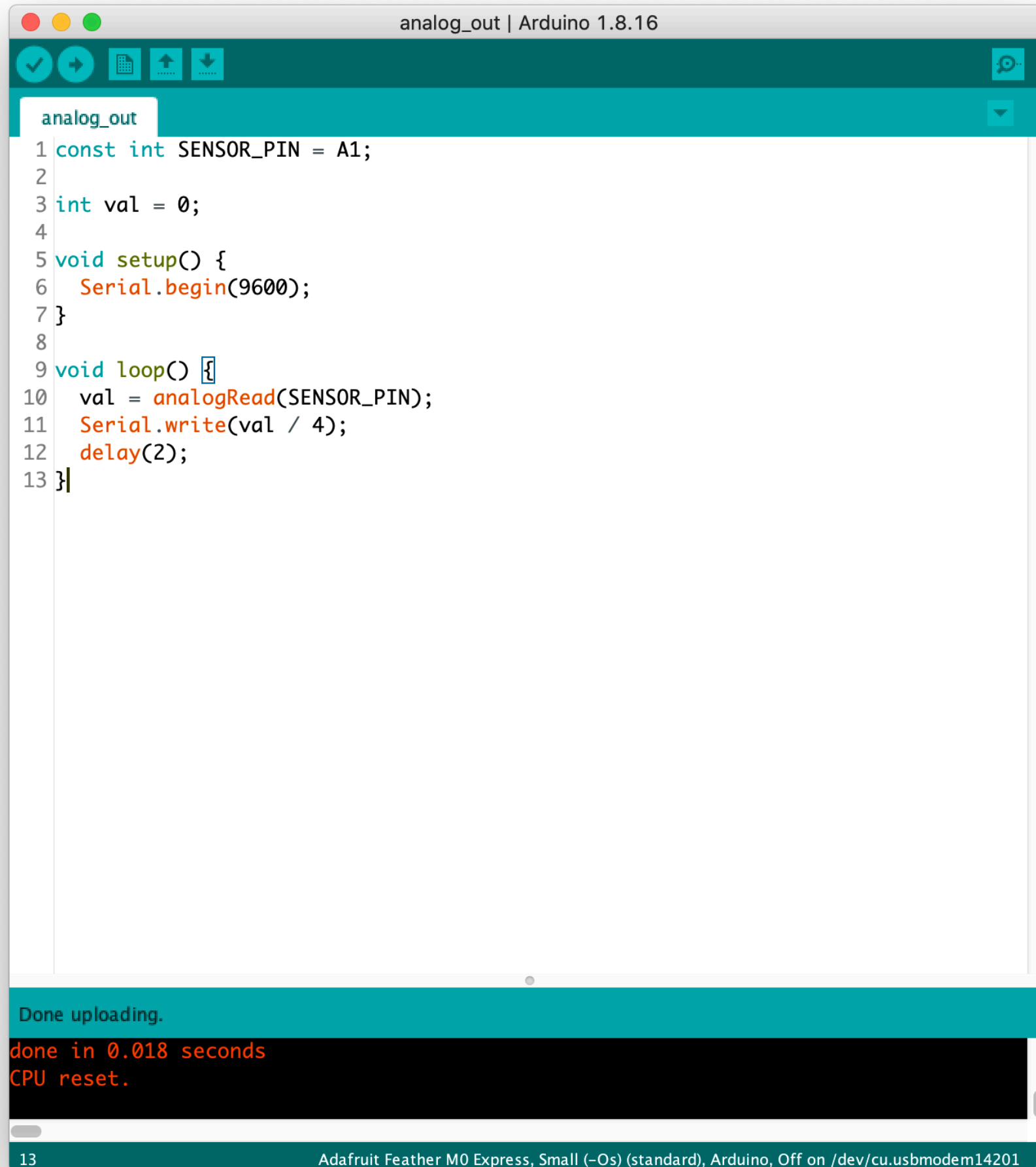


READ VALUES VIA SERIAL WITH PROCESSING

POTENTIOMETER (OR OTHER ANALOG SENSOR) INPUT



WRITE VALUES VIA SERIAL WITH ARDUINO



The screenshot shows the Arduino IDE interface. The title bar reads 'analog_out | Arduino 1.8.16'. The menu bar includes 'File', 'Edit', 'Tools', and 'Help'. The toolbar contains icons for checking, running, uploading, and downloading. The main editor area shows the following code:

```
1 const int SENSOR_PIN = A1;
2
3 int val = 0;
4
5 void setup() {
6   Serial.begin(9600);
7 }
8
9 void loop() {
10  val = analogRead(SENSOR_PIN);
11  Serial.write(val / 4);
12  delay(2);
13 }
```

Below the editor is a status bar showing 'Done uploading.' and 'done in 0.018 seconds CPU reset.' The bottom status bar indicates the board is 'Adafruit Feather M0 Express, Small (-Os) (standard), Arduino, Off on /dev/cu.usbmodem14201'.

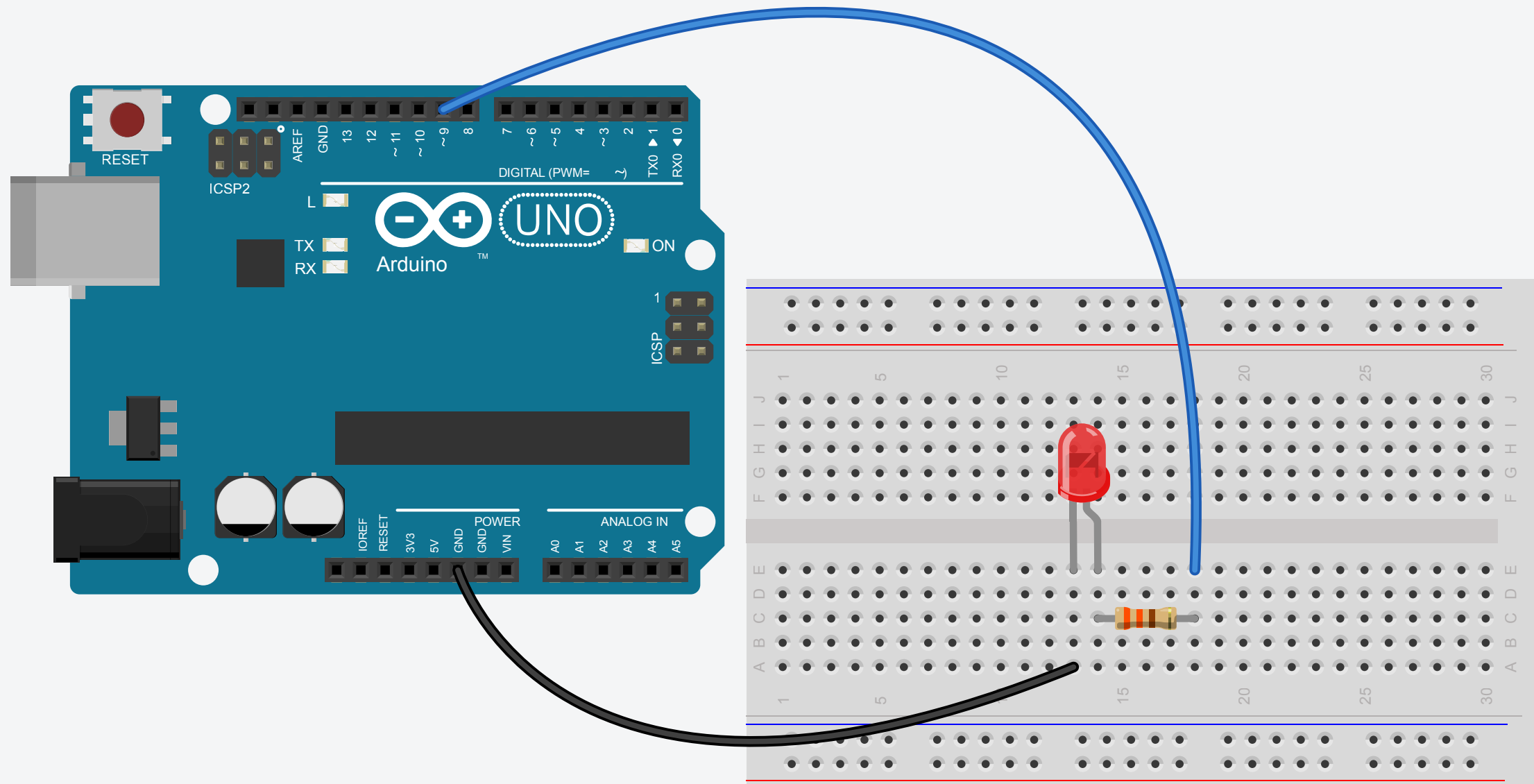

```
1 import processing.serial.*;
2
3 Serial port;
4 int val = 0;
5
6 void setup() {
7   size(200, 200);
8   noStroke();
9
10  printArray(Serial.list());
11  String portName = Serial.list()[2];
12  port = new Serial(this, portName, 9600);
13 }
14
15 void draw() {
16   background(val);
17 }
18
19 void serialEvent(Serial port) {
20   val = port.read();
21   println(val);
22 }
23
24
25
26
```

172
172
172
171
171
172

READ VALUES VIA SERIAL WITH PROCESSING

PROCESSING -> ARDUINO

PWM LED OUTPUT



processing_serial_out | Processing 3.3.6

processing_serial_out

```
1 import processing.serial.*;
2
3 Serial port;
4
5
6 void setup() {
7   size(256, 200);
8   printArray(Serial.list());
9   String portName = Serial.list()[2];
10  port = new Serial(this, portName, 9600);
11 }
12
13 void draw() {
14   for (int i=0; i<256; i++) {
15     stroke(i);
16     line(i, 0, i, height);
17   }
18   port.write(mouseX);
19 }
20
21
22
23
24
25
26
27
28
29
```

Done saving.

```
[1] "/dev/cu.SLAB_USBtoUART"
[2] "/dev/cu.SLAB_USBtoUART"
[3] "/dev/cu.SoundCoremini-SerialPor"
[4] "/dev/tty.Bluetooth-Incoming-Port"
[5] "/dev/tty.JBLXtreme-SPPDev"
[6] "/dev/tty.SLAB_USBtoUART"
[7] "/dev/tty.SoundCoremini-SerialPor"
```

Console Errors Updates 2

WRITE VALUES VIA SERIAL WITH PROCESSING



```
analog_in | Arduino 1.8.5
analog_in
1 const int LED = 11;
2
3 void setup() {
4   Serial.begin(9600);
5   pinMode(LED, OUTPUT);
6 }
7
8 void loop() {
9   byte input;
10
11   if (Serial.available()) {
12     input = Serial.read();
13     analogWrite(LED, input);
14   }
15 }
```

Done uploading.

Sketch uses 1902 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 184 bytes (8%) of dynamic memory, leaving 1864 bytes for

15 Arduino/Genuino Uno on /dev/cu.SLAB_USBtoUART

READ VALUES VIA SERIAL WITH ARDUINO