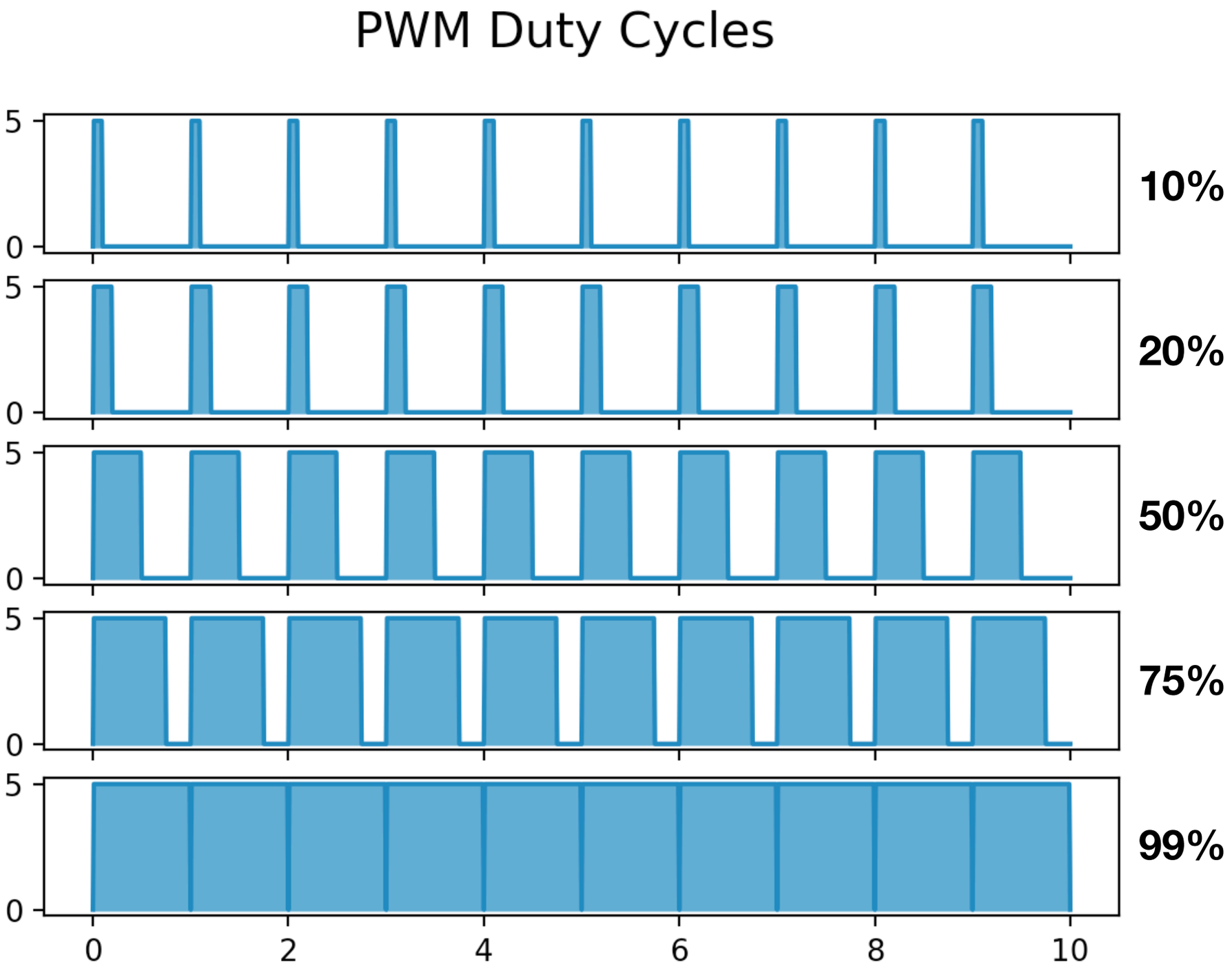
# Arduino 3

Thomas Schwarz

# Pulse Width Modulation

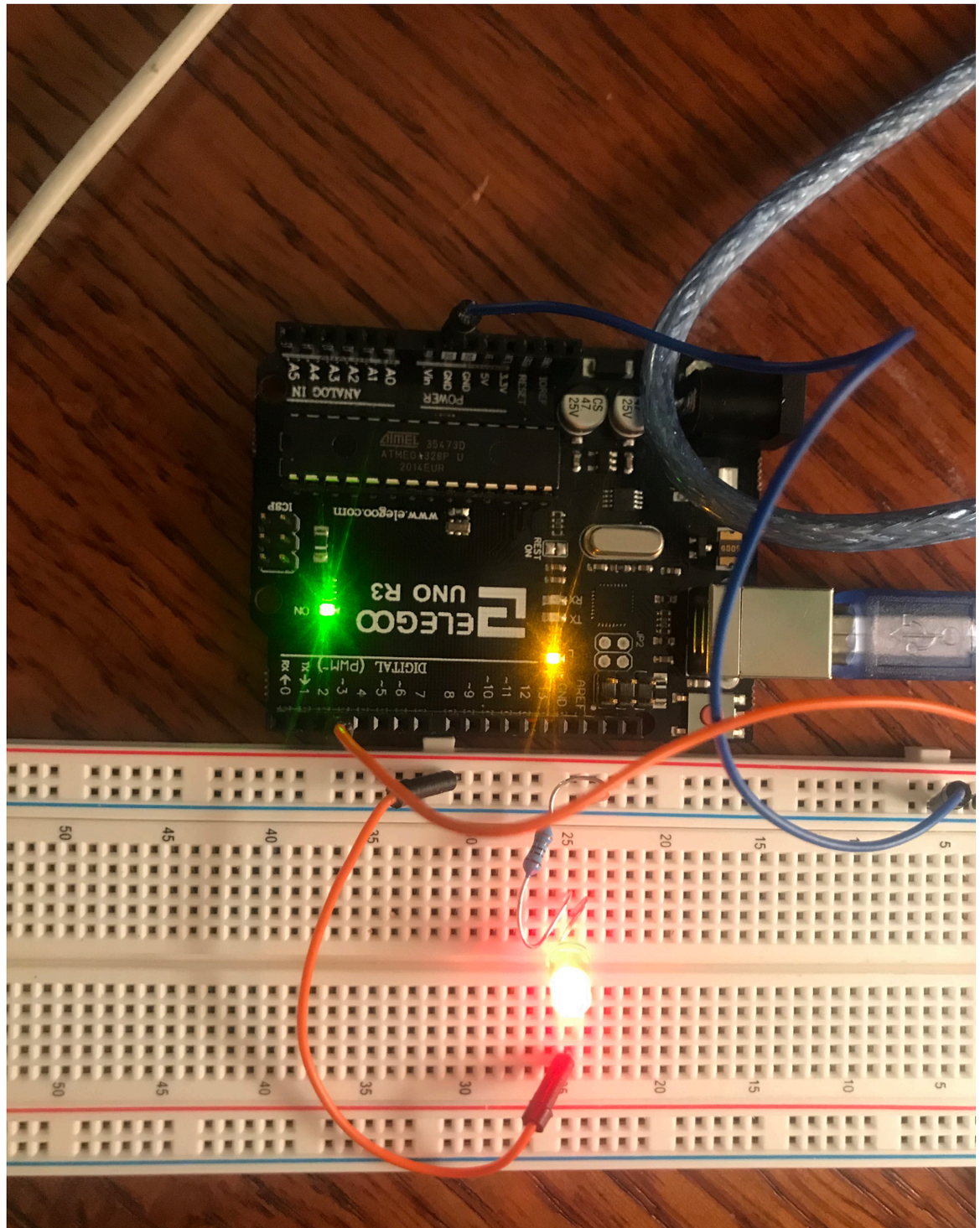
- We can use Voltage or Resistance to control the output of a LED
- With an Arduino: Use Pulse Width Modulation
  - Output oscillates between full voltage and no voltage
  - At 50 cycles per second, so we cannot see it

# Pulse Width Modulation



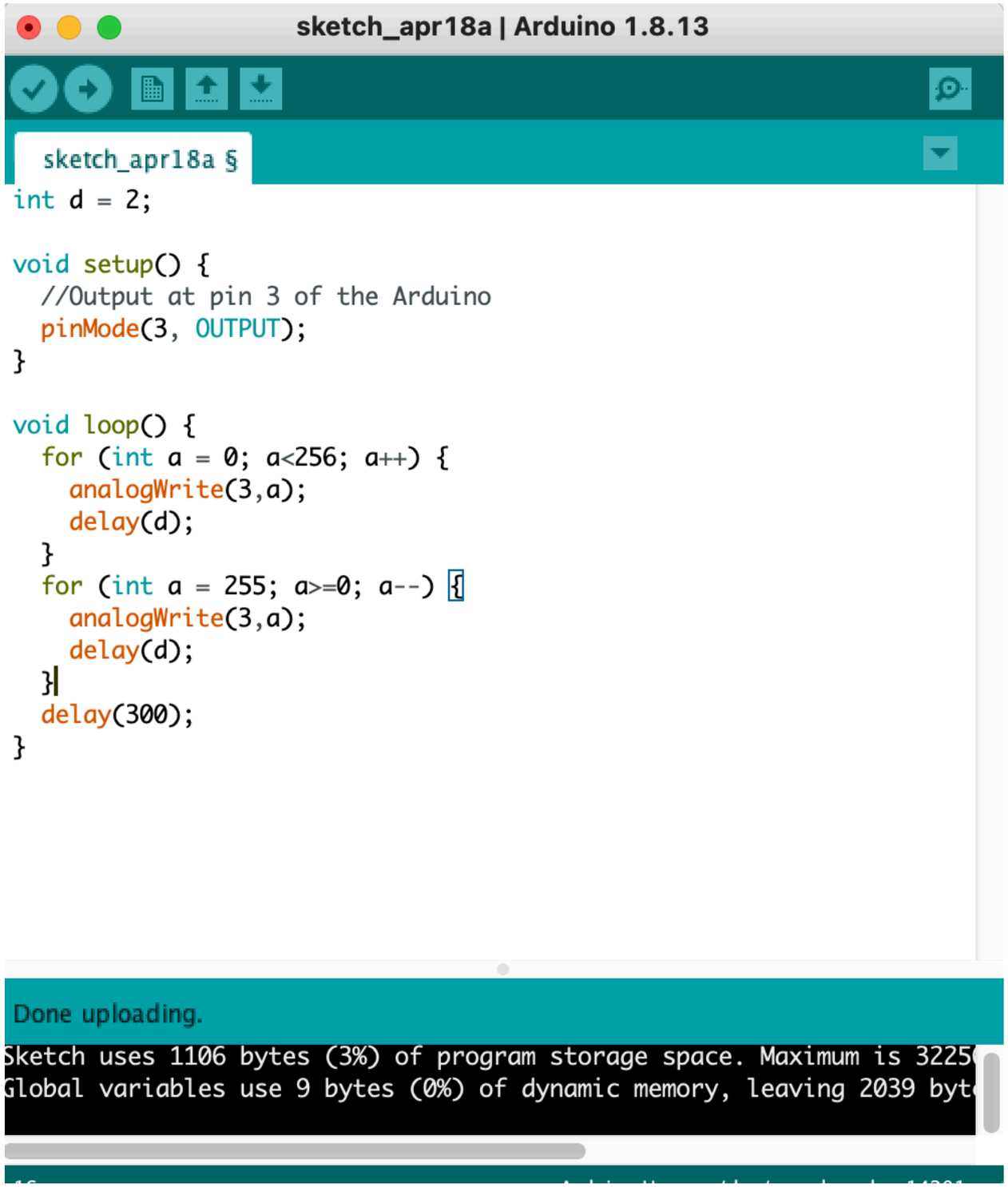
# Pulse Width Modulation

- Any Arduino port with a tilde ~ can create PWM output
- We use pin 3.
- Connect Diode Anode to Pin 3
- Connect Diode Kathode to 300 Ohm resistor
- Connect resistor to GND



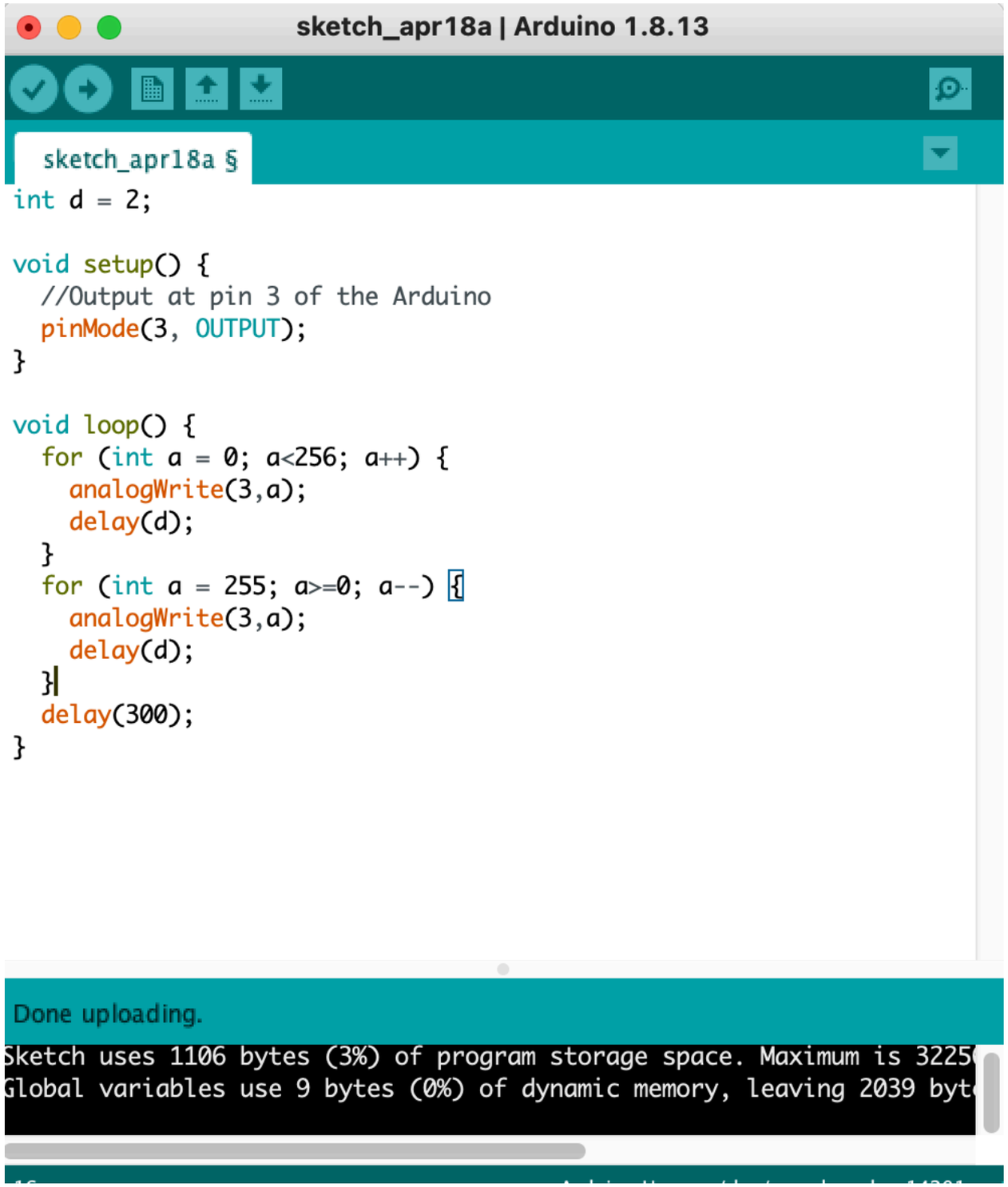
# Pulse Width Modulation

- C variables have a type
- We set a global variable d to 2 (d for delay)
- In setup, we set pin 3 as Output
- In loop, we have a C-style loop
  - declares int a,
  - while a smaller than 256 = 2**8:
  - increment a
  - and then use analogWrite to set a PWM at pin 3



# Pulse Width Modulation

- We can change d to set the speed of the LED "breathing"

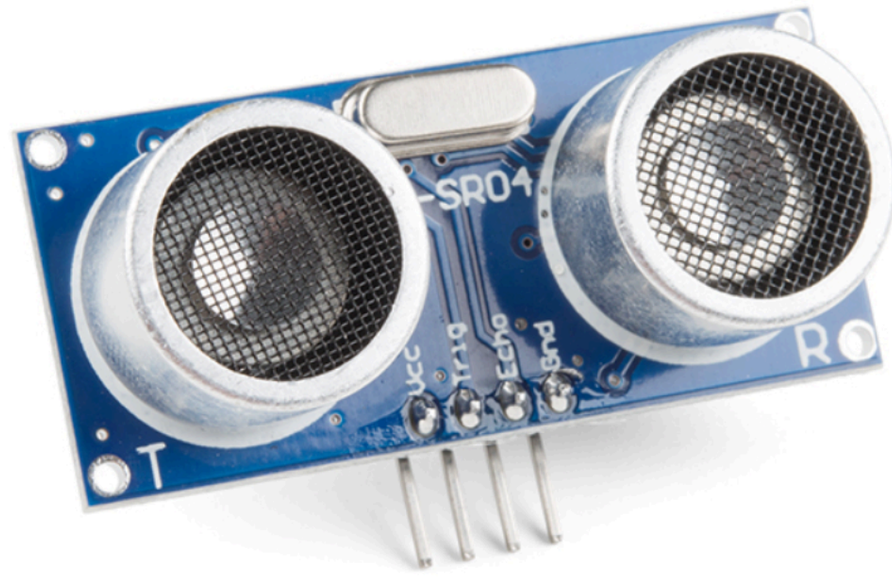


# Libraries and Inputs

- Some components need Libraries
  - A collection of definitions and functions similar to a Python module
  - Typical libraries are compressed in a .zip format
  - To install a library, go to Arduino IDE, Sketch, and then install a zip library.

# Libraries and Inputs

- Ultrasonic Sensor:
  - A low costs device that uses ultra-sound to measure distances with an accuracy of less than a centimeter
  - Needs the HCSR04-1.1.0.zip library which you can downloa from Elegoo or from arduinolibraries.info

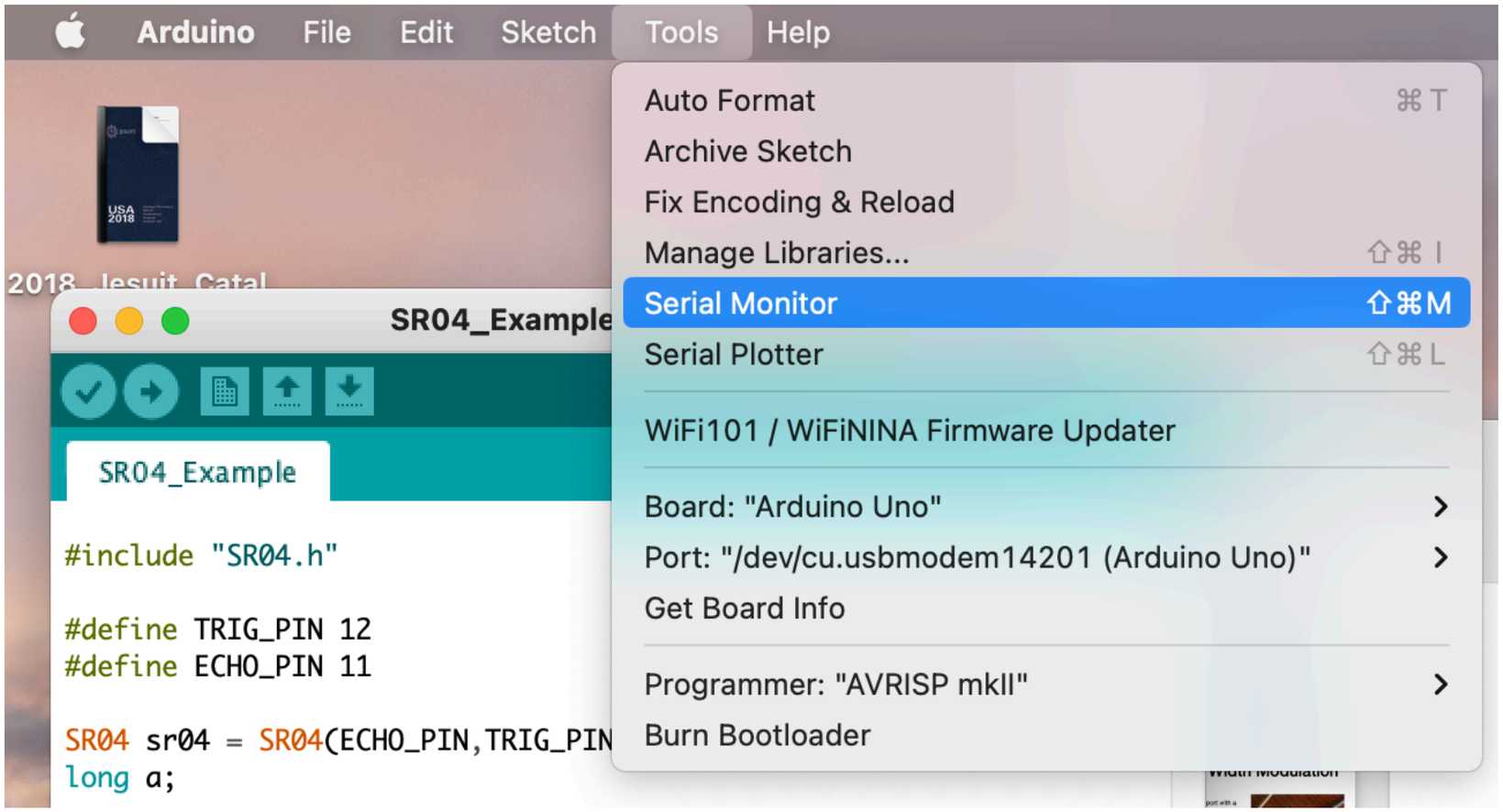


# Libraries and Inputs

- First step:
  - Download and install the library
- Second step: Connect the distance sensor to your Arduino
  - Find MF wires
- Connect ground to ground and VCC to 5V
- Connect Trig to pin 12
- Connect Echo to pin 11

# Libraries and Inputs

- Third step:
  - Open the Serial Monitor under Tools



# Libraries and Inputs

- Fourth Step: Write the sketch

- We use `#include SR04.h` to make use of the library

- We define the pins:
  - `#define TRIG_PIN 12`
  - `#define ECHO_PIN 11`

```
#include "SR04.h"

#define TRIG_PIN 12
#define ECHO_PIN 11
```

# Libraries and Inputs

- We also create a SR04 device object and a long integer a
  - Long means 64bits, which is what the SR04 returns as its distance measure

```
SR04 sr04 = SR04(ECHO_PIN,TRIG_PIN);
long a;
```

# Libraries and Inputs

- Our setup function needs to start talking to the serial port
- The 9600 is the BAUD rate, do not change that.

```
void setup() {
    Serial.begin(9600);
    delay(1000);
}
```

# Libraries and Inputs

- The loop measures the distance every second
- The first line gets the long int distance
- The second line prints it to the Serial monitor
- print just prints,
- println prints and creates a new line
- Then we wait for a second

```
'
void loop() {
    a=sr04.Distance();
    Serial.println(a);
    delay(1000);
}
```

# Libraries and Inputs

```
#include "SR04.h"

#define TRIG_PIN 12
#define ECHO_PIN 11

SR04 sr04 = SR04(ECHO_PIN,TRIG_PIN);
long a;

void setup() {
   Serial.begin(9600);//Initialization of Serial Port
   delay(1000);
}

void loop() {
   a=sr04.Distance();
   Serial.println(a);
   delay(1000);
}
```

# Libraries and Inputs

- On the serial monitor, you can now see what the Arduino returns