

Getting started with Arduino

Prof Wouter Buytaert
Imperial College London

Arduino: a microcontroller ecosystem

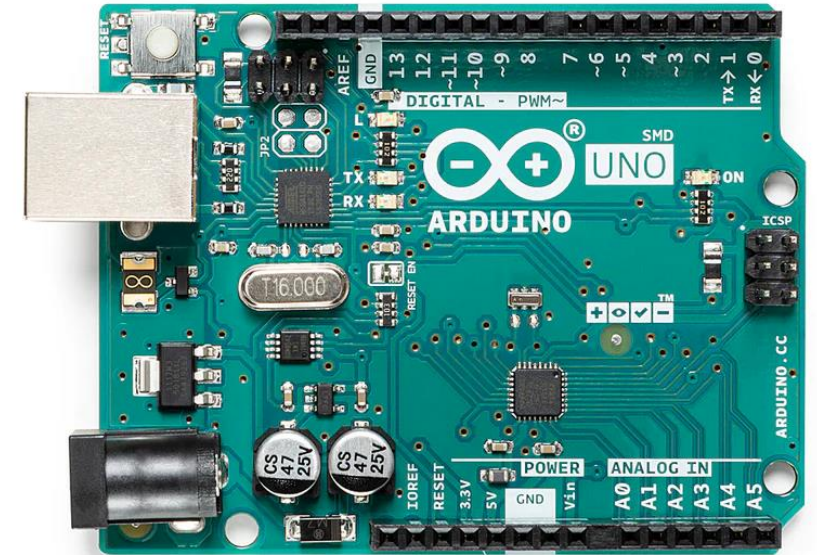
*Arduino is an **open-source hardware** and software company, project, and user community that designs and manufactures single-board **microcontrollers** and microcontroller kits for building digital devices.* *
(Wikipedia)

Microcontrollers:

- Mini computer
- Everywhere: digital thermometers, microwave ovens, ...
- Specialized for specific purposes

Arduino:

- Democratizing technology
- Easy to use, affordable
- For different people with different levels of experience
- Maker movement



Source: store.arduino.cc

* <https://en.wikipedia.org/wiki/Arduino>

History of Arduino Uno



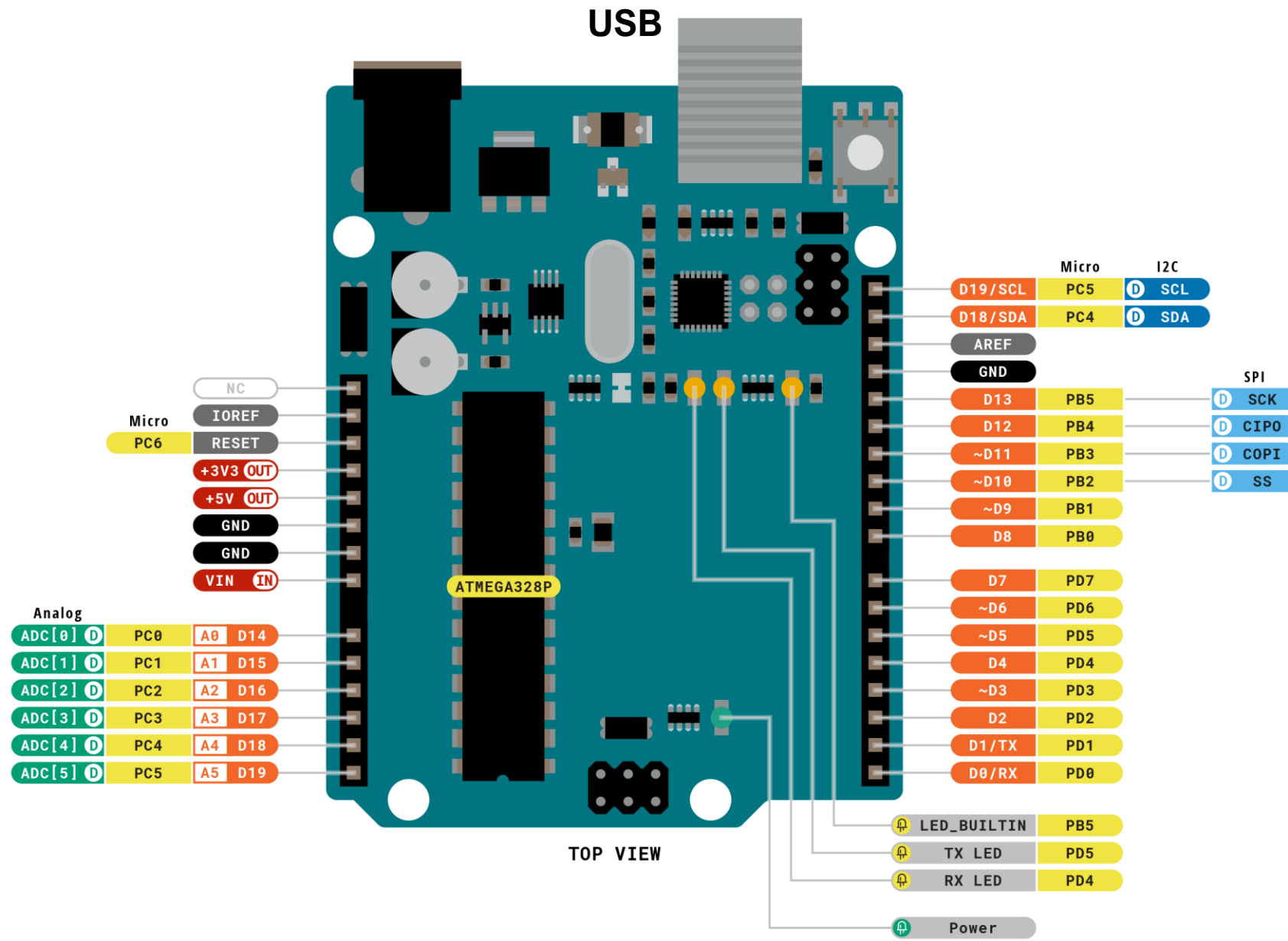
Arduino Uno WiFi REV2
Source: store.arduino.cc

Source: "One board to rule them all: History of the Arduino UNO" (blog.arduino.cc)

Arduino Uno

Legend:

■ Digital	■ I2C
■ Power	■ SPI
■ Ground	■ Analog
■ Main Part	

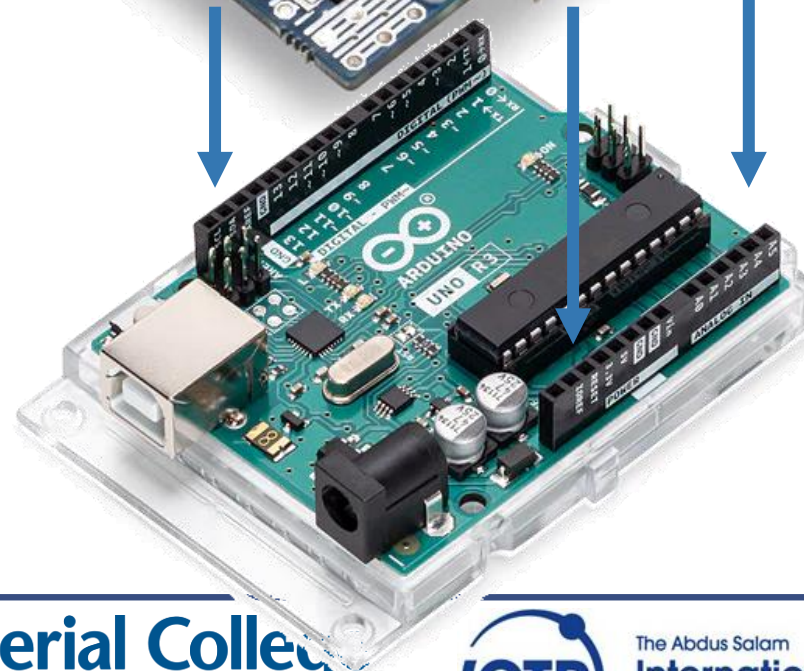
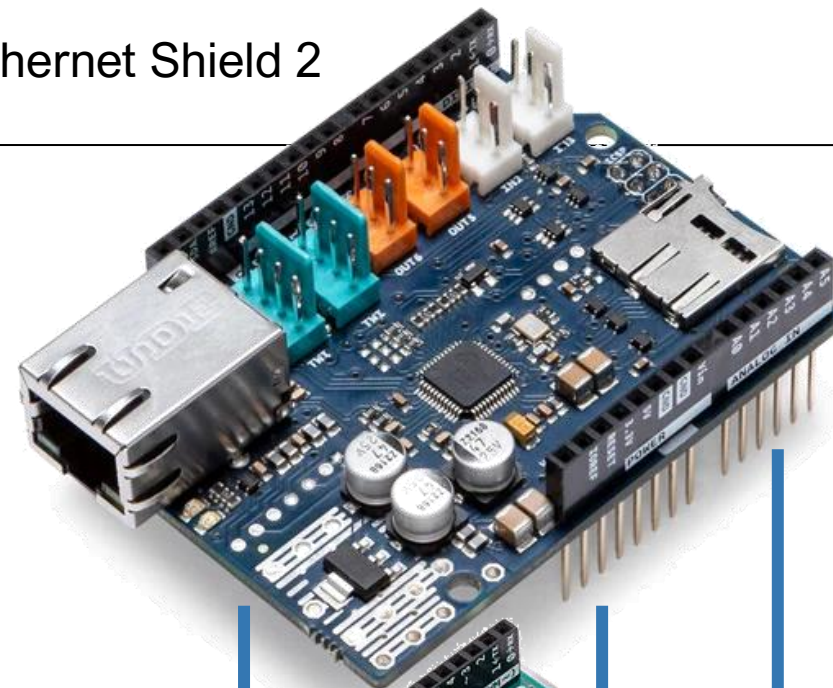
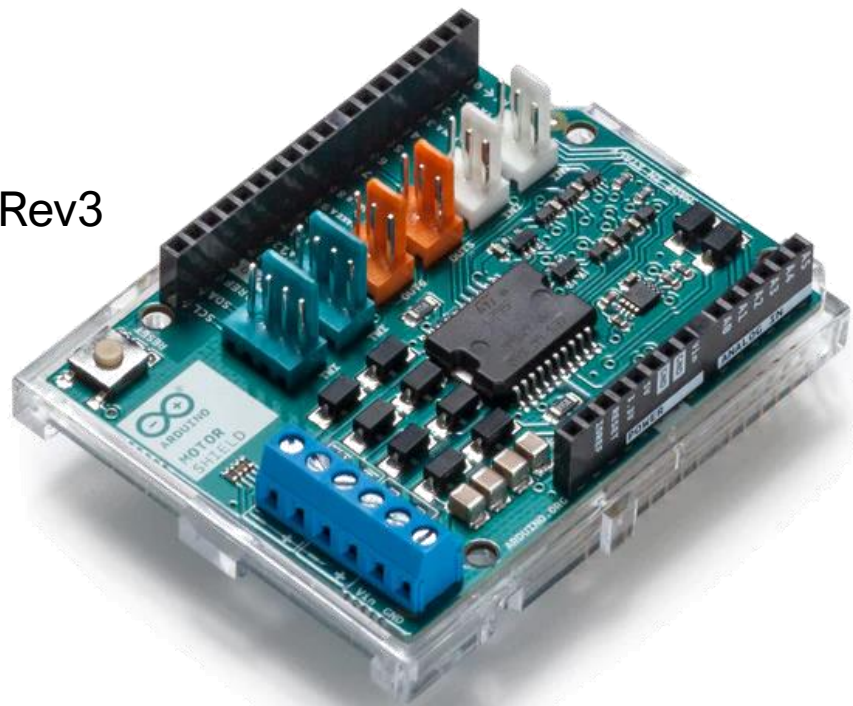


Source: store.arduino.cc (Arduino U REV3)

Arduino Shields

Ethernet Shield 2

Motor Shield Rev3



Source: store.arduino.cc

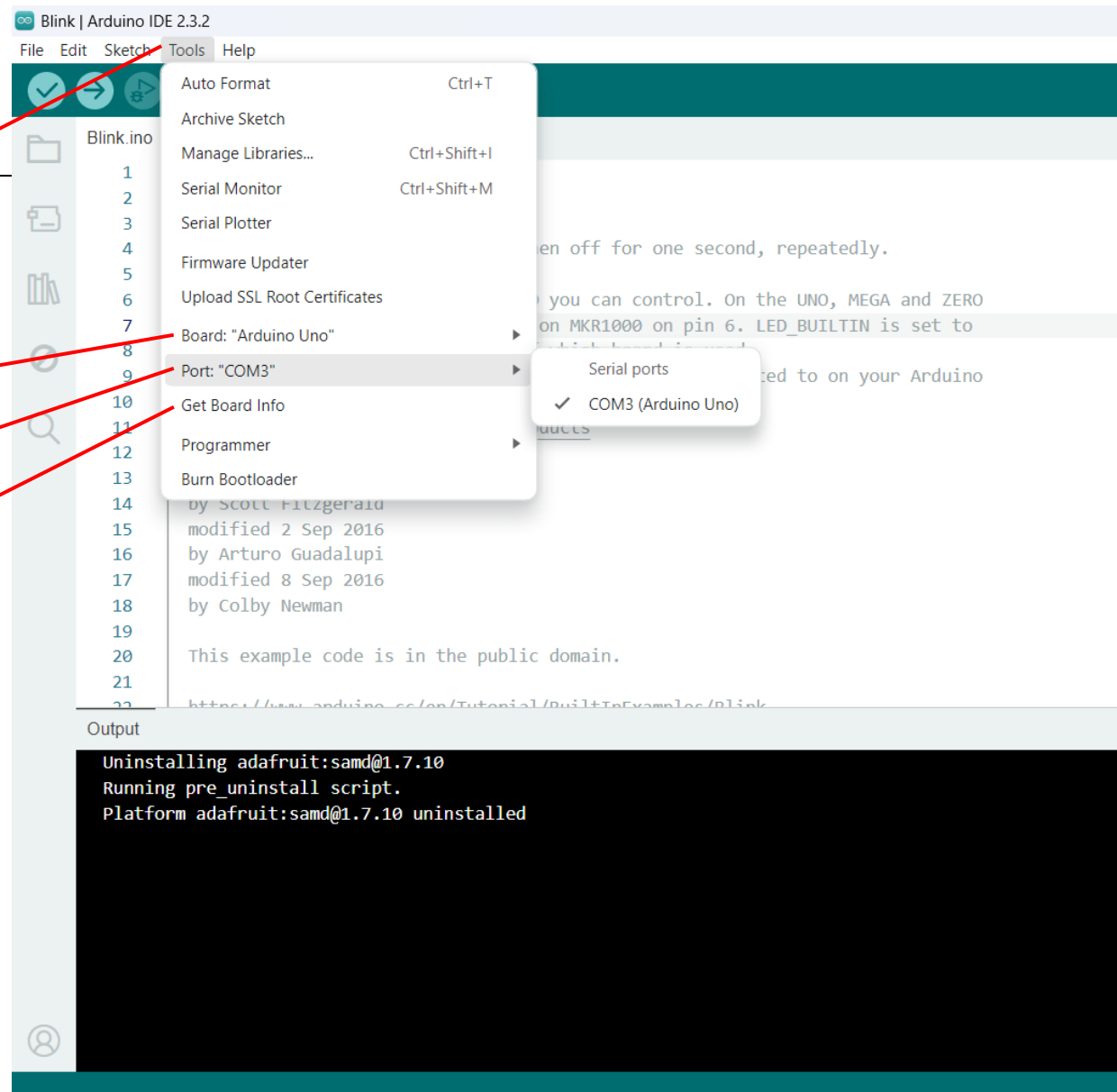
Arduino IDE

Tools menu

Board selector

Port selector

Get board info



Blink | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Arduino Uno

Blink.ino

```
4 Turns an LED on for one second, then off for one second, repeatedly.
5
6 Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
7 it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
8 the correct LED pin independent of which board is used.
9 If you want to know what pin the on-board LED is connected to on your Arduino
10 model, check the Technical Specs of your board at:
11 https://www.arduino.cc/en/Main/Products
12
13 modified 8 May 2014
14 by Scott Fitzgerald
15 modified 2 Sep 2016
16 by Arturo Guadalupi
17 modified 8 Sep 2016
18 by Colby Newman
19
20 This example code is
21
22 https://www.arduino.cc/en/tutorial/BUILTINexamples/BLINK
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
```

Board Info

Board Info

BN: Arduino Uno
VID: 0x2341
PID: 0x0043
SN: 55736323939351217021

OK

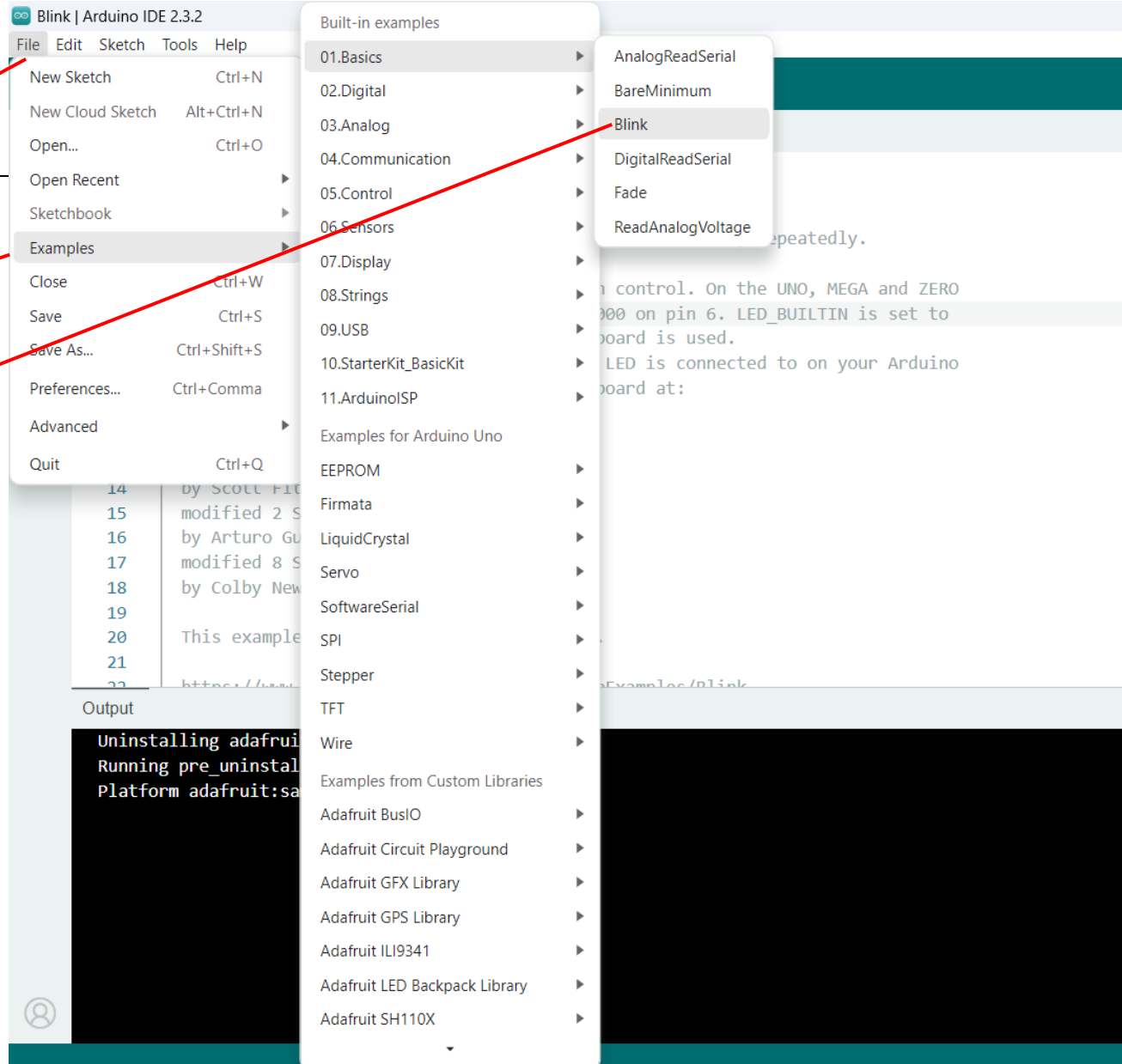
Serial Monitor Output

Arduino IDE

File menu

Examples

Blink example



The blink example

```
void setup() {  
    pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(1000);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(1000);  
}
```

Programming microprocessors: concepts

- Embedded systems

Embedded systems are computational devices built for a specific purpose (as opposed to general-purpose computers). They are typically smaller, less powerful, and more energy efficient. A wide range of processor types exist for embedded systems.

- Embedded programming

Embedded systems are often programmed for a specific task. Compiled languages such as C and C++ are often used because they are fast and versatile, and memory efficient. Often, embedded processors only support a limited subset of instructions.

- Programming an Arduino board

The original Arduino code is based on the AVR C programming language. However, it can be extended with C and C++ libraries. Other programming languages such as MicroPython and CircuitPython are increasingly popular, especially on boards with more memory.

Programming an Arduino

- The basic structure of an Arduino “sketch”

```
void setup() {  
  // put your setup code here, to run once:  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

- Further resources
 - <https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics>
 - Arduino software -> file -> examples
 - <https://learn.adafruit.com>

The blink example

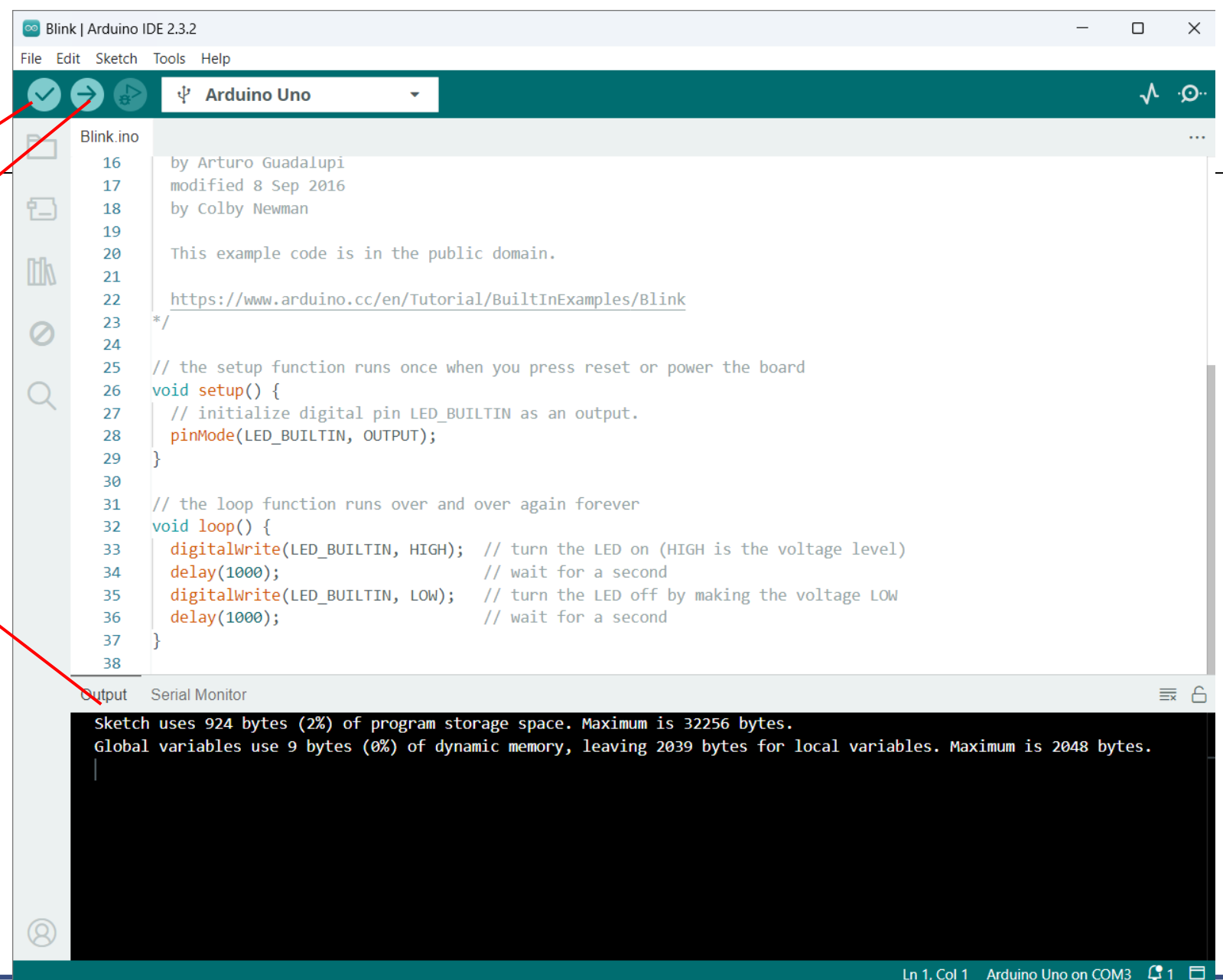
```
void setup() {  
    pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(1000);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(1000);  
}
```

Arduino IDE

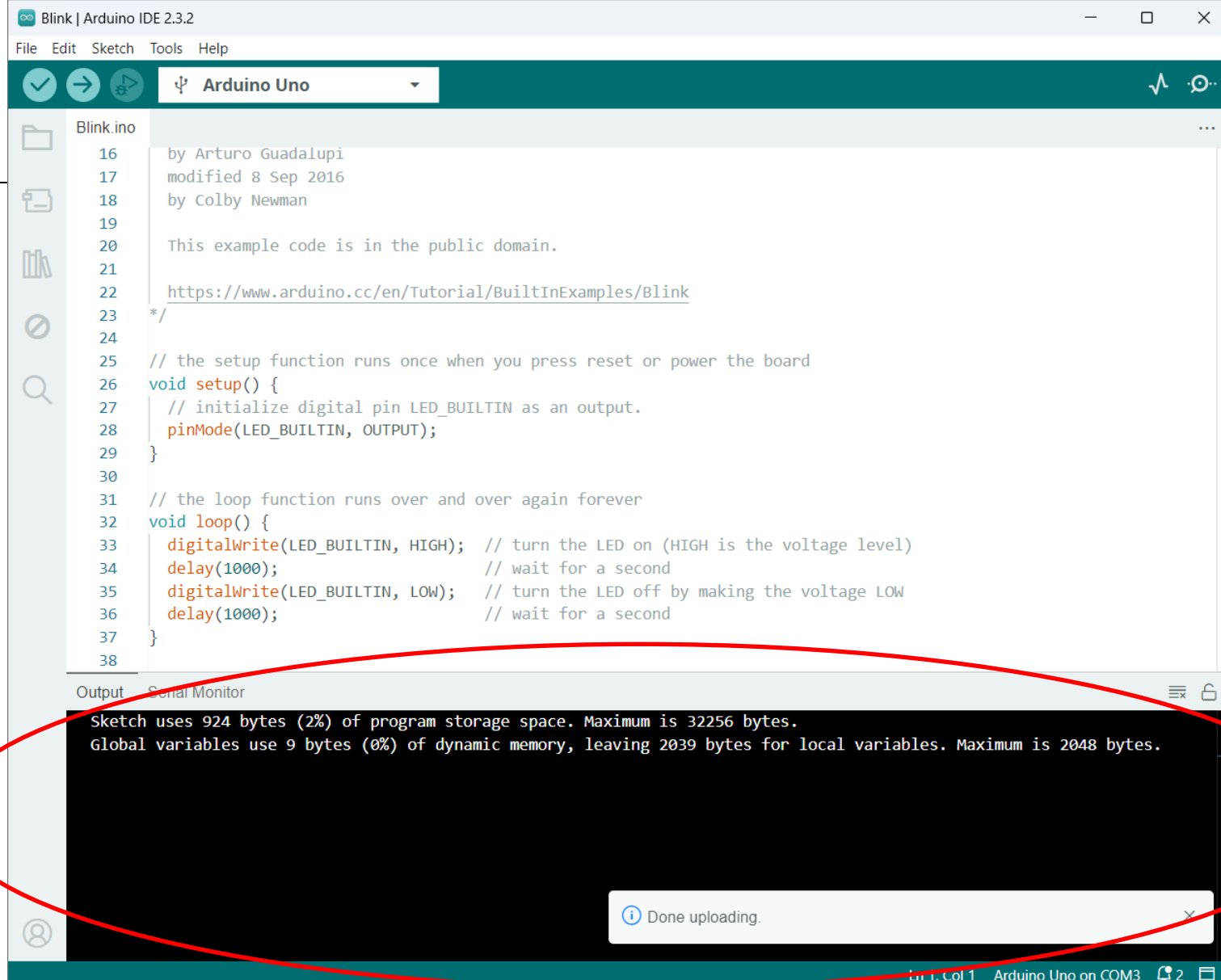
Compile button

Upload button

Output window



Arduino IDE



Blink | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Arduino Uno

```
16 by Arturo Guadalupi
17 modified 8 Sep 2016
18 by Colby Newman
19
20 This example code is in the public domain.
21
22 https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000); // wait for a second
35   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36   delay(1000); // wait for a second
37 }
38
```

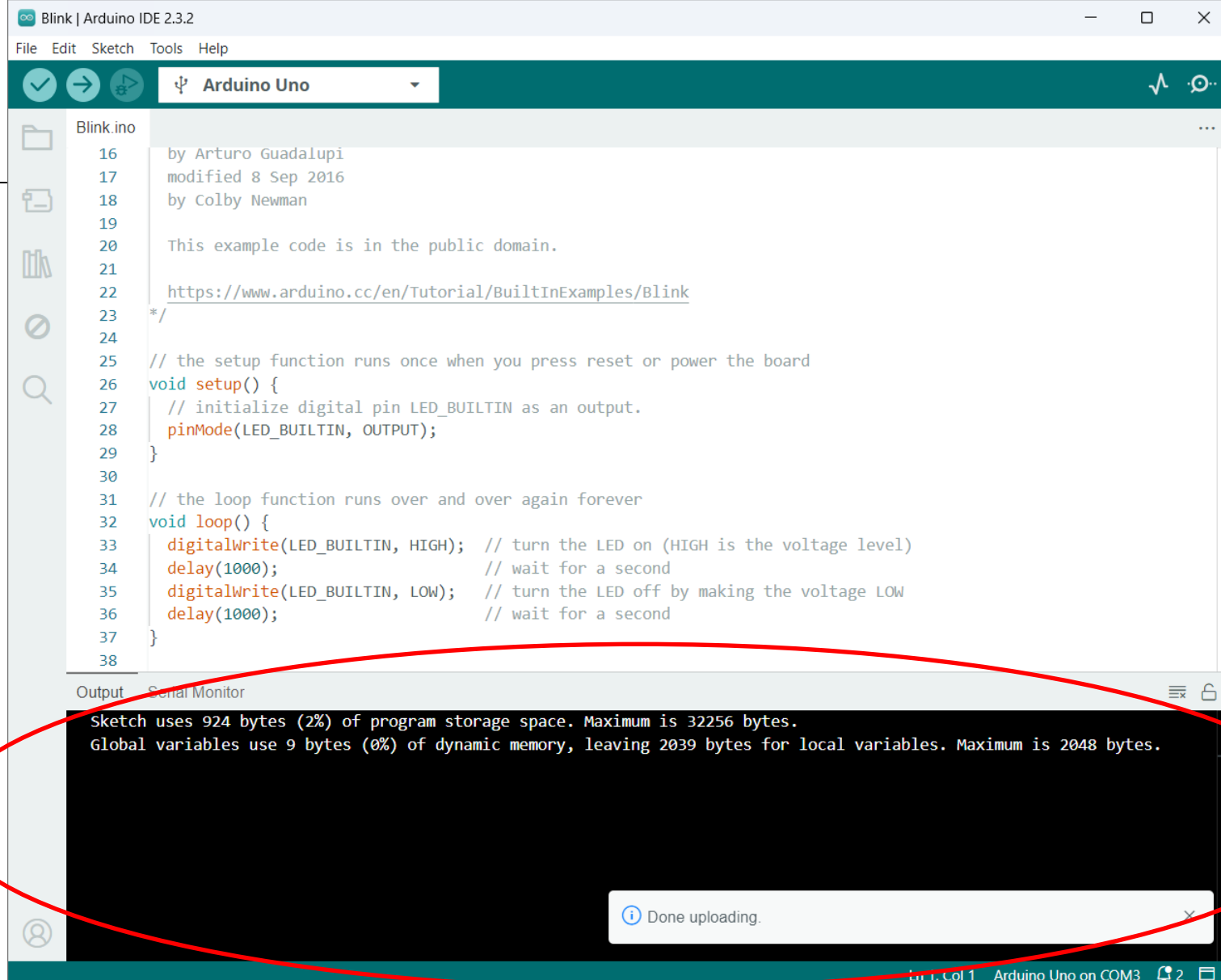
Output Serial Monitor

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

Done uploading.

Ln 1, Col 1 Arduino Uno on COM3

Arduino IDE



Blink | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Arduino Uno

```
16 by Arturo Guadalupi
17 modified 8 Sep 2016
18 by Colby Newman
19
20 This example code is in the public domain.
21
22 https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000); // wait for a second
35   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36   delay(1000); // wait for a second
37 }
38
```

Output Serial Monitor

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

Done uploading.

Ln 1, Col 1 Arduino Uno on COM3

Exercise 1:

Modify the “blink” script such that the LED on the board blinks 5 times per second.

Advanced exercise:

Modify the blink script, such that the LED blinks first long (1s) and then repeatedly short (100ms). Test the function of the reset button

Serial communication

Arduino has various digital communication interfaces. These are useful to communicate with a variety of peripheral devices, from single sensors to other Arduinos.

It can also communicate to the PC it is connected to via the USB cable. This is useful to get direct feedback, for example for debugging.

The “serial monitor” interface allows us to communicate with the Arduino, in two directions: we can ask the Arduino to send communication to the PC, and we can also send communication from the PC to the Arduino.

Arduino IDE

The screenshot displays the Arduino IDE interface. The main editor window shows the code for 'ASCIITable.ino'. The code includes a multi-line comment describing the program's purpose: printing byte values in various formats (raw binary, ASCII-decimal, hex, octal, and binary). It also includes a circuit note (no external hardware needed), author information (created 2006 by Nicholas Zambetti, modified 9 Apr 2012 by Tom Igoe), and a public domain disclaimer. The code ends with a URL to the Arduino tutorial page.

```
1  /*
2  ASCII table
3
4  Prints out byte values in all possible formats:
5  - as raw binary values
6  - as ASCII-encoded decimal, hex, octal, and binary values
7
8  For more on ASCII, see http://www.asciitable.com and http://en.wikipedia.org/wiki/ASCII
9
10 The circuit: No external hardware needed.
11
12 created 2006
13 by Nicholas Zambetti <http://www.zambetti.com>
14 modified 9 Apr 2012
15 by Tom Igoe
16
17 This example code is in the public domain.
18
19 https://www.arduino.cc/en/Tutorial/BuiltInExamples/ASCIITable
20 */
```

The Serial Monitor window at the bottom shows the output of the program, which is a character map for ASCII characters from 0 to 127. Each line shows the character followed by its decimal, hexadecimal, octal, and binary representations. A red oval highlights the output text.

```
Message (Enter to send message to 'Arduino Uno' on 'COM3') New Line 9600 baud
15:40:43.913 -> ASCII Table ~ Character Map
15:40:43.926 -> !, dec: 33, hex: 21, oct: 41, bin: 100001
15:40:43.934 -> ", dec: 34, hex: 22, oct: 42, bin: 100010
15:40:43.956 -> #, dec: 35, hex: 23, oct: 43, bin: 100011
15:40:43.988 -> $, dec: 36, hex: 24, oct: 44, bin: 100100
15:40:44.053 -> %, dec: 37, hex: 25, oct: 45, bin: 100101
15:40:44.087 -> &, dec: 38, hex: 26, oct: 46, bin: 100110
15:40:44.152 -> ', dec: 39, hex: 27, oct: 47, bin: 100111
15:40:44.185 -> (, dec: 40, hex: 28, oct: 50, bin: 101000
15:40:44.217 -> ), dec: 41, hex: 29, oct: 51, bin: 101001
15:40:44.250 -> *, dec: 42, hex: 2A, oct: 52, bin: 101010
15:40:44.316 -> +, dec: 43, hex: 2B, oct: 53, bin: 101011
```

Exercise 2:

Explore the following sketches available in the examples:

- communications -> ASCIItable.ino
- strings -> CharacterAnalysis.ino

Note: If you open the serial monitor after uploading your sketch, and you do not see any output, then you may need to push the reset button to restart the sketch

Data storage

The Arduino UNO has a small amount of storage in its microprocessor. Most of this storage is used for the sketch and can only be written when the sketch is uploaded.

However, part of this storage can be used by the sketch (read, write). This can be accessed using as EEPROM memory, using the EEPROM library.

This memory is persistent, i.e. the contents will be saved even if the Arduino is disconnected from power.

Similar storage capability can be achieved by adding a separate EEPROM chip to your electronic design (as we will see later).

Exercise 3:

Explore EEPROM storage by means of the following scripts:

- EEPROM -> eeprom.get
- EEPROM -> eeprom.put

Advanced exercise:

Store the names of your group members in the Arduino UNO's EEPROM, swap the UNO with another team, and retrieve the other team's names

```

1  /*
2  * EEPROM Read
3  *
4  * Reads the value of each byte of the EEPROM and prints it
5  * to the computer.
6  * This example code is in the public domain.
7  */
8
9  #include <EEPROM.h>
10
11 // start reading from the first byte (address 0) of the EEPROM
12 int address = 0;
13 byte value;
14
15 void setup() {
16   // initialize serial and wait for port to open:
17   Serial.begin(9600);
18   while (!Serial) {
19     ; // wait for serial port to connect. Needed for native USB port only
20   }
21 }

```

Message (Enter to send message to 'Arduino Uno' on 'COM3') New Line 9600 baud

```

16:14:26.312 -> 515    255
16:14:26.838 -> 516    255
16:14:27.339 -> 517    255
16:14:27.815 -> 518    255
16:14:30.109 -> 0      121
16:14:30.601 -> 1      233
16:14:31.091 -> 2      246
16:14:31.604 -> 3      66
16:14:32.083 -> 4      195
16:14:32.583 -> 5      245

```

```

20 float field1;
21 byte field2;
22 char name;
23 };
24
25 void setup() {
26   Serial.begin(9600);
27   extern HardwareSerial Serial;
28   while (!Serial) {
29     ; // wait for serial port to connect. Needed for native USB port only
30   }
31
32   float f = 123.456f; //Variable to store in EEPROM.
33   int eeAddress = 0; //Location we want the data to be put.
34
35
36   //One simple call, with the address first and the object second.
37   EEPROM.put(eeAddress, f);
38
39   Serial.println("Written float data type!");
40
41   /** put is designed for use with custom structures also. */

```

Message (Enter to send message to 'Arduino Uno' on 'COM3') New Line 9600 baud

```

16:04:17.241 -> Written float data type!
16:04:17.248 -> Written custom data type!
16:04:17.267 ->
16:04:17.267 -> View the example sketch eeprom_get to see how you can retrieve the values!

```

Exercise 4 (Advanced):

Write your own sketch in which the Arduino executes the following steps:

- Blink the LED 1 second on startup
- Loop forever through the following actions:
 - write an integer to the EEPROM
 - sleep for 1 minute