ARDUINO
OPEN-SOURCE
COMMUNITY

Arduino

Assist. Prof. Nadica Miljković, PhD
University of Belgrade – School of Electrical Engineering
e-mail: nadica.miljkovic@etf.bg.ac.rs

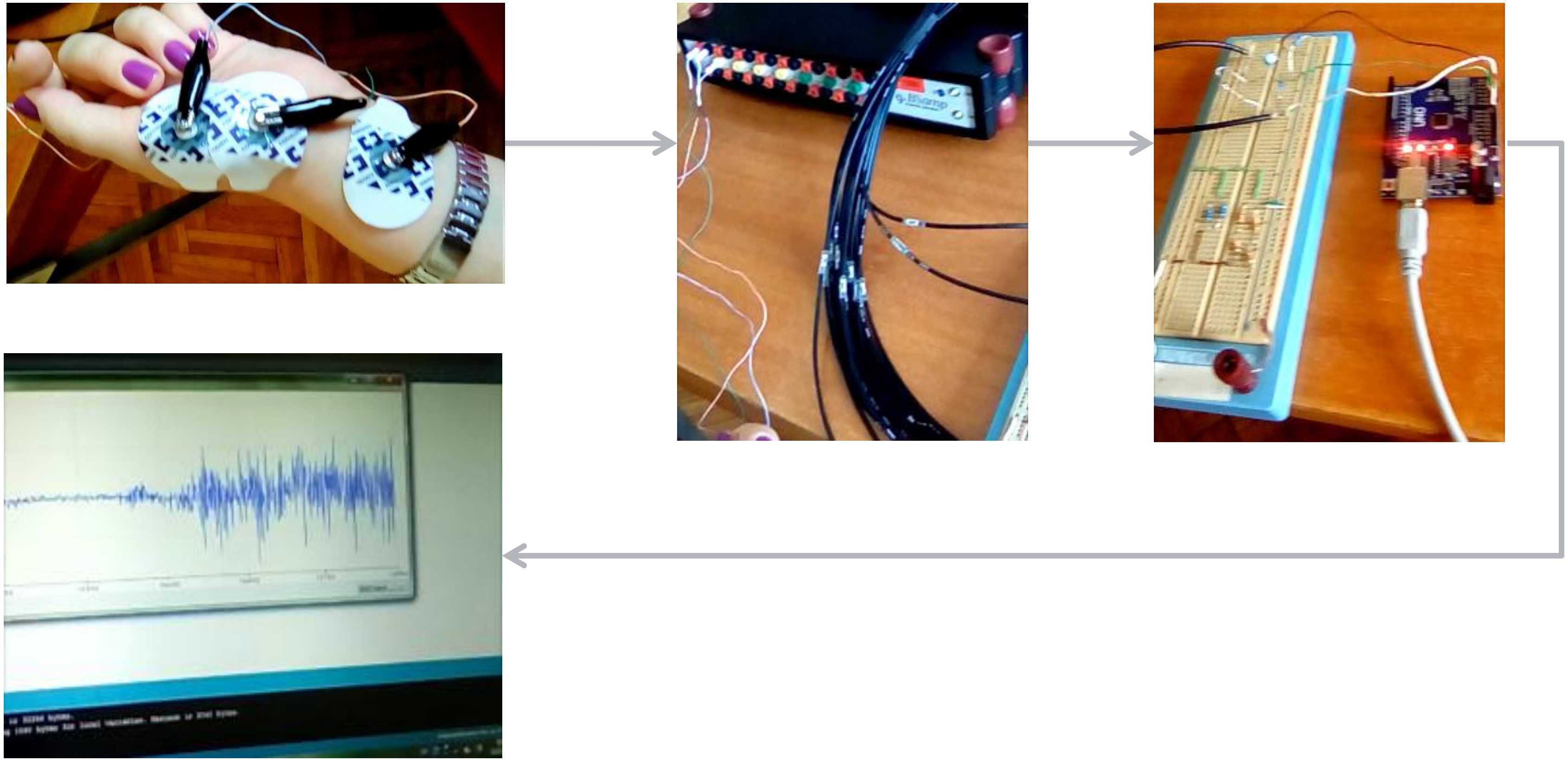
For this presentation following materials were used: Alan G. Smith, Introduction to Arduino: A piece of cake!, [Online], Assessed December 8, 2018, https://www.introtoarduino.com/downloads/IntroArduinoBook.pdf, 2011.

Also, materials from Measurement Computing Devices are used: http://automatika.etf.bg.ac.rs/sr/13e053msr.

I would recommend Technology Tutorials available at: http://www.toptechboy.com/, Assessed Dec., 8, 2018.

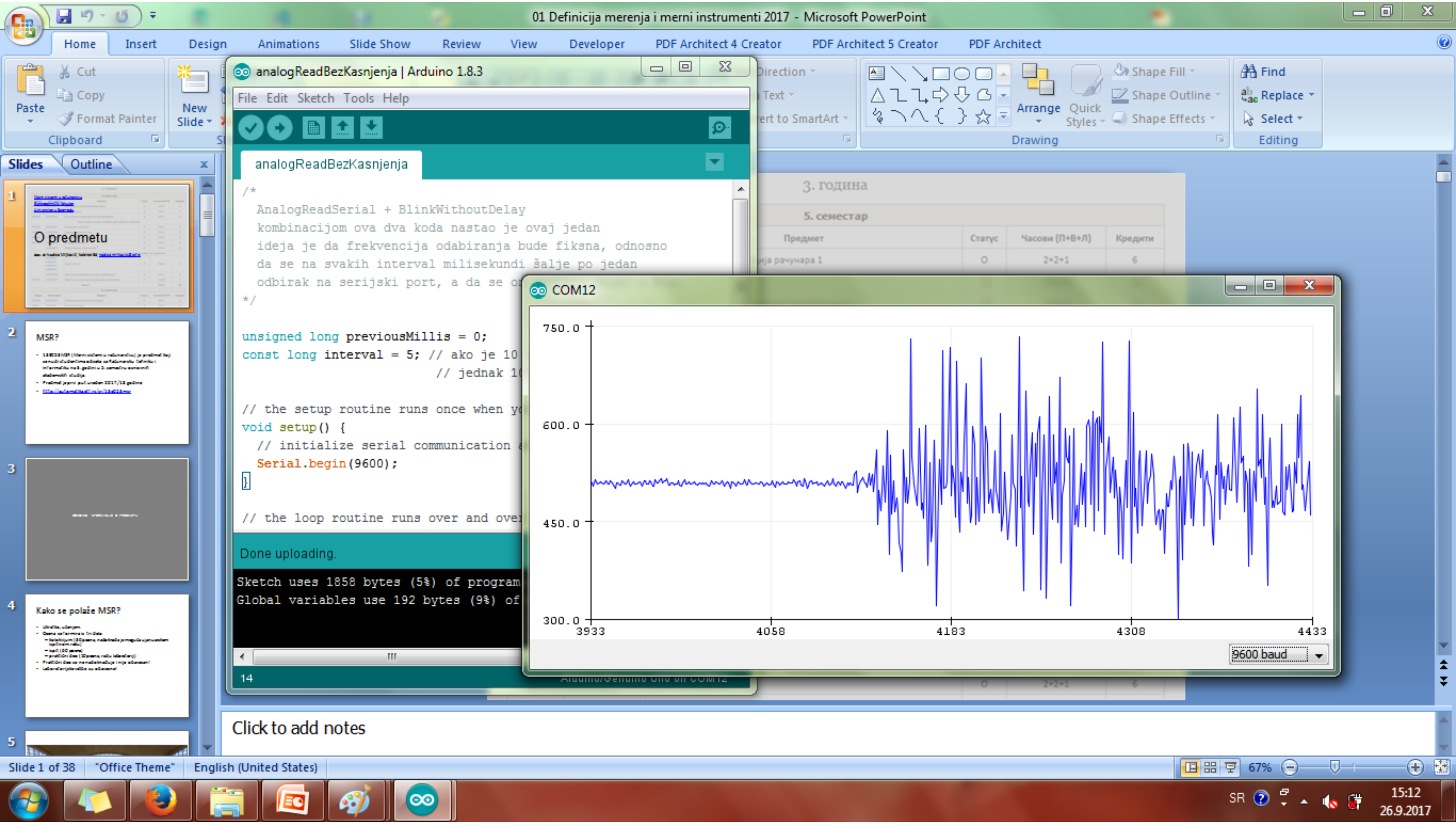
HOW CAN YOU BENEFIT FROM USING ARDUINO?
LIVE DEMOS

Muscle contraction measurement



- Electrophysiological amplifier (g.tec, Austria), conditioning circuit, and UNO R3 microcontroller board are connected to the Ag/AgCl electrodes for measurement of electrical muscle activity of *abductor policis brevis* (lat.).
- Simple and efficient measurement with UNO R3 hardware and Arduino software.

Closer look to EMG signal



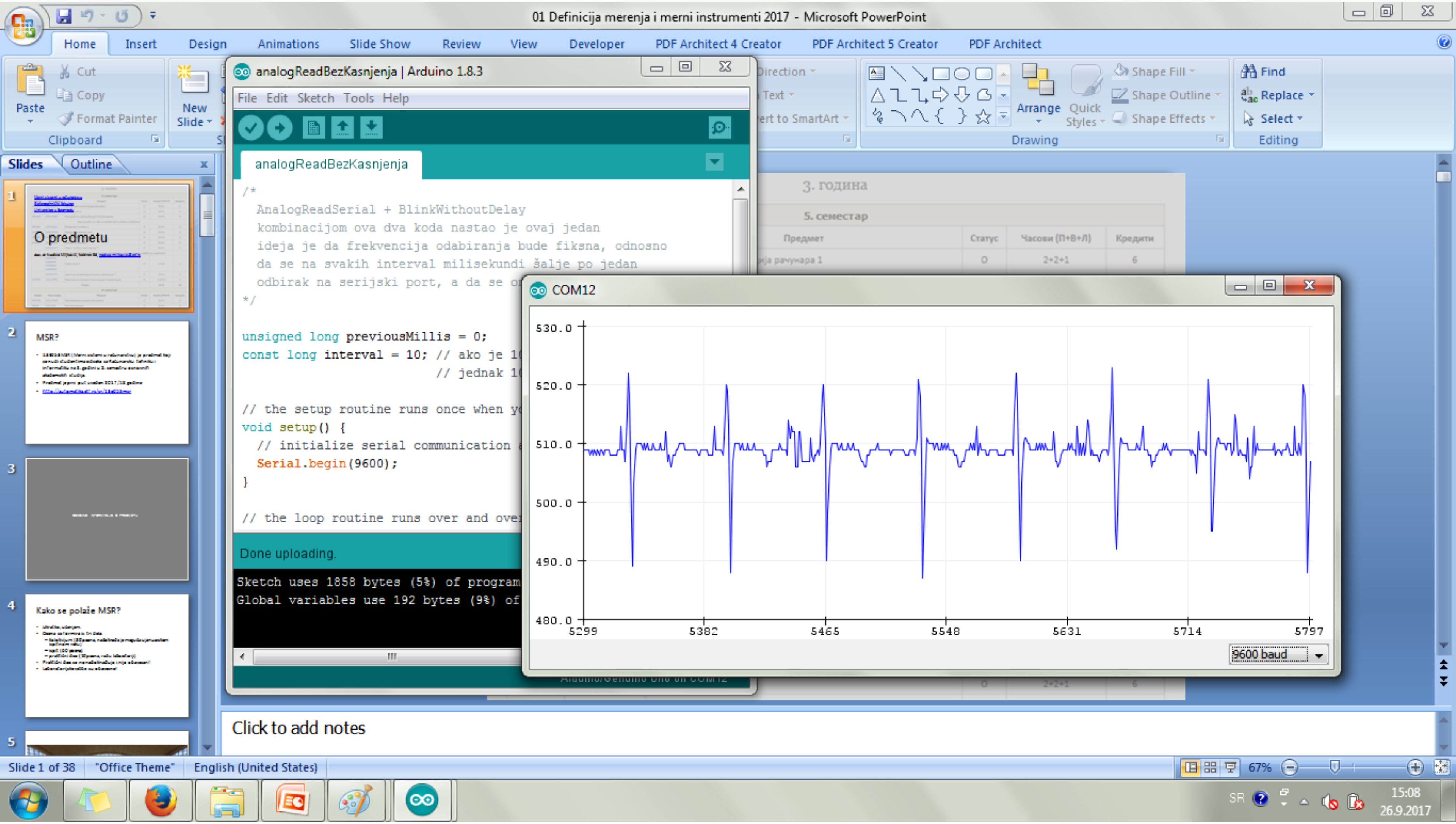
```
/*
  AnalogReadSerial + BlinkWithoutDelay
  kombinacijom ova dva koda nastao je ovaj jedan
  ideja je da frekvencija odabiranja bude fiksna, odnosno
  da se na svakih interval milisekundi šalje po jedan
  odbirak na serijski port, a da se o
*/

unsigned long previousMillis = 0;
const long interval = 5; // ako je 10
                         // jednak 1

// the setup routine runs once when y
void setup() {
  // initialize serial communication a
  Serial.begin(9600);
}

// the loop routine runs over and ove
```

Done uploading.

```
Sketch uses 1858 bytes (5%) of program
Global variables use 192 bytes (9%) of
```

14

ECG measurements



```
/*
  AnalogReadSerial + BlinkWithoutDelay
  kombinacijom ova dva koda nastao je ovaj jedan
  ideja je da frekvencija odabiranja bude fiksna, odnosno
  da se na svakih interval milisekundi šalje po jedan
  odbirak na serijski port, a da se o
*/

unsigned long previousMillis = 0;
const long interval = 10; // ako je 1
                          // jednak 1

// the setup routine runs once when y
void setup() {
  // initialize serial communication 
  Serial.begin(9600);
}

// the loop routine runs over and ove
```

Done uploading.

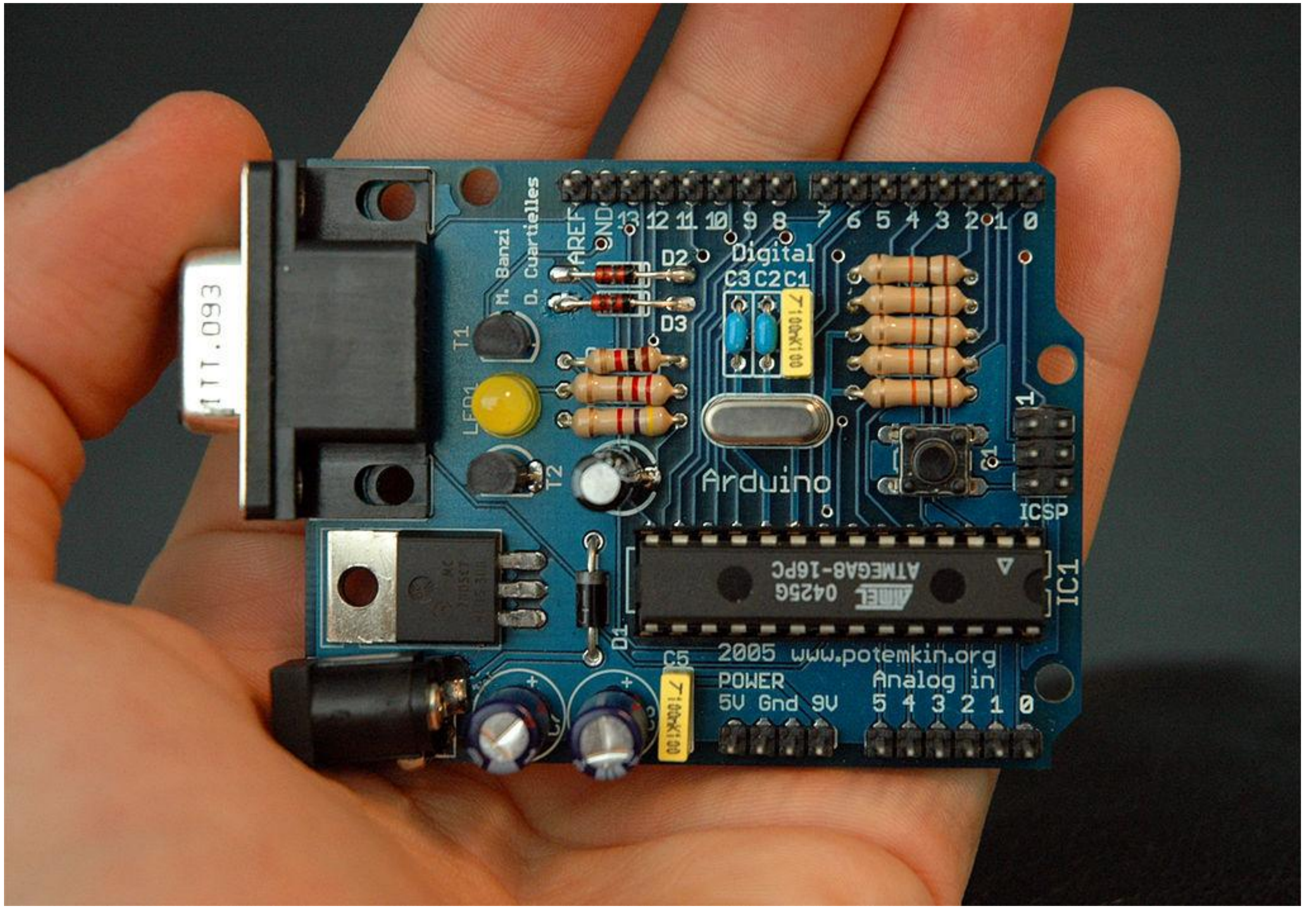
```
Sketch uses 1858 bytes (5%) of program
Global variables use 192 bytes (9%) of
```

Measurement Computing Devices

- Course on Measurement Computing Devices MCD was introduced in 2017. More at: http://automatika.etf.bg.ac.rs/sr/13e053msr.
- Video of selected solutions from MCD Challenge held in 2017 and available at: https://www.youtube.com/watch?v=C-opCv-skYM&feature=youtu.be.

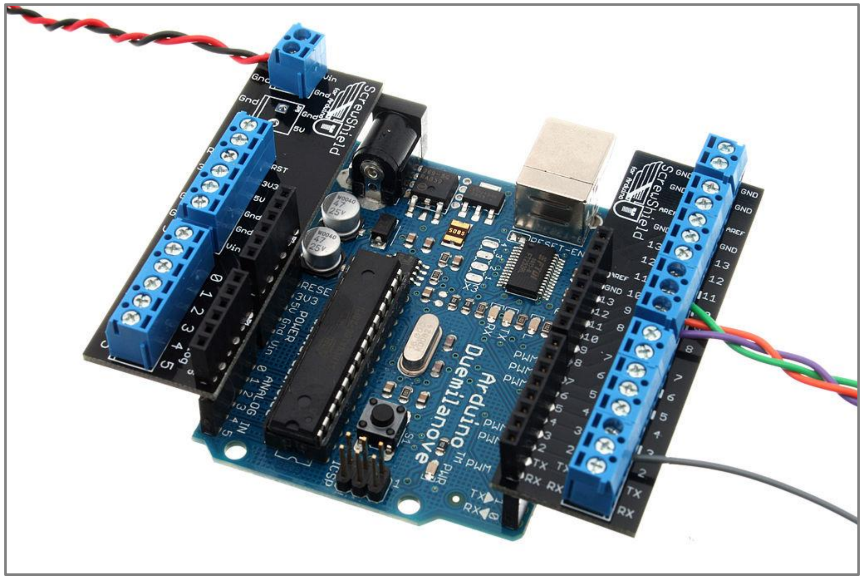
INTRODUCTION TO ARDUINO HARDWARE

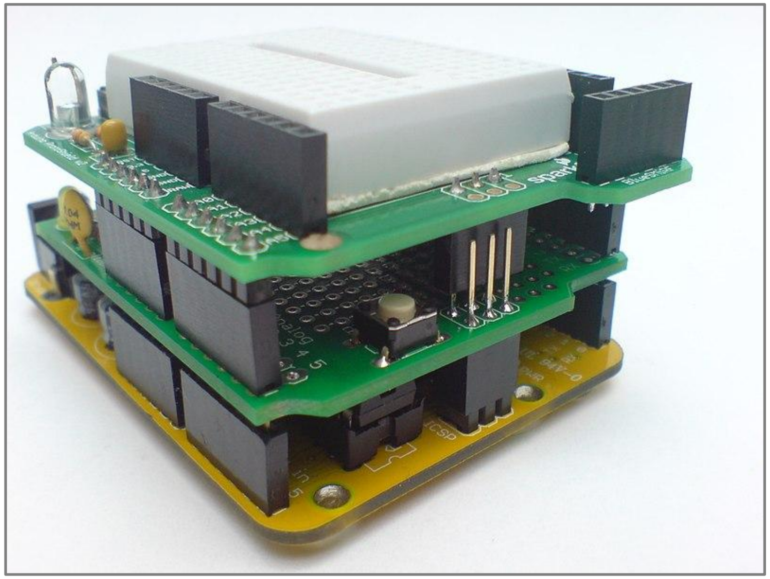
Few historical & interesting facts…



By Nicholas Zambetti - http://www.arduino.cc/, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=9182627.

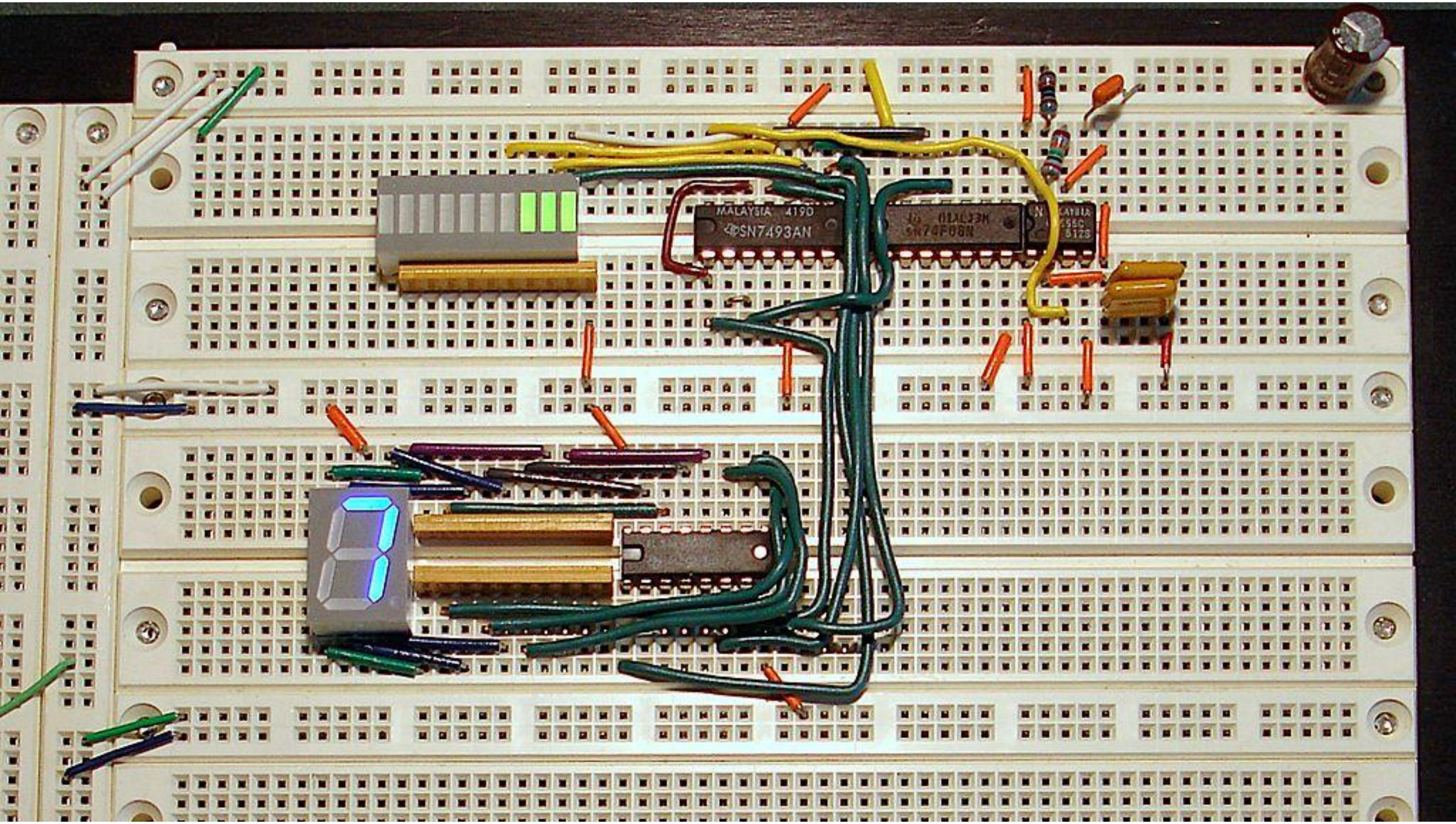
…





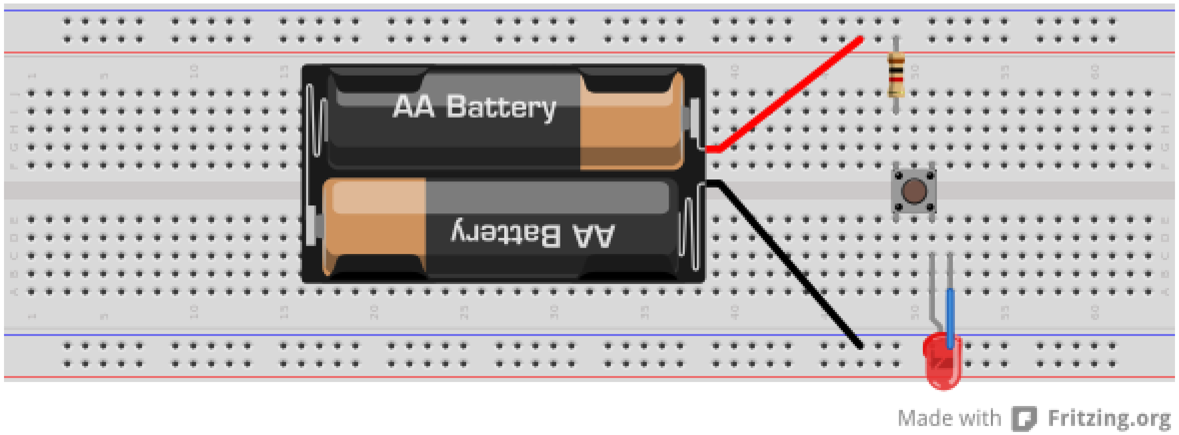
Left panel:
By oomlout - Flickr: Wingshield on Arduino - ARSH-05-WI, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=15911319.
Right panel:
By Marlon J. Manrique - Flickr: Arduino Protoboard Shield, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=15916962.

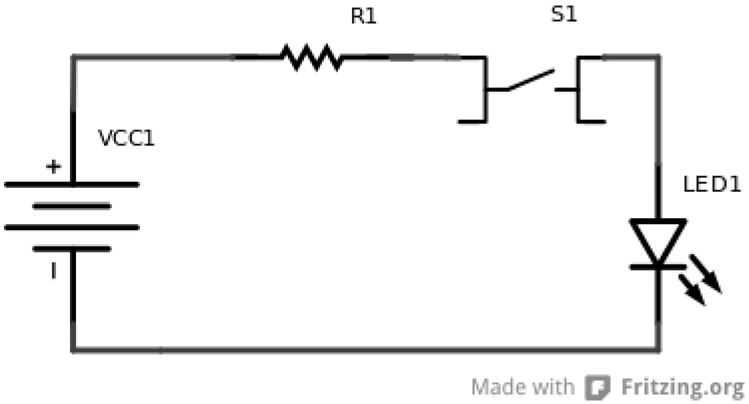
…



By en:User:Fulladder - en:Image:Breadboard counter.jpg, Copyrighted free use, https://commons.wikimedia.org/w/index.php?curid=1776425.

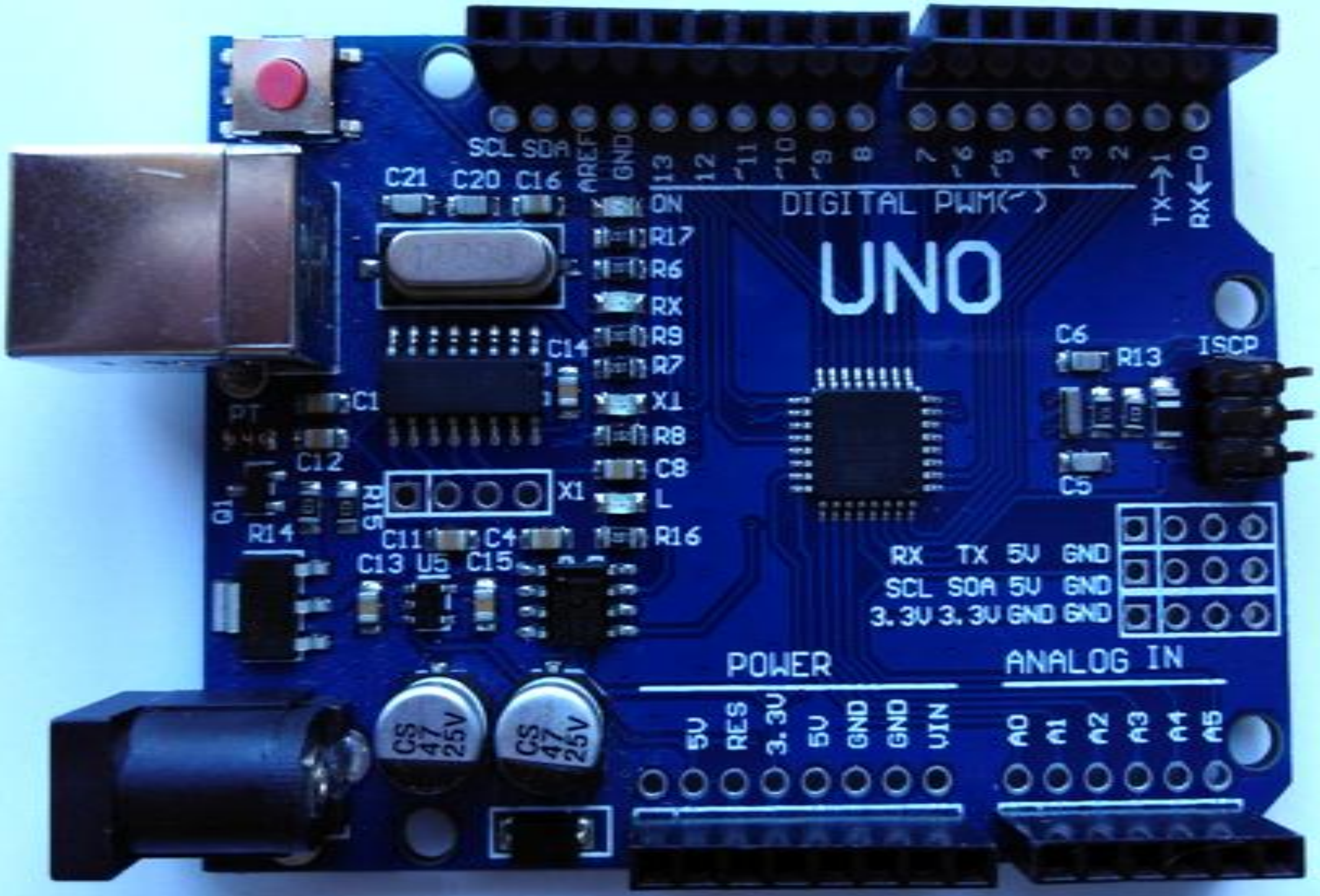
Fritzing, http://fritzing.org/home/



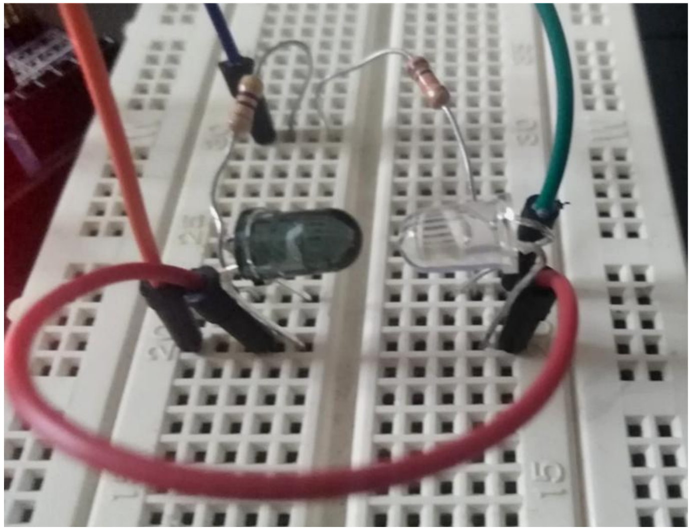


Upper panel: By LA2 - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=22714341.
Lower panel: By LA2 - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=22714342.

UNO R3 board


SCL SDA AREF GND 13 12 ~11 ~10 ~9 8
7 ~6 ~5 4 ~3 2 TX→1 RX←0
DIGITAL PWM(~)
UNO
ON
RX
L
ISCP
RX TX 5V GND
SCL SDA 5V GND
3.3V 3.3V GND GND
POWER
5V RES 3.3V 5V GND GND VIN
ANALOG IN
A0 A1 A2 A3 A4 A5

Sensor



Infrared sensor designed for lab. assignment for Measurement Computing Devices course at the University of Belgrade.

Transducer (https://en.wikipedia.org/wiki/Transducer) is device that converts a signal from one form of energy to other form.

Sensor (https://en.wikipedia.org/wiki/Sensor) is a transducer that converts signal from any form of energy to electrical energy.

There is relatively large number of sensors (https://en.wikipedia.org/wiki/List_of_sensors) used in a variety of applications… Including IoT (https://en.wikipedia.org/wiki/Internet_of_things)…

Sensor's characteristics

Basic static characteristics are:

1. Range (minimal and maximal measurable quantities)
2. Stability
3. Accuracy & precision
4. Hysteresis
5. Resolution (the smallest change it can detect in the quantity that it is measuring, see: https://en.wikipedia.org/wiki/Sensor#Resolution)*
6. Repeatability
7. Linearity (output is a linear function of the input)

* For digital sensors it is a resolution of A/D conversion. If the resolution is 12 bits and voltage range 0-5 V, then the resolution equals $5/2^{12}$ = 0.0012 V.

INTRODUCTION TO PROGRAMMING ARDUINO BOARDS

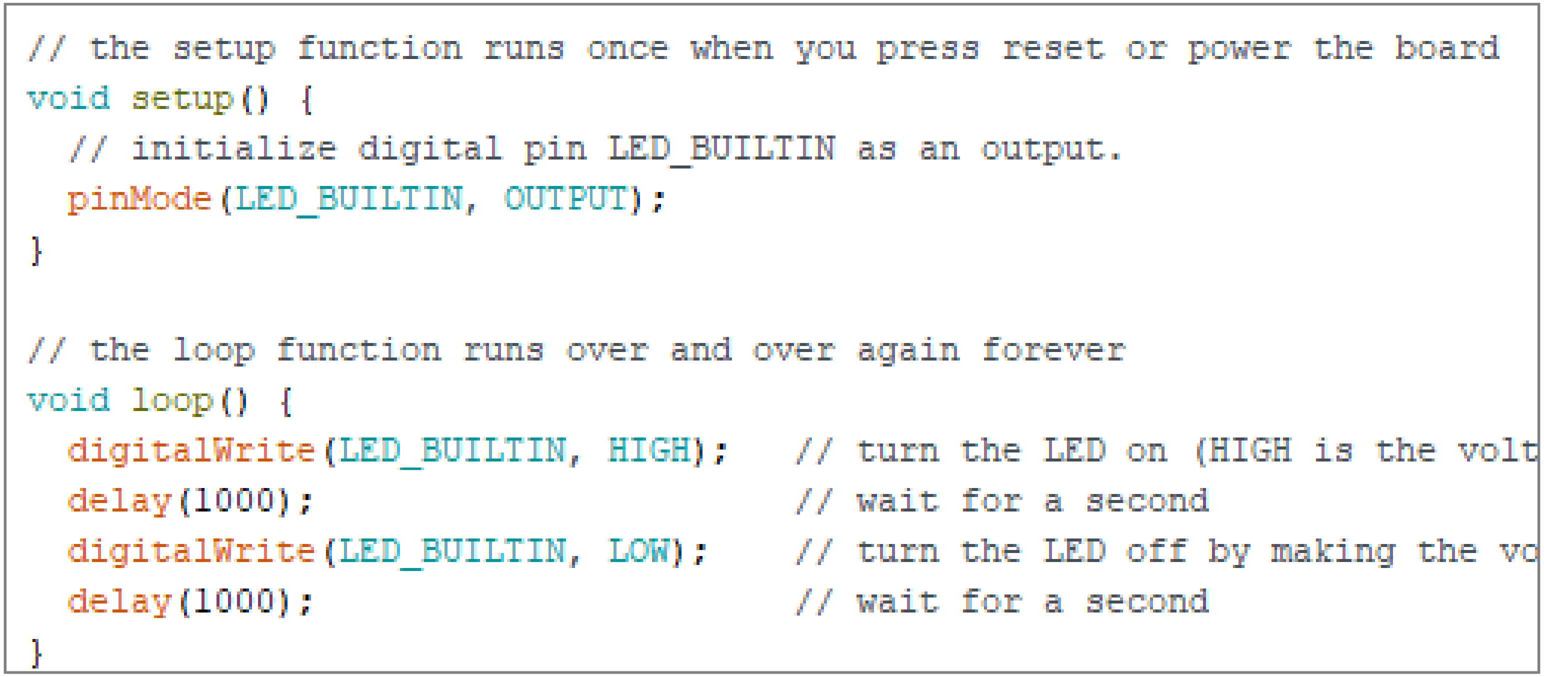
Arduino IDE

- Arduino hardware can be programmed from any programming language, but Arduino IDE is commonly used.
- Integrated Development Environment – IDE
- It is written in Java, C and C++. It can be installed on Windows, macOS and Linux (cross-platform application), https://en.wikipedia.org/wiki/Arduino.
  - Arduino programming is very similar to C++.
- NOTE: If you decide to use bulit-in examples and functions, you should check their compatibility with the available hardware.

AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

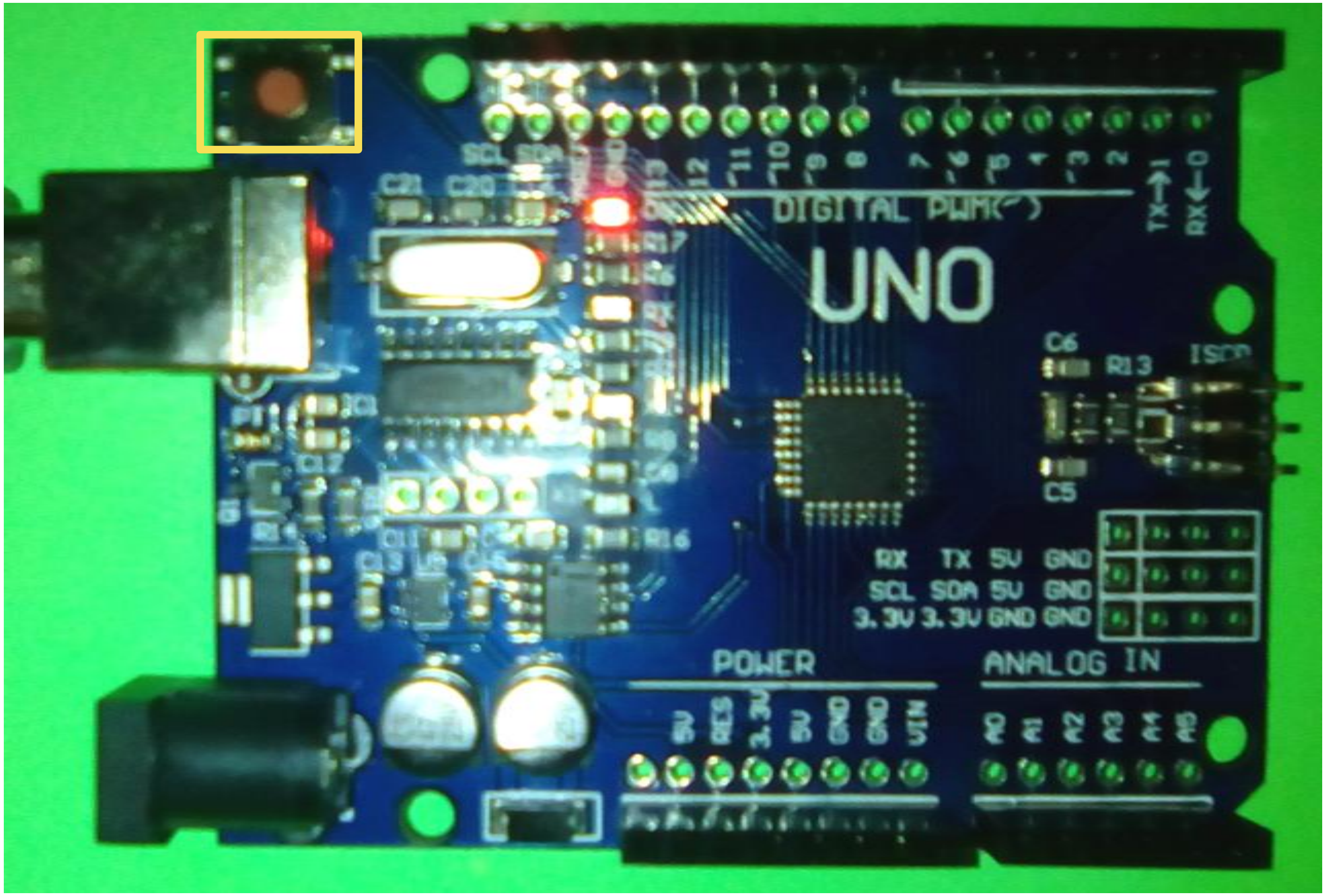
Arduino sketch



```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the volt
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the vo
  delay(1000);                       // wait for a second
}
```

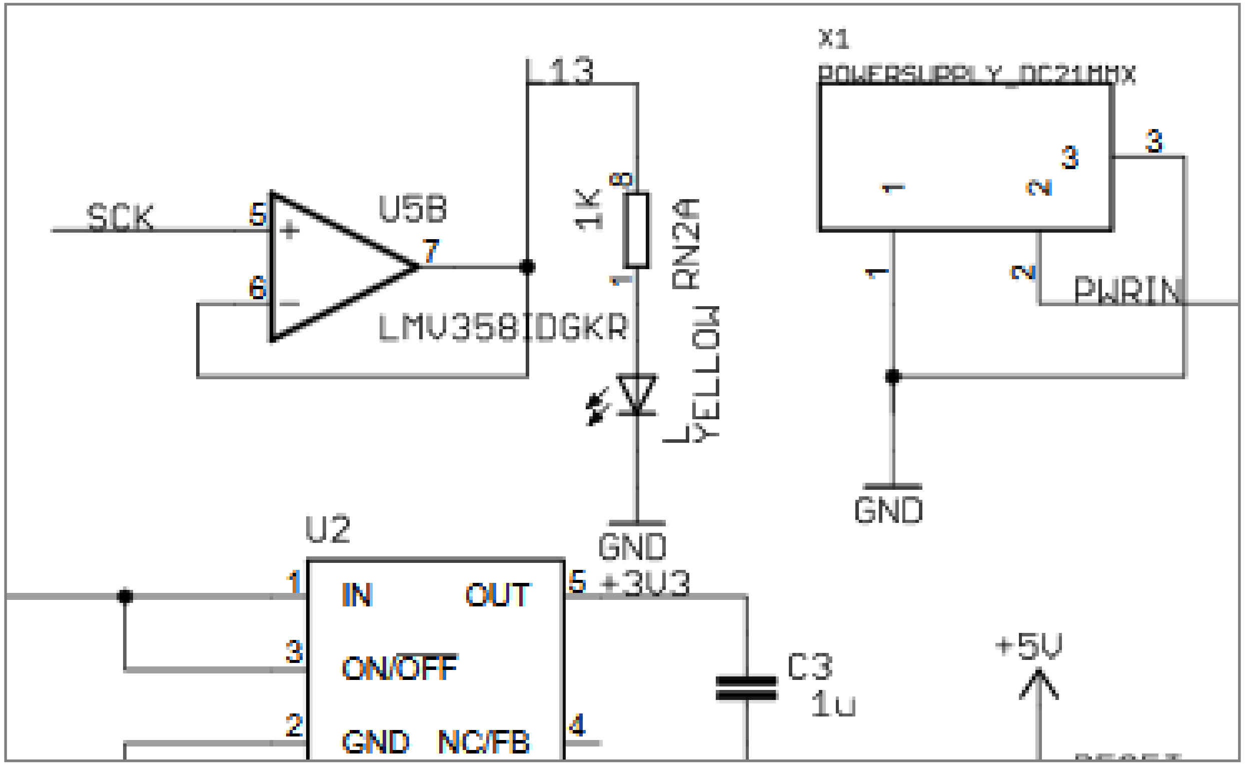
- Program written in Arduino IDE is called sketch.
- Sketches are saved with ***.ino*** extension in a separate folder.
- Minimal sketch contains two functions:
  - setup() that commonly contains code for initialization and
  - loop() executes repeatedly.
- Image represents simple built-in ***Blink.ino*** sketch.

RESET button



- Reinitializes program.
- Similar to power off.

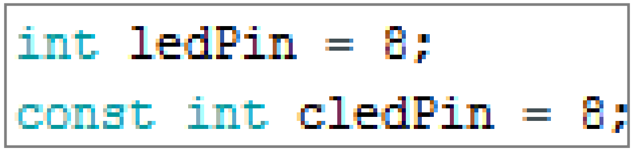
Built-in LED



- Pin 13 in UNO R3 microcontroller board and Arduino UNO contain built-in SMD (surface-mount technology) LED (light-emitting diode) marked as “L” on the board.
- Image presents Arduino UNO schematic with built-in diode, [Online], Assessed Dec. 8, 2018, https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf .
- Very useful for testing the board. Let’s try it.

VARIABLES AND STRUCTURES

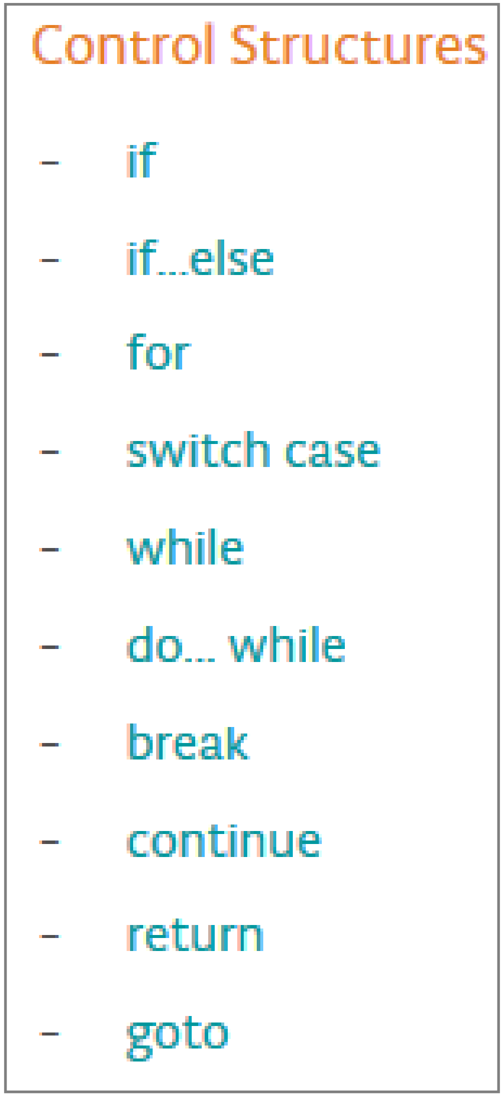
How to define variables?



```
int ledPin = 8;
const int cledPin = 8;
```

- Global:
  - Before *setup()* function or before *loop()* function if there is no need for initialization in *setup()*.
  - Integer variables are presented in image above (digital pin number).
  - Usually, names of constants can begin with letter “c”.
- Local:
  - Inside functions (*setup()*, *loop()* or other user-defined functions).

Control structures

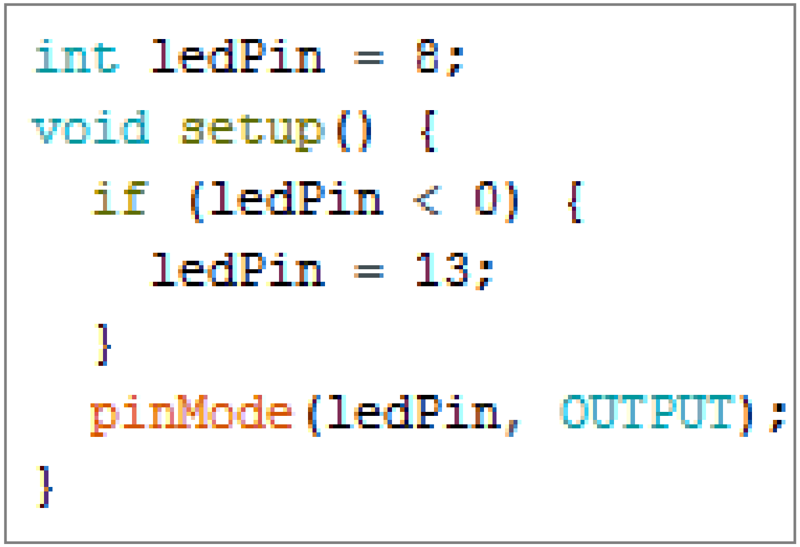


Control Structures

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

- Types of control structures are presented in Table.
- Details can be found at: https://www.arduino.cc/en/Reference/HomePage.

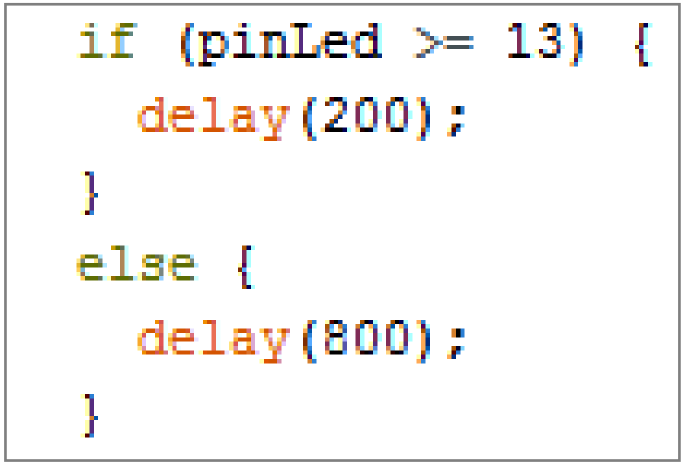
if structure



```
int ledPin = 8;
void setup() {
  if (ledPin < 0) {
    ledPin = 13;
  }
  pinMode(ledPin, OUTPUT);
}
```

- Presented in image above.
- Begin and end are marked with curly brackets “{}”.
- What is the function of the code above?

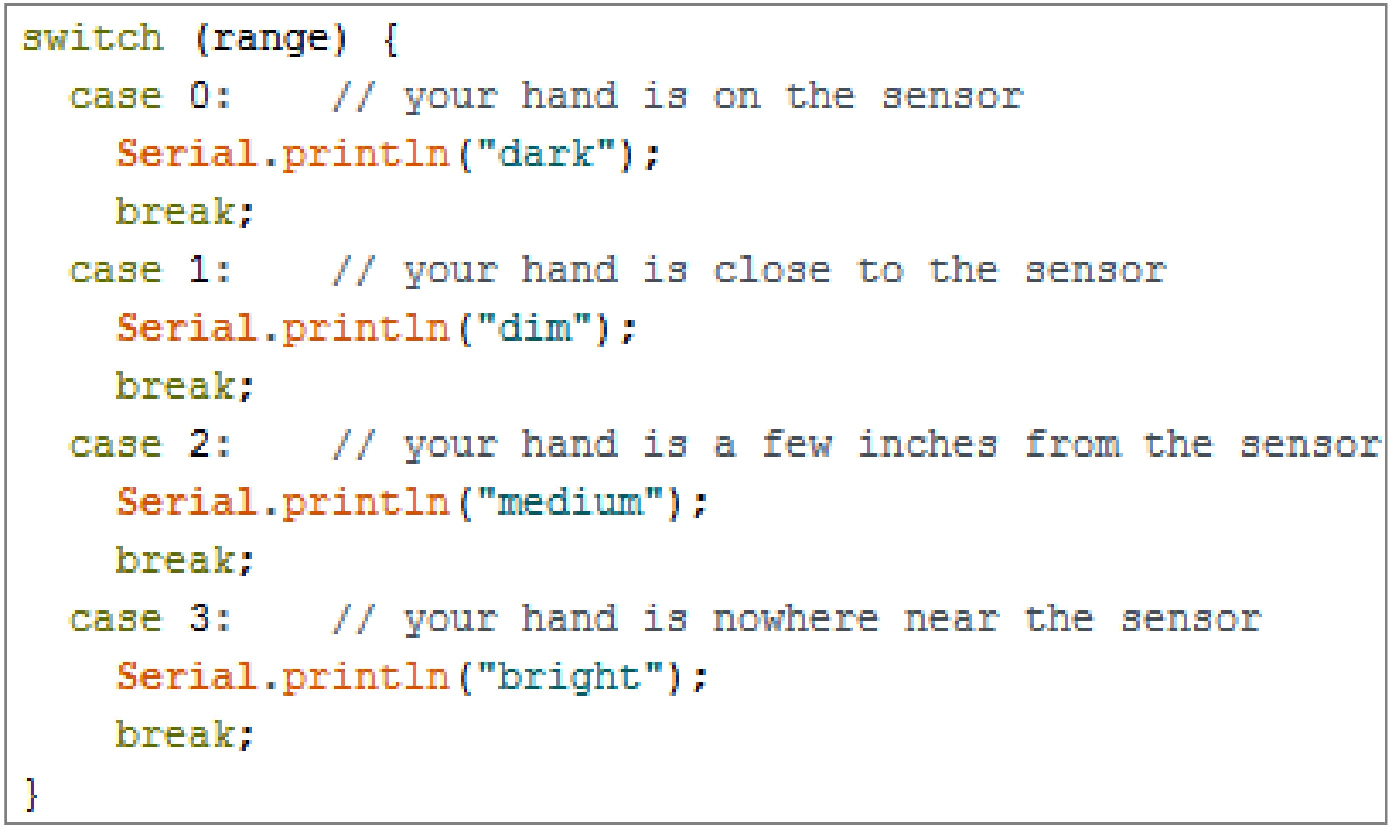
if else structure



```
if (pinLed >= 13) {
  delay(200);
}
else {
  delay(800);
}
```

- Presented in image above.
- Begin and end are marked with curly brackets “{}”.
- Auto-fill option is ON: when you type the “begin” sign “{” and press ENTER for new line, the “end” sign is automatically generated “}”. We all like to be spoiled sometimes.

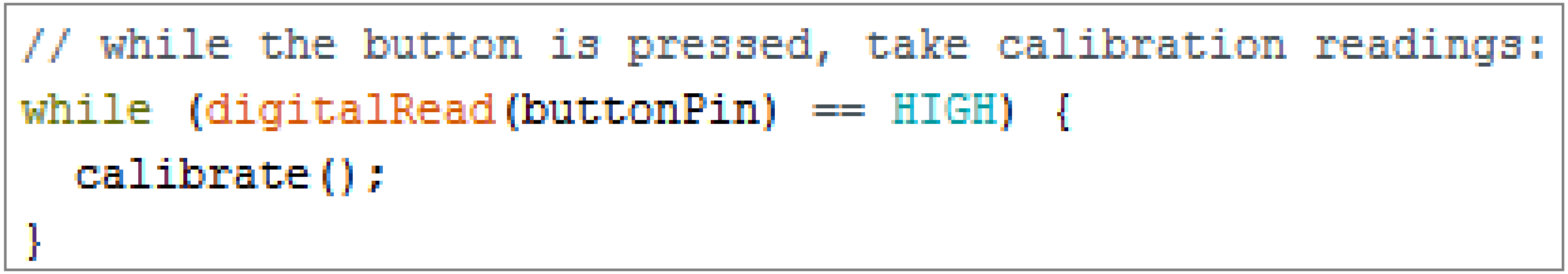
switch structure



```
switch (range) {
  case 0:    // your hand is on the sensor
    Serial.println("dark");
    break;
  case 1:    // your hand is close to the sensor
    Serial.println("dim");
    break;
  case 2:    // your hand is a few inches from the sensor
    Serial.println("medium");
    break;
  case 3:    // your hand is nowhere near the sensor
    Serial.println("bright");
    break;
}
```

- The built-in example from ***switchCase.ino*** is presented.
- Sample case structure is presented in image above.

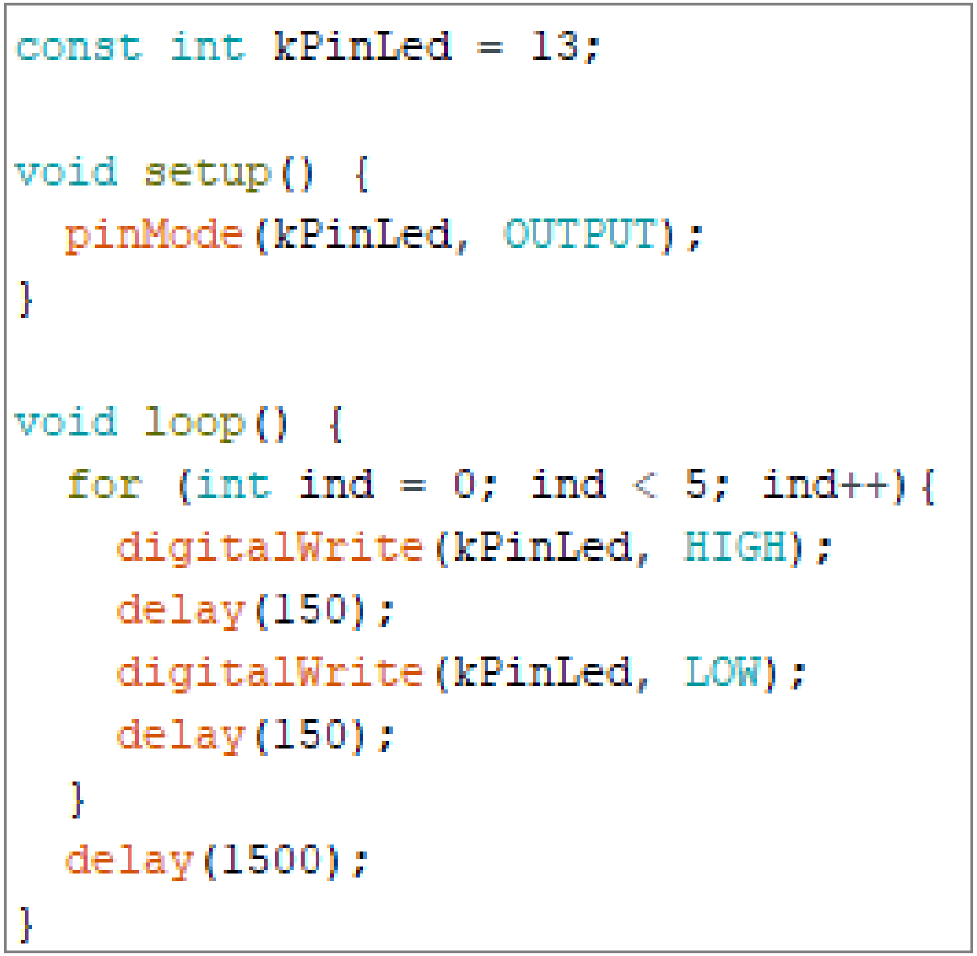
while loop



```
// while the button is pressed, take calibration readings:
while (digitalRead(buttonPin) == HIGH) {
  calibrate();
}
```

- Sample *while* loop is presented in image above.
- Built-in example code is presented from ***WhileStatementConditional.iso***.

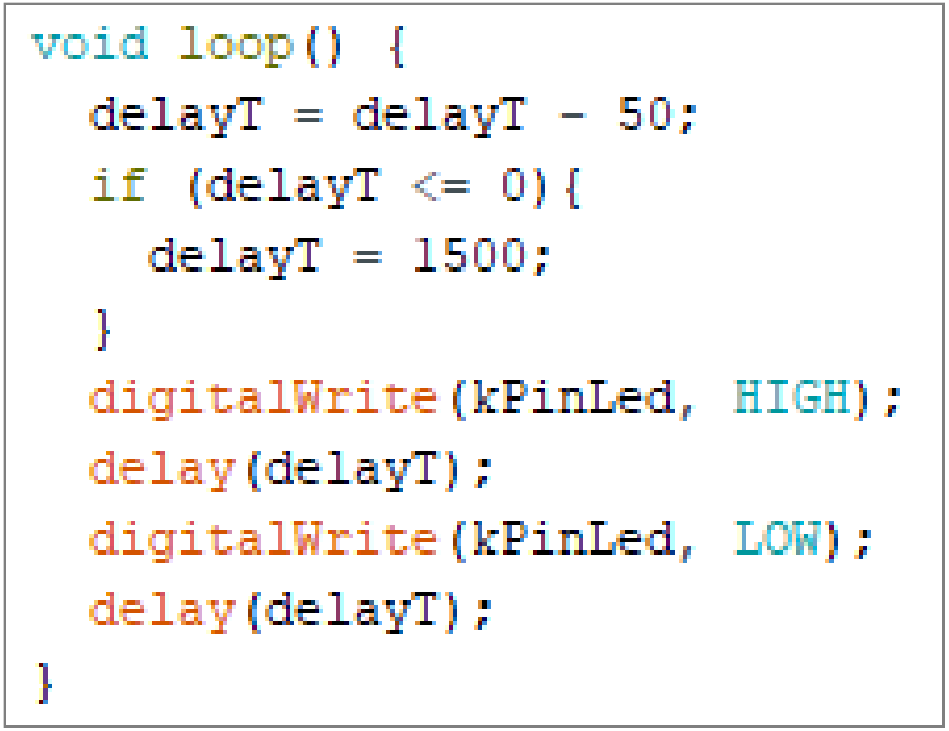
for loop



```
const int kPinLed = 13;

void setup() {
  pinMode(kPinLed, OUTPUT);
}

void loop() {
  for (int ind = 0; ind < 5; ind++){
    digitalWrite(kPinLed, HIGH);
    delay(150);
    digitalWrite(kPinLed, LOW);
    delay(150);
  }
  delay(1500);
}
```

- Sample *for* loop is presented in image.
- What is the result of this code?
- Code is adopted/inspired from: Alan G. Smith, Introduction to Arduino: A piece of cake!, 2011.

What does this program do?

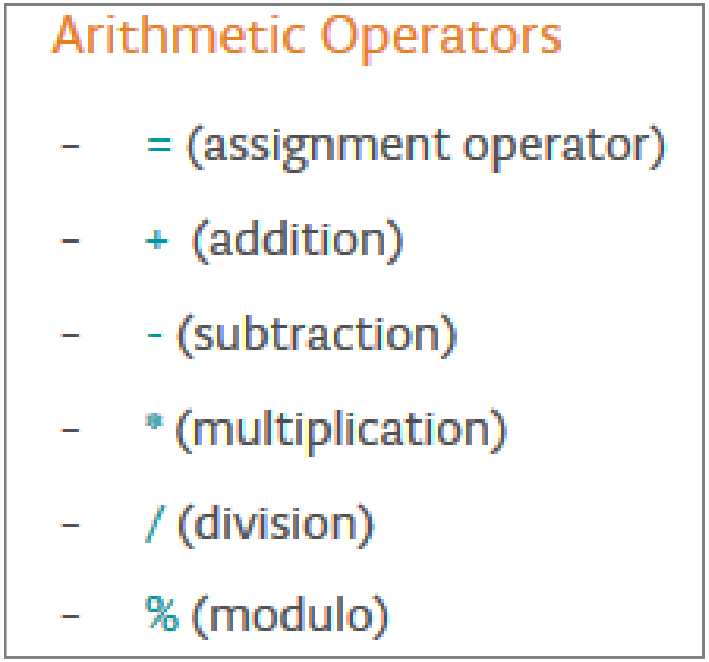


```
void loop() {
  delayT = delayT - 50;
  if (delayT <= 0){
    delayT = 1500;
  }
  digitalWrite(kPinLed, HIGH);
  delay(delayT);
  digitalWrite(kPinLed, LOW);
  delay(delayT);
}
```

- Code is adopted/inspired from: Alan G. Smith, Introduction to Arduino: A piece of cake!, 2011.
- Only part of the code is presented in image above. What is missing?
- Usability of this code? Holidays are approaching…

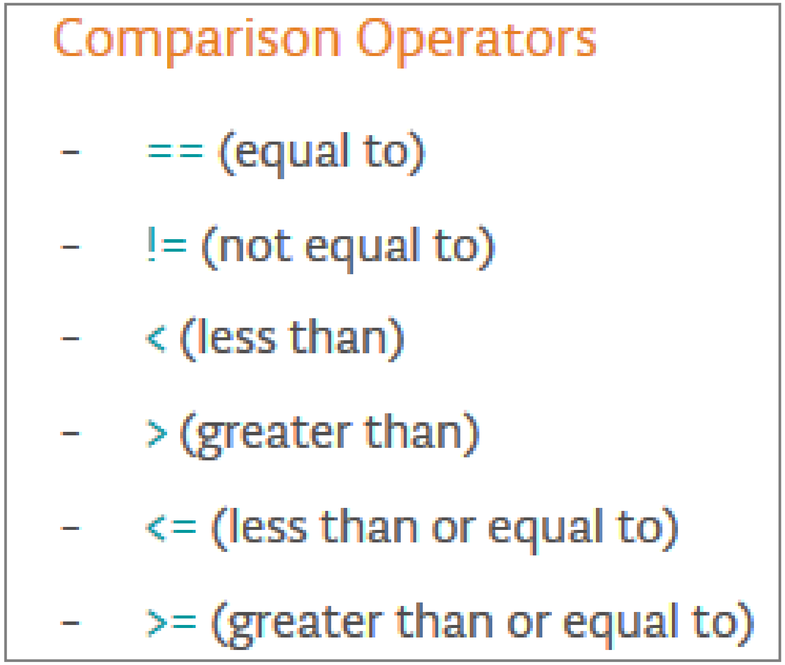
OPERATORS

Arithmetic and comparison operators



Arithmetic Operators

- = (assignment operator)
- + (addition)
- - (subtraction)
- * (multiplication)
- / (division)
- % (modulo)

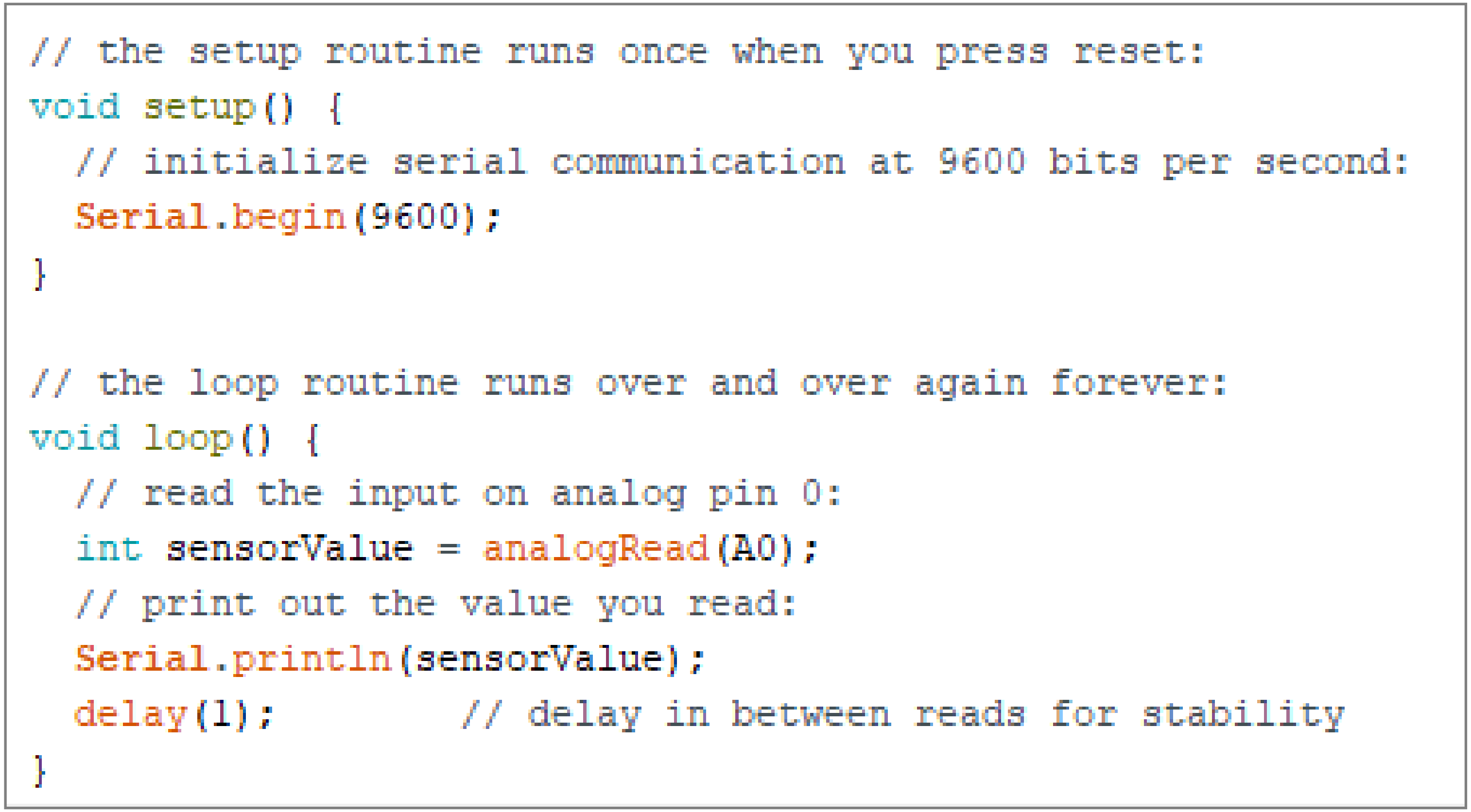


Comparison Operators

- == (equal to)
- != (not equal to)
- < (less than)
- > (greater than)
- <= (less than or equal to)
- >= (greater than or equal to)

- More at: https://www.arduino.cc/en/Reference/HomePage.
- Integer division is performed by "/" if numerator and denominator are integers. Be very careful! Let's see an example.

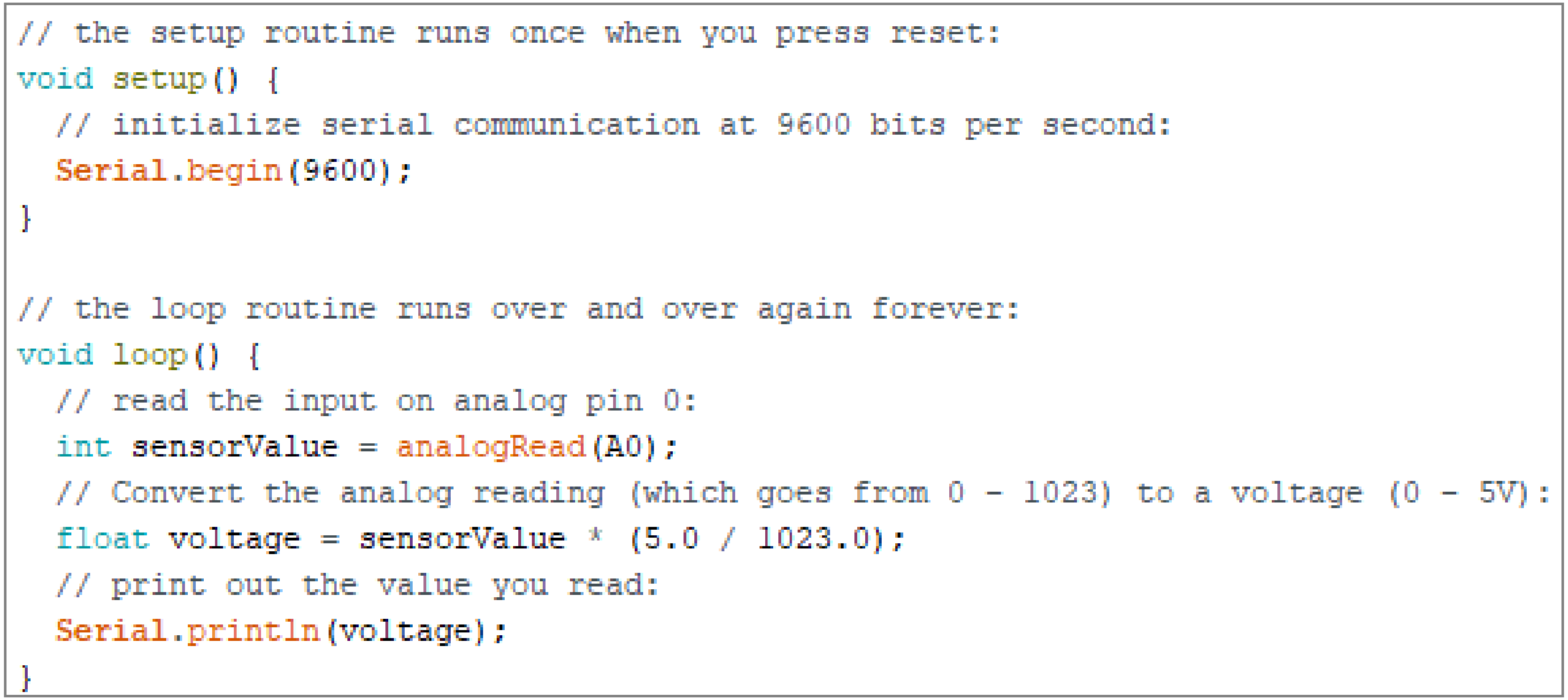
Built-in analog read serial



```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

- Let’s test this code.
- Let’s test the sensor.
- I prepared photo resistor with voltage divider, but you can use any other sensor or circuit. You will measure voltage with A0.

Built-in read analog voltage

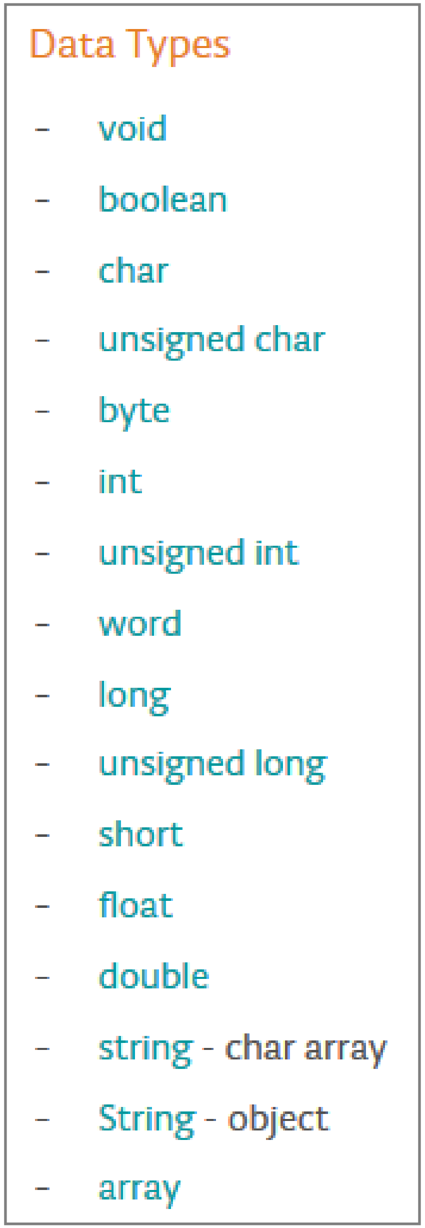


```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
  float voltage = sensorValue * (5.0 / 1023.0);
  // print out the value you read:
  Serial.println(voltage);
}
```

- The same circuit.
- Different code.
- How different?

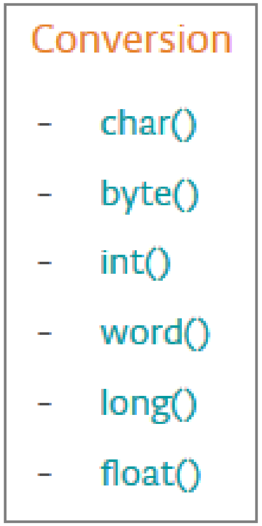
Data types

- More at: https://www.arduino.cc/en/Reference/HomePage.
- For conversion use recommended functions.



Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- short
- float
- double
- string - char array
- String - object
- array



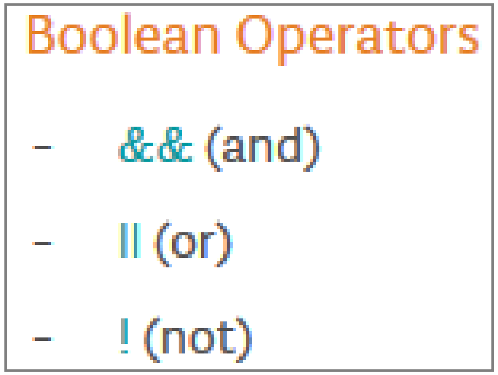
Conversion

- char()
- byte()
- int()
- word()
- long()
- float()

Serial port

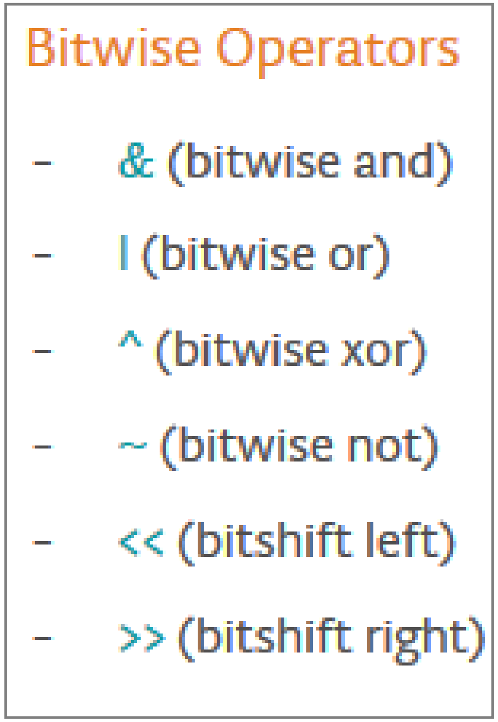
- Serial port is used for communication between the microcontroller board and a computer or other devices.
- Serial Monitor and Serial Plotter can be used to check the content of the serial port (note that baud rates should be the same in the serial monitor and your Arduino sketch).
- More at: https://www.arduino.cc/reference/en/language/functions/communication/serial/.

Logical and compound operators



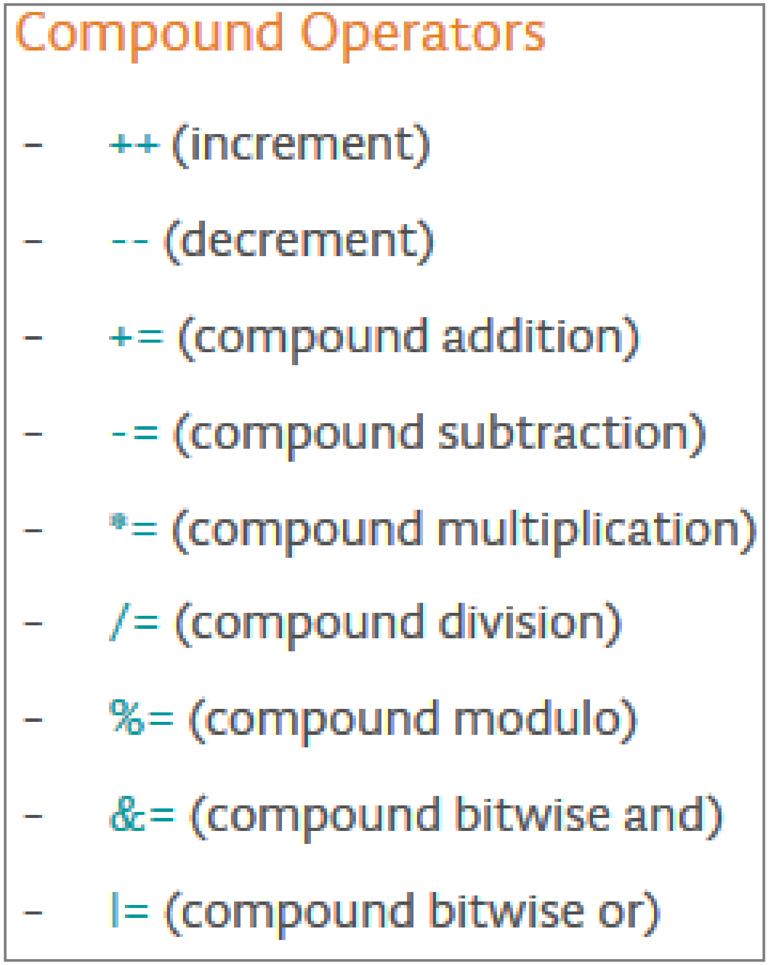
Boolean Operators

- && (and)
- || (or)
- ! (not)



Bitwise Operators

- & (bitwise and)
- | (bitwise or)
- ^ (bitwise xor)
- ~ (bitwise not)
- << (bitshift left)
- >> (bitshift right)

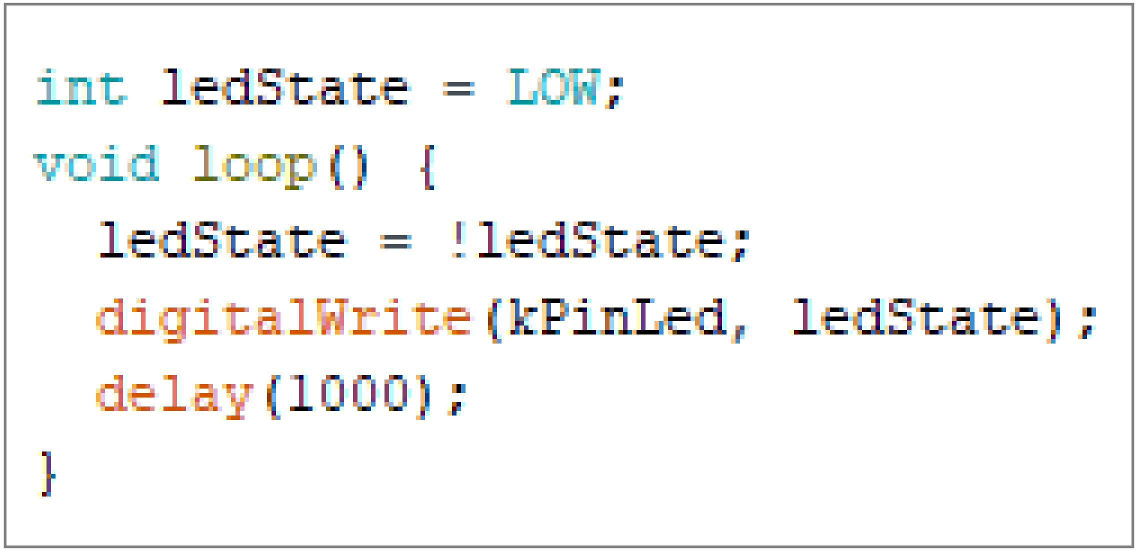


Compound Operators

- ++ (increment)
- -- (decrement)
- += (compound addition)
- -= (compound subtraction)
- *= (compound multiplication)
- /= (compound division)
- %= (compound modulo)
- &= (compound bitwise and)
- |= (compound bitwise or)

More at: https://www.arduino.cc/en/Reference/HomePage.

What does the program do?

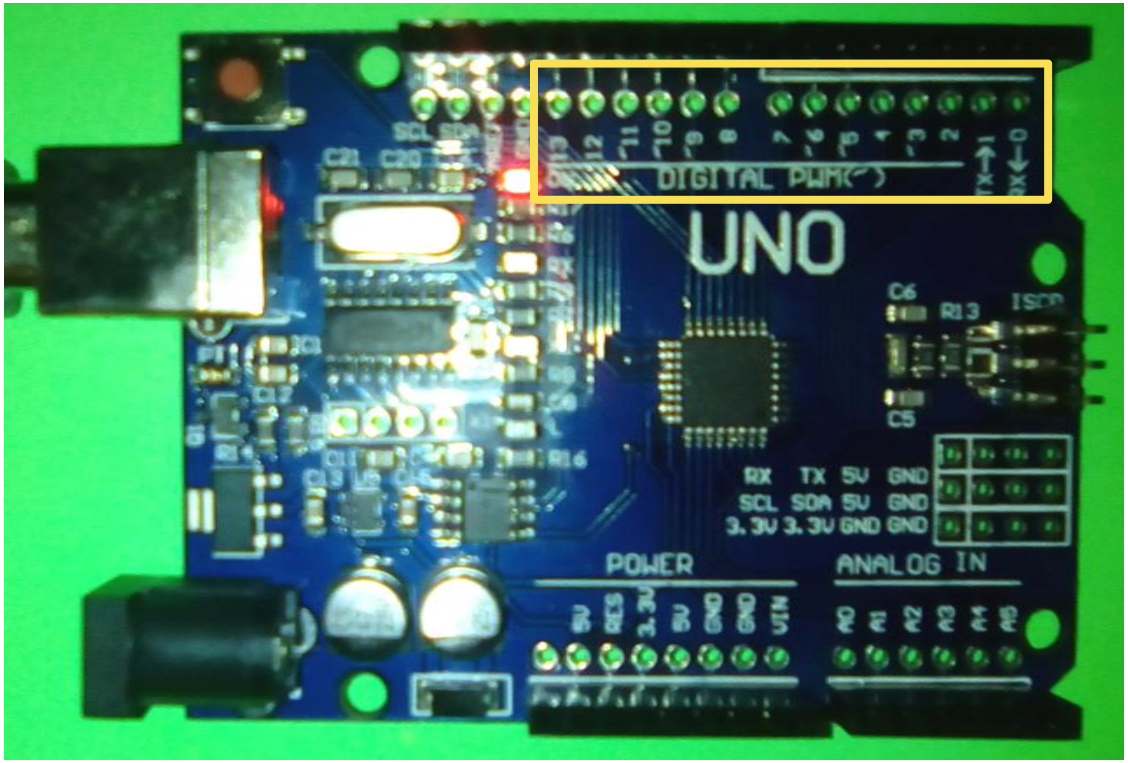


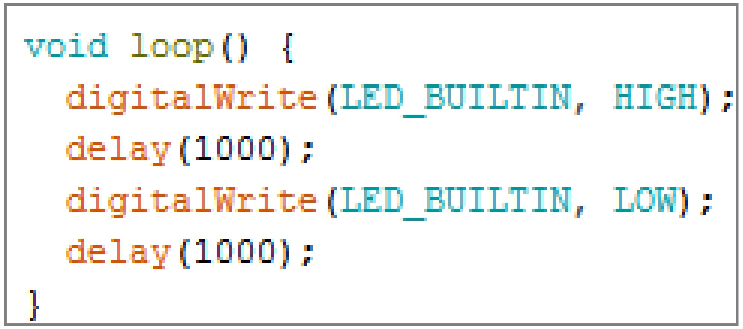
```
int ledState = LOW;
void loop() {
  ledState = !ledState;
  digitalWrite(kPinLed, ledState);
  delay(1000);
}
```

- Code is adopted/inspired from: Alan G. Smith, Introduction to Arduino: A piece of cake!, 2011.
- LOW and HIGH are default logical variables. Instead of them, you can use FALSE and TRUE.

PWM

Pulse Width Modulation PWM





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

- Instead of *digitalWrite(), analogWrite()* is used.
- “DIGITAL PWM” on microcontroller board next to digital pins.
- NOTE: UNO board does not have analog output/s.

PWM definition(s)

- Pulse Width Modulation (PWM) is a fancy term for describing a type of digital signal. Pulse width modulation is used in a variety of applications including sophisticated control circuitry (https://learn.sparkfun.com/tutorials/pulse-width-modulation).
- Pulse-width modulation (PWM), or pulse-duration modulation (PDM), is a modulation technique used to encode a message into a pulsing signal (https://en.wikipedia.org/wiki/Pulse-width_modulation).
- Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off (https://www.arduino.cc/en/Tutorial/PWM).

PWM

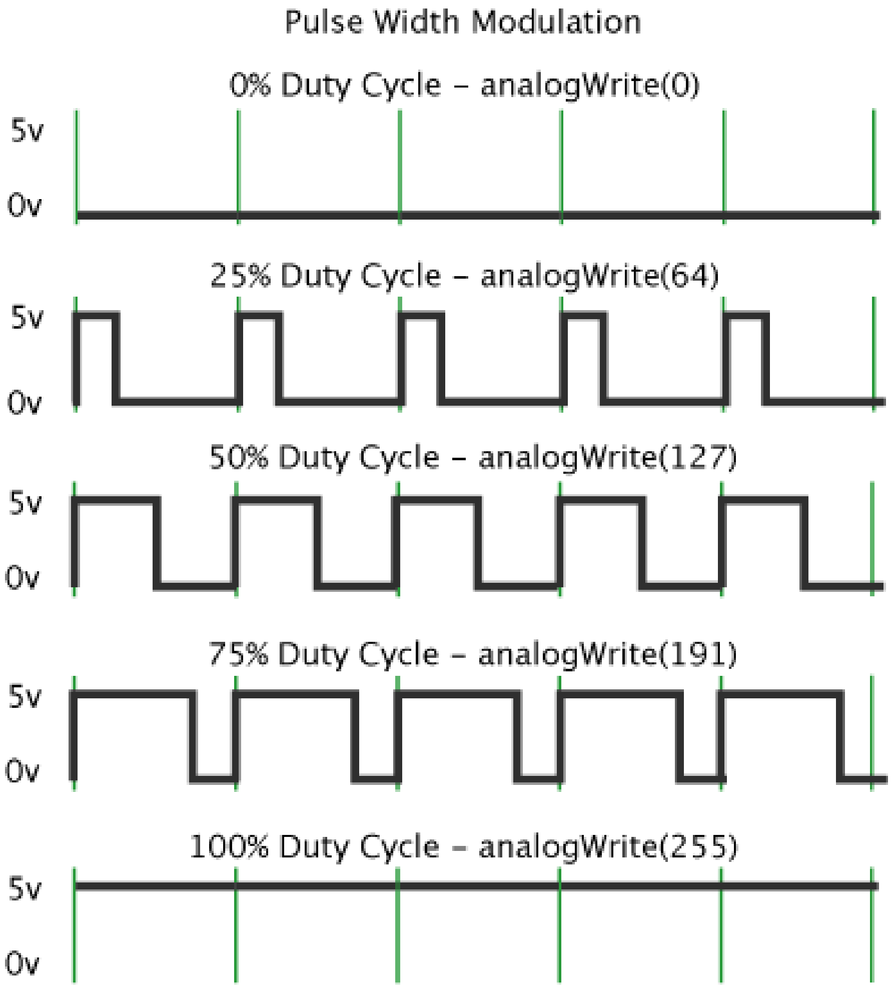
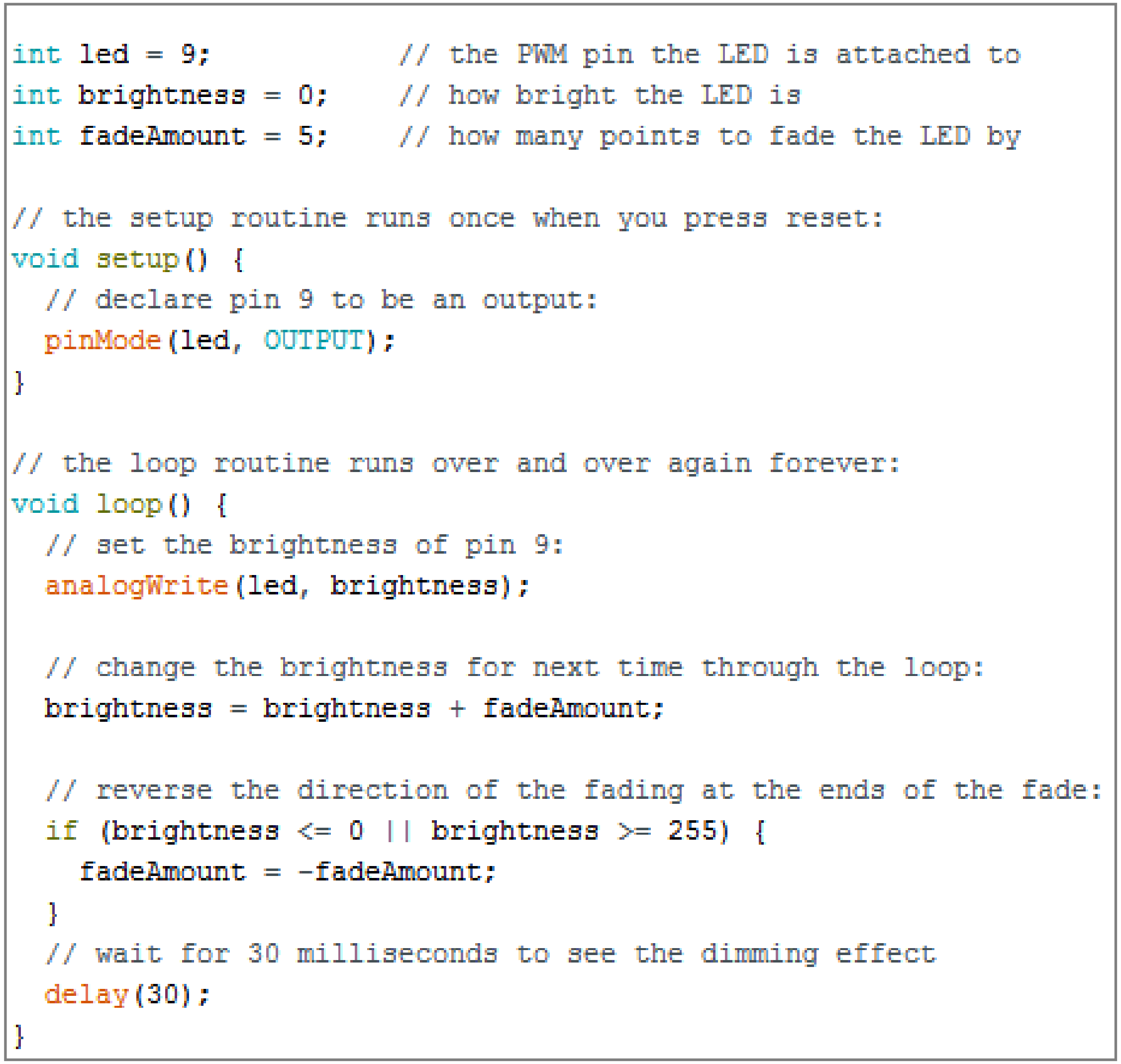

Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
5v
0v
25% Duty Cycle – analogWrite(64)
5v
0v
50% Duty Cycle – analogWrite(127)
5v
0v
75% Duty Cycle – analogWrite(191)
5v
0v
100% Duty Cycle – analogWrite(255)
5v
0v

Image is adopted from: https://www.arduino.cc/en/uploads/Tutorial/pwm.gif.

analogWrite() example

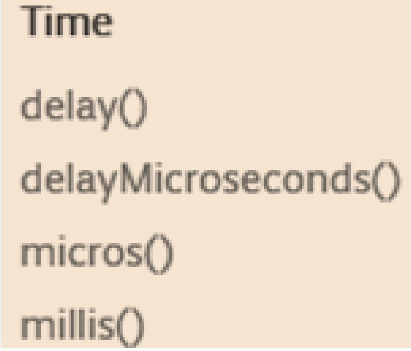


```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

- Built-in ***Fade.ino*** sketch is presented in image.
- What does this sketch do?
- You can control LED brightness, but also motors by digital ports.

MORE USEFUL FUNCTIONS

Time functions



Time

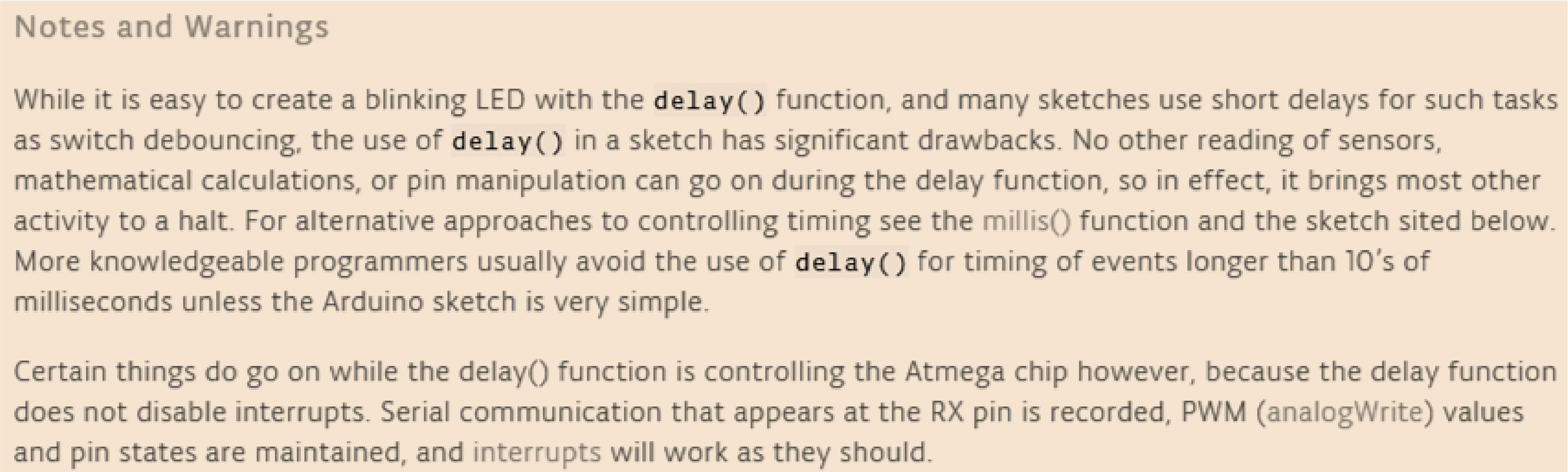
delay()

delayMicroseconds()

micros()

millis()

- Why you shouldn’t use delay?
- Be aware of constraints.
- I recommend ***BlinkWithoutDelay.ino*** from: https://www.arduino.cc/en/Tutorial/BlinkWithoutDelay.
- Text for this slide is adopted from: https://www.arduino.cc/reference/en/language/functions/time/delay/.

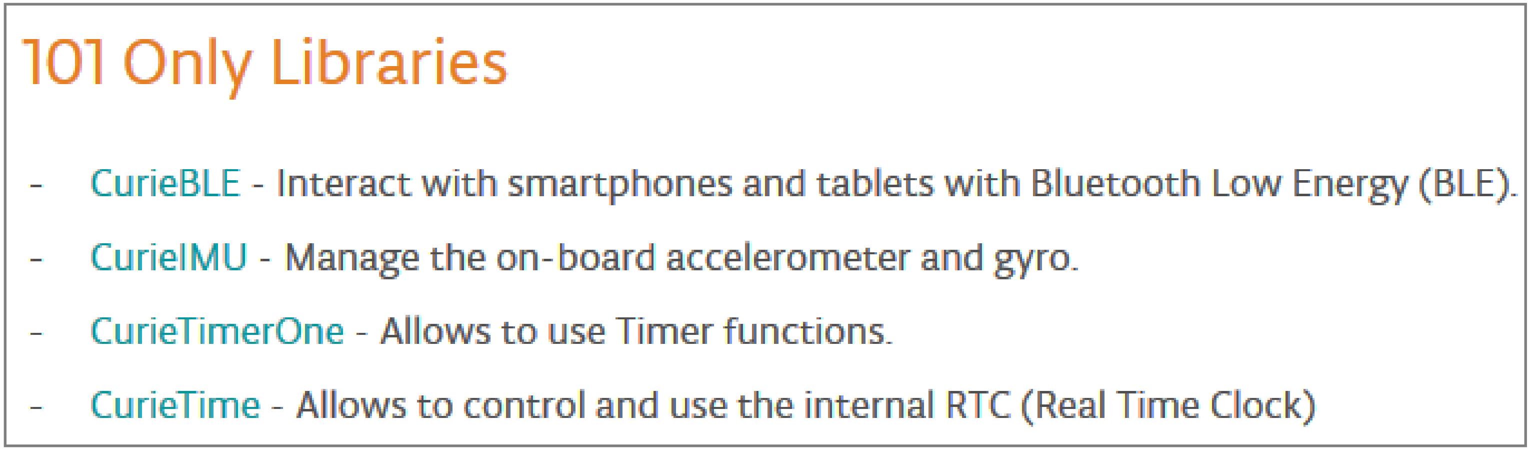


Notes and Warnings

While it is easy to create a blinking LED with the `delay()` function, and many sketches use short delays for such tasks as switch debouncing, the use of `delay()` in a sketch has significant drawbacks. No other reading of sensors, mathematical calculations, or pin manipulation can go on during the delay function, so in effect, it brings most other activity to a halt. For alternative approaches to controlling timing see the millis() function and the sketch sited below. More knowledgeable programmers usually avoid the use of `delay()` for timing of events longer than 10’s of milliseconds unless the Arduino sketch is very simple.

Certain things do go on while the delay() function is controlling the Atmega chip however, because the delay function does not disable interrupts. Serial communication that appears at the RX pin is recorded, PWM (analogWrite) values and pin states are maintained, and interrupts will work as they should.

Other Arduino libraries

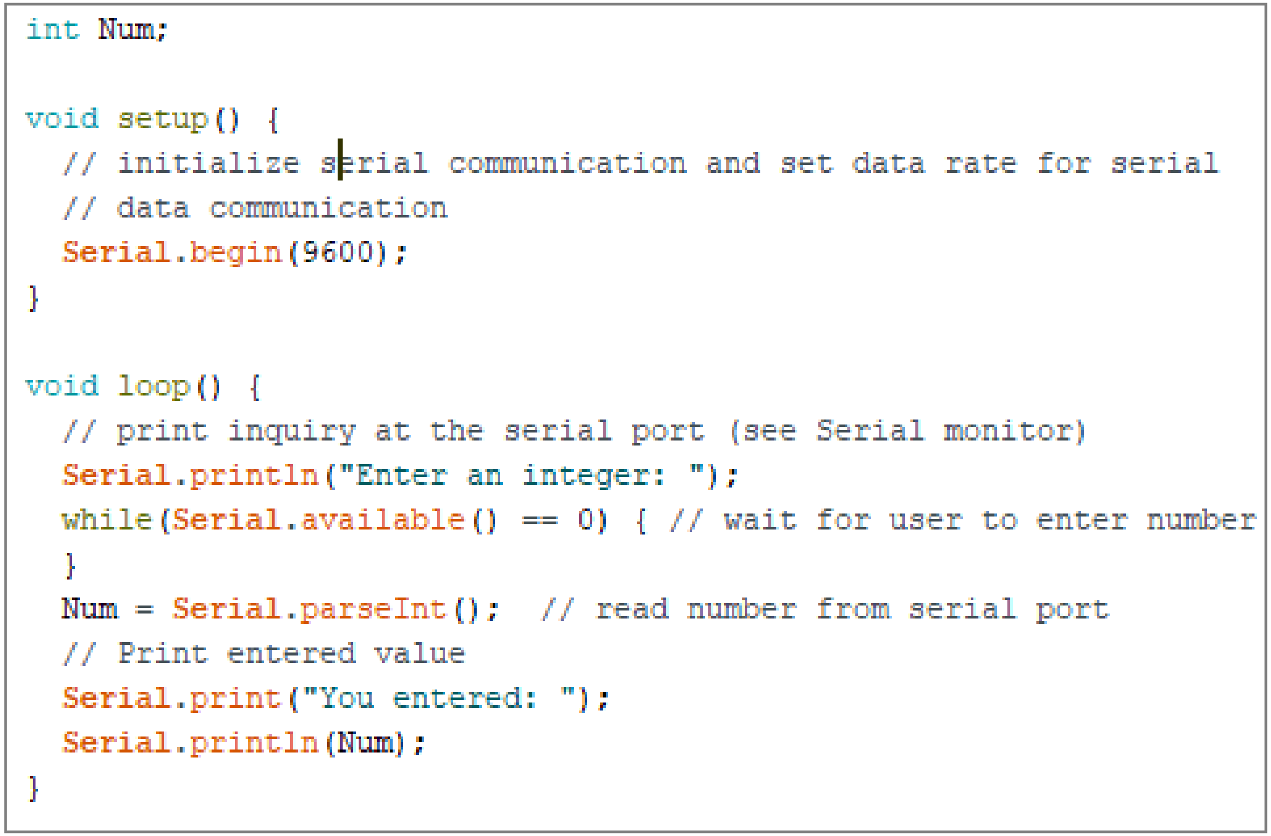


101 Only Libraries

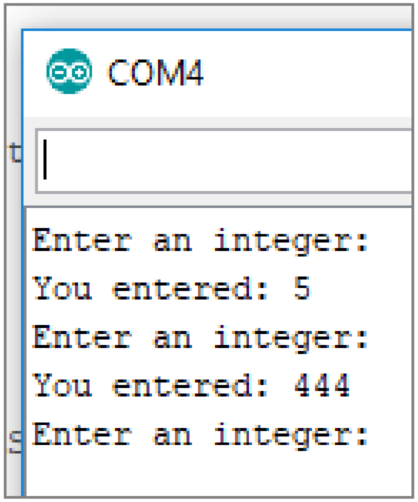
- CurieBLE - Interact with smartphones and tablets with Bluetooth Low Energy (BLE).
- CurieIMU - Manage the on-board accelerometer and gyro.
- CurieTimerOne - Allows to use Timer functions.
- CurieTime - Allows to control and use the internal RTC (Real Time Clock)

- Some libraries are intended for use with specialized hardware only.
- In order to import libraries choose from *Sketch* drop-down menu option *Import Library*. Usually in ***.zip*** format.
- More at: https://www.arduino.cc/en/Hacking/LibraryTutorial.
- Advice: Never trust completely libraries downloaded from unknown or unreliable sources! Test them yourself and be careful.

In case you need interactivity



```
int Num;

void setup() {
  // initialize serial communication and set data rate for serial
  // data communication
  Serial.begin(9600);
}

void loop() {
  // print inquiry at the serial port (see Serial monitor)
  Serial.println("Enter an integer: ");
  while(Serial.available() == 0) { // wait for user to enter number
  }
  Num = Serial.parseInt();  // read number from serial port
  // Print entered value
  Serial.print("You entered: ");
  Serial.println(Num);
}
```



```
Enter an integer:
You entered: 5
Enter an integer:
You entered: 444
Enter an integer:
```

- Serial port can be used for both output and input.
- Use *Serial.parseInt()* function: https://www.arduino.cc/en/Serial/ParseInt.
- Sample code is presented.
- Let’s try it?
- What will happen if user enters 1.3?

ARDUINO
OPEN-SOURCE
COMMUNITY

Arduino

Thank you EESTEC for inviting me to joint this workshop!

Good luck with your projects!

Special thanks to Prof. Predrag Pejović (http://tnt.etf.rs/~peja/) for supporting my work in introducing Arduino to the ETF curriculum.

Slides and materials for this presentation are available online at: https://github.com/NadicaSm/.

More code and materials (in Serbian) at: http://automatika.etf.bg.ac.rs/sr/13e053msr.