DIY Workshop

By Dr Sangit Sasidhar

Acknowledgements:

- Dr Rajesh Panicker
- www.Arduino.cc
- www.sparkfun.com

(Some slides from Arduino introduction slides by Linz Craig, Nick Poole, Prashanta Aryal, Theo Simpson, Tai Johnson, and Eli Santistevan)

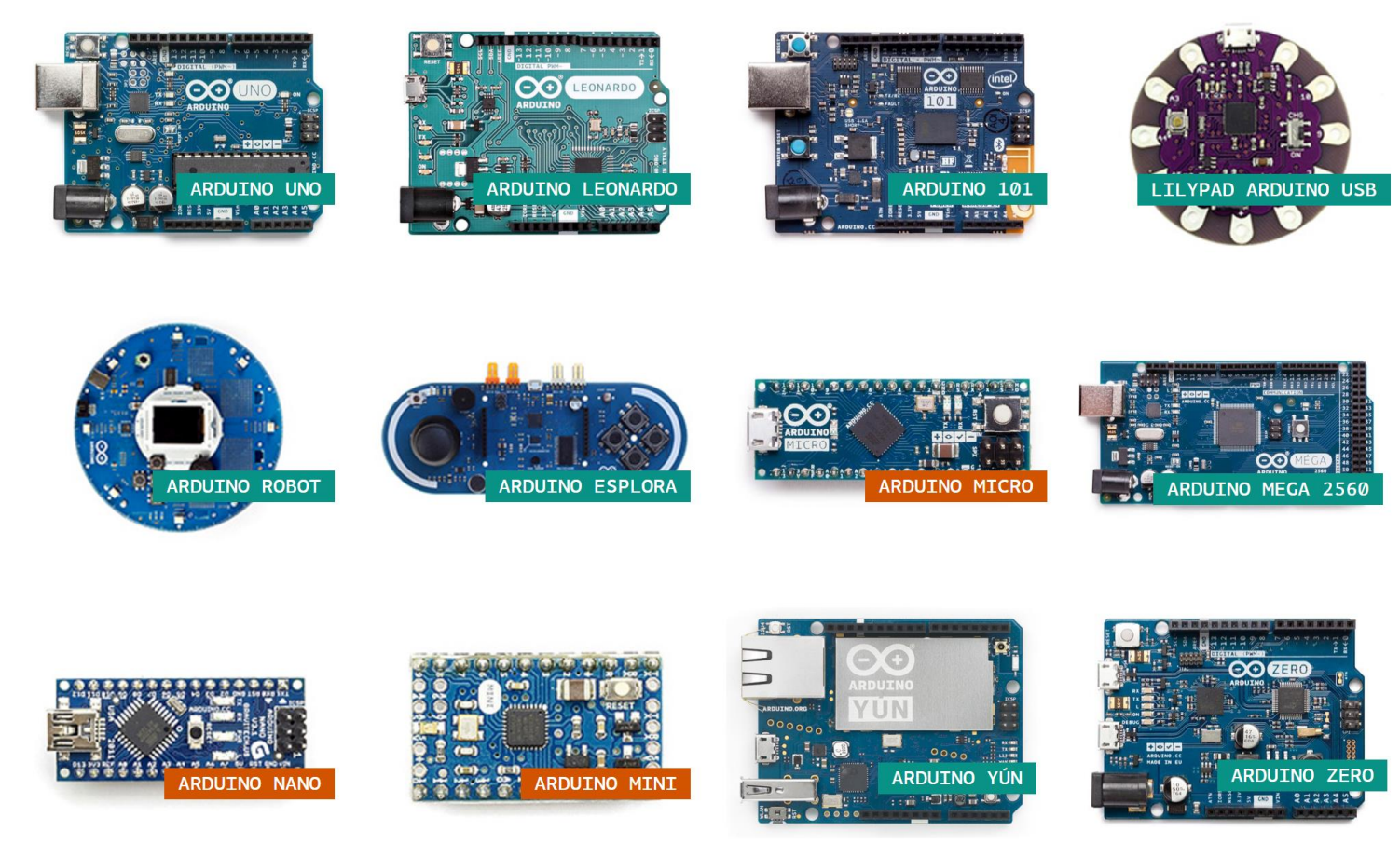
What is Arduino?

- Arduino is an open-source electronics platform based on easy-to-use hardware and software
- Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online
- You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (C++), and the Arduino Software (IDE)
- A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike

Why Arduino?

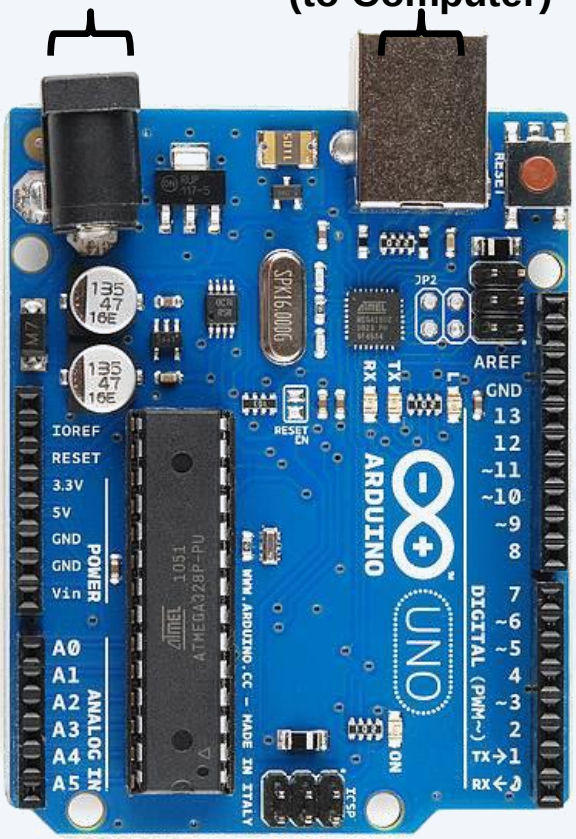
- Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments
- Inexpensive
- Cross-platform (IDE works on Windows, Mac and Linux, Raspberry Pi)
- Simple, clear programming environment
- Open-source hardware empowering users to build them independently and eventually adapt them to their particular needs
- Software growing through the contributions of users worldwide

Arduino Boards (the “Brain”)



https://www.arduino.cc/en/Main/Products

Arduino Uno (most popular)



Input vs. Output

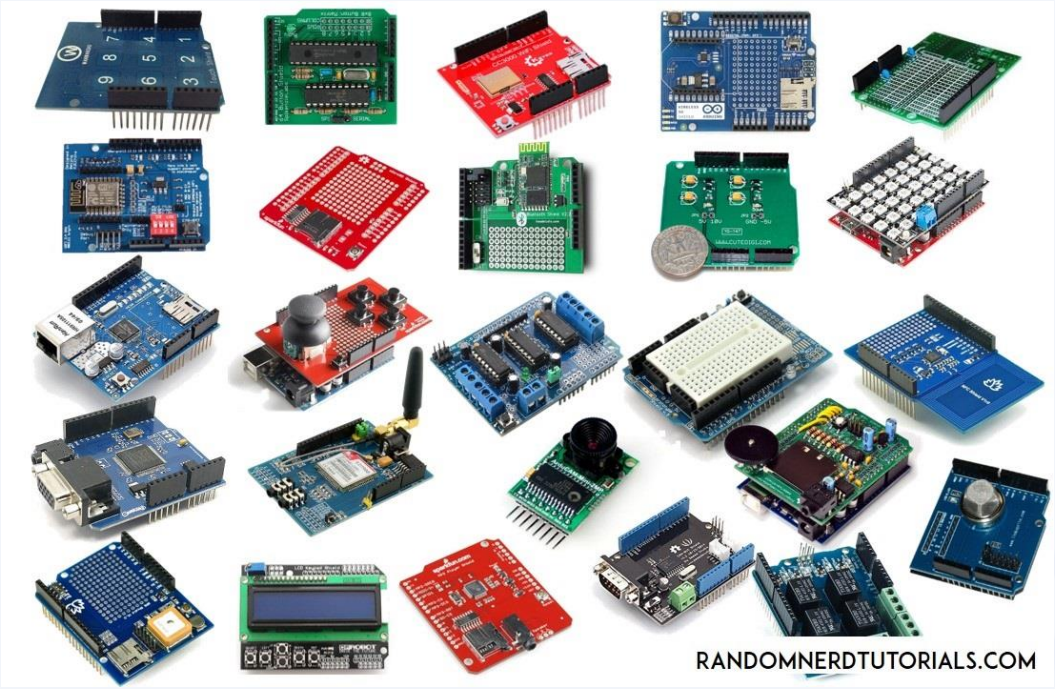
Referenced from the perspective of the Arduino Board

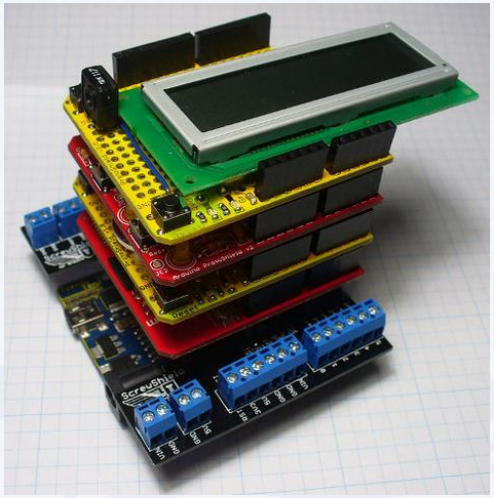
Inputs is a signal / information going into the board

Output is any signal exiting the board

- **Almost all systems that use physical computing will have some form of output**
- **A device which can provide input(s) is called an input device, usually referred to as sensors. Ex: Light sensors (LDRs), Accelerometers, Push buttons**
- **Output devices are usually referred to as actuators Ex: Motors, LEDs**

Shields (the “Body Parts”)

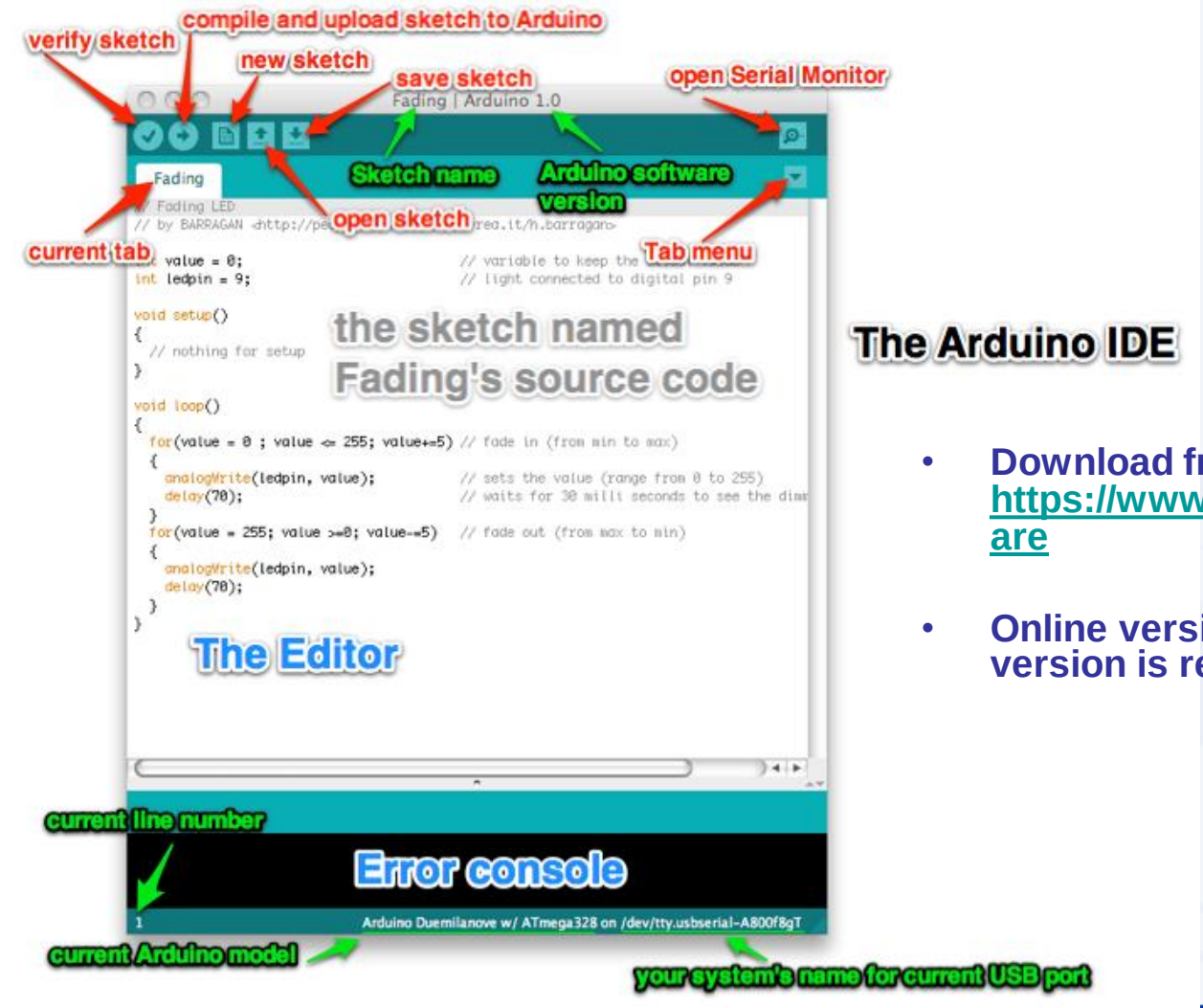




Stacked shields

- Shields provide an easy way to interface sensor and actuators with the Arduino – avoids having to wire them up manually
- Shields can be stacked (terms and conditions apply!)
- You can design your own shields

Arduino IDE

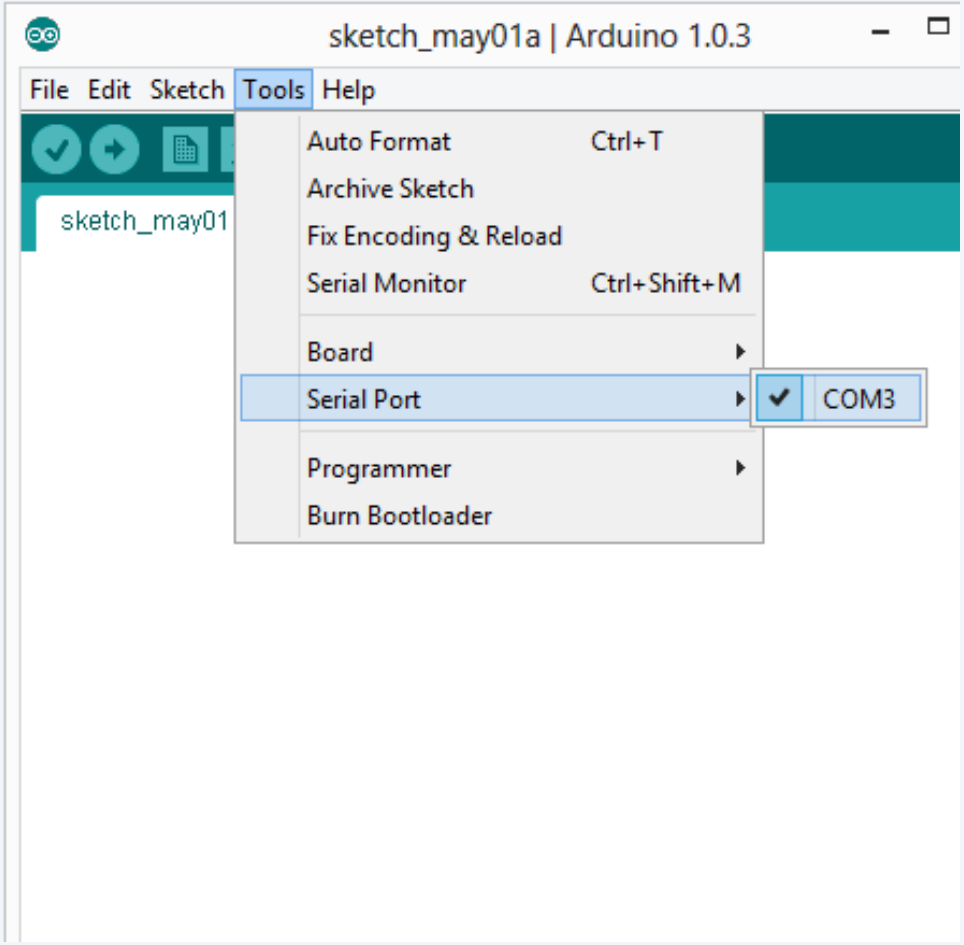


The Arduino IDE

- Download from https://www.arduino.cc/en/Main/Software
- Online version available (but local version is recommended)

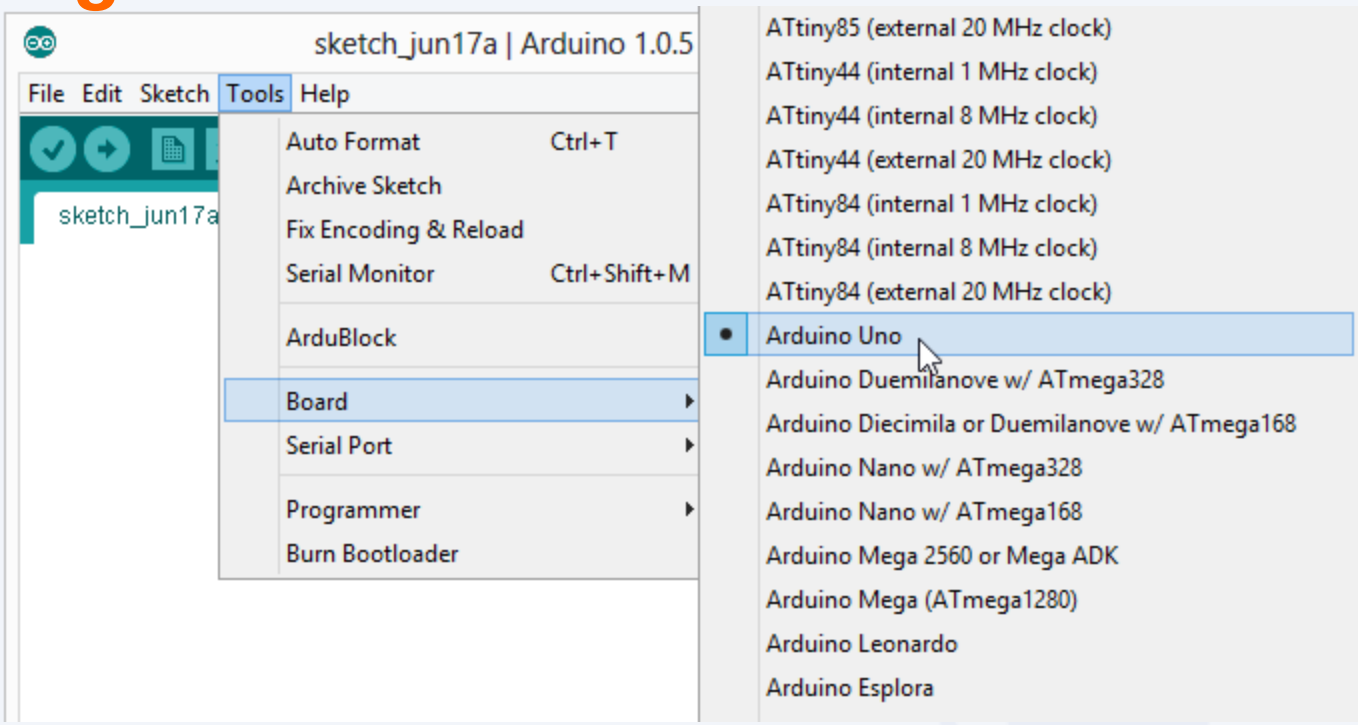
Image Courtesy : http://www.hacdc.org/summer-school-2013/

Settings: Tools → Serial Port



- Your computer communicates to the Arduino via a serial port → through a USB-Serial adapter
- Check to make sure that the drivers are properly installed
- The Serial Port wouldn't be 'COM1'.

Settings: Tools → Board



- **Next, double-check that the proper board is selected under the Tools→Board menu**

Comments

- Comments are for you – the programmer and your friends…or anyone else human/ Artificial Intelligence that might read your code
- Comments are *not run* on the Arduino board

```
// this is for single line comments
// it's good to put a description at the
// top and before anything 'tricky'
/* this is for multi-line comments
   Like this…
   And this….
*/
```

BareMinimum | Arduino 1.0.5

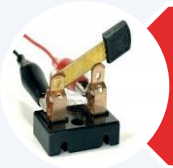
File Edit Sketch Tools Help

BareMinimum §

```
// Name of sketch
// Brief Description
// Date:
//

void setup()
{
  // put your setup code here, to run once:

}

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

}
```

comments

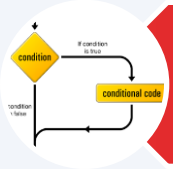
BIG 6 CONCEPTS



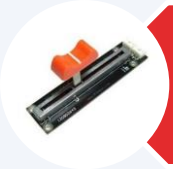
digitalWrite()

analogWrite()

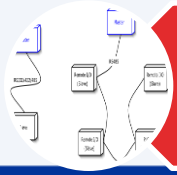
digitalRead()



if() statements



analogRead()



Serial communication

Let's Get Started..

Blinky

"Hello World" of Physical Computing

how do we implement this?

Digital Output

Three commands to know…

```
pinMode(pin, INPUT/OUTPUT);
ex: pinMode(13, OUTPUT);

digitalWrite(pin, HIGH/LOW);
ex: digitalWrite(13, HIGH);

delay(time_ms);
ex: delay(2500); // delay of 2.5 sec.

// NOTE: -> commands are CASE-sensitive
```

https://www.arduino.cc/en/Reference

Blinky

Blinky | Arduino 1.8.2

File Edit Sketch Tools Help

Blinky

```
/*
  Blinky
  Turns on an LED on for one second, then off for one second, repeatedly.
*/
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

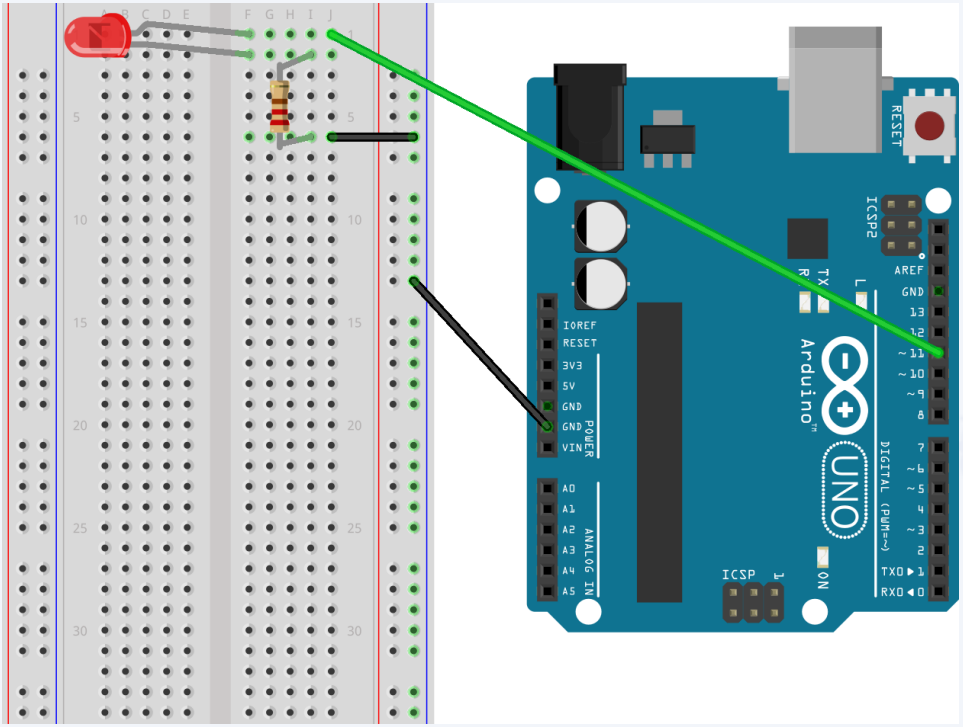
Done Saving.

1 Arduino/Genuino Uno on COM1

Type this code, click "Upload" and observe the LED close to pin 13

You have just completed your first Arduino program!

Blinking the LED



Move the green wire from the power pin to <u>pin 13</u> on the Arduino board without changing the program

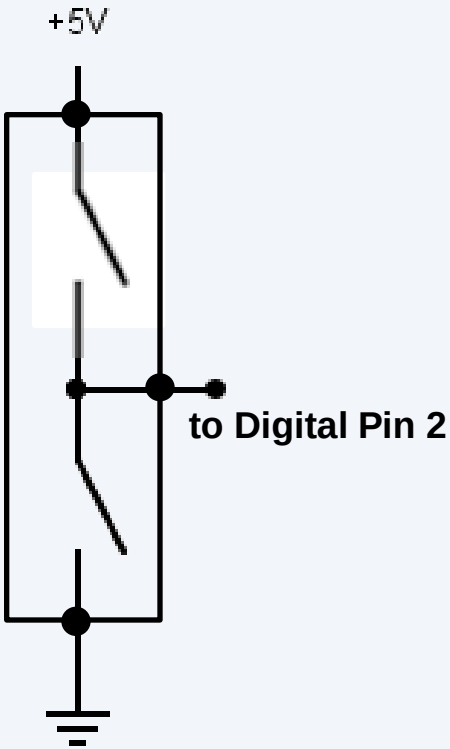
Try changing the connection from pin 13 to pin 11 (as shown in the image). How should your program be modified?

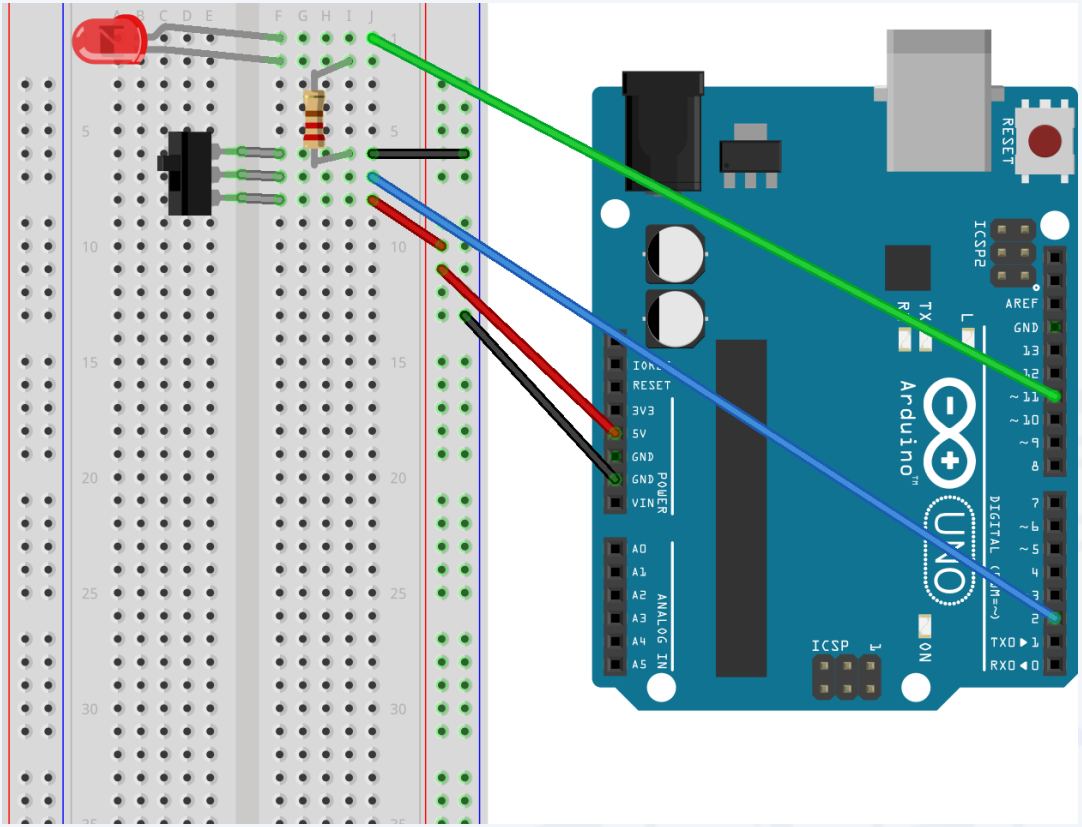
Digital Input

Digital Sensors

- Digital sensors are (more) straight forward (than Analog)
- No matter what the sensor there are only two settings: On and Off
- Signal is always either HIGH (On) or LOW (Off)
- Voltage signal for HIGH will be 5V (more or less) on Arduino Uno. Other Arduinos could use different voltages!
- Voltage signal for LOW will be 0V on most systems

Digital Input – Switch





Digital Input

- Connect digital input to your Arduino using Pins # 0 – 13 (Avoid pins # 0 & 1 though as they are used for *Serial* later, and pin #11 and 13 as we are already using it)

- Digital Input needs a `pinMode` command:

```
pinMode (pinNumber, INPUT);
```

Make sure to use ALL CAPS for INPUT

- To get a digital reading:

```
int buttonState = digitalRead (pinNumber);
```

- Digital Input values are only HIGH (On) or LOW (Off)

We set it equal to the function digitalRead(pushButton)

The function digitalRead() will return the value 1 or 0, depending on whether the button is being pressed or not being pressed.

We declare a variable as an integer.

```
int buttonState = digitalRead(pushButton);
```

We name it buttonState

The value 1 or 0 will be saved in the variable buttonState.

Recall that the pushButton variable stores the number 2

http://opensourcehardwarejunkies.com/tutorial-03-digitalread-and-serial-port-communication/

Programming: Conditional Statements

if()

```
void loop()
{
      int buttonState = digitalRead(2);
      if(buttonState == HIGH)
      {     // do something
      }
      else
      {  // do something else
      }
}
```

Exercise 1

Modify your blinky program such that it blinks only when the switch is turned ON

Hint :

In `setup(), pinMode(2, INPUT);` should be inserted

Place the code for blinking the LED below `// do something` in the previous slide

Exercise 1 Solution

Exercise_1 | Arduino 1.8.2

File Edit Sketch Tools Help

Exercise_1

```
void setup()
{
  pinMode(11, OUTPUT); // initialize digital pin 11 as an output.
  pinMode(2, INPUT); // initialize digital pin 2 as an input.
}

void loop()
{
  int buttonState = digitalRead(2); // read the button
  if(buttonState == HIGH)
  { // do something - here we just blink the LED
    digitalWrite(11, HIGH);
    delay(500);
    digitalWrite(11, LOW);
    delay(500);
  }
}
```

Done compiling.

```
Build options changed, rebuilding all
Sketch uses 1100 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for loca
```

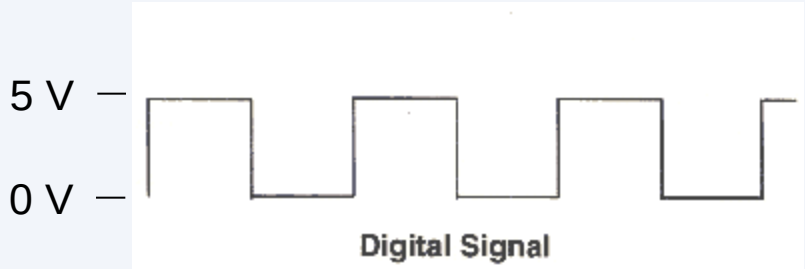
17 Arduino/Genuino Uno on COM1

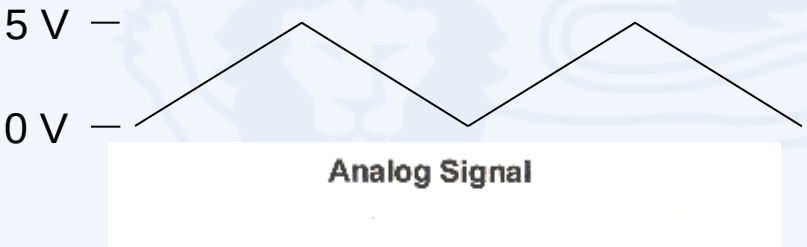
Analog Output

Analog vs. Digital

- **Arduinos are digital devices – ON or OFF. Also called discrete**

- **Analog signals are anything that can be a full range of values. Examples?**



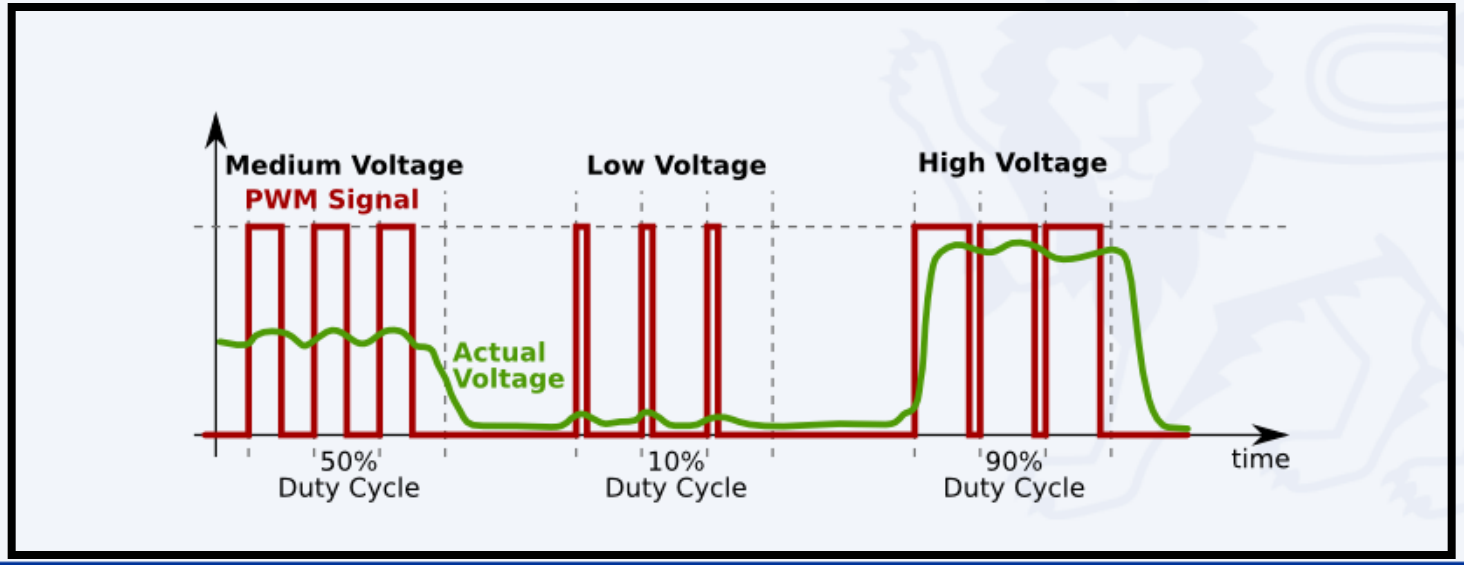


- **How do we generate the effect of *analog* using *digital*?**

Analog vs. Digital

- To create (mimic) an analog signal, the Arduino uses a technique called Pulse Width Modulation (PWM). By varying the duty cycle, we can mimic an "average" analog voltage

Pulse Width Modulation (PWM)



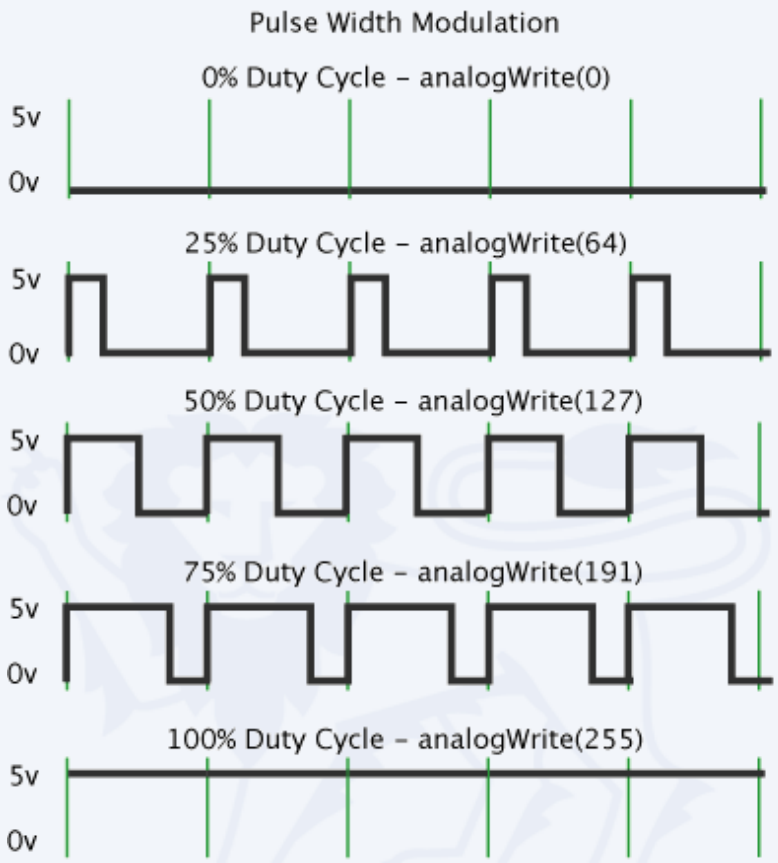
analogWrite()

```
analogWrite(pin, val);
```

`pin` – refers to the OUTPUT pin (limited to pins 3, 5, 6, 9, 10, 11.) – denoted by a ~ symbol

`val` – 8 bit value (0 – 255).

0 => 0V | 255 => 5V



Exercise 2

- **Create a program such that the LED brightness gradually increases from 0 to 255, and then goes abruptly to 0**

Hints :

- Use pin 11. If you are already having the LED connected to pin 11, you need not change any connection. Why can't you use pin 13?
- You will have to use a **for** loop. Lookup **for** in https://www.arduino.cc/en/Reference
- The delay should be around 5-10 milliseconds
- Can you modify your program to decrease the brightness gradually from 255 to 0 instead of an abrupt change?

Exercise 2 Solution

Exercise_2 | Arduino 1.8.2

File Edit Sketch Tools Help

Exercise_2

```
void setup()
{
  pinMode(11, OUTPUT);
}

void loop()
{
  for(int i=0; i<=255; i++) //increase gradually
  {
    analogWrite(11, i);
    delay(10);
  }
  for(int i=255; i>=0; i--) //decrease gradually
  {
    analogWrite(11, i);
    delay(10);
  }
}
```

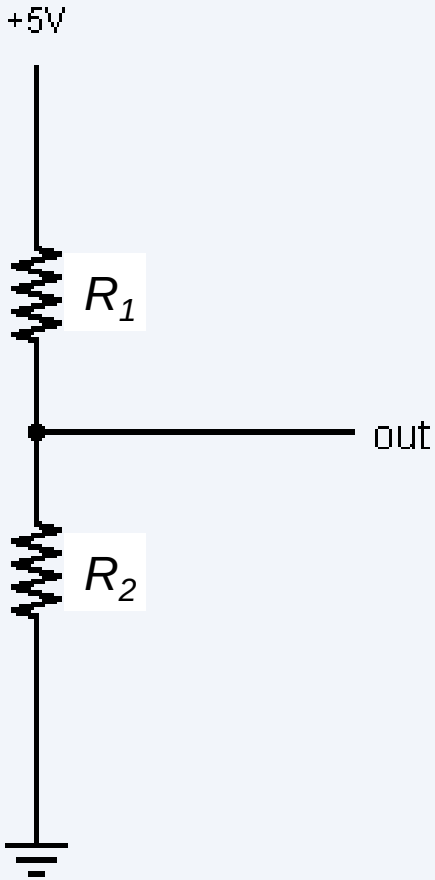
Done compiling.

```
Build options changed, rebuilding all
Sketch uses 1130 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for loca
```

18 Arduino/Genuino Uno on COM1

Analog Input

Voltage Divider

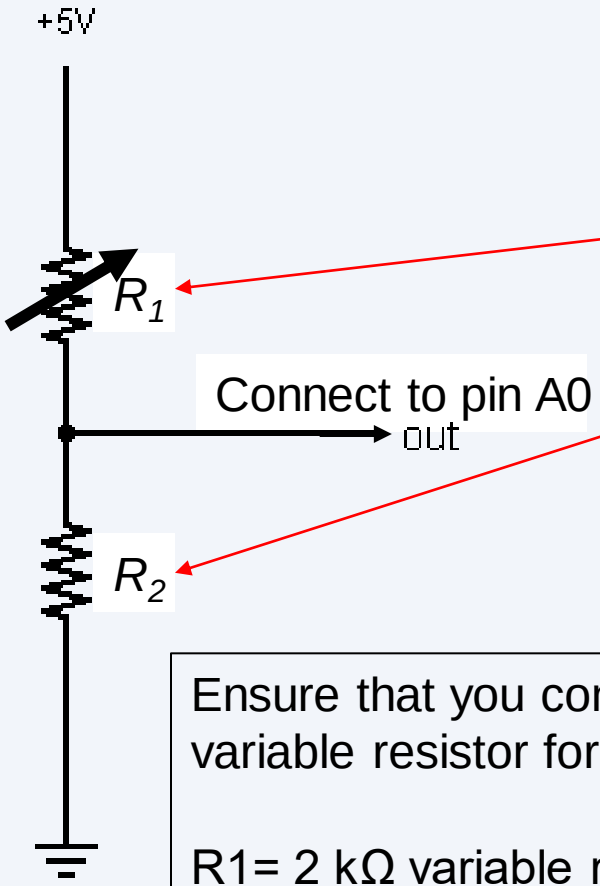


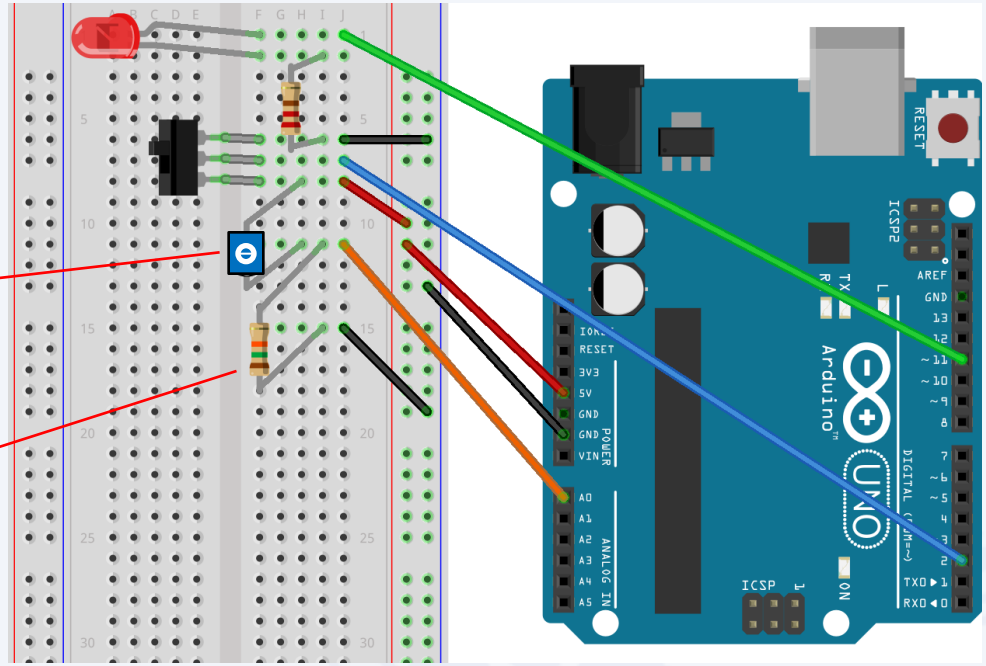
$$V_{R1} = V_{CC} \cdot \left(\frac{R_1}{\mathrm{R}_{Total}} \right)$$

$$V_{R2} = V_{CC} \cdot \left(\frac{R_2}{\mathrm{R}_{Total}} \right)$$

$$R_{Total} = R_1 + R_2$$

Variable Resistor

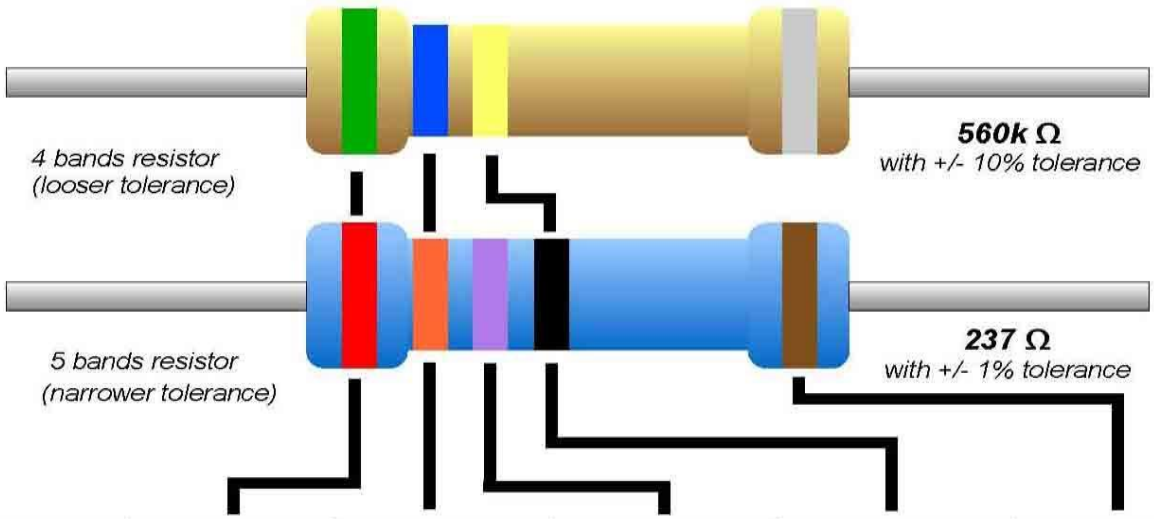




Ensure that you connect the middle and either of the end pins of the variable resistor for R1

R1= 2 kΩ variable resistor
R2 = 1 kΩ resistor

Resistor Color Code



Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	x 1 Ω	
Brown	1	1	1	x 10 Ω	+/- 1%
Red	2	2	2	x 100 Ω	+/- 2%
Orange	3	3	3	x 1K Ω	
Yellow	4	4	4	x 10K Ω	
Green	5	5	5	x 100K Ω	+/- .5%
Blue	6	6	6	x 1M Ω	+/- .25%
Violet	7	7	7	x 10M Ω	+/- .1%
Grey	8	8	8		+/- .05%
White	9	9	9		
Gold				x .1 Ω	+/- 5%
Silver				x .01 Ω	+/- 10%

analogRead()

Arduino uses a 10-bit A/D Converter:

this means that you get input values from 0 to 1023

0 V → 0

5 V → 1023

Ex:

```
int sensorValue = analogRead(A0);
```

Exercise 3

- **Modify your blinky program such that it blinks faster when the variable resistor value is higher, and slower when it is lower.**

Hint :

Check whether the sensorValue is greater than say, 512 (approx. half of the maximum, which is 1023) using an if condition

Exercise 3 Solution

Exercise_3.ino | Arduino 1.8.2

File Edit Sketch Tools Help

Exercise_3.ino

```
void setup()
{
  pinMode(11, OUTPUT); // initialize digital pin 11 as an output.
}

void loop()
{
  int delayVal;
  int lightValue = analogRead(0); // read the analog value
  if(lightValue > 512)
  {
    delayVal = 500;
  }
  else
  {
    delayVal = 100;
  }
  digitalWrite(11, HIGH);
  delay(delayVal);
  digitalWrite(11, LOW);
  delay(delayVal);
}
```

Done compiling.

```
Sketch uses 1006 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for loca
```

22 Arduino/Genuino Uno on COM1

Serial Communication

Using Serial Communication

Method used to transfer data between two devices.

Data passes between the computer and Arduino through the USB cable. Data is transmitted as zeros (‘0’) and ones (‘1’) sequentially.

Arduino dedicates Digital I/O pin # 0 to receiving and Digital I/O pin #1 to transmit

Serial Monitor & analogRead()

sketch_apr02a | Arduino 1.0.3

File Edit Sketch Tools Help

sketch_apr02a §

```
// analogRead() & Serial.print()
//
//

int sensorValue = 0;
int sensorPin = A0;

void setup()
{
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop()
{
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(100);  // waits by about 0.1 sec
}
```

Opens up a Serial Terminal Window

COM3 (Arduino/Genuino Uno)

Send

□2□□□□□□□□□□□

Autoscroll | No line ending | 9600 baud

Initializes the Serial Communication

Set bit rate (baud rate)

Prints/sends data to serial

Should be set to be the same

Sending a Message

```
void setup()
{
  // initialize serial:
  Serial.begin(9600);
}
void loop ( )
{
  Serial.print("Hands on ") ;
  Serial.print("Learning ") ;
  Serial.println("is Fun!!!") ;
}
```

Homework Exercise

- **Modify your program such that**
- **When the switch is off**
  - The message OFF is repeatedly printed via Serial, and the system will not do anything else
- **When the switch is on**
  - The LED will gradually turn on or off at a speed which is directly proportional to the value of the variable resistor
  - The message ON is repeatedly printed via Serial
- **Use a Serial bit/baud rate of 115200 instead of 9600 for the above**

Thank You!!