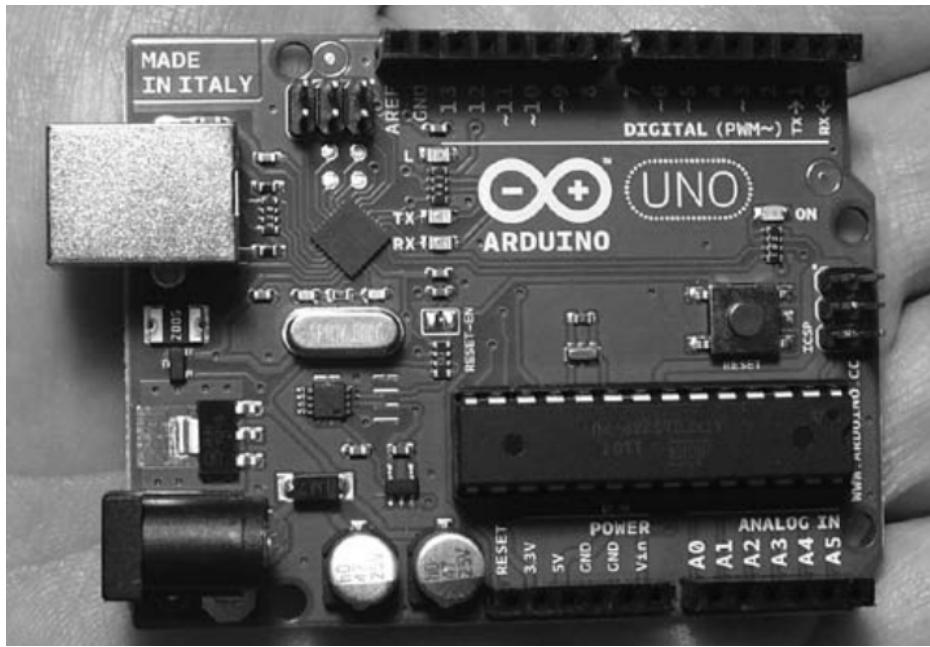


What is Arduino ?

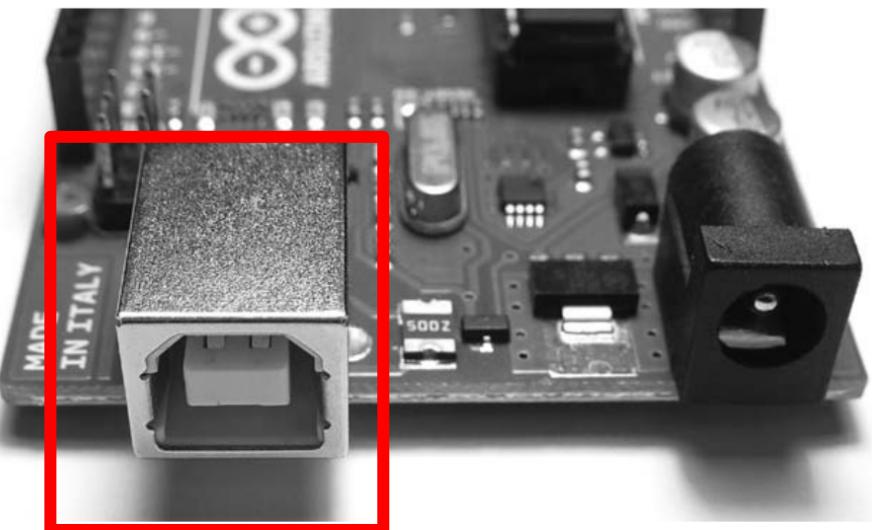


The Arduino is a tiny computer system that can be programmed with your instructions to interact with various forms of input and output

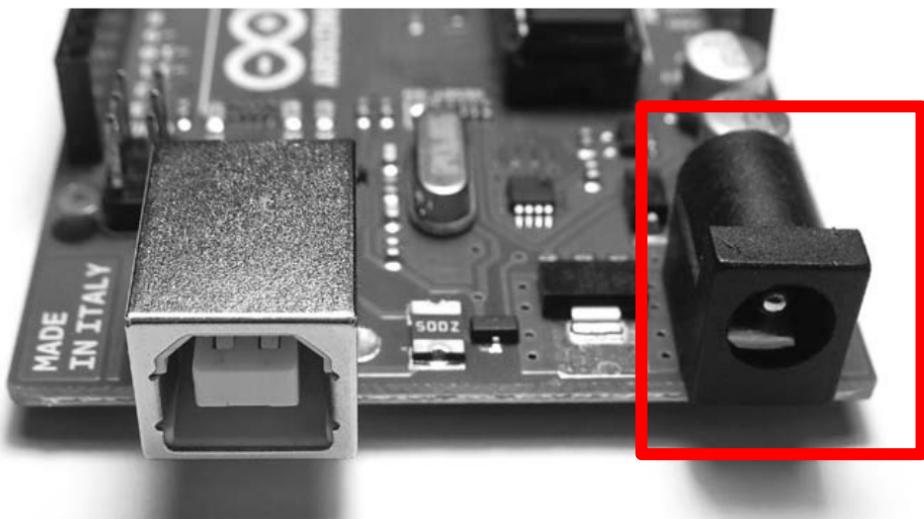
Quick Tour of Arduino

USB Connector

- Connects Arduino to Computer
- To Supply Power
- Upload Instructions to Arduino
- Send and Receive Data

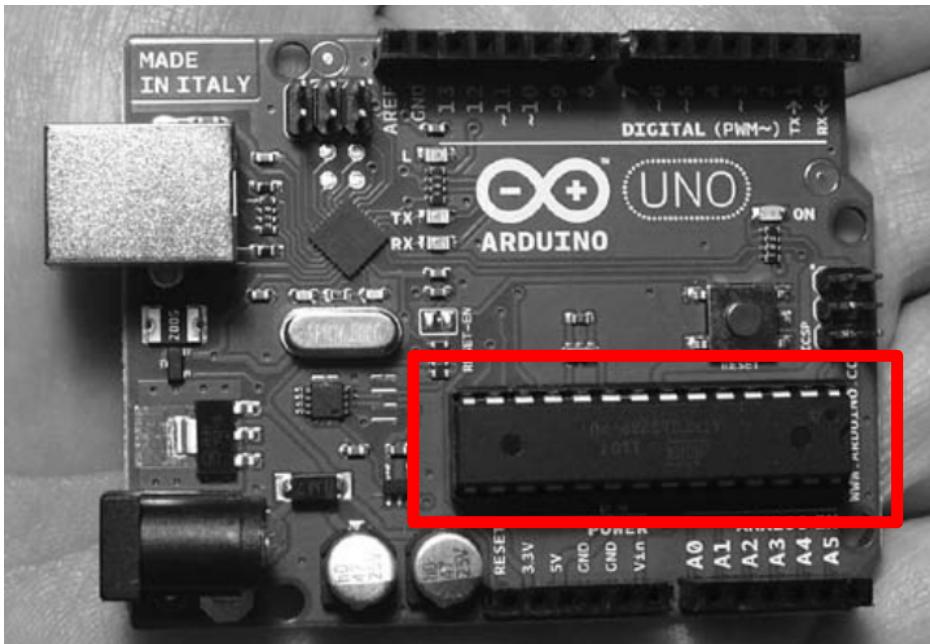


Quick Tour of Arduino



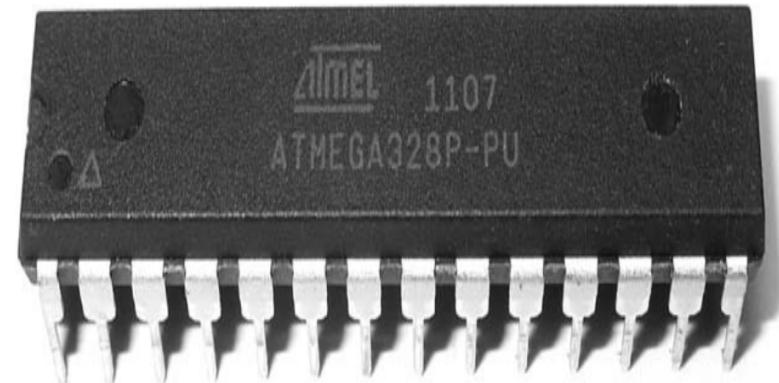
- Power the Arduino with DC 9-12V plug

Quick Tour Arduino



MicroController

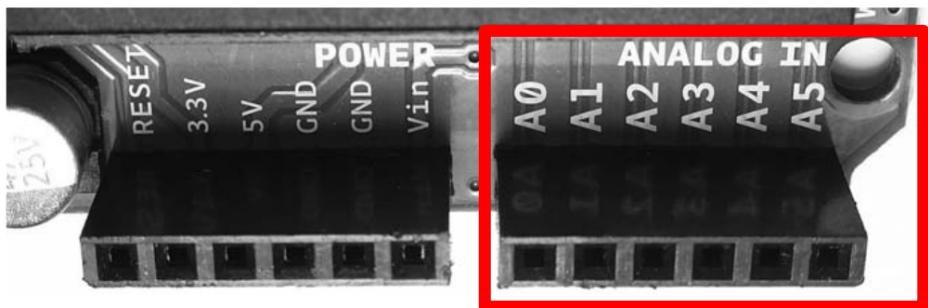
- Brain of the Arduino
- Executes Instructions
- Contains Memory



Quick Tour Arduino

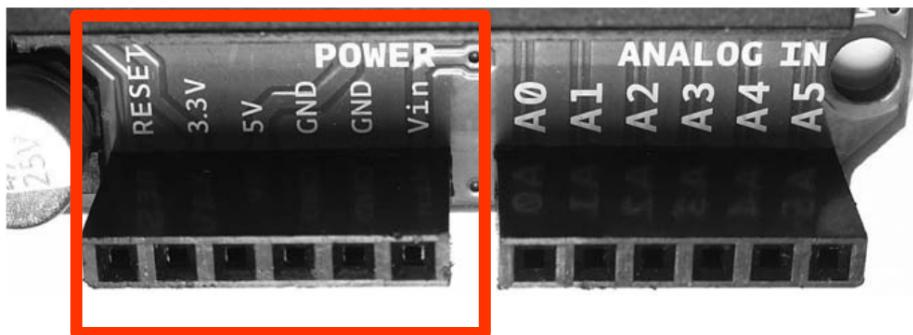
Analog Inputs

- Six Analog In Pins
- Measure Electrical Signals vary in voltage



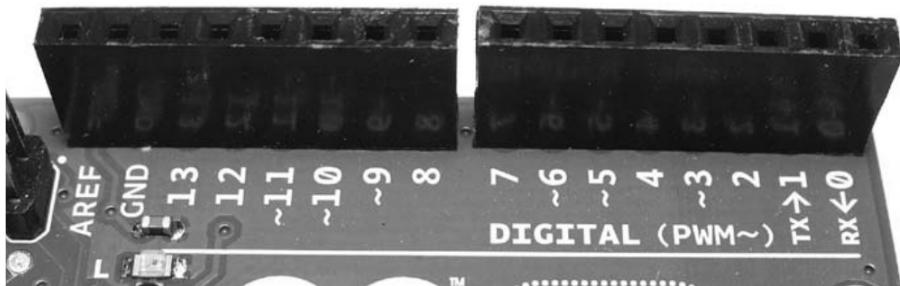
Quick Tour of Arduino

- Power Connections
- External RESET



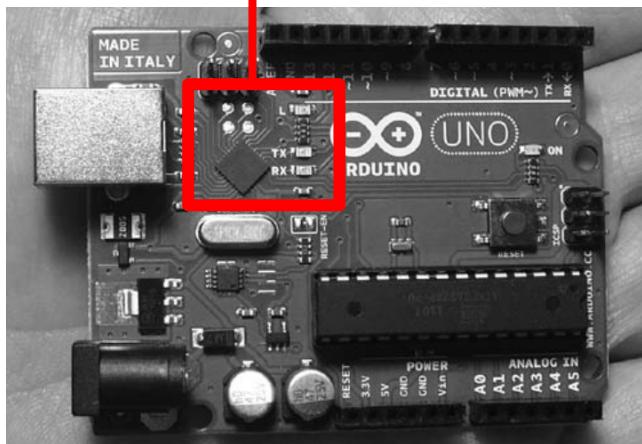
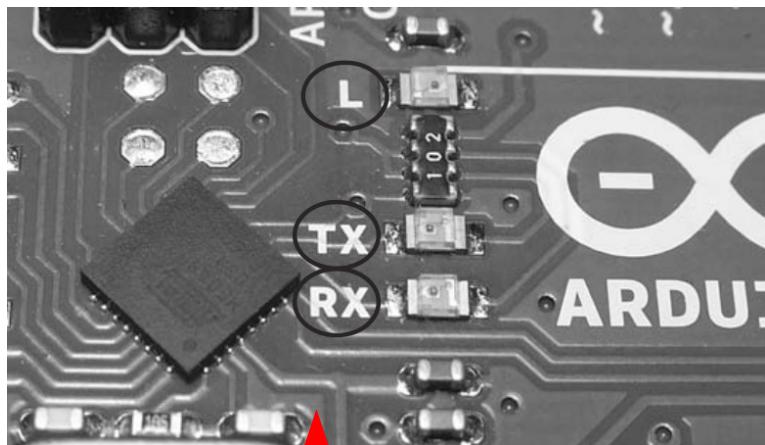
Quick Tour of Arduino

Digital Input and Output Pins



- Detect the presence of signal or Can Generate on Command
- ‘ ~ ‘ Labelled pins can generate varying electrical signal

Quick Tour of Arduino



On Board Led's

- TX, RX light up when Data being Transmitted and Received
- L is connected to Digital Pin 13

Software - Download

Access the Online IDE

Arduino.cc/en/Main/Software

ARDUINO WEB EDITOR

Start coding online with the [Arduino Web Editor](#), save your sketches in the cloud, and always have the most up-to-date version of the IDE, including all the contributed libraries and support for new Arduino boards. The Arduino Web Editor is one of the [Arduino Create platform's tools](#).

[Try It Now](#) [Getting Started](#)

Download the Arduino IDE

ARDUINO 1.8.1

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

Windows Installer
Windows ZIP file for non admin install

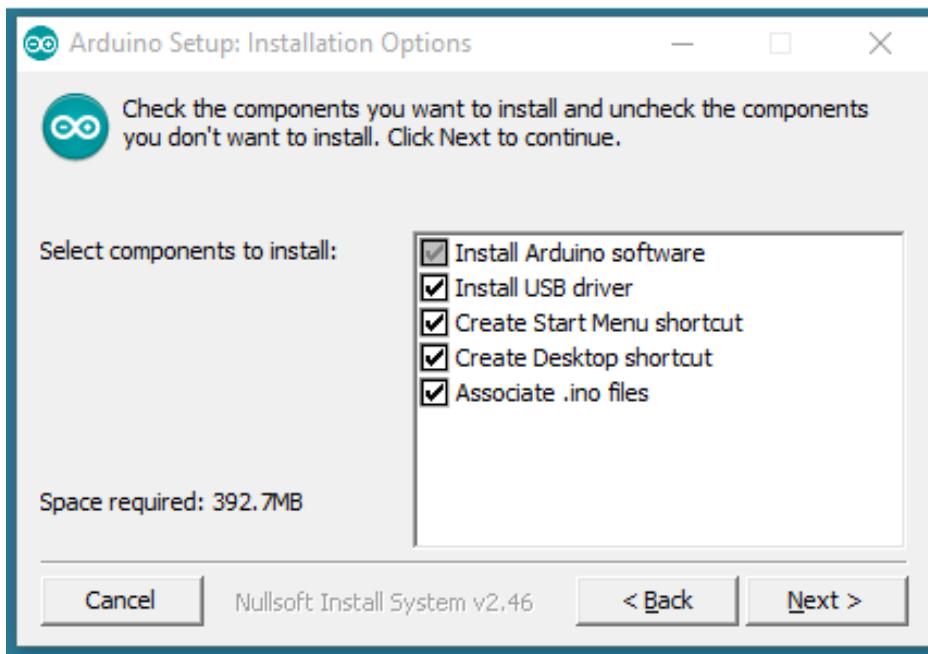
Windows app Get

Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

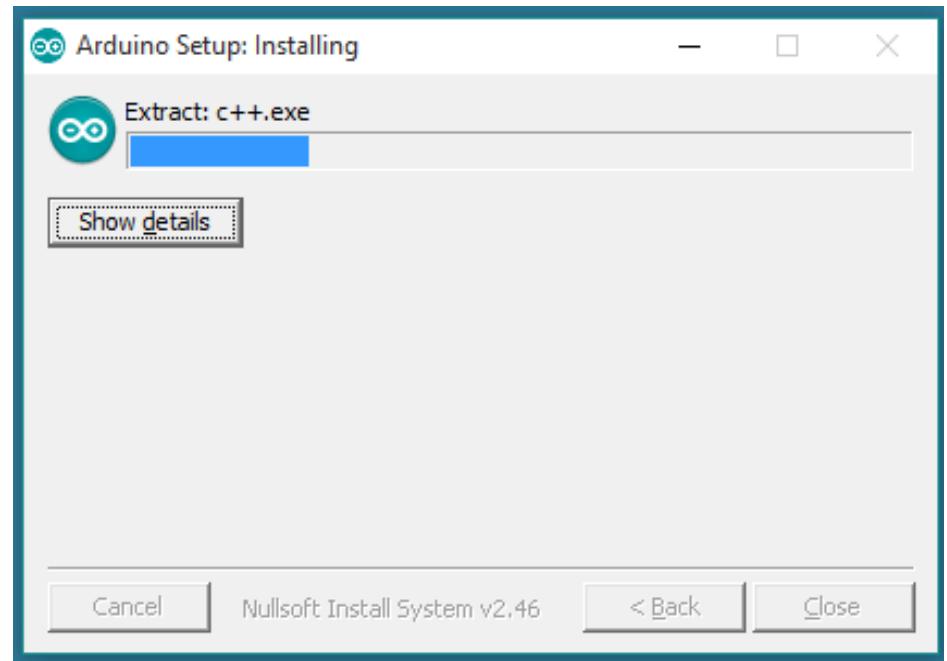
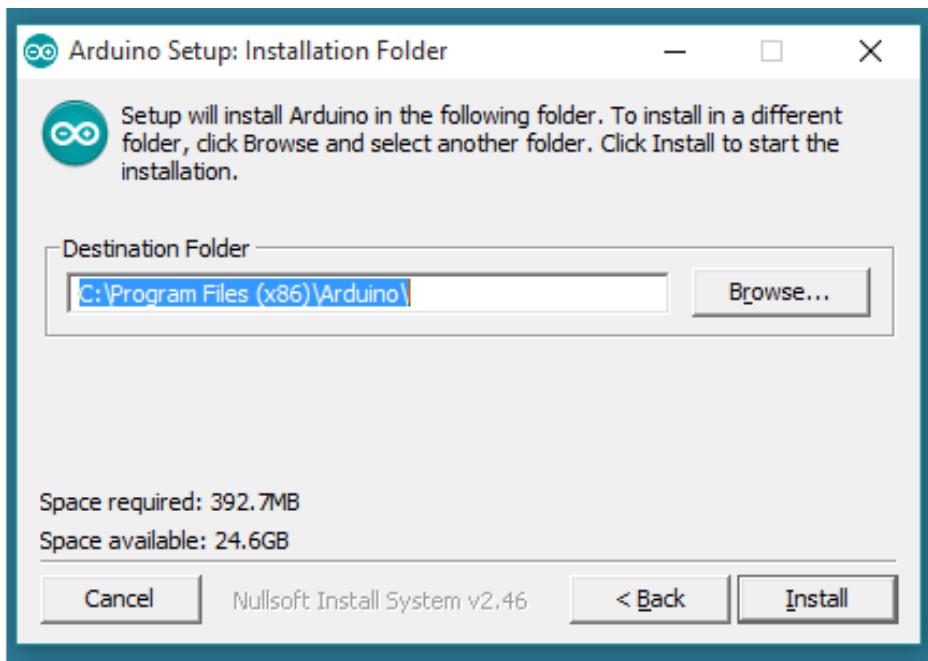
[Release Notes](#)
[Source Code](#)
[Checksums \(sha512\)](#)

Installation

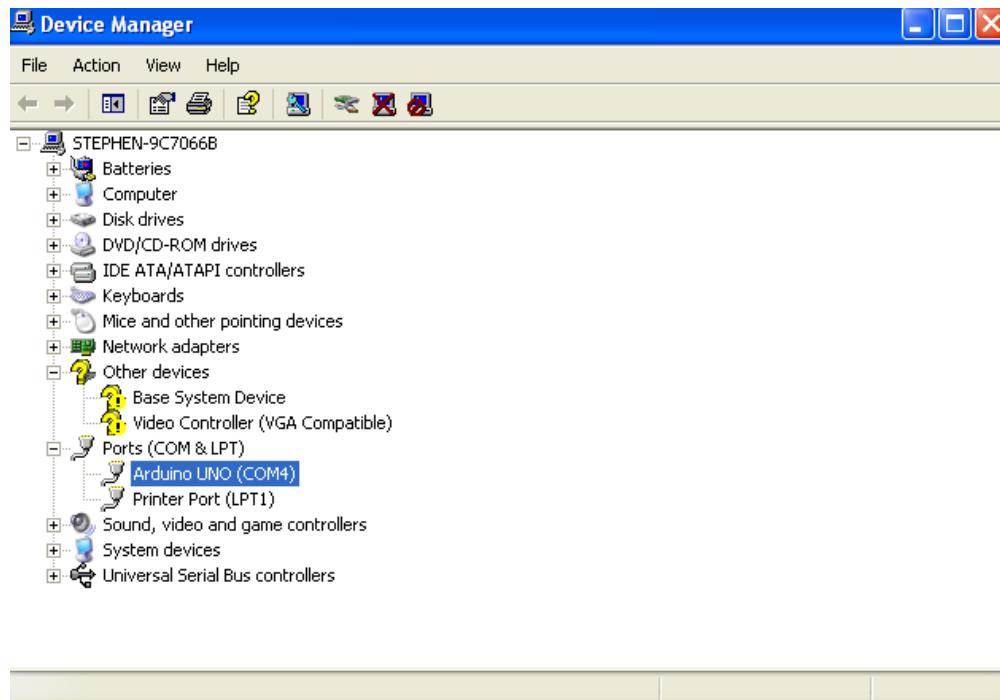
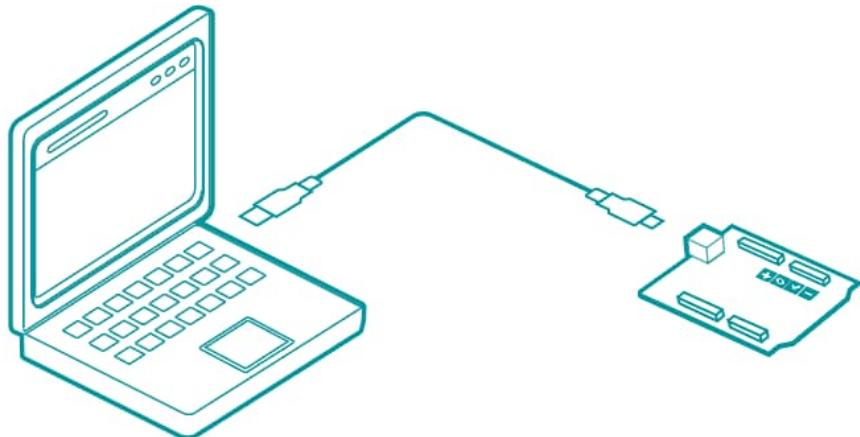


- Double click on the Downloaded Exe File
- Allow OS to install Drivers

installation

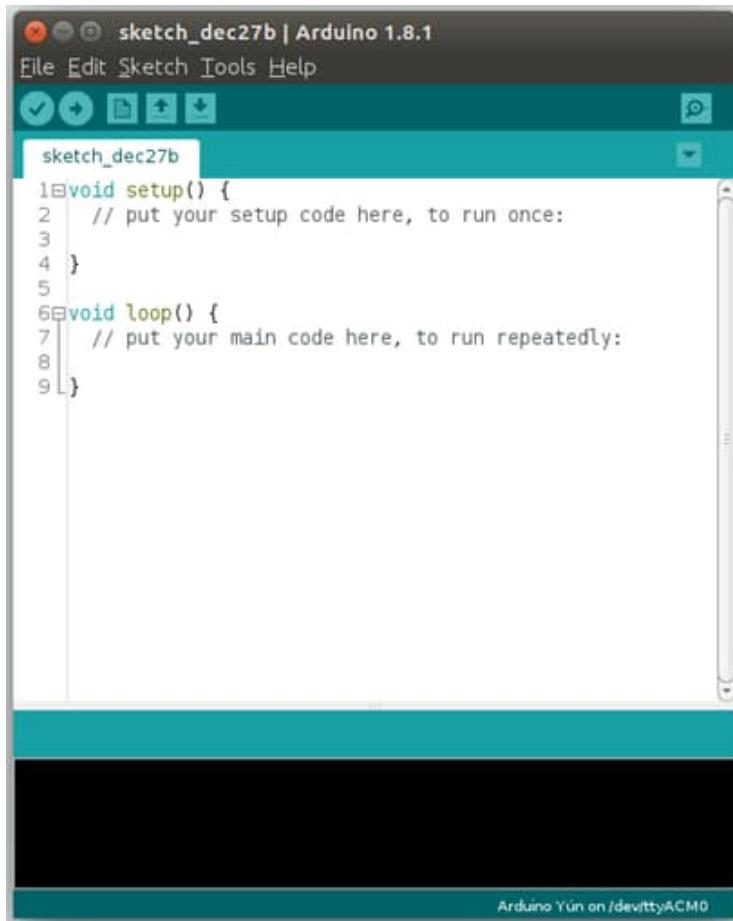


Installation - Driver



- Connect Arduino to Computer using USB Cable
- Open Device Manager
- Look under Ports
- Note Arduino UNO COM port

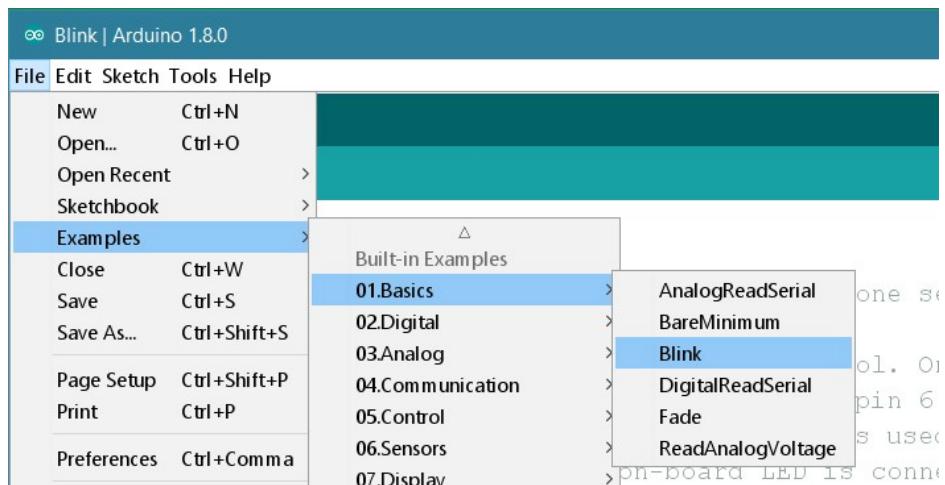
IDE – Test Run



- Double Click on the Arduino Icon in Desktop
- The Tab is known as Sketch
- It Contains the program that tells your board what to do

Test Run - Blink It

Go to File->Examples->Basics-> Blink and click on it



```
File Edit Sketch Tools Help
Blink
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.

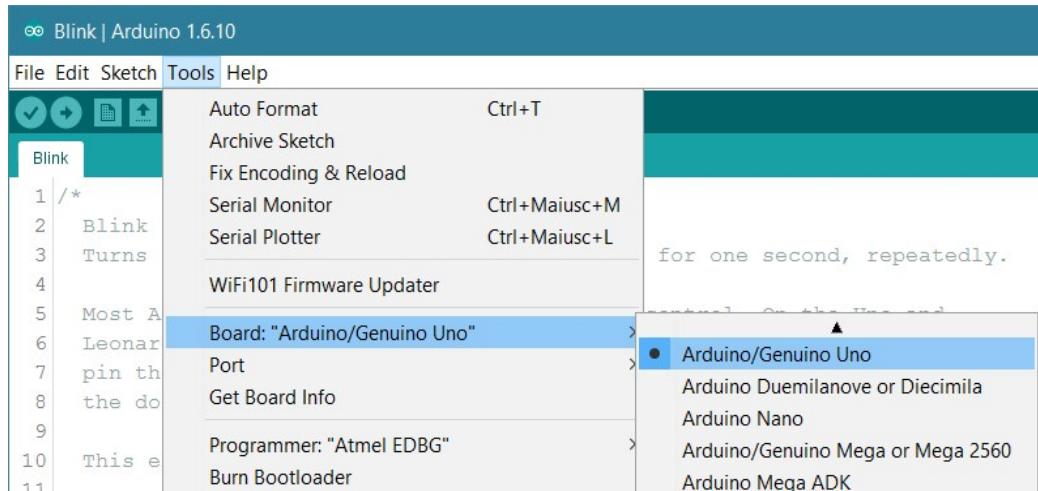
*/
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output:
  pinMode(led, OUTPUT);
}

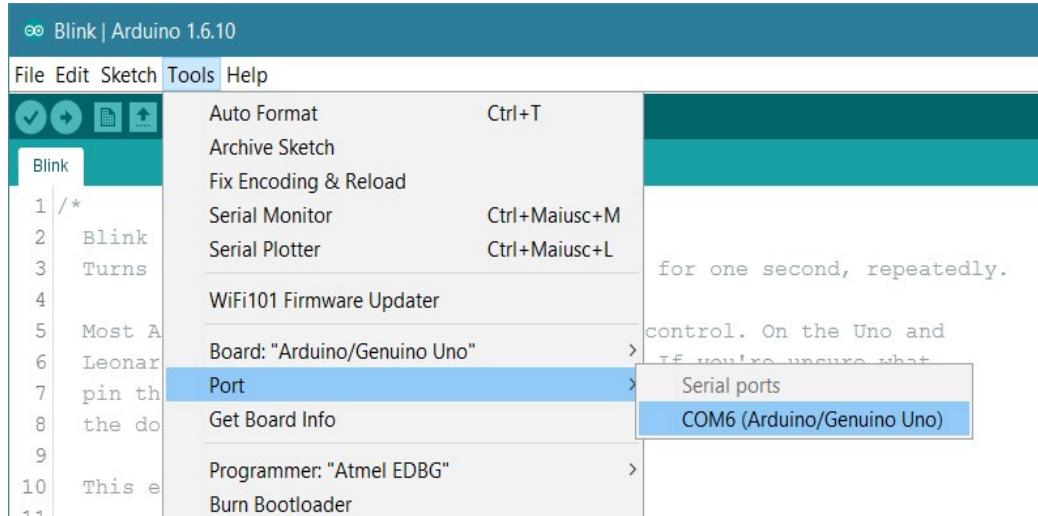
// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);               //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

The image shows the Arduino IDE with the "Blink" sketch selected. The code window displays the standard Blink example. The code initializes pin 13 as an output and then enters a loop where it alternates between HIGH and LOW states every second using the digitalWrite and delay functions.

Test Run - Blink It

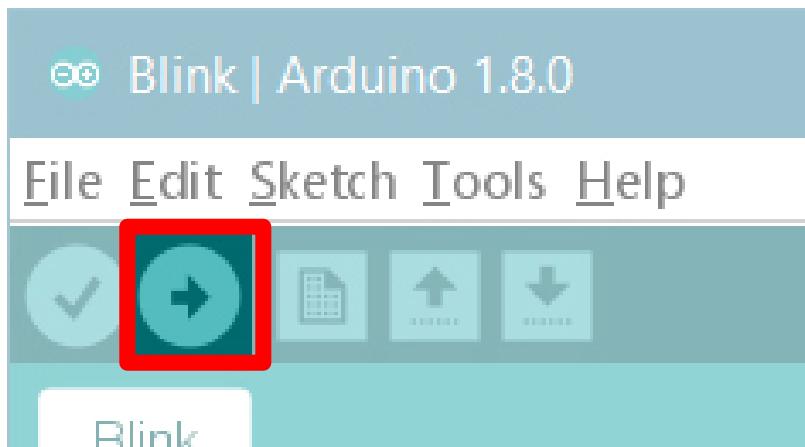


- Select Board type in Tools -> Board
- Select UNO



- Select COM port noted Previously in Tools -> Port

Test Run - Upload



- Click the upload button
- you should see the RX and TX leds on the board flashing
- After few seconds on board led blinks

Introduction to Arduino Programming

Structure

BareMinimum §

```
1 void setup() {  
2 }  
3  
4 void loop() {  
5 }  
6  
7 }  
8 }
```

- Every Program must contain these two functions
- Setup is preparation it is the first function to run and run only once
- Loop is execution it is executed continuously

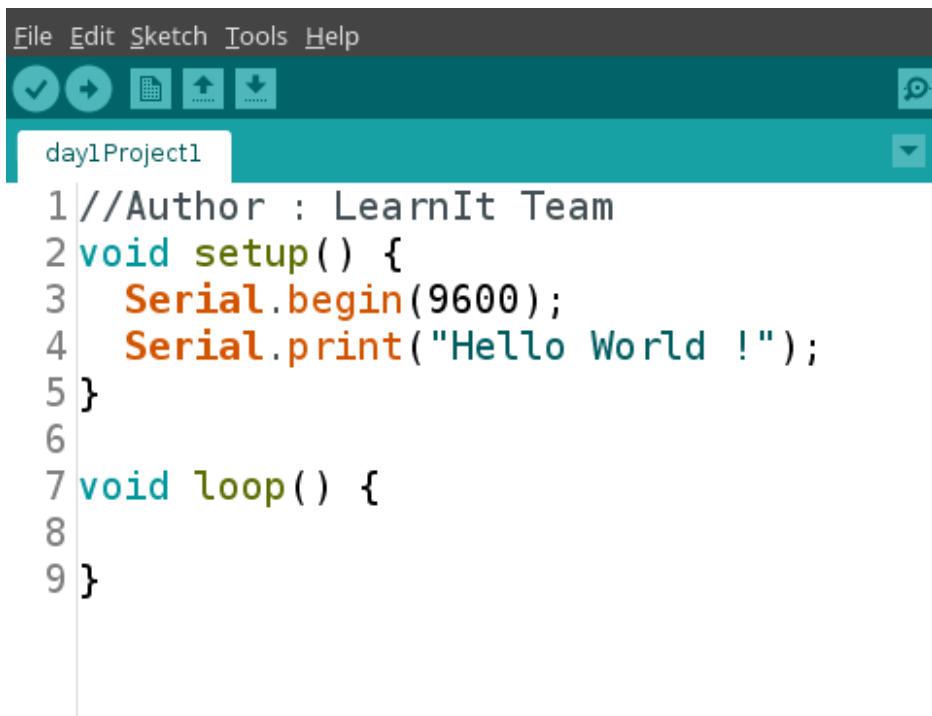
Fuctions

- Function is block of code that has a name and a block of statements when the function is called
- Setup() and loop() are built-in functions
 - we will talk about custom functions later

Basic Syntax

- Each Statement Ends with semicolon “;”
`int a = 13;`
- Curly braces always come in pairs; they are used to define the start and end of functions, loop, conditional statements
- `//` is used for single line comment
- `/* */` used to wrap multiline comment

Hello World

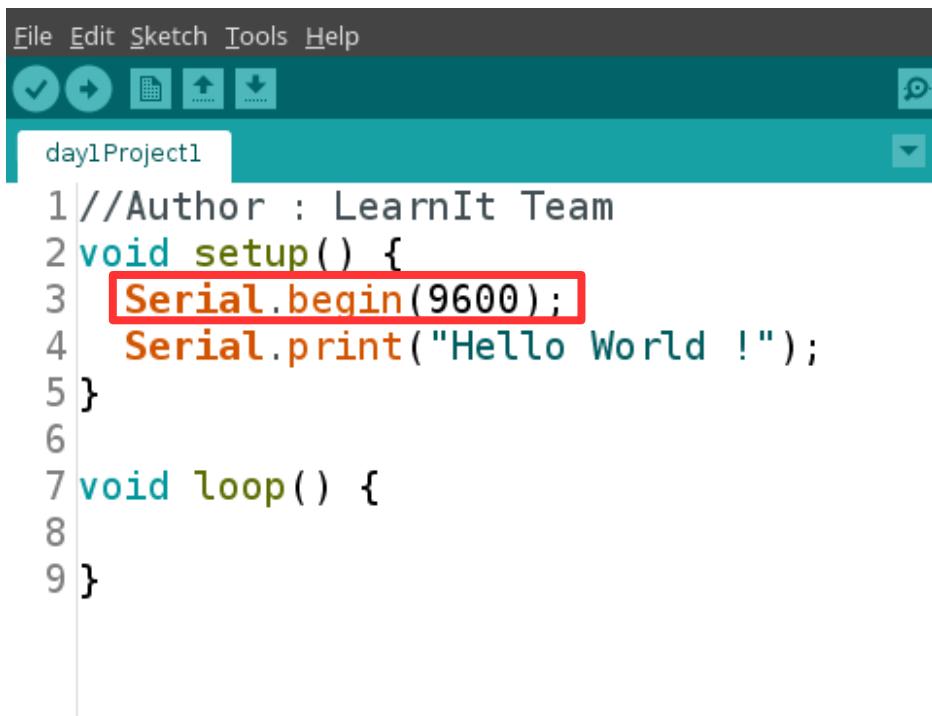


The image shows a screenshot of the Arduino IDE. The title bar says "File Edit Sketch Tools Help". Below the title bar are several icons: a checkmark, a circular arrow, a square, an upward arrow, and a downward arrow. The main window displays a code editor with the following text:

```
1 //Author : LearnIt Team
2 void setup() {
3   Serial.begin(9600);
4   Serial.print("Hello World !");
5 }
6
7 void loop() {
8
9 }
```

- USB is used to send and Receive Data Serially
- Serial Port / Com port talks with certain speed, called Baudrate

Hello World



The image shows a screenshot of the Arduino IDE. The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for upload, download, and other functions. The central workspace shows a sketch named "day1Project1". The code is as follows:

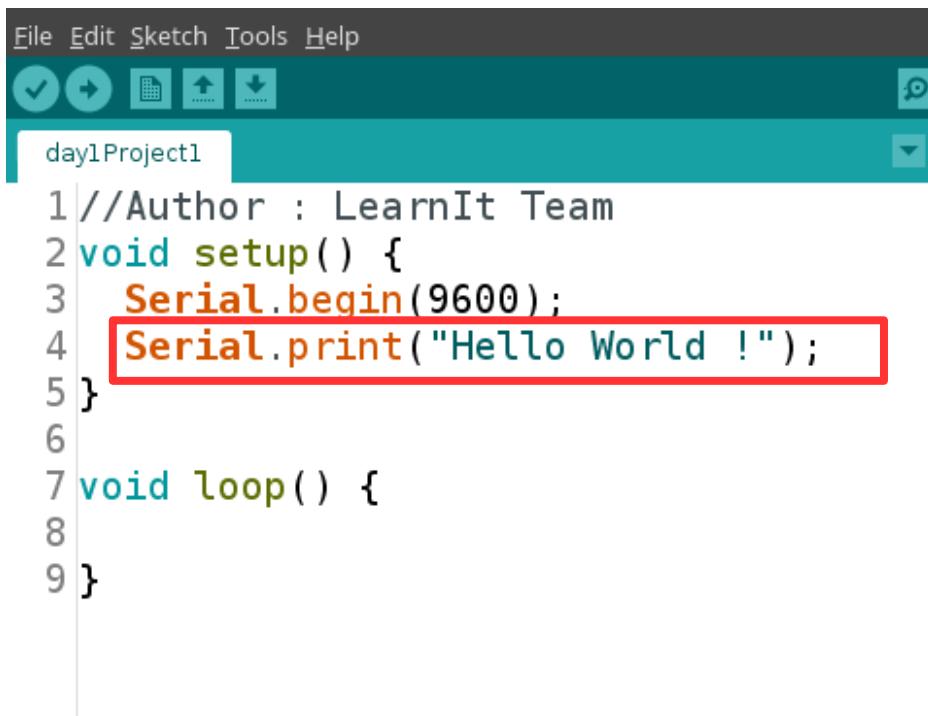
```
1 //Author : LearnIt Team
2 void setup() {
3     Serial.begin(9600);
4     Serial.print("Hello World !");
5 }
6
7 void loop() {
8
9 }
```

The line `Serial.begin(9600);` is highlighted with a red rectangle.

Serial.begin(9600) basically tells the Arduino to start talking over serial port at a speed of 9600 baud

Baudrate – Bits per second

Hello World



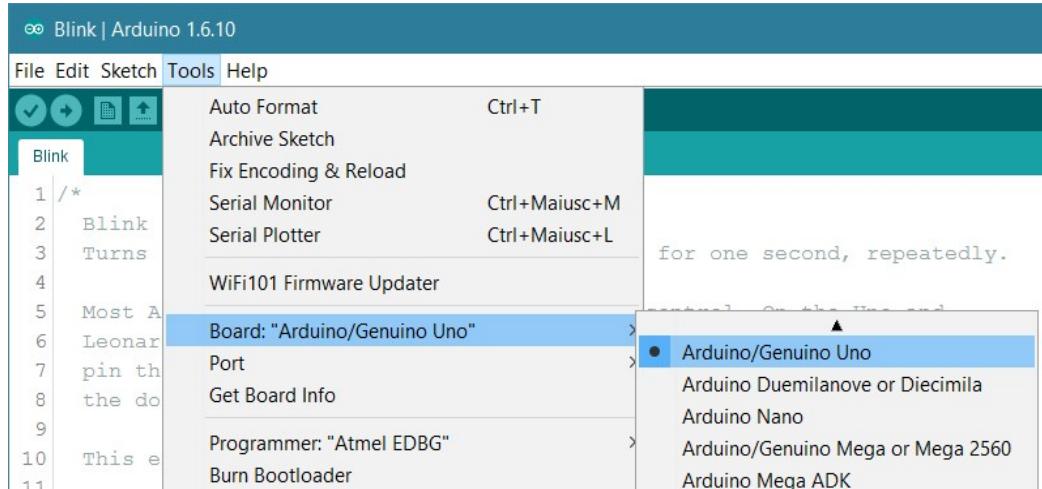
The image shows the Arduino IDE interface. The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for upload, download, and other functions. The central workspace shows a sketch named "day1Project1". The code is as follows:

```
1 //Author : LearnIt Team
2 void setup() {
3     Serial.begin(9600);
4     Serial.print("Hello World !");
5 }
6
7 void loop() {
8
9 }
```

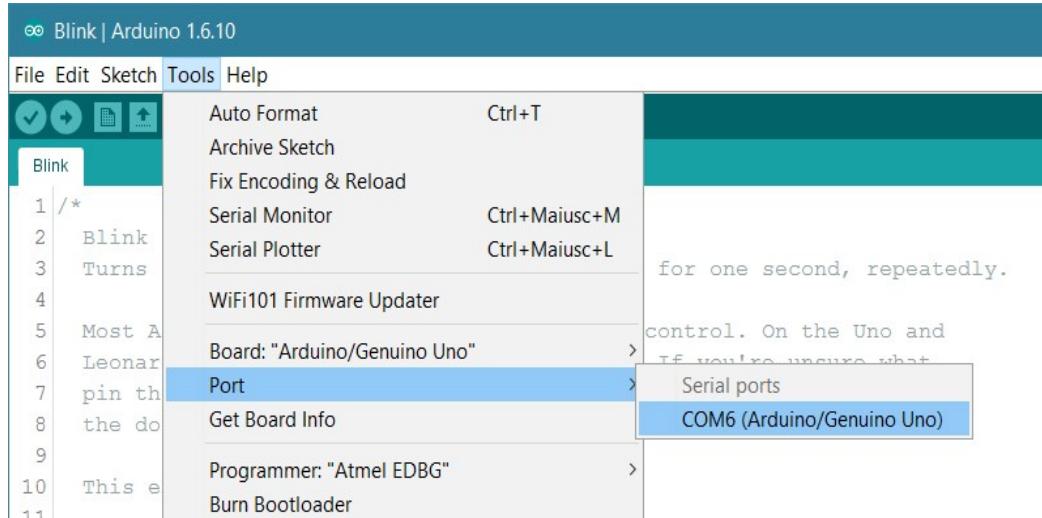
The line `Serial.print("Hello World !");` is highlighted with a red rectangular box.

- `Serial.print` tells the Arduino to send a text line to Serial port
- Notice the **Semicolon** `(;)` - every statement must end with semicolon

Select Port and Board

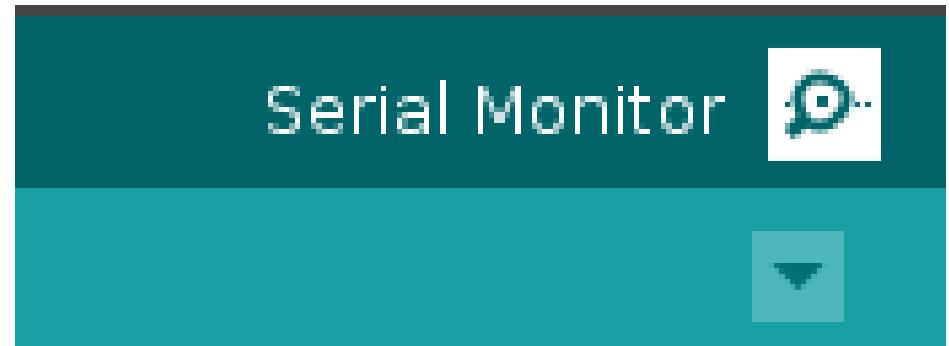
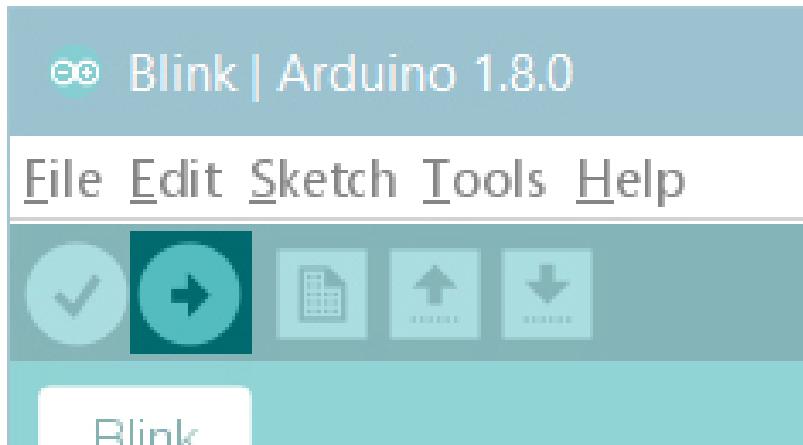


- Select Board type in Tools -> Board
- Select UNO



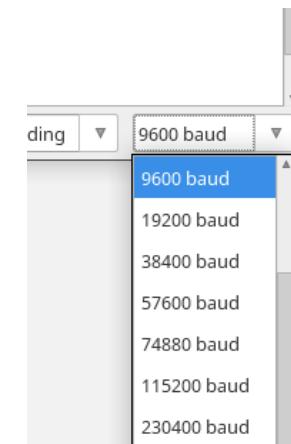
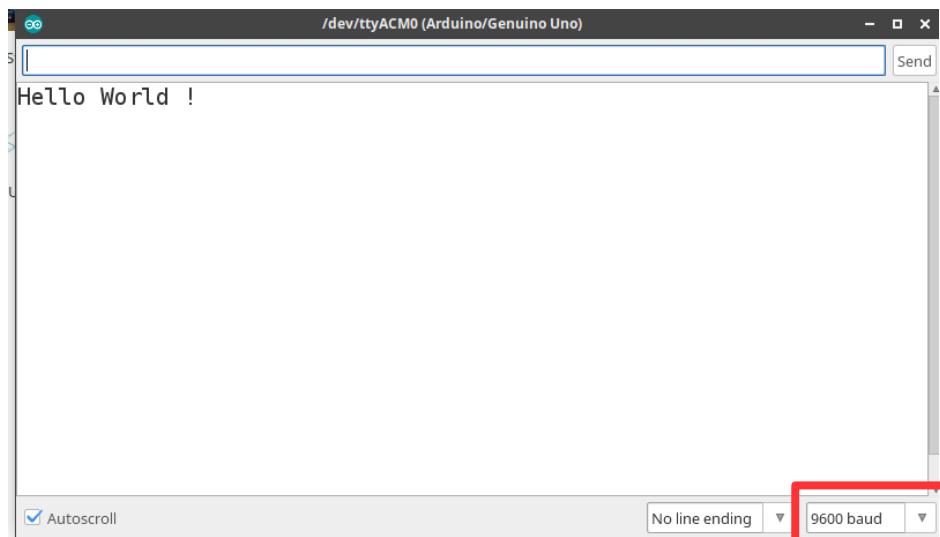
- Select COM port noted Previously in Tools -> Port

Upload & Open Serial Monitor



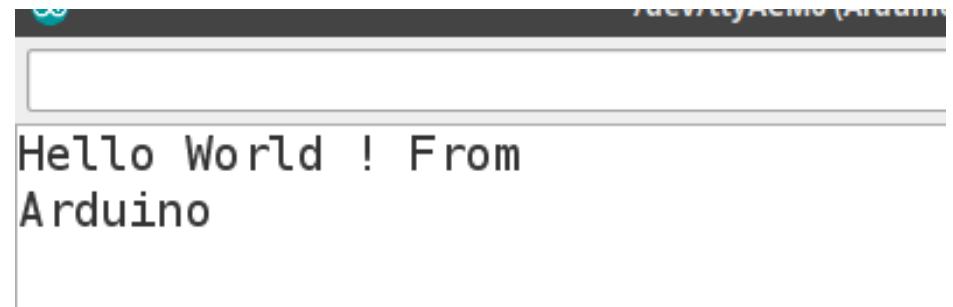
- Upload the Sketch
- Observe the tx & rx
blinking on Board
- Click on the
magnifying glass to
open Serial Monitor

Try with Differnt
Baudrates by changing
in Serial.begin and
Serial Monitor



Print vs println

```
day1Project2
1 //Author : LearnIt Team
2 void setup() {
3   Serial.begin(9600);
4   Serial.print("Hello World !");
5   Serial.println(" From");
6   Serial.print("Arduino");
7 }
8
9 void loop() {
10}
11}
```



“Arduino “ Starts in
new line But “From” is
on same ine

Setup vs loop

day1Project3

```
1 //Author : LearnIt Team
2 void setup() {
3   Serial.begin(9600);
4   Serial.print("Hello World !");
5   Serial.println(" From");
6   Serial.print("Arduino");
7 }
8
9 void loop() {
10  Serial.println("Looping");
11 }
```

```
Hello World ! From
ArduinoLooping
Looping
```

Statement in loop is
repeating

Delay

```
day1Project4
1 //Author : LearnIt Team
2 void setup() {
3   Serial.begin(9600);
4   Serial.print("Hello World !");
5   Serial.println(" From");
6   Serial.print("Arduino");
7 }
8
9 void loop() {
10   Serial.println("Looping");
11   delay(1000);
12 }
```

- Delay Pauses the program for the amount of time (in milliseconds) specified as parameter
- it brings all other activities to a halt

Variables

- A variable is a place to store a piece of data
- It has a name, a value, and a type

int a = 13;

- creates a variable whose name is **a**, whose value is **13**, and whose type is **int**

```
day1Project5
1 //Author : LearnIt Team
2 int a = 13;
3 void setup() {
4   Serial.begin(9600);
5   Serial.println("value of a is");
6   Serial.print(a);
7 }
8
9 void loop() {
10 }
```

Variable types & Rules

Type	Description
int	Integer max size is 2^8
float	Floating point value
char	Character
String	String

- consist of any letters (a to z and A to Z)
- contain the numbers 0 to 9, but may not start with a number, e.g. 3var is not allowed, but var3 is allowed
- Variables may not have the same names as Arduino language keywords, e.g. you can not have a variable named int
- Variables must have unique names i.e. you can not have two variables with the same name
- Variable names are case sensitive, so Count and count are two different variables
- Variables may not contain any special characters, except the underscore (_), e.g. top_score

Variable Scope

- scope is the lifetime and visibility a variable has in a program
 - Global : A variable with global scope is visible from its point of definition to the end of the file in which it is defined
 - Local : A variable with local scope extends from its point of definition to the end of the function block in which it is define

```
1 int gval; // any function will see this variable
2
3 void setup()
4 {
5     // ...
6 }
7
8 void loop()
9 {
10    int i; // "i" is only "visible" inside of "loop"
11    float f; // "f" is only "visible" inside of "loop"
12    // ...
13 }
```

Arithmetic operators

Symbol	Name	Example	Explanation
=	assignment	Led = 13	Store the value 13 at variable led
+	addition	A = B + C	Add the variable B,C and assign the result to C
-	subtraction	A = B - C	Subtract C from B and assign the result to A
*	multiplication	A = B*C	Multiply B and C and assign the result to A
/	division	A = B/C	Divide B with C and assign the result to A
%	modulo	A = B%C	Calculate the Remainder of division B/C and assign it to the A

```
1 // Author : LearnIt team
2 int A = 12;
3 int B = 5;
4
5 void setup() {
6     int C;
7     Serial.begin(9600);
8     Serial.print(" A value is -> ");
9     Serial.print(A);
10    Serial.print(" B value is -> ");
11    Serial.print(B);
12    Serial.print(" C value is -> ");
13    Serial.println(C);
14
15    int sum = A + B;
16    Serial.print("A+B -> ");
17    Serial.println(sum);
18
19    int diff = A - B;
20    Serial.print("A-B -> ");
21    Serial.println(diff);
22
23    int mul = A * B;
24    Serial.print("A*B -> ");
25    Serial.println(mul);
26
27    int divi = A / B;
28    Serial.print("A/B -> ");
29    Serial.println(divi);
30
31    int modulo = A % B;
32    Serial.print("A%B -> ");
33    Serial.println(modulo);
34 }
35
36 void loop() {
37
38 }
```

```
A value is -> 12 B value is -> 5 C value is -> 0  
A+B -> 17  
A-B -> 7  
A*B -> 60  
A/B -> 2  
A%B -> 2
```

Comparison Operators

Result of statement with comparison operator is either TRUE or FALSE

Symbol	Name	Example	Explanation
<code>==</code>	Is equal to	<code>X == Y</code>	X is equal to y
<code>!=</code>	Not equal to	<code>X != Y</code>	X is not equal to Y
<code><</code>	Less than	<code>X < Y</code>	X less than Y
<code>></code>	Greater than	<code>X > Y</code>	X greater than Y
<code><=</code>	Less than or equal to	<code>X <= Y</code>	X less than or equal to Y
<code>>=</code>	Greater than or equal to	<code>X >= Y</code>	X greater than or equal to Y

`0 < x < 1` is invalid

day1Project7

```
1 // Author : LearnIt team
2 int A=12,B=13,C=12;
3 void setup() {
4     Serial.begin(9600);
5     Serial.print("A==A    is ");
6     Serial.println(A==A);
7     Serial.print("A==B    is ");
8     Serial.println(A==B);
9
10    Serial.print("A!=A    is ");
11    Serial.println(A!=A);
12    Serial.print("A!=B    is ");
13    Serial.println(A!=B);
14
15    Serial.print("A<B    is ");
16    Serial.println(A<B);
17    Serial.print("C>B    is ");
18    Serial.println(C>B);
19
20    Serial.print("A<=B    is ");
21    Serial.println(A<=B);
22
23    Serial.print("B<=B    is ");
24    Serial.println(B<=B);
25
26    Serial.print("A=B    is ");
27    Serial.println(A=B);
28 }
29 void loop() {
30 }
31
32 }
```

A==A	is 1
A==B	is 0
A!=A	is 0
A!=B	is 1
A<B	is 1
C>B	is 0
A<=B	is 1
B<=B	is 1
A=B	is 13

Boolean Operators

Symbol	Name	Example	Explanation
<code>&&</code>	Logical AND	<code>X>Y && Y<Z</code>	True only if both conditions are true
<code> </code>	Logical OR	<code>X>Y Y< Z</code>	True if any one of condition is true
<code>!</code>	NOT	<code>!X</code>	Invert the boolean value X

Lets start Tinkering