Arduino Programming

(For Beginners)

- Introduction to Arduino
- UNO Overview
- Programming Basics
- Arduino Libraires

Siji Sunny

siji@melabs.in

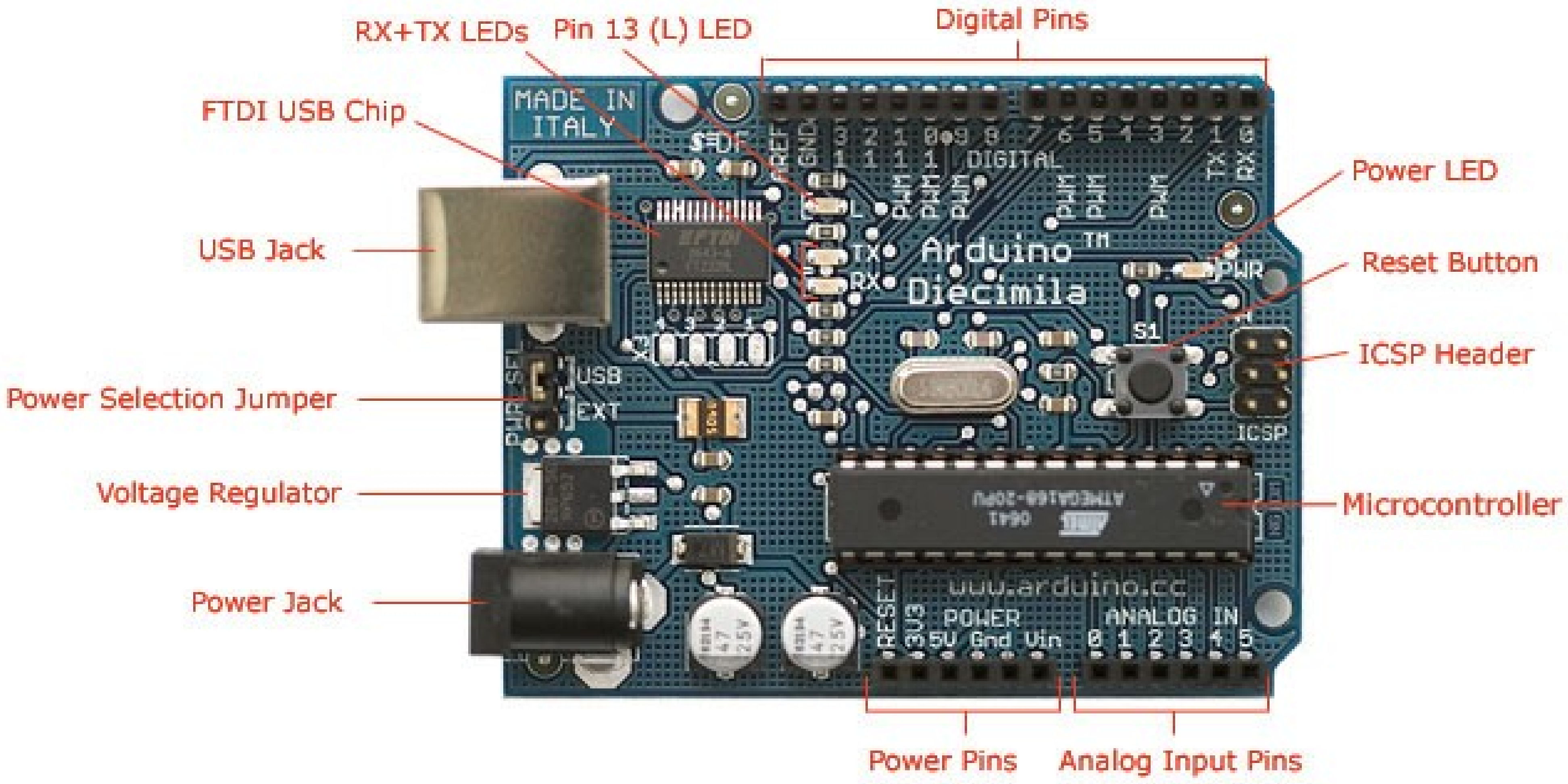
WHAT IS ARDUINO

- The Arduino project started in 2003 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy
- Open Source Hardware and Software Platform
- single-board microcontroller
- Allows to building digital devices and interactive objects that can sense and control objects in the physical world.

DEVELOPMENT ENVIRONMENT

- The Arduino Uno can be programmed with the Arduino software
- IDE(integrated development environment)
- The Atmega328 on the Arduino Uno comes preburned with a Bootloader that allows you to upload new code to it without the use of an external hardware programmer.
- You can also bypass the Bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header.

Arduino UNO



Photograph by SparkFun Electronics. Used under the Creative Commons Attribution Share-Alike 3.0 license.

UNO SPECIFCATION

- The Arduino Uno can be programmed with the Arduino software
- Microcontroller – Atmega328
- Operating Voltage 5V and 3.3 V
- Digital I/O Pins -14
- Analog Input Pins 6
- Flash Memory 32 KB (ATmega328) of which 0.5 KB used by Bootloader
- SRAM – 2KB
- EEPROM -1KB

MEMORY/STORAGE

- There are three pools of memory in the microcontroller used on avr-based Arduino boards :
  - Flash memory (program space), is where the Arduino sketch is stored.
  - SRAM (static random access memory) is where the sketch creates and manipulates variables when it runs.
  - EEPROM is memory space that programmers can use to store long-term information.
- Flash memory and EEPROM memory are non-volatile (the information persists after the power is turned off). SRAM is volatile and will be lost when the power is cycled.

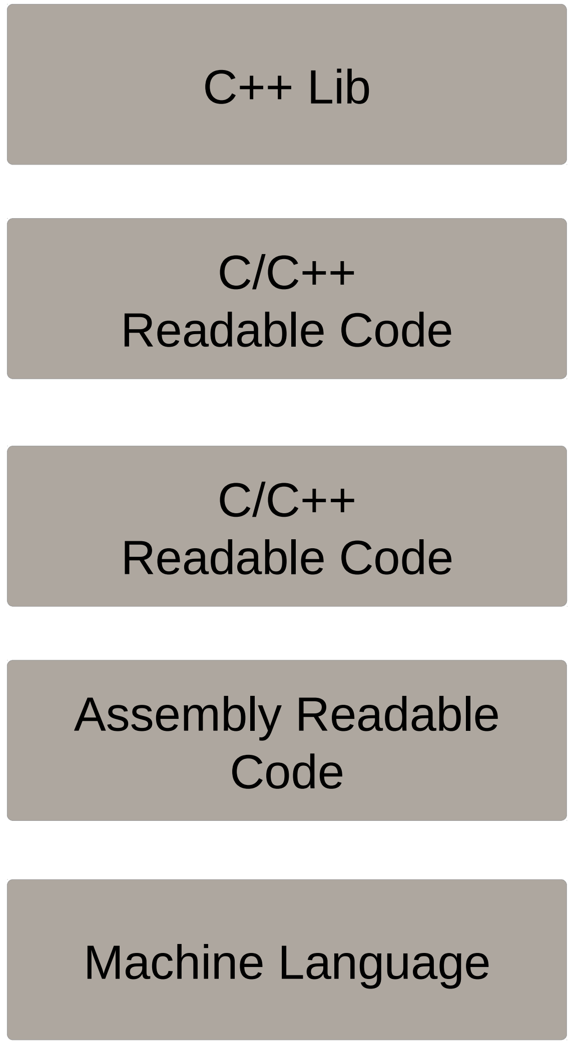
ARDUINO PROGRAMMING -GLOSSARY

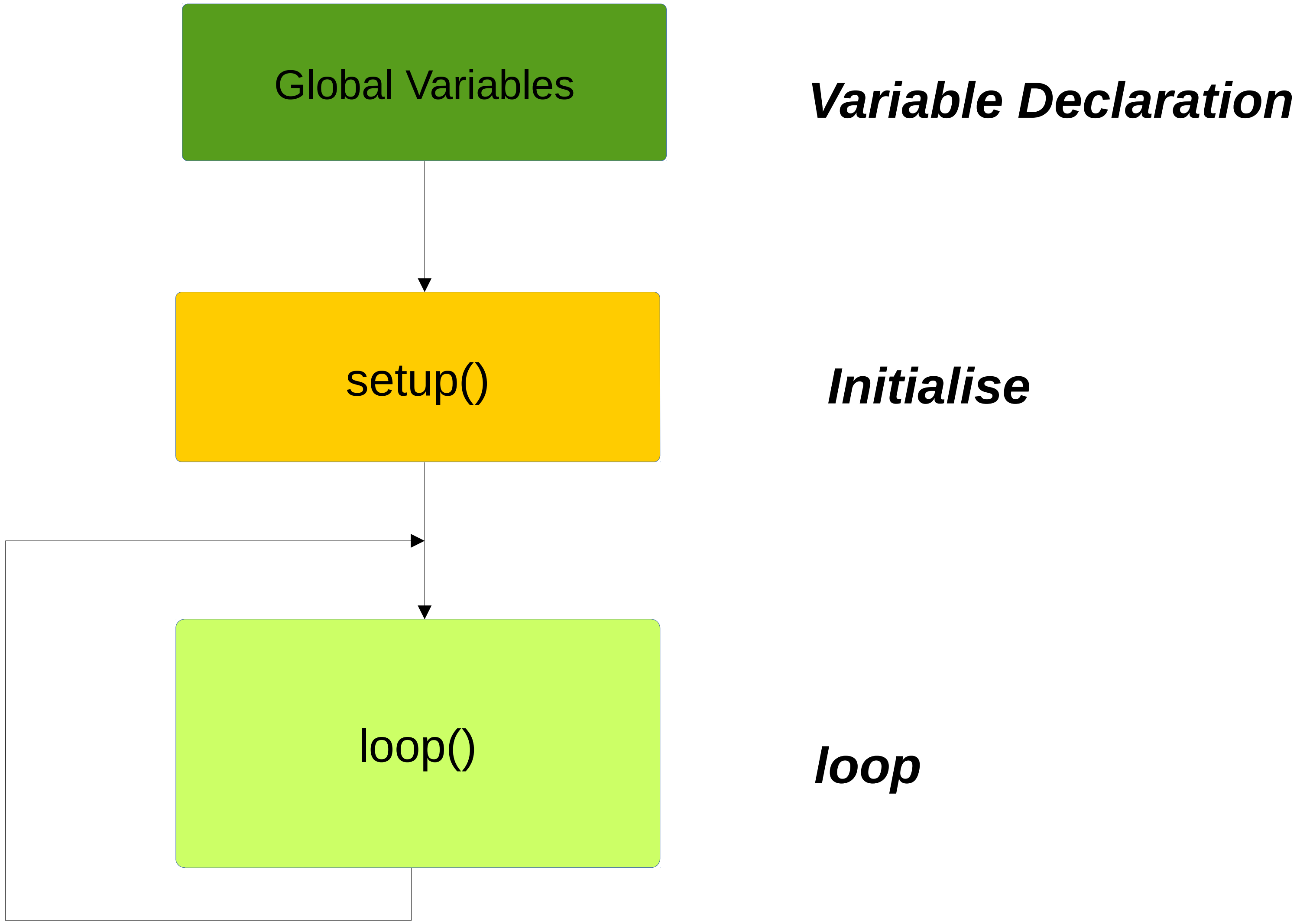
- Sketch – Program that runs on the board
- Pin – Input or Output connected to something – Eg: Output to an LED input to an Switch
- Digital – 1 (high) or 0 (Low) -ON/OFF
- Analog – Range 0-255 (Led brightness)
- Arduino IDE – Comes with C/C++ lib named as Wiring
- Programs are written in C & C++ but only having two funtcions -

  Setup() - Called once at the start of program, works as initialiser

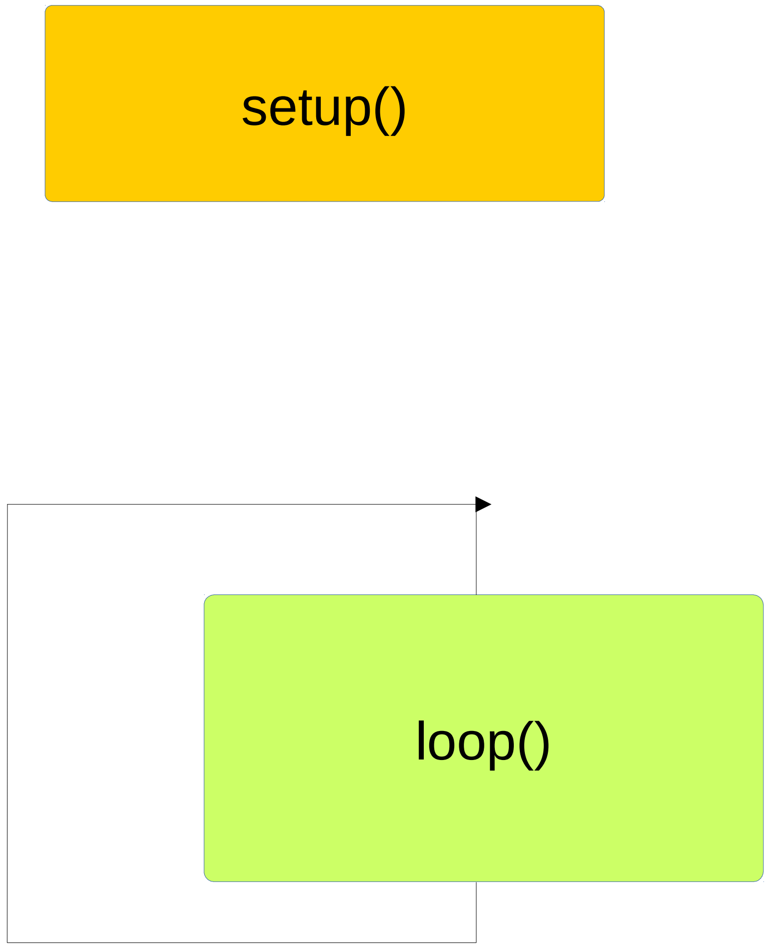
  Loop() - Called repeatedly until the board is powered-off

SKETCH


C++ Lib
C/C++ Readable Code
C/C++ Readable Code
Assembly Readable Code
Machine Language


Global Variables
Variable Declaration
setup()
Initialise
loop()
loop

SKETCH -setup()/loop()



pinMode() - set pin as input or output

serialBegin() - Set to talk to the computer

digitalWrite() - set digital pin as high/low

digtialRead() -read a digital pin state

wait()- wait an amount of time

SKETCH -Example

```
// 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 voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

For Statement -example

```
// Dim an LED using a PWM pin

int PWMpin = 10; // LED in series with 470 ohm resistor on pin 10
void setup()
{
  // no setup needed
}
void loop()
{
   for (int i=0; i <= 255; i++) {
      analogWrite(PWMpin, i);

//Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or
drive a motor at various speeds
      delay(10);
   }
}
```

If --- else

The if statement checks for a condition and executes the proceeding statement or set of statements if the condition is 'true'.

Syntax

```
if (condition)
{
    //statement(s)
}
```

Parameters

condition: a boolean expression i.e., can be true or false

Switch/Case statements

```
switch (var) {
    case 1:
        //do something when var equals 1
    break;
    case 2:
        //do something when var equals 2
    break;
    default:
        // if nothing else matches, do the default
        // default is optional
    }
```

For Statement

- The for statement is used to repeat a block of statements enclosed in curly braces.
- An increment counter is usually used to increment and terminate the loop.
- The for statement is useful for any repetitive operation, and is often used in combination with arrays to operate on collections of data/pins.

```
for (initialization; condition; increment) {
  //statement(s);
}
```

For Statement -example

```
// Dim an LED using a PWM pin

int PWMpin = 10; // LED in series with 470 ohm resistor on pin 10
void setup()
{
  // no setup needed
}
void loop()
{
   for (int i=0; i <= 255; i++) {
      analogWrite(PWMpin, i);

//Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or
drive a motor at various speeds
      delay(10);
   }
}
```

while statement

A while loop will loop continuously, and infinitely, until the expression inside the parenthesis, () becomes false.

```
while(condition){
       // statement(s)
}
```

Example :

```
var = 0;
while(var < 200){
      // do something repetitive 200 times
 var++;
}
```

Do – While statement

The do…while loop works in the same manner as the while loop, with the exception that the condition is tested at the end of the loop, so the do loop will always run at least once.

```
        do
            {
             // statement block
            } while (condition);
```

Example -

```
do
    {
         delay(50);          // wait for sensors to stabilize
          x = readSensors();  // check the sensors

    }     while (x < 100);
```

Data Types

- Integers, booleans, and characters
- Float: Data type for floating point numbers (those with a decimal point). They can range from 3.4028235E+38 down to -3.4028235E+38. Stored as 32 bits (4 bytes).
- Long: Data type for larger numbers, from -2,147,483,648 to 2,147,483,647, and store 32 bits (4 bytes) of information.
- String: On the Arduino, there are really two kinds of strings: strings (with a lower case ‘s’) can be created as an array of characters (of type char). String (with a capital ‘S’), is a String type object.

```
Char stringArray[10] = “isdi”;
String stringObject = String(“isdi”);
```

The advantage of the second method (using the String object) is that it allows you to use a number of built-in methods, such as length(), replace(), and equals().

ARDUINO -Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch.

select it from Sketch > Import Library.

Standard Libraries

- **EEPROM** - reading and writing to "permanent" storage
-
- **Ethernet / Ethernet 2** - for connecting to the internet using the Arduino Ethernet Shield, Arduino Ethernet Shield 2 and Arduino Leonardo ETH
-
- **Firmata** - for communicating with applications on the computer using a standard serial protocol.
-
- **GSM** - for connecting to a GSM/GRPS network with the GSM shield.
-
- **LiquidCrystal** - for controlling liquid crystal displays (LCDs)
-
- **SD** - for reading and writing SD cards
-
- **Servo** - for controlling servo motors
-
- **SPI** - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
-
- **Stepper** - for controlling stepper motors
-
- **TFT** - for drawing text , images, and shapes on the Arduino TFT screen
-
- **WiFi** - for connecting to the internet using the Arduino WiFi shield