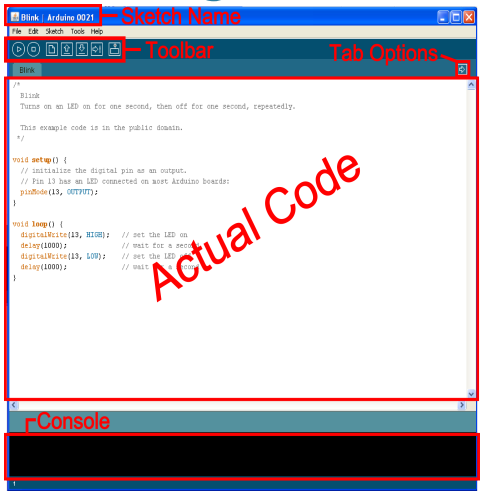
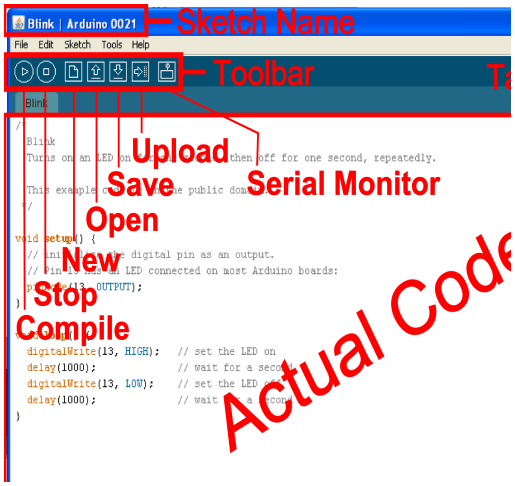
Getting to Know the IDE

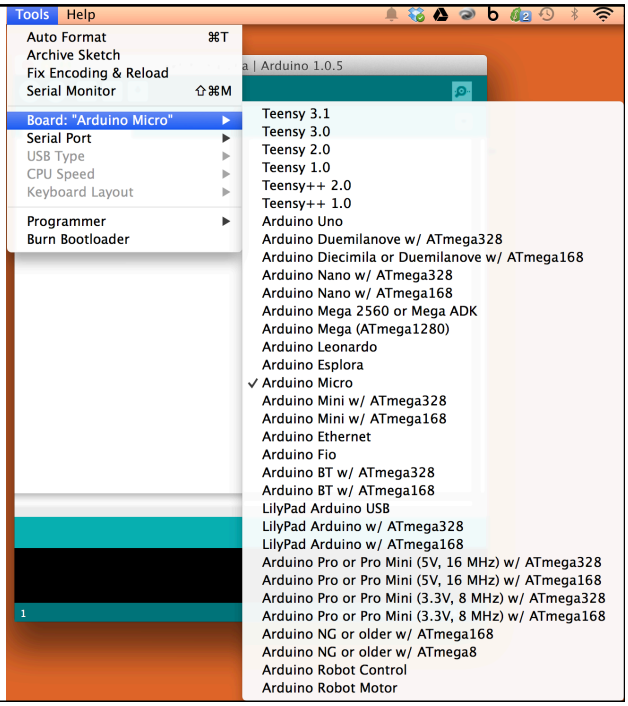




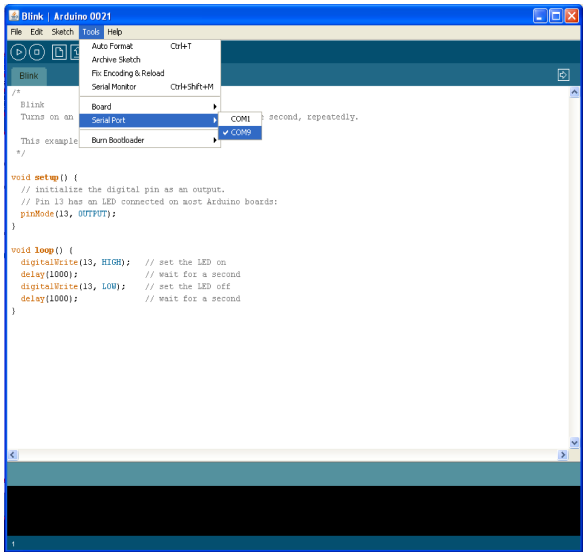
iRoboticist.com

Select the board

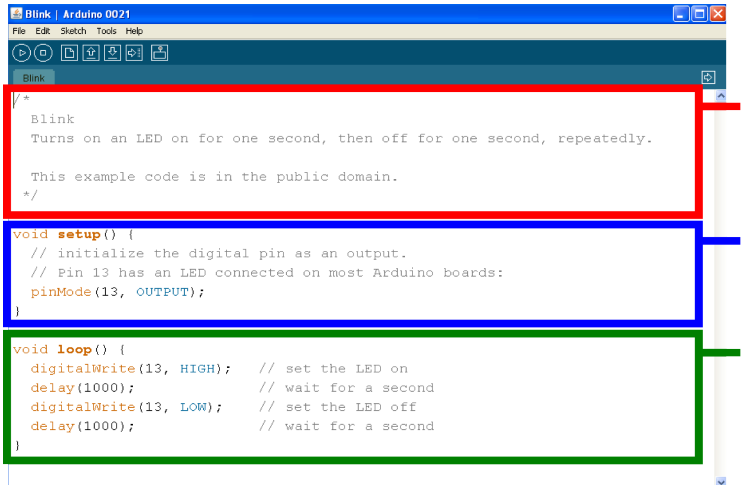
- To connect the Arduino Leonardo or Micro to your computer, you'll need a Micro-B USB cable.
- When programming the Micro, you must choose "Arduino Micro" from the **Tools > Board** menu in the Arduino IDE.



Serial Port / COM Port



Parts of the Sketch



Comments / Explaining the game

Setup / Stretching or tying shoes

Loop / Playing the game

Comments

- Comments can be anywhere
- Comments created with // or /* and */
- Comments do not affect code
- You may think that you don't need comments, but trust me you will.

Operators

The equals sign

= is used to assign a value

== is used to compare values

Operators

And & Or

&& is “and”

|| is “or”

Variables

Basic variable types:

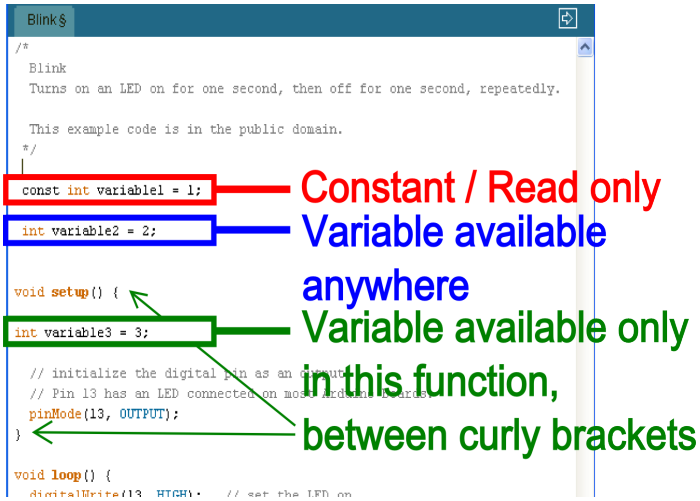
- Boolean : (True | False)
- Integer(16Bit) : (-32768….-1,0,2,3…32767)
- Character : (‘A’, ‘B’, ‘a’…) – ASCII
- Float(32 bit) : (-3.4028235^{38} to 3.4028235^{38})

Assigning Variables

- Boolean: *variableName = true;*
  variableName = false;
- Integer: *variableName = 32767;*
  variableName = -32768;
- Character: *variableName = 'A';*
  stringName[] = "iRoboticist";

Variable Scope

- Where you declare your variables matters

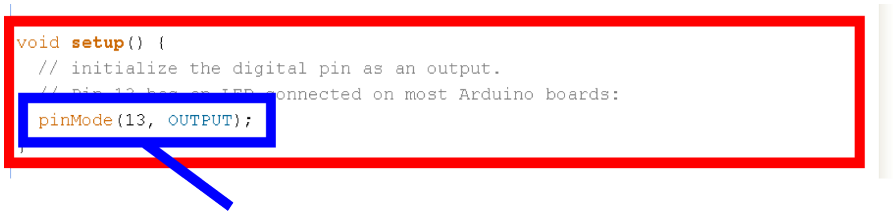


Basic Code Structure

```
void setup() {

  // The setup function comes before the loop function and is necessary for all Arduino sketches

  //The setup header will never change, everything else that occurs in setup happens inside the curly brackets

  //put your setup code here, to run once

  }

void loop() {

  // put your main code here, to run repeatedly

  }
```

iRoboticist.com

Setup: *void setup ( ) { pinMode (); }*



Outputs are declare in setup, this is done by using the pinMode function

This particular example declares digital pin # 13 as an output, remember to use CAPS

Setup: *void setup ( ) { Serial.begin;}*

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  Serial.begin(9600);
}
```

Serial communication also begins in setup

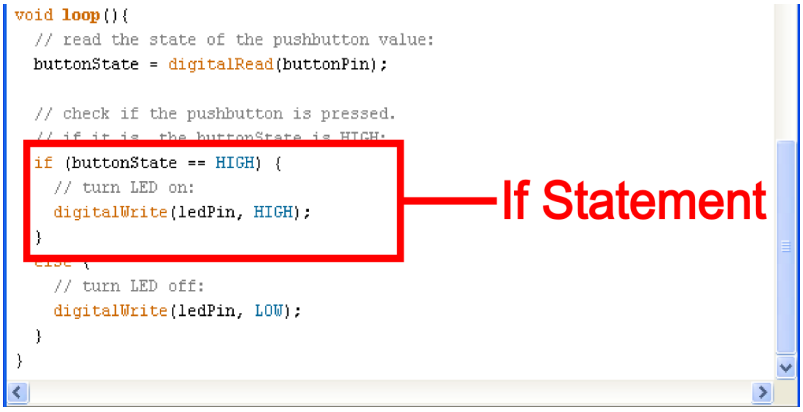
Setup: Internal Pullup Resistors *void setup ( ) { digitalWrite (); }*

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  digitalWrite(12, HIGH);
}
```

You can enable internal pullup resistors in setup, to do so digitalWrite the pin HIGH

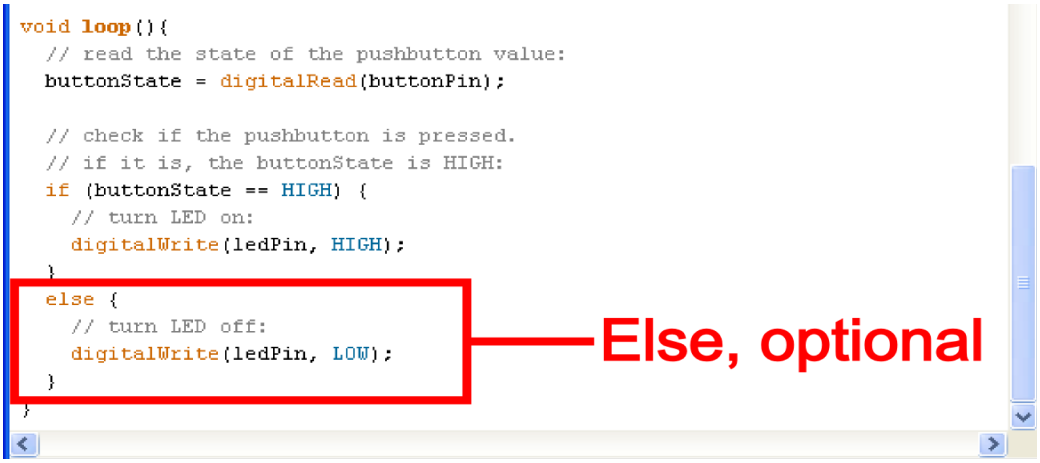
If Statements

if (this is true) { do this; }



Else

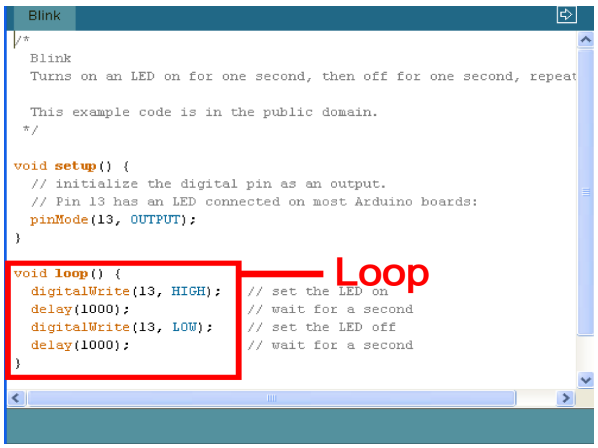
else { do this; }

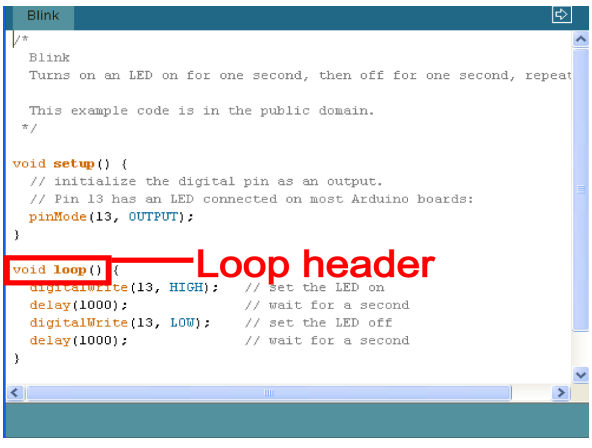


Basic Repetition

- loop
- For
- while

Basic Repetition: ***void loop ( ) { }***





Basic Repetition: ***void loop ( ) { }***

- The "void" in the header is what the function will return (or spit out) when it happens, in this case it returns nothing so it is void
- The "loop" in the header is what the function is called, sometimes you make the name up, sometimes (like loop) the function already has a name
- The "( )" in the header is where you declare any variables that you are "passing" (or sending) the function, the loop function is never "passed" any variables

Basic Repetition: for loop

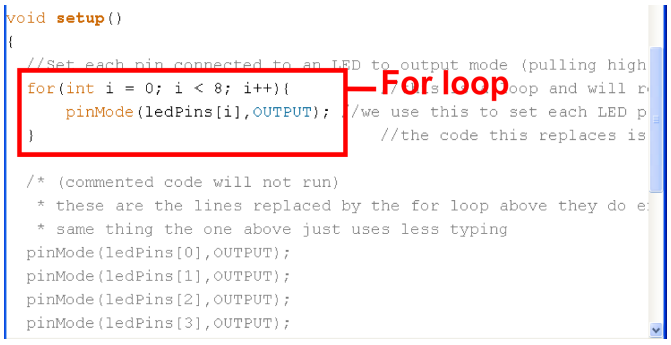
for (int count = 0; count<10; count++)

{

//for action code goes here

//this could be anything

}



Basic Repetition: While

Using Variable

```
while ( count<10 )

{

count = count ++;

/* should include a way to change count
variable so the computer is not stuck inside
the while loop forever */

}
```

Using Input

```
while ( digitalRead(buttonPin)==1 )

{

/*instead of changing a variable you just
   read a pin so the computer exits when
   you press a button or a sensor is
   tripped*/

}
```

Switch Case

```
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
        }
```

I/O Ports

- Programmable as Input OR Output Ports
- Further categorized as Analog and Digital Ports
- Defined I/O in setup()
  - LOW = 0 = Output
  - HIGH = 1 = Input

iRoboticist.com

Digital I/O

- The digital pins on an Arduino board can be used for general purpose input and output via the following commands.
  - pinMode()
  - digitalRead()
  - digitalWrite()
- Each pin has an internal pull-up resistor which can be turned on and off using digitalWrite() (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input.
- **The maximum current per pin is 40 mA.**

Digital I/O

- **pinMode()**
  - Syntax
    - pinMode(pin, mode)
  - Parameters
    - pin: the number of the pin whose mode you wish to set
    - mode: INPUT, OUTPUT, or INPUT_PULLUP. (see the digital pins page for a more complete description of the functionality.)

iRoboticist.com

Digital Output :**digitalWrite()**

- Description
  - Write a HIGH or a LOW value to a digital pin.
  - If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.
  - If the pin is configured as an INPUT, writing a HIGH value with digitalWrite() will enable an internal 20K pullup resistor. Writing LOW will disable the pullup. The pullup resistor is enough to light an LED dimly, so if LEDs appear to work, but very dimly, this is a likely cause. The remedy is to set the pin to an output with the pinMode() function.
- Syntax
  - digitalWrite(pin, value)
- Parameters
  - pin: the pin number
  - value: HIGH or LOW

Digital Output – Hands ON

- Blink LED
  - How to connect
  - How LED works
  - LED I & V drop

Digital Input :**digitalRead()**

- Description
  - Reads the value from a specified digital pin, either HIGH or LOW.
- Syntax
  - digitalRead(pin)
- Parameters
  - pin: the number of the digital pin you want to read (*int*)
- Returns
  - HIGH or LOW
- READ Switch ON/OFF
  - Switch De-bounce

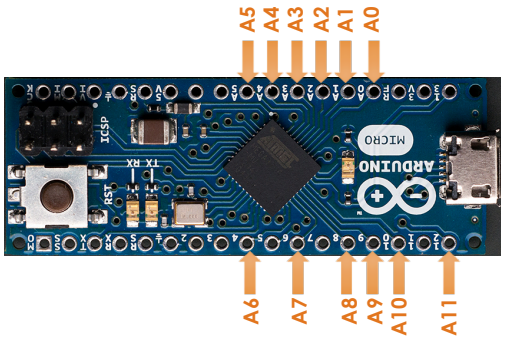
iRoboticist.com

Analog I/O

- The analog input pins support 10-bit analog-to-digital conversion (ADC) using the analogRead() function.
- Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 21 through analog input 10 as digital pin 11.
  - analogReference()
  - analogRead()
  - analogWrite() - *PWM*

Analog Input & Reference

- The Arduino Micro has 12 analog input pins



Analog Input :**analogRead()**

- Description
  - Reads the value from the specified analog pin. The board has 0-bit analog to digital converter. This means that it will map input voltages between 0 and 5 volts into integer values between 0 and 1023. This yields a resolution between readings of: 5 volts / 1024 units or, .0049 volts (4.9 mV) per unit. The input range and resolution can be changed using analogReference().
  - It takes about 100 microseconds (0.0001 s) to read an analog input, so the maximum reading rate is about 10,000 times a second.
- Syntax
  - analogRead(pin)
- Parameters
  - pin: the number of the analog input pin to read from (0 to 5 on most boards, 0 to 7 on the Mini and Nano, 0 to 15 on the Mega)
- Returns
  - int (0 to 1023)
- Note
  - If the analog input pin is not connected to anything, the value returned by analogRead() will fluctuate based on a number of factors (e.g. the values of the other analog inputs, how close your hand is to the board, etc.).

Analog Reference

- If the power supply voltage changes, so will your signal, but the ADC reference will also change the ADC's range to compensate, so you will continue to get correct results.

Analog Input

- Voltage Divider
- LDR
- A to D converter

iRoboticist.com

Analog Commands: **analogWrite()**

- Description
  - 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. After a call to **analogWrite()**, the pin will generate a steady square wave of the specified duty cycle until the next call to **analogWrite()**. The frequency of the PWM signal is approximately 490 Hz.
  - On Arduino Micro,this function works on 3, 5, 6, 9, 10, 11 & 13.
  - You do not need to call pinMode() to set the pin as an output before calling analogWrite().
  - ***The analogWrite function has nothing to do with the analog pins or the analogRead function.***
- Syntax
  - analogWrite(pin, value)
- Parameters
  - pin: the pin to write to.
  - value: the duty cycle: between 0 (always off) and 255 (always on).

Analog Output

- LED Brightness Control
- DC Motor Speed Control

PWM Signal Generation

- LED On/Off
- Voltage Reduction by PWM

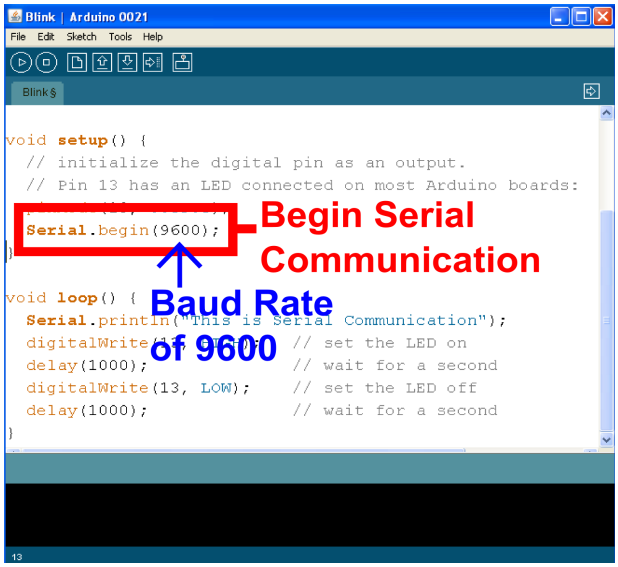
Code (if-else)

```
if ( You like semicolons ) {
        Stay here to learn to code
  }

else {
        Go build robot!
  }
```

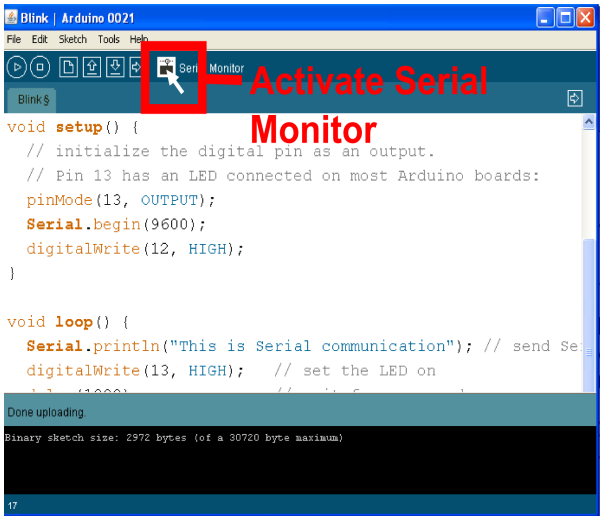
Serial Communication

Serial Communication is the transferring and receiving of information between two machines, the Micro dedicates pin # 0 to receiving information and pin #1 to transferring information

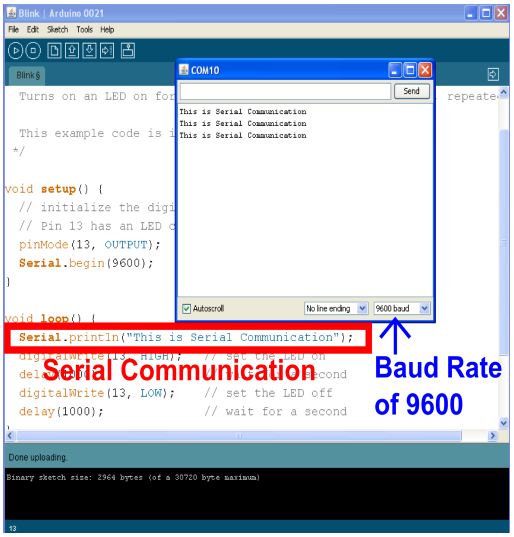
Serial in Setup



Serial Monitor

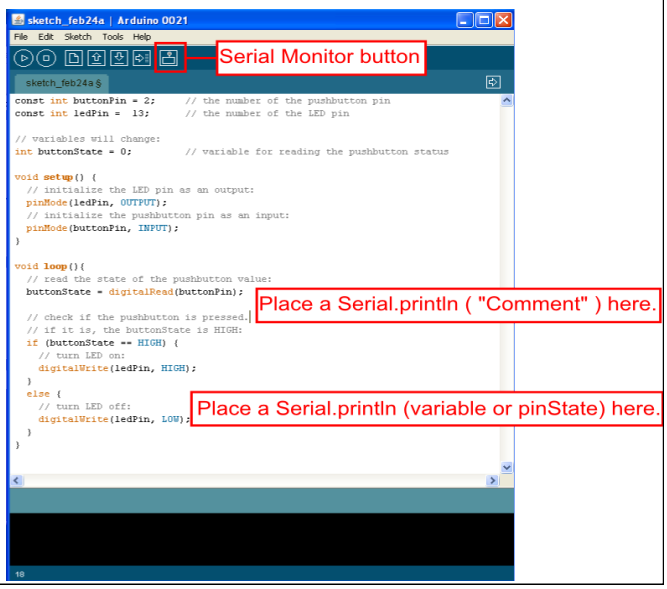


Serial Communication



Serial Activity

- Communication
- Troubleshooting circuits
- Debugging Code



Serial Communication: Serial Setup

```
void setup ( ) {
Serial.begin ( 9600 ) ;
}
```

In this case the number 9600 is the baudrate at which the computer and Arduino communicate

Serial Communication: Sending a Message

```
void loop ( ) {
Serial.print ( "Constructivism & " ) ;

Serial.println ( "Mechatronics" ) ;
}
```

Serial Communication: Serial Debugging

```
void loop ( ) {
int xVar = 10 ;
Serial.print ( "Variable xVar is " ) ;
Serial.println ( xVar ) ;
}
```

Serial Communication: Serial Troubleshooting

```
void loop ( ) {
Serial.print ( "Digital pin 9 reads " ) ;
Serial.println ( digitalRead ( 9 ) ) ;
}
```