ABLEDUCATION™
ACTIVITY BASED LEARNING

# Serial Communication with Arduino

## Serial.begin(baud rate)

It sets the data rate in bits per second (baud) for serial data transmission. For communicating with the computer, use one of these rates: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200

**Example:**

```
void setup()
{
  Serial.begin(9600); //opens serial port, sets data rate to 9600 bps
}

void loop() {}
```

**Serial.println(val);** -Prints data to the serial port ASCII text. Numbers are printed using an ASCII character for each digit. Floats are similarly printed as ASCII digits. Bytes are sent as a single character.

**Example:**

```
int x = 0;    // variable
void setup() {
  Serial.begin(9600);   // open the serial port at 9600 bps.
}
void loop() {
  Serial.print("NO FORMAT");    // prints a label
  Serial.print("\t");     // prints a tab
}
```

**Serial.print(val, format);** -An optional second parameter specifies the format to use permitted values are BIN (binary), OCT (octal), DEC (decimal), HEX (hexadecimal). For floating point numbers, this parameter specifies the number of decimal places to use.

**<u>Example:</u>**

```
int x = 0;    // variable
void setup() {
  Serial.begin(9600);      // open the serial port at 9600 bps:    }
void loop() {
for(x=0; x< 64; x++){
    Serial.print(x, DEC);  // print as an ASCII-encoded decimal
    Serial.print(x, HEX);  // print as an ASCII-encoded hexadecimal
 }}
```

**Serial.println(val);** - Prints data to the serial port ASCII text. Same as in **Serial.print(val);.** only difference is that the next value is printed in the next line.

__**Example:**__

```
int x = 0;    // variable
void setup() {
  Serial.begin(9600);      // open the serial port at 9600 bps
}
void loop() {
  Serial.println("NO FORMAT");       // prints a label
  Serial.println(" NEXT LINE");              // prints a label
 }
```

**Serial.println(val, format);** - Prints data to the serial port ASCII text with the format defined. Same as in **Serial.print(val, format);.** only difference is that the next value is printed in the next line.

**Serial.available(); -** Get the number of bytes (characters) available for reading from the serial port. This is data that's already arrived and stored in the serial receive buffer (which holds 64 bytes)

**Serial.read(); -** Reads incoming serial data.

## Example:

```
int x = 0;    // variable
void setup() {
  Serial.begin(9600);      // open the serial port at 9600 bps:
}
void loop() {
for(x=0; x< 64; x++){
    Serial.println(x, DEC);  // print as an ASCII-encoded decimal
    Serial.println(x, HEX);  // print as an ASCII-encoded hexadecimal
 }}
```

**Serial.write(); -** Writes binary data to the serial port. This data is sent as a byte or series of bytes.

**Serial.end(); -** Disables serial communication, allowing the RX and TX pins to be used for general input and output. To re-enable serial communication, call **Serial.begin().**

**Example:**

```
int incomingByte = 0;  // for incoming serial data
 void setup()
 {
   Serial.begin(9600);  //opens serial port, sets data  rate to 9600 bps
 }
```

```
void loop()
{
    Serial.write(45); // send a byte with the value 45
    Serial.write(“hello”); //send the string “hello”
    if (Serial.available() > 0)
      {
    incomingByte = Serial.read();
    Serial.print("I received: ");
    Serial.println(incomingByte, DEC);
     }
}
```

**Project Link : [https://youtu.be/WqDLrUhAg2s](https://youtu.be/WqDLrUhAg2s)**