

The Serial Monitor

Serial may sound like a tasty breakfast, but it is actually very different. The word serial means "one after another". In computers, *serial communication* is transmitting pieces of information *one after another, or one thing at a time*.

In the Arduino Esplora, we use the serial port in order to communicate another device with a serial port, like your computer! We use the **Serial Monitor** to read the information on our serial port. Download the code below and upload it to the Arduino Esplora.

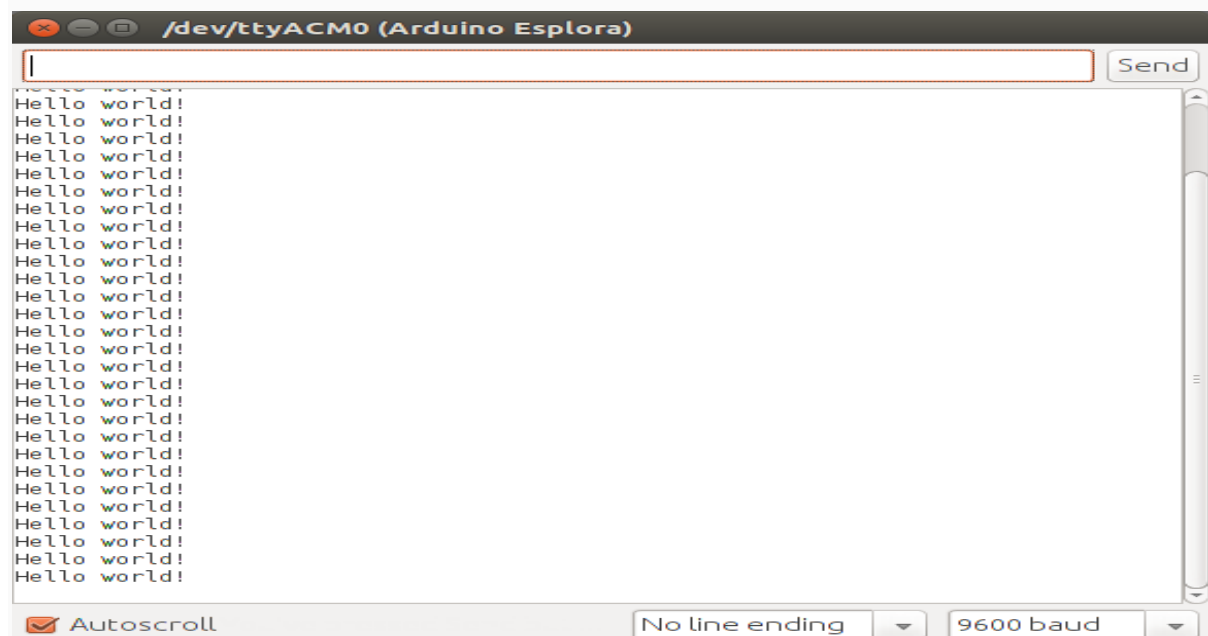
```
void setup() {  
  Serial.begin(9600); // Start serial communication to the computer at 9600 bits per second  
}  
  
void loop() {  
  Serial.println("Hello world!"); // Print hello world, ending with a new line  
  delay(1000);  
}
```

Activity

When you load the program above, you will see that nothing happens. That's because the Esplora is sending data to the computer. To see this data, we need to open the **Serial Monitor**. In the Arduino Software, there is a small icon on the top right. After you have uploaded your program, click this button to observe the programs output.



If everything is working, you should see "Hello World!" appear every second on a new line.



Let's analyse the code from the previous page. First let's look in the setup block, which runs only once.

There is only one line:

```
Serial.begin(9600);
```

This line initialises Serial communication with the computer at 9600* **bits per second** - similar to the speed of your internet except a lot slower.

Next we go into the **loop**.

```
Serial.println("Hello world!");  
delay(1000);
```

These two lines say: print a string of characters "Hello World!" to the Serial port, then wait 1 second (1000 milliseconds).

*9600 is a default conventional speed. We could make this value higher so that it streams faster, but we really don't need to since we are simply sending numbers and letters at a time.

Sending Sensor data to the Computer


We can communicate more than just the static text, "Hello World!", to the computer. We can also send numbers to the computer. These numbers could be representative of the levels we detect with our sensors, or the result of a number we've calculated. Load the following program onto your Esplora board, and see how we can display whatever value the **slider** reads on the computer screen.

Activity

```
#include <Esplora.h>
```

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

```
void loop() {  
    int slider = Esplora.readSlider(); // read the slider and store it in slider  
    Serial.print("The slider is: ");  
    Serial.println(slider);           // print the value with a new line after it  
    delay(500);  
}
```

After you have uploaded it to the Esplora, open up the Serial monitor  and you can see what the value of the slider is.

You can see with the slider, and all the sensor values, are saved as an integer. An integer is simply a whole number - that can be either positive or negative.

Activity

Challenge

Modify the above program so that instead of printing from the slider, it prints the **value of the joystick X and Y co-ordinates**. You may need to look at the [Esplora library reference](#) in order to know what function to use.

HELPFUL TIP: PRINTING TEXT IN BETWEEN NUMBERS

To print extra characters in between your values, you can use **Serial.print("<STRING>")**, which prints text without breaking to a new line. For example:

```
int myAge = 16;
```

```
int yourAge =15;
```

```
Serial.print("I am ");
```

```
Serial.print(myAge);
```

```
Serial.print(" years old. And you are ");
```

```
Serial.print(yourAge);
```

```
Serial.println(" years old.");
```

Would print in the Serial Monitor:

I am 16 years old. And you are 15 years old.

Notice how **Serial.print()** does not print a new line after the characters unlike **Serial.println()**.

Now modify the previous program so that it reads from SWITCH_1 and SWITCH_3, instead of the Joystick.

Hint: You will need to use the `Esplora.readButton()` function, just like we did for using the switches to control the LEDs.

Questions

1. What is the value when the joystick is pushed as far as you can to the RIGHT?
2. What is the value when the joystick is pushed as far as you can to the LEFT?
3. What is the value when the joystick is pushed as far as you can UP?
4. What is the value when the joystick is pushed as far as you can DOWN?
5. What is the value when the Switch_1 is NOT pressed in?

6. What is the value when the Switch_1 IS pressed in?
7. What is the value when the Switch_3 is NOT pressed in?
8. What is the value when the Switch_3 IS pressed in?