Arduino Uno Setup FAQ Sheet

Uploading Code to Arduino Uno

Uploading Process:

1. Connect your Arduino Uno to your computer via USB.
2. Download and install the Arduino Integrated Development Environment (IDE) from the Arduino website: https://www.arduino.cc/en/software. Choose the version appropriate for your operating system (e.g., Windows, macOS, Linux). Once downloaded, open Arduino IDE.
3. Write your own code or open the Arduino sketch (code) you want to upload.

Copy the following code:

```
/* FSR testing sketch.
Connect one end of FSR to power, the other end to Analog 0.
Then connect one end of a 10K resistor from Analog 0 to ground

For more information see https://learn.adafruit.com/force-sensitive-resistor-fsr */

int fsrPin = 0;     // the FSR and 10K pulldown are connected to a0
int fsrReading;     // the analog reading from the FSR resistor divider
int fsrVoltage;     // the analog reading converted to voltage
unsigned long fsrResistance;  // The voltage converted to resistance, can be very big so make "long"
unsigned long fsrConductance;
long fsrForce;       // Finally, the resistance converted to force

void setup(void) {
  Serial.begin(9600);   // We'll send debugging information via the Serial monitor
}

void loop(void) {
  fsrReading = analogRead(fsrPin);
  Serial.print("Analog reading = ");
  Serial.println(fsrReading);

  // analog voltage reading ranges from about 0 to 1023 which maps to 0V to 5V (= 5000mV)
  fsrVoltage = map(fsrReading, 0, 1023, 0, 5000);
  Serial.print("Voltage reading in mV = ");
  Serial.println(fsrVoltage);

  if (fsrVoltage == 0) {
    Serial.println("No pressure");
  } else {
    // The voltage = Vcc * R / (R + FSR) where R = 10K and Vcc = 5V
    // so FSR = ((Vcc - V) * R) / V        yay math!
    fsrResistance = 5000 - fsrVoltage;     // fsrVoltage is in millivolts so 5V = 5000mV
    fsrResistance *= 10000;                // 10K resistor
    fsrResistance /= fsrVoltage;
    Serial.print("FSR resistance in ohms = ");
```

```
    Serial.println(fsrResistance);

    fsrConductance = 1000000;           // we measure in micromhos so
    fsrConductance /= fsrResistance;
    Serial.print("Conductance in microMhos: ");
    Serial.println(fsrConductance);

    // Use the two FSR guide graphs to approximate the force
    if (fsrConductance <= 1000) {
      fsrForce = fsrConductance / 80;
      Serial.print("Force in Newtons: ");
      Serial.println(fsrForce);
    } else {
      fsrForce = fsrConductance - 1000;
      fsrForce /= 30;
      Serial.print("Force in Newtons: ");
      Serial.println(fsrForce);
    }
  }
  Serial.println("--------------------");
  delay(1000);
}
```

4. Select the correct board and port:
   a. Board: Go to `Tools` > `Board` and select *Arduino Uno*.
   b. Port: Go to `Tools` > `Port` and select the port to which your Arduino Uno is connected (usually something like COM3 or /dev/ttyUSB0 on Windows or Linux, respectively).
5. Click the *Upload* button (right-arrow icon). The code will compile and upload to your Arduino Uno.

Troubleshooting:

- If you encounter errors, ensure you have selected the correct board and port.
- Check whether any libraries required by your sketch need to be installed (`Sketch` > `Include Library` > `Manage Libraries...`).

Using the Serial Monitor

1. Opening the Serial Monitor:
   a. After uploading your sketch, you can open the Serial Monitor to view messages sent from the Arduino Uno.
   b. Go to `Tools` > `Serial Monitor` or press `Ctrl+Shift+M`.
2. Setting the Baud Rate:
   a. Ensure the baud rate in the Serial Monitor matches the baud rate specified in your Arduino sketch (`Serial.begin(baud_rate);`).
   b. You can change the baud rate in the Serial Monitor's dropdown menu at the bottom right.
3. Interpreting the Results:
   a. Messages sent from Arduino Uno via `Serial.print()` or `Serial.println()` will appear in the Serial Monitor.

b. Use the Serial Monitor to debug your code, monitor sensor readings, or display status updates from your Arduino project.

Using the Serial Plotter

1. Opening the Serial Plotter:
   a. The Serial Plotter graphs numerical data sent from the Arduino Uno over time.
   b. Go to `Tools` > `Serial Plotter`.

2. Plotting Data:
   a. Ensure the data sent from Arduino is numerical (e.g., sensor readings).
   b. Use `Serial.print()` to send data in a format that the Serial Plotter can interpret.
   c. Each `Serial.println()` sends a new data point that is plotted against time.

3. Customizing the Plot:
   a. Adjust the graph range and scaling using options in the Serial Plotter interface.
   b. Customize the appearance of plotted lines for clarity.

FAQs/Troubleshooting

Q: Why is my Arduino not uploading code?
A: Ensure the correct board and port are selected in the Arduino IDE under `Tools`. Check whether the Arduino Uno is powered on and connected via a reliable USB cable.

Q: How do I know which baud rate to use?
A: The baud rate in your sketch (`Serial.begin(baud_rate);`) and in the Serial Monitor must match. Common rates are 9,600, 115,200, etc., depending on your application.

Q: Why is my Serial Monitor showing gibberish or nothing?
A: Check the baud rate settings; they must match between your sketch and the Serial Monitor. Also, ensure your `Serial.print()` statements are correctly formatted.

Q: How can I plot sensor data using the Serial Plotter?
A: Use `Serial.print()` in your Arduino code to send numerical data. Open the Serial Plotter (`Tools` > `Serial Plotter`) to visualize this data over time.

Q: What should I do if my data isn't plotting correctly?
A: Verify that your Arduino code is sending data in a format the Serial Plotter can interpret (e.g., using `Serial.print()` for each data point).

By following these steps and guidelines, you should be able to effectively upload code to your Arduino Uno, using the Serial Monitor for debugging and the Serial Plotter to visualize data from your Arduino projects. Adjust baud rates and ensure that data formats are correct to get the best results from your interactions with the Arduino Uno.