Introduction to Arduino

The Arduino project was started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller at a cost of $50. In 2003 Hernando Barragán created the development platform *Wiring* as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas. The project goal was to create simple, low cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega128 microcontroller, an IDE based on Processing and library functions to easily program the microcontroller. In 2005, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, extended Wiring by adding support for the cheaper ATmega8 microcontroller. The new project, forked from Wiring, was called *Arduino*.

Arduinos are different types: Arduinome, a MIDI controller device that mimics the Monome. Ardupilot, drone software and hardware. ArduSat, a cubesat based on Arduino

Arduino Set Up

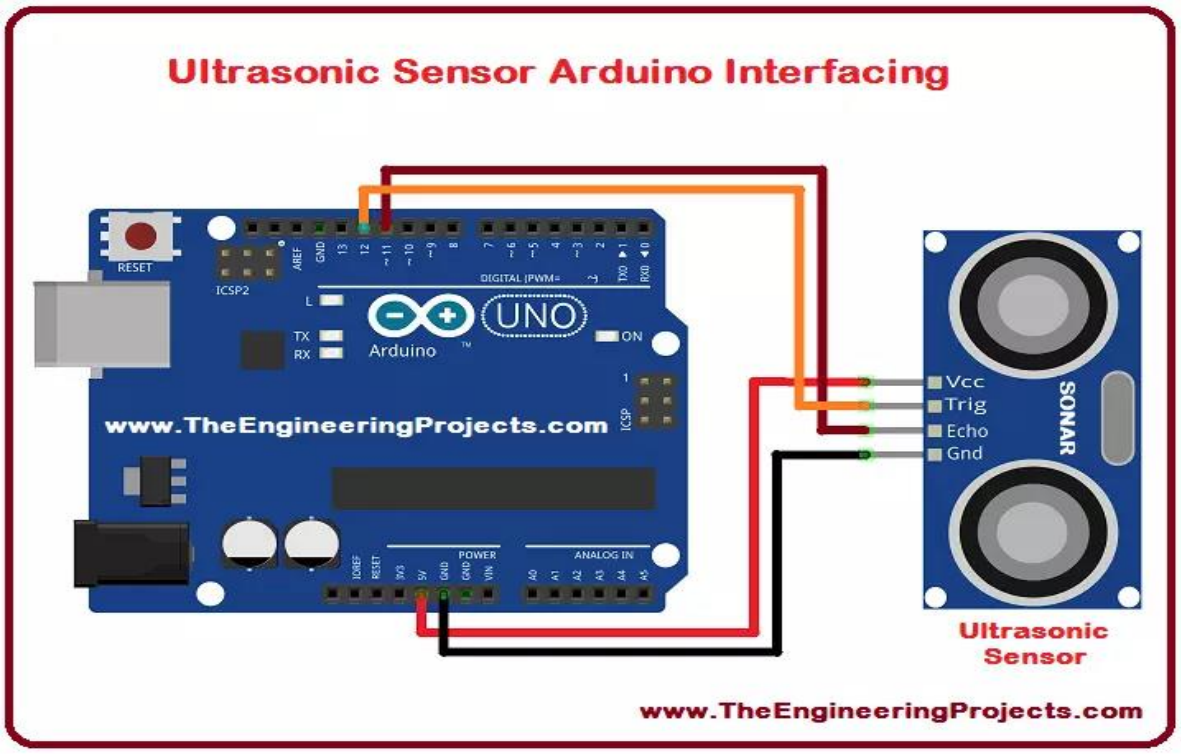
How to Set Up a Arduino into windows

Download the Arduino Software (IDE)

- Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a portable installation.
- After downloading, setup the exe file into your device.

Interfacing sensors with Arduino IDE

1. Interfacing the Ultrasonic sensor with Arduino IDE



Hardware Required

- Arduino Ide
- Ultrasonic Sensor (HC-SR04)
- Jumper wires
- Bread board (optional)

The HC-SR04 is an ultrasonic ranging module. This economical sensor provides 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm. Each HC-SR04 module includes an ultrasonic transmitter, a receiver and a control circuit.

There are **Four Pins** on the HC-SR04. They are:

- Vcc (5V supply)
- Gnd (Ground)
- Trig (Trigger)
- Echo (Receive)

SONAR (sound navigation and ranging) is very similar to the RADAR system. In both of the systems, the detection is based on the propagation of waves between the target and detector. There are two types of sonar systems, first is active sonar systems. Here the wave propagates from the transmitter to the target and back to the receiver, which is similar to pulse-echo radar. Then there are passive sonar systems. Here the target is the source of the energy which propagates to the receiver, which is similar to passive infrared detection. There is, however, a fundamental difference between sonar and radar systems. The energy transferred by acoustic waves is propagating in the water.

Ultrasonic transmitter: The transmittal the sound pulses and it uses 40kHz frequency for it.

Ultrasonic receiver: The receiver detects the reflected pulses. It outputs a signal which is used to measure the distance from the object to the sensor.

The connections are as follows:

- Vcc to 5V Pin of the Arduino.
- Gnd to Gnd Pin of the Arduino.
- Trig to Digital Pin.
- Echo to Digital pin.

Refer the schematics for more clarity on the connections.

Few things to remember while building the circuit

- Avoid placing the sensor on metal surfaces to avoid short circuits which might burn the sensor.
- It is recommended to put electrical tape on the back side of the sensor.
- You can also directly connect the Ultrasonic sensor to the Arduino with jumper wires directly.

Working of the Ultrasonic sensor

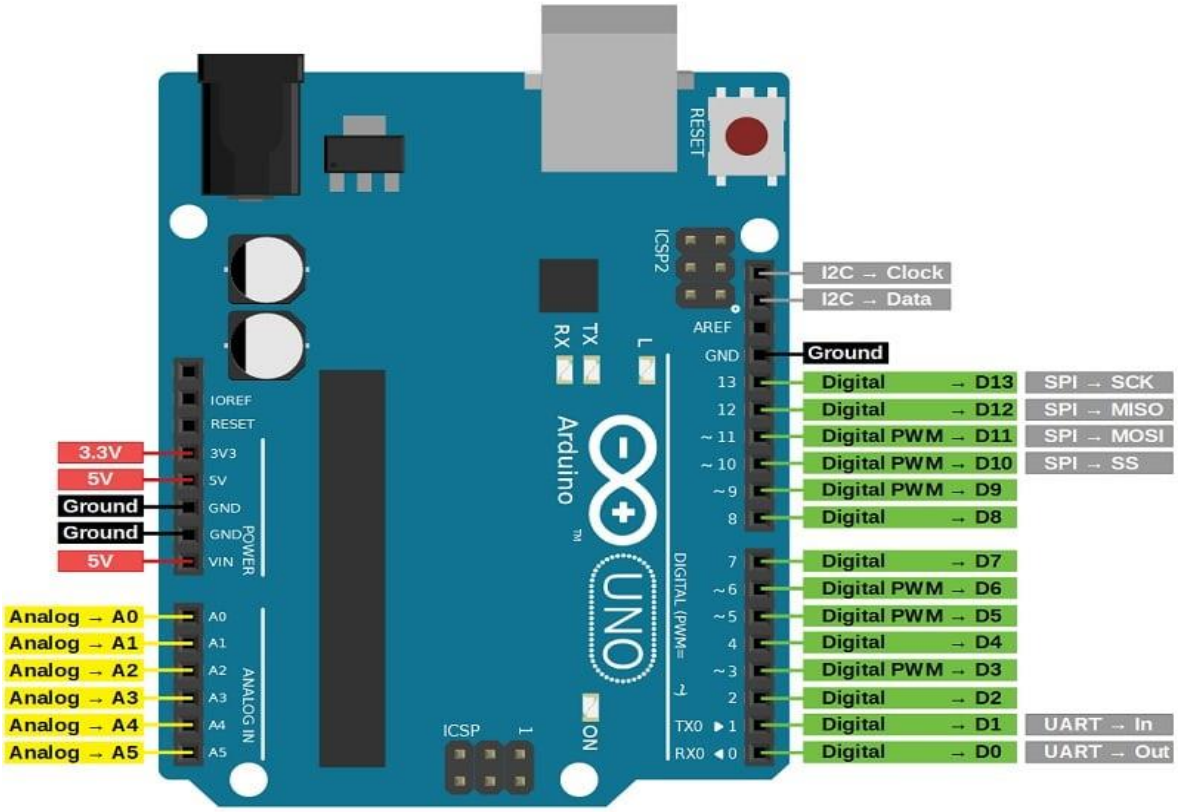
A 5V pulse is applied to the TRIG pin for at least 10 microseconds.

.
An Ultrasonic Sensor is a device that measures distance to an object using **Sound Waves**. It works by sending out a sound wave at ultrasonic frequency and waits for it to bounce back from the object. Then, the time delay between transmission of sound and receiving of the sound is used to calculate the distance.

It is done using the formula **Distance = (Speed of sound * Time delay) / 2**

We divide the distance formula by 2 because the sound waves travel a round trip i.e from the sensor and back to the sensor which doubles the actual distance.

The HC-SR04 is a typical ultrasonic sensor which is used in many projects such as obstacle detector and electronic distance measurement tapes. In this Instructable I'll teach you how to interface the HC-SC04 with an Arduino Uno.



Interfacing with Arduino
The circuit diagrams for this are very simple as we simply have to connect all 4 pins to the GPIO pins of the Arduino.

Code for Arduino IDE

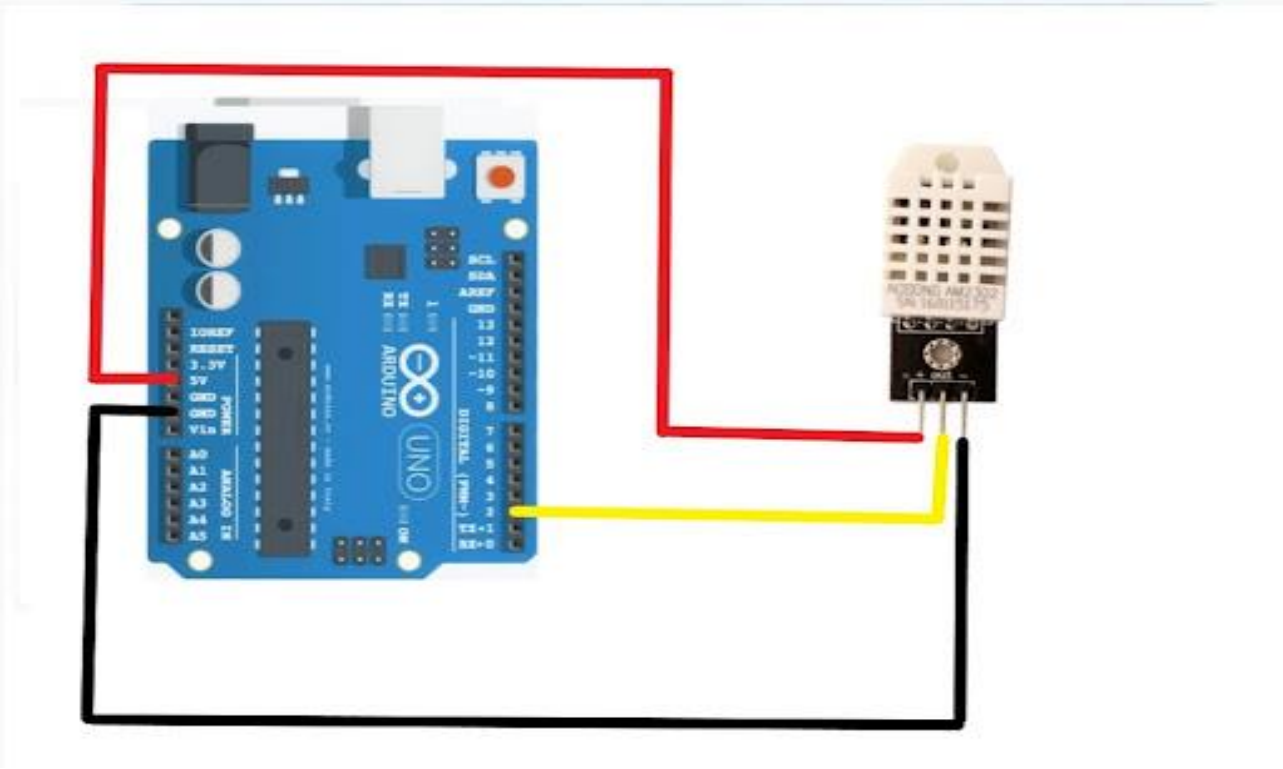
```
int trig = 7;
int echo = 6;
int timeInMicro;
int distanceInCm;
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(7, OUTPUT);
  pinMode(6, INPUT);
}

void loop()
{
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  timeInMicro = pulseIn(echo, HIGH);

  distanceInCm = ((timeInMicro / 29) / 2);

  Serial.println(distanceInCm);
  delay(100);
}
```

Upload this code to the Arduino hardware board via Arduino ide terminal and you will see the distance on the serial monitor.

I hope you learned something about the Ultrasonic sensor and its interfacing with arduino. And I hope you enjoyed it, Thank you.

2. Interfacing the Temperature sensor(DHT 22) with Arduino IDE



Hardware Requirements:

- **DHT22 has 3 pins are,**

➢ Pin 1 – VCC or +5v

➢ Pin 2 – Data or output pin

➢ Pin 3 – Ground.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory. Small size & low consumption & long transmission distance(20m) enable DHT22 to be suited in all kinds of harsh application occasions.

The DHT22 sensor has a better resolution and a wider temperature and humidity measurement range. However, it is a bit more expensive, and you can only request readings with an interval of 2 seconds. Despite their differences, they work in a similar way and you can use the same code to read the temperature and relative humidity.

Working of the Temperature sensor (DHT 22):

Here we can observe that a pull-up resistor is connected between +5v, output pin and the Pin 2 on the DHT22 sensor. This is not required as there is an inbuilt pull up resistor present in the Arduino Uno.

After completing all the connections as shown above, we connect the USB Port on the Arduino Uno to the PC/Laptop.

Installing Libraries

For installing drivers open Arduino IDE and go to **Sketch > Include Library > Manage Libraries.** The library manager should open.Here search for '**DHT**' and install the DHT library from **Adafruit**.

The DHT22 Temperature Humidity Sensor is a great choice for measuring ambient temperature and humidity. This sensor consists of a capacitive humidity sensor and a heat resistor. The DHT22 sensor has a wide range of temperature and humidity measurement. It can measure temperature in range of -40 to +125 degrees Celsius with 0.5 degrees Celsius accuracy.

Humidity is measured in range of 0 to 100% with 2.5% accuracy. This sensor transmits data through a single digital pin, which makes it easy to communicate with various microcontrollers. The sampling frequency is also 0.5Hz. This means that it updates the temperature and humidity data every two seconds.

Code for Arduino IDE

```
#include <DHT.h>

#define DHTPIN 3    // Pin where the DHT22 is connected

#define DHTTYPE DHT11  // DHT22 (AM2302) sensor type

DHT dht(DHTPIN, DHTTYPE);

void setup() {

  Serial.begin(9600);

  dht.begin();
```

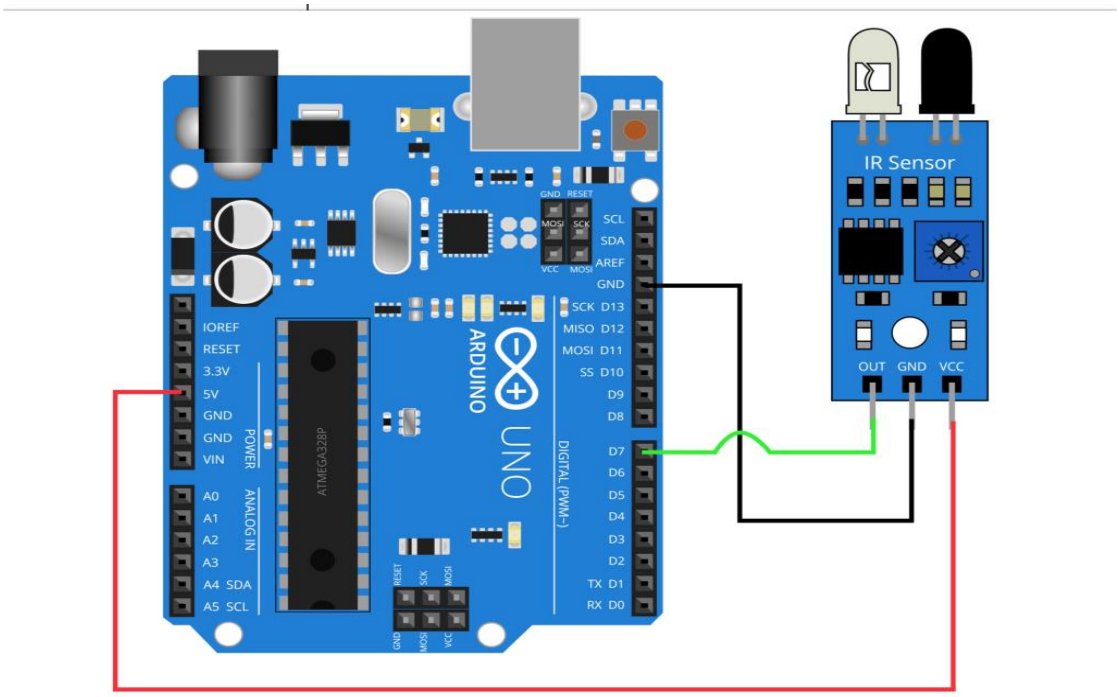
```
}

void loop()

{

  delay(2000);  // Delay for 2 seconds between readings

  float temperature = dht.readTemperature();  // Read temperature in
Celsius

  if (isnan(temperature)) {

    Serial.println("Failed to read from DHT sensor!");

  } else {

    Serial.print("Temperature: ");

    Serial.print(temperature);

    Serial.println(" °C");

  }

}
```

We are done with writing the code, we will verify/compile the code by clicking on the tick button on the left corner on the top of the Arduino IDE screen. Once the code is compiled, we will click on the upload button that is right next to the compile button. By clicking this we will be uploading the code onto the Arduino Uno.

After uploading the code to the Arduino, open the Serial Monitor at a baud rate of 9600. You should get sensor readings every two seconds. Here's what you should see in your Arduino IDE Serial Monitor.

I hope you learned something about the temperature sensor and its interfacing with arduino. And I hope you enjoyed it, Thank you.

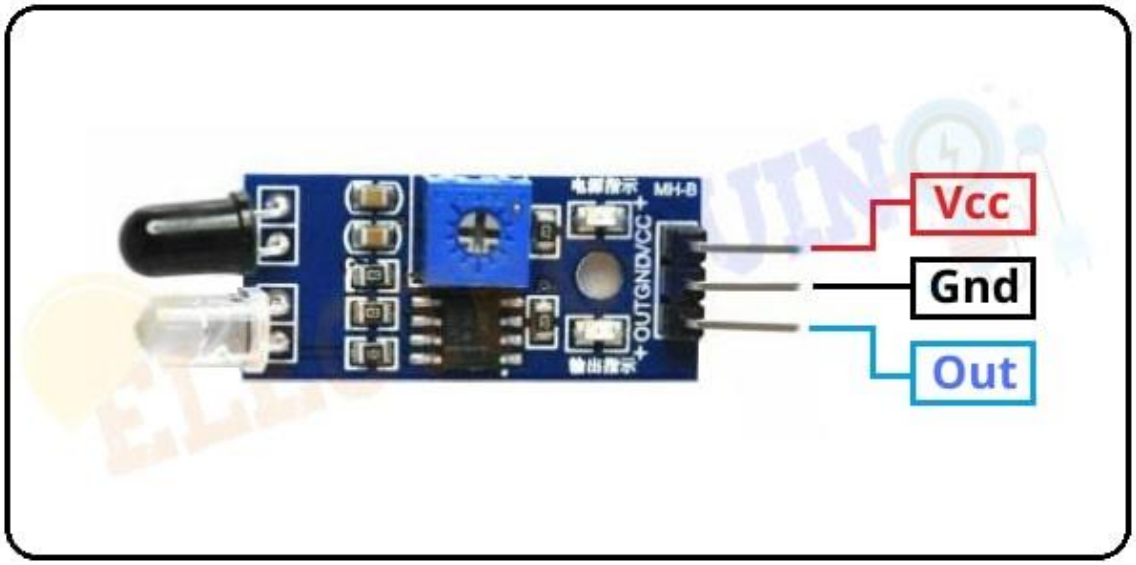
3. Interfacing the Temperature sensor(DHT 22) with Arduino IDE



Interfacing IR Sensor with Arduino is a first step to understanding how to use an IR sensor in different projects. Here we will make a basic project, where the IR sensor detects an object and indicated by an LED. Using this project we will also learn how to connect this sensor to the Arduino and how the Arduino reads output data from the sensor using programming/code.
The project concept is very simple when an obstacle or object comes in front of the IR sensor, then the sensor gives an output to the Arduino, then the Arduino sent output voltage to turn on the LED. When the sensor doesn't detect an obstacle or object, then the Arduino turns off the LED.

Hardware Requirements:

- Arduino UNO
- IR Sensor Module
- LED
- Jumper wires



We need to connect the IR sensor with Arduino properly to read the output of the sensor. First of all, we connected the sensor **Vcc pin** to the Arduino **5v pin** and the **GND** pin is connected to the Arduino **ground (GND)** pin, to activate the IR sensor module. Then connect the sensor output pin to one of the digital pin of Arduino to read the output value from the IR sensor module

Circuit Wiring

Components Pin	Arduino Pin
IR sensor Vcc Pin	+ 5v Pin
IR sensor GND Pin	GND (ground) pin
IR sensor OUT Pin	Digital pin “D3”

Working of the IR sensor:

In this circuit, the sensor output pin is connected to the digital pin 3. The IR or obstacle sensor gives logic HIGH (1) as output when there is no obstacle in front of it, and it will give logic LOW (0) output when an obstacle is placed in front of it. We need to read these logical changes on the Arduino. The LED is an indicator which is connected to Arduino digital pin 11. Interfacing the IR sensor with Arduino works using simple Logic. That is If Arduino reads LOW data from the sensor output pin, LED should turn ON. Else if Arduino reads HIGH data from the sensor output pin, LED should turn OFF.

The IR sensor is used to detect the presence of an object in its range. If there is an object in its vicinity, the IR output reads low and, the LED glows. The IR output reads high if there is no object in its range and the LED is turned OFF. A relay switch can be employed to control the LED following the IR sensor output.

Code for Arduino IDE

```
int IRSensor = 9; // connect IR sensor module to Arduino pin D9
int LED = 13; // connect LED to Arduino pin 13

void setup(){
  Serial.begin(115200); // Init Serial at 115200 Baud Rate.
  Serial.println("Serial Working"); // Test to check if serial is working or not
  pinMode(IRSensor, INPUT); // IR Sensor pin INPUT
  pinMode(LED, OUTPUT); // LED Pin Output
}

void loop(){
  int sensorStatus = digitalRead(IRSensor); // Set the GPIO as Input
  if (sensorStatus == 1) // Check if the pin high or not
  {
    // if the pin is high turn off the onboard Led
    digitalWrite(LED, LOW); // LED LOW
    Serial.println("Motion Detected!"); // print Motion Detected! on the serial
monitor window
  }
  else  {
```

```
    //else turn on the onboard LED
    digitalWrite(LED, HIGH); // LED High
    Serial.println("Motion Ended!"); // print Motion Ended! on the serial monitor
window
  }
}
```

This code will now read the values of the additional IR sensors and print them to the serial monitor, along with the values from the previously declared IR sensors. You can continue adding as many IR sensors as you need in a similar manner, just make sure to give each IR sensor a unique constant name and corresponding digital pin.

I hope you learned something about the IR sensor and its interfacing with arduino. And I hope you enjoyed it, Thank you.