**Humanoid Robot**

**Step-by-step Process:**

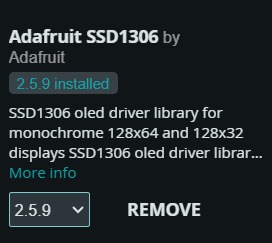
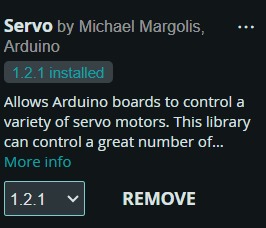
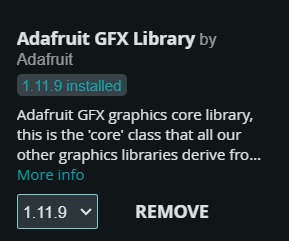
**1. Importing Libraries**

**For Arduino:**

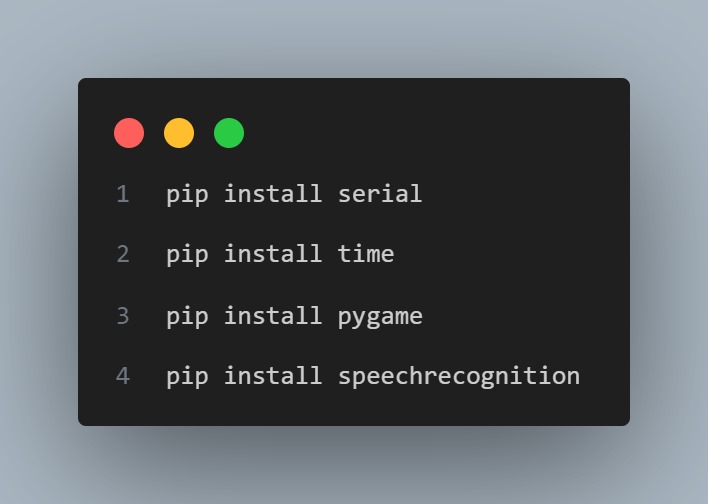
Go to tools -> Include Library -> Manage Libraries...Find and Install below Libraries:

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include <Servo.h>

**The Python script starts by importing the necessary libraries:**

* **serial**: For establishing serial communication with the Arduino board.
* **pygame:** For playing audio files.
* **speech\_recognition**: For recognizing voice commands.

pip install SpeechRecognition==3.8.1

pip install pygame==2.0.1

pip install pyserial==3.5

**Note: the specific version of each module or library.**

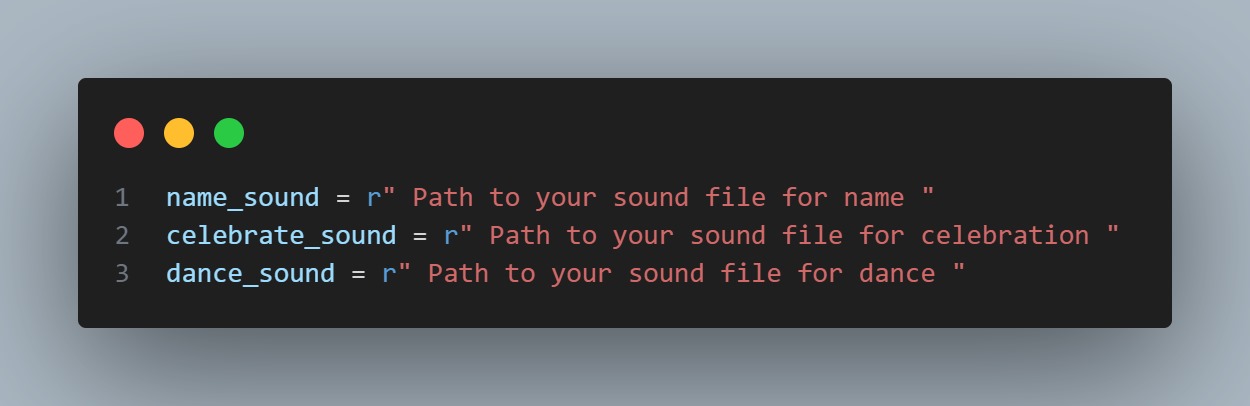
**2. Establish Serial Connection**

The script creates a serial connection (ser) with the Arduino board:

You need to adjust the COM port according to your setup.

**3. Define Audio File Paths**

Paths to audio files are defined for specific actions:

* name\_sound: Path to the audio file for the robot introducing itself.
* celebrate\_sound: Path to the audio file for a celebration action.
* dance\_sound: Path to the audio file for a dancing action.
* Reset sound,left\_turning sound, right-turning sound, info, and many more.

**4. Define Robot Action Functions**

Functions are defined for various actions the robot can perform:

happy(), sad(): Set the robot to a happy or sad emotional state.

turn\_left(), turn\_right(): Instruct the robot to turn left or right.

reset(): Reset the robot's position to default.

hello(): Perform a "hello" gesture.

Info(): giving information about the building structure

Temp(): reading the real-time temperature

name(), ronaldo(), dance(),surprise(), angry(): Trigger specific predefined actions on the robot.

**5. Voice Recognition Loop**

* The main loop listens for voice commands using the speech\_recognition library:
* It listens for spoken commands through the device Microphone.

**6. Recognize and Execute Commands**

* Inside the loop, the script recognizes the spoken command and executes the corresponding function:
* It recognizes commands such as 'happy', 'sad', 'turn left', 'turn right', etc.
* It calls the respective function for the recognized command.

**7. Audio Feedback for Specific Commands**

* For specific commands ('what is your name', 'celebrate', 'dance'), it plays associated audio files using pygame.mixer:

if command == 'what is your name':

name()

pygame.mixer.music.load(name\_sound)

pygame.mixer.music.play()

**8. Print Arduino Feedback**

* The script also checks for any feedback from the Arduino board:

if ser.in\_waiting > 0:

data = ser.readline().decode().strip()

print("Arduino says:", data)

**9. Termination Condition**

The loop continues until the user says 'exit':

elif command == 'exit':

break

**10. Wait for Audio Playback Completion**

After executing the command, the script waits for the audio playback to complete:

pygame.mixer.music.get\_busy():

pygame. time.Clock().tick(10)

**11. Close Serial Connection**

Finally, when the loop exits (when the 'exit' command is given), it closes the serial connection:

ser.close()