**Resberry Pi OLED display Google Sheets content Python Script Documentation**

**1. Overview**

This Python script is designed to run on a Raspberry Pi and control an OLED display connected via I2C communication. It interacts with a Google Sheets script to fetch data and display it on the OLED screen. The script allows users to navigate through the data using physical buttons connected to GPIO pins.

**2. Dependencies**

**asyncio:** Asynchronous I/O library for managing asynchronous operations.

**requests:** HTTP library for making requests to the Google Sheets API.

**json:** Library for working with JSON data.

**time:** Standard Python library for timerelated functions.

**RPi.GPIO:** Library for accessing the GPIO pins on the Raspberry Pi.

**board:** Part of the Adafruit Blinka library, provides an abstract way to access hardware pins.

**digitalio:** Part of the Adafruit Blinka library, provides digital input and output capabilities.

**PIL:** Python Imaging Library for image processing.

**adafruit\_ssd1306:** Library for controlling SSD1306based OLED displays.

**3. Hardware Requirements**

* Raspberry Pi
* SSD1306based OLED display
* Push buttons connected to GPIO pins for navigation

**4. Setup**

**4.1. Installing Dependencies**

Ensure that all required Python libraries are installed on your Raspberry Pi. You can use pip to install them:

pip install asyncio requests adafruitblinka adafruitcircuitpythonssd1306 Pillow

**4.2. Wiring**

Connect the SSD1306 OLED display to the Raspberry Pi using the I2C interface. Connect the push buttons to the GPIO pins (nextButtonPin and prevButtonPin) for navigation.

**4.3. Google Sheets Setup**

Set up a Google Sheets script to serve data to the Raspberry Pi. The script should accept commands (next, previous, current, all) along with row and column numbers and return the corresponding data.

**4.4. Preferences File**

Create a JSON file named preferences.json to store preferences such as current row, current column, total rows, total columns, and all cell values.

**5. Usage**

1. Run the Python script on the Raspberry Pi:

python your\_script\_name.py

2. The script will initialize the OLED display, set up GPIO pins, and load preferences from the preferences.json file.

3. Press the navigation buttons to move between cells and display the corresponding data on the OLED screen.

4. The script will periodically fetch updated data from the Google Sheets script based on the specified update time.

**6. Customization**

Modify variables such as SCREEN\_WIDTH, SCREEN\_HEIGHT, updateTime, serverAddress, nextButtonPin, prevButtonPin, etc., according to your specific requirements.

**7. Error Handling**

The script includes error handling for network requests (Timeout, RequestException) and file I/O operations (FileNotFoundError).

**8. Notes**

Ensure that your Google Sheets script is accessible and returns data in the expected format to avoid errors.

Test the script thoroughly with different scenarios to ensure proper functionality.