1. Why do both the SerialTest-Write.py and SerialLightControl-Client.py scripts use the encode() method of the string datatype when writing data to the serial port?

Serial communication requires data to be transmitted as raw bytes instead of text strings. Using the “encode()” method converts strings into byte format. Byte format, such as UTF-8 or ASCII, is needed for machines to properly interpret the commands. Without this additional encoding, Python would send Unicode, which could lead to communication errors or corrupted data. “encode()” is essential for smooth & error free communication between Python & serial connected devices.

1. Why does the SerialTest-Read.py script use the decode() method of the string datatype when reading the data from the serial port?

Similar to the “encode()” method, the “decode()” method is used because data received from a serial port is transmitted as raw bytes, not readable text. Since Python reads from the serial port, which needs to be converted back into a regular Unicode strig for easier processing. Using “decode()” method, it converts the data using specified encoding to ensure the correct interpretation of special character, numbers, & text. Without it, we would encounter harder to manipulate data or incorrectly shown characters.

1. What is the purpose of the try/except block in both the SerialLightControl-Client.py script and the SerialLightControl-Server.py script?

“Try/Except” blocks ensure that the program handles errors gracefully during serial communication. If this connection fails, the “except” black catches the error, prevents a crash while allowing for a clean exit or user feedback. By improving reliability, especially in hardware interactions where sudden failures could leave devices in an unstable state.

1. Why is it necessary to make sure that the GPIO pins are always returned to their original state at the end of program run?

By resetting the GPIO pins at the end of a program, we can prevent hardware damages. In ensuring no pins stay stick on. This always avoids conflicts when restarting the program or running other scrips that might use the same pins. Lastly, it’s most important to keep your electronics stable & predictable.