PIPSTA111 – WEB SEND PYTHON CODE TUTORIAL

Contents

Revision History

Revision	Author	Date	Description
1.0	AH	09/12/14	First Release

Difficulty Level:

- This tutorial by necessity incorporates SQL commands in order to manipulate the database.
- The code does not work 'out-of-the-box' as the user will be required to provide various web-hosting parameters.

Time to Complete:

- Aside from moving between Python and web-hosted databases, this tutorial should not take more than half an hour to complete.

Who Should Read This Document

This tutorial is suitable for those wishing to understand the function of the **WebSend.py** script, either for demonstration use or in order to tailor the script to their own application. Whilst knowledge of Python and SQL are beneficial, the code is fairly readable even without knowledge in these areas.

Readers should have gone through at least ***PIPSTA010, PIPSTA011*** and perhaps ***PIPSTA012*** for a full overview of system functionality.

Warning

Although this is a primitive implementation, it is quick, simple to understand and portable between web hosting companies. This is not necessarily the most robust nor secure implementation, so please be aware of these limitations and DO NOT use this as the basis for anything other than a demonstration system!

Prerequisites

- It is expected that you have a fully-working, web-connected Raspberry Pi and Pipsta
- You should have the following settings from your web hosting company by having followed ***PIPSTA010- Simple Web Printing – Pre-requisites*** :
    - Username
    - Password
    - Host IP address
    - Database name
    - Port

How it Works

Database Functions

The Python script uses the module MySQLdb to communicate with the web-hosted MySQL database created in ***PIPSTA010***. The database functions are:

- **connect_to_db()** – which passes the DB_CONFIG dictionary of named arguments pertaining to your web-hosted database.
- **read_uid()** which returns the latest **job_id + 1**, i.e. the next available ID
- **insert_data()** – which adds the data into the database, populating the two fields **printdata** and **printer_id** at the 'cursor' position (effectively at the next available **job_id**). The **printer_id** is simply the printer serial number on the underside of the paper cup (the number beginning with G, visible through the back plastics or produced by double-clicking the button next to the printer LED.) Clearly, sending a job *from* this Pipsta to the *same* Pipsta over the web is a little inefficient, but it serves to 'close the loop' for the purposes of the demonstration. If you have more than one Pipsta, and both are connected to the same database, you should then be able to send print jobs from one to the other, anywhere in the world!

Other Functions

It should be discernible that the above functions do nearly all the required work to add the print data to the database. The additional requirements are:

- **parse_args()** takes two command line arguments: **printer_id** (i.e. the serial number visible on the underside of the paper cup) and **file** (the path and filename containing the data to be printed.)
- **binascii.hexlify()**, a part of the binascii module, converts standard data to a string of ASCII-encoded hex digits. Strictly speaking, **WebSend.py** permits only simple text in the range ASCII 0x20 to 0x7F, and therefore this is over-specified. However, it should be noted that this implementation allows the range ASCII 0x00 to 0xFF, and therefore is suitable for advanced graphical commands whose payload data bytes require this full range.

Extending Functionality

In principle, graphical data output from other scripts could be directed to your database instead of directly to your local printer. Consider some of the following:

- Banners
- QR-Codes
- Receipts (with logos)
- Certificates
- Food orders

Furthermore, it is possible to develop the code to print one type of job locally and send different but related data to the cloud. This allows –for example—a long-form customer receipt to be produced locally with a short-form variant being sent to another printer for audit purposes.

We will describe some other opportunities for enhancing functionality in the tutorials:

- ***PIPSTA013 – Modifying Web-Printing to a Pool of Printers*** and
- ***PIPSTA014 – NFC Secure Printing***

Shutting Pipsta Down Safely

Whilst the printer is resilient when it comes to powering down, the Raspberry Pi must undergo a strict shutdown process to avoid corrupting the Micro SD card. The most straightforward method of doing this is to double-click the 'Shutdown' icon on the desktop.

If you are already in LXTerminal, type **sudo shutdown –h now** to shut-down the Raspberry Pi immediately.

Always make sure ALL activity on the Raspberry Pi's green LED (the LED on the right) has stopped before removing the power!

■End of Document■