PUTTY *DRIVER*

INTRODUCTION

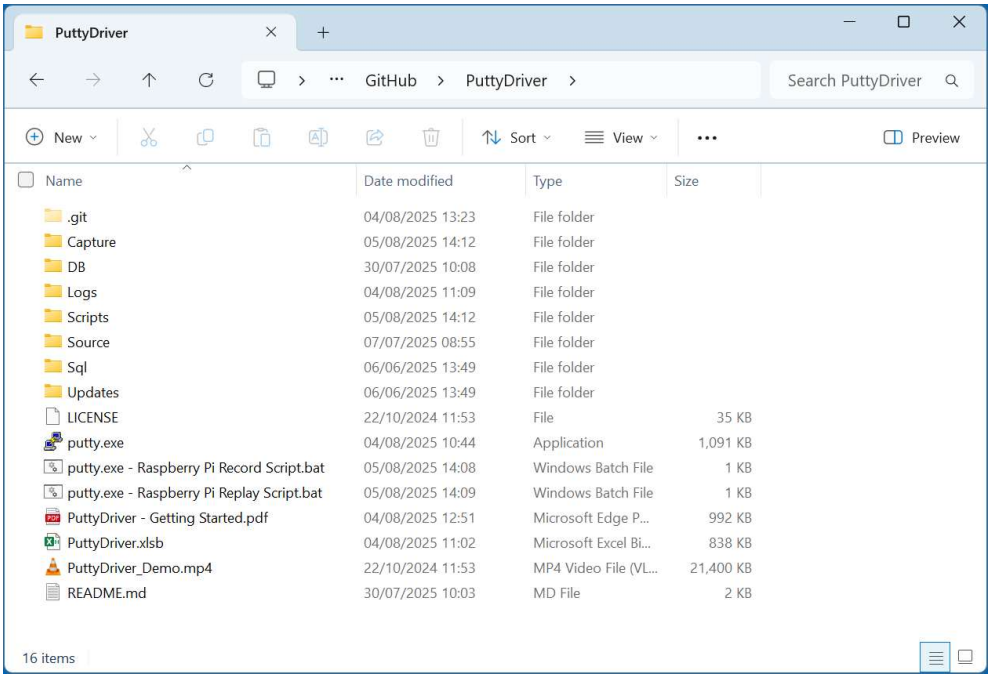
PuttyDriver[(v2)] adds automation and robotics functionality to the popular open source PuTTY SSH and Telnet client developed by Simon Tatham and others. Any server/operating system/application accessible via SSH (or Telnet) should work.

PuTTY screens and ***input commands*** are captured automatically. ***Scripts*** can be recorded, edited and replayed to assist with automation of Legacy Applications and Legacy Systems Administration, via a standard SSH or Telnet connection.

Centos7/Informix 4GL and Raspberry Pi OS example files accompany this document (See ***Scripts*** and ***Capture*** folders). PuttyDriver uses 'expected' PuTTY screens ***text*** and ***cursor positions***, to control execution of PuTTY input commands.

PuttyDriver version of ***putty.exe***, is a modified 0.83 version of PuTTY with a small amount of additional C code, to provide the scripting and message queue interface. Project files can be found at https://github.com/christyler80/PuttyDriver.

To get started, download the Zip file from Putty Driver GitHub project and extract into a folder, for example:-



PuttyDriver currently consists of two application files - ***PuttyDriver.xlsb*** and ***putty.exe,*** together with a SQLite database, for which ODBC driver for SQLite must be also installed and can be downloaded from here.

Excel workbook ***PuttyDriver.xlsb*** is intended to assist design, record and test ***Scripts,*** and also interfaces with the provided ***PuttyDriver.db*** SQL database (SQLite), to schedule and manage PuttyDriver ***Sessions*** across multiple ***Servers*** and ***Scripts***.

See the PuttyDriver Getting Started section of this document for details of how to setup and start running PuttyDriver.

There is also a brief mp4 video that shows PuttyDriver being used with ***PuttyDriver.xlsb*** [(v1)].

At this time:

- 32-bit or 64-bit versions of Microsoft Windows 10 operating system or later are supported.
- Excel 2019 and later versions are supported. Earlier Excel versions may work but have not been tested.
- Other operating systems (e.g., Apple, Linux) are not currently supported.
- Putty Driver includes a SQLite database (***PuttyDriver.db***). Any SQL database should work, e.g. SQL Server.
- Putty Driver uses DB Data Manager for the database interface.

Chris Tyler, Summer 2025
www.christyler.uk

PUTTYDRIVER GETTING STARTED

1. Download the Zip file from Putty Driver GitHub project (PuttyDriver-main.zip) and extract into a folder.
2. To test, run PuttyDriver ***putty.exe*** and connect to a server using SSH (or Telnet) via IP or Server Name.
3. Check that login and password are working OK and close ***putty.exe***.

If the test has worked, the Putty Driver version of ***putty.exe,*** can now be used to record new scripts and replay existing scripts – see PuttyDriver Command Options and Quick Start sections below.

Alternatively, the ***PuttyDriver.xlsb*** workbook can be used to record or run scripts (using ***putty.exe***) and interface with the provided SQLite ***PuttyDriver.db*** database (see ***DB*** folder), for managing multiple PuttyDriver ***Sessions*** across multiple **Servers** and ***Scripts*** (see PuttyDriver Excel Workbook section of this document below for details).

PUTTYDRIVER COMMAND OPTIONS - QUICK START

PuttyDriver version of ***putty.exe***, adds the following to the standard PuTTY command line options.

-recordscript	-- switch to turn script recording ***On***.
-script <full path\file name>	-- run specified Script using ***putty.exe***.
-capturefile	-- specify the capture file name (Optional). If folder path is not specified, files will be saved in the '***Capture***' folder.
-nocapture	-- run script with capture turned off.
-keycodesfile	-- custom KeyCodes.txt file (see Scripts folder for default file).
-logfile	-- Custom log file name (see Logs folder).
-nolog	-- run script with no logging.
-screenspeed	-- Screen speed between commands (milliseconds) or word 'slow' (100 milliseconds).

RECORD SCRIPT - QUICKSTART

Using the Windows Command Prompt, ***cd*** into the folder where the PuttyDriver files have been extracted and run the following command (see other ***Command Options*** above) with your ***login***, ***server*** (or IP), ***password*** and ***filename*** details:-

putty.exe -ssh <login@server> -pw <password> -recordscript -capturefile <filename>

e.g. putty.exe -ssh pi@raspberrypi.home -pw raspberry -recordscript -capturefile pi_script_#1

Enter a few commands (e.g. **pwd**, ***ls***), close the putty.exe session and if PuttyDriver has worked correctly, see new files ***pi_script_#1_yymmdd_hhmmss.inputs*** and ***pi_script_#1_yymmdd_hhmmss.capture*** in the ***Capture*** folder.

RUN SCRIPT - QUICKSTART

To execute script *pi_script_#1* recorded in the example above, rename ***pi_script_#1_yymmdd_hhmmss.inputs*** *to* ***pi_script_#1.script*** run the following command and see new files in the ***Capture*** folder:-

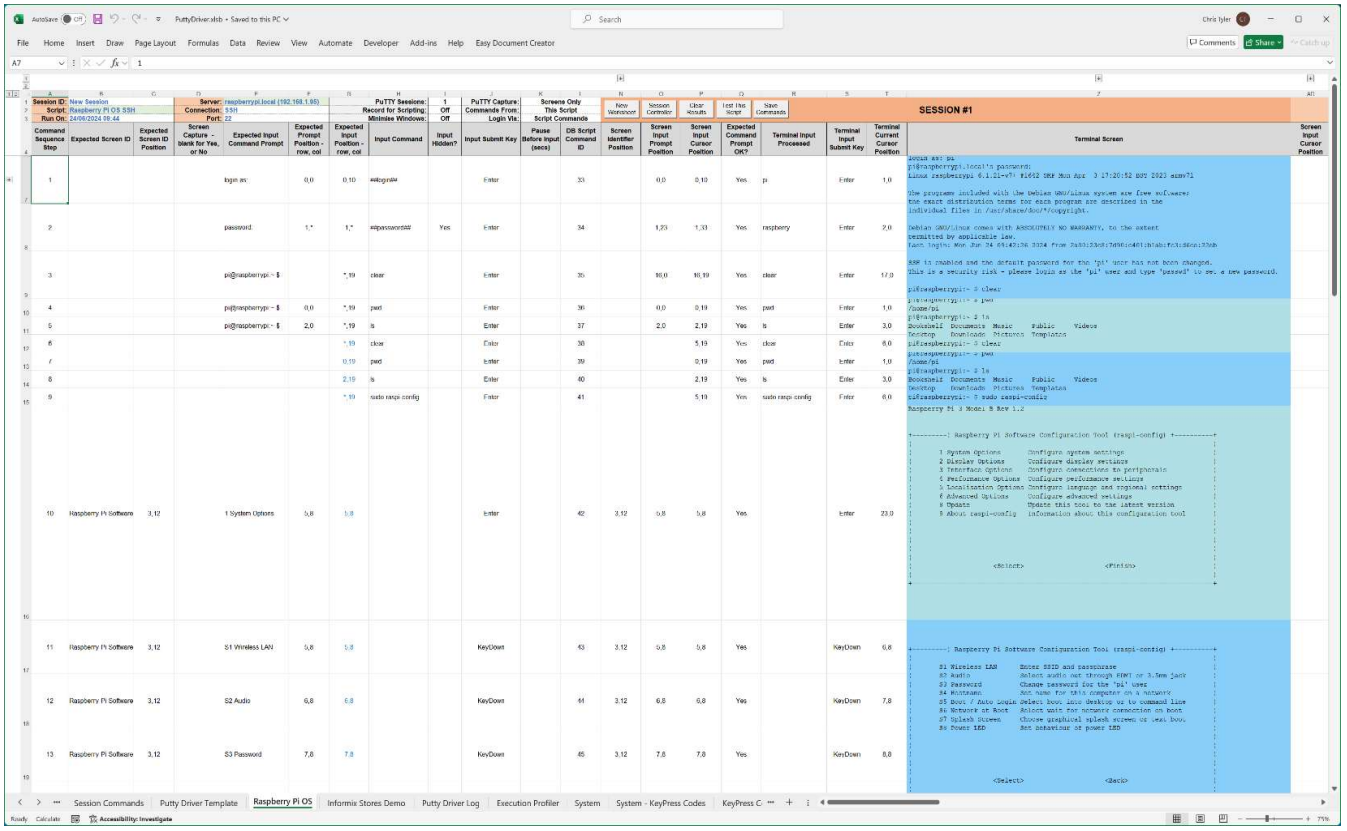
putty.exe -ssh pi@raspberrypi.home -pw raspberry -script Capture\pi_script_#1.script

IMPORT SCRIPT AND CAPTURE FILES INTO MICROSOFT EXCEL

Use the '***Load Script From File'*** button on the PuttyDriver.xlsb worksheets (see sections below for details) to import PuttyDriver ***script***, ***inputs*** or ***capture*** files. These worksheets can be used to design, modify, save and run scripts for testing. PuttyDriver.xlsb also contains functionality for managing ***servers, scripts*** and scheduling/running of ***scripts***.

PUTTYDRIVER EXCEL WORKBOOK - PUTTYDRIVER.XLSB

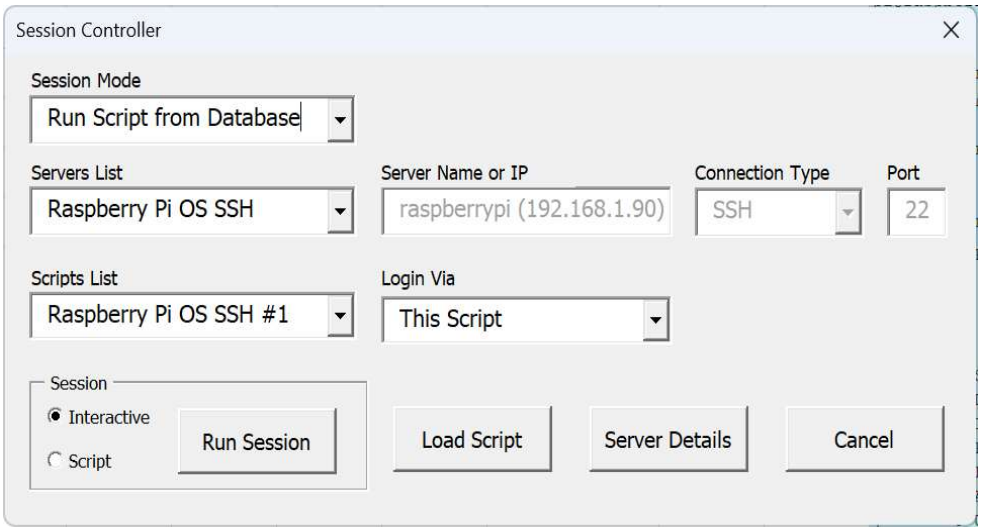
1. ***Raspberry Pi OS*** worksheet image below, shows the results from the ***Scripts\raspberrypi*** sample script, run on a Raspberry Pi with Raspberry OS installed.
2. ***Script*** commands are stored in columns A->K. Columns L onwards, store the PuttyDriver results, including details of commands processed and PuTTY screens captured.



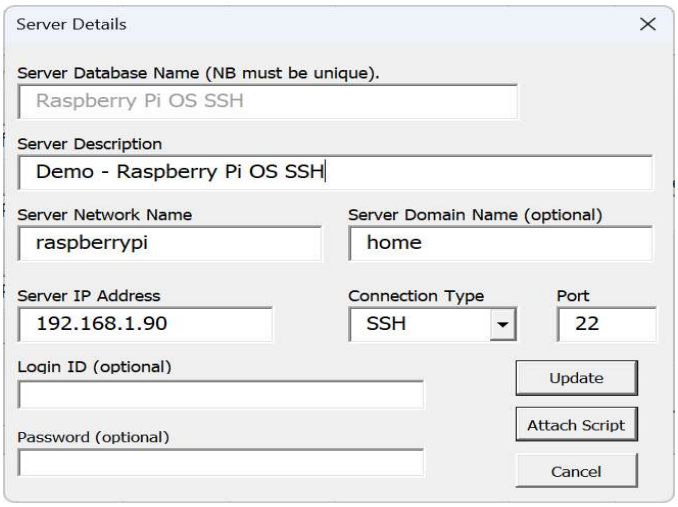
3. PuttyDriver ***Session Controller*** provides forms for registering ***Servers*** with the PuttyDriver database, creating ***Scripts*** and attaching ***Scripts*** to ***Servers*** (see Sessions Controller and Database and System sections below).
4. ***New Worksheet***, ***Clear Results***, ***Load Script***, ***Test Script*** and ***Save Commands*** Controls allow ***Scripts*** to be recorded/created, modified and tested from within the ***PuttyDriver*** worksheets.
5. Using the 'Test This Script' button is recommended for recording and replaying script commands for new scripts.
6. ***Expected Screen ID/Position***, ***Expected Input Command Prompt/Position*** and ***Expected Cursor Position*** control how ***putty.exe*** executes user inputs, so that the correct inputs are processed by ***putty.exe*** only when the expected prompts, have been found at their expected positions on the expected PuTTY screen.
7. Script Commands should be as precise as possible, by specifying exact ***Expected Screen ID/Position***, ***Expected Input Command Prompt/Position*** and ***Expected Cursor Position*** for every user input.
8. '*' value can be used when exact row or column cannot be specified. However, these should be kept to a minimum.
9. When running scripts, PuTTY screens are captured automatically. PuTTY cursor position, user inputs and screen text immediately before each user input, are also captured automatically.
10. After testing a new script, Use the 'Save Commands' button to save the script commands for future use.
11. PuttyDriver settings (see rows 1->3) which control how ***putty.exe*** is run are set automatically, but some can be overridden. For script development, setting ***Putty Run Mode*** to ***Interactive*** can be useful when testing scripts.
12. ***Important:*** Thorough testing of ***Scripts*** and monitoring of ***Scripts*** execution are both vital and 100% the responsibility of users at all times and in any environment. See **License** section at the end of this document.

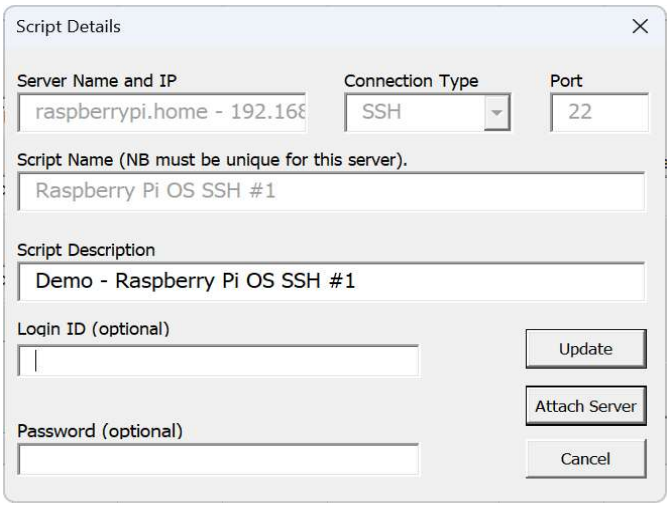
PUTTYDRIVER EXCEL WORKBOOK – SESSIONS CONTROLLER

1. The ***Sessions Controller*** form provides access to forms for registering ***Servers*** and new ***Scripts*** with the PuttyDriver database, attaching ***Scripts*** to ***Servers***, running ***Sessions*** and other options for designing and testing scripts.



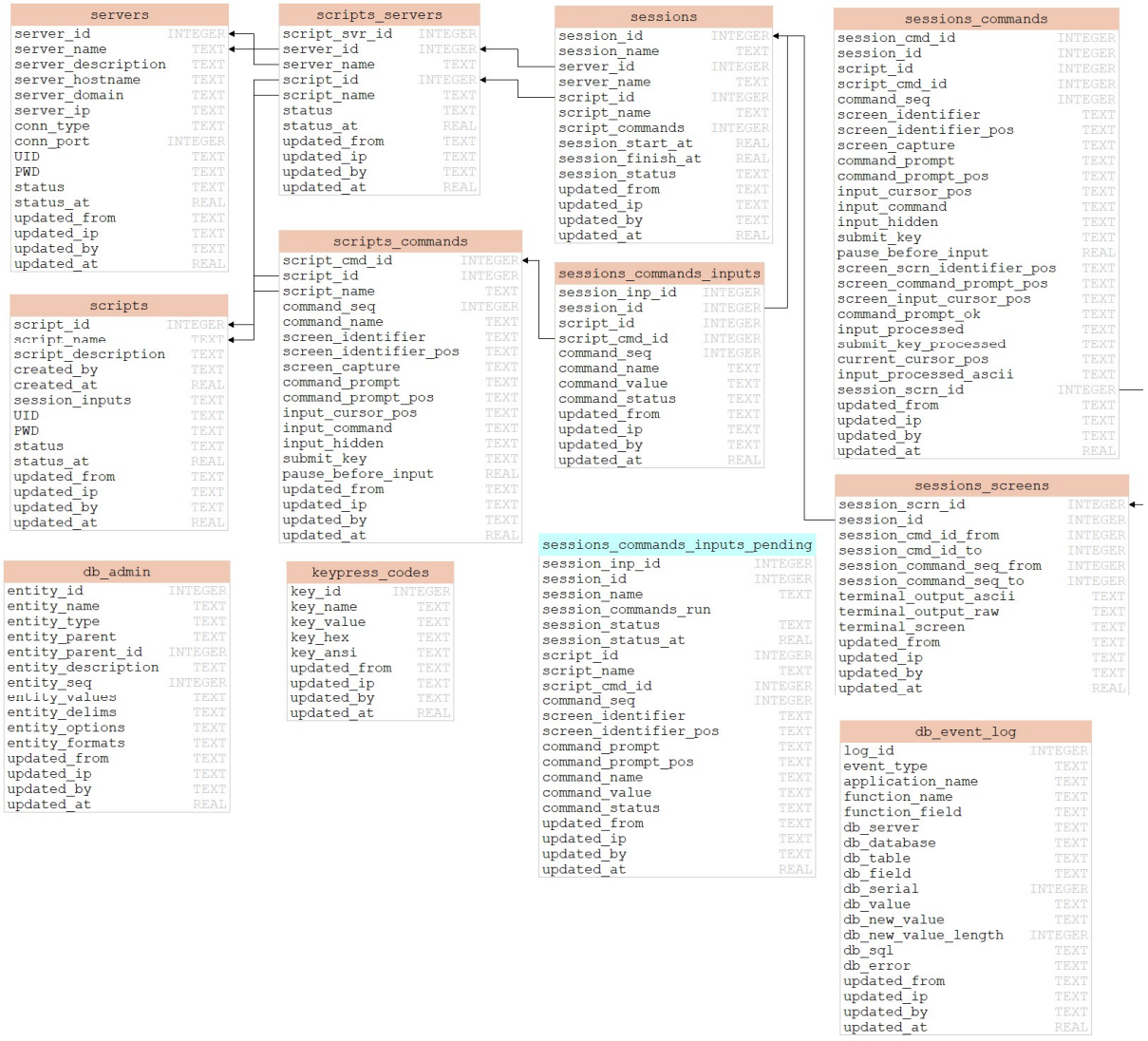
2. ***Please Note:*** 'Interactive' mode is currently only available with 32-bit versions of Excel.
3. ***Sessions Controller*** form 'Load Script' button allows ***Scripts*** and ***Screen Capture*** files to be loaded into PuttyDriver.xlsb worksheets. 'Run Session' with run the Script for the Server shown.
4. The ***Server Details*** button and ***Servers List*** control provide access to the ***Server Details*** form. Clicking on the ***Scripts List*** control provides access to the ***Script Details*** form (see form images below).





5. Data from these forms is stored in PuttyDriver database tables ***servers*** and ***scripts***. Table ***servers_scripts*** links ***scripts*** to individual ***servers***. See Database and System sections below for an overview of the PuttyDriver database.
6. ***Server Details*** form allows the server settings (e.g. IP Address) to be validated before being saved to the PuttyDriver database using the ***Update*** button.
7. As already discussed, Script execution can be controlled using the '*Expected Screen ID*', '*Expected Screen ID Position*', '*Expected Input Command Prompt*', '*Expected Cursor Position*' and '*Expected Cursor Position*' worksheet columns.
8. See the 'Raspberry PI OS' worksheet for examples of how the '*Expected*' fields can be set and use of '*' wildcard for Cursor position, where either X or Y are known but not both (e.g., after running Linux ***ls*** (list files) command).
9. ***Scripts*** and ***Session*** statuses of ***created***, ***scheduled***, ***completed***, ***started*** and ***failed*** are supported by PuttyDriver.
10. PuttyDriver Scheduling functionality is being developed and expected from Q4 2025 via a separate .NET application.

1. PuttyDriver database structure is summarised below and see SQL files in the ***DB*** folder for full schema details.



2. PuttyDriver can be integrated with existing systems, by writing data directly into the ***servers***, ***scripts***, ***scripts_servers***, ***scripts_commands***, ***sessions*** and ***sessions_commands_inputs*** tables, or by running externally generated scripts via the Linux command line - see PUTTYDRIVER COMMAND OPTIONS section above.
3. Data can also be loaded into the PuttyDriver database and/or amended, using the DB Data Manager worksheet 'DB Data Updates' that is included in ***PuttyDriver.xlsb*** workbooks.

LICENSE