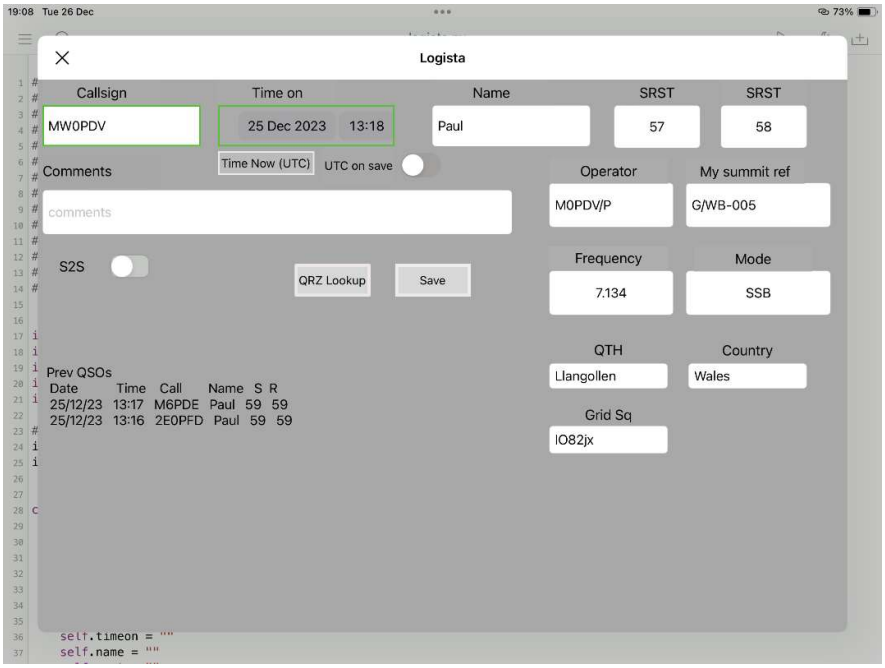
Logista logging tool for SOTA activations



MW0PDV

12/28/2023

Logista logging tool for SOTA

Contents

Disclaimer

Pythonista is an enabler for using Python code on an iPad, but does have a drawback in that Pythonista app is not free. I have no affiliation or connection with Pythonista, it's just an app that allows me to write Python code on an iPad with access to the IOS user interface.

Preamble

Logista is an open source logging tool for SOTA activations that is intended to run on an iPad under Pythonista 3. I am sharing the Logista Python code in the event that someone may find it useful.
I wrote Logista primarily to fit my preferred SOTA activation, logging workflow. That is using a paper log and then typing it up once at home. However, I still wanted to maintain the ability to log live on an activation and so Logista also offers this functionality. I have been using Hamlog on my iPad for a while and this has heavily influenced the style of Logista.
The name, is a little corny, Logista is a portmanteau word of logging and Pythonista.

Before we get into Logista, what is Pythonista

Ok so why do I need to have the Pythonista app installed on my iPad?
Well, the Logista code is written in Python and Pythonista 3 is a Python IDE (Integrated Development Environment), that allows Python 3.10 code to run on an iPad, with access to the iOS user interface.
If you have ever tried to find a Python IDE for iOS on an Ipad, you will have almost certainly come across Pythonista.
Pythonista also allows you to run Python code from a shortcut on your iPad desktop and this is how I prefer to run Logista on my iPad.

Logista functionality

The Logista logging tool is intended to assist in the creation of a SOTA activation log and creates two files. These files are a SOTA v2 CSV file for uploading to the SOTA database, and also an ADIF log file for importing into your preferred logging program. Logista is not a full blown logging program in that it only creates the files and currently has no log browsing or query features. Logista does have a scrolling window of previously logged QSO's. If logging live you can set Logista to use the current UTC time as the record is saved, but more in time feature below.

If QRZ has been setup in the cfg.txt file, Logista will perform an auto callsign lookup once Enter is pressed, or you click into another text field. The border of the callsign text field will turn green to confirm that the callsign has been entered. If the QRZ lookup is successful the name, QTH, country and grid square will be auto populated. Entry of remaining fields is manual.

When the form is saved QSO specific fields will be cleared. Your summit, operator callsign, frequency and mode will not be cleared. This speeds up QSO entry as only the callsign, time and reports need to be entered for each QSO.

The date picker is setup so that you only have to roll the time on minute by minute. This is to reflect how the time is presented on a paper log. If, however you wished to log in real time you can use the UTC time now button to set current UTC time or switch on the UTC on save to use the current UTC time at the time of saving. Also note that the date picker border will also turn green to confirm that a date has been entered unless you are using UTC time on save.

Summit to Summits can be entered by clicking the S2S switch to on, this will then reveal a text field for the other summit, the S2S switch resets on save.

I have also added a Last QSO, text window to help keep track of the last QSO's. They will eventually scroll off the bottom of the window but you can drag the window up and down if required. Previous QSO's are not reloaded when Logista restarts as they are only stored in memory. Logista starts each session with new SOTA.csv and Logista.adi files.

The SOTA(x).csv and Logista(x).adi files are written to an iCloud sub directory of Pythonista3 called Log_Files. If this directory does not exist it will be created. At the start of each session, Logista checks to see if SOTA0.csv and Logist0.adi exist. If they do not exist, they will be created, if they do exist Logista will increment the number at the end of the file name until it finds a file that doesn't exist. Of course, for good housekeeping it is recommended to regularly delete old files. I always find it helpful when uploading a SOTAx.csv file to the SOTA database to select iCloud files, then select recent in file explorer, and the newest files should be the first file. The SOTA.csv also populates the notes section with the first name from the QRZ lookup in the notes section on SOTA csv. The notes section in the form is only written to the ADIF file.

Installation

Please note that although I have tested and tried to break things, this Logista version is still very much a beta release. I have not be able to fully test installation on other iPads and so any feedback on installation would be appreciated.

The required Logista files can be found on my GitHub page here;
https://github.com/pdevlin13/Logista.git

Installation assumes that you already have Pythonista installed and working on your iPad.

The required files are:

Logista.py	the main code file.
sota_log_form.pyui	the Pythonista GUI file called from Logista.py
qrz_lookup.py	the QRZ lookup code called from Logista.py
cfg.txt	the configuration file for setting up QRZ

all of the required libraries should be packaged as standard in Pythonista.

You can download the files directly from GitHub, all of these files must be in the same directory on your iPad. I find Microsoft OneDrive helpful for transferring and syncing files between my PC and iPad. On my iPad I save OneDrive files to IOS files and vice versa.
You can open the files from within Pythonista, you actually only need to open Logista.py and run it.
If you wish the use the QRZ lookup and you have a QRZ XML subscription you will need to edit the cfg.txt and add your QRZ credentials. You can edit this by opening in the file in Pythonista.
When you run Logista a file called cache.txt is created, this file caches the QRZ session key.

Example screen recordings

Example screen recording of retrospective logging

Example screen recording Live logging

Screen Shots of Logista

Figure 1 Logista screenshot

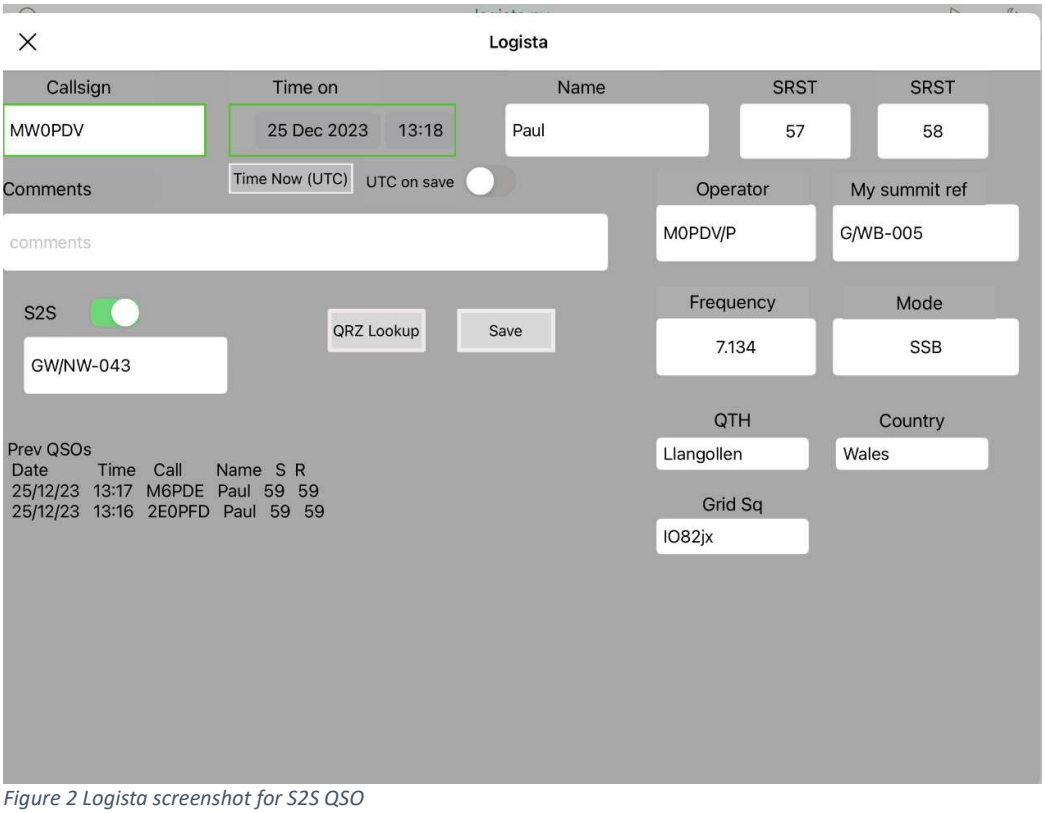


Figure 2 Logista screenshot for S2S QSO