Starfeeder

Whisker Starfeeder: README

Purpose

Manages radiofrequency identification (RFID) readers and weighing balances, and talks to a Whisker client (http://www.whiskercontrol.com/).

Author/licensing

Single-folder binary distribution

Unzip the distributed file and double-click the `starfeeder` program. That's it.

Linux source installation

End users should opt for the single-folder binary distribution instead.

Install

From a command prompt:

```
sudo apt-get install python3 python3-pip  # install Python with pip
python3 -m virtualenv /PATH/TO/MY/NEW/VIRTUALENV  # make a virtualenv
source /PATH/TO/MY/NEW/VIRTUALENV/bin/activate  # activate the virtualenv

# pip install starfeeder --process-dependency-links  # install from PyPI -- NOT YET IMPLEMENTED

cd /MY/WORKING/DIR
git clone https://egret.psychol.cam.ac.uk/git/starfeeder  # Fetch code. Private for now.
pip install -e .  # Install from working directory into virtualenv.
```

Run

```
/PATH/TO/MY/NEW/VIRTUALENV/bin/starfeeder
```

Windows source installation

Deprecated, as it's complex.

Install

1. You need to have Python 3 installed (which will come with `pip`, `pyvenv`, and sometimes `virtualenv`). Obtain it from https://www.python.org/ and install it. We'll suppose you've installed Python at `C:\Python34`.
2. On Windows 10, install a copy of `cmake`, because PySide wants it. Also Qt. Also Git if you want to work with repositories directly. Possibly other things. (I have this working on Windows XP but not Windows 10; PySide is not building itself happily.)
3. Then fire up a Command Prompt and do:

```
C:\Python34\Tools\Scripts\pyvenv.py C:\PATH\TO\MY\NEW\VIRTUALENV

C:\PATH\TO\MY\NEW\VIRTUALENV\Scripts\activate

pip install starfeeder --process-dependency-links
```

Run

Run the `starfeeder` program from within your virtual environment.

Windows: just the GUI

For normal use:

```
C:\PATH\TO\MY\NEW\VIRTUALENV\Scripts\pythonw.exe C:\PATH\TO\MY\NEW\VIRTUALENV\Scripts\starfeeder-script.py
```

Windows: to see command-line output

Use this for database upgrades, command-line help, and to see debugging output:

```
C:\PATH\TO\MY\NEW\VIRTUALENV\Scripts\starfeeder
```

You can append `-v` for more verbose output, or `--help` for full details.

If you use this method to run the graphical user interface (GUI) application, **do not** close the console window (this will close the GUI app).

Changelog

v0.1.2 (2015-12-23)

- Initial release.
- Hardware tested via Windows XP, Windows 10, and Ubuntu 14.04.

v0.1.3 (2015-12-26)

- Ugly `moveToThread()` hack fixed by declaring `QTimer(self)` rather than `QTimer()`.
- More general updates to declare parents of `QObject` objects, except in GUI code where it just clutters things up needlessly. Note that `QLayout.addWidget()`, `QLayout.addLayout()`, and `QWidget.setLayout()` all take ownership.
- Bugfix related to using lambdas as slots (PySide causes a segmentation fault on exit; https://bugreports.qt.io/browse/PYSIDE-88).
- Launch PDF manual as help.
- Retested with hardware on Windows XP and Linux.

v0.1.4 (2015-12-26)

- callback_id set by GUI, not by derived classes of SerialOwner

v0.1.5 (2016-02-27)

- bugfix to BaseWindow.on_rfid_state()

v0.2.0 (2016-04-07)

- GUI log window, for PyInstaller environments.
- Uses Whisker Python library.
- Switch to Arrow datetimes internally.
- Bugfix in error handling when trying to open non-existent serial ports.

v0.2.3 (2016-04-19) v0.2.4 (2016-04-19)

- Bugfix.

v0.2.5

- Internal changes only?

v0.2.6 (2016-11-24)

- Python type hints.
- NOTE that to install Python 3.4 (required for this version of PySide) under Ubuntu 16.10, you need to: - download Python 3.4.4 source, then:

```
$ tar xvf Python-3.4.4.tgz
$ cd Python-3.4.4
$ configure --enable-shared
$ make
$ sudo make install

    # now unbreak wrong symlink and replace with old:
$ sudo rm /usr/bin/python3  # "make install" made this point to python3.4
$ sudo ln -s /usr/bin/python3.5 /usr/bin/python3

    # now set up library links
$ sudo ln -s /usr/local/lib/libpython3.4m.so.1.0 /usr/lib/x86_64-linux-gnu/libpython3.4m.so.1.0

    # this should now work:
$ python3.4
```

- Upgraded from pyserial 3.0.1 to 3.2.1 … also allows the use of Linux pseudoterminals for testing; http://stackoverflow.com/questions/34831131
- Passwords obscured in debug-level database URLs.
- Top-level exception tracebacks go to log (like all others), not to print() using traceback.print_exc().
- BalanceController could send 'ICRNone', which is wrong; the frequency 10 Hz was offered in the dialogue, but should have been 12. Validity check added.
- Bug workaround: PROBLEM - sometimes, `WeightWhiskerTask.on_mass()` received something that was not a `MassEvent`. Not sure why (it doesn't look like anything else is ever sent); could this be a PySide signals bug?

  ATTEMPT 1 - Workaround is to verify type on receipt (and complain loudly if wrong but ignore/continue). - … no; irremediable bug in PySide (see development notes); it fails to keep references to signal parameters, so sometimes they go AWOL.

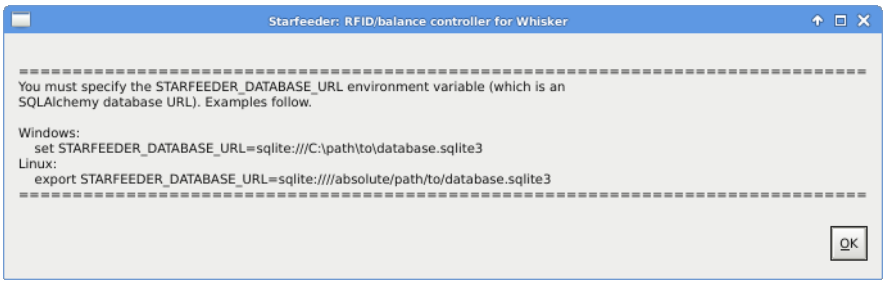
  ATTEMPT 2 - Switched from PySide to PyQt5, and thus GPLv3 licensing. - Generally, this seems much better. - Even then, apparent corruption in "bytes" object passed from SerialController.process_data() -> SerialController.line_received -> [change thread] -> RfidController.on_receive Sometimes the received bytes object is b", not what was sent. PyQt does some sort of autoconversion to C++ objects; see http://pyqt.sourceforge.net/Docs/PyQt5/signals_slots.html ; and the problem appears to go away by using an encapsulating Python object… Not ideal! Does it also affect str? No, str seems OK. BUG REPRODUCED RELIABLY in pyqt5_signal_with_bytes.py. Reported to PyQt mailing list on 2016-12-01. SO FOR NOW: AVOID bytes OBJECTS IN PyQt5 SIGNALS.

Installation

To install, you just unzip it and run the `starfeeder.exe` executable.

Database setup

When you first run Starfeeder you might see this:

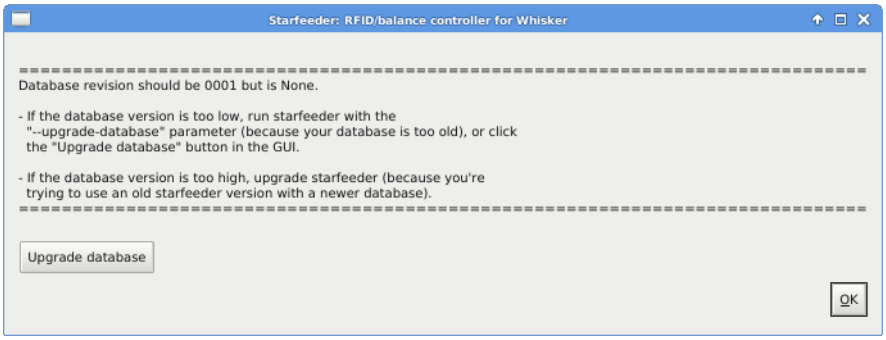


Before it'll work you'll also need to set an environment variable to point to a database. The simplest would be to use SQLiteh(ttps://www.sqlite.org/), like this:

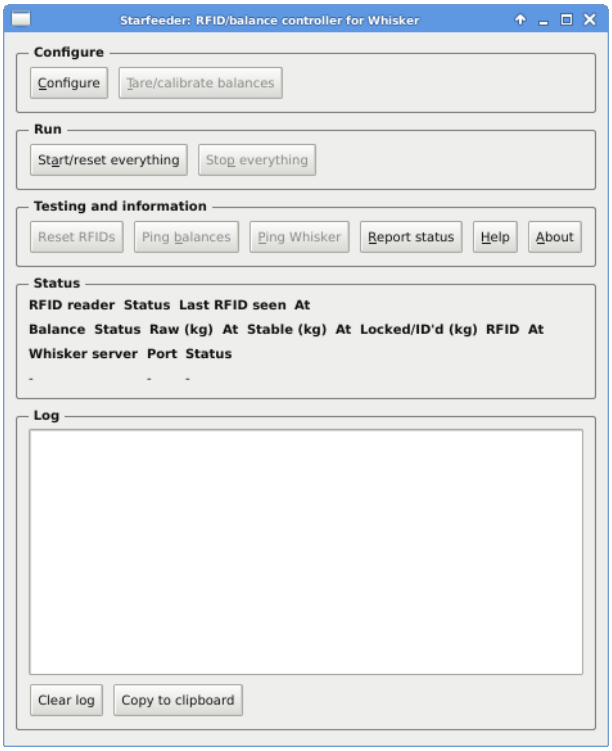
```
set STARFEEDER_DATABASE_URL=sqlite:///C:\path\to\starfeeder_database.sqlite3
```

… and then run Sqliteman (http://sqliteman.yarpen.cz/) to inspect the database. But the software supports other databases (e.g. MySQL, others: http://docs.sqlalchemy.org/en/latest/core/engines.html) via this URL scheme. The configuration and data are stored in this database.

If the database isn't structured correctly, as it won't be if you've just invented the URL (so a blank database is autocreated), the first time you run it you get a button to "Upgrade database" (then close and re-run).

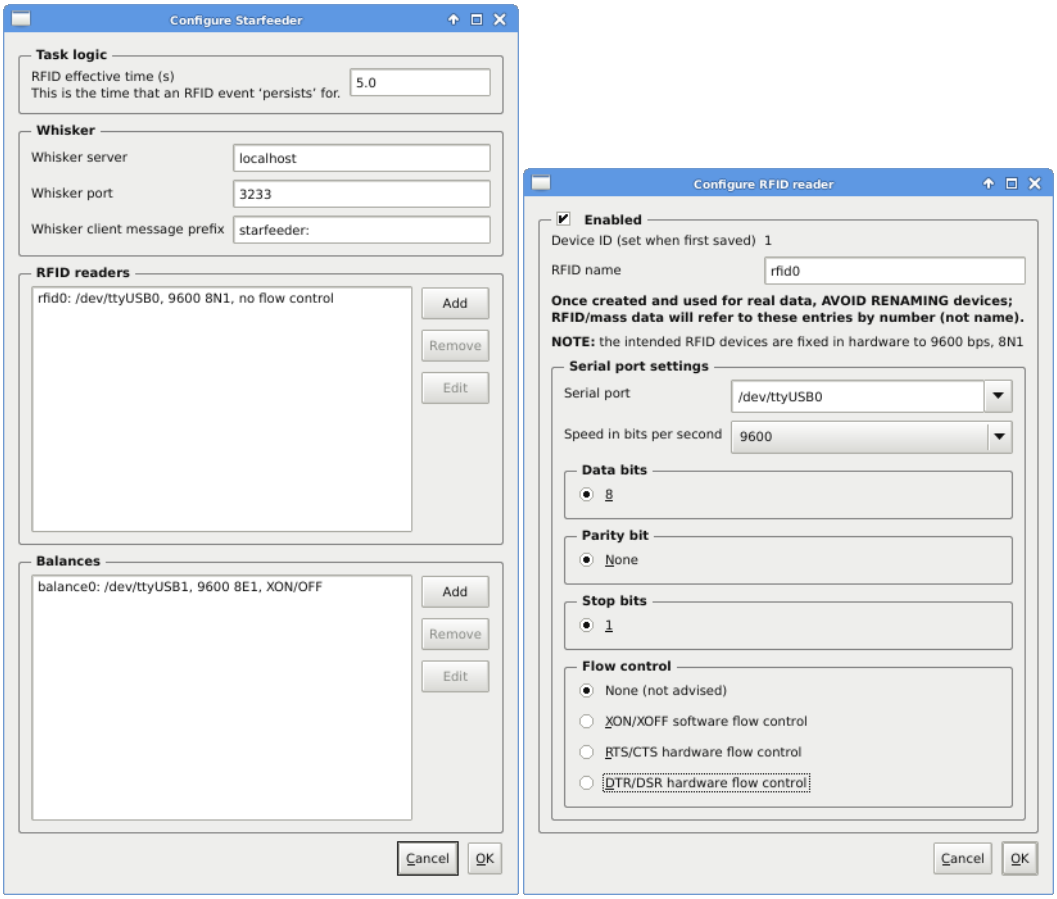


Here's the happy starting screen:

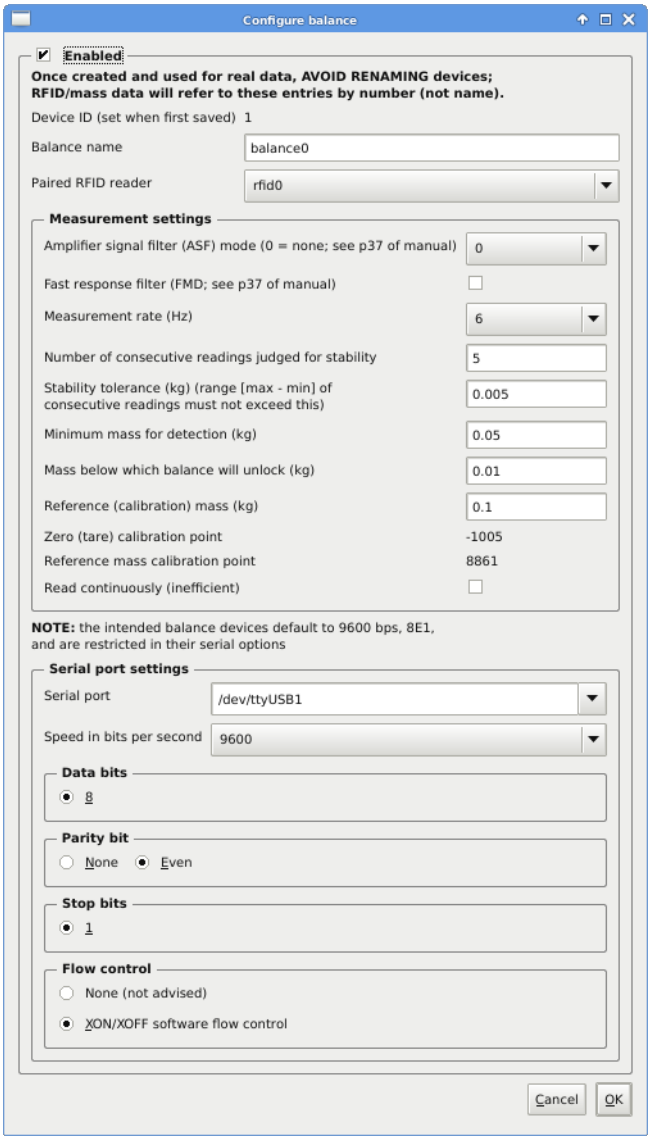


Configuration

Configure like this:



I am not completely sure if the RFID reader supports any flow control. Under Linux, XON/XOFF and RTS/CTS don't break it (but that doesn't mean it uses them). Under Windows, RTS/CTS breaks it. The physical serial interface inside it is RS-485, which is two-wire and therefore not only can't have RTS/CTS but is unidirectional. I don't know if the USB interface adds anything; I suspect not. So the best option is either XON/XOFF or None – probably None.



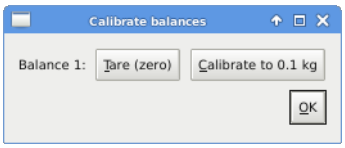
The balance says (p15 of its manual) that it copes with XON/XOFF.

When you click “Start…” it starts the RFID readers, balances, and connects to Whisker.

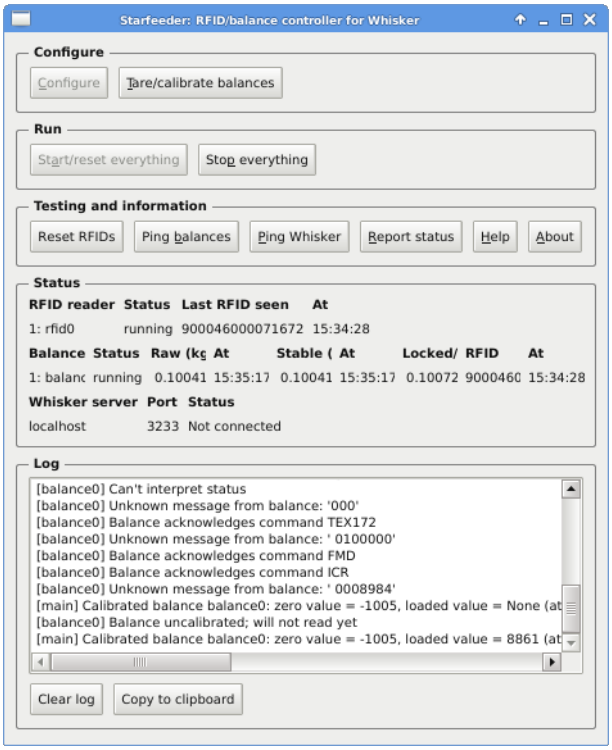
Data storage and output

All data goes to the database (as well as to Whisker if connected).

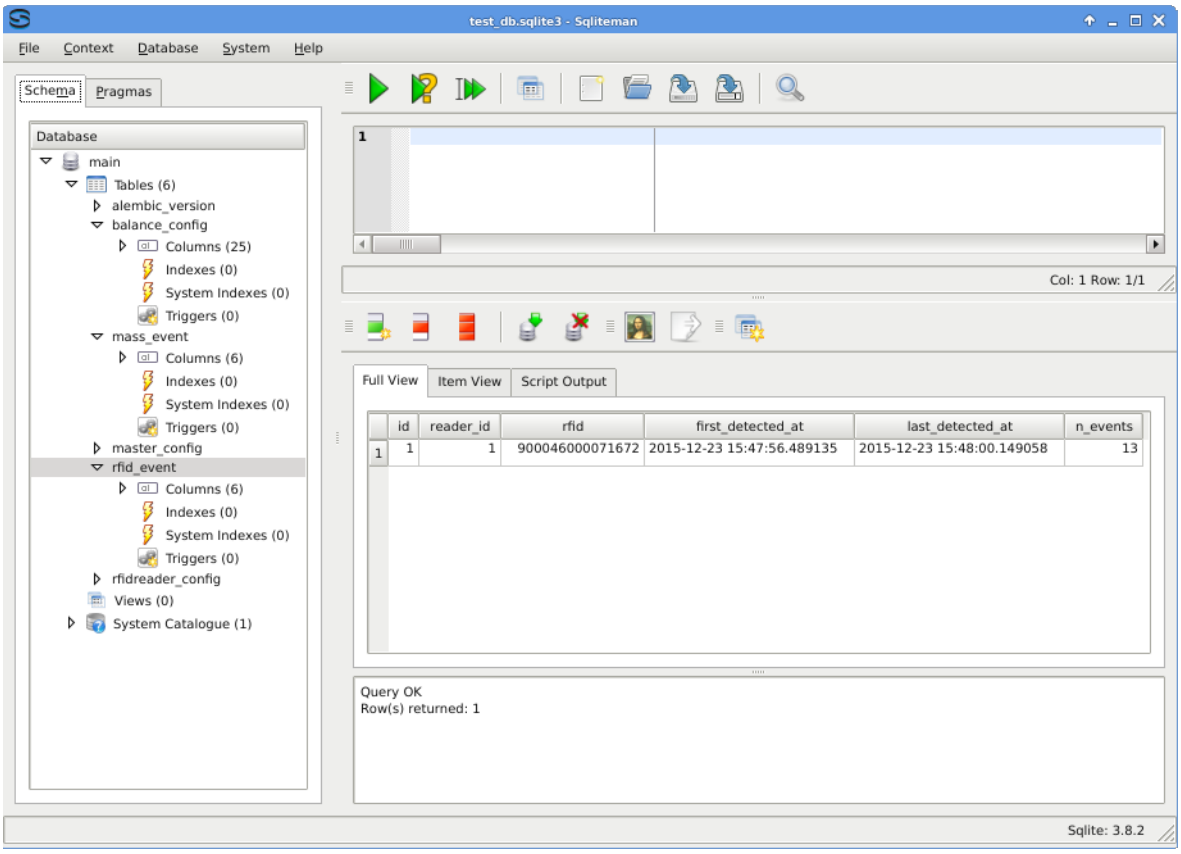
The balance won’t produce numbers until it’s tared and calibrated. The default is for an 0.1kg calibration mass, so you just start; click the Tare/calibrate button; then remove all mass and click “Tare”, then put the 0.1kg mass on and click “Calibrate 0.1kg”. It’ll store the readings so you don’t necessarily have to recalibrate next time.

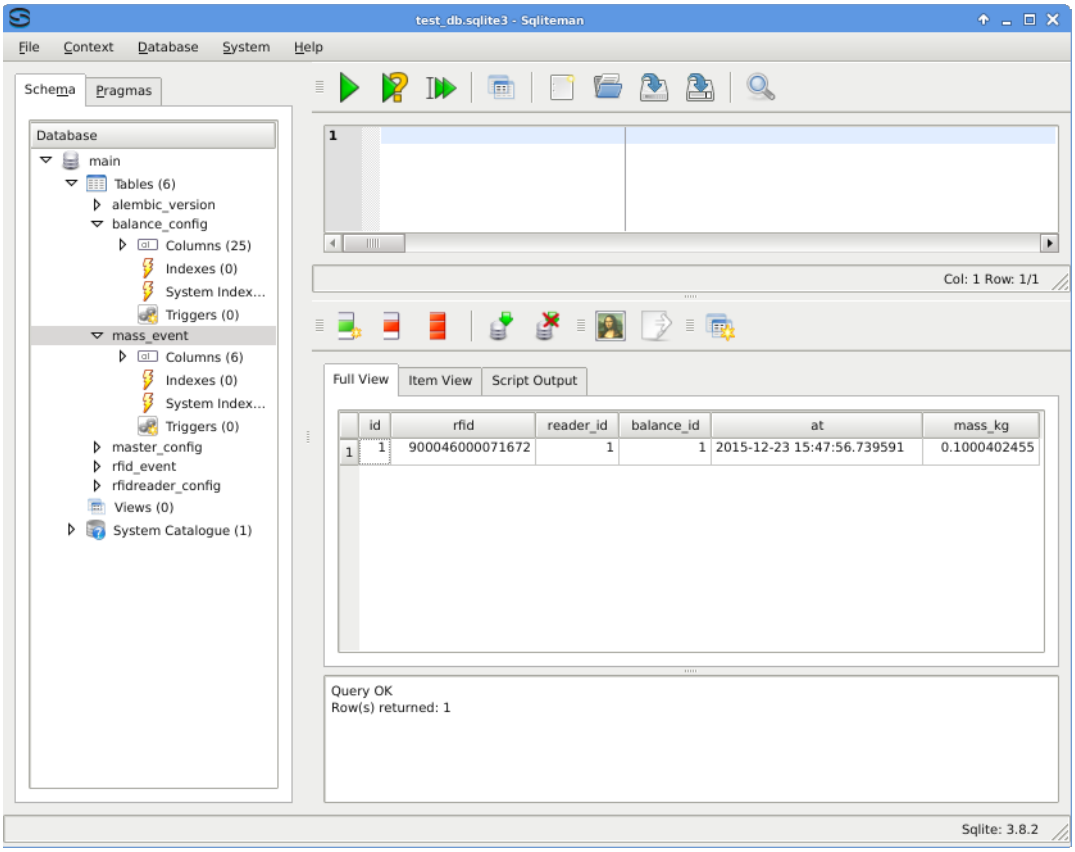


When it’s running and things are happening, it looks like this:

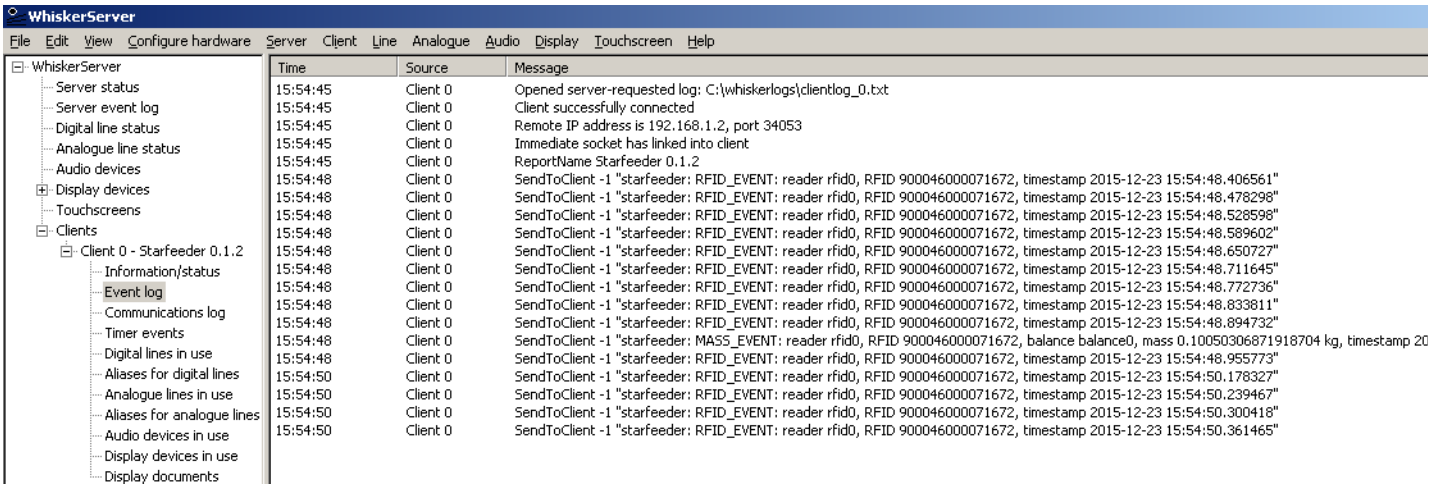


Here are some examples of the output, as seen by Sqliteman:





… and by Whisker:



Operating systems tested

- Linux (Ubuntu 14.04)
- Windows XP
- Windows 10

Troubleshooting

On Windows:

- Download and install PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/); this is a good terminal emulator (as well as an excellent SSH client).

On Linux:

- Lots of ways. But using PySerial (as Starfeeder does):

```
pip install https://github.com/pyserial/pyserial/tarball/3e02f7052747521a21723a618dccf303065da732  # install PySerial 3.0b1

python -m serial.tools.miniterm /dev/ttyUSB0 9600 --eol LF --develop --parity N  # RFID reader

python -m serial.tools.miniterm /dev/ttyUSB1 9600 --eol LF --develop --parity E  # Balance
```

Regardless of operating system:

- For the RFID readers:

- Connect to the correct COM port using the settings 9600, 8N1, XON/XOFF (do not use RTS/CTS under Windows).
- A few key commands are shown below.
- Note that the commands are case-sensitive and single-character only (do not send a newline or you will cancel ongoing reads).
- If it doesn't understand something, it will say "?".

Action	You type	Reply
Reset	`x`	`MULTITAG-125 01`
Start reading	`c`	nothing, then RFID tag codes as they are detected
Stop reading	`p`	`S`

- For the balance:
  - Connect to the correct COM port using the settings 9600,**8E1**, XON/XOFF.
  - The balance is particularly frustrating, as it usually doesn't say anything if you get the syntax wrong. Occasionally it says `?`.
  - Specimen commands are shown below.

Action	You type	Reply
Restart	`RES;`	nothing
Request status	`ESR?;`	`000`
Request ASCII output	`COF3;`	`0`
Request 10 readings	`MSV?10;`	data