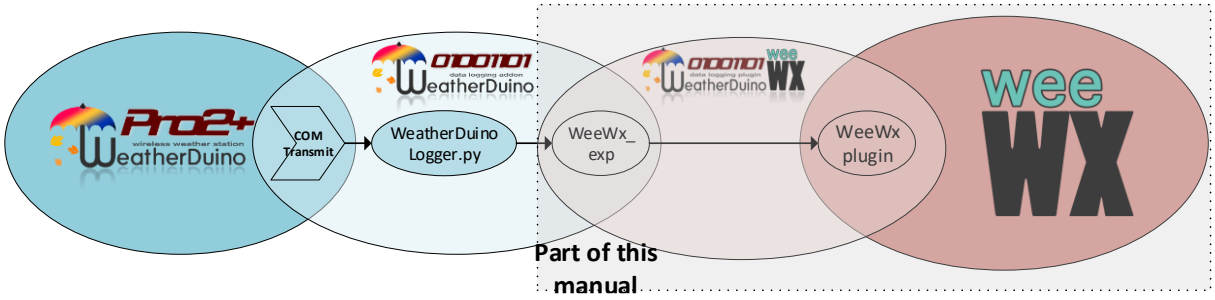
WeeWx plugin
for
WeatherDuino Pro2 Plus and
WeatherDuino Logger add-on

by engolling

31.12.2020

Get all the data captured by the WeatherDuino Pro2 Plus system into the WeeWx weather software.

In combination with the WeatherDuino logging add-on, you can import all the processed signals to WeeWx. Therefore, the database schema of WeeWx is extended to contain all of the selected signals.
Here is a scheme how the modules are working (together).



To use the WeeWx plugin you have to install the WeatherDuino Logger add-on und the logging script written in python first.
The logging script has to be running successfully first before installing the WeeWx plugin!

WeeWx should also be already running properly and getting LOOP data from the connected weather station.
The WeeWx plugin is supported with WeeWx running on Python2.7 and Python 3!

How to use and install the WeatherDuino logging add-on please refer to this forum thread.

Features:

- Get extra signals in full precision into WeeWx
- Get signals of multiple wind- and rain sensors into WeeWx
- Get signals of the AQM into WeeWx

Installation

First copy the plugin file in the “BIN ROOT” in the folder “user”.
How to find the “BIN ROOT”of WeeWx - please refer to the WeeWx user guide.

Since the plugin reads the “WeeWx_Exp.txt” output file of the WeatherDuino logging script, **you have to specify the absolute path to this file**. Depending to your system and folder structure, this may vary.

At the moment it is hardcoded on **three** positions:

First in line 10

```
#First read units from unit line of the export file
filename = '/home/pi/WeatherDuino/WeeWx_Exp.txt'
with open(filename) as f:
  for line,row in enumerate(f.readlines()):
    if line == 0:
      names = row.strip().split(";")
    if line == 1:
      unit_groups = row.strip().split(";")
```

Second in line 41

```
    super(WeeWxService, self).__init__(engine, config_dict)
    d = config_dict.get('WeatherDuino_logger_service', {})
    self.filename = d.get('filename', '/home/pi/WeatherDuino/WeeWx_Exp.txt')
    syslog.syslog(syslog.LOG_INFO, "WeatherDuino: using %s" % self.filename)
    self.bind(weewx.NEW_ARCHIVE_RECORD, self.read_file)
```

Third in line 147

```
import schemas.wview
filename = '/home/pi/WeatherDuino/WeeWx_Exp.txt'

with open(filename) as f:
```

The first part (line 1- 35) is responsible to provide the correct units in WeeWx.

Be aware, each unit group and unit type that is defined in the transmission layout file in the logging script and appears in the “WeeWx_Exp.txt” file has to be described in the standard unit set of WeeWx or in this part of the configuration!

See the WeeWx customization guide and the source code which unit groups and unit types are supported. If you use unsupported units you might have to add them in this part.

The second part (line 37 – 145) does the actual data handling and passes is to the WeeWx archive.

The third part (line 146 - 158) extends the database scheme. This part is only needed once at the beginning and whenever you change anything in the layout file of the logging script.

Before starting over to make the first modifications in WeeWx please check that the “WeeWx_Exp.txt” file it written properly and contains all your defined data. **Make sure that there are no signal names containing a “ “ space or a “.” dot character. Be careful, some names contain a dot by default.** This will make WeeWx crashing while generating the database.

To get the plugin working properly the weewx.conf file has to be modified.
The first step is to extend the database with the new schema, therefore uncomment the old schema and refer to the new one.

```
[DataBindings]

  [[wx_binding]]
    # The database must match one of the sections in [Databases].
    # This is likely to be the only option you would want to change.
    database = archive_sqlite
                #database = archive_mysql
    # The name of the table within the database
    table_name = archive
    # The manager handles aggregation of data for historical summaries
    manager = weewx.wxmanager.WXDaySummaryManager
    # The schema defines the structure of the database.
    # It is *only* used when the database is created.
    #schema = schemas.wview.schema
     schema = user.WeeWx_WeatherDuino_Logger_plugin.schema_WeatherDuino
```

Next, add the data collecting service to the engine section:

```
[Engine]

  [[Services]]
    # This section specifies the services that should be run. They are
    # grouped by type, and the order of services within each group
    # determines the order in which the services will be run.
    prep_services = weewx.engine.StdTimeSynch
    data_services = user.WeeWx_WeatherDuino_Logger_plugin.WeeWxService,
    process_services = weewx.engine.StdConvert, weewx.engine.StdCalibrate, weewx.engine.StdQC,
weewx.wxservices.StdWXCalculate
    archive_services = weewx.engine.StdArchive
    restful_services = weewx.restx.StdStationRegistry, weewx.restx.StdWunderground,
weewx.restx.StdPWSweather, weewx.restx.StdCWOP, weewx.restx.StdWOW,
weewx.restx.StdAWEKAS
    report_services = weewx.engine.StdPrint, weewx.engine.StdReport
```

After that, all modifications of the weewx.conf are done and the new database have to be generated. Run the following command while being in the path of the configuration directory:
wee_database weewx.conf --reconfigure

This will generate a new extended database named *weewx.sdb_new*.
When using SQLite as database system you have to rename the new generated database to replace the initial one.
cd SQLITE_ROOT
mv weewx.sdb_new weewx.sdb

At the end, run the following command while being in the path of the configuration directory
wee_database weewx.conf --rebuild-daily
to rebuild the daily summaries.

See also the [WeeWx customization guide.]

Now you are finished. As long as the WeatherDuino Logging script and WeeWx is running all extra data will be imported and archived in the WeeWx database. The signals are available to display them via tags in the skin files or for generating some graphs.

Details of successful imports or errors can be found in the corresponding logfile or syslog if the debug output is enabled.