# EPANET File Utility

## Introduction

The EPANET file utility is intended to support the reading/loading of different types of EPANET files. A collection of utilities is planned, with each file type being a separate Python package. In the first release, there is only one package: EPANETOutputFile.

## Licence

The software is available under an MIT licence. Each Python package is expected to be extensible through plugins, which can have a different licence (which may not be an open source licence) from the main package. Such plugins can further manipulate the data read from the EPANET file.

## Platforms

Written for Python 2.6 and 2.7 to run on:

* MacOS X® (development done on 10.9.1, but it should work on earlier versions) and
* Windows® (development done on Vista®, but it should work on other versions).

GNU xgettext is also required to generate the language template (.pot) file. The most recent version of each has been used successfully.

A simple GUI is also being developed using wxPython 3.0.0 on MacOS X® and Windows®.

# EPANETOutputFile

## Translations

### .POT file

A Portable Object Template (POT) file is included in the EPANETOutputFile directory: EPANETOutputFile.pot.

We generate our .POT file from the source using i18n.bat (or i18n.sh) which contains the command:

python "c:\Program Files (x86)\Python27\Tools\i18n\pygettext.py" -van -d EPANETOutputFile -p EPANETOutputFile EPANETOutputFile\EPANETOutputFile.py EPANETOutputFile\EOFTInternalPlugin.py

### .PO/.MO files

Translations are created in the locale directory. One is created as a simple example: en\_AU – Australian English.

This contains two files in the locale\en\_AU\LC\_MESSAGES directory:

EPANETOutputFile.po Portable Object file – text version of translated strings

EPANETOutputFile.mo Machine Object file – binary version of translated strings

**POEdit** is a simple tool for working with translations. See www.poedit.net.

New translations can be created and existing ones can be updated when the EPANETOutputFile.POT file is updated.

### Specifying Locale

Normally locale is specified using the LANG environment variable or other related variables. Working in Windows® can be more difficult as the LANG variable is normally not set and the default Windows®/Python locale can be different from the Posix locale. Australian English is an example of this. To specify Australia English, you need to set the environment variable:

set LANG=en\_AU

or

set LANG=en\_AU.UTF-8

## Plugins

Plugins live as packages inside the EPANETOutputFile/plugins directory. Plugins will normally do some form of analysis of the data in the output file or be extra tools. Plugins might provide:

(a) export to different file formats (i.e. not just CSV)

(b) graph support

(c) compute averages, minima, maxima

Keeping the names of plugins unique is necessary.

A demo plugin is provided. This plugin has the following structure:

|  |  |
| --- | --- |
| **Object** | **Description** |
| EPANETOutputFile\plugins\demo | Directory containing demo plugin |
| EPANETOutputFile\plugins\demo\\_\_init\_\_.py | Package containing plugin code (see details below). |

A plugin must be a class which inherits from EPANETOutputFilePlugin.EOFTPlugin. This class can then override functions to listen to the messages required to do the work of the plugin. The demo plugin adds some extra functionality to display:

1. Extra information about the output file
2. Extra information from the output file prolog section
3. Extra information from the dynamic results section of the output file
4. Information about the plugins loaded
5. Display verbose message information

Display of this extra information is controlled by new group of command line options which are added by the plugin.

With the plugin installed, the usage message is extended to

Usage: ReadEPANETOutputFile.py [options] filename

Options:

--version show program's version number and exit

-h, --help show this help message and exit

-v, --verbose display verbose output

-a, --all display all output file sections (default)

-s, --silent don't display any output file sections

-p, --prolog display prolog section

-n PROLOG\_NODE\_CSV, --prolog\_node\_csv=PROLOG\_NODE\_CSV

write CSV for nodes from prolog to PROLOG\_NODE\_CSV

-l PROLOG\_LINK\_CSV, --prolog\_link\_csv=PROLOG\_LINK\_CSV

write CSV for links from prolog to PROLOG\_LINK\_CSV

-e, --energy\_use display energy use section

-E ENERGY\_CSV, --energy\_use\_csv=ENERGY\_CSV

write CSV from energy use section to ENERGY\_CSV

-d, --dynamic\_results

display dynamic results section

-N DYNAMIC\_NODE\_CSV, --dynamic\_node\_csv=DYNAMIC\_NODE\_CSV

write CSV for nodes from dynamic results to

DYNAMIC\_NODE\_CSV

-L DYNAMIC\_LINK\_CSV, --dynamic\_link\_csv=DYNAMIC\_LINK\_CSV

write CSV for links from dynamic results to

DYNAMIC\_LINK\_CSV

-c, --coda, --epilog display file epilog

Demo Plugin Options:

Extra output file and summary infomation.

--demo\_info display extra info about output file

--demo\_prolog\_info display extra prolog info

--demo\_dynamic\_results\_info

display extra dynamic results info

--demo\_plugin\_info display info about plugins loaded

--demo\_verbose display plugin messages

--demo\_all display all demo plugin info

Anyone wanting to write plugins should look at the demo plugin and read the documentation and suggestions included in the file EPANETOutputFile\plugins\demo\\_\_init\_\_.py.

There is currently no ability to disable loading of plugins.

## Standalone program

ReadEPANETOutputFile.py is provided as a standalone program using the EPANETOutputFile package.

Usage is displayed with the ‘–h’ option.

## Importing the package

The EPANETOutputFile package can be imported and used to load an EPANET output file into memory. The ReadEPANETOutputFile.py script shows how to do this. Once the file has been loaded, the data has been loaded into dictionaries and arrays of dictionaries which can be used in any way desired. For example, a tool could easily be created to load and compare two different EPANET output files.

## Py2EXE

WaterSums uses this utility to export Comma Separated Variable (CSV) files by converting the Python script into a Windows® executable using Py2EXE for Python 2.6. A configuration script is found in EPANETOutputFile/setup.py. This requires the VC90 C runtime which requires a licence from Microsoft® to distribute.