

A Command Line Tool for Aggregating Precipitation Amounts Using Grid Coordinates

Version

1

WEATHER DATA TOOL

U.S. Environmental Protection Agency

Weather Data Tool User Guide

U.S. ENVIRONMENTAL PROTECTION AGENCY

Weather Data Tool User Guide

U.S. Environmental Protection Agency
Homeland Security Research Program
Research Triangle Park, NC 27711

Disclaimer

The U.S. Environmental Protection Agency (EPA) through its Office of Research and Development funded and managed the research described herein under EP-C-15-004, PR-ORD-16-01029 to Tetra Tech. It has been subjected to the Agency's review and has been approved for publication. Note that approval does not signify that the contents necessarily reflect the views of the Agency. Any mention of trade names, products, or services does not imply an endorsement by the U.S. Government or EPA. The EPA does not endorse any commercial products, services, or enterprises. The contractor role did not include establishing Agency policy

For further information or to provide feedback, please contact:

Timothy Boe
US EPA Office of Research and Development
National Homeland Security Research Center (NHSRC)
Decontamination and Consequence Management Division (DCMD)
919-541-2617
boe.timothy@epa.gov

Anne Mikelonis
US EPA Office of Research and Development
National Homeland Security Research Center (NHSRC)
Decontamination and Consequence Management Division (DCMD)
919-541-0579
mikelonis.anne@epa.gov

Revision Sheet

Release No.	Date	Revision Description	Revised By
Rev. 0	01/17/2017	Document Creation	Jesse Elfalan
Rev. 1	03/28/2017	Version 1 Feature Updates	Jesse Elfalan
Rev. 2	05/23/2017	Documentation Updates	Jesse Elfalan
Rev. 3	04/12/2018	Documentation Updates	Ken Wilkinson

CONTENTS

1.0 Overview	1
1.1 Application Overview	1
1.1.1 Software Prerequisites: Python and Command Prompt	1
1.1.2 Modes of Operation: Interactive Mode and Batch Mode	1
1.1.3 Web-based Weather Data Providers	1
1.1.4 Application Configuration (config.ini)	2
1.1.5 Document Nomenclature and Additional Information	2
2.0 Getting Started	2
2.1 Setup	2
2.1.1 Check Python Installation	2
2.1.2 Weather Data Tool Installation, Uninstall	3
2.1.2.1 Installation	3
2.1.2.2 Uninstall	4
2.1.3 Configure config.ini file	4
2.2 Interactive Mode	6
2.2.1 Running Interactive Mode	6
2.2.2 Exiting the inner loop and application	7
2.2.3 Toggle Data Type	8
2.3 Batch Mode	8
2.3.1 Batch Mode Arguments	8
2.3.2 Creating a Batch file	9
3.0 Output	10
4.0 Options	10
4.1 Logging and Verbose Options	10
4.2 Usage and Time zone Options	11
5.0 Resources	12
5.1 Software Packages	12
5.1.1 Python	12
5.1.2 Terminal/Command Prompt	12
5.2 Data Providers	12
5.2.1 Forecast Data Provider	12
5.2.2 Historical Data Provider	12

1.0 OVERVIEW

The Weather Data Tool is a command line interface application written in Python that facilitates the acquisition of time series precipitation data from national Forecast and Historical weather data providers and generates an output file to be consumed by the EPA Storm Water Management Model (SWMM). The application conducts lightweight processing and formatting of the downloaded data to generate an output file containing rainfall precipitation values greater than 0 and conforms to the specified DSI-3240 file format.

1.1 Application Overview

The purpose of this application is to mitigate the amount of work required for retrieving and formatting Forecast and Historical rainfall precipitation data. The application consists of the following platforms and components:

- Python and Command Prompt
- Modes of Operation: Interactive Mode & Batch Mode
- Web-based Weather Data Providers (Forecast & Historical)
- Application Configuration (config.ini)

1.1.1 Software Prerequisites: Python and Command Prompt

In order to run the application, the operating system must provide a command line interface or command prompt application and an installation of **Python version 3.4.4 or 3.5.2**. A command prompt application such as the Windows Command Prompt or a Unix/Linux Terminal is sufficient to run the application. Links to download these resources are provided in the Resources section if your operating system does not have these prerequisites.

1.1.2 Modes of Operation: Interactive Mode and Batch Mode

The Weather Data Tool application has two modes of operation: Interactive Mode and Batch Processing Mode. Interactive mode prompts the user for input parameters in a sequential manner where the user manually types out each parameter to instruct the application to download data from the corresponding data providers; this mode of operation is useful for single usage or single request types. Batch Processing Mode is an automated version of this instruction process where the application is run by a batch file containing pre-constructed request commands; this mode of operation is useful for large sets of requests. More information on the application configuration and modes of operation can be found in the Getting Started section.

1.1.3 Web-based Weather Data Providers

The application uses HTTP web services from two data providers to download data. Forecast data is provided by the National Oceanic and Atmospheric Administration (NOAA) weather service, and historical data is provided by NASA Land Data Assimilation Systems (NLDAS). NOAA forecast response data is returned in an XML based format and NLDAS is returned as a data stream. Links to resources related to these data providers are available in the Resources section of this document. NLDAS data are based on daily rainfall measurements that are downscaled and interpolated onto a 1/8" degree grid as well as undergo a statistical procedure to

disaggregate the daily values to hourly values. Users are highly encouraged to read more about the NLDAS data before using to gauge applicability to their SWMM project.

1.1.4 Application Configuration (config.ini)

All user and application settings are located in the config.ini file. Settings such as the output file directory and file extension can be specified in this settings file. More detail on the setup and configuration of this settings file can be found in the Getting Started section.

1.1.5 Document Nomenclature and Additional Information

Request: Refers to the HTTP request with given parameters to be sent to the data provider server.

Data Type: Refers to Forecast or Historical time series data.

Time zone: Refers to the local time zone of the longitude and latitude coordinates. The request start and end dates are converted from local time to UTC time and submitted to the server. NLDAS historical data will be returned as a time series in UTC, which is converted back to local time. Then, an hour is subtracted from the date-time, so the date-time bin is consistent with SWMM. NOAA forecast data is returned as a time series in local time, see QPF below.

Grid Coordinates: NLDAS (historical data provider) uses a grid coordinate system instead of latitude and longitude coordinates. The application automatically converts the user input for latitude and longitude coordinates to grid coordinates using the NLDAS conversion algorithm. Note that Alaska and Hawaii have been omitted from this grid system.

Quantitative Precipitation Forecast (QPF): Refers to the expected amount of precipitation accumulated over a specified time period over a specified area. The NOAA forecast data returned contains the liquid precipitation over a 6-hour period. The application converts from inches/6-hours to inches/hour, resulting in the output of steady-state rain intensity records at the start of each hour, over the six-hour period, suitable for import into SWMM. Conversion to metric is performed as needed.

2.0 GETTING STARTED

2.1 Setup

2.1.1 Check Python Installation

Before you can run the application, be sure you have a working installation of **Python 3.4.4** or **3.5.2** installed on your system. To open the command prompt for Windows 7 operating system, type in the Windows search bar of the start menu “command prompt”.

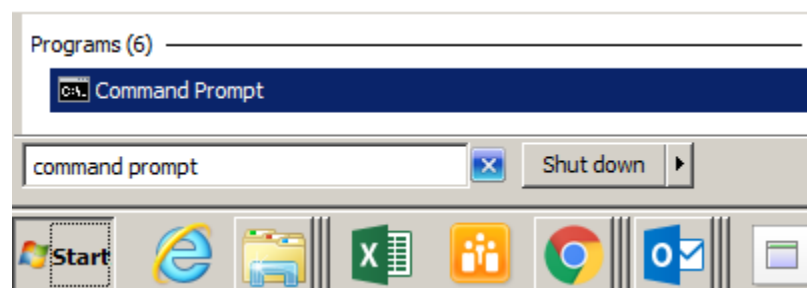


Figure 1. Search for Command Prompt

In command prompt, check to see if python is installed on your system by typing:

```
python -V
```

or

```
py -V
```

If Python is installed correctly, the prompt should print the current version installed. If the prompt does not return a version number, please refer to the Resources section to download and install Python for your machine.

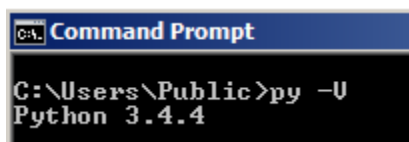


Figure 2. Python Version

Additionally, if there is more than one version of Python installed on your system, you can choose to run a particular version by typing in the command line the version of python before the script to be run. To run version 3.6, enter:

```
py -3.6 EpaWeatherTool.py
```

2.1.2 Weather Data Tool Installation, Uninstall

2.1.2.1 Installation

First, unzip the distribution file for the application in a desired location. Open the command prompt and navigate to this application folder. (For more information on how to navigate the file system in a command prompt please refer to the Resources section).

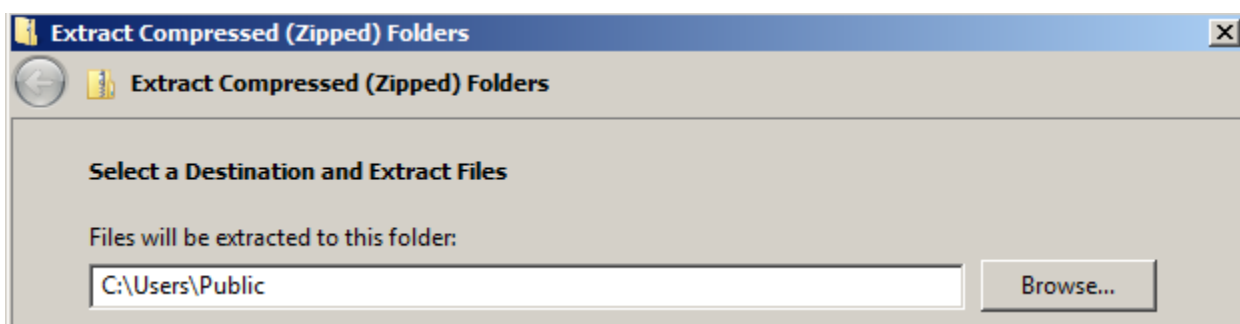


Figure 3. Unzip Distribution

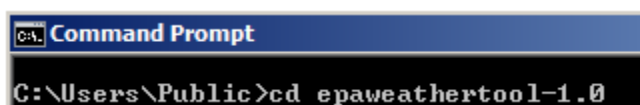


Figure 4. Navigate to application folder

We will use the Python package manager **Pip** to install the application and application dependencies. Install the application by running the command (note the “**space**” and “**dot**” at the end of the command to install all dependency packages):

```
py -m pip install .
```

This will run the setup script which will download and install the application and the application dependencies. The current dependency packages are: **requests v.2.12.4**, **pytz 2016.10**, and **timezonefinder 1.5.7**.



Figure 5. Using Pip

2.1.2.2 Uninstall

From a command prompt in the application folder, uninstall the product and dependency packages using the following commands:

```
py -m pip uninstall requests
```

```
py -m pip uninstall pytz
```

```
py -m pip uninstall timezonefinder
```

Lastly, delete the application folder.

2.1.3 Configure config.ini file

By default, the tool can be run without any changes to the configuration file, however certain parameters can be set to be applied automatically when running the tool. To edit these settings, using a text editor open the configuration file (config.ini) located in the application folder. This file contains information for default settings for the Weather Data Tool such as the output file directory, the default file extension, precipitation measurement units and time zone. This file is the default point of reference for the application, so any desired configuration changes must be done through this file. The configurable attributes are defined by the section names in brackets '[']'.

1. API

This section is used to define the base URL(s) for the data providers. The default values for these items have already been configured to run with the application and do not require any changes.

2. File Output Path

Configure the default output file path where new time series files are to be saved. Under the section labeled **IO_Dir**, specify an *absolute* path to the desired file directory in your file system, by default this has been set to the output_files directory in the source folder. If changing the

output folder path, be sure the user has write permissions to this directory or the application will be unable to create new files in this directory. Note that this directory must be specified in order to run the application.

```
[IO_Dir]
out_dir: .\output_files
```

Figure 6. Default output path

3. File Extension, *precip_id*, *precip_units*

Next, set the file extension of the output file (by default the extension has been set to a comma-separated value or **CSV**. (NOTE: The delimiter or symbol that separates the values in the output file is a comma ‘,’ by default. This attribute may be changed in the ConfigMod.py file). The value for **data_format** is for output reference purposes and **precip_id** corresponds to the ID label specified in the DSI-3240 data format. Changing the **precip_id** will change the ID label in the output file. The **precip_units** accept a value of either *metric* or *standard* which are measured in *mm* or *inches* respectively, where the value will be converted to either one for the output file.

```
[Format]
file_ext: csv

data_format: DSI-3240

precip_id: AllPrec

#metric or standard (mm or inches)
precip_units: standard
```

Figure 7. Settings under Format

4. Time-zone (optional)

Under the options section, a default **time zone** may be set. However, note that for the default time zone option the same time zone value will be applied for each request that is made. To override this behavior, leave this option blank so that the application will prompt for a time zone value when run in interactive mode. For batch mode requests, the time zone must be specified as one of the parameters (for more information on this, refer to the Batch Mode sub-section 2.3).

```
[Options]
#To list all available timezone codes, supply ,
# Examples: US/Alaska, US/Arizona, US/Central,
timezone: auto
```

Figure 8. Time zone Setting

5. Date-restrict (optional)

To restrict the data returned to the precise start and end dates, switch the parameter to “**on**” to turn the date filter on.

6. Hour-restrict (optional)

To restrict the data returned to a specific date and time range, switch the date and hour parameters to “**on**” to turn the hour filter on. In the Interactive mode, an additional prompt for the hour range will be added. In Batch mode, these hour parameters must be specified as integers from 0 to 23 (for more information on this, refer to the Batch Mode sub-section 2.3).

NOTE: All changes made to the config.ini file must be saved and the application restarted for changes to take effect.

2.2 Interactive Mode

Interactive mode is a mode of operation that prompts the user for information in a sequential manner to build a request. This mode of operation can be used to make Forecast and Historical time series data requests one at a time.

2.2.1 Running Interactive Mode

IMPORTANT: Prior to running the application in either interactive or batch mode, be sure that you are in the application folder in the command prompt where the main script file, **EpaWeatherTool.py**, is located.

To run the application in interactive mode, simply type:

```
python EpaWeatherTool.py
```

or

```
py EpaWeatherTool.py
```

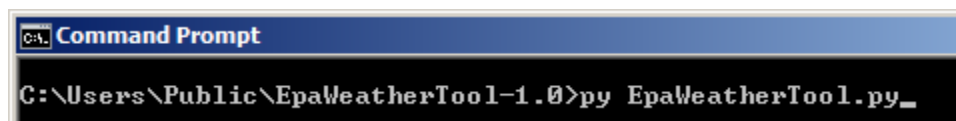


Figure 9. Run Application

The application will now start in interactive mode. To turn on additional features such as logging and verbose output, supply the options **-l** or **-logging** and **-v** or **-verbose** after the application name to start the modules (for more information on logging and verbose options, see the Options section).

Logging and Verbosity turned on:

```
py EpaWeatherTool.py -l -v
```

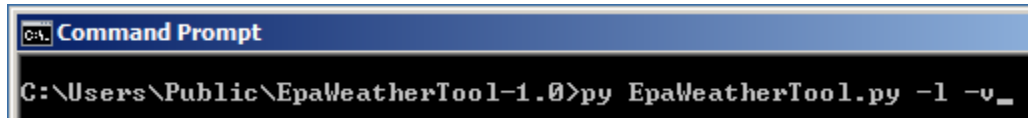


Figure 10. Run with logging and verbose features

The application will then sequentially prompt the user for the necessary information: 1 or 2 for Forecast or Historical time series data, latitude/longitude coordinates, time zone (refers to local time zone of latitude and longitude coordinates), start and end dates, and output filename. The application will automatically catch incorrect input types or unrecognized symbols and notify the user of the problem. The following are examples of acceptable input formats and types:

EXAMPLE:

1. Forecast or Historical data type:

1
2

2. Latitude and Longitude:

35.8970,78.8602
35.89,-78.8602
35.89, 78.86

3. Start/End Dates (Format specified as mm/dd/yyyy):

Forecast:

1-4-2017
1-30-2017

Historical:

1-1-2010
1-30-2010

3.1. Optional Hour Range (integer 0-23):

0,22

4. Time-zone (run -tz option for a list of available time zone names):

US/Pacific
US/Eastern
US/Mountain

5. Filename (standard symbol for file names apply):

test_filename

On the success of correct input, the application will make an HTTP request to the appropriate data source provider to retrieve the time series data. The response data is automatically downloaded, filtered, formatted and saved as a new file in the specified output directory.

NOTE: If the name of a newly generated output file is the same as an existing file, the existing file will be overwritten. Please be sure to provide unique file names or be aware an existing file may be overwritten.

2.2.2 Exiting the inner loop and application

The application is built on two levels, an outer and inner loop. The outer loop is where the user makes the selection for the data type or time series data they would like to build a request for and the inner loop is where the information for the rest of the request is made. In the case where the wrong data type has been selected, the user may exit the inner loop by typing 'exit'. The

application will then revert back to the top level or outer loop where it will prompt the user for the data type again.

Typing **'exit'** again at the top level of the application will exit the application. After a request has been completed, the application will automatically revert back to the outer loop where the user may either choose to make another request or quit the application by typing **'exit'** or **'n'** for no. Typing **'y'** for yes to make another request will automatically select the same data type of the previous request. Additionally, to force quit the application press **[CTRL] + [C]**.

2.2.3 Toggle Data Type

To switch between Forecast and Historical data types, return to the top level of the application where the application will prompt to make another request (if you are in the inner loop type **'exit'** as described previously). Type **'t'** to trigger the prompt for the data type and select the desired data type.

2.3 Batch Mode

Batch mode allows the user to automate the process of making requests by creating a batch (.bat) file that passes arguments and parameters to the application. Options such as logging and verbosity can also be turned on to generate log files for each request that is made.

2.3.1 Batch Mode Arguments

Batch mode takes both positional and optional arguments to read in the input parameters and options for logging and verbosity. The two required positional arguments are for an override output directory and input parameters defined in square brackets, [].

The override or alternative output directory path may or may not be used but must always be supplied as an argument; if the override is not used then the argument must pass the **'default'** parameter, where default will use the output file path specified in the config.ini file. Parameters are specified in square brackets '[]' and separated by commas ','. These parameters must also be specified in the following order:

[DataType,Longitude,Latitude,StartDate,EndDate,StartHr,EndHr,Timezone,Output_Filename]

NOTE: Parameters must be entered with NO spaces

There is a total of 10 parameters that **MUST** be specified:

1. DataType = **1** or **2** (Forecast or Historical)
2. Longitude = **37.9780**
3. Latitude = **122.0311**
4. StartDate = **1-4-2017**
5. EndDate = **1-30-2017**
6. StartHour = **1** (integer from 0-23)
7. EndHour = **1** (integer from 0-23)
8. Timezone = **US/Pacific**
9. Precip_id = **example_id**
10. Filename = **test_file1**

This array of parameters would then be:

```
[1,37.9780,122.0311,1-4-2017,1-30-2017,1,1,US/Pacific, example_id,test_file1]
```

Putting it all together, the line constructed should follow this pattern:

```
py ApplicationName.py | options | output_directory | parameters
```

Some examples of the usages are provided below:

EXAMPLE:

With no options, using default output file directory:

```
py EpaWeatherTool.py default [1,37.9780,122.0311,1-4-2017,1-30-2017,0,0,US/Pacific,example_id,test_file1]
```

```
py EpaWeatherTool.py default [1,37.9780,122.0311,1-2-2010,1-3-2010,2,23,US/Pacific,example_id,test_file2]
```

With logging and verbosity options turned on:

```
py EpaWeatherTool.py -l -v default [2,37.9780,122.0311,1-4-2017,1-30-2017,1,23,US/Pacific,example_id,test_file3]
```

```
py EpaWeatherTool.py -l -v default [1,37.9780,122.0311,4-4-2017,4-30-2017,0,0,US/Pacific,forecast_precip,concord_ca]
py EpaWeatherTool.py -l -v default [2,37,122,1-4-2016,1-6-2016,10,10,America/Chicago,historical_precip,dodge_city_ks]
py EpaWeatherTool.py -l -v default [2,40.7128,74.0059,1-4-2016,1-6-2016,10,10,America/New_York,historical_precip,New_York_ny]
```

Figure 11. Example batch file statements

Additional information on precip_id

In the config.ini file, precip_id can be set to the keyword 'custom' to allow users to specify their own precip_id at runtime, or set the value to any word which will be used for all output files generated. If the precip_id parameter is set to 'custom' in the config file but batch mode is run, the precip_id provided in the batch mode parameters will be used as there is no prompt in batch mode.

2.3.2 Creating a Batch file

A batch file can easily be created in any text editor application. In Windows, open a text editor application such as Notepad and add the lines that you have constructed as in the previous examples above. In addition to the sequence of argument lines, a delay (using the timeout command) must be added between each line to compensate for the lag in the file writing process. For most cases a delay of 3 to 5 seconds between each request is sufficient. If the delay is omitted some output and log files may not be created. Save this file with an extension of **.bat**. An example batch file would look something like this:

EXAMPLE:

```
py EpaWeatherTool.py default [1,37.9780,122.0311,1-4-2017,1-30-2017,0,1,US/Pacific,forecast_precip,test_file1]
```

```
timeout /t 3
```

```

py EpaWeatherTool.py default [2,37.9780,122.0311,1-2-2010,1-3-
2010,0,23,US/Pacific,historical_precip,test_file2]

timeout /t 3

py EpaWeatherTool.py default [2,37.9780,122.0311,1-5-2008,1-6-
2008,0,22,US/Pacific,historical_precip,test_file3]

py EpaWeatherTool.py -l -v default [1,37.9780,122.0311,4-4-2017,4-30-2017,0,0,US/Pacific,forecast_precip,concord_ca]
timeout /t 3

py EpaWeatherTool.py -l -v default [2,37,122,1-4-2016,1-6-2016,10,10,America/Chicago,historical_precip,dodge_city_ks]
timeout /t 3

py EpaWeatherTool.py -l -v default [2,40.7128,74.0059,1-4-2016,1-6-2016,10,10,America/New_York,historical_precip,New_York_ny]
timeout /t 3

```

Figure 12. Example batch file with timeout

3.0 OUTPUT

The columns in the comma separated output file are in the following order: Precip_id, Year, Month, Day, Hour, and Precipitation Measurement. The table below is an example of data with these corresponding header fields.

Table 1. Example output with column header added

precip_id	year	month	day	hour	precip value
all_precip	2017	4	3	15	0.06
all_precip	2017	4	3	16	0.05
all_precip	2017	4	3	17	0.01
all_precip	2017	4	4	3	0.02
all_precip	2017	4	4	4	0.9
all_precip	2017	4	4	5	0.59

When no precipitation is present, an output file will be generated with a note in the file stating that there was no precipitation for the provided date range.

4.0 OPTIONS

4.1 Logging and Verbose Options

The Logging and Verbose options are additional features to help enhance visibility and feedback during the processing of a request. As described previously, these options are supplied on the command line with **-l** or **-logging** and **-v** or **-verbose** to turn these modules on.

When operating in interactive mode, logging will generate a log file in the **logs** directory in the root directory of the application for the duration of the session. When the logging option is supplied for the batch mode operations, a new log file will be created in the **logs** folder for each line in the batch file as each request counts as a single session. Verbosity will output additional information that may be useful in a debug scenario.

Within the log file, information is separated into 4 columns. From left to right are the timestamp of the event, the module associated with the event, the status level of the event (info, debug,

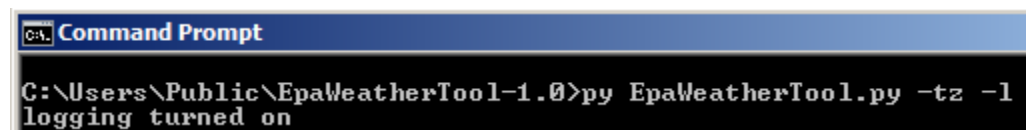
error) and the message contents. If an error occurs due to an invalid input or during the processing of a request, the log file will indicate at what point the error occurred and the possible cause of the error.

```
01-16 08:47:33 app.root      INFO      logging turned on
01-16 08:47:33 app.root      INFO      args received: Namespace(about=False, dirpath='default',
01-16 08:47:33 app.root      INFO      verbosity turned on
01-16 08:47:33 app.root      INFO      output dir: <DEFAULT> None
01-16 08:47:33 app.control   INFO      Process started, validating input and running routine...
01-16 08:47:33 app.control   INFO      Validating Latitude and Longitude...
```

Figure 13. Example Log File

4.2 Usage and Time zone Options

Usage (help) and time zone options **-h** and **-tz**, respectively, are printouts for the user's reference. Time zones come from the IANA time zone database and are imported using a Python time zone library. Additionally, to have a hard copy of the available time zones for reference you can supply the **-tz** with **-l** option which will write the time zones out to a log file.



```
C:\> Command Prompt
C:\Users\Public\EpaWeatherTool-1.0>py EpWeatherTool.py -tz -l
logging turned on
```

Figure 14. Write the time zones to a log file

5.0 RESOURCES

5.1 Software Packages

5.1.1 Python

- <https://www.python.org/downloads/windows/>
- <https://wiki.python.org/moin/BeginnersGuide/Download>
- <https://docs.python.org/3.6/using/index.html>

5.1.2 Terminal/Command Prompt

- <http://simplyadvanced.net/blog/cheat-sheet-for-windows-command-prompt/>
- <https://gist.github.com/LeCoupa/122b12050f5fb267e75f>
- <https://git-scm.com/downloads>

5.2 Data Providers

5.2.1 Forecast Data Provider

- <http://graphical.weather.gov/xml/rest.php>

Digital Weather Markup Language Generator

- https://graphical.weather.gov/xml/sample_products/browser_interface/ndfdXML.htm

5.2.2 Historical Data Provider

- <https://disc.gsfc.nasa.gov/hydrology/data-rods-time-series-data>



PRESORTED STANDARD
POSTAGE & FEES PAID
EPA
PERMIT NO. G-35

Office of Research and Development (8101R)
Washington, DC 20460
Official Business
Penalty for Private Use
\$300