gpxconverterDocumentation

Release 1.7.4

Nidhal Baccouri

Nov 18, 2021

Contents:

CHAPTER 1

gpx-converter

1.1 GPX manipulation for humans

Python package for manipulating gpx files and easily convert gpx to other different formats.

- Free software: MIT license
- Documentation: https://gpx-converter.readthedocs.io.

1.2 When & Why

- You need to convert GPX to other formats
- You need to convert other formats like csv to GPX
- You want to interpolate the GPX coordinates
- High level of abstraction

- Stable API
- easy to use and to extend

1.3 Motivation

I decided to create this project because I had gpx data that I needed to manipulate. I searched for a python package for this but I did not find what I was looking for, therefore I created the gpx-converter package to make gpx files manipulation very easy. Furthermore, the package contains methods for applying interpolation on the gpx data. This feature was very helpful for me since I also needed to interpolate the gpx data and convert it to csv. Feel free to contribute or to give me feedback anytime :)

1.4 Features

- Convert gpx files to other formats such as csv, numpy arrays, dataframes, excel and json
- Convert csv files to gpx
- Apply interpolation on the gpx data

1.5 Installation

```
$ pip install gpx-converter
```

1.6 Quick Usage

```
from gpx_converter import Converter
```

Just read the gpx to dictionary

```
dic = Converter(input_file='your_input.gpx').gpx_to_dictionary(latitude_key='latitude
↪', longitude_key='longitude')
# now you have a dictionary and can access the longitudes and latitudes values from␣
↪the keys
latitudes = dic['latitude']
longitudes = dic['longitude']
```

Convert GPX to other formats

- Convert from gpx to csv:

```
Converter(input_file='your_input.gpx').gpx_to_csv(output_file='your_output.csv')
```

- Convert from gpx to excel sheets:

```
Converter(input_file='your_input.gpx').gpx_to_excel(output_file='your_output.xlsx')
```

- Convert from gpx to json:

```
Converter(input_file='your_input.gpx').gpx_to_json(output_file='your_output.json)
```

- Convert gpx file to dataframe:

```
df = Converter(input_file='your_input.gpx').gpx_to_dataframe()
```

- Convert gpx file to numpy array:

```
np_array = Converter(input_file='your_input.gpx').gpx_to_numpy_array()
```

Now convert other formats to GPX

- csv to gpx

```
Converter(input_file='your_input.csv').csv_to_gpx(lats_colname='column_name_of_
↪latitudes',
                                                  longs_colname='column_name_of_
↪longitudes',
                                                  output_file='your_output.gpx')
```

- excel to gpx

```
Converter(input_file='your_input.xlsx').excel_to_gpx(lats_colname='column_name_of_
↪latitudes',
                                                     longs_colname='column_name_of_
↪longitudes',
                                                     output_file='your_output.gpx')
```

- dataframe to gpx (notice that the method is static)

```
Converter.dataframe_to_gpx(input_df=your_df,
                           lats_colname='column_name_of_latitudes',
                           longs_colname='column_name_of_longitudes',
                           output_file='your_output.gpx')
```

- json to gpx

```
Converter(input_file='your_input.json').json_to_gpx(input_df=your_df,
                                                    lats_colname='column_name_of_
↪latitudes',
                                                    longs_colname='column_name_of_
↪longitudes',
                                                    output_file='your_output.gpx')
```

- Automate the conversion of multiple csv file to gpx:

```
Converter.convert_multi_csv_to_gpx(dirpath='your_directory/')
```

- Apply spline interpolation on gpx file (you need to install scipy for this to work):

```
interpolated_coordinates = Converter.spline_interpolation(cv=your_array_of_control_
↪vertices)
```

1.7 Usage from terminal

Alternatively you can use the gpx_converter directly from terminal. You just need to pass the function, input file and output file as arguments.

- function: the conversion method you want to use. For example "gpx_to_csv"
- input file: path to your input file
- output file: path to your output file

```
$ gpx_converter --function "gpx_to_csv" --input_file "home/your_input.gpx" --output_
↪file "home/your_output.csv"
```

or maybe you prefer the short version

```
$ gpx_converter -func "gpx_to_csv" -in "home/your_input.gpx" -out "home/your_output.
↪csv"
```

1.8 Links

Check this article to know more about gpx files and how to use the gpx-converter package. https://medium.com/p/57da00bd36fc/edit

1.9 Contributions

Contributions are always welcome. Make sure you check the guidlines first https://gpx-converter.readthedocs.io/en/latest/contributing.html

CHAPTER 2

Installation

2.1 Stable release

To install gpx_converter, run this command in your terminal:

```
$ pip install gpx_converter
```

This is the preferred method to install gpx_converter, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for gpx_converter can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/nidhaloff/gpx_converter
```

Or download the tarball:

```
$ curl -OJL https://github.com/nidhaloff/gpx_converter/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

```
from gpx_converter import Converter
```

Just read the gpx to dictionary

```
dic = Converter(input_file='your_input.gpx').gpx_to_dictionary(latitude_key='latitude
↪', longitude_key='longitude')
# now you have a dictionary and can access the longitudes and latitudes values from␣
↪the keys
latitudes = dic['latitude']
longitudes = dic['longitude']
```

Convert GPX to other formats

- Convert from gpx to csv:

```
Converter(input_file='your_input.gpx').gpx_to_csv(output_file='your_output.csv')
```

- Convert from gpx to excel sheets:

```
Converter(input_file='your_input.gpx').gpx_to_excel(output_file='your_output.xlsx')
```

- Convert from gpx to json:

```
Converter(input_file='your_input.gpx').gpx_to_json(output_file='your_output.json)
```

- Convert gpx file to dataframe:

```
df = Converter(input_file='your_input.gpx').gpx_to_dataframe()
```

- Convert gpx file to numpy array:

```
np_array = Converter(input_file='your_input.gpx').gpx_to_numpy_array()
```

Now convert other formats to GPX

- csv to gpx

```
Converter(input_file='your_input.csv').csv_to_gpx(lats_colname='column_name_of_
↪latitudes',
                                                  longs_colname='column_name_of_
↪longitudes',
                                                  output_file='your_output.gpx')
```

- excel to gpx

```
Converter(input_file='your_input.xlsx').excel_to_gpx(lats_colname='column_name_of_
↪latitudes',
                                                     longs_colname='column_name_of_
↪longitudes',
                                                     output_file='your_output.gpx')
```

- dataframe to gpx (notice that the method is static)

```
Converter.dataframe_to_gpx(input_df=your_df,
                           lats_colname='column_name_of_latitudes',
                           longs_colname='column_name_of_longitudes',
                           output_file='your_output.gpx')
```

- json to gpx

```
Converter(input_file='your_input.json').json_to_gpx(input_df=your_df,
                                                    lats_colname='column_name_of_
↪latitudes',
                                                    longs_colname='column_name_of_
↪longitudes',
                                                    output_file='your_output.gpx')
```

- Automate the conversion of multiple csv file to gpx:

```
Converter.convert_multi_csv_to_gpx(dirpath='your_directory/')
```

- Apply spline interpolation on gpx file (you need to install scipy for this to work):

```
interpolated_coordinates = Converter.spline_interpolation(cv=your_array_of_control_
↪vertices)
```

3.1 Usage from terminal

Alternatively you can use the gpx_converter directly from terminal. You just need to pass the function, input file and output file as arguments.

- function: the conversion method you want to use. For example "gpx_to_csv"
- input file: path to your input file
- output file: path to your output file

```
$ gpx_converter --function "gpx_to_csv" --input_file "home/your_input.gpx" --output_
↪file "home/your_output.csv"
```

or maybe you prefer the short version

```
$ gpx_converter -func "gpx_to_csv" -in "home/your_input.gpx" -out "home/your_output.
↪csv"
```

CHAPTER 4

gpx_converter

4.1 gpx_converter package

4.1.1 Submodules

4.1.2 gpx_converter.base module

Top-level package for gpx_converter.

class gpx_converter.base.**Converter**(*input_file=None*)

Bases: object

main class converter that holds all conversion methods

static convert_multi_csv_to_gpx(*dirpath*, *lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname=None*, *alts_colname=None*)

convert multiple csv file from directory to gpx dirpath: directory path where the csv files are lats_colname: name of the latitudes columns longs_colname: name of the longitudes columns times_colname: name of the time columns alts_colname: name of the altitudes columns

csv_to_gpx(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname=None*, *alts_colname=None*, *output_file=None*)

convert csv file to gpx lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the time column alts_colname: name of the altitudes column output_file: path of the output file

static dataframe_to_gpx(*input_df*, *lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname=None*, *alts_colname=None*, *output_file=None*)

convert pandas dataframe to gpx input_df: pandas dataframe lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the time column alts_colname: name of the altitudes column output_file: path of the output file

excel_to_gpx(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname=None*, *alts_colname=None*, *output_file=None*)

convert csv file to gpx lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the time column alts_colname: name of the altitudes column output_file: path of the output file

gpx_to_csv(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname='time'*, *alts_colname='altitude'*, *output_file=None*)
convert a gpx file to a csv lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the times column alts_colname: name of the altitude column output_file: output file where the csv file will be saved

gpx_to_dataframe(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname='time'*, *alts_colname='altitude'*)
convert gpx file to a pandas dataframe lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the times column alts_colname: name of the altitude column

gpx_to_dictionary(*latitude_key='latitude'*, *longitude_key='longitude'*, *time_key='time'*, *altitude_key='altitude'*)

gpx_to_excel(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname='time'*, *alts_colname='altitude'*, *output_file=None*)
convert a gpx file to a excel lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the times column alts_colname: name of the altitude column output_file: output file where the csv file will be saved

gpx_to_json(*lats_keyname='latitude'*, *longs_keyname='longitude'*, *times_keyname='time'*, *alts_keyname='altitude'*, *output_file=None*)
convert a gpx file to json lats_keyname: name of the key which will hold all latitude values longs_keyname: name of the key which will hold all longitude values times_keyname: name of the key which will hold all time values alts_keyname: name of the key which will hold all the altitude values output_file: output file where the csv file will be saved

gpx_to_numpy_array()

json_to_gpx(*lats_colname='latitude'*, *longs_colname='longitude'*, *times_colname=None*, *alts_colname=None*, *output_file=None*)
convert csv file to gpx lats_colname: name of the latitudes column longs_colname: name of the longitudes column times_colname: name of the time column alts_colname: name of the altitudes column output_file: path of the output file

static spline_interpolation(*cv*, *n=100*, *degree=3*, *periodic=False*)
Calculate n samples on a bspline

cv : Array of control vertices n : Number of samples to return degree: Curve degree periodic: True - Curve is closed

False - Curve is open

4.1.3 gpx_converter.csv_to_gpx module

4.1.4 gpx_converter.gpx_to_csv module

4.1.5 gpx_converter.gpx_to_excel module

4.1.6 gpx_converter.gpx_to_json module

4.1.7 Module contents

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/nidhaloff/gpx_converter/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

gpx_converter could always use more documentation, whether as part of the official gpx_converter docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/nidhaloff/gpx_converter/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *gpx_converter* for local development.

1. Fork the *gpx_converter* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/gpx_converter.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv gpx_converter
$ cd gpx_converter/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 gpx_converter tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/nidhaloff/gpx_converter/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ pytest tests.test_gpx_converter
```

5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 6

Credits

6.1 Development Lead

- Nidhal Baccouri <nidhalbacc@gmail.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index