
pygeobase Documentation

Release

Sebastian Hahn

May 13, 2015

1	Contents	3
1.1	License	3
1.2	Developers	3
1.3	pygeobase	4
2	Indices and tables	7
	Python Module Index	9

This is the documentation of **pygeobase**.

Note: This is the main page of your project's [Sphinx](#) documentation. It is formatted in [reStructuredText](#). Add additional pages by creating `rst`-files in `docs` and adding them to the [toctree](#) below. Use then [references](#) in order to link them from this page. It is also possible to refer to the documentation of other Python packages with the [Python domain syntax](#). By default you can reference the documentation of [Sphinx](#), [Python](#), [matplotlib](#), [NumPy](#), [Scikit-Learn](#), [Pandas](#), [SciPy](#). You can add more by extending the `intersphinx_mapping` in your Sphinx's `conf.py`.

Contents

1.1 License

Copyright (c) 2015, Vienna University of Technology, Department of Geodesy and Geoinformation.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of pygeogrids nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.2 Developers

- Sebastian Hahn <sebastian.hahn@geo.tuwien.ac.at>
- Christoph Paulik <christoph.paulik@geo.tuwien.ac.at>
- Thomas Mistelbauer <thomas.mistelbauer@geo.tuwien.ac.at>

1.3 pygeobase

1.3.1 pygeobase package

Submodules

pygeobase.io_base module

class `pygeobase.io_base.GriddedStaticBase` (*path, grid, ioclass, mode='r', fn_format='{:04d}'*)

Bases: `object`

The GriddedStaticBase class uses another IO class together with a grid object to read/write a dataset under the given path.

Methods

<code>close()</code>	Close file.
<code>flush()</code>	Flush data.
<code>iter_gp()</code>	Yield all values for all grid points.
<code>read(*args, **kwargs)</code>	Takes either 1 or 2 arguments and calls the correct function
<code>read_gp(gpi)</code>	Read data for given grid point.
<code>write(data)</code>	Write data.
<code>write_gp(gpi, data)</code>	Write data for given grid point.

close()

Close file.

flush()

Flush data.

iter_gp()

Yield all values for all grid points.

read(*args, **kwargs)

Takes either 1 or 2 arguments and calls the correct function which is either reading the gpi directly or finding the nearest gpi from given lat,lon coordinates and then reading it

read_gp(gpi)

Read data for given grid point.

Parameters `gpi` : int

Grid point index.

Returns `data` : `numpy.ndarray`

Time series data.

write(data)

Write data.

Parameters `data` : `numpy.ndarray`

Data records. A field 'gpi', indicating the grid point index has to be included.

write_gp(gpi, data)

Write data for given grid point.

Parameters `gpi` : int

Grid point index.

data : numpy.ndarray

Data

class `pygeobase.io_base.GriddedTsBase` (*path, grid, ioclass, mode='r', fn_format='{:04d}'*)

Bases: `object`

The GriddedTsBase class uses another IO class together with a grid object to read/write a time series dataset under the given path.

Parameters `path` : string

Path to dataset.

grid : `pytesmo.grid.grids.BasicGrid` of `CellGrid` instance

Grid on which the time series data is stored.

ioclass : class

IO class

mode : str, optional

File mode and can be read 'r', write 'w' or append 'a'. Default: 'r'

cell_format : str, optional

The string format of the cell files. Default: '{:04d}'

Methods

<code>close()</code>	Close file.
<code>flush()</code>	Flush data.
<code>get_nearest_gp_info(lon, lat)</code>	get info for nearest grid point
<code>iter_ts()</code>	Yield time series for all grid points.
<code>read_gp(gpi, **kwargs)</code>	Reads time series for a given grid point index.
<code>read_ts(*args, **kwargs)</code>	Takes either 1 or 2 arguments and calls the correct function
<code>write_gp(gpi, data, **kwargs)</code>	Write data for given grid point.
<code>write_ts(*args, **kwargs)</code>	Takes either 2 or 3 arguments (the last one always needs to be the data to be written) and calls

close()

Close file.

flush()

Flush data.

get_nearest_gp_info (*lon, lat*)

get info for nearest grid point

Parameters `lon` : float

Longitude coordinate.

lat : float

Latitude coordinate.

Returns `gpi` : int

Grid point index of nearest grid point.

gp_lon : float

Longitude coordinate of nearest grid point.

gp_lat : float

Latitude coordinate of nearest grid point.

gp_dist : float

Geodetic distance to nearest grid point.

iter_ts ()

Yield time series for all grid points.

read_gp (*gpi*, ***kwargs*)

Reads time series for a given grid point index.

Parameters **gpi** : int

grid point index

Returns **data** : pandas.DataFrame

pandas.DataFrame with DateTimeIndex

read_ts (**args*, ***kwargs*)

Takes either 1 or 2 arguments and calls the correct function which is either reading the gpi directly or finding the nearest gpi from given lat,lon coordinates and then reading it

write_gp (*gpi*, *data*, ***kwargs*)

Write data for given grid point.

Parameters **gpi** : int

Grid point index.

data : numpy.ndarray

Data records.

write_ts (**args*, ***kwargs*)

Takes either 2 or 3 arguments (the last one always needs to be the data to be written) and calls the correct function which is either writing the gpi directly or finding the nearest gpi from given lon, lat coordinates and then reading it.

class pygeobase.io_base.**ImageBase**

Bases: `object`

class pygeobase.io_base.**StaticBase**

Bases: `object`

class pygeobase.io_base.**TsBase**

Bases: `object`

Module contents

Indices and tables

- `genindex`
- `modindex`
- `search`

p

`pygeobase`, [6](#)

`pygeobase.io_base`, [4](#)

C

close() (pygeobase.io_base.GriddedStaticBase method), 4
close() (pygeobase.io_base.GriddedTsBase method), 5

F

flush() (pygeobase.io_base.GriddedStaticBase method), 4
flush() (pygeobase.io_base.GriddedTsBase method), 5

G

get_nearest_gp_info() (pygeobase.io_base.GriddedTsBase method), 5

GriddedStaticBase (class in pygeobase.io_base), 4

GriddedTsBase (class in pygeobase.io_base), 5

I

ImageBase (class in pygeobase.io_base), 6

iter_gp() (pygeobase.io_base.GriddedStaticBase method), 4

iter_ts() (pygeobase.io_base.GriddedTsBase method), 6

P

pygeobase (module), 6

pygeobase.io_base (module), 4

R

read() (pygeobase.io_base.GriddedStaticBase method), 4

read_gp() (pygeobase.io_base.GriddedStaticBase method), 4

read_gp() (pygeobase.io_base.GriddedTsBase method), 6

read_ts() (pygeobase.io_base.GriddedTsBase method), 6

S

StaticBase (class in pygeobase.io_base), 6

T

TsBase (class in pygeobase.io_base), 6

W

write() (pygeobase.io_base.GriddedStaticBase method), 4

write_gp() (pygeobase.io_base.GriddedStaticBase method), 4

write_gp() (pygeobase.io_base.GriddedTsBase method), 6

write_ts() (pygeobase.io_base.GriddedTsBase method), 6