
Sippers Documentation

Release 1.1.1

GISCE-TI, S.L.

May 28, 2018

Contents

1	Quickstart	3
1.1	Installation	3
1.2	Importing a file	3
2	API	5
2.1	SIPS files	5
2.2	Adapters	6
2.3	Backends	8
2.4	Models	9
2.5	Parsers	9
2.6	Utils	10
2.7	Logging	10
3	Indices and tables	11
	Python Module Index	13

version: 1.1.1 Contents:

1.1 Installation

```
$ pip install sippers
```

1.2 Importing a file

```
$ sippers import --file /path/to/file \  
  --backend mongodb://localhost/db
```


2.1 SIPS files

class sippers.file.PackedSipsFile (path, strict=False)

Packed SIPS file.

Process of content of zip file is processed with iterators to keep the minimal memory footprint.

Example:

```
with PackedSipsFile('/tmp/PACKED.SIPS.zip') as packed:
    for sips_file in packed:
        for line in sips_file:
            print sips_file.stats.progress
            print line
        print packed.stats.progress
```

close()

Close the file descriptor.

class sippers.file.PackedSipsFileStats (size, n_files)

Stats for a Packed SIPS file (with zip).

progress

Progress in % and the number of files read.

Format example: 80% (8/10)

class sippers.file.SipsFile (path, fd=None, parser=None, strict=False, resume=None)

SIPS file.

Process of content of file is processed with iterators to keep the minimal memory footprint.

Example:

```
with SipsFile('/tmp/SIPS.TXT') as sips_file:
    for line in sips_file:
        print sips_file.stats.progress
        print line
```

Parameters

- **path** – Path of SIPS file
- **fd** – File descriptor (use this if you have already opened the file)
- **parser** – Force to use a parser
- **strict** – Strict schema validation
- **resume** (*SipsFileStats*) – Resume an imported file passing a *SipsFileStats*

class `sippers.file.SipsFileStats` (*size*)
Stats for a SIPS file

elapsed_time
Elapsed time processing the file.

progress
Progress in % of read content.

size
Human readable size of the SIPS file.

2.2 Adapters

class `sippers.adapters.MeasuresAdapter` (*extra=None, only=None, exclude=(), prefix=u", strict=None, many=False, context=None, load_only=(), dump_only=(), partial=False*)

Base Measures Adapter.

class `sippers.adapters.SipsAdapter` (*extra=None, only=None, exclude=(), prefix=u", strict=None, many=False, context=None, load_only=(), dump_only=(), partial=False*)

Base SIPS Adapter.

`sippers.adapters.pre_insert` (*fn=None, raw=False*)
Filter to use before inserting the document to the database.

Useful when you must to do some operation between the parsed data and the already inserted data.

Example for Hidrocantabrico when the measures file doesn't have the CUPS code and have the internal contract code:

```
@pre_insert
def fix_name(self, data):
    backend = self.backend
    result = backend.get(self.backend.ps_collection, {
        'ref': data['name'], 'cod_distri': '0026'
    })
    if result:
        data['name'] = result[0]['name']
    return data
```

2.2.1 Endesa

```
endesa.TARIFFS = {'20A': '2.0A', '20DHA': '2.0DHA', '20DHS': '2.0DHS', '21A': '2.1A', '21D': '2.1D'}
```

```
class sippers.adapters.endesa.EndesaBaseAdapter (extra=None, only=None, exclude=(),
                                                prefix="u", strict=None, many=False,
                                                context=None, load_only=(),
                                                dump_only=(), partial=False)
```

Endesa SIPS Adapter

fix_dates (data)

Fix the dates in the SIPS file from ENDESA.

In the endesa sips file dates are YYYYMMDD or 0 or 00000000. With this all fields.DateTime fields are caught and parsed to a correct format YYYY-MM-DDT00:00:00.

fix_floats (data)

Fix floats numbers.

Replace , to .

```
class sippers.adapters.endesa.EndesaMeasuresAdapter (extra=None, only=None,
                                                    exclude=(), prefix="u",
                                                    strict=None, many=False,
                                                    context=None, load_only=(),
                                                    dump_only=(), partial=False)
```

```
class sippers.adapters.endesa.EndesaSipsAdapter (extra=None, only=None, exclude=(),
                                                  prefix="u", strict=None, many=False,
                                                  context=None, load_only=(),
                                                  dump_only=(), partial=False)
```

Endesa SIPS Adapter.

adapt_tarifa (data)

Fix the ATR Tariff code

Using *TARIFFS*

```
sippers.adapters.endesa.TARIFFS = {'20A': '2.0A', '20DHA': '2.0DHA', '20DHS': '2.0DHS', '21A': '2.1A', '21D': '2.1D'}
```

Mapping of ENDESA tariffs codes.

2.2.2 Iberdrola

```
class sippers.adapters.iberdrola.IberdrolaMeasuresAdapter (extra=None, only=None,
                                                           exclude=(), prefix="u",
                                                           strict=None, many=False,
                                                           context=None, load_only=(),
                                                           dump_only=(), partial=False)
```

```
class sippers.adapters.iberdrola.IberdrolaSipsAdapter (extra=None, only=None,
                                                         exclude=(), prefix="u",
                                                         strict=None, many=False,
                                                         context=None, load_only=(),
                                                         dump_only=(), partial=False)
```

2.2.3 Hidrocantábrico

```
class sippers.adapters.hidrocantabrico.HidrocantabricoMeasuresAdapter (extra=None,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    pre-
                                                                    fix=u",
                                                                    strict=None,
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False)

class sippers.adapters.hidrocantabrico.HidrocantabricoSipsAdapter (extra=None,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    prefix=u",
                                                                    strict=None,
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False)
```

2.3 Backends

`sippers.backends.get_backend(url)`
Get the backend class by and URL.

Parameters `url` – URL for identify a backend.

`sippers.backends.register(name, cls)`
Register a backend

Use this function to register a Backend class for and schema. If you want to register your own backend you can do:

```
class YourAwesomeBackend(BaseBackend):
    pass

register('awesome', YourAwesomeBackend)
```

Then with the URL `awesome://user:pass@host/db` with the function `get_backend()`

Parameters

- **name** – Backend name
- **cls** (*class*) – Backend class

2.3.1 MongoDB

```
class sippers.backends.mongodb.MongoDBBackend(uri=None)
    MongoDB Backend

    insert_cnmc_measure (value)
        cnmc measures come a measure per line, cannot replace the whole block as in insert_measures
```

2.4 Models

```
class sippers.models.Document(data, adapter)
    Document object

    Parameters
        • data – Data parsed.
        • adapter – Adapter used to parse this data.

    This document is used to encapsulated a object in sippers.parsers.parser.Parser.parse_line()

    backend_data
        Get data after using the filter sippers.adapters.pre_insert()

class sippers.models.MeasuresSchema(extra=None, only=None, exclude=(), prefix="u",
                                     strict=None, many=False, context=None, load_only=(),
                                     dump_only=(), partial=False)

    Base model for measures.

class sippers.models.SipsSchema(extra=None, only=None, exclude=(), prefix="u", strict=None,
                                 many=False, context=None, load_only=(), dump_only=(),
                                 partial=False)

    Base model for SIPS
```

2.4.1 Endesa

2.4.2 Iberdrola

2.4.3 Hidrocarbúrico

2.5 Parsers

```
class sippers.parsers.parser.Parser
    Base parser interface.

    parse_line (line)
        Parse a line of a SIPS file.

        Parameters line – line of the file

sippers.parsers.parser.register(cls)
    Register a parser
```

2.5.1 Endesa

2.5.2 Iberdrola

2.5.3 Hidrocarb rico

2.6 Utils

Bits & Bytes related humanization.

Copyright (c) 2010 Jason Moiron and Contributors <https://github.com/jmoiron/humanize/blob/master/humanize/filesize.py>

`sippers.utils.build_dict(headers, data)`
Build a dict with headers and list of data.

Example:

```
build_dict(['foo', 'bar'], [1, 2])
{'foo': 1, 'bar': 2}
```

Parameters

- **headers** (*list*) – List of headers
- **data** (*list*) – List of data

`sippers.utils.naturalsize(value, binary=False, gnu=False, format='%i')`
Format a number of byteslike a human readable filesize (eg. 10 kB). By default, decimal suffixes (kB, MB) are used. Passing `binary=True` will use binary suffixes (KiB, MiB) are used and the base will be 2^{10} instead of 10^3 . If `gnu` is True, the binary argument is ignored and GNU-style (ls -sh style) prefixes are used (K, M) with the 2^{10} definition. Non-gnu modes are compatible with jinja2's `filesizeformat` filter.

2.7 Logging

2.7.1 sippers.logging

Implements the logging support for SIPPERS

You can use logging everywhere using:

```
from sippers import logger
logger.info('Info message')
```

`sippers.logging.setup_logging(level=None, logfile=None)`
Setups sippers logging system.

It will setup sentry logging if SENTRY_DSN environment is defined

Parameters

- **level** – logging.LEVEL to set to logger (defaults INFO)
- **logfile** – File to write the log

Returns logger

CHAPTER 3

Indices and tables

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

S

- `sippers.adapters`, [6](#)
- `sippers.adapters.endesa`, [7](#)
- `sippers.adapters.hidrocantabrico`, [8](#)
- `sippers.adapters.iberdrola`, [7](#)
- `sippers.backends`, [8](#)
- `sippers.backends.mongodb`, [9](#)
- `sippers.file`, [5](#)
- `sippers.logging`, [10](#)
- `sippers.models`, [9](#)
- `sippers.parsers.parser`, [9](#)
- `sippers.utils`, [10](#)

A

`adapt_tarifa()` (sippers.adapters.endesa.EndesaSipsAdapter method), 7

B

`backend_data` (sippers.models.Document attribute), 9
`build_dict()` (in module sippers.utils), 10

C

`close()` (sippers.file.PackedSipsFile method), 5

D

`Document` (class in sippers.models), 9

E

`elapsed_time` (sippers.file.SipsFileStats attribute), 6
`EndesaBaseAdapter` (class in sippers.adapters.endesa), 7
`EndesaMeasuresAdapter` (class in sippers.adapters.endesa), 7
`EndesaSipsAdapter` (class in sippers.adapters.endesa), 7

F

`fix_dates()` (sippers.adapters.endesa.EndesaBaseAdapter method), 7
`fix_floats()` (sippers.adapters.endesa.EndesaBaseAdapter method), 7

G

`get_backend()` (in module sippers.backends), 8

H

`HidrocantabricoMeasuresAdapter` (class in sippers.adapters.hidrocantabrico), 8
`HidrocantabricoSipsAdapter` (class in sippers.adapters.hidrocantabrico), 8

I

`IberdrolaMeasuresAdapter` (class in sippers.adapters.iberdrola), 7

`IberdrolaSipsAdapter` (class in sippers.adapters.iberdrola), 7
`insert_cnmc_measure()` (sippers.backends.mongodb.MongoDBBackend method), 9

M

`MeasuresAdapter` (class in sippers.adapters), 6
`MeasuresSchema` (class in sippers.models), 9
`MongoDBBackend` (class in sippers.backends.mongodb), 9

N

`naturalsize()` (in module sippers.utils), 10

P

`PackedSipsFile` (class in sippers.file), 5
`PackedSipsFileStats` (class in sippers.file), 5
`parse_line()` (sippers.parsers.parser.Parser method), 9
`Parser` (class in sippers.parsers.parser), 9
`pre_insert()` (in module sippers.adapters), 6
`progress` (sippers.file.PackedSipsFileStats attribute), 5
`progress` (sippers.file.SipsFileStats attribute), 6

R

`register()` (in module sippers.backends), 8
`register()` (in module sippers.parsers.parser), 9

S

`setup_logging()` (in module sippers.logging), 10
`sippers.adapters` (module), 6
`sippers.adapters.endesa` (module), 7
`sippers.adapters.hidrocantabrico` (module), 8
`sippers.adapters.iberdrola` (module), 7
`sippers.backends` (module), 8
`sippers.backends.mongodb` (module), 9
`sippers.file` (module), 5
`sippers.logging` (module), 10
`sippers.models` (module), 9

sippers.parsers.parser (module), [9](#)
sippers.utils (module), [10](#)
SipsAdapter (class in sippers.adapters), [6](#)
SipsFile (class in sippers.file), [5](#)
SipsFileStats (class in sippers.file), [6](#)
SipsSchema (class in sippers.models), [9](#)
size (sippers.file.SipsFileStats attribute), [6](#)

T

TARIFFS (in module sippers.adapters.endesa), [7](#)
TARIFFS (sippers.adapters.endesa attribute), [7](#)