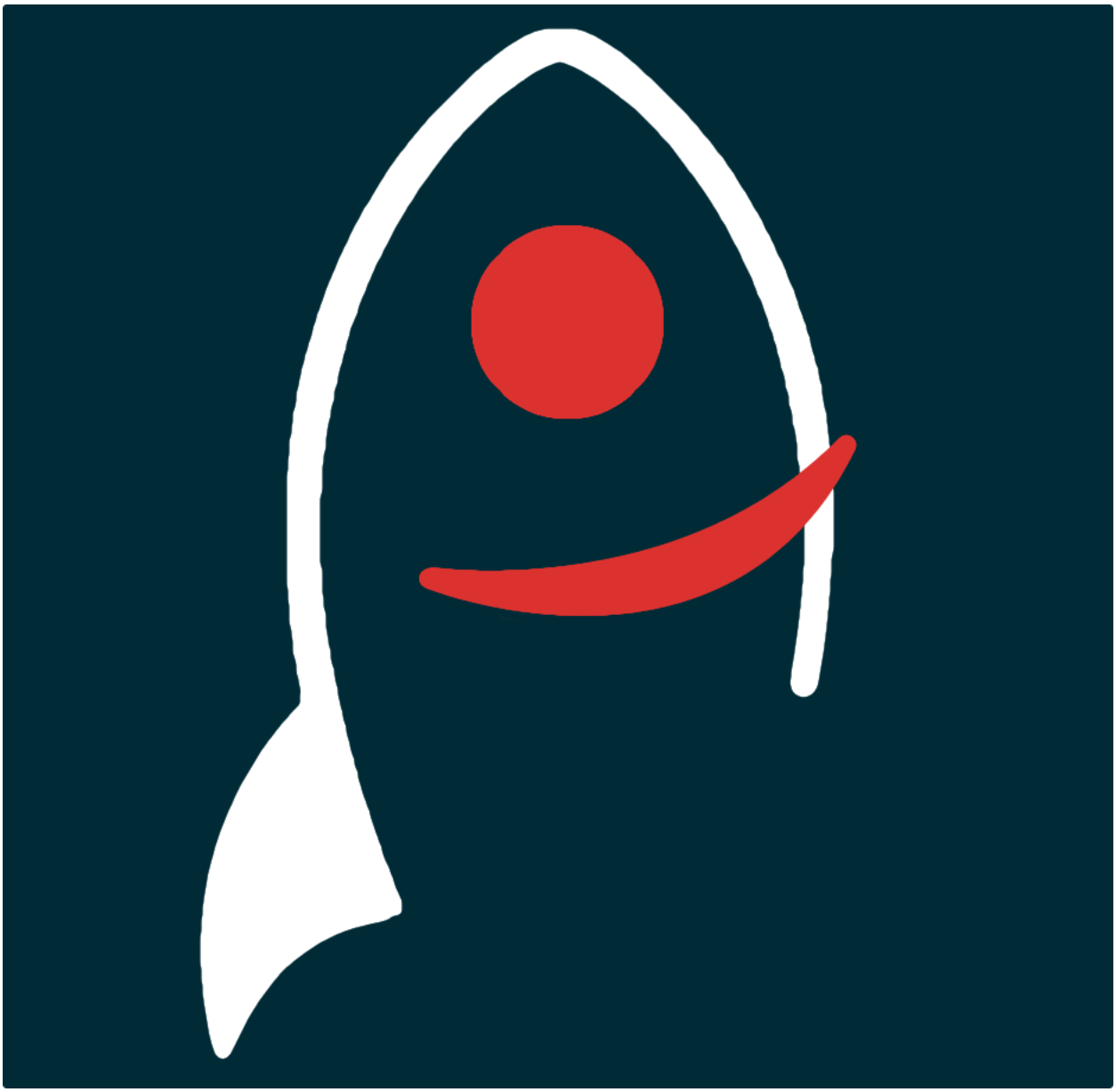


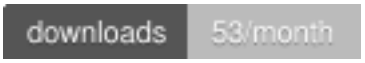
atelParser Documentation

Release v1.0.2

Dave Young

2023

TABLE OF CONTENTS



scrape and parse content of ATels posted on The Astronomer's Telegram website, identify individual objects by name and coordinates.

Documentation for atelParser is hosted by Read the Docs (last stable version and latest version). The code lives on github. Please report any issues you find here.

CHAPTER

ONE

FEATURES

- Report the latest ATel count
- Download all ATel as raw HTML pages. After a first download, can be run on a regular basis to download only new/missing ATels.
- Parse ATels to extract coordinates and transient source names to indexed MySQL database tables which can then be used in your own projects.

CHAPTER

TWO

HOW TO CITE ATELPARSER

If you use `atelParser` in your work, please cite using the following BibTeX entry:

```
@software{
    Young_atelParser,
    author = {Young, David R.},
    doi = {10.5281/zenodo.8037458},
    license = {GPL-3.0-only},
    title = ,
    url = {https://github.com/thespacedoctor/atelParser}
}
```

2.1 Installation

The easiest way to install atelParser is to use `pip` (here we show the install inside of a conda environment):

```
conda create -n atelParser python=3.7 pip
conda activate atelParser
pip install atelParser
```

Or you can clone the github repo and install from a local version of the code:

```
git clone git@github.com:thespacedoctor/atelParser.git
cd atelParser
python setup.py install
```

To upgrade to the latest version of atelParser use the command:

```
pip install atelParser --upgrade
```

To check installation was successful run `atelParser -v`. This should return the version number of the install.

2.1.1 Development

If you want to tinker with the code, then install in development mode. This means you can modify the code from your cloned repo:

```
git clone git@github.com:thespacedoctor/atelParser.git
cd atelParser
python setup.py develop
```

Pull requests are welcomed!

2.2 Initialisation

Before using atelParser you need to use the `init` command to generate a user settings file. Running the following creates a yaml settings file in your home folder under `~/.config/atelParser/atelParser.yaml`:

```
atelParser init
```

The file is initially populated with atelParser's default settings which can be adjusted to your preference.

If at any point the user settings file becomes corrupted or you just want to start afresh, simply trash the `atelParser.yaml` file and rerun `atelParser init`.

2.2.1 Modifying the Settings

Once created, open the settings file in any text editor and make any modifications needed. The most important setting is the `atel-directory` as this lets `atelParser` know where to download the ATel HTML files to. Change this value to your preferred location.

```
atel-directory: ~/git_repos/atel-archive/html
```

2.2.2 Basic Python Setup

If you plan to use `atelParser` in your own scripts you will first need to parse your settings file and set up logging etc. One quick way to do this is to use the `fundamentals` package to give you a logger, a settings dictionary and a database connection (if connection details given in settings file):

```
## SOME BASIC SETUP FOR LOGGING, SETTINGS ETC
from fundamentals import tools
from os.path import expanduser
home = expanduser("~")
settingsFile  = home + "/.config/atelParser/atelParser.yaml"
su = tools(
    arguments={"settingsFile": settingsFile},
    docString=__doc__,
)
arguments, settings, log, dbConn = su.setup()
```

2.3 Latest ATel Count

The simplest tool in the `ATelParser` toolbox is the latest ATel count, reporting the number of the last reported ATel.

2.3.1 From the Command-Line

To run the count from the command-line run:

```
> atel count
14318 ATels have been reported as of 2021/01/13 10:48:11s
```

2.3.2 From Python Code

To get the count from python use the `get_latest_atel_number` method:

```
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
latestNumber = atels.get_latest_atel_number()
```

2.4 Downloading ATels

2.4.1 From the Command-Line

To download new/missing ATels run `atel download` from the command-line:

```
> atel download
Waiting for a randomly selected 35s before downloading ATel #14317
Waiting for a randomly selected 101s before downloading ATel #14318
...
```

Note a random time between 0-180s is injected between ATel page downloads so not to overwhelm the ATel servers.

2.4.2 From Python Code

Before you begin to code you will need to parse your settings file and set up logging etc. One quick way to do this is to use the `fundamentals` package to give you a logger, a settings dictionary and a database connection (if connection details given in settings file):

```
## SOME BASIC SETUP FOR LOGGING, SETTINGS ETC
from fundamentals import tools
from os.path import expanduser
home = expanduser("~")
settingsFile  = home + "/.config/atelParser/atelParser.yaml"
su = tools(
    arguments={"settingsFile": settingsFile},
    docString=__doc__,
```

(continues on next page)

(continued from previous page)

```
)
arguments, settings, log, dbConn = su.setup()
```

Assuming you have set up your `atel-directory` location in the settings file (see Initialisation), you can download all new/missing ATels pages with the following code snippet.

```
## DOWNLOAD ALL NEW ATEL PAGES
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
atelsToDownload = atels.get_list_of_atels_still_to_download()
atels.download_list_of_atels(atelsToDownload)
```

Once run, you should find one HTML file per ATel in your `atel-directory` folder. You can find more information on the `download` class here

2.5 Parsing ATels To A Database

After downloading the ATel HTML files you now have the option of adding the content of the ATels to a MySQL database and to parse this content to generate indexed tables of coordinates and transient source names.

Connection details are needed in the ATel settings file for the parser to access the database.

The parser will create and populate the following 3 tables.

- `atel_fullcontent`: containing a list of ATels and their full-text content.
- `atel_names`: a list of transient source names found via regex matching of the ATel text content. Transients from new surveys and mangled names my get missed (please report via github issues if you find a problem).
- `atel_coordinates`: sky-position coordinates as parsed from the ATel content and converted to decimal degrees (also indexed via 3 different HTM level IDs). Some coordinates may have been missed if written in an obscure syntax (or just incorrectly).

The indexed transient source data in these tables can then be used in your own projects.

2.5.1 From the Command-Line

To parse the downloaded ATels from the command-line run:

```
> atel parse
```

2.5.2 From Python Code

If scripting the parsing of the ATels in your own code, use the `mysql` class to parse the ATels and ingest them into the MySQL database tables:

```
from atelParser import mysql
parser = mysql(
    log=log,
    settings=settings,
    reParse=reparseFlag
)
parser.atels_to_database()
parser.parse_atels()
parser.populate_htm_columns()
```

2.6 Todo List

Todo:

- Make sure todo list is working. ✓

(The *original entry* is located in /home/docs/checkouts/readthedocs.org/user_builds/atelparser/checkouts/master/docs/source/_template_
line 1.)

2.7 Release Notes

v1.0.2 - May 10, 2022

- **Fixed**: docs now building

v1.0.1 - January 14, 2021

- **Fixed**: dependency clash with other packages for pymysql version

v1.0.0 - January 13, 2021

- **ENHANCEMENT** full documentation

v0.4.0 - May 4, 2020

- Now compatible with Python 3.*
- **Fixed**: adding requests, pymysql and pandas as dependencies

CHAPTER

THREE

API REFERENCE

3.1 Modules

atelParser.commonutils	*common tools used throughout package*
atelParser.utKit	*Unit testing tools*

3.1.1 commonutils *(module)*

common tools used throughout package

3.1.2 utKit *(module)*

Unit testing tools

Classes

utKit(moduleDirectory[, dbConn])	*Override dryx utKit*

3.2 Classes

atelParser.download	*Download ATels as Raw HTML files*
atelParser.mysql	*Import ATel into MySQL database and parse for names and coordinates*

3.2.1 download *(class)*

class download(*log*, *settings=False*)
Bases: object

Download ATels as Raw HTML files

Key Arguments

- log – logger
- settings – the settings dictionary

Usage

To setup your logger, settings and database connections, please use the fundamentals package (see tutorial here).

To initiate a download object, use the following:

```
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
```

Methods

download_list_of_atels(atelNumbers)	*download the HTML files of all the missing ATels*
get_latest_atel_number()	*get latest atel number by parsing the RSS feed for the ATel site*
get_list_of_atels_still_to_download()	*get list of atels still to download by determining which ATels have been downloaded and diffing this against the latest ATel number*

download_list_of_atels(*atelNumbers*)
download the HTML files of all the missing ATels

Key Arguments

- atelNumbers – the list of ATel numbers to download

Usage

To download new and missing ATel to your atel-directory use this code:

```
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
atelsToDownload = atels.get_list_of_atels_still_to_download()
atels.download_list_of_atels(atelsToDownload)
```

get_latest_atel_number()
get latest atel number by parsing the RSS feed for the ATel site

Return

- number – the number of the latest ATel

Usage

```
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
latestNumber = atels.get_latest_atel_number()
```

get_list_of_atels_still_to_download()
get list of atels still to download by determining which ATels have been downloaded and diffing this against the latest ATel number

Return

- atelNumbersToDownload – a list of the ATel numbers that need downloaded

Usage

```
from atelParser import download
atels = download(
    log=log,
    settings=settings
)
atelsToDownload = atels.get_list_of_atels_still_to_download()
```

3.2.2 mysql *(class)*

class mysql(*log*, *settings=False*, *reParse=False*)
Bases: object

Import ATel into MySQL database and parse for names and coordinates

Key Arguments

- log – logger
- settings – the settings dictionary
- reParse – re-parse all existing atels? Useful if new names have been added to the parse-list

Usage

To setup your logger, settings and database connections, please use the fundamentals package (see tutorial here).

To initiate a mysql object, use the following:

```
from atelParser import mysql
parser = mysql(
    log=log,
    settings=settings
)
```

Methods

atels_to_database()	*Parse ATels into a mysql db.*
parse_atels()	*Parse the content of the ATels in the database*
populate_htm_columns()	*populate htm columns in the atel_coordinates table*

atels_to_database()
Parse ATels into a mysql db.

Parser to add ATels into a mysql db - each ATel has 'element' data (top level - title, author ...) and 'item' data (object specific data - ra, dec, mag, name ...). The parser will add one row per 'item' (object) into the db table

Usage

```
from atelParser import mysql
parser = mysql(
    log=log,
    settings=settings
)
parser.atels_to_database()
```

parse_atels()
Parse the content of the ATels in the database

Appending the various components and values to the db. Also includes the ability convert the atels to markdown, highlighting matches of the parsing regexs.

Usage

```
from atelParser import mysql
parser = mysql(
    log=log,
    settings=settings
)
parser.parse_atels()
```

populate_htm_columns()
populate htm columns in the atel_coordinates table

Usage

To add the HTM columns (levels 10, 13 & 16) to the atel_coordinates database add this to your code:

```
from atelParser import mysql
parser = mysql(
    log=log,
    settings=settings
)
parser.populate_htm_columns()
```

3.3 Functions

atelParser.mysql.clean_supernova_name	*Clean a SN name.*

3.3.1 clean_supernova_name (*function*)

clean_supernova_name(*log*, *snName*)

Clean a SN name.

This function will attempt to clean up the name so that it is somewhat homogeneous with SN/transient from the same survey/atel system.

Key Arguments

- `log` – logger
- `snName` – sn name to be cleaned (string)

Return

- `snName` – cleaned sn name (string)

3.4 A-Z Index

Modules

atelParser.commonutils	*common tools used throughout package*
atelParser.utKit	*Unit testing tools*

Classes

atelParser.download	*Download ATels as Raw HTML files*
atelParser.mysql	*Import ATel into MySQL database and parse for names and coordinates*

Functions

atelParser.mysql.clean_supernova_name	*Clean a SN name.*

CHAPTER

FOUR

RELEASE NOTES

v1.0.2 - May 10, 2022

- **Fixed**: docs now building

v1.0.1 - January 14, 2021

- **Fixed**: dependency clash with other packages for pymysql version

v1.0.0 - January 13, 2021

- **ENHANCEMENT** full documentation

v0.4.0 - May 4, 2020

- Now compatible with Python 3.*
- **Fixed**: adding requests, pymysql and pandas as dependencies

PYTHON MODULE INDEX

INDEX