maser-py Documentation

Release 0.1.0

X.Bonnin, B.Cecconi, Q.N.Nguyen

December 18, 2015

CONTENTS

CHAPTER

ONE

INTRODUCTION

The MASER python package contains modules to deal with services and data provided in the framework of the MASER project (Mesures, Analyses et Simulations d'Emissions Radio).

For more information about MASER, please visit: http://maser.lesia.obspm.fr/

CHAPTER

TWO

INSTALLATION

2.1 System Requirements

In order to install MASER, make sure to have Python 3.4 or higher available on your system.

The package installation requires the following Python modules: - setuptools (12.0.5 or higher)

If they are not found, the following Python packages will be downloaded automatically during the installation: - openpyxl - spacepy - simplejson

MASER-PY has been tested on the following Operating Systems: - Mac OS X 10.10, 10.11 - Debian Jessie 8.2

In order to use the "cdf" submodule, the NASA CDF software distribution shall be installed and configured on your system. Especially, make sure that the directory containing the CDF binary executables is on your $PATH, and the $CDF_LIB env. var. is set.

2.2 How to get MASER-PY

To download MASER-PY, enter the following command from a terminal:

```
git clone git://git.renater.fr/maser/maser-py.git
```

Make sure to have Git (https://git-scm.com/) installed on your system.

If everything goes right, you should have a new local "maser-py" directory created on your disk.

2.3 How to set up MASER-PY

To set up the package on your system, enter the following command from the "maser-py" directory:

```
python3 setup.py install
```

This should install the maser-py package on your system.

To check that the installation ends correctly, you can enter:

```
maser-py
```

, which should return something like:

"This is maser-py package VX.Y.Z"

If you have an issue durint installation, please read the "Troubleshooting" section for help.

2.4 How to run MASER-PY

If the installation has ended correctly, you can run MASER-PY:

- From a Python interpreter session, by entering “import maser”.
- Using the command line interface available for some MASER-PY modules.

For more details about the MASER-PY modules, please read the user manual.

CHAPTER

THREE

OVERVIEW

The MASER-PY package contains the following modules:

cdf Module to handle the NASA Common Data Format (CDF).

helio Module to get and plot the HELIO Virtual Observatory data.

stereo Module to handle the STEREO NASA mission data.

wind Module to handle the Wind NASA mission data.

tools Module containing common tool methods for MASER-PY

In order to work, the MASER-PY package modules rely on additional files and directories:

data Directory containing support data

CHAPTER

FOUR

THE *CDF* MODULE

The *cdf* module is divided in two submodules:

- *cdfconverter*, which allows users to convert CDF skeleton files into master CDF binary files
- *cdfvalidator*, which allows users to perform some validations on CDF files.

For more information about the CDF format, please visit http://cdf.gsfc.nasa.gov/.

4.1 The *cdfconverter* submodule

cdfconverter contains the following classes:

- *Xlsx2skt*, convert an Excel 2007 format file into a CDF skeleton table in the ASCII format. The organization of the Excel file shall follow some rules defined in the present document (see the section "Excel file format definition" below)
- *Skt2cdf*, convert a CDF skeleton table in ASCII format into a CDF master binary file. This module calls the "skeletoncdf" program from the NASA CDF software distribution.

Both classes can be imported from Python or called directly from a terminal using the dedicated command line interface.

4.1.1 The Xlsx2skt class

To import the Xlsx2skt class from Python, enter:

```
from maser.cdf.cdfconverter import Xlsx2skt
```

Excel file format definition

This section describes the organization of the input skeleton file in Excel format.

Note that:

- xlsx2skt supports the Excel 2007 format only (i.e., .xlsx).
- Only zVariables are supported

Make sure to respect the letter case, since the xlsx2skt parser is case sensitive!

The input Excel file shall contain the following sheets:

- header

- GLOBALattributes
- zVariables
- VARIABLEattributes
- Options
- NRV

The first row of each sheet shall be used to provide the name of the columns.

header sheet

The "header" sheet shall contain the following columns:

CDF_NAME Name of the CDF master file (without the extension)

DATA ENCODING Type of data encoding

MAJORITY Majority of the CDF data parsing ("COLUMN" or "ROW")

FORMAT Indicates if the data are saved in a single ("SINGLE") or on multiple ("MULTIPLE") CDF files

GLOBALattributes sheet

The "GLOBALattributes" sheet shall contain the following columns:

Attribute Name Name of the global attribute

Entry Number Index of the current entry starting at 1

Data Type CDF data type of the global attribute (only the "CDF_CHAR" type is supported)

Value Value of the current entry

zVariables sheet

The "zVariables" sheet shall contain the following columns:

Variable Name Name of the zVariable

Data Type CDF data type of the zVariable

Number Elements Number of elements of the zVariable (shall be always 1, except for CDF_[U]CHAR" type)

Dims Number of dimension of the zVariable (shall be 0 if the variable is a scalar)

Sizes If the variable is not a scalar, provides its dimension sizes.

Record Variance Indicates if the variable values can change ("T") or not ("F") from a record to another.

Dimension Variances Indicates how the variable values vary over each dimension.

VARIABLEattributes sheet

The "VARIABLEattributes" sheet shall contain the following columns:

Variable Name Name of the zVariable

Attribute Name Name of the variable attribute

Data Type CDF data type of the variable attribute

Value Value of the variable attribute

Options sheet

The "Options" sheet shall contain the following columns:

CDF_COMPRESSION Type of compression of the CDF file ("None" or empty field indicates no compression)

CDF_CHECKSUM Checksum algorithm of the CDF file ("None" or empty field indicates no checksumming)

VAR_COMPRESSION Type of compression of each CDF variable ("None" or empty field indicates no compression)

VAR_SPARSERECORDS value of sparese records ("None" or empty field indicates no sparese value)

VAR_PADVALUE padvalue to provide to each variable. This option only works in the case where all of the CDF variables has the same data type. In the other cases, users should use the –Auto_pad input keyword.

NRV sheet

The "NRV" sheet shall contain the following columns:

Variable Name Name of the zVariable

Index Index of the current NR row

Value Value of the current NR row

Command line interface

To display the help of the module, enter:

```
xlsx2skt --help
```

The full calling sequence is:

```
xlsx2skt [-h] [-O] [-V] [-Q] [-A] [-I] [-s [skeleton]] xlsx_file
```

Input keyword list:

-h, -help	Display the module help
-s, --skeleton	skeleton Name of the output skeleton table in ASCII format. If not provided, use the name of the input file replacing the extension by '.skt'.
-o, --output_dir	Path of the output directory. If not provided, use the directory of the input file.
-A, --Auto_pad	If provided, the module will automatically set the pad values (i.e, !VAR_PADVALUE) for each CDF variable
-I, --Ignore_none	If provided, the module will skip rows for which the Attribute/Variable name columns are empty. By default, the module returns an error if a empty Attribute/Variable name value is encountered.
-O, --Overwrite	Overwrite existing output ASCII skeleton table
-V, --Verbose	Talkative mode

Example

To test the cdfconverter program, use the dedicated scripts/test_cdfconverter.sh bash script.

Limitations & Known Issues

Here are some identified limitations to the module uses:

- Values provided in the "Options" sheet is valid for all of CDF file and variables. The module does not allow to set (yet) the values for each variable individually. **THUS, WE STRONGLY RECOMMEND TO USE THE –Auto_pad INPUT KEYWORD (then edit the resulting skeleton table to modify the !VAR_PADVALUE if required).**

4.1.2 The *Skt2cdf* class

To import the Skt2cdf class from Python, enter:

```
from maser.cdf.cdfconverter import Skt2cdf
```

Command line interface

To display the help of the module, enter:

```
skt2cdf --help
```

The full calling sequence is:

```
skt2cdf [-h] [-O] [-V] [-Q] [-s [executable]] [-c [output_cdf]] skeleton
```

Input keyword list:

-h, -help Display the module help

-c, --cdf output_cdf Name of the output CDF master binary file. If not provided, use the name of the input file replacing the extension by '.cdf'.

-o, --output_dir Path of the output directory. If not provided, use the directory of the input file.

-s, --skeletoncdf executable Path of the NASA GSFC CDF "skeletoncdf" executable. If not provided, the program will search for the executable in the $PATH env. variable.

-O, --Overwrite Overwrite existing output ASCII skeleton table

-V, --Verbose Talkative mode

-Q, --Quiet Quiet mode

Example

To test the cdfconverter program, use the dedicated scripts/test_cdfconverter.sh bash script.

4.2 The *cdfvalidator* submodule

The *cdfvalidator* submodule provides tools to validate a CDF format file.

It contains only one *Validate* class that regroups all of the validation methods.

4.2.1 The *Validate* class

To import the *Validate* class from Python, enter:

```
from maser.cdf.cdfvalidator import Validate
```

The Model validation test

The *Validate* class allows user to check if a given CDF format file contains specific attributes or variables, by providing a so-called "cdfvalidator model file".

This model file shall be in the JSON format. All items and values are case sensitive. It can include the following JSON objects:

Table 4.1: CDFValidator JSON objects

JSON object	Description
GLOBALattributes	Contains the list of global attributes to check
VARIABLEattributes	Contains the list of variable attributes to check
zVariables	Contains the list of zvariables to check

Note that any additional JSON object will be ignored.

The table below lists the JSON items that are allowed to be found in the *GLOBALattributes*, *VARIABLEattributes* and *zVariables* JSON objects.

Table 4.2: CDFValidator JSON object items

JSON item	JSON type	Priority	Description
attributes	vector	optional	List of variable attributes. An element of the vector shall be a JSON object that can contain one or more of the other JSON items listed in this table
dims	integer	optional	Number of dimensions of the CDF item
entries	vector	optional	Entry value(s) of the CDF item to be found
hasvalue	boolean	optional	If it is set to true, then the current CDF item must have at least one nonzero entry value
name	string	mandatory	Name of the CDF item (attribute or variable) to check
sizes	vector	optional	Dimension sizes of the CDF item
type	attribute	optional	CDF data type of the CDF item

Command line interface

To display the help of the module, enter:

```
cdfvalidator --help
```

The full calling sequence is:

```
cdfvalidator [--help] [--Verbose] [--Quiet] [--log_file [log_file]] \
[--ISTP] [--CDFValidate [executable]] [--model_file [model_file]] skeleton
```

Input keyword list:

-h, -help Display the module help

-l, --log_file Path of the output log file.

-I, --ISTP Perform the ISTP compliance validation test

-m, --model_file Path to the input model file in JSON format (see "Model validation test" section for more information).

-C, --CDFValidate executable Path of the NASA GSFC CDF "CDFValidate" executable. If it is not provided, the module will search in the directories defined in %%$PATH%%.

-Q, --Quiet Quiet mode

-V, --Verbose Talkative mode

Example

To test the cdfvalidator program, use the dedicated scripts/test_cdfvalidator.sh bash script.

CHAPTER

FIVE

THE *WIND* MODULE

The Wind module provides methods to deal with the Wind NASA mission data.

5.1 The *waves* submodule

CHAPTER

SIX

THE *STEREO* MODULE

The stereo module provides methods to deal with the STEREO NASA mission data.

6.1 The *swaves* submodule

CHAPTER

SEVEN

THE *HELIO* MODULE

The helio module provides methods to deal with the HELIO Virtual Observatory services and data.

7.1 The *hfc* submodule

CHAPTER

EIGHT

TROUBLESHOOTING

Here a list of known issues. If the problem persists, you can contact the MASER developer team at: maser.support@groupes.renater.fr.

- //I have an error message "[Errno 13] Permission denied:" during installation//:

This means that you don't have the right to install the package in your Python "site-packages" local directory. To solve this problem, install the package as a super user (e.g., using sudo command for instance), or modify the "site-packages" user access permissions.

- //I have an error message "Exception: Cannot find CDF C library. Try os.putenv("CDF_LIB", library_directory) before import." during the installation"//:

This means that the %%$CDF_LIB%% environmment variable is not set. This variable is required to run the CDF software distribution from Python. For more information about how to set up this software., visit the CDF home page at http://cdf.gsfc.nasa.gov/.

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex
- search