iradinaGUI Documentation

Release 1.0.0

CEA DEN/DANS/DM2S/STMF/LGLS

CONTENTS

1	User's manual	3
	1.1 Iradina GUI main widget	4
	1.2 Iradina tree view widget	5
	1.3 Log Run Code widget	7
	1.4 Explore Dir widget	9
	1.5 Plot widget	13
	1.6 Plot widget usage	16
	1.7 Tips	20
2	Programmer's guide	25
	2.1 Prerequisites	26
	2.2 All-in-one installation	26
	2.3 All-in-one installation Linux	26
	2.4 All-in-one installation Windows7-10	27
	2.5 Development installations	28
	2.6 IradinaGUI configuration	
	2.7 Usage of iradinaGUI	31
	2.8 Iradina code compilation	
	2.9 Documentation	34
3	Frequently Asked Questions	37
	3.1 Add an Item in this FAQ	
	3.2 FAQ	38
4	Code documentation	39
	4.1 Code documentation	40
5	Release Notes	75
	5.1 Release notes	75
Py	hon Module Index	77
In	ex	79



Warning:

- 1. Find a *pdf* version of this documentation here¹.
- 2. Find *Iradina code* manual here².

The iradinaGUI code is a GUI³ (Graphical User Interface) used to perform operations with Iradina code. This GUI code is a set of Python3⁴ scripts files.

CONTENTS 1

 $^{^1\} iradina GUI/doc/build/latex/iradina GUI.pdf$

² iradinaGUI/doc/src/iradinaDocuments/20140804_iradina_manual.pdf ³ https://en.wikipedia.org/wiki/Graphical_user_interface

⁴ https://docs.python.org/3.5

2 CONTENTS

CHAPTER

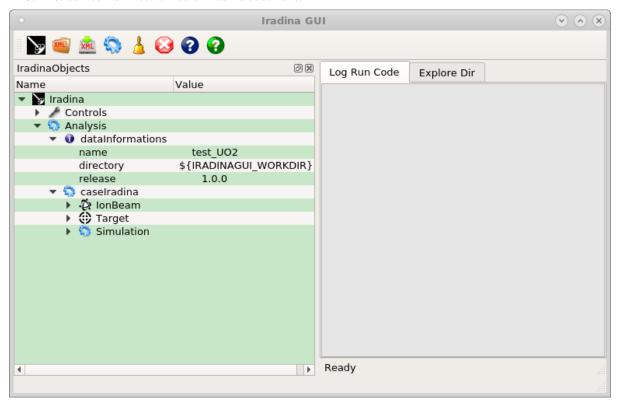
ONE

USER'S MANUAL

1.1 Iradina GUI main widget

From this main widget named Iradina GUI users can:

- 1. Prepare Iradina code data.
- 2. Run Iradina code.
- 3. Plot curves from result files of Iradina code runs.



1.1.1 Main widget toolbar



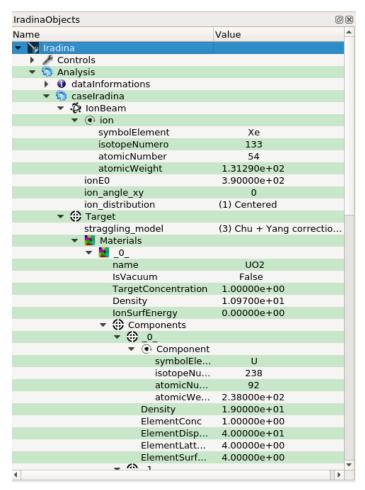
This toolbar contains icons related to actions, from left to right:

- 1. New Iradina data. Create iradina case from scratch
- 2. Load Iradina data. Load case from previously saved case in a file Xml.
- 3. Save Iradina data. Save current iradina case in a file Xml.
- 4. Launch Iradina calculus. Launch Iradina code on current iradina case.
- 5. Refresh IradinaObjects tree view.
- 6. Clear Iradina data model, remove Iradina data tree (in IradinaObject widget).
- 7. Iradina GUI help. Display this current documentation in a browser (html mode).
- 8. Iradina code help. Display Iradina code manual in a browser (pdf mode).

1.2 Iradina tree view widget

From this tree view widget named IradinaObjects users can:

1. Prepare Iradina code data.



1.2.1 Modify tree view widget items values

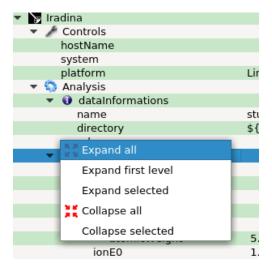
There are some values as leaves of tree. Names and tooltips are almost as Iradina code naming usage.

- 1. **Simple scalar values**. User can modify value on *mouse-left-double-click*, selecting tree item nodes **hovering column value**.
- 2. **Other specific values**. User can modify value on *mouse-right-click*, to get a contextual menu for modification, selecting tree item nodes **hovering columns name and value**.

1.2.2 Tree view widget menus

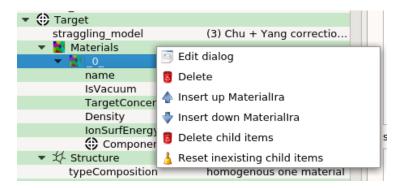
There are some menus, as *contextual menu* on *mouse-right-click* when selected tree item nodes. Some menu are generic, and other are specific to node, as contextual actions. This is a **NOT exhaustive** list of menus:

Expand/collapse menu



This menu contains some actions to expand or collapse all or selected part of data tree. To activate this menu users have to *mouse-right-click* on **head of arrow** of tree item nodes (at **left** of item icon).

Delete/Insert menu



This menu contains some actions to insert, delete and reset all or selected part of data tree. To activate this menu users have to *mouse-right-click* on 'name' of tree item nodes (at right of item icon). The concerned items are usually not leaves (are items without a value).

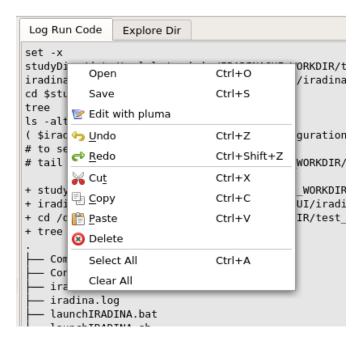
1.3 Log Run Code widget

This widget displays log trace of Iradina code execution.

Iradina code is executed when users activate Launch Iradina calculus button in Main widget toolbar (page 4).

```
Log Run Code
                Explore Dir
                                                                                        .
set -x
studyDir=/data/tmplgls/wambeke/IRADINAGUI_WORKDIR/test_U02
iradinaExe=/data/tmplgls/wambeke/iradinaGUI/iradinaCode/iradina_linux64.exe
cd $studyDir
tree
ls -alt
( $iradinaExe -p 9 -data ../data -c ./Configuration.in | tee ./iradina.log )&
# to see iradina progress
# tail -f /data/tmplgls/wambeke/IRADINAGUI_WORKDIR/test_UO2/iradina.log
+ studyDir=/data/tmplgls/wambeke/IRADINAGUI WORKDIR/test UO2
+ iradinaExe=/data/tmplgls/wambeke/iradinaGUI/iradinaCode/iradina_linux64.exe
+ cd /data/tmplgls/wambeke/IRADINAGUI_WORKDIR/test_U02
   · Composition.in
    Configuration.in
    iradinaGui.xml
   iradina.log
    launchIRADINA.bat
    launchIRADINA.sh
   Materials.in
    output
       ira.disp.sum
      ira.disp.z8.m16.000.mat0.elem1
       ira.disp.z92.m238.000.mat0.elem0
      - ira.information
      - ira.int.sum
       ira.int.z8.m16.000.mat0.elem1
      - ira.int.z92.m238.000.mat0.elem0
Ready
```

1.3.1 Log Run Code widget menu



This menu contains some actions to display, but **also** edit all or selected part of current log trace, considering log trace as an ascii file. To activate this menu users have to *mouse-right-click* **somewhere in** Log Run Code widget.

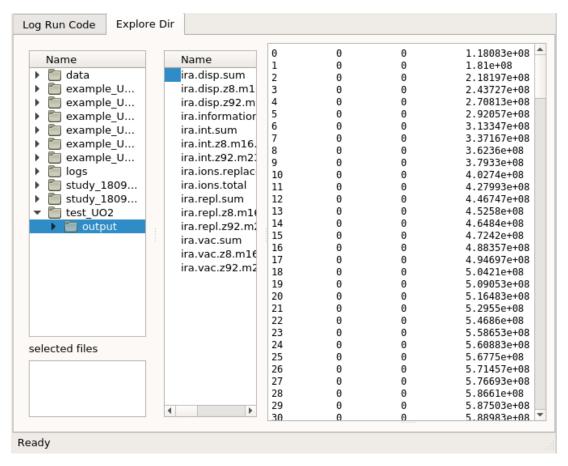
Note: Using *Open* and *Save* actions in this menu, users can use this widget as an elementary text file viewer/editor.

1.4 Explore Dir widget

This widget displays the contents of user iradinaGUI working directory. This directory is usually referenced as *IRADINAGUI_WORKDIR*.

Its usage is like a **simple** file explorer.

Note: Theses result files of Iradina code are located in sub-directories named output.



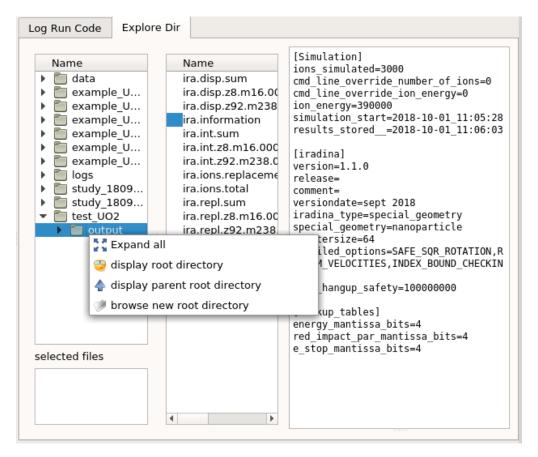
There are three main widgets (from left to right):

- 1. Directories names widget
- 2. Files names widget
- 3. File contents widget

Note: The lower *selected files* widget is for future improvments, no usage *yet*.

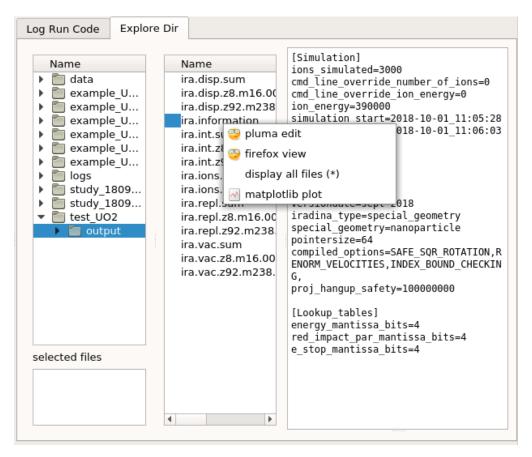
There are some contextual menus to explore directories, and to display input/output text files.

1.4.1 Directories names widget menu



This menu contains some elementary actions to navigate in **all** disk directories.

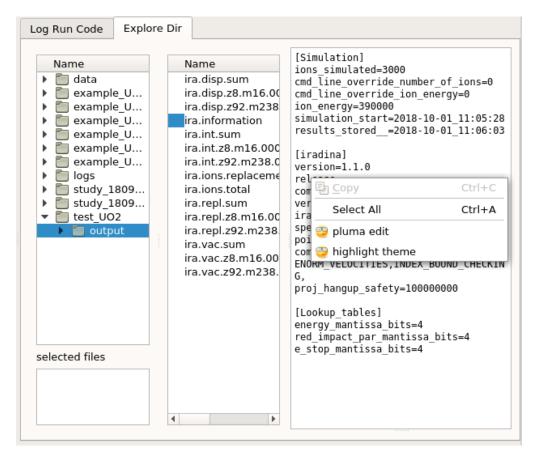
1.4.2 Files names widget menu



This menu contains some actions to apply on selected file.

Note: An useful action is the lower **matplotlib plot**, to plot curves from result files of Iradina code as post-treatment.

1.4.3 Files contents widget menu



This menu contains some elementary actions to apply on displayed file. The files are *syntax highlighted* if possible, using highlight⁵ tool, only on Linux distributions for now.

Warning: The lower *highlight theme* action is displayed only for Linux distribution.

⁵ http://www.andre-simon.de/doku/highlight/en/highlight.php

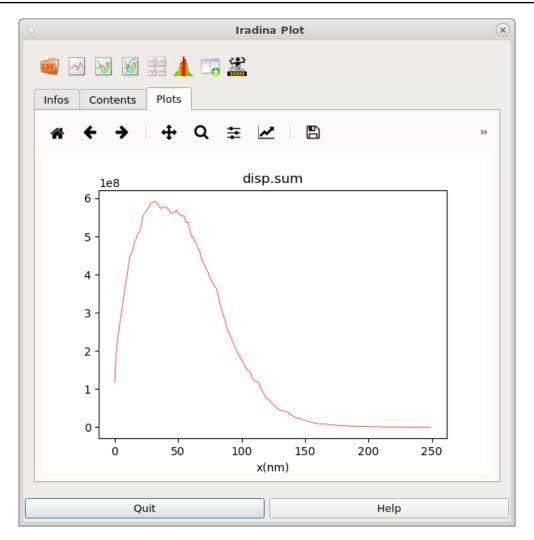
1.5 Plot widget

This widget is to explore, modify and make plots from data. It appears when users activate **matplotlib plot** in *Files names widget menu* (page 11).

It uses some important Python packages:

- 1. PANDAS⁶ (Python Data Analysis Library) to read and store data in memory.
- 2. MATPLOTLIB⁷ (Python 2D plotting library) to plot curves.

Warning: Data are read from CSV⁸ formated files. Note that result files of Iradina code use whitespace/tabulation for separator character, *not comma*, for readability.



There are four main widgets (from up to down, left to right):

- 1. Plots toolbar widget
- 2. Infos tab widget
- 3. Contents tab widget
- 4. Plots tab widget

1.5. Plot widget

⁶ https://pandas.pydata.org

⁷ https://matplotlib.org

⁸ https://en.wikipedia.org/wiki/Comma-separated_values

1.5.1 Plot toolbar widget



This menu contains some buttons, as actions, which are, from left to right:

- 1. Read a new data set from file, CSV⁹ formatted.
- 2. Plot curve y=f(x). 2 columns (x,y) selected.
- 3. Plot 2 curves $(y_1,y_2)=f(x)$, two distinct Y-axis. 3 columns (x,y_1,y_2) selected.
- 4. Plot n curves $y1, \dots yn = f(x)$, one Y-axis, (n+1) columns $(x, y1, \dots yn)$ selected.
- 5. Plot n distinct curves $y1, \dots yn = f(x), (n+1)$ columns $(x,y1, \dots yn)$ selected.
- 6. Plot histogram, one columns selected, (automatic number of intervals).
- 7. Execute PANDAS¹⁰ expressions, modification **on the fly** of *Contents tab widget*.
- 8. Initialize data ellipse in Contents tab widget (as a newbie example), plot the ellipse.

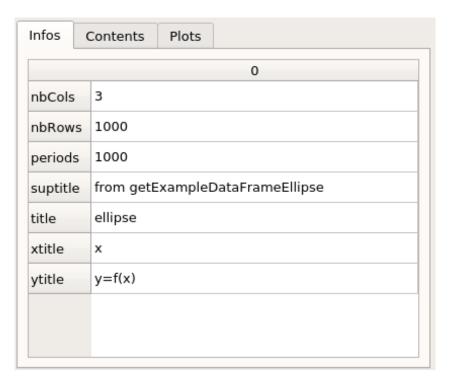
With this toolbar, user may process:

- 1. Select one, or more columns in *Contents tab widget*, then call plot of curve(s) using plot buttons of this plot toolbar. A MATPLOTLIB plot is displayed in *Plots tab widget*. See *Example of user plot* (page 18).
- 2. Create, remove, calculate new(s) colums of data in *Contents tab widget*. See *Example of user PANDAS expression* (page 16).
- 3. Filter, sort etc. lines of data in *Contents tab widget*, that means also get (and trace) a subset of *Contents tab widget*.

⁹ https://en.wikipedia.org/wiki/Comma-separated_values

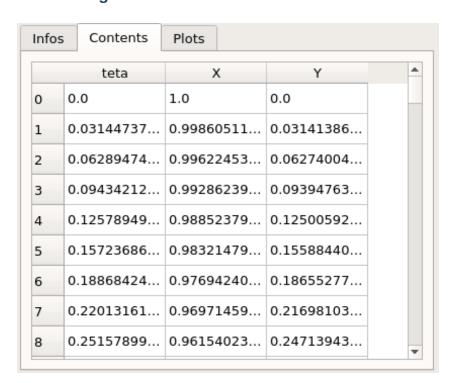
¹⁰ https://pandas.pydata.org

1.5.2 Infos tab widget



This widget contains some contextual informations from CSV data file, if they are present.

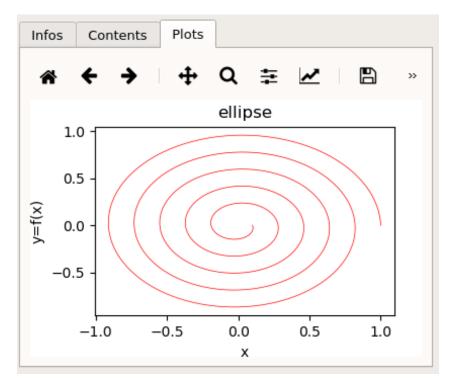
1.5.3 Contents tab widget



This widget displays current data, allowing some *elementary* modifications (see your appropriate spreadsheet for more functionalities (Excel, or else).

1.5. Plot widget

1.5.4 Plots tab widget



This widget displays current plot, as an instance of matplotlib¹¹ figure.

1.5.5 Interactive navigation toolbar matplotlib



Using this standard toolbar of matplotlib¹², user may customize current plot.

For more informations, users should refer to the description of navigation toolbar MATPLOTLIB¹³.

1.6 Plot widget usage

1.6.1 Example of user PANDAS expression

User may modify data of *Contents tab widget* (page 15), Clicking on menu actions button *Execute PANDAS* expressions of *Plot toolbar widget* (page 14).

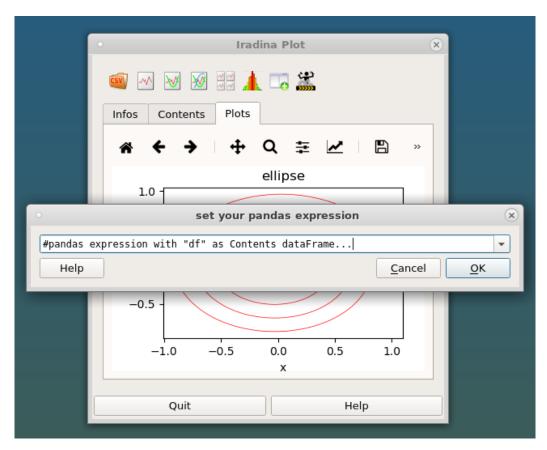


A dialog box appears set your pandas expression. An expression is an python affectation expression 'df = something', where 'df' is a pandas dataframe instance displayed in Contents tab widget (page 15).

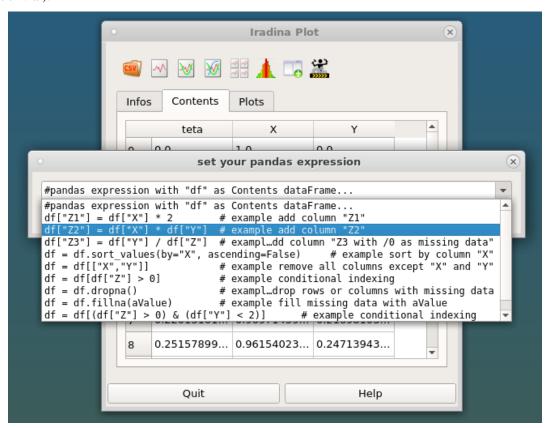
¹¹ https://matplotlib.org

¹² https://matplotlib.org

¹³ http://matplotlib.org/users/navigation_toolbar.html?highlight=toolbar



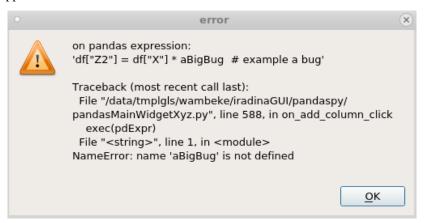
Scrolling, there are some examples of PANDAS¹⁴ dataframe expressions, (read the documentation, sometimes it is **not trivial**).



Users have to type their own expression, and confirm (OK). Do not worry, in case of error(s) in expression, an

¹⁴ https://pandas.pydata.org

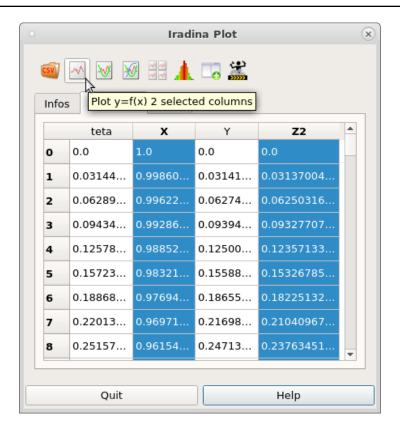
error dialog box appears.



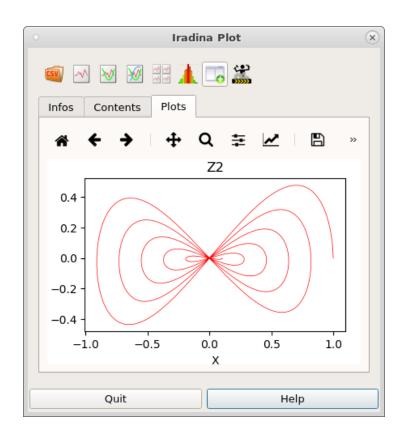
1.6.2 Example of user plot

When selecting, *for example*, theses two columns from ellipse: (X, Z2=X*Y), and Clicking on button 2 of *Plot toolbar widget* (page 14).

Warning: Sometimes, users have to make a **multiple-selection** of *expected* number of colums in *Contents tab widget* **before** plot actions. The order of multiple-selection is significant.



User get this plot



1.7 Tips

1.7.1 Tip 01



1.7.2 Tip 05

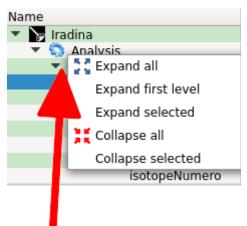


Click here to restore your iradina case data

1.7.3 Tip 08

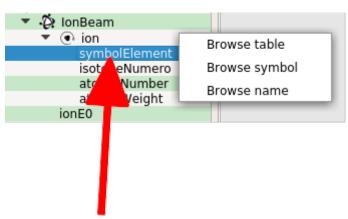


1.7.4 Tip 20



Right Click here to get tree expand menu

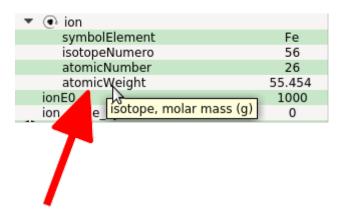
1.7.5 Tip 25



Right Click here to get contextual menu

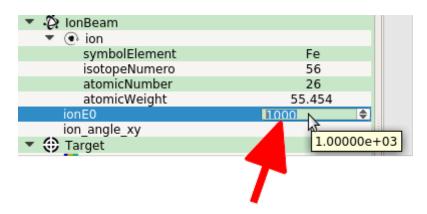
1.7. Tips 21

1.7.6 Tip 28



Mouse over to get contextual help tooltip

1.7.7 Tip 30



Double left Click here to modify values

1.7.8 Tip 98



Click here to read the iradina-code manual

1.7.9 Tip 99



Click here to read the full-fledged manual

1.7. Tips 23

CHAPTER	
•·······	
TWO	

PROGRAMMER'S GUIDE

2.1 Prerequisites

There are some definitions, and links.

- 1. Iradina¹⁵ code, and its manual.
- 2. PYTHON¹⁶ 3.5, with packages PyQt5¹⁷, numpy, matplotlib¹⁸, pandas¹⁹, etc. (usually named py3qt5).
- 3. PyInstaller²⁰ 3.4, free sofware to make iradinaGUI bundle (*only* valid for Linux).
- 4. 7-zip²¹, free sofware to compress/uncompress .7z files (for Windows installations).

Installation needs a PYTHON²² 3.6 interpreter, which is included in *All-in-one* iradina installations (Linux *and* Windows). see *Python installation Linux* (page 28).

PyInstaller²³ is a program that freezes Python programs in *bundle*, which is **almost** a python package. For more information about *bundle*, see PyInstaller manual²⁴.

2.2 All-in-one installation

These installations contain in **one** compressed file:

- 1. the iradinaGUI python scripts.
- 2. An interpreter PYTHON²⁵.
- 3. The Iradina code (GPL), its source files, and two executable files, one for Linux and one Windows.
- 4. The useful Corteo²⁶ data base (4bits).

Warning: Corteo data base used version in iradinaGUI is NOT Version 20160816.

2.3 All-in-one installation Linux

Warning: This is a PyInstaller²⁷ bundle, installed locally **where users want**. Python interpreter (named py3qt5) is simultaneously installed,

Source tar file *iradinaGUI_bundle_xxxx.tgz* is the **one** compressed archive file of the PyInstaller bundle in *one folder* mode. See more information here²⁸.

There are two ways to install iradinaGUI:

1. Install iradinaGUI directly, using usual *file manager* functionalities: uncompress tar file in user's choice directory.

¹⁵ https://www.nano.uni-jena.de/en/iradina.html

¹⁶ https://docs.python.org/3.5

¹⁷ https://pypi.org/project/PyQt5

¹⁸ https://matplotlib.org

¹⁹ https://pandas.pydata.org

²⁰ https://www.pyinstaller.org

²¹ https://www.7-zip.org

https://docs.python.org/3.5

²³ https://www.pyinstaller.org

²⁴ https://pyinstaller.readthedocs.io/en/stable

²⁵ https://docs.python.org/3.5

²⁶ http://www.lps.umontreal.ca/~schiette/index.php?n=Recherche.Corteo

²⁷ https://www.pyinstaller.org

²⁸ https://pyinstaller.readthedocs.io/en/stable/operating-mode.html#bundling-to-one-folder

2. Install iradinaGUI typing bash command, in usual terminal:

```
# install
                            # which is really where you want
cd yourChoiceDirectory
tar -xf .../iradinaGUI_bundle_xxxx.tgz
# launch
cd iradinaGUI_bundle
                        # folder name as linux pyinstaller bundle
./iradinaGUI -h
                           # on line help
./iradinaGUI -g -w ...
                           # launch GUI
```

2.4 All-in-one installation Windows7-10

Warning:

- 1. This is **not** a PyInstaller²⁹ bundle.
- 2. The **mandatory located** root directory is *C:\Users\Public\iradina*.
- 3. The **mandatory located** iradinaGUI directory is *C:\Users\Public\iradina\iradinaGUI*.
- 4. The **mandatory located** Python interpreter py3qt5 directory is *C:\Users\Public\iradina\miniconda3*.
- 5. The sofware tool to uncompress .7z files is 7-zip³⁰, which **has to be installed**.

Source .7z file *iradinaGUI_xxxxx*.7z is the **one** compressed archive file.

There are two ways to install iradinaGUI:

- 1. Install iradinaGUI directly, using usual file manager functionalities: uncompress .7z file in mandatory C:\Users\Public directory.
- 2. Install iradinaGUI typing DOS command, in usual (Windows7/10-cmd.exe shell):

```
rem install
cd C:\Users\Public # this is mandatory location, useful for all users
"C:\Program Files\7-Zip\7z.exe" x .../iradina_xxxx.7z
C:\User\Public\iradina\iradinaGUI\LaunchIradinaGUI.bat
```

Note: To launch iradina GUI, you may use Windows shortcut C:\User\Public\iradina\iradinaGUI\LaunchIradinaGUI(.lnk)

²⁹ https://www.pyinstaller.org

³⁰ https://www.7-zip.org

2.5 Development installations

A development installation of iradinaGUI allows programmer improvements. It is a classical usage of Python³¹ packages. It needs a directory for Python³² interpreter (usually named *miniconda3*), and another directory for iradinaGUI scripts (usually named *iradinaGUI*),

In fine, user could find (and use) command iradinaGUI directly after a detar/unzip installation. Or a git clone.

Warning:

- 1. **Windows7-10** all-in-one installation is a *development installation* of iradinaGUI, which allows programmer improvements.
- 2. Users find miniconda3 and iradinaGUI directories in parent directory named C:\Users\Public\iradina.
- 3. **Linux** all-in-one installation PyInstaller³³ bundle is **NOT** like that.
- 4. To get *development installation* **Linux** (freely located) of iradinaGUI, users have to follow the two next chapters.

2.5.1 Python installation Linux

To install python3 (and its mandatory packages PyQt5 etc.) *locally*, we suggest to use miniconda³⁴. Note that *miniconda* is windows7-10 compliant.

Note: You may use this Python interpreter for another python scripting code than iradinaGUI.

For information:

- 1. https://conda.io/miniconda.html
- 2. https://conda.io/docs/index.html

Example of install (Linux-bash):

```
bash Miniconda3-latest-Linux-x86_64.sh
# -> Miniconda3 will now be installed into this location:
# -> /volatile/common/miniconda3 (for example. It is located as you want.)
# -> Thank you for installing Miniconda3!
export PATH=/volatile/common/miniconda3/bin:$PATH
which conda
# -> /volatile/common/miniconda3/bin/conda
conda create -- name py3qt5 python=3 \
   pip jupyter matplotlib numpy pandas pandas-datareader \
   pyqt=5 scipy sympy jsonschema pyyaml libxml2 paramiko
# -> Solving environment: done
\# \rightarrow Proceed ([y]/n)? y
# -> Downloading and Extracting Packages
# -> To activate this environment, use:
# -> source activate py3qt5
conda info --envs
# -> conda environments:
```

(continues on next page)

³¹ https://docs.python.org/3.5

³² https://docs.python.org/3.5

³³ https://www.pyinstaller.org

³⁴ http://conda.pydata.org/miniconda.html

(continued from previous page)

2.5.2 iradinaGUI installation Linux

Warning: Python interpreter py3qt5 is supposed to be set and useful in environment path. Usually command *source activate py3qt5* assume that.

Example of install/launch (*Linux-bash*):

```
cd whereYouWant
tar -xf .../iradinaGUI_xxxx.tgz
cd iradinaGUI
ls -l iradinaGUI  # the launch executable command (is a script python)
which python  # --> py3qt5
./iradinaGUI -h  # on line help
./iradinaGUI -g -w ... # launch GUI
```

2.5.3 iradinaGUI installation Windows7-10

Warning: Python interpreter py3qt5 is supposed to be set and useful in environment path, at **mandatory** usual location *C:\Users\Public\iradina\miniconda3*. Usually command *conda activate py3qt5* assume that.

Example of install/launch (Windows7/10-cmd.exe shell), using 7-zip³⁵:

```
C:\
cd C:\Users\Public\iradina  # this is mandatory location, useful for all users
"C:\Program Files\7-Zip\7z.exe" x .../iradinaGUI_xxxxx.7z
cd C:\Users\Public\iradina\iradinaGUI
where python  # --> py3qt5
python iradinaGUI -h  # on line help
python iradinaGUI -g -w ...  # launch GUI
```

Note: To launch iradina GUI, you may use Windows shortcut $C:\User\Public\iradina\IradinaGUI\LaunchIradinaGUI\(.lnk)$.

³⁵ https://www.7-zip.org

2.6 IradinaGUI configuration

IradinaGUI uses files to store its configuration parameters. It uses ConfigParser³⁶ package, from The Python Standard Library.

Two configuration files are created or used at iradinaGUI launch, and located at IRADINAGUI_WORKDIR directory.

- 1. file .../IRADINAGUI_WORKDIR/iradinaGUI_user.cfg
- 2. file .../IRADINAGUI_WORKDIR/iradinaGUI_default.cfg

Note: User may edit/modify file iradinaGUI_user.cfg

2.6.1 Syntax

See https://docs.python.org/3/library/configparser.html

2.6.2 Description

The effective configuration **is a merge** of these two previous files. Parameters in *iradinaGUI_user.cfg* override parameters in *iradinaGUI_default.cfg*.

User will find **all allowed parameters** in systematically up-to-dated *iradinaGUI_default.cfg*.

 $^{^{36}\} https://docs.python.org/3/library/configparser.html$

2.7 Usage of iradinaGUI

2.7.1 **Usage**

IradinaGUI usage is a Command Line Interface (CLI³⁷), which is Windows and Linux compatible.

```
iradinaGUI --[options]
```

Options of iradinaGUI

Useful but not exhaustive generic options of iradinaGUI CLI.

Option -help or -h

Get help as simple text.

```
iradinaGUI --help # get list of existing options
```

Option -doc or -d

Get documentation as browser html.

```
iradinaGUI --doc # see html doc
```

Option -verbose or -v

Change verbosity level (default is 'info').

```
# execute iradinaGUI command in verbose debug mode
iradinaGUI -v debug
```

Option -workdir or -w

Change working directory (user data directory). Default is ../IRADINAGUI_WORKDIR

```
# execute iradinaGUI in user choice working directory
iradinaGUI -w .../MY_WORKDIR
```

³⁷ https://en.wikipedia.org/wiki/Command-line_interface

2.8 Iradina code compilation

IradinaGUI uses a specific version of Iradina code, modified by J.P. Crocombette (cea), which is tagged version 1.1.x for now. This code comes from original version 1.0.8 by Christian Borschel.

Users find two current compiled executable files, which are used by iradinaGUI, located at .../iradinaGUI/iradinaCode.

- 1. iradina_mingw64.exe, compiled by MinGW38 gcc compiler, for Windows (64 bits).
- 2. iradina_linux64.exe, compiled by GNU³⁹ gcc⁴⁰ compiler, for Linux (64 bits).

A development installation of Iradina code allows programmer improvments. The following chapters explain Iradina code compilation processes.

2.8.1 Iradina code sources

With GPL licence, sources are available in iradinaGUI directory tree, located at .../iradinaGUI/iradinaCodes/iradina_cea.

User find also the useful Corteo⁴¹ data base, located at .../iradinaGUI/iradinaCodes/data_4bit

```
.../iradinaCodes > tree
   data_4bit
      10.asp
      - 11.asp
etc.
      corteo.mat
      — erfinv.dat
   doc
       - 20140804_iradina_manual.pdf
       - Corteo20160816.pdf
       - iradina-1-s2.0-S0168583X11006318-main.pdf
      Iradina_tuto_installation.pdf
    iradina_cea

    compileIradina.bat

       - fileio.c
       - fileio.h
       - fromcorteo.c
       - fromcorteo.h
       - geometry.c
       - geometry.h
       - indexvalues6bit.h
       indexvalues.h
       iradina.c
        iradina.h
      - license.txt
      makefile_cea
       target.c
       target.h
       transport.c
       transport.h
       utils.c
       - utils.h
   compileIradina.lnk
  - README.txt
```

³⁸ http://www.mingw.org

³⁹ https://www.gnu.org/home.en.html

⁴⁰ https://www.gnu.org/software/gcc

⁴¹ http://www.lps.umontreal.ca/~schiette/index.php?n=Recherche.Corteo

2.8.2 Iradina code compilation Linux

Example of compilation (*Linux-bash*):

```
# this is your location
cd .../iradinaGUI/iradinaCodes/iradina_cea
# verifications
cat README.txt
# compilation
make -f makefile_cea clean
make -f makefile_cea iradina
make -f makefile_cea installGUI # install executable in iradinaGUI/iradinaCode
```

2.8.3 Iradina code compilation Windows7-10

Warning:

- 1. $MinGW^{42}$ is supposed to be set and useful in environment path, at an usual location C:MinGW (for example).
- 2. Git-windows⁴³ is supposed to be set and useful in environment path, at an usual location *C:\Program Files\Git* (for example). In order to use like-Linux commands.

Example of compilation (Windows7/10-cmd.exe shell):

```
# this is mandatory location
C:\
cd C:\Users\Public\iradina\iradinaGUI\iradinaCodes\iradina_cea
# verifications
where make  # --> C:\MinGW\bin\make.exe
where gcc  # --> C:\MinGW\bin\gcc.exe
where uname  # --> C:\Program Files\Git\usr\bin\uname.exe
# compilation
make -f makefile_cea clean
make -f makefile_cea iradina
make -f makefile_cea installGUI # install executable in iradinaGUI/iradinaCode
```

Note: To launch Iradina code compilation, you may use Windows shortcut $C:\User\Public\iradina\Codes\compileIradina(.lnk)$.

⁴² http://www.mingw.org

⁴³ https://git-scm.com

2.9 Documentation

2.9.1 Doc consultation

To display iradinaGUI html documentation in your web browser *firefox*, or *else*. The initial entry file is located at *iradinaGUI/doc/build/html/index.html*.

```
# Linux bash, as an example
cd .../iradinaGUI
firefox doc/build/html/index.html &
# or as CLI_
iradinaGUI --doc
```

2.9.2 Doc modification

To modify iradinaGUI documentation with simple editor pluma, or else.

Read the manual, see http://www.sphinx-doc.org/en/stable/tutorial.html, or may be copy/paste from 'Show Source' item.

```
# Linux bash, as an example
cd ...iradinaGUI/
tmp=$(find doc -name "*.rst")
pluma $tmp &
```

2.9.3 Doc compilation Linux

On a Linux system, to compile iradinaGUI html documentation, programmers use installed GNU⁴⁴ *make*, and SPHINX⁴⁵.

Warning: To make documentation **pdf** programmers needs installed *texlive* package (preferably up to date version). See: https://www.tug.org/texlive/quickinstall.html

```
cd ...iradinaGUI/doc
cat README # read some environment setup information
# ... and read it
make
 Please use `make <target>' where <target> is one of
 html          to make standalone HTML files
dirhtml         to make HTML files named index.html in directories
 singlehtml to make a single large HTML file
 pickle to make pickle files
          to make JSON files
 ison
 htmlhelp to make HTML files and a HTML help project
 qthelp to make HTML files and a qthelp project
 devhelp to make HTML files and a Devhelp project
 epub to make an epub latex to make LaTeX f
           to make LaTeX files, you can set PAPER=a4 or PAPER=letter
 latexpdf to make LaTeX files and run them through pdflatex
 text to make text files
 linkcheck to check all external links for integrity
                                                                    (continues on next page)
```

 $^{^{44}}$ https://www.gnu.org/home.en.html

⁴⁵ http://sphinx-doc.org

(continued from previous page)

```
doctest to run all doctests embedded in the documentation (if enabled)

# and then
make html  # make html doc
make latexpdf  # make pdf doc
```

2.9. Documentation 35

СНАРТ	ER
THRE	ΞE

FREQUENTLY ASKED QUESTIONS

3.1 Add an Item in this FAQ

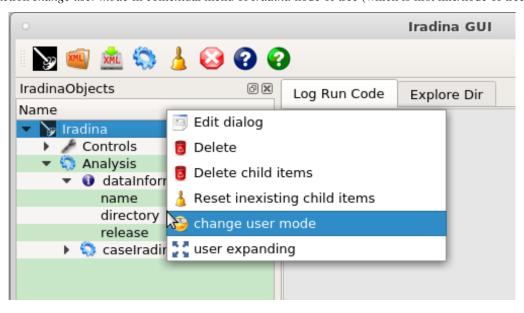
Edit iradinaGUI/doc/src/FAQ.rst and read documentation Doc compilation Linux (page 34).

3.2 **FAQ**

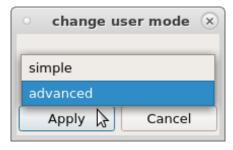
3.2.1 Why there is NOT all options of Iradina code in IradinaObjects tree?

Default user mode is set to **simple**, for beginners use.

Select action change user mode in contextual menu of Iradina node of tree (which is first line/node of tree).



Then Apply as advanced.



User can edit/modify value *General.usermode* parameter in the *IradinaGUI configuration* (page 30), to get a permanent setting.

3.2.2 Next question?

Here there is your next answer etc.

CHAPT	ER
ΕΩI	ID

CODE DOCUMENTATION

4.1 Code documentation

4.1.1 iradinapy

iradinapy package

Subpackages

iradinapy.colorama package

Submodules

iradinapy.colorama.ansi module

This module generates ANSI character codes to printing colors to terminals. See: http://en.wikipedia.org/wiki/ANSI_escape_code

```
class iradinapy.colorama.ansi.AnsiBack
    Bases: iradinapy.colorama.ansi.AnsiCodes (page 40)
    BLACK = 40
    BLUE = 44
    CYAN = 46
    GREEN = 42
    LIGHTBLACK_EX = 100
    LIGHTBLUE_EX = 104
    LIGHTCYAN_EX = 106
    LIGHTGREEN_EX = 102
    LIGHTMAGENTA_EX = 105
    LIGHTRED_EX = 101
    LIGHTWHITE_EX = 107
    LIGHTYELLOW_EX = 103
    MAGENTA = 45
    RED = 41
    RESET = 49
    WHITE = 47
    YELLOW = 43
class iradinapy.colorama.ansi.AnsiCodes
    Bases: object
class iradinapy.colorama.ansi.AnsiCursor
    Bases: object
    BACK (n=1)
    DOWN (n=1)
    FORWARD (n=1)
    POS (x=1, y=1)
```

```
UP (n=1)
class iradinapy.colorama.ansi.AnsiFore
    Bases: iradinapy.colorama.ansi.AnsiCodes (page 40)
    BLUE = 34
    CYAN = 36
    GREEN = 32
    LIGHTBLACK EX = 90
    LIGHTBLUE EX = 94
    LIGHTCYAN_EX = 96
    LIGHTGREEN_EX = 92
    LIGHTMAGENTA EX = 95
    LIGHTRED_EX = 91
    LIGHTWHITE EX = 97
    LIGHTYELLOW EX = 93
    MAGENTA = 35
    RED = 31
    RESET = 39
    WHITE = 37
    YELLOW = 33
class iradinapy.colorama.ansi.AnsiStyle
    Bases: iradinapy.colorama.ansi.AnsiCodes (page 40)
    BRIGHT = 1
    DIM = 2
    NORMAL = 22
    RESET_ALL = 0
iradinapy.colorama.ansi.clear_line(mode=2)
iradinapy.colorama.ansi.clear_screen(mode=2)
iradinapy.colorama.ansi.code_to_chars(code)
iradinapy.colorama.ansi.set_title(title)
iradinapy.colorama.ansitowin32 module
                                                                    convert=None,
class iradinapy.colorama.ansitowin32.AnsiToWin32 (wrapped,
                                                      strip=None, autoreset=False)
    Bases: object
    Implements a 'write()' method which, on Windows, will strip ANSI character sequences from the text, and
    if outputting to a tty, will convert them into win32 function calls.
    ANSI\_CSI\_RE = re.compile('\x01?\x1b\\[((?:\d|;)*)([a-zA-Z])\x02?')
    ANSI_OSC_RE = re.compile('\x01?\x1b\\]((?:.|;)x?)(\x07)\x02?')
    call_win32 (command, params)
```

```
convert_ansi (paramstring, command)
     convert_osc(text)
     extract_params (command, paramstring)
     get_win32_calls()
     reset_all()
     should_wrap()
         True if this class is actually needed. If false, then the output stream will not be affected, nor will
         win32 calls be issued, so wrapping stdout is not actually required. This will generally be False on
         non-Windows platforms, unless optional functionality like autoreset has been requested using kwargs
         to init()
     write(text)
     write_and_convert (text)
         Write the given text to our wrapped stream, stripping any ANSI sequences from the text, and optionally
         converting them into win32 calls.
     write_plain_text (text, start, end)
class iradinapy.colorama.ansitowin32.StreamWrapper(wrapped, converter)
     Bases: object
     Wraps a stream (such as stdout), acting as a transparent proxy for all attribute access apart from method
     'write()', which is delegated to our Converter instance.
     write (text)
iradinapy.colorama.ansitowin32.is_a_tty(stream)
iradinapy.colorama.ansitowin32.is_stream_closed(stream)
iradinapy.colorama.initialise module
iradinapy.colorama.initialise.colorama_text(*args, **kwargs)
iradinapy.colorama.initialise.deinit()
iradinapy.colorama.initialise.init(autoreset=False,
                                                                              strip=None,
                                                             convert=None,
                                           wrap=True)
iradinapy.colorama.initialise.reinit()
iradinapy.colorama.initialise.reset_all()
iradinapy.colorama.initialise.wrap_stream(stream, convert, strip, autoreset, wrap)
iradinapy.colorama.win32 module
iradinapy.colorama.win32.SetConsoleTextAttribute(*_)
iradinapy.colorama.win32.winapi_test(*_)
```

iradinapy.colorama.winterm module

```
class iradinapy.colorama.winterm.WinColor
    Bases: object
    BLACK = 0
    BLUE = 1
    CYAN = 3
    GREEN = 2
    GREY = 7
    MAGENTA = 5
    RED = 4
    YELLOW = 6
class iradinapy.colorama.winterm.WinStyle
    Bases: object
    BRIGHT = 8
    BRIGHT_BACKGROUND = 128
    NORMAL = 0
class iradinapy.colorama.winterm.WinTerm
    Bases: object
    back (back=None, light=False, on_stderr=False)
    cursor_adjust (x, y, on_stderr=False)
    erase_line (mode=0, on_stderr=False)
    erase_screen (mode=0, on_stderr=False)
     fore (fore=None, light=False, on_stderr=False)
    get_attrs()
    get_position(handle)
    reset_all (on_stderr=None)
     set_attrs(value)
     set_console (attrs=None, on_stderr=False)
     \verb|set_cursor_position|| (position=None, on\_stderr=False)|
     set_title(title)
     style (style=None, on_stderr=False)
```

Module contents

iradinapy.example package

Submodules

iradinapy.example.essai_logging_1 module

http://sametmax.com/ecrire-des-logs-en-python/ https://docs.python.org/3/library/time.html#time.strftime

essai utilisation logger plusieurs handler format different

```
/usr/lib/python2.7/logging/__init__.pyc
init MyLogger, fmt='%(asctime)s :: %(levelname)-8s :: %(message)s', level='20'
2018-03-11 18:51:21 :: INFO :: test logger info 2018-03-11 18:51:21 :: WARNING :: test logger warning 2018-03-11 18:51:21 :: ERROR :: test logger error 2018-03-11 18:51:21 :: CRITICAL :: test logger critical
init MyLogger, fmt='None', level='10'
2018-03-11 18:51:21 :: DEBUG :: test logger debug test logger debug 2018-03-11 18:51:21 :: INFO :: test logger info test logger info 2018-03-11 18:51:21 :: WARNING :: test logger warning test logger warning 2018-03-11 18:51:21 :: ERROR :: test logger error test logger error 2018-03-11 18:51:21 :: CRITICAL :: test logger critical test logger critical
iradinapy.example.essai_logging_1.getMyLogger(fmt=None, level=None)
iradinapy.example.essai_logging_1.testLogger1()
```

iradinapy.example.essai_logging_2 module

http://sametmax.com/ecrire-des-logs-en-python/ https://docs.python.org/3/library/time.html#time.strftime

essai utilisation logger un handler format different sur info() pas de format et su other format

```
/usr/lib/python2.7/logging/__init__.pyc
init MyLogger, fmt='%(asctime)s :: %(levelname)-8s :: %(message)s', level='20'
test logger info 2018-03-11 18:51:51 :: WARNING :: test logger warning 2018-03-11 18:51:51 ::
ERROR :: test logger error 2018-03-11 18:51:51 :: CRITICAL :: test logger critical

class iradinapy.example.essai_logging_2.MyFormatter (fmt=None, style='%')
Bases: logging.Formatter

format (record)
```

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

```
iradinapy.example.essai_logging_2.getMyLogger()
iradinapy.example.essai_logging_2.initMyLogger(fmt=None, level=None)
iradinapy.example.essai_logging_2.testLogger1()
```

Module contents

Submodules

```
iradinapy.abcdExpression module
from '3(a2bc)' to 'abbcabbcabbc' without regexp, not recursive for not smart poor people
iradinapy.abcdExpression.getIndiceFromChar(aChar)
     returns 0 for 'a', 1 for 'b' etc., max is 'z'
iradinapy.abcdExpression.toAbcd(aStr, verbose=False, details=False)
     '10(abc)' to return '10*(a+b+c)' raise exception if problem
iradinapy.abcdExpression.toEval0123(aStr, verbose=False)
     '3(a2bc)' to return [0,1,1,2,0,1,1,2,0,1,1,2] for 'abbcabbcabbc' raise exception if problem
iradinapy.abcdExpression.toEvalAbcd(aStr, verbose=False)
     '3(a2bc)' to return 'abbcabbcabbc' raise exception if problem
iradinapy.abcdExpression.toEvalAbcdForTooltip(aStr, length=20)
     set results in lenght characters lines
iradinapy.analysisIra module
class iradinapy.analysisIra.AnalysisIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     appendHistoryFileManager(action)
     createDocLaunch()
     getActionsContextMenu()
     gitCommit()
     gitkLaunch()
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     packageLaunch()
     postTreatments()
     printROOTContext()
     runPythonCode()
     searchURANIEMethod()
     setDefaultValues()
```

toFileIra()

updateRootlogonLaunch()

not virtual, could be used

```
class iradinapy.analysisIra.AttributeDataFrameIra(value=None)
    Bases: xyzpy.intFloatListXyz.StrXyz
```

```
drawGraphic()
drawUnivariate()
getActionsContextMenu()
getAttributeName()
```

```
getExpressionsInParent()
    getFileInParent()
class iradinapy.analysisIra.DataInformationsIra
    Bases: xyzpy.baseXyz._XyzConstrainBase
    general informations about
    getEtudeWorkdir()
         return as os.path.join(directory,name)
    getEtudeWorkdirBrut()
         return as ${IRADINAGUI WORKDIR}
    getVersion()
    isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     setDefaultValues()
         not virtual, could be used
class iradinapy.analysisIra.DataIra
    Bases: xyzpy.baseXyz._XyzConstrainBase
    DataFrame
    copyFileInData()
    drawDialogDataServer()
    getActionsContextMenu()
    getAllAttributesName()
    getNameExpanded()
     setAttributes()
         append attributes from '#COLUMN_NAMES' from file .dat, ordered as useful in uranie
class iradinapy.analysisIra.DataManagerIra(nameObject=")
    Bases: iradinapy.analysisIra.ListOfFileViewerXyz (page 47)
     copyAllFileInData()
    getActionsContextMenu()
class iradinapy.analysisIra.ExpressionIra(value=None)
    Bases: xyzpy.intFloatListXyz.ExpressionXyz
    createEditorData(parent=None)
    drawGraphic()
    drawUnivariate()
    getActionsContextMenu()
         no browse
    getAllAttributesNameInParents()
    getAttributeName()
         synonym for coherency with AttributeIra
    getExpressionsInParent()
    getFileInParent()
    getName()
     isNameUnique(name)
         test unicity of name in DataIra search in lists DataIra.attributes and DataIra.expressions
```

```
isValidExpression(value)
         test if expression is valid in a uranie TDataServer
class iradinapy.analysisIra.FileIra(value=None)
     Bases: xyzpy.intFloatListXyz.FileViewerXyz
     initialize type of files extension for interest .C etc.
class iradinapy.analysisIra.FunctionIra(value=None)
     Bases: iradinapy.analysisIra.FileIra(page 47)
     initialize type of files extension for function files
class iradinapy.analysisIra.HistoryFileManagerXyz
     Bases: xyzpy.baseXyz._XyzConstrainBase
     store in string for xml save history on some files, with hashing control as cp, mv etc... actions from
     gui/model actions one line by action
     appendHistoryAction(action)
     appendHistoryCopyOf (originFile, newFile)
     clearHistory()
     getCompleteFileName (aFile)
     getFileHash(aFile)
     getIdentFile (aFile)
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     setDefaultValues()
         not virtual, could be used
class iradinapy.analysisIra.LibraryIra(value=None)
     Bases: iradinapy.analysisIra.FileIra (page 47)
     initialize type of files extension for libraries files .C .so
class iradinapy.analysisIra.ListOfAttributeIra(nameObject=")
     Bases: xyzpy.baseXyz.ListOfBaseXyz
     getAllAttributesName()
class iradinapy.analysisIra.ListOfExpressionIra(nameObject=")
     Bases: xyzpy.baseXyz.ListOfBaseXyz
     getAllAttributesName()
     qetExpressions()
class iradinapy.analysisIra.ListOfFileViewerXyz (nameObject=")
     Bases: xyzpy.baseXyz.ListOfBaseXyz
     base class, used only inheritage, modify _allowedClasses etc
     addItem(aClass=None)
         override method ListOfBaseXyz
     addItems (aClass=None)
     addItemsSlot (status, aClass=None)
         new pyqt5 state override in lambda
    browseViewerDialog()
    browseViewerExecOnApply (selectedFiles)
         part of Apply on browseViewerDialog
     getActionsContextMenu()
```

```
getDirectory()
         could be dynamic in inheritage
    getNamesExpanded()
    getNoLocal()
    getTargetDirectory()
         could be dynamic in inheritage
class iradinapy.analysisIra.ListOfFunctionIra(nameObject=")
    Bases: iradinapy.analysisIra.ListOfFileViewerXyz (page 47)
class iradinapy.analysisIra.ListOfLibraryIra(nameObject=")
    Bases: iradinapy.analysisIra.ListOfFileViewerXyz (page 47)
class iradinapy.analysisIra.ListOfMacroIra(nameObject=")
    Bases: iradinapy.analysisIra.ListOfFileViewerXyz (page 47)
class iradinapy.analysisIra.ListOfUserFileIra(nameObject=")
    Bases: iradinapy.analysisIra.ListOfFileViewerXyz (page 47)
     getDirectory()
         dynamic override _directory
    getTargetDirectory()
         dynamic override _targetDirectory
class iradinapy.analysisIra.MacroIra(value=None)
    Bases: iradinapy.analysisIra.FileIra (page 47)
    initialize type of files extension for macro .C .py
class iradinapy.analysisIra.MacroManagerIra
    Bases: xyzpy.baseXyz._XyzConstrainBase
     getNamesExpanded()
    getNoLocal()
    setDefaultValues()
         not virtual, could be used
class iradinapy.analysisIra.UserFileIra(value=None)
    Bases: xyzpy.intFloatListXyz.FileViewerXyz
    initialize type of files extension for user interest
iradinapy.analysisIra.drawGraphic(self)
    common method draw histogram for classes as AttributeIra or ExpressionIra
iradinapy.analysisIra.drawUnivariate(self)
    common method draw univariate for classes as AttributeIra or ExpressionIra
iradinapy.analysisIra.fn_heterogenous_random_multiple_materials(case, aS-
iradinapy.analysisIra.fn_homogenous_one_material(case, aStream, options={/})
iradinapy.analysisIra.fn_multiLayer_multiple_materials(case,
                                                                       aStream,
                                                                tions=\{\})
iradinapy.analysisIra.getCurrentRowColumn (indiceCurrent,
                                                                nbmaxGrid.
                                                                             rowBe-
                                                 fore=True)
    returns current (row, column) in nbmax elements in grid for indice current (0 to nbmaxGrid)
iradinapy.analysisIra.getDefaultRowColumn(nb)
    return (row, column)
```

```
iradinapy.analysisIra.getRandomConcMaterial(irandoms)
     returns random indice in Materials (using TargetConcentration) irandoms is list as cumul of TargetConcen-
iradinapy.analysisIra.qet value random multiple materials(case)
     return random value for integer indice in Materials using getRandomConcMaterial
iradinapy.analysisIra.join(*v)
     as os.path.join but set antislash as slash, even for windows, keep windows 'c:'
iradinapy.analysisIra.normalize(aList)
iradinapy.analysisIra.toFileCompositionIn(case, aStream, name)
     see 'Creating new composition file' line 1222 in utils.C
iradinapy.analysisIra.toFileConfigurationIn(case, aStream, name=")
iradinapy.analysisIra.toFileMaterialIn(case, aStream, name=")
iradinapy.analysisIra.toFileStructureIn(case, aStream, name=")
iradinapy.analysisIra.toValue(aStr, name, value)
iradinapy.caselradina module
class iradinapy.caseIradina.BeamSpreadIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 1.
class iradinapy.caseIradina.BoolFalseIra(value=None)
     Bases: xyzpy.intFloatListXyz.BoolXyz
     ['False', 'True'] with write config file iradina strCfg [0,1] default False
     strCfg()
class iradinapy.caseIradina.BoolTrueIra(value=None)
     Bases: iradinapy.caseIradina.BoolFalseIra(page 49)
     ['False', 'True'] with write config file iradina strCfg [0, 1] default True
class iradinapy.caseIradina.CaseIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about case and for launch iradina
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     setDefaultValues()
         not virtual, could be used
class iradinapy.caseIradina.CellCountxIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntSupEq1Xyz
     int positive with default 100
class iradinapy.caseIradina.CellCountyIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntSupEq1Xyz
     int positive with default 1
class iradinapy.caseIradina.CellCountzIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntSupEq1Xyz
     int positive with default 1
```

```
class iradinapy.caseIradina.CellDepthxIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 1000. (nm)
     toXml(**kwargs)
         set tooltip_1 attibute xml for tooltip as long name element (value column 1)
class iradinapy.caseIradina.CellMultiLayerxIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrXyz
     'abcd' values for multiLayer description
     getCount()
         returns len(self) as 8 for '2(abcd)'
     getIndiceFromChar(aChar)
         returns 0 for 'a', 1 for 'b' etc.
     getMultiLayerMaterial (indice, verbose=False)
     toEval0123()
         returns [0,1,2,3,0,1,2,3] for '2(abcd)'
     toEvalAbcd()
         returns 'abcdabcd' for '2(abcd)'
class iradinapy.caseIradina.CellSizexIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 10.
class iradinapy.caseIradina.CellSizeyIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 100.
class iradinapy.caseIradina.CellSizezIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 100.
class iradinapy.caseIradina.CompositionFileTypeIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrInListXyz
     If the file is just one column of values then FileType should be set to 1. If the file contains 4 columns (x, y,
     z, value) then set it to 0.
     strCfg()
         get index from '(i) blah blah'
class iradinapy.caseIradina.ConcentrationIra(value=None, minMax=None)
     Bases: xyzpy.intFloatListXyz.FloatRangeXyz
     initial value as 1.
class iradinapy.caseIradina.DensityIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 0. supposed g/cm3, have to convert to at/cm3
     getActionsContextMenu()
         append action 'Set defaut value'
     getCalculatedValue()
         prorata ElementConc(s) and Components densities a trivial approximation
     normalize (aList)
     setCalculatedValue()
```

```
strCfg()
         iradina needs density of the material in atoms/cm3
         prorata ElementConc(s) and Components atomic weights
     toXml(**kwargs)
         set tooltip_1 attibute xml for tooltip as long name element (value column 1)
class iradinapy.caseIradina.DensityTargetComponentIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     TODO could set default density from wiki localMendeleiev or user config
     getActionsContextMenu()
         append action 'Set defaut value'
     getUserConfigValue()
     getWikipediaValue()
     setDefaultDensityUser()
     setDefaultDensityWiki()
class iradinapy.caseIradina.DisplayIntervalIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntSupEq1Xyz
     int positive with default 20
class iradinapy.caseIradina.ElementCountIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntRangeXyz
     as define MAX_NO_MATERIALS 20 as Maximum number of different materials
class iradinapy.caseIradina.ElementReplEnergyIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatXyz
     float positive with default value -1 as None
class iradinapy.caseIradina.FlightLengthTypeIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrInListXyz
     strCfq()
         get index from '(i) blah blah'
class iradinapy.caseIradina.FloatListIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrXyz
     accept [val1, val2, val3]
class iradinapy.caseIradina.IonAngleIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntRangeXyz
     angle incidence xy (degree) integer range -90 to 90 default 0.
     toVx()
     toVy()
     toXml (**kwargs)
         kwarg are for optional future option of added details in xml tree
class iradinapy.caseIradina.IonBeamIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about beam iradina
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
```

```
class iradinapy.caseIradina.IonDistributionIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrInListXyz
     strCfq()
         get index from '(i) blah blah'
class iradinapy.caseIradina.IonDoseIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatXyz
     float positive with default value -1 as None
class iradinapy.caseIradina.IonE0Ira(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 10e3 keV
class iradinapy.caseIradina.IonMIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     Ion mass (g/mol) float positive with default 1.
class iradinapy.caseIradina.IonVxIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     vector incidence x float positive with default 1.
class iradinapy.caseIradina.IonVyIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     vector incidence y float positive with default 0.
class iradinapy.caseIradina.IonVzIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     vector incidence z float positive with default 0
class iradinapy.caseIradina.IsotopeIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about IsotopeIra
     browseElement()
     checkValues (verbose=True)
     getActionsContextMenu()
          append action 'Append file projectile'
     isHidden (nameAttr)
          to know if attribute is currently displayed in treeView and other dialog widget
     on_attributesChange (verbose=False)
     setDefaultValues()
         not virtual, could be used
class iradinapy.caseIradina.ListOfMaterialIra(nameObject=")
     Bases: xyzpy.baseXyz.ListOfBaseXyz
     Important note: it is strongly recommended NOT to create one file with all materials that you know for
     all of your simulations! You should just include the materials you really need for the current simulation,
     because iradina will create 2.6 MByte scattering matrices for every possible combination of two elements
     in the target! So the memory usage increases with the square of the number of different elements! If your
     materials contain 92 different elements, iradina needs 22 GByte of memory in the 4-MSB version or 352
     GByte in the 6-MSB version
     getAllAttributesName()
\textbf{class} \  \, \texttt{iradinapy.caseIradina.ListOfTargetComponentsIra} \, (\textit{nameObject="}")
     Bases: xyzpy.baseXyz.ListOfBaseXyz
```

```
class iradinapy.caseIradina.MaterialIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about material of target iradina
     getElementCount()
     getElementsConc()
     getElementsDispEnergy()
     getElementsLattEnergy()
     getElementsM()
     getElementsReplEnergy()
     getElementsSurfEnergy()
     getElementsSymbol()
     getElementsZ()
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     normalize (aList)
class iradinapy.caseIradina.MaterialNameIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrXyz
     string no more 24 characters target.c: if(strlen(MaterialName)>=25){MaterialName[24]=";}
class iradinapy.caseIradina.MaxNoIonIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntPosXyz
     int positive with default 20000
class iradinapy.caseIradina.MinEnergyIra(value=None)
     Bases: xyzpy.intFloatListXyz.FloatPosXyz
     float positive with default 5.
class iradinapy.caseIradina.NormalizeOutputIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrInListXyz
     strCfg()
         get index from '(i) blah blah'
class iradinapy.caseIradina.Seed1Ira(value=None)
     Bases: xyzpy.intFloatListXyz.IntPosXyz
     int positive with default 123
class iradinapy.caseIradina.Seed2Ira(value=None)
     Bases: xyzpy.intFloatListXyz.IntPosXyz
     int positive with default 456
class iradinapy.caseIradina.SeedIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntPosXyz
     python random.seed(a=None) to initialize internal state of the random number generator as None or no
     argument seeds from current time or from an operating system specific randomness source if available
class iradinapy.caseIradina.SimulationIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about simulation iradina
     isHidden(nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
```

```
class iradinapy.caseIradina.SimulationTypeIra(value=None)
    Bases: xyzpy.intFloatListXyz.StrInListXyz
     strCfq()
         get index from '(i) blah blah'
class iradinapy.caseIradina.StatusUpdateIntervalIra(value=None)
    Bases: xyzpy.intFloatListXyz.IntPosXyz
    int positive with default 10000
class iradinapy.caseIradina.StorageIntervalIra(value=None)
    Bases: xyzpy.intFloatListXyz.IntPosXyz
     int positive with default 2000
class iradinapy.caseIradina.StorePathLimitIra(value=None)
     Bases: xyzpy.intFloatListXyz.IntSupEq1Xyz
    int positive with default 50
class iradinapy.caseIradina.StorePathLimitRecoilsIra(value=None)
    Bases: xyzpy.intFloatListXyz.IntXyz
    int positive with default -1 as None
class iradinapy.caseIradina.StragglingModelIra(value=None)
    Bases: xyzpy.intFloatListXyz.StrInListXyz
     strCfq()
         get index from '(i) blah blah'
class iradinapy.caseIradina.StructureIra
    Bases: xyzpy.baseXyz._XyzConstrainBase
    general informations about target iradina
     getCellSizeX()
     getMultiLayerMaterial (indice)
    get_cell_count_x()
     isHidden (nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
class iradinapy.caseIradina.TargetComponentIra
    Bases: xyzpy.baseXyz._XyzConstrainBase
    general informations about TargetComponentIra
    browseElement()
    checkValues (verbose=True)
    getActionsContextMenu()
         append action 'Append file projectile'
     isHidden(nameAttr)
         to know if attribute is currently displayed in treeView and other dialog widget
     on_attributesChange (verbose=False)
     setDefaultValues()
         not virtual, could be used
class iradinapy.caseIradina.TargetIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about target iradina
```

iradinapy.coloringlra module

simple tagging as '<color>' for simple coloring log messages on terminal(s) window or unix or ios using backend colorama. Using '<color>' because EZ human readable, So '<color>' are not supposed existing in log message. "{}".format() is not choosen because "{}" are present in log messages of contents of python dict (as JSON) etc.

Usage:

>> import iradinapy.coloringIra as COLS

Example:

>> log("this is in <green>color green<reset>, OK is in blue: <blue>OK?")

```
class iradinapy.coloringIra.ColoringStream
    Bases: object
```

write my stream class only write and flush are used for the streaming https://docs.python.org/2/library/logging.handlers.html https://stackoverflow.com/questions/31999627/storing-logger-messages-in-a-string

```
flush()
  write (astr)

iradinapy.coloringIra.cleanColors (msg)
  clean the message of color tags '<red>...

iradinapy.coloringIra.indent (msg, nb, car='')
  indent nb car (spaces) multi lines message except first one

iradinapy.coloringIra.log (msg)
  elementary log stdout for debug if _verbose

iradinapy.coloringIra.replace (msg, tags)

iradinapy.coloringIra.toColor (msg)
```

automatically clean the message of color tags '<red> ... if the terminal output stdout is redirected by user if not, replace tags with ansi color codes

```
iradinapy.coloringIra.toColor_AnsiToWin32 (msg)
    for test debug no wrapping
```

iradinapy.configIra module

```
This file is the main API for config configparser for iradinaGUI
class iradinapy.configIra.ConfigManager(runner)
     Bases: object
     Manages the read/write of config files of iradinaGUI, and merges if useful.
     file iradinaGUI_user.cfg
     file iradinaGUI_default.cfg
     assertUserDefaultFiles()
         if inexisting, create config files user and default. relative to workdir
     checkFileExist (filename)
         filename as name relative to workdir
     getDefaultConfig()
         new instance default config
     getMainConfig()
         main as merged config of default plus overrides of user new instance main config
     getRealPath (name)
     getUserConfig()
         new instance user config
     getWorkdir()
     setMainConfig(cfg)
         set a user main config as global, is user choice
iradinapy.configIra.getCurrentMode()
iradinapy.configIra.getExistingModes()
iradinapy.configIra.getMainConfig()
iradinapy.configIra.isHidden(item, nameAttr=None, modeName=None)
     avoid Components[*]. Density because finantch pattern [seq] matches any character in seq. use Compo-
     nents*.Density instead
iradinapy.configIra.setCurrentMode(modeName)
iradinapy.controlSimulationIra module
class iradinapy.controlSimulationIra.ControlSimulationIra
     Bases: xyzpy.baseXyz._XyzConstrainBase
     general informations about
     setDefaultValues()
         not virtual, could be used
class iradinapy.controlSimulationIra.UserModeIra(value=None)
     Bases: xyzpy.intFloatListXyz.StrInListXyz
```

iradinapy.controllerlra module

```
\label{eq:controllerIra.join} iradinapy.controllerIra.join (*$\nu$) as os.path.join but set antislash as slash, even for windows, keep windows 'c:' iradinapy.controllerIra.launchFromSalomePyConsole()
```

iradinapy.dateTime module

```
This file contains DateTime and DeltaTime class
Usage:
>> import dateTime as DATT
>> ini = DATT.DateTime("now")
>> # some stuff
>> fin = DATT.DateTime("now")
>> duration = DATT.DeltaTime(ini, fin)
class iradinapy.dateTime.DateTime(when=None)
     Bases: object
     assume storing a date and hour, and conversions
     Usage:
    >> import dateTime as DATT
    >> now = DATT.DateTime("now")
    >> print("now is %s" % now)
    FORMAT_DATEHOUR_CONFIG = '%Y%m%d_%H%M%S'
    FORMAT_DATE_CONFIG = '%Y%m%d'
     FORMAT_FILE = '%Y%m%d_%H%M%S'
     FORMAT_HOUR_CONFIG = '%H%M%S'
    FORMAT_HUMAN = '\$Y-\$m-\$d \$H:\$M:\$S'
    FORMAT_PACKAGE = '%Y-%m-%d %H:%M'
    FORMAT_XML = '%Y/%m/%d %Hh%Mm%Ss'
    MSG_UNDEFINED = 'UndefinedTime'
     addSeconds (secs)
         add seconds at time
     getSecondsToNow()
     getValue()
     isOk()
         return True if ok
     localTime()
     raiseIfKo()
```

raise an exception with message why if not ok, else return self. This trick is to write usage

```
Usage:
          >> aTimeOk = aTime.raiseIfKo() # raise Exception if KO
          >> doSomethingWithaTimeOk(aTimeOk) # here i am sure that is OK
     setValue (time)
          choice as not deep copying if mutables value
     toSeconds()
     toStrDateConfig()
     toStrDateHourConfig()
     toStrFile()
          use self.FORMAT_FILE, sortable, 2018-05-07... as '20180507_235958'
     toStrHourConfig()
     toStrHuman()
          use self.FORMAT_HUMAN
     toStrPackage()
     toStrXml()
class iradinapy.dateTime.DeltaTime(t1=None, t2=None)
     Bases: object
     assume storing a duration, delta between two DateTime, and conversions
     Usage:
     >> import dateTime as DATT
     >> t1 = DATT.DateTime("now")
     >> time.sleep(3)
     >> t2 = DATT.DateTime("now")
     >> delta = DATT.DeltaTime(t1, t2)
     >> print("delta time is %s" % delta)
     MSG_UNDEFINED = 'UndefinedDeltaTime'
     getT1(t)
     getT2(t)
     getValue()
          idem toSeconds()
     isOk()
         return True if ok
     raiseIfKo()
          raise an exception with message why if not ok, else return self. This trick is to write usage
         Usage:
         >> aDeltaTimeOk = adeltaTime.raiseIfKo() # raise Exception if KO
         >> doSomethingWithaDeltaTimeOk(aDeltaTimeOk) # here i am sure that is OK
     setT1(t)
     setT2(t)
```

```
toMinutes()
     toSeconds()
     toStrHms()
          all unities, hours and minutes and seconds as '2h34m56s'
     toStrHuman()
          automatic best unity, hours or minutes or seconds
iradinapy.dateTime.date_to_datetime(date)
     From a string date as pyconf config.VARS.datehour 'YYYYMMDD_HHMMSS' returns [year, month, day,
     hour, minutes, seconds]
          Parameters date – (str) The date in format YYYYMMDD_HHMMSS
          Returns (tuple) as (str,str,str,str,str,str) The same date and time in separate variables.
iradinapy.dateTime.fromDateHourConfig(datehour)
     datehour as pyconf config.VARS.datehour 'YYYYMMDD_HHMMSS'. Returns datetime.datetime
iradinapy.dateTime.fromTimeStamp(val)
     Returns datetime.datetime
iradinapy.dateTime.getWeekDayNow()
     Returns monday as 0, tuesday as 1 etc.
iradinapy.dateTime.parse_date(date)
     Transform as pyconf config.VARS.datehour 'YYYYMMDD_HHMMSS' to 'YYYY-MM-DD hh:mm:ss'.
          Parameters date – (str) The date to transform
          Returns (str) The date in the new format
iradinapy.dateTime.sleep(seconds)
     as time.sleep(seconds)
iradinapy.dateTime.timedelta_total_seconds(timedelta)
     Replace total_seconds from datetime module in order to be compatible with old python versions
          Parameters timedelta – (datetime.timedelta) The delta between two dates
          Returns (float) The number of seconds corresponding to timedelta.
iradinapy.iradinaFilePatterns module
all file patterns of supposedly (sometimes) created by iradinaGui replaces '@xxx@' as file.in autotools
iradinapy.iradinaFilePatterns.execReplaces(aStr, replaces=[])
     append standart useful replaces to users replaces
iradinapy.iradinaFilePatterns.filterColumsNamesForUranie (header)
     assume inexisting header array with empty strings "lenght of #COLUMN_NAMES fill incomplete array
iradinapy.iradinaFilePatterns.filterHeaderDat(line)
     extract (key, value) from line from file .dat uranie/salome/paraview
iradinapy.iradinaFilePatterns.getBaseFiles()
     list of useful mandatory files in uranie etude directory
iradinapy.iradinaFilePatterns.getDefaultDivide(nb)
     Canvas. Divide by default
iradinapy.iradinaFilePatterns.getFilePatterns(name, replaces=[])
iradinapy.iradinaFilePatterns.getFixedUsefulDirs(rootDir)
     unconditionaly fixed useful
```

```
iradinapy.iradinaFilePatterns.getHeaderContentsFileDat(filenameIni)
     header description of .dat uranie/salome/paraview returns a dict with keys NAME,DATE... message if
     inexisting file
iradinapy.iradinaFilePatterns.qetHeaderFileDat(filename, Verbose=False)
    header description of .dat uranie/salome/paraview returns a dict with keys #NAME, #DATE...
iradinapy.iradinaFilePatterns.getIgnoreFilesForCpack(rootDir,
                                                                                  Ver-
                                                                         useful,
                                                                bose=False)
    list of usefless directories in uranie etude directory, not packaging saved
iradinapy.iradinaFilePatterns.getOtherUsefulDirsForCpack(rootDir)
     unconditionaly fixed useful are not in other useful directories get user subdirs with CMakeLists.txt as useful
iradinapy.iradinaFilePatterns.getPatternKeys()
iradinapy.iradinaFilePatterns.getStdReplaces()
     standart useful replaces, @xxx@ as .in autotools
iradinapy.iradinaFilePatterns.getTypesOfVisualize()
iradinapy.iradinaFilePatterns.getUsefulDirsForCpack(rootDir)
iradinapy.iradinaFilePatterns.getUselessDirsForCpack(rootDir)
    list of useless directories in uranie etude directory, not packaging saved
iradinapy.iradinaFilePatterns.getUselessFilesForCpack(rootDir, useful)
    list of usefless files in uranie etude directory, not packaging saved
iradinapy.iradinaFilePatterns.getVisualizeMethod(aName)
iradinapy.iradinaFilePatterns.isPresentCMakeLists(aDir)
     test if CMakeLists.txt exists in directory
iradinapy.iradinaFilePatterns.isUranieColumn(name)
iradinapy.iradinaFilePatterns.listOnlyDirs(rootDir)
     return list of expanded directories names in directory
iradinapy.iradinaFilePatterns.removeDuplicates(*args)
iradinapy.iradinaFilePatterns.visualize_TDS_graphic(FILEDAT, ATTS, EXPR=[],
                                                              DIVIDE=None,
                                                                              FILS=[],
                                                              OPTS=[], Verbose=True)
     ATTS = (("x1:x2"), ("x3:x5")) for example 2 Draws
iradinapy.iradinaFilePatterns.visualize_TDS_univariate(FILEDAT,
                                                                                 ATTS,
                                                                  EXPR=[],
                                                                                   DI-
                                                                  VIDE=None, FILS=[],
                                                                  OPTS=[],
                                                                                  Ver-
                                                                  bose=True)
iradinapy.iradinaGui module
```

This file is the main API file for iradinaGUI

```
Warning: NO '__main__ ' call allowed,
Use '../iradinaGUI' (in parent directory)
```

Usage: see file ../iradinaGUI

```
class iradinapy.iradinaGui.ArgumentParserNoExit (prog=None, usage=None, descrip-
                                                                 tion=None, epilog=None, par-
                                                                 ents=[], formatter_class=<class</pre>
                                                                 'argparse.HelpFormatter'>,
                                                                prefix_chars='-',
                                                                                           from-
                                                                file_prefix_chars=None,
                                                                argument_default=None,
                                                                 conflict handler='error',
                                                                 add help=True,
                                                                                             al-
                                                                low_abbrev=True)
     Bases: argparse.ArgumentParser
     change Exiting method as no exit
     exit (status=0, message=None)
     filter_existing_file (string)
     filter_float_positive(string)
     filter_int_positive(string)
     filter list(string)
          parser filter from string 'xx,yy,zz,...' returns list (if not error with python exec(value=[xx,yy,zz,...]))
     filter_list_float (string)
          parser filter from string 'xx,yy,zz,...' returns list (if not error with python exec(value=[xx,yy,zz,...]))
     filter_list_int(string)
          parser filter from string 'xx,yy,zz,...' returns list (if not error with python exec(value=[xx,yy,zz,...]))
     filter_logLevel (aStr)
     filter_range (string)
          parser filter from string 'vmin,vmax' returns list (if not error with python exec(value=[xx,yy]))
     filter_square (string)
     filter_workdir(string)
     getLogLevels()
     getLogLevelsStr()
class iradinapy.iradinaGui.Ira(logger)
     Bases: object
     The main class that stores all the commands of iradinaGui (usually known as 'runner' argument in Command
     classes)
     assumeAsList (strOrList)
     execute_cli (cli_arguments)
          select first argument as a command in directory 'commands', and launch on arguments
              Parameters cli_arguments - (str or list) The iradinaGUI CLI arguments (as sys.argv)
     getAnswer (msg)
          question and user answer (in console) if confirm mode and not batch mode.
          returns 'YES' or 'NO' if confirm mode and not batch mode
          returns 'YES' if batch mode
     getBatchMode()
     getColoredVersion()
          get colored iradinaGui version message
```

```
getConfig()
     getConfirmMode()
         assimiled as integer incremented on _idCommandHandlers
     getLogger()
     getOptions()
     get_help()
         get general help colored string
    parseArguments (arguments)
    print_help()
         prints iradinaGui general help
     runGUI()
         a main window for iradinaGUI
     setConfirmMode(value)
     show doc()
         show iradinaGui general documentation
iradinapy.iradinaGui.assumeAsList(strOrList)
     return a list as sys.argv if string
iradinapy.iradinaGui.cmdsdir = '/volatile2/wambeke/TULEAP_MATIX/MATIX_26-CO7/SOURCES/IRA
     if DBG.isDeveloper(): workdirdefault =
                                            os.path.realpath(os.path.join(rootdir,
                                                                                      "IRADI-
         NAGUI_WORKDIR"))
     else: workdirdefault = os.path.expandvars(os.path.join("$HOME", "IRADINAGUI_WORKDIR"))
iradinapy.iradinaGui.getVersion()
     get version number as string
iradinapy.iradinaGui.launchIra(command)
     launch iradinaGUI as subprocess.Popen command as string ('iradinaGUI -help' for example) used for
     unittest, or else...
         Returns RCO.ReturnCode
iradinapy.iradinaSettings module
class iradinapy.iradinaSettings.IradinaSettings
     Bases: settingspy.settings.Settings (page 72)
     may be future link to Qt QSettings for future do not forget window environment variables are NOT case
     sensitive user have to write environment variables as syntax ${...}
     policy: names setting variables beginning with "_" contains environment variables reference
iradinapy.iradinaSettings.checkAll()
     used as singleton
iradinapy.iradinaSettings.checkEnvVar(val)
     used as singleton
iradinapy.iradinaSettings.getExpandedVar(name)
     used as singleton
iradinapy.iradinaSettings.getIradinaSysMacrosDir()
iradinapy.iradinaSettings.getSettings()
     used as singleton
```

```
iradinapy.iradinaSettings.getVar(name)
    used as singleton

iradinapy.iradinaSettings.setEnvVar(envVar, value)

iradinapy.iradinaSettings.setEnvVarByDefault(envVar, valueDefault)
```

iradinapy.logginglra module

iradinaGui logger. using logging package

Define one logger with one handler on stdout and one handler on file for production. Define another one logger for unittest.

see: http://sametmax.com/ecrire-des-logs-en-python

Define two LoggerIra instances in iradinaGui, no more need.

- loggerDefault as production/development logger
- _loggerUnittest as unittest logger

see use of handlers of _loggerDefault for log console and log files xml, txt

console handler:

- info(): no format
- error() warning() trace() debug() etc. :

formatted indented on multi lines messages using handlers

file handlers:

- info() error() warning() trace() debug() etc. :formatted indented on multi lines messages using handlers

WARNING:

log step and log trace are present on stdout console or log file following level handlers settings

```
class iradinapy.loggingIra.DefaultFormatter(fmt=None, datefmt=None, style='%')
    Bases: logging.Formatter
```

format (record)

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

setColorLevelname (levelname)

set color implies color special characters and tabulate levelname length of string

```
class iradinapy.loggingIra.FileTxtFormatter(fmt=None, datefmt=None, style='%')
    Bases: logging.Formatter
```

```
format (record)
```

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

```
class iradinapy.loggingIra.FileXmlFormatter (fmt=None, datefmt=None, style='%')
    Bases: logging.Formatter
    format (record)
```

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

```
there is exception information, it is formatted using formatException() and appended to the message.
class iradinapy.loggingIra.LoggerIra(name, level=20)
     Bases: logging.Logger
     Inherited class logging.Logger for logger iradinaGui
     add a level STEP as log.step(msg)
     add a level TRACE as log.trace(msg)
     below log.info(msg)
     above log.debug(msg)
     to assume message step inside files xml 'command's internal traces'
     to assume store long log asci in files txt outside files xml
     see: /usr/lib64/python2.7/logging/__init__.py etc.
      close()
           final stuff for logger, done at end iradinaGui flushed and closed xml files have to be not overri-
           den/appended
      closeFileHandlerForCommand(cmdInstance)
      getMainCommandHandler()
           returns handler for colored stdout console/terminal for human user eye iradinaGUI outputs
      initLinkForCommand(cmdParent, cmdNew)
      logStep(step)
           current logger.info() step as 'header ... etc ... '
      logStep_begin (header, step=")
           initialize for main handler (tty as stdout) a one line message for steps (... of compilation for example)
           as no return line message with logger.info() level
           example:
           'header ... first temporary step message ...'
           'header ... etc ...' (on same line)
           'header ... OK' (on same line)
```

logStep_end(step, tab=None)

last logger.info() step as 'header ... OK' or 'header ... KO'

setFileHandlerForCommand(cmdParent, cmdInstance)

add file handler to logger to set log files for a iradinaGui command. when command is known from pyconf/config instance

Example:

log files names for command prepare

with micro commands clean/source/patch

- ~/LOGS/20180510_140606_prepare_lenovo.xml
- ~/LOGS/OUT/20180510_140606_prepare_lenovo.txt
- ~/LOGS/micro_20180510_140607_clean_lenovo.xml
- ${\sim}/LOGS/OUT/micro_20180510_140607_clean_lenovo.txt$

etc.

setLevelMainHandler(level)

```
step (msg, *args, **kwargs)
```

Log 'msg % args' with severity '_STEP'.

testNoReturn()

test when message ending '...' and level info then no return mode

```
trace (msg, *args, **kwargs)
```

Log 'msg % args' with severity 'TRACE'.

xx_isEnabledFor (level)

Is this logger enabled for level 'level'? currently not modified from logging.Logger class, here only for call log debug.

class iradinapy.loggingIra.StreamHandlerIra(stream=None)

 $Bases: \verb|logging.StreamHandler||$

A handler class which writes logging records, appropriately formatted, to a stream. Note that this class does not close the stream, as sys.stdout or sys.stderr may be used.

from logging. Stream Handler class, modified for 'no return' mode line if '...' at end of record message

emit (record)

Emit a record.

If a formatter is specified, it is used to format the record. The record is then written to the stream with a trailing newline. If exception information is present, it is formatted using traceback.print_exception and appended to the stream. If the stream has an 'encoding' attribute, it is used to determine how to do the output to the stream.

isLastRecordHaveNoReturn()

to memorize if last info record is 'no return' mode (as ending '...') avoid define inherited __init__

isNeedFirstReturn(record)

'no return' mode valid only if 2 consecutives info messages if not, needs insert return line BEFORE (warning, debug, or other) current record message

class iradinapy.loggingIra.UnittestFormatter (fmt=None, datefmt=None, style='%')

Bases: logging.Formatter

format (record)

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The

message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

```
class iradinapy.loggingIra.UnittestStream
     Bases: object
     write my stream class only write and flush are used for the streaming
     https://docs.python.org/2/library/logging.handlers.html
     https://stackoverflow.com/questions/31999627/storing-logger-messages-in-a-string
     flush()
     getLogs()
     getLogsAndClear()
     write(astr)
          final method called when message is logged
class iradinapy.loggingIra.XmlHandler(capacity)
     Bases: logging.handlers.BufferingHandler
     log outputs in memory as BufferingHandler. Write ElementTree in file and flush are done once when method
     close is called, to generate xml file.
     atts = {
           "fileName": xml file name of micro command
           "command": cmd, # 'compile' or 'prepare' etc.
           "passed": res, # 'O' or '1'
           "launchedCommand": fullcmd, # 'compile TOTO -etc'
     see: https://docs.python.org/2/library/logging.handlers.html
      close()
          prepare ElementTree from existing logs and write xml file
          warning: avoid iradinaGUI logging message in logger close phase
      createLogField()
          prepare formatted string from self.buffer LogRecord for xml 'Log' node using handler formatter
     createLogFieldFromScrath()
          prepare formatted string from self.buffer LogRecord for xml 'Log' node local format
      set config(config)
          config is supposedly non existing, no overwrite accepted
      set_target_file (filename)
          filename is file name xml with path supposedly non existing, no overwrite accepted
iradinapy.loggingIra.filterLevel(aLevel)
     filter levels logging values from firsts characters levels. No case sensitive.
     example:
      'i' -> 'INFO'
```

'cRiT' -> 'CRITICAL'

```
iradinapy.loggingIra.getCurrentLogger()
     get one of _loggerDefault or _loggerUnittest as first created by getDefautLogger() or getUnittestLogger():
iradinapy.loggingIra.getDefaultLogger()
     official method to get the only one instance of Default Logger
iradinapy.loggingIra.getListOfStrLogRecord(listOfLogRecord)
     Returns one line string for logging LogRecord description
iradinapy.loggingIra.getLogger()
iradinapy.loggingIra.getMessage(self)
     modified from logging.__init__.LogRecord.getMessage, better message on format error Return the message
     for this LogRecord.
     Return the message for this LogRecord after merging any user-supplied arguments with the message.
iradinapy.loggingIra.getStrDirLogger(logger)
     Returns multi line string for logger description, with dir(logger). Used for debug
iradinapy.loggingIra.getStrHandler(handler)
     Returns one line string for handler description (as inexisting __repr__) to avoid create inherited classe(s)
     handler
iradinapy.loggingIra.getStrLogRecord(logRecord)
     Returns one line string for simple logging LogRecord description
iradinapy.loggingIra.getStrShort(msg)
     Returns short string for msg (as first caracters without line feed
iradinapy.loggingIra.getUnittestLogger()
     official method to get the only one instance of Unittest Logger
iradinapy.loggingIra.indent (msg, nb, car='')
     indent nb car (spaces) multi lines message except first one
iradinapy.loggingIra.indentUnittest(msg, prefix='|')
     indent multi lines message except first one with prefix. prefix default is designed for less spaces for size
     logs files and keep logs human eye readable
iradinapy.loggingIra.initLoggerAsDefault(logger, fmt=None, level=None)
     init logger as prefixed message and indented message if multi line exept info() outed 'as it' without any
     format. level could be modified during execution
iradinapy.loggingIra.initLoggerAsUnittest(logger, fmt=None, level=None)
     init logger as silent on stdout/stderr used for retrieve messages in memory for post execution unittest https:
     //docs.python.org/2/library/logging.handlers.html
iradinapy.loggingIra.log(msg, force=False)
     elementary log when no logging.Logger yet
iradinapy.loggingIra.testLogger_1 (logger)
     small test
iradinapy.loggingIra.testMain()
iradinapy.loggingIra.testNoReturn(logger)
     test when message ending '...' and level info then no return mode
```

iradinapy.mainWindowlra module

iradinapy.modellra module

```
class iradinapy.modelIra.ModelIra
     Bases: \verb"xyzpy.baseXyz._XyzConstrainBase"
     general instance to group all iradina data files in a directory input file iradina release?
     getActionsContextMenu()
     getEtudeWorkdirBrut (expanded=True)
     getEtudeWorkdirExpanded()
     getHistoryFile()
     isHidden (nameAttr)
          to know if attribute is currently displayed in treeView and other dialog widget
     setDefaultValues()
          not virtual, could be used
     setFromFileIra (fileName, verbose=False)
          override inherited method
     toStrIra()
          override inherited method
     userExpand()
          filename patterns ',??' warning not for '[' ']' as 'alist[]' no found way to quote meta-character
          https://docs.python.org/2/library/fnmatch.html
     userMode()
          change user mode
```

iradinapy.testPrerequisitesIra module

```
iradinapy.testPrerequisitesIra.TestImports()
    test of prerequisites import python for IradinaGui message for problem(s), aborting immediatly.
iradinapy.testPrerequisitesIra.TestIradinaFeatures()
    test of iradina CODE features
iradinapy.testPrerequisitesIra.error(message)
```

iradinapy.treeViewIra module

```
iradinapy.treeViewIra.verboseEvent = False
    cosmetic stuff for treeView Iradina
```

iradinapy.utilsIra module

utilities for iradinaGUI general useful simple methods all-in-one import iradinapy.utilsIra as UTS

```
Usage:
>> import iradinapy.utilsIra as UTS
>> UTS.ensure_path_exists(path)
iradinapy.utilsIra.Popen (command, shell=True, cwd=None, env=None, stdout=-1, stderr=-1,
                                logger=None)
     make subprocess.Popen(cmd), with call logger.trace and logger.error if problem as returncode != 0
iradinapy.utilsIra.addSpaces(idx, aStr)
iradinapy.utilsIra.black(msg)
iradinapy.utilsIra.blue (msg)
iradinapy.utilsIra.critical(msg)
iradinapy.utilsIra.cyan (msg)
iradinapy.utilsIra.deepcopy_list(input_list)
     Do a deep copy of a list
          Parameters input_list – (list) The list to copy
          Returns (list) The copy of the list
iradinapy.utilsIra.ensure file exists(aFile, aDefaultFile)
     Create a file if not existing, copying from default file
          Parameters
                • aFilepath – (str) The file to ensure existence
               • aDefaultFile – (str) The default file to copy if not existing
iradinapy.utilsIra.ensure_path_exists(path)
     Create a path if not existing
          Parameters path – (str) The path.
iradinapy.utilsIra.error(msg)
iradinapy.utilsIra.formatTuples(tuples)
     Format 'label = value' the tuples in a tabulated way.
          Parameters tuples – (list) The list of tuples to format
          Returns (str) The tabulated text. (as mutiples lines)
iradinapy.utilsIra.formatValue(label, value, suffix=")
     format 'label = value' with the info color
          Parameters
               • label – (int) the label to print.
               • value – (str) the value to print.
               • suffix – (str) the optionnal suffix to add at the end.
iradinapy.utilsIra.get_iradinaGUI_version(config)
iradinapy.utilsIra.get_tmp_filename(config, name)
iradinapy.utilsIra.green(msg)
```

```
iradinapy.utilsIra.header(msg)
iradinapy.utilsIra.info(msg)
iradinapy.utilsIra.label(msg)
iradinapy.utilsIra.magenta(msg)
iradinapy.utilsIra.merge_dicts(*dict_args)
     Given any number of dicts, shallow copy and merge into a new dict, precedence goes to key value pairs in
     latter dicts.
iradinapy.utilsIra.normal(msg)
iradinapy.utilsIra.red(msg)
iradinapy.utilsIra.remove_item_from_list(input_list, item)
     Remove all occurences of item from input_list
         Parameters input_list – (list) The list to modify
         Returns (list) The without any item
iradinapy.utilsIra.replace_in_file (file_in, str_in, str_out)
     Replace <str_in> by <str_out> in file <file_in>. save a file old version as file_in + '_old'
         Parameters
               • file_in – (str) The file name
               • str_in – (str) The string to search
               • str_out – (str) The string to replace.
iradinapy.utilsIra.reset (msg)
iradinapy.utilsIra.sleep(sec)
iradinapy.utilsIra.success(msg)
iradinapy.utilsIra.tabColor(*args)
     return tabulated colored string from args, assume true length of color tags as <OK> <info> etc. to correct
     alignment when tags are interpreted as (no-length-spacing) color for colorama use or else
iradinapy.utilsIra.warning(msg)
iradinapy.utilsIra.white (msg)
iradinapy.utilsIra.yellow(msg)
Module contents
4.1.2 configparserpy
configparserpy package
Subpackages
configparserpy.test package
Submodules
configparserpy.test.test_130_configParserUtils module
utilities for best use ConfigParser
```

```
see:
```

```
https://wiki.python.org/moin/ConfigParserExamples
https://docs.python.org/2/library/configparser.html
```

```
class configparserpy.test.test_130_configParserUtils.TestCase (methodName='runTest')
    Bases: unittest.case.TestCase
    Test the configParserUtils.py
    test_000()
    test_010()
    test_020()
    test_030()
    test_032()
    test_034()
    test_100()
    test_999()
```

Module contents

Submodules

configparserpy.configParserUtils module

```
utilities for best use ConfigParser
see: https://wiki.python.org/moin/ConfigParserExamples
class configparserpy.configParserUtils.UtSafeConfigParser(*args, **kwargs)
     Bases: configparser.SafeConfigParser
     SafeConfigParser with ExtendedInterpolation, and __repr__, and readDefaultAndUser to merge default and
     user config
     copy()
     isEmpty()
     readDefaultAndUser (aStrDefault, aStrUser)
          merge user overriding origin defaults
     readFromStr (aStr, merge=False)
          allow merge only explicitly
     toCatchAll (verbose=False)
          permits class attribute writings, (but raise on accentuation and avoid spaces in section names)
          cfg = UtSafeConfigParser()
          cfg.readFromStr(" | [General]
          reporter = tintin
          "")
          config = cfg.toCatchAll()
          print(config.General.reporter) # -> "tintin"
```

```
toDict (verbose=False)
     toDictTuple (verbose=False)
     toOrderedDict (verbose=False)
     writeToStr()
          allow merge only explicitly
configparserpy.configParserUtils.getConfigFromDefaultAndUserStr(aStrDefault,
                                                                                  aStrUser)
     simple create config from Default an overrrides from User as strings
configParserUtils.getConfigFromFile(aFile)
configparserpy.configParserUtils.getConfigFromStr(aStr)
     simple create config from contents as string
Module contents
4.1.3 settingspy
settingspy package
Submodules
settingspy.setStyleFactory module
settingspy.setStyleFactory.run()
settingspy.settings module
class settingspy.settings.Settings
     Bases: object
     may be future link to Qt QSettings for future do not forget window environment variables are NOT case
     sensitive user have to write environment variables as syntax ${...}
     policy: names setting variables beginning with "_" contains environment variables reference
     checkAll()
          expand var in settings return (ok, aDict) ok is False or True aDict is settings as aDict[key] = (value,
          interpretedValue)
     checkEnvVar (aStrWithEnvVar)
          check all env vars contained as syntax ${...} defined in environ
     getExpandedVar (name)
          returns expanded value of var name, or None if inexisting or problem in expand
     getRealPath (aPathWithEnvVar)
          resolve file path env variable as ${HOME}/toto etc... with os.path.expandvars interpretation of env
     getVar (name, Verbose=True)
          return NOT expanded value of var name
     setVar (name, value, Verbose=True)
          no control, user choice with checkAll()
settingspy.settings.getExpandedVar(name)
     used as singleton
```

```
settingspy.settings.getVar(name)
   used as singleton

settingspy.settings.getVar(name)
   used as singleton

settingspy.settings.setEnvVar(envVar, value)
   with message warning if change

settingspy.settings.setEnvVarByDefault(envVar, valueDefault)
   only for single environ variables: envVar='HOME' for ${HOME} or $HOME valueDefault have to be expanded (i.e. without '$')
```

Module contents

settingspy for salome matix

4.1.4 filewatcherpy

filewatcherpy package

Subpackages

filewatcherpy.test package

Submodules

filewatcherpy.test.test_340_fileWatcher module

```
class filewatcherpy.test.test_340_fileWatcher.TestCase (methodName='runTest')
    Bases: unittest.case.TestCase
    test_010()
    test_020()
    test_030()
    test_999()
```

Module contents

Submodules

filewatcherpy.fileWatcher module

```
filewatcherpy.fileWatcher.exampleLaunchStandalone()
filewatcherpy.fileWatcher.getFileWatcher()
    use it as singleton
filewatcherpy.fileWatcher.getRealPath(aPathWithEnvVar)
    resolve env variable as $HOME/toto etc... with expandvars
filewatcherpy.fileWatcher.getRealPath_obsolete(aPathWithEnvVar)
    resolve env variable as $HOME/toto etc... with subprocess shell interpretation of env var
```

Module contents

filewatcherpy for salome matix

CHAPTER

FIVE

RELEASE NOTES

5.1 Release notes

In construction.

PYTHON MODULE INDEX

```
C
configparserpy, 72
configparserpy.configParserUtils,71
configparserpy.test,71
configparserpy.test.test 130 configParserUtils,
f
filewatcherpy, 74
filewatcherpy.fileWatcher, 73
filewatcherpy.test, 73
filewatcherpy.test.test_340_fileWatcher,
iradinapy, 70
iradinapy.abcdExpression,45
iradinapy.analysisIra,45
iradinapy.caseIradina,49
iradinapy.colorama,43
iradinapy.colorama.ansi,40
iradinapy.colorama.ansitowin32,41
iradinapy.colorama.initialise, 42
iradinapy.colorama.win32,42
iradinapy.colorama.winterm, 43
iradinapy.coloringIra,55
iradinapy.configIra,56
iradinapy.controllerIra,57
iradinapy.controlSimulationIra,56
iradinapy.dateTime, 57
iradinapy.example, 45
iradinapy.example.essai_logging_1,43
iradinapy.example.essai_logging_2,44
iradinapy.iradinaFilePatterns, 59
iradinapy.iradinaGui, 60
iradinapy.iradinaSettings, 62
iradinapy.loggingIra,63
iradinapy.mainWindowIra,68
iradinapy.modelIra,68
iradinapy.testPrerequisitesIra,68
iradinapy.treeViewIra,68
iradinapy.utilsIra,69
settingspy, 73
settingspy.setStyleFactory,72
settingspy.settings,72
```

INDEX

A	В
addItem() (iradinapy.analysisIra.ListOfFileViewerXy	z BACK () (iradinapy.colorama.ansi.AnsiCursor
method), 47	method), 40
addItems() (iradi-	back() (iradinapy.colorama.winterm.WinTerm
napy.analysisIra.ListOfFileViewerXyz	method), 43
method), 47	BeamSpreadIra (class in iradinapy.caseIradina), 49
addItemsSlot() (iradi-	BLACK (iradinapy.colorama.ansi.AnsiBack attribute),
napy.analysisIra.ListOfFileViewerXyz	40
method), 47	BLACK (iradinapy.colorama.ansi.AnsiFore attribute),
addSeconds() (iradinapy.dateTime.DateTime	41
method), 57	BLACK (iradinapy.colorama.winterm.WinColor at-
addSpaces () (in module iradinapy.utilsIra), 69	tribute), 43
AnalysisIra (<i>class in iradinapy.analysisIra</i>), 45	black() (in module iradinapy.utilsIra), 69
ANSI_CSI_RE (iradi-	BLUE (iradinapy.colorama.ansi.AnsiBack attribute), 40
napy.colorama.ansitowin32.AnsiToWin32	BLUE (iradinapy.colorama.ansi.AnsiFore attribute), 41
attribute), 41	BLUE (iradinapy.colorama.winterm.WinColor at-
ANSI_OSC_RE (iradi-	tribute), 43
napy.colorama.ansitowin32.AnsiToWin32	blue() (in module iradinapy.utilsIra), 69
attribute), 41	BoolFalseIra (class in iradinapy.caseIradina), 49
AnsiBack (<i>class in iradinapy.colorama.ansi</i>), 40	BoolTrueIra (class in iradinapy.caseIradina), 49
AnsiCodes (class in iradinapy.colorama.ansi), 40	BRIGHT (iradinapy.colorama.ansi.AnsiStyle attribute),
AnsiCursor (class in iradinapy.colorama.ansi), 40	41
AnsiFore (class in iradinapy.colorama.ansi), 41	BRIGHT (iradinapy.colorama.winterm.WinStyle
AnsiStyle (class in iradinapy.colorama.ansi), 41	attribute), 43
AnsiToWin32 (class in iradi-	BRIGHT_BACKGROUND (iradi-
napy.colorama.ansitowin32), 41	napy.colorama.winterm.WinStyle attribute),
appendHistoryAction() (iradi-	43
napy.analysisIra.HistoryFileManagerXyz	browseElement() (iradi-
method), 47	napy.caseIradina.IsotopeIra method),
appendHistoryCopyOf() (iradi-	52
napy.analysisIra.HistoryFileManagerXyz	browseElement() (iradi-
method), 47	napy. case Iradina. Target Component Ira
appendHistoryFileManager() (iradi-	method), 54
napy.analysisIra.AnalysisIra method),	browseViewerDialog() (iradi-
45	napy.analysisIra.ListOfFileViewerXyz
ArgumentParserNoExit (class in iradi-	method), 47
napy.iradinaGui), 60	browseViewerExecOnApply() (iradi-
assertUserDefaultFiles() (iradi-	napy.analysisIra.ListOfFileViewerXyz
napy.configIra.ConfigManager method),	method), 47
56	С
assumeAsList() (in module iradinapy.iradinaGui),	O
62	call_win32() (iradi-
assumeAsList() (iradinapy.iradinaGui.Ira	napy.colorama.ansitowin32.AnsiToWin32
method), 61	method), 41
AttributeDataFrameIra (class in iradi-	CaseIra (class in iradinapy.caseIradina), 49
napy.analysisIra), 45	CellCountxIra (class in iradinapy.caseIradina), 49

CellCountyIra (class in iradinapy.caseIradina), 49	ControlSimulationIra (class in iradi-
CellCountzIra (class in iradinapy.caseIradina), 49	napy.controlSimulationIra), 56
CellDepthxIra (class in iradinapy.caseIradina), 49	convert_ansi() (iradi-
CellMultiLayerxIra (class in iradi-	napy.colorama.ansitowin32.AnsiToWin32
napy.caseIradina), 50 CellSizexIra (class in iradinapy.caseIradina), 50	method), 41 convert_osc() (iradi-
CellSizeyIra (class in iradinapy.caseIradina), 50	napy.colorama.ansitowin32.AnsiToWin32
CellSizezIra (class in iradinapy.caseIradina), 50	method), 42
checkAll() (in module iradinapy.iradinaSettings),	copy () (configparserpy.configParserUtils.UtSafeConfigParser
62	method), 71
checkAll() (settingspy.settings.Settings method), 72	copyAllFileInData() (iradi-
<pre>checkEnvVar() (in module iradi-</pre>	napy.analysis Ira. Data Manager Ira method),
napy.iradinaSettings), 62	46
checkEnvVar() (settingspy.settings.Settings	copyFileInData() (iradinapy.analysisIra.DataIra
method), 72	method), 46
checkFileExist() (iradi-	createDocLaunch() (iradi-
napy.configIra.ConfigManager method), 56	napy.analysisIra.AnalysisIra method), 45
checkValues() (iradinapy.caseIradina.IsotopeIra	createEditorData() (iradi-
method), 52	napy.analysisIra.ExpressionIra method),
checkValues() (iradi-	46
napy. case Iradina. Target Component Ira	createLogField() (iradi-
method), 54	napy.loggingIra.XmlHandler method),
<pre>cleanColors() (in module iradinapy.coloringIra),</pre>	66
55	createLogFieldFromScrath() (iradi-
clear_line() (in module iradi-	napy.loggingIra.XmlHandler method), 66
<pre>napy.colorama.ansi), 41 clear_screen() (in module iradi-</pre>	critical() (in module iradinapy.utilsIra), 69
napy.colorama.ansi), 41	cursor_adjust() (iradi-
clearHistory() (iradi-	napy.colorama.winterm.WinTerm method),
napy.analysisIra.HistoryFileManagerXyz	43
method), 47	CYAN (iradinapy.colorama.ansi.AnsiBack attribute), 40
close() (iradinapy.loggingIra.LoggerIra method),	CYAN (iradinapy.colorama.ansi.AnsiFore attribute), 41
64	CYAN (iradinapy.colorama.winterm.WinColor at-
$\verb close () (iradinapy.loggingIra.XmlHandler method),$	tribute), 43
66	cyan () (in module iradinapy.utilsIra), 69
closeFileHandlerForCommand() (iradi-	D
napy.loggingIra.LoggerIra method), 64	D
cmdsdir (in module iradinapy.iradinaGui), 62	DataInformationsIra (class in iradi-
code_to_chars() (in module iradi-	napy.analysisIra), 46
napy.colorama.ansi), 41 colorama text() (in module iradi-	DataIra (class in iradinapy.analysisIra), 46
colorama_text() (in module iradi- napy.colorama.initialise), 42	DataManagerIra (class in iradinapy.analysisIra), 46
ColoringStream (class in iradinapy.coloringIra), 55	date_to_datetime() (in module iradi- napy.dateTime), 59
CompositionFileTypeIra (class in iradi-	DateTime (class in iradinapy.dateTime), 57
napy.caseIradina), 50	deepcopy_list() (in module iradinapy.utilsIra),
ConcentrationIra (class in iradi-	69
napy.caseIradina), 50	DefaultFormatter (class in iradi-
ConfigManager (class in iradinapy.configIra), 56	napy.loggingIra), 63
configparserpy	deinit() (in module iradinapy.colorama.initialise),
module,72	42
configparserpy.configParserUtils	DeltaTime (class in iradinapy.dateTime), 58
module, 71	DensityIra (class in iradinapy.caseIradina), 50
configparserpy.test	DensityTargetComponentIra (class in iradi-
module, 71	napy.caseIradina), 51
$\begin{array}{c} \texttt{configparserpy.test.test_130_configPar} \\ \texttt{module}, 70 \end{array}$	৭৯৮৮ম প্রদক্ষেণাару.colorama.ansi.AnsiStyle attribute), 41

DisplayIntervalIra (class in iradi-	
napy.caseIradina), 51 DOWN () (iradinapy.colorama.ansi.AnsiCursor	napy.loggingIra), 63
	- -
method), 40	module, 74
drawDialogDataServer() (iradi-	
napy.analysisIra.DataIra method), 46	module,73
drawGraphic() (in module iradinapy.analysisIra),	
48	module,73
drawGraphic() (iradi-	filewatcherpy.test.test_340_fileWatcher
napy. analysis Ira. Attribute Data Frame Ira	module, 73
method), 45	FileXmlFormatter (class in iradi-
drawGraphic() (iradi-	napy.loggingIra), 64
napy.analysisIra.ExpressionIra method),	filter_existing_file() (iradi-
46	napy.iradinaGui.ArgumentParserNoExit
drawUnivariate() (in module iradi-	method), 61
napy.analysisIra), 48	filter_float_positive() (iradi-
drawUnivariate() (iradi-	napy.iradinaGui.ArgumentParserNoExit
napy.analysisIra.AttributeDataFrameIra	method), 61
method), 45	filter_int_positive() (iradi-
drawUnivariate() (iradi-	napy.iradinaGui.ArgumentParserNoExit
napy.analysisIra.ExpressionIra method),	method), 61
46	filter_list() (iradi-
10	napy.iradinaGui.ArgumentParserNoExit
E	method), 61
ElementCountIra (class in iradinapy.caseIradina),	
51	napy.iradinaGui.ArgumentParserNoExit
ElementReplEnergyIra (class in iradi-	method), 61
napy.caseIradina), 51	filter_list_int() (iradi-
emit() (iradinapy.loggingIra.StreamHandlerIra	napy.iradinaGui.ArgumentParserNoExit
method), 65	method), 61
ensure_file_exists() (in module iradi-	filter_logLevel() (iradi-
napy.utilsIra), 69	napy.iradinaGui.ArgumentParserNoExit
ensure_path_exists() (in module iradi-	method), 61
napy.utilsIra), 69	filter_range() (iradi-
erase_line() (iradi-	napy.iradina Gui. Argument Parser No Exit
napy.colorama.winterm.WinTerm method),	method), 61
43	filter_square() (iradi-
erase_screen() (iradi-	napy.iradinaGui.ArgumentParserNoExit
napy.colorama.winterm.WinTerm method),	method), 61
43	filter_workdir() (iradi-
error() (in module iradinapy.testPrerequisitesIra),	napy.iradinaGui.ArgumentParserNoExit
68	method), 61
error() (in module iradinapy.utilsIra), 69	filterColumsNamesForUranie() (in module
exampleLaunchStandalone() (in module file-	iradinapy.iradinaFilePatterns), 59
watcherpy.fileWatcher), 73	filterHeaderDat() (in module iradi-
execReplaces() (in module iradi-	napy.iradinaFilePatterns), 59
napy.iradinaFilePatterns), 59	filterLevel() (in module iradinapy.loggingIra),
execute_cli() (iradinapy.iradinaGui.Ira method),	66
61	FlightLengthTypeIra (class in iradi-
exit() (iradinapy.iradinaGui.ArgumentParserNoExit	napy.caseIradina), 51
method), 61	FloatListIra (class in iradinapy.caseIradina), 51
ExpressionIra (class in iradinapy.analysisIra), 46	flush() (iradinapy.coloringIra.ColoringStream
extract_params() (iradi-	method), 55
napy.colorama.ansitowin32.AnsiToWin32	flush() (iradinapy.loggingIra.UnittestStream
method), 42	method), 66
,,	fn_heterogenous_random_multiple_materials()
F	(in module iradinapy.analysisIra), 48
FileIra (class in iradinapy.analysisIra), 47	fn_homogenous_one_material() (in module
L LLCLLA (CIASS III II AAIIAP Y.AIIAI YSISII A J, 🕇 /	

iradinapy.analysisIra), 48	method), 42
${\tt fn_multiLayer_multiple_materials()} \ \ ({\it in}$	
module iradinapy.analysisIra), 48	napy.analysisIra.AnalysisIra method),
fore() (iradinapy.colorama.winterm.WinTerm	45
method), 43	getActionsContextMenu() (iradinatter napy.analysisIra.AttributeDataFrameIra
format () (iradinapy.example.essai_logging_2.MyForm method), 44	method), 45
format() (iradinapy.loggingIra.DefaultFormatter	
method), 63	napy.analysisIra.DataIra method), 46
format () (iradinapy.loggingIra.FileTxtFormatter method), 63	getActionsContextMenu() (iradi- napy.analysisIra.DataManagerIra method),
format() (iradinapy.loggingIra.FileXmlFormatter	46
method), 64	getActionsContextMenu() (iradi-
format() (iradinapy.loggingIra.UnittestFormatter	napy.analysisIra.ExpressionIra method),
method), 65	46
FORMAT_DATE_CONFIG (iradi-	· ·
napy.dateTime.DateTime attribute), 57	napy.analysisIra.ListOfFileViewerXyz
FORMAT_DATEHOUR_CONFIG (iradi-	method), 47
napy.dateTime.DateTime attribute), 57	getActionsContextMenu() (iradi-
FORMAT_FILE (iradinapy.dateTime.DateTime attribute), 57	napy.caseIradina.DensityIra method), 50
FORMAT_HOUR_CONFIG (iradi-	getActionsContextMenu() (iradi-
napy.dateTime.DateTime attribute), 57	napy.caseIradina.DensityTargetComponentIra
FORMAT_HUMAN (iradinapy.dateTime.DateTime at-	method), 51
tribute), 57	<pre>getActionsContextMenu() (iradi-</pre>
FORMAT_PACKAGE (iradinapy.dateTime.DateTime at-	napy.caseIradina.IsotopeIra method),
tribute), 57	52
FORMAT_XML (iradinapy.dateTime.DateTime at-	getActionsContextMenu() (iradi-
tribute), 57	napy.caseIradina.TargetComponentIra
formatTuples() (in module iradinapy.utilsIra), 69 formatValue() (in module iradinapy.utilsIra), 69	<pre>method), 54 getActionsContextMenu() (iradi-</pre>
FORWARD () (iradinapy.colorama.ansi.AnsiCursor	napy.modelIra.ModelIra method), 68
method), 40	getAllAttributesName() (iradi-
fromDateHourConfig() (in module iradi-	napy.analysisIra.DataIra method), 46
napy.dateTime), 59	<pre>getAllAttributesName() (iradi-</pre>
<pre>fromTimeStamp() (in module iradinapy.dateTime), 59</pre>	napy.analysisIra.ListOfAttributeIra method), 47
FunctionIra (class in iradinapy.analysisIra), 47	<pre>getAllAttributesName() (iradi-</pre>
	napy. analysis Ira. List Of Expression Ira
G	method), 47
get_attrs() (iradi-	getAllAttributesName() (iradi-
napy.colorama.winterm.WinTerm method), 43	napy.caseIradina.ListOfMaterialIra method), 52
<pre>get_cell_count_x() (iradi-</pre>	<pre>getAllAttributesNameInParents() (irad-</pre>
napy.caseIradina.StructureIra method), 54	inapy.analysisIra.ExpressionIra method), 46
<pre>get_help() (iradinapy.iradinaGui.Ira method), 62</pre>	$\verb"getAnswer()" (\textit{iradinapy.iradinaGui.Ira method}), 61$
<pre>get_iradinaGUI_version() (in module iradi-</pre>	getAttributeName() (iradi-
napy.utilsIra), 69	napy.analysisIra.AttributeDataFrameIra
<pre>get_position() (iradi-</pre>	<pre>method), 45 getAttributeName() (iradi-</pre>
napy.colorama.winterm.WinTerm method),	napy.analysisIra.ExpressionIra method),
qet_tmp_filename() (in module iradi-	46
<pre>get_tmp_filename() (in module iradi- napy.utilsIra), 69</pre>	getBaseFiles() (in module iradi-
<pre>get_value_random_multiple_materials()</pre>	napy.iradinaFilePatterns), 59
(in module iradinapy.analysisIra), 49	getBatchMode() (iradinapy.iradinaGui.Ira
get_win32_calls() (iradi-	method), 61
napy.colorama.ansitowin32.AnsiToWin32	getCalculatedValue() (iradi-

		method),	53
	50		getElementsReplEnergy() (iradi-
getCel	lSizeX()	(iradi-	napy.caseIradina.MaterialIra method),
		method),	53
	54	G . I	getElementsSurfEnergy() (iradi-
getCol	oredVersion() (iradinapy.iradir method), 61	naGui.Ira	napy.caseIradina.MaterialIra method), 53
getCom	pleteFileName()	(iradi-	<pre>getElementsSymbol() (iradi-</pre>
	napy.analysisIra.HistoryFileManagemethod), 47	erXyz	napy.caseIradina.MaterialIra method), 53
getCon	fig() (iradinapy.iradinaGui.Ira me	ethod), 62	<pre>getElementsZ() (iradi-</pre>
getCon	<pre>figFromDefaultAndUserStr(module configparserpy.configPar</pre>		napy.caseIradina.MaterialIra method), 53
	72		getEtudeWorkdir() (iradi-
getCon	<pre>figFromFile() (in module parserpy.configParserUtils), 72</pre>	config-	napy.analysisIra.DataInformationsIra method), 46
getCon	figFromStr() (in module	config-	<pre>getEtudeWorkdirBrut() (iradi-</pre>
getCon	<pre>parserpy.configParserUtils), 72 firmMode() (iradinapy.iradinal)</pre>	naGui.Ira	napy.analysisIra.DataInformationsIra method), 46
	method), 62		<pre>getEtudeWorkdirBrut() (iradi-</pre>
getCou	nt()	(iradi-	napy.modelIra.ModelIra method), 68
	napy.caseIradina.CellMultiLayerxIr	\dot{a}	<pre>getEtudeWorkdirExpanded() (iradi-</pre>
	method), 50		napy.modelIra.ModelIra method), 68
getCur	rentLogger() (in module napy.loggingIra), 67	iradi-	<pre>getExistingModes() (in module iradi- napy.configIra), 56</pre>
getCur	rentMode() (in module napy.configIra), 56	iradi-	<pre>getExpandedVar() (in module iradi- napy.iradinaSettings), 62</pre>
getCur	rentRowColumn() (in modula napy.analysisIra), 48	e iradi-	<pre>getExpandedVar() (in module settingspy.settings), 72</pre>
getDef	aultConfig() napy.configIra.ConfigManager	(iradi- method),	<pre>getExpandedVar() (settingspy.settings.Settings method), 72</pre>
	56		<pre>getExpressions() (iradi-</pre>
getDef	<pre>aultDivide() (in module napy.iradinaFilePatterns), 59</pre>	iradi-	napy.analysisIra.ListOfExpressionIra method), 47
getDef	aultLogger() (in module napy.loggingIra), 67	iradi-	<pre>getExpressionsInParent() (iradi- napy.analysisIra.AttributeDataFrameIra</pre>
aetDef	aultRowColumn() (in modul	e iradi-	method), 45
2	napy.analysisIra), 48		<pre>getExpressionsInParent() (iradi-</pre>
getDir	ectory() napy.analysisIra.ListOfFileViewerX	(iradi-	napy.analysisIra.ExpressionIra method), 46
	method), 48	,	getFileHash() (iradi-
getDir	ectory() napy.analysisIra.ListOfUserFileIra	(iradi- method),	napy.analysisIra.HistoryFileManagerXyz method), 47
	48	,,	<pre>getFileInParent() (iradi-</pre>
getEle:	mentCount () napy.caseIradina.MaterialIra	(iradi- method),	napy.analysisIra.AttributeDataFrameIra method), 46
	53	,,,	<pre>getFileInParent() (iradi-</pre>
getEle:	mentsConc() napy.caseIradina.MaterialIra	(iradi- method),	napy.analysisIra.ExpressionIra method), 46
cet Flo	53 mentsDispEnergy()	(iradi-	getFilePatterns() (in module iradinapy.iradinaFilePatterns), 59
уссыте.	napy.caseIradina.MaterialIra 53	method),	getFileWatcher() (in module file-
~~+ ₽1 ~·	mentsLattEnergy()	(iradi-	<pre>watcherpy.fileWatcher), 73 getFixedUsefulDirs() (in module iradi-</pre>
Aerrie.		method),	napy.iradinaFilePatterns), 59
	53	ciiouj,	getHeaderContentsFileDat() (in module
getEle:	mentsM()	(iradi-	iradinapy.iradinaFilePatterns), 59
,	napy.caseIradina.MaterialIra	method),	getHeaderFileDat() (in module iradi-

napy.iradinaFilePatterns), 60	48
<pre>getHistoryFile() (iradinapy.modelIra.ModelIra</pre>	<pre>getNoLocal() (iradi-</pre>
method), 68	napy.analysisIra.ListOfFileViewerXyz
getId() (iradinapy.iradinaGui.Ira method), 62	method), 48
<pre>getIdentFile() (iradi-</pre>	getNoLocal() (iradi-
napy.analysisIra.HistoryFileManagerXyz method), 47	napy.analysisIra.MacroManagerIra method), 48
<pre>getIgnoreFilesForCpack() (in module iradi- napy.iradinaFilePatterns), 60</pre>	getOptions() (iradinapy.iradinaGui.Ira method), 62
<pre>getIndiceFromChar() (in module iradi-</pre>	<pre>getOtherUsefulDirsForCpack() (in module</pre>
napy.abcdExpression), 45	iradinapy.iradinaFilePatterns), 60
<pre>getIndiceFromChar() (iradi-</pre>	<pre>getPatternKeys() (in module iradi-</pre>
napy.caseIradina.CellMultiLayerxIra	napy.iradinaFilePatterns), 60
method), 50	<pre>getRandomConcMaterial() (in module iradi-</pre>
<pre>getIradinaSysMacrosDir() (in module iradi-</pre>	napy.analysisIra), 48
napy.iradinaSettings), 62	<pre>getRealPath() (in module file-</pre>
<pre>getListOfStrLogRecord() (in module iradi-</pre>	watcherpy.fileWatcher), 73
napy.loggingIra), 67	getRealPath() (iradi-
<pre>getLogger() (in module iradinapy.loggingIra), 67</pre>	napy.configIra.ConfigManager method),
<pre>getLogger() (iradinapy.iradinaGui.Ira method), 62</pre>	56
<pre>getLogLevels() (iradi-</pre>	<pre>getRealPath() (settingspy.settings.Settings</pre>
napy.iradinaGui.ArgumentParserNoExit	method), 72
method), 61	<pre>getRealPath_obsolete() (in module file-</pre>
<pre>getLogLevelsStr() (iradi-</pre>	watcherpy.fileWatcher), 73
napy.iradinaGui.ArgumentParserNoExit	getSecondsToNow() (iradi-
method), 61	napy.dateTime.DateTime method), 57
<pre>getLogs() (iradinapy.loggingIra.UnittestStream</pre>	getSettings() (in module iradi-
method), 66	napy.iradinaSettings), 62
<pre>getLogsAndClear() (iradi-</pre>	getSettings() (in module settingspy.settings), 72
napy.loggingIra.UnittestStream method),	<pre>getStdReplaces() (in module iradi-</pre>
66	napy.iradinaFilePatterns), 60
<pre>getMainCommandHandler() (iradi-</pre>	<pre>getStrDirLogger() (in module iradi-</pre>
napy.loggingIra.LoggerIra method), 64	napy.loggingIra), 67
<pre>getMainConfig() (in module iradinapy.configIra),</pre>	<pre>getStrHandler() (in module iradi-</pre>
56	napy.loggingIra), 67
<pre>getMainConfig() (iradi-</pre>	<pre>getStrLogRecord() (in module iradi-</pre>
napy.configIra.ConfigManager method),	napy.loggingIra), 67
56	<pre>getStrShort() (in module iradinapy.loggingIra),</pre>
<pre>getMessage() (in module iradinapy.loggingIra), 67</pre>	67
getMultiLayerMaterial() (iradi-	<pre>getT1() (iradinapy.dateTime.DeltaTime method), 58</pre>
napy.caseIradina.CellMultiLayerxIra	getT2() (iradinapy.dateTime.DeltaTime method), 58
method), 50	<pre>getTargetDirectory() (iradi-</pre>
<pre>getMultiLayerMaterial() (iradi-</pre>	napy.analysisIra.ListOfFileViewerXyz
napy.caseIradina.StructureIra method),	method), 48
54	<pre>getTargetDirectory() (iradi-</pre>
<pre>getMyLogger() (in module iradi-</pre>	napy.analysisIra.ListOfUserFileIra method),
napy.example.essai_logging_1), 44	48
getMyLogger() (in module iradi-	<pre>getTypesOfVisualize() (in module iradi-</pre>
napy.example.essai_logging_2), 44	napy.iradinaFilePatterns), 60
getName() (iradinapy.analysisIra.ExpressionIra	<pre>getUnittestLogger() (in module iradi-</pre>
method), 46	napy.loggingIra), 67
<pre>getNameExpanded() (iradi-</pre>	<pre>getUsefulDirsForCpack() (in module iradi-</pre>
napy.analysisIra.DataIra method), 46	napy.iradinaFilePatterns), 60
<pre>getNamesExpanded() (iradi-</pre>	<pre>getUselessDirsForCpack() (in module iradi-</pre>
napy.analysisIra.ListOfFileViewerXyz	napy.iradinaFilePatterns), 60
method), 48	getUselessFilesForCpack() (in module iradi-
<pre>getNamesExpanded() (iradi-</pre>	napy.iradinaFilePatterns), 60
napy.analysisIra.MacroManagerIra method),	

napy.configIra.ConfigManager method),	<pre>initMyLogger() (in module iradi-</pre>
56	<pre>napy.example.essai_logging_1), 44</pre>
<pre>getUserConfigValue() (iradi-</pre>	<pre>initMyLogger() (in module iradi-</pre>
napy. case Iradina. Density Target Component Irachterian Transfer Transfe	napy.example.essai_logging_2), 44
method), 51	IonAngleIra (class in iradinapy.caseIradina), 51
<pre>getValue() (iradinapy.dateTime.DateTime method),</pre>	IonBeamIra (class in iradinapy.caseIradina), 51
57	IonDistributionIra (class in iradi-
<pre>getValue() (iradinapy.dateTime.DeltaTime</pre>	napy.caseIradina), 51
method), 58	IonDoseIra (class in iradinapy.caseIradina), 52
getVar() (in module iradinapy.iradinaSettings), 62	IonEOIra (class in iradinapy.caseIradina), 52
getVar() (in module settingspy.settings), 73	IonMIra (class in iradinapy.caseIradina), 52
getVar() (settingspy.settings.Settings method), 72	IonVxIra (class in iradinapy.caseIradina), 52
getVersion() (in module iradinapy.iradinaGui), 62	IonVyIra (class in iradinapy.caseIradina), 52
getVersion() (iradi-	IonVzIra (class in iradinapy.caseIradina), 52
napy. analysis Ira. Data Informations Ira	Ira (class in iradinapy.iradinaGui), 61
method), 46	iradinapy
<pre>getVisualizeMethod() (in module iradi-</pre>	module, 70
napy.iradinaFilePatterns), 60	iradinapy.abcdExpression
<pre>getWeekDayNow() (in module iradinapy.dateTime),</pre>	module, 45
59	iradinapy.analysisIra
getWikipediaValue() (iradi-	module, 45
napy.caseIradina.DensityTargetComponentIra	
method), 51	module, 49
getWorkdir() (iradinapy.configIra.ConfigManager	iradinapy.colorama
method), 56	module, 43
gitCommit() (iradinapy.analysisIra.AnalysisIra method), 45	iradinapy.colorama.ansi module,40
gitkLaunch() (iradinapy.analysisIra.AnalysisIra	iradinapy.colorama.ansitowin32
method), 45	module, 41
GREEN (iradinapy.colorama.ansi.AnsiBack attribute), 40	<pre>iradinapy.colorama.initialise module, 42</pre>
GREEN (iradinapy.colorama.ansi.AnsiFore attribute),	iradinapy.colorama.win32
41	module, 42
GREEN (iradinapy.colorama.winterm.WinColor attribute), 43	<pre>iradinapy.colorama.winterm module, 43</pre>
green () (in module iradinapy.utilsIra), 69	iradinapy.coloringIra
GREY (iradinapy.colorama.winterm.WinColor at-	module, 55
tribute), 43	iradinapy.configIra
П	module, 56
H header() (in module iradinapy.utilsIra), 69	<pre>iradinapy.controllerIra module,57</pre>
HistoryFileManagerXyz (class in iradi-	<pre>iradinapy.controlSimulationIra</pre>
napy.analysisIra), 47	module, 56
Ī	iradinapy.dateTime
I	module, 57
indent() (in module iradinapy.coloringIra), 55	iradinapy.example
indent() (in module iradinapy.loggingIra), 67	module, 45
indentUnittest() (in module iradi-	iradinapy.example.essai_logging_1
napy.loggingIra), 67	module, 43
info() (in module iradinapy.utilsIra), 70	<pre>iradinapy.example.essai_logging_2 module, 44</pre>
init () (in module iradinapy.colorama.initialise), 42	iradinapy.iradinaFilePatterns
initLinkForCommand() (iradi-	module, 59
napy.loggingIra.LoggerIra method), 64	iradinapy.iradinaGui
initLoggerAsDefault() (in module iradi-	module, 60
napy.loggingIra), 67	iradinapy.iradinaSettings
<pre>initLoggerAsUnittest() (in module iradi- napy.loggingIra), 67</pre>	module, 62
mpy.mggmgmu), 07	iradinapy.loggingIra

module, 63	isOk() (iradinapy.dateTime.DeltaTime method), 58
<pre>iradinapy.mainWindowIra module, 68</pre>	IsotopeIra (class in iradinapy.caseIradina), 52 isPresentCMakeLists() (in module iradi-
iradinapy.modelIra	napy.iradinaFilePatterns), 60
module, 68	isUranieColumn() (in module iradi-
iradinapy.testPrerequisitesIra	napy.iradinaFilePatterns), 60
module, 68	isValidExpression() (iradi-
<pre>iradinapy.treeViewIra</pre>	napy.analysisIra.ExpressionIra method).
module, 68	46
iradinapy.utilsIra	1
module, 69	J
IradinaSettings (class in iradi-	join () (in module iradinapy.analysisIra), 49
napy.iradinaSettings), 62	join () (in module iradinapy.controllerIra), 57
is_a_tty() (in module iradi-	1
napy.colorama.ansitowin32), 42	L
is_stream_closed() (in module iradi-	label() (in module iradinapy.utilsIra), 70
napy.colorama.ansitowin32), 42	launchFromSalomePyConsole() (in module
isEmpty() (config-	iradinapy.controllerIra), 57
parserpy.configParserUtils.UtSafeConfigParse	rlaunchIra() (in module iradinapy.iradinaGui), 62
method), 71 isHidden() (in module iradinapy.configIra), 56	LibraryIra (class in iradinapy.analysisIra), 47
isHidden() (iradinapy.analysisIra.AnalysisIra	LIGHTBLACK_EX (iradinapy.colorama.ansi.AnsiBack
method), 45	attribute), 40
isHidden() (iradi-	LIGHTBLACK_EX (iradinapy.colorama.ansi.AnsiFore
napy.analysisIra.DataInformationsIra	attribute), 41
method), 46	LIGHTBLUE_EX (iradinapy.colorama.ansi.AnsiBack attribute), 40
isHidden() (iradi-	LIGHTBLUE_EX (iradinapy.colorama.ansi.AnsiFore
napy.analysisIra.HistoryFileManagerXyz	attribute), 41
method), 47	LIGHTCYAN_EX (iradinapy.colorama.ansi.AnsiBack
isHidden() (iradinapy.caseIradina.CaseIra	attribute), 40
method), 49	LIGHTCYAN_EX (iradinapy.colorama.ansi.AnsiFore
isHidden() (iradinapy.caseIradina.IonBeamIra	attribute), 41
method), 51	LIGHTGREEN_EX (iradinapy.colorama.ansi.AnsiBack
isHidden() (iradinapy.caseIradina.IsotopeIra	attribute), 40
method), 52	LIGHTGREEN_EX (iradinapy.colorama.ansi.AnsiFore
isHidden() (iradinapy.caseIradina.MaterialIra method), 53	attribute), 41
isHidden() (iradinapy.caseIradina.SimulationIra	LIGHTMAGENTA_EX (iradi-
method), 53	napy.colorama.ansi.AnsiBack attribute).
isHidden() (iradinapy.caseIradina.StructureIra	40
method), 54	LIGHTMAGENTA_EX (iradi- napy.colorama.ansi.AnsiFore attribute).
isHidden() (iradi-	41
napy.caseIradina.TargetComponentIra	LIGHTRED_EX (iradinapy.colorama.ansi.AnsiBack
method), 54	attribute), 40
isHidden() (iradinapy.caseIradina.TargetIra	LIGHTRED_EX (iradinapy.colorama.ansi.AnsiFore at-
method), 54	tribute), 41
isHidden() (iradinapy.modelIra.ModelIra method),	LIGHTWHITE_EX (iradinapy.colorama.ansi.AnsiBack
68	attribute), 40
isLastRecordHaveNoReturn() (iradi-	LIGHTWHITE_EX (iradinapy.colorama.ansi.AnsiFore
napy.loggingIra.StreamHandlerIra method),	attribute), 41
65	LIGHTYELLOW_EX (iradi-
isNameUnique() (iradi- napy.analysisIra.ExpressionIra method),	napy.colorama.ansi.AnsiBack attribute).
46	40
isNeedFirstReturn() (iradi-	LIGHTYELLOW_EX (iradi-
napy.loggingIra.StreamHandlerIra method),	napy.colorama.ansi.AnsiFore attribute). 41
65	ListOfAttributeIra (class in iradi-
isOk() (iradinapy.dateTime.DateTime method), 57	napy.analysisIra), 47

ListOfExpressionIra (class in iradinapy.analysisIra), 47	<pre>filewatcherpy.test.test_340_fileWatcher, 73</pre>
ListOfFileViewerXyz (class in iradi-	iradinapy,70
napy.analysisIra), 47	iradinapy.abcdExpression,45
ListOfFunctionIra (class in iradi-	iradinapy.analysisIra,45
napy.analysisIra), 48	iradinapy.caseIradina,49
ListOfLibraryIra (class in iradi-	iradinapy.caserradina,49
napy.analysisIra), 48	iradinapy.colorama.ansi,40
ListOfMacroIra (class in iradinapy.analysisIra),	
48	iradinapy.colorama.ansitowin32,41
	iradinapy.colorama.initialise,42
ListOfMaterialIra (class in iradi-	iradinapy.colorama.win32,42
napy.caseIradina), 52	iradinapy.colorama.winterm,43
ListOfTargetComponentsIra (class in iradi-	iradinapy.coloringIra,55
napy.caseIradina), 52	iradinapy.configIra,56
ListOfUserFileIra (class in iradi-	iradinapy.controllerIra,57
napy.analysisIra), 48	iradinapy.controlSimulationIra,56
listOnlyDirs() (in module iradi-	iradinapy.dateTime,57
napy.iradinaFilePatterns), 60	iradinapy.example,45
localTime() (iradinapy.dateTime.DateTime method), 57	<pre>iradinapy.example.essai_logging_1, 43</pre>
log() (in module iradinapy.coloringIra), 55	iradinapy.example.essai_logging_2,
log() (in module iradinapy.loggingIra), 67	44
LoggerIra (class in iradinapy.loggingIra), 64	iradinapy.iradinaFilePatterns,59
logStep() (iradinapy.loggingIra.LoggerIra	iradinapy.iradinaGui,60
method), 64	iradinapy.iradinaSettings,62
logStep_begin() (iradi-	iradinapy.loggingIra, 63
napy.loggingIra.LoggerIra method), 64	iradinapy.mainWindowIra,68
logStep_end() (iradinapy.loggingIra.LoggerIra	iradinapy.modelIra,68
method), 64	iradinapy.modellia,00
method), 04	iradinapy.treeViewIra,68
M	iradinapy.treeviewira,00
	settingspy, 73
MacroIra (class in iradinapy.analysisIra), 48	settingspy,73 settingspy.setStyleFactory,72
MacroManagerIra (class in iradinapy.analysisIra),	settingspy.settings,72
48	MSG_UNDEFINED (iradinapy.dateTime.DateTime at-
MAGENTA (iradinapy.colorama.ansi.AnsiBack at-	
tribute), 40	tribute), 57
MAGENTA (iradinapy.colorama.ansi.AnsiFore at-	MSG_UNDEFINED (iradinapy.dateTime.DeltaTime at-
tribute), 41	tribute), 58
MAGENTA (iradinapy.colorama.winterm.WinColor at-	MyFormatter (class in iradi-
tribute), 43	napy.example.essai_logging_2), 44
magenta() (in module iradinapy.utilsIra), 70	N
MaterialIra (class in iradinapy.caseIradina), 52	
MaterialNameIra (class in iradinapy.caseIradina), 53	NORMAL (<i>iradinapy.colorama.ansi.AnsiStyle attribute</i>), 41
MaxNoIonIra (class in iradinapy.caseIradina), 53	NORMAL (iradinapy.colorama.winterm.WinStyle
merge_dicts() (in module iradinapy.utilsIra), 70	attribute), 43
MinEnergyIra (class in iradinapy.caseIradina), 53	normal() (in module iradinapy.utilsIra), 70
ModelIra (class in iradinapy.modelIra), 68	normalize() (in module iradinapy.analysisIra), 49
module	normalize() (iradinapy.caseIradina.DensityIra
confignarserpy, 72	method), 50
<pre>configparserpy.configParserUtils, 71</pre>	normalize() (iradinapy.caseIradina.MaterialIra method), 53
configparserpy.test,71	NormalizeOutputIra (class in iradi-
configparserpy.test.test_130_config	
70	S T. V
filewatcherpy, 74	0
filewatcherpy.fileWatcher,73	on_attributesChange() (iradi-
filewatcherpy.test,73	napy.caseIradina.IsotopeIra method),

52 on_attributesChange() (iradi-	<pre>reset_all()</pre>
napy.caseIradina.TargetComponentIra	method), 42
method), 54	reset_all() (iradi-
Р	napy.colorama.winterm.WinTerm method), 43
packageLaunch() (iradi-	run () (in module settingspy.setStyleFactory), 72
napy.analysisIra.AnalysisIra method),	runGUI () (iradinapy.iradinaGui.Ira method), 62
45	runPythonCode() (iradi- napy.analysisIra.AnalysisIra method),
<pre>parse_date() (in module iradinapy.dateTime), 59 parseArguments() (iradinapy.iradinaGui.Ira</pre>	45
method), 62	
Popen () (in module iradinapy.utilsIra), 69	S
POS () (iradinapy.colorama.ansi.AnsiCursor method),	searchURANIEMethod() (iradi-
40	napy.analysisIra.AnalysisIra method),
<pre>postTreatments() (iradi-</pre>	45
napy.analysisIra.AnalysisIra method),	Seedlira (class in iradinapy.caseIradina), 53
45	Seed2Ira (class in iradinapy.caseIradina), 53
<pre>print_help() (iradinapy.iradinaGui.Ira method), 62</pre>	SeedIra (class in iradinapy.caseIradina), 53 set_attrs() (iradi-
printROOTContext() (iradi-	napy.colorama.winterm.WinTerm method),
napy.analysisIra.AnalysisIra method),	43
45	<pre>set_config() (iradinapy.loggingIra.XmlHandler method), 66</pre>
R	set_console() (iradi-
<pre>raiseIfKo() (iradinapy.dateTime.DateTime method), 57</pre>	napy.colorama.winterm.WinTerm method), 43
raiseIfKo() (iradinapy.dateTime.DeltaTime	set_cursor_position() (iradi-
method), 58	napy.colorama.winterm.WinTerm method),
readDefaultAndUser() (config-	43
parserpy.configParserUtils.UtSafeConfigParser	rset_target_file() (iradi-
method), 71	napy.loggingIra.XmlHandler method), 66
readFromStr() (config-	rset_title() (in module iradinapy.colorama.ansi),
method), 71	41
RED (iradinapy.colorama.ansi.AnsiBack attribute), 40	set_title() (iradi-
RED (iradinapy.colorama.ansi.AnsiFore attribute), 41	napy.colorama.winterm.WinTerm method),
RED (iradinapy.colorama.winterm.WinColor attribute),	43
43	setAttributes() (iradinapy.analysisIra.DataIra
red() (in module iradinapy.utilsIra), 70	method), 46
reinit() (in module iradinapy.colorama.initialise),	setCalculatedValue() (iradi-
42	napy.caseIradina.DensityIra method), 50
remove_item_from_list() (in module iradi-	setColorLevelname() (iradi-
<pre>napy.utilsIra), 70 removeDuplicates() (in module iradi-</pre>	napy.loggingIra.DefaultFormatter method),
napy.iradinaFilePatterns), 60	63
replace() (in module iradinapy.coloringIra), 55	setConfirmMode() (iradinapy.iradinaGui.Ira
replace_in_file() (in module iradi-	method), 62
napy.utilsIra), 70	SetConsoleTextAttribute() (in module iradi-
${\tt RESET} \ (iradinapy.colorama.ansi.AnsiBack \ attribute),$	napy.colorama.win32), 42
40	setCurrentMode() (in module iradi-
RESET (iradinapy.colorama.ansi.AnsiFore attribute),	<pre>napy.configIra), 56 setDefaultDensityUser() (iradi-</pre>
reset() (in module iradinapy.utilsIra), 70	napy.caseIradina.DensityTargetComponentIra
RESET_ALL (iradinapy.colorama.ansi.AnsiStyle at-	method), 51
tribute), 41	setDefaultDensityWiki() (iradi-
reset_all() (in module iradi-	napy.caseIradina.DensityTargetComponentIra
nany colorama initialise) 42	method), 51

setDefaultValues()	(iradi-	SimulationTypeIra (class in iradi-
napy.analysisIra.AnalysisIra	method),	napy.caseIradina), 53
45		<pre>sleep() (in module iradinapy.dateTime), 59</pre>
setDefaultValues()	(iradi-	sleep() (in module iradinapy.utilsIra), 70
napy.analysisIra.DataInformatic	onsIra	StatusUpdateIntervalIra (class in iradi-
method), 46		napy.caseIradina), 54
setDefaultValues()	(iradi-	step () (iradinapy.loggingIra.LoggerIra method), 65
napy.analysisIra.HistoryFileMar	nagerXyz	StorageIntervalIra (class in iradi-
method), 47		napy.caseIradina), 54
setDefaultValues()	(iradi-	StorePathLimitIra (class in iradi-
napy.analysisIra.MacroManager	,	napy.caseIradina), 54
48	,,	StorePathLimitRecoilsIra (class in iradi-
setDefaultValues()	(iradi-	napy.caseIradina), 54
napy.caseIradina.CaseIra metho		StragglingModelIra (class in iradi-
setDefaultValues()	(iradi-	napy.caseIradina), 54
napy.caseIradina.IsotopeIra	method),	
52		method), 49
setDefaultValues()	(iradi-	
napy.caseIradina.TargetCompon	ientIra	method), 50
method), 54		strCfg() (iradinapy.caseIradina.DensityIra
setDefaultValues()	(iradi-	method), 50
- ·	rolSimulationI	rastrCfg()(iradinapy.caseIradina.FlightLengthTypeIra
method), 56		method), 51
setDefaultValues()	(iradi-	strCfg() (iradinapy.caseIradina.IonDistributionIra
napy.modelIra.ModelIra method	7), 68	method), 52
setEnvVar() (in module iradinapy.iraa	linaSettings),	strCfg() (iradinapy.caseIradina.NormalizeOutputIra method), 53
setEnvVar() (in module settingspy.sett	tings), 73	strCfg() (iradinapy.caseIradina.SimulationTypeIra
setEnvVarByDefault() (in mod	-	method), 54
napy.iradinaSettings), 63		strCfg()(iradinapy.caseIradina.StragglingModelIra
- ·	odule set-	method), 54
tingspy.settings), 73		strCfg()(iradinapy.caseIradina.TypeCompositionIra
setFileHandlerForCommand()	(iradi-	method), 55
napy.loggingIra.LoggerIra meth		StreamHandlerIra (class in iradi-
setFromFileIra() (iradinapy.model		napy.loggingIra), 65
method), 68	ma.modelma	StreamWrapper (class in iradi-
setLevelMainHandler()	(iradi-	napy.colorama.ansitowin32), 42
napy.loggingIra.LoggerIra meth	,	StructureIra (class in iradinapy.caseIradina), 54
<pre>setMainConfig()</pre>	(iradi- method),	style() (iradinapy.colorama.winterm.WinTerm method), 43
56	meinoa),	success () (in module iradinapy.utilsIra), 70
setT1() (iradinapy.dateTime.DeltaTime	mathad) 50	success () (in module tradinapy.uttistra), 10
set11() (tradinapy.dateTime.DeltaTime set12() (iradinapy.dateTime.DeltaTime		Т
= :		
Settings (class in settingspy.settings), T	12	tabColor() (in module iradinapy.utilsIra), 70
settingspy		TargetComponentIra (class in iradi-
module, 73		napy.caseIradina), 54
settingspy.setStyleFactory		TargetIra (class in iradinapy.caseIradina), 54
module, 72		test_000() (config-
settingspy.settings module,72		parserpy.test.test_130_configParserUtils.TestCase method), 71
<pre>setValue() (iradinapy.dateTime.DateT</pre>	ime method),	test_010() (config-
58		parserpy.test.test_130_configParserUtils.TestCase
setVar() (settingspy.settings.Settings m	ethod), 72	method), 71
should_wrap()	(iradi-	test_010() (filewatcherpy.test.test_340_fileWatcher.TestCas
napy.colorama.ansitowin32.Ans	iToWin32	method), 73
method), 42		test_020() (config-
show_doc() (iradinapy.iradinaGui.Ira i	method), 62	parserpy.test.test_130_configParserUtils.TestCase
SimulationIra (class in iradinapy.cas		method), 71

```
test_020() (filewatcherpy.test.test_340_fileWatcher.TestCase
                                                            napy.abcdExpression), 45
        method), 73
                                                                                                (iradi-
                                                    toEval0123()
test_030()
                                                             napy.caseIradina.CellMultiLayerxIra
                                           (config-
        parserpy.test.test_130_configParserUtils.TestCase
                                                             method), 50
        method), 71
                                                    toEvalAbcd()
                                                                          (in
                                                                                   module
                                                                                                 iradi-
test_030() (filewatcherpy.test.test_340_fileWatcher.TestCase
                                                            napy.abcdExpression), 45
                                                                                                (iradi-
        method), 73
                                                    toEvalAbcd()
test_032()
                                           (config-
                                                             napy.caseIradina.CellMultiLayerxIra
        parserpy.test.test_130_configParserUtils.TestCase
                                                             method), 50
        method), 71
                                                    toEvalAbcdForTooltip() (in module iradi-
test_034()
                                           (config-
                                                             napy.abcdExpression), 45
        parserpy.test.test_130_configParserUtils.TestCatseFileCompositionIn()
                                                                                  (in module
                                                                                                iradi-
        method), 71
                                                             napy.analysisIra), 49
                                           (config- toFileConfigurationIn() (in module iradi-
test_100()
        parserpy.test.test_130_configParserUtils.TestCase
                                                             napy.analysisIra), 49
        method), 71
                                                    toFileIra()
                                                                       (iradinapy.analysisIra.AnalysisIra
test_999()
                                          (config-
                                                             method), 45
        parserpy.test.test_130_configParserUtils.TestCats@FileMaterialIn()
                                                                                      module
                                                                                                 iradi-
                                                                                (in
                                                             napy.analysisIra), 49
test_999() (filewatcherpy.test.test_340_fileWatcher.TextClaseLeStructureIn()
                                                                                      module
                                                                                                 iradi-
        method), 73
                                                             napy.analysisIra), 49
TestCase
                   (class
                                           config-
                                                   toMinutes()
                                                                          (iradinapy.dateTime.DeltaTime
                                 in
        parserpy.test.test_130_configParserUtils),
                                                             method), 59
        71
                                                    toOrderedDict()
                                                                                               (config-
                    (class
                                                             parser py. config Parser Utils. Ut Safe Config Parser
TestCase
                                              file-
                                   in
        watcherpy.test.test_340_fileWatcher), 73
                                                             method), 72
TestImports()
                       (in
                                module
                                                    toSeconds()
                                                                          (iradinapy.dateTime.DateTime
                                            iradi-
        napy.testPrerequisitesIra), 68
                                                             method), 58
TestIradinaFeatures()
                              (in module
                                                                          (iradinapy.dateTime.DeltaTime
                                            iradi-
                                                    toSeconds()
        napy.testPrerequisitesIra), 68
                                                             method), 59
testLogger1()
                       (in
                                module
                                            iradi-
                                                    toStrDateConfig()
                                                                                                (iradi-
        napy.example.essai_logging_1), 44
                                                             napy.dateTime.DateTime method), 58
                                                                                                (iradi-
                                module
                                                    toStrDateHourConfig()
testLogger1()
                       (in
                                            iradi-
                                                             napy.dateTime.DateTime method), 58
        napy.example.essai_logging_2), 44
testLogger_1 () (in module iradinapy.loggingIra),
                                                                          (iradinapy.dateTime.DateTime
                                                    toStrFile()
        67
                                                             method), 58
testMain() (in module iradinapy.loggingIra), 67
                                                    toStrHms()
                                                                          (iradinapy.dateTime.DeltaTime
testNoReturn() (in module iradinapy.loggingIra),
                                                             method), 59
                                                    toStrHourConfig()
                                                                                                (iradi-
testNoReturn() (iradinapy.loggingIra.LoggerIra
                                                             napy.dateTime.DateTime method), 58
        method), 65
                                                                          (iradinapy.dateTime.DateTime
                                                    toStrHuman()
timedelta_total_seconds() (in module iradi-
                                                             method), 58
        napy.dateTime), 59
                                                                          (iradinapy.dateTime.DeltaTime
                                                    toStrHuman()
                                                             method), 59
toAbcd() (in module iradinapy.abcdExpression), 45
toAtomCm3()
                   (iradinapy.caseIradina.DensityIra
                                                    toStrIra() (iradinapy.modelIra.ModelIra method),
        method), 51
toCatchAll()
                                           (config-
                                                   toStrPackage()
                                                                          (iradinapy.dateTime.DateTime
        parserpy.configParserUtils.UtSafeConfigParser
                                                             method), 58
        method), 71
                                                    toStrXml() (iradinapy.dateTime.DateTime method),
toColor() (in module iradinapy.coloringIra), 55
                                                   toValue() (in module iradinapy.analysisIra), 49
toColor_AnsiToWin32()
                              (in module iradi-
        napy.coloringIra), 55
                                                    toVx() (iradinapy.caseIradina.IonAngleIra method),
toDict() (configparserpy.configParserUtils.UtSafeConfigParser 51
                                                    toVy() (iradinapy.caseIradina.IonAngleIra method),
        method), 71
toDictTuple()
                                                             51
                                           (config-
        parser py. config Parser Utils. Ut Safe Config Parser \verb+toXml()
                                                                   (iradinapy.caseIradina.CellDepthxIra
        method), 72
                                                             method), 50
toEval0123()
                      (in
                               module
                                            iradi- toXml() (iradinapy.caseIradina.DensityIra method),
```

```
51
                                                    write_and_convert()
                                                                                                (iradi-
                                                            napy.colorama.ansitowin32.AnsiToWin32
toXml()
                 (iradinapy.caseIradina.IonAngleIra
        method), 51
                                                            method), 42
trace() (iradinapy.loggingIra.LoggerIra method),
                                                    write_plain_text()
                                                                                                (iradi-
                                                            napy.colorama.ansitowin32.AnsiToWin32
TypeCompositionIra
                            (class
                                            iradi-
                                                            method), 42
                                      in
        napy.caseIradina), 55
                                                    writeToStr()
                                                                                              (config-
                                                            parser py. config Parser Utils. Ut Safe Config Parser
U
                                                            method), 72
UnittestFormatter
                           (class
                                      in
                                            iradi-
                                                    X
        napy.loggingIra), 65
                                                   XmlHandler (class in iradinapy.loggingIra), 66
UnittestStream (class in iradinapy.loggingIra),
                                                    xx_isEnabledFor()
                                                                                                (iradi-
UP() (iradinapy.colorama.ansi.AnsiCursor method),
                                                            napy.loggingIra.LoggerIra method), 65
        40
                                                    Y
                                           (iradi-
updateRootlogonLaunch()
        napy.analysisIra.AnalysisIra
                                         method),
                                                   YELLOW (iradinapy.colorama.ansi.AnsiBack attribute),
        45
userExpand()
                      (iradinapy.modelIra.ModelIra
                                                   YELLOW (iradinapy.colorama.ansi.AnsiFore attribute),
        method), 68
                                                             41
UserFileIra (class in iradinapy.analysisIra), 48
                                                    YELLOW (iradinapy.colorama.winterm.WinColor at-
userMode() (iradinapy.modelIra.ModelIra method),
                                                            tribute), 43
        68
                                                   yellow() (in module iradinapy.utilsIra), 70
UserModeIra
                      (class
                                            iradi-
        napy.controlSimulationIra), 56
UtSafeConfigParser
                            (class
                                           config-
        parserpy.configParserUtils), 71
V
verboseEvent (in module iradinapy.treeViewIra),
visualize_TDS_graphic() (in module iradi-
        napy.iradinaFilePatterns), 60
visualize_TDS_univariate()
                                          module
                                     (in
        iradinapy.iradinaFilePatterns), 60
W
warning () (in module iradinapy.utilsIra), 70
WHITE (iradinapy.colorama.ansi.AnsiBack attribute),
WHITE (iradinapy.colorama.ansi.AnsiFore attribute),
white() (in module iradinapy.utilsIra), 70
winapi_test()
                       (in
                               module
                                            iradi-
        napy.colorama.win32), 42
WinColor (class in iradinapy.colorama.winterm), 43
WinStyle (class in iradinapy.colorama.winterm), 43
WinTerm (class in iradinapy.colorama.winterm), 43
                               module
wrap_stream()
                       (in
                                            iradi-
        napy.colorama.initialise), 42
write()(iradinapy.colorama.ansitowin32.AnsiToWin32
        method), 42
write()(iradinapy.colorama.ansitowin32.StreamWrapper
        method), 42
              (iradinapy.coloringIra.ColoringStream
write()
        method), 55
                (iradinapy.loggingIra.UnittestStream
write()
```

method), 66