
pymcxray Documentation

Release 0.1.3

Hendrix Demers

May 23, 2019

Contents

1	pymcxdray	3
1.1	Features	3
1.2	Credits	3
2	Installation	5
2.1	Stable release	5
2.2	For developer	5
2.3	From sources	6
3	Usage	7
3.1	simulation_test_maps	8
3.2	Configuration file	9
3.3	Batch file	10
3.4	Simulation subclass	10
4	Contributing	17
4.1	Types of Contributions	17
4.2	Get Started!	18
4.3	Pull Request Guidelines	19
4.4	Tips	19
5	Credits	21
5.1	Development Lead	21
5.2	Contributors	21
6	History	23
6.1	0.1.2 (2017-06-26)	23
6.2	0.1.0 (2017-03-07)	23
7	ToDo	25
8	pymcxdray	29
8.1	pymcxdray package	29
9	Indices and tables	87
	Python Module Index	89

Contents:

Python scripts for using mcxray software

- Free software: Apache Software License 2.0
- Documentation: <https://pymcxray.readthedocs.io>.

1.1 Features

- TODO

1.2 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Stable release

Todo: Add pypi installation for this project.

Warning: Right now, the *Stable release* installation does not work.

To install pymcxray, run this command in your terminal:

```
$ pip install pymcxray
```

This is the preferred method to install pymcxray, as it will always install the most recent stable release.

If you don't have `pip` installed, this [Python installation guide](#) can guide you through the process.

2.2 For developer

Clone the public repository:

```
$ git clone git://github.com/drix00/pymcxray
```

Go in the project folder and install it with pip in developer mode:

```
$ cd pymcxray
$ pip install -e .
```

Note: The project use Git LFS for the test data file. Follow the information on [Git LFS](#) to get the test data when the repository is pull.

2.3 From sources

The sources for pymcxray can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/drix00/pymcxray
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/drix00/pymcxray/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

To use pymcrray in a project look in the *examples* folder.

To run the example, you need to specify the task you want to do.

```
$python simulation_test_maps.py generate
```

Note: On windows use the `py` command and not `python` if you have more than one python version installed.

Currently these tasks are available:

- generate
- check
- read
- analyze
- scheduled_read

At first, the script may look complex, but starting with a previous script it is easy to create a new script for you simulation.

Each script has two parts:

- **a class that subclass *mcxray._Simulations* were**
 - the simulation parameters are specified
 - results needed extracted from the simulation
 - analysis of the results can be done (this can be also done in another script)
- **a main section were**
 - batch files are generated for the simulation
 - the different task are selected using commend line argument.

Note: These two parts could be separate in two python script (file) without problem. But it seem more logical to kept them together and only have one python script file.

3.1 simulation_test_maps

To explain this example, we are going to start from the main section.

The batch file is define in a run method, which is called by the `__main__`, i.e., only run if the script is called from the command line, but not run if imported.

When the script is ready, these task are run in that order:

1. generate
2. check
3. read
4. analyze

```
$python simulation_test_maps.py generate
```

Start the batch files in parallel. It is easier to just select them all in Windows Explorer and right-click and open them.

```
BatchSimulationTestMapsMM2017_1.bat  
BatchSimulationTestMapsMM2017_2.bat
```

To check the progress of all simulations, i.e., how many simulations are done and todo

```
$python simulation_test_maps.py check
```

When all simulations are done, extract the results and save it in a hdf5 file.

```
$python simulation_test_maps.py read
```

The specified results can be analysed using the script (from the hdf5 file) or analysed using another script. For example, xray-spectrum-modeling project analyse the hdf5 file generate by this script.

An alternative to use the command line argument for these tasks, it to add them directly in the script and uncommenting the task you want to run in the main section of the script.

```
if __name__ == '__main__': #pragma: no cover  
    import sys  
    ...  
    if len(sys.argv) == 1:  
        sys.argv.append(mcxray.ANALYZE_TYPE_GENERATE_INPUT_FILE)  
        #sys.argv.append(mcxray.ANALYZE_TYPE_CHECK_PROGRESS)  
        #sys.argv.append(mcxray.ANALYZE_TYPE_ANALYZE_RESULTS)  
        #sys.argv.append(mcxray.ANALYZE_TYPE_ANALYZE_SCHEDULED_READ)  
    ...
```

By default the logging level is set at `logging.WARN` and above, but `pymcxray` gives a lot information at the `logging.INFO` level so it is recommended to set the logger level as follow

```
...
logging.getLogger().setLevel(logging.INFO)
...
```

The complete `__main__` is given below

```
if __name__ == '__main__': #pragma: no cover
    import sys
    logging.getLogger().setLevel(logging.INFO)
    logging.info(sys.argv)
    if len(sys.argv) == 1:
        sys.argv.append(mcxray.ANALYZE_TYPE_GENERATE_INPUT_FILE)
        #sys.argv.append(mcxray.ANALYZE_TYPE_CHECK_PROGRESS)
        #sys.argv.append(mcxray.ANALYZE_TYPE_ANALYZE_RESULTS)
        #sys.argv.append(mcxray.ANALYZE_TYPE_ANALYZE_SCHEDULED_READ)
    run()
```

The run method have three components

- a configuration file, see *Configuration file*
- a batch file object, see *Batch file*
- one or more simulation subclass, see *Simulation subclass*

Here is a example of a complete run method

```
def run():
    # import the batch file class.
    from pymcxray.BatchFileConsole import BatchFileConsole

    # Find the configuration file path
    configuration_file_path = get_current_module_path(__file__, "MCXRay_latest.cfg")
    program_name = get_mcxray_program_name(configuration_file_path)

    # Create the batch file object.
    batch_file = BatchFileConsole("BatchSimulationTestMapsMM2017", program_name,
    ↪ numberFiles=10)

    # Create the simulation object and add the batch file object to it.
    analyze = SimulationTestMapsMM2017(relativePath=r"mcxray/SimulationTestMapsMM2017
    ↪ ",
                                     configurationFilePath=configuration_file_path)
    analyze.run(batch_file)
```

3.2 Configuration file

The configuration file is a ini style configuration file for using pymcxray. It define the paths needed to generate and run the simulations.

```
[Paths]
mcxrayProgramName=console_mcxray_x64.exe
resultsMcGillPath=D:\Dropbox\hdemers\professional\results\simulations
mcxrayArchivePath=D:\Dropbox\hdemers\professional\softwareRelease\mcxray
mcxrayArchiveName=2016-04-11_11h41m28s_MCXRay_v1.6.6.0.zip
```

See the documentation of these functions for more detail on each option

- `pymcxray.get_mcxray_program_name()`
- `pymcxray.get_results_mcgill_path()`
- `pymcxray.get_mcxray_program_path()`
- `pymcxray.get_mcxray_archive_path()`

3.3 Batch file

The batch file is responsible to create the simulation structure with a copy of mcxray program. Batch files are generated to easily run the simulations. One important parameter to set is the *numberFiles*, this is the number of batch files generated and that can be run in parallel. For maximum efficiency it should be set as the number of logical processors minus 1 or 2. For example, on a computer with 12 logical processors, the *numberFiles* should be set at 10.

See `pymcxray.BatchFileConsole.BatchFileConsole` documentation for more information about the other parameters.

3.4 Simulation subclass

The simulation subclass allows to generate a lot of simulations by varying simulations parameters.

The main features are

- **generate input files**
 - regenerate input files of simulations not done
- check the progress of the simulations: done and todo
- **extract results from the completed simulation**
 - save the results in a hdf5 file for easier analysis
- optionally do the analysis of the results

To do that the user need to subclass `pymcxray.mcxray._Simulations` and overwrite these method

- `pymcxray.mcxray._Simulations._initData()` (required)
- `pymcxray.mcxray._Simulations.getAnalysisName()` (required)
- `pymcxray.mcxray._Simulations.createSpecimen()` (required)
- `pymcxray.mcxray._Simulations.read_one_results_hdf5()` (optional)
- `pymcxray.mcxray._Simulations.analyze_results_hdf5()` (optional)

Warning: If any of the required method is modified, the simulation have to be redone completely. It is recommended to just delete the root path for the analysis and generate the input files and do the simulations. For this example, delete *SimulationTestMapsMM2017* folder.

Warning: If `pymcxray.mcxray._Simulations.read_one_results_hdf5()` is modified. In some case, the hdf5 need to be deleted.¹ Furthermore, if the results were deleted: *delete_result_files* is *True*, the simulation have to be redone.

Below are given example for each method, for more detail see the method documentation.

3.4.1 Init data

This method is used to specify the options for the analysis and also the parameters used in the simulations.

The most important options for the analysis are:

- *use_hdf5* to use the recommended hdf5 method. If it is *False* the older serial method will be used.
- *delete_result_files* if it is *True*, the result file are deleted after added in the hdf5 file. Very useful when creating a lot files like for a map.
- *createBackup* if *True* create a backup of the hdf5 file before adding more results to it. Usefull to not loss data in case of error or crash, but you should delete backup file manually as they can take a lot of space.

Warning: If *delete_result_files* is *True*, all results are deleted for a simulation and only the results specified in `pymcxray.mcxray._Simulations.read_one_results_hdf5()` are kept. If `pymcxray.mcxray._Simulations.read_one_results_hdf5()` is modified to extract more results, the simulation have to be simulate again.

This is the recommended values for the options, only change them when you are sure everything is working OK.

```
def _initData(self):
    self.use_hdf5 = True
    self.delete_result_files = False
    self.createBackup = True
```

The simulation parameters are specified in this method.

Note: If not specified, the script use MCXRay default parameters. Start MCXRay program to see the default value of each parameters.

To change simulation parameters, create a `pymcxray.SimulationsParameters.SimulationsParameters` object

```
self._simulationsParameters = SimulationsParameters()
```

Two kinds of parameter can be added:

- varied specified with a list of values
- fixed specified with a single value

The script will automatically generate a simulation for all combination of the varied parameters.

Warning: Adding a lot of varied parameters with a lot of values can generate a lot of simulations. Above 1000 simulations, all tasks of the script will be slow because of the generation or reading of a lot of files. It is recommended to start with 2 or 3 varied parameters and short list of values and test all tasks of the script. When the tests are OK and results make sense, you can increase the list of values. To add more varied parameters, it is recommended to create a new script with again only 2 or 3 varied parameters.

The parameters that can be added are defined as keyword starting with *PARAMETER_* in the module *pymcrray.SimulationsParameters*. If you don't find the parameter you want request an "enhancement" at <https://github.com/drix00/pymcrray/issues>.

Here is an example how-to add simulation parameters

```
from pymcrray.SimulationsParameters import SimulationsParameters, PARAMETER_INCIDENT_
    ENERGY_keV, PARAMETER_NUMBER_ELECTRONS, \
PARAMETER_BEAM_POSITION_nm, PARAMETER_NUMBER_XRAYS
...

class SimulationTestMapsMM2017(mcxray._Simulations):
    def _initData(self):
        ...

        # Local variables for value and list if values.
        energy_keV = 30.0
        number_electrons = 10000

        #number_xrays_list = [10, 20, 30, 50, 60, 100, 200, 500, 1000]
        number_xrays_list = [10]
        xs_nm = np.linspace(-5.0e3, 5.0e3, 3)
        probePositions_nm = [tuple(position_nm) for position_nm in np.transpose([np.
    tile(xs_nm, len(xs_nm)), np.repeat(xs_nm, len(xs_nm))]).tolist()]

        # Simulation parameters
        self._simulationsParameters = SimulationsParameters()

        self._simulationsParameters.addVaried(PARAMETER_NUMBER_XRAYS, number_xrays_
    list)
        self._simulationsParameters.addVaried(PARAMETER_BEAM_POSITION_nm,
    probePositions_nm)

        self._simulationsParameters.addFixed(PARAMETER_INCIDENT_ENERGY_keV, energy_
    keV)
        self._simulationsParameters.addFixed(PARAMETER_NUMBER_ELECTRONS, number_
    electrons)
```

3.4.2 Analysis name

This method specify the name of the analysis or experiment for which the simulation are done. Normally similar to the name of the class and mostly used as basename for the input files and result files. In case of two scripts writing the same path, it allows to differentiate them, but it is not recommended to run two scripts in the same folde.

```
class SimulationTestMapsMM2017(mcxray._Simulations):
    ...
    def getAnalysisName(self):
        return "SimulationTestMapsMM2017"
    ...
```

3.4.3 Create specimen

This method is used to create the specimen for each simulation. The argument of the method contains the option of the specific simulation and can be used to create the specimen. The *pymcrray.Simulation.Simulation* module contains predefined specimen which can be use in this method.

Here an example how-to use the *parameters* argument and the predefined specimen

```
def createSpecimen(self, parameters):
    weightFractions = parameters[PARAMETER_WEIGHT_FRACTIONS]

    elements = [(self.atomicNumberA, weightFractions[0]),
                (self.atomicNumberB, weightFractions[1])]
    specimen = Simulation.createAlloyBulkSample(elements)
    return specimen
```

A more complex example, where each region are specified is given below

```
def createSpecimen(self, parameters):
    # Create the specimen with a name and number of regions.
    specimen = Specimen.Specimen()
    specimen.name = "Maps01"
    specimen.numberRegions = 10

    # Region 0
    region = Region.Region()
    region.numberElements = 0
    region.regionType = RegionType.REGION_TYPE_BOX
    parameters = [-10000000000.0, 10000000000.0, -10000000000.0, 10000000000.0, 0.0, ↵
↵20000000000.0]
    region.regionDimensions = RegionDimensions.RegionDimensionsBox(parameters)
    specimen.regions.append(region)

    # Region 1
    region = Region.Region()
    region.numberElements = 2
    region.elements = [Element.Element(27, massFraction=0.01), Element.Element(26, ↵
↵massFraction=0.99)]
    region.regionType = RegionType.REGION_TYPE_BOX
    parameters = [-7.5e4, -2.5e4, -7.5e4, -2.5e4, 0.0, 0.2e4]
    region.regionDimensions = RegionDimensions.RegionDimensionsBox(parameters)
    specimen.regions.append(region)
    ...
```

Warning: Creating a specimen in MCXRay is complicate as sometime you need to create a empty region 0. Look at the predefined specimen in `pymcrray.Simulation.Simulation` for help. Drawing the trajectory with the `FileFormat.Results.ElectronTrajectoriesResults` will help debug the specimen. You can also request a “help wanted” at <https://github.com/drix00/pymcrray/issues>.

3.4.4 Read one simulation results

This method extract results from one complete simulation and added them in a hdf5 group for this simulation. The result that can be extracted are in the package `pymcrray.FileFormat.Results` and only the class implementing `write_hdf5()` can be extracted. If the desired results does not implement the `write_hdf5()` method, request an “enhancement” at <https://github.com/drix00/pymcrray/issues>.

Note: The format of the hdf5 file is not well documented. Check the implementation of the `write_hdf5()` method and request an “enhancement” at <https://github.com/drix00/pymcrray/issues> for the documentation.

Note: The program HDFView is useful to look at the hdf5 file. See <https://support.hdfgroup.org/products/java/hdfview/>.

Here is an example how-to extract the electron results (BSE, TE, ...)

```
def read_one_results_hdf5(self, simulation, hdf5_group):
    electronResults = ElectronResults.ElectronResults()
    electronResults.path = self.getSimulationsPath()
    electronResults.basename = simulation.resultsBaseName
    electronResults.read()
    electronResults.write_hdf5(hdf5_group)
```

So far this class are implemented with hdf5 support

- `pymcxray.FileFormat.Results.ElectronResults.ElectronResults`
- `pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic`
- `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic`
- `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm`
- `pymcxray.FileFormat.Results.XrayIntensities.XrayIntensities`
- `pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted`
- `pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected`

3.4.5 Analyze all simulations

This method is only needed for the task analyze.

Often the method start by calling `mcxray._Simulations.readResults()` to read all new results and add them in the hdf5 file.

The example below shows how-to open the hdf5 file in memory for somewhat fast analysis simulation. The file is read only at the beginning and stored in memory.

```
def analyze_results_hdf5(self): #pragma: no cover
    self.readResults()

    file_path = self.get_hdf5_file_path()
    with h5py.File(file_path, 'r', driver='core') as hdf5_file:
        hdf5_group = self.get_hdf5_group(hdf5_file)
        logging.info(hdf5_group.name)
```

Todo: Document `pymcxray.mcxray`

Todo: Document `pymcxray.SimulationsParameters`

Todo: Document `pymcxray.Simulation`

Todo: Document `pymcxray.FileFormat.Results.ElectronResults.ElectronResults`

Todo: Document `pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic`

Todo: Document `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic`

Todo: Document `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm`

Todo: Document `pymcxray.FileFormat.Results.XrayIntensities.XrayIntensities`

Todo: Document `pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted`

Todo: Document `pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected`

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at <https://github.com/drix00/pymcxray/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

4.1.4 Write Documentation

pymcxray could always use more documentation, whether as part of the official pymcxray docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/drix00/pymcxray/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *pymcxray* for local development.

1. Fork the *pymcxray* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/pymcxray.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv pymcxray
$ cd pymcxray/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 pymcxray tests
$ python setup.py test or py.test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/drix00/pymcxray/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_pymcxray
```


CHAPTER 5

Credits

5.1 Development Lead

- Hendrix Demers <hendrix.demers@mail.mcgill.ca>

5.2 Contributors

None yet. Why not be the first?

6.1 0.1.2 (2017-06-26)

- Add create multi-layers specimen.
- Add number of layers X, Y, and Z as parameters.

6.2 0.1.0 (2017-03-07)

- First release on PyPI.

CHAPTER 7

ToDo

ToDo: Add units.

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/pymcxray/ElementProperties.g_FermiEnergy`, line 6.)

ToDo: Add units.

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/pymcxray/ElementProperties.g_kFermi`, line 6.)

ToDo: Add units.

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/pymcxray/ElementProperties.g_plasmonEnergy`, line 6.)

ToDo: Add pypi installation for this project.

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/installation.rst`, line 13.)

ToDo: Document `pymcxray.mcxray`

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst`, line 436.)

Todo: Document `pymcxray.SimulationsParameters`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 437.)

Todo: Document `pymcxray.Simulation`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 438.)

Todo: Document `pymcxray.FileFormat.Results.ElectronResults.ElectronResults`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 439.)

Todo: Document `pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 440.)

Todo: Document `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 441.)

Todo: Document `pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 442.)

Todo: Document `pymcxray.FileFormat.Results.XrayIntensities.XrayIntensities`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 443.)

Todo: Document `pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted`

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst, line 444.)

Todo: Document `pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected`

(The [original entry](#) is located in `/home/docs/checkouts/readthedocs.org/user_builds/pymcxray/checkouts/stable/docs/usage.rst`, line 445.)

8.1 pymcxray package

8.1.1 Subpackages

pymcxray.FileFormat package

Subpackages

pymcxray.FileFormat.Results package

Subpackages

pymcxray.FileFormat.Results.exported package

Submodules

pymcxray.FileFormat.Results.exported.DataMap module

Read data map exported by MCXRay.

```
class pymcxray.FileFormat.Results.exported.DataMap.DataMap (filepath)
    Bases: object
        imageName
        pixels
        read ()
        saveImage (imageFilepath=None)
        showImage ()
```

size

`pymcrray.FileFormat.Results.exported.DataMap.run()`

pymcrray.FileFormat.Results.exported.XrayIntensityXY module

Read x-ray intensity distribution in XY exported from mcrray GUI.

```
class pymcrray.FileFormat.Results.exported.XrayIntensityXY.XrayIntensityXY
    Bases: object

    readData (filepath)
```

pymcrray.FileFormat.Results.exported.test_DataMap module

Tests for the module *DataMap*.

```
class pymcrray.FileFormat.Results.exported.test_DataMap.TestDataMap (methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module DataMap.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp ()
        Setup method.

    tearDown ()
        Teardown method.

    testSkeleton ()
        First test to check if the testcase is working with the testing framework.

    test_init ()
        Tests for method init.

    test_read ()
        Tests for method read.
```

pymcrray.FileFormat.Results.exported.test_XrayIntensityXY module

Tests for the module *XrayIntensityXY*.

```
class pymcrray.FileFormat.Results.exported.test_XrayIntensityXY.TestXrayIntensityXY (methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module XrayIntensityXY.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp ()
        Setup method.

    tearDown ()
        Teardown method.
```

testOpenFile()

Test if the test data file can be open.

testSkeleton()

First test to check if the testcase is working with the testing framework.

pymcrray.FileFormat.Results.exported.tests module

Module contents

Submodules

pymcrray.FileFormat.Results.BaseResults module

BaseResults

```
class pymcrray.FileFormat.Results.BaseResults.BaseResults (path=”, base-  
name=’MCXRay’)
```

Bases: object

basename

filepath

path

pymcrray.FileFormat.Results.BeamParameters module

MCXRay beam parameters from results file.

```
class pymcrray.FileFormat.Results.BeamParameters.BeamParameters
```

Bases: object

acquisitionTime_s

current_A

diameter90_A

gaussianMean

gaussianSigma

incidentEnergy_keV

readFromLines (*lines*)

tiltAngle_deg

pymcrray.FileFormat.Results.DetectorParameters module

MCXRay detector parameters from results file.

```
class pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters
```

Bases: object

angleBetweenDetectorSpecimenNormal_deg

angleBetweenDetectorXAxis_deg

```
beamDetectorDistance_cm
crystalDensity_g_cm3
crystalName
crystalRadius_cm
crystalThickness_cm
deadLayerThickness_A
diffusionLength_A
noiseEdsDetector_eV
readFromLines (lines)
solidAngle_deg
surfaceQualityFactor
takeoffAngleEffective_deg
takeoffAngleNormalIncidence_deg
thicknessAir_um
thicknessAlWindow_um
thicknessBeWindow_um
thicknessH2O_um
thicknessMoxtek_um
thicknessOil_um
thicknessTiWindow_um
```

pymcxray.FileFormat.Results.Dump module

MCXRay dump results file.

```
class pymcxray.FileFormat.Results.Dump.Dump
    Bases: object
    read (filepath)
```

pymcxray.FileFormat.Results.ElectronExistResults module

Read electron exit results simulated with MCXRay.

```
class pymcxray.FileFormat.Results.ElectronExistResults.ElectronDetector
    Bases: object
    azimuthalAngleRange_deg
    detectElectrons (data)
    energyRange_keV
    maximumAzimuthalAngle_deg
    maximumEnergy_keV
```

```

    maximumPolarAngle_deg
    minimumAzimuthalAngle_deg
    minimumEnergy_keV
    minimumPolarAngle_deg
    polarAngleRange_deg
class pymcrray.FileFormat.Results.ElectronExistResults.ElectronExistResults (*args,
                                                                    **kwargs)
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    data
    getEnergyDistribution (numberBins=10)
    numberData
    read ()

```

pymcrray.FileFormat.Results.ElectronParameters module

MCXRay electron parameters results file.

```

class pymcrray.FileFormat.Results.ElectronParameters.ElectronParameters
    Bases: object
    backscatteredRatio
    eRatio
    internalRatio
    meanAzimuthalAngleCollision_deg
    meanDistanceBetweenCollisions_A
    meanNumberCollisionPerElectrons
    meanPolarAngleCollision_deg
    numberSimulatedElectrons
    readFromLines (lines)
    skirtRatio
    throughRatio

```

pymcrray.FileFormat.Results.ElectronResults module

Read ElectronResults MCXRay results file.

```

class pymcrray.FileFormat.Results.ElectronResults.ElectronResults
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    fieldNames
    fractionBackscatteredElectrons
    fractionInternalElectrons
    fractionSkirtedElectrons

```

```
fractionTransmittedElectrons  
numberBackscatteredElectrons  
numberElectronCollisions  
numberInternalElectrons  
numberSimulatedElectrons  
numberSkirtedElectrons  
numberTransmittedElectrons  
read()  
write_hdf5(hdf5_group)
```

pymcrray.FileFormat.Results.ElectronTrajectoriesResults module

Read MCXray electron trajectories results file.

```
class pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision
```

Bases: object

```
collisionType  
correctedX_A  
correctedY_A  
correctedZ_A  
energy_keV  
indexRegion  
x_A  
y_A  
z_A
```

```
class pymcrray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults (
```

Bases: object

```
drawXY (title="", corrected=False, x_limit=None, y_limit=None)  
drawXZ (title="", corrected=False, theta_deg=0.0, colorType='colorTrajectoryType', trajectoryIn-  
dexes=None, x_limit=None, y_limit=None)  
drawYZ (title="", corrected=False, x_limit=None, y_limit=None)  
getElectronGunPositions_nm()  
read (filepath)  
write_hdf5 (hdf5_group)
```

```
class pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Trajectory
```

Bases: object

```
addCollision (collision)  
collisions  
index
```

trajectoryType

```
pymcrray.FileFormat.Results.ElectronTrajectoriesResults.run()  
pymcrray.FileFormat.Results.ElectronTrajectoriesResults.runAuCThinFilm()  
pymcrray.FileFormat.Results.ElectronTrajectoriesResults.runFogging()
```

pymcrray.FileFormat.Results.ElementParameters module

MCXRay element parameters result file.

```
class pymcrray.FileFormat.Results.ElementParameters.ElementParameters  
    Bases: object
```

pymcrray.FileFormat.Results.Intersections module

MCXRay intersections results file.

```
class pymcrray.FileFormat.Results.Intersections.Intersections  
    Bases: object  
    extractFromLines(lines)
```

pymcrray.FileFormat.Results.MicroscopeParameters module

MCXRay microscope parameter in results file.

```
class pymcrray.FileFormat.Results.MicroscopeParameters.MicroscopeParameters  
    Bases: object  
    beamParameters  
    detectorParameters  
    readFromLines(lines)
```

pymcrray.FileFormat.Results.ModelParameters module

MCXRay model parameters from results file.

```
class pymcrray.FileFormat.Results.ModelParameters.ModelParameters  
    Bases: object  
    atomCollisionModel  
    atomCollisionScreeningModel  
    atomCrossSectionModel  
    atomCrossSectionScreeningModel  
    atomElectronRangeModel  
    atomEnergyLossModel  
    atomMeanIonizationPotentialModel  
    atomScreeningModel
```

```
bremsstrahlungCrossSectionModel
characteristicCrossSectionModel
readFromLines(lines)
regionEnergyLossModel
```

pymcrray.FileFormat.Results.Phirhoz module

MCXRay phirhoz result file.

```
class pymcrray.FileFormat.Results.Phirhoz.Phirhoz(symbol, shell)
    Bases: object

    depths_A
    intensity
    readFromLines(lines)
    shell
    symbol
    values
```

pymcrray.FileFormat.Results.PhirhozElement module

MCXRay result file phirhoz element.

```
class pymcrray.FileFormat.Results.PhirhozElement.PhirhozElement
    Bases: object

    isIonizationShell_K
    isIonizationShell_L
    isIonizationShell_M
    readFromLine(line)
    symbol
    weightFraction
```

pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic module

Read PhirhozEmittedCharacteristic file from MCXRay.

```
class pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults

    depth_A
    depth_nm
    fieldNames
    phirhozs
    read(regionID=0)
```



```
write_hdf5 (hdf5_group)
```

pymcrray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm module

Read mcrray phirhoz thin film emitted results.

```
class pymcrray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm.PhirhozEmittedChara
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    fieldNames
    getIntensity (regionID, atomicSymbol, xrayLine)
    intensities
    numberRegions
    read ()
```

pymcrray.FileFormat.Results.PhirhozGenerated module

MCXRay phirhoz generated result file.

```
class pymcrray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated
    Bases: object
    electronParameters
    getCharacteristicPhiRhoZ (regionID)
    microscopeParameters
    modelParameters
    numberRegions
    read (filepath)
    readElectron (lines)
    readMicroscope (lines)
    readRegions (lines)
    readSimulationParameters (lines)
    readSolidSimulationModels (lines)
    simulationParameters
```

pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic module

Read PhirhozGeneratedCharacteristic file from MCXRay program.

```
class pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacter
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    depth_A
    depth_nm
    fieldNames
```

```
phirhozs
read (regionID=0)
write_hdf5 (hdf5_group)
```

pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm module

Read mcxray phirhoz thin film generated results.

```
class pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedC
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    fieldNames
    getIntensity (regionID, atomicSymbol, xrayLine)
    get_subshells ()
    get_symbols ()
    intensities
    numberRegions
    read ()
    write_hdf5 (hdf5_group)
```

pymcxray.FileFormat.Results.PhirhozRegion module

MCXRay phirhoz results file for a region.

```
class pymcxray.FileFormat.Results.PhirhozRegion.PhirhozRegion (numberEnergyWindows,
                                                                numberLayersZ)
    Bases: object
    readBackgroundPhirhozs (lines)
    readCharacteristicPhirhozs (lines)
    readElements (lines)
    readFromLines (lines)
    readPhirhozDistributions (lines)
    readRegionVolume (lines)
    regionID
```

pymcxray.FileFormat.Results.RegionParameters module

MCXRay region parameters result file.

```
class pymcxray.FileFormat.Results.RegionParameters.RegionParameters
    Bases: object
    elements
    layerThickness_A
    numberElements
```

regionID

pymcxray.FileFormat.Results.RegionVolume module

MCXRay result file region volume.

```
class pymcxray.FileFormat.Results.RegionVolume.RegionVolume
    Bases: object
        readFromLines (lines)
        regionID
```

pymcxray.FileFormat.Results.SimulationParameters module

MCXRay simulation parameters results file.

```
class pymcxray.FileFormat.Results.SimulationParameters.SimulationParameters
    Bases: object
        edsMaximumEnergy_keV
        generalizedWalk
        interpolationType
        maximumLiveTime_s
        numberChannels
        numberElectrons
        numberEnergyWindows
        numberLayersX
        numberLayersY
        numberLayersZ
        numberPhotons
        readFromLines (lines)
        useLiveTime_s
```

pymcxray.FileFormat.Results.Spectra module

MCXRay spectra result file.

```
class pymcxray.FileFormat.Results.Spectra.Spectra
    Bases: object
        getElementSpectra ()
        getElementSpectrum (regionID, elementName)
        getRegionParameters (regionID)
        getRegionSpectrum (regionID)
        getSpecimenSpectrum ()
```

```
numberRegions
read (filepath)
readRegion (lines)
readRegions (lines)
readSpecimen (lines)
```

pymcrray.FileFormat.Results.SpectraEDS module

Read MCXRay spectra EDS results file.

```
class pymcrray.FileFormat.Results.SpectraEDS.SpectraEDS
    Bases: object
    characteristicProbability
    elements
    numberCharateristicPeaks
    numberElements
    numberSimulatedPhotons
    readFileObject (inputFile)
    readFilepath (filepath)
    readPartialSpectraReferenceSection (lines)
    readRegionSpectraSection (lines)
    readTestInputSection (lines)
    regionID
```

```
pymcrray.FileFormat.Results.SpectraEDS.runExample()
```

pymcrray.FileFormat.Results.Spectrum module

MCXRay spectrum result file.

```
class pymcrray.FileFormat.Results.Spectrum.Spectrum
    Bases: object
    backgroundIntensities
    characteristicIntensities
    energies_keV
    intensities
```

pymcrray.FileFormat.Results.SpectrumEDS module

Read MCXRay EDS spectrm from results file.

```
class pymcrray.FileFormat.Results.SpectrumEDS.SpectrumEDS (lines=None)
    Bases: object
```

```

channels
countsList
enegies_keV
readLines (lines)

```

pymcrray.FileFormat.Results.Tags module

MCXRay tags used in output files.

```

exception pymcrray.FileFormat.Results.Tags.TagNotFoundError
    Bases: ValueError

```

```

pymcrray.FileFormat.Results.Tags.findAllTag (tag, lines, contains=None)

```

```

pymcrray.FileFormat.Results.Tags.findTag (tag, lines)

```

pymcrray.FileFormat.Results.XrayIntensities module

Xray intensities result file from MCXRay.

```

class pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    fieldNames
    getAtomicNumberLineEnergySets ()
    getDetectedIntensity (atomicNumber, xrayLine)
    getIntensity (data_type, region_id, atomic_number, xray_line)
    getIntensityEmitted (atomicNumber, xrayLine, total=True)
    getIntensityEmittedDetected (atomicNumber, xrayLine, total=True)
    getIntensityGenerated (atomicNumber, xraySubshell, total=False)
    getIntensityGeneratedDetected (atomicNumber, xraySubshell)
    get_result_types ()
    get_xray_lines ()
    intensities
    numberIntensities
    read ()
    write_hdf5 (hdf5_group)

```

pymcrray.FileFormat.Results.XraySimulatedSpectraRegion module

description

```

class pymcrray.FileFormat.Results.XraySimulatedSpectraRegion.XraySimulatedSpectraRegion
    Bases: pymcrray.FileFormat.Results.BaseResults.BaseResults
    channelNumbers

```

```
detectedIntensities
eNetPeak
energiesReference_keV
energies_keV
fieldNames
peakToBackgrpound
peakToBackgrpoundAverage
read (regionID=0)
simulatedIntensities
```

pymcxray.FileFormat.Results.XraySimulatedSpectraSpecimen module

description

```
class pymcxray.FileFormat.Results.XraySimulatedSpectraSpecimen.XraySimulatedSpectraSpecimen
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    energies_keV
    fieldNames
    read ()
    totals
```

pymcxray.FileFormat.Results.XraySpectraAtomEmittedDetectedLines module

Read MCXRay XraySpectraAtomEmittedDetectedLines file.

```
class pymcxray.FileFormat.Results.XraySpectraAtomEmittedDetectedLines.XraySpectraAtomEmitted
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    characteristics
    energies_keV
    fieldNames
    read (regionID)
```

pymcxray.FileFormat.Results.XraySpectraRegionEmitted module

Read XraySpectraRegionEmitted mcxray result file.

```
class pymcxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    bremsstrahlungValue_1_ekeVsr (energy_keV)
    bremsstrahlung_1_ekeVsr
    characteristicValue_1_ekeVsr (energy_keV)
    characteristic_1_ekeVsr
```

```

energies_keV
fieldnames
read(regionID=0)
totalValue_1_ekeVsr(energy_keV)
total_1_ekeVsr

```

```
pymcxray.FileFormat.Results.XraySpectraRegionEmitted.run()
```

pymcxray.FileFormat.Results.XraySpectraRegionsEmitted module

Read all XraySpectraRegionsEmitted mcxray result files for all regions.

```

class pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted
    Bases: pymcxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
    read()
    write_hdf5(hdf5_group)

```

```
pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.run()
```

pymcxray.FileFormat.Results.XraySpectraSpecimen module

Read XraySpectraSpecimen MCXRay results file.

```

class pymcxray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    backgrounds
    characteristics
    energies_keV
    fieldNames
    read()
    totals
    write_hdf5(hdf5_group)

```

pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected module

Read XraySpectraSpecimenEmittedDetected MCXRay results file.

```

class pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected
    Bases: pymcxray.FileFormat.Results.BaseResults.BaseResults
    backgrounds
    characteristics
    energies_keV
    fieldNames

```

```
read()
totals
write_hdf5(hdf5_group)
```

pymcxray.FileFormat.Results.test_BaseResults module

Tests for the module *BaseResults*.

```
class pymcxray.FileFormat.Results.test_BaseResults.TestBaseResults (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *BaseResults*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp()
    Setup method.

tearDown()
    Teardown method.

testSkeleton()
    First test to check if the testcase is working with the testing framework.

test_basename()
    Tests for method basename.

test_init()
    Tests for method init.

test_path()
    Tests for method path.
```

pymcxray.FileFormat.Results.test_BeamParameters module

Tests for module *BeamParameters*.

```
class pymcxray.FileFormat.Results.test_BeamParameters.TestBeamParameters (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *BeamParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp()
    Setup method.

tearDown()
    Teardown method.

testSkeleton()
    First test to check if the testcase is working with the testing framework.

test_readFromLines()
    Tests for method readFromLines.
```


pymcxray.FileFormat.Results.test_DetectorParameters module

Tests for module *DetectorParameters*.

```
class pymcxray.FileFormat.Results.test_DetectorParameters.TestDetectorParameters (methodName=  
    Bases: unittest.case.TestCase  
  
    TestCase class for the module DetectorParameters.  
  
    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the  
    instance does not have a method with the specified name.  
  
    setUp ()  
        Setup method.  
  
    tearDown ()  
        Teardown method.  
  
    testSkeleton ()  
        First test to check if the testcase is working with the testing framework.  
  
    test_readFromLines ()  
        Tests for method readFromLines.
```

pymcxray.FileFormat.Results.test_Dump module

Tests for module *Dump*

```
class pymcxray.FileFormat.Results.test_Dump.TestDump (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    TestCase class for the module Dump.  
  
    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the  
    instance does not have a method with the specified name.  
  
    setUp ()  
        Setup method.  
  
    tearDown ()  
        Teardown method.  
  
    testSkeleton ()  
        First test to check if the testcase is working with the testing framework.  
  
    test_read ()  
        Tests for method read.
```

pymcxray.FileFormat.Results.test_ElectronExistResults module

pymcxray.FileFormat.Results.test_ElectronParameters module

Tests for module *ElectronParameters*.

```
class pymcxray.FileFormat.Results.test_ElectronParameters.TestElectronParameters (methodName=  
    Bases: unittest.case.TestCase  
  
    TestCase class for the module ElectronParameters.
```

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Tear down method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_readFromLines()

Tests for method *readFromLines*.

`pymcrray.FileFormat.Results.test_ElectronParameters.getLinesAndReference()`

pymcrray.FileFormat.Results.test_ElectronResults module

Tests for the module *XraySpectraSpecimen*.

class `pymcrray.FileFormat.Results.test_ElectronResults.TestElectronResults` (*methodName='runTest'*)

Bases: `unittest.case.TestCase`

TestCase class for the module *ElectronResults*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Tear down method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_read()

Tests for method *read*.

pymcrray.FileFormat.Results.test_ElementParameters module

Tests for the module *ElementParameters*.

class `pymcrray.FileFormat.Results.test_ElementParameters.TestElementParameters` (*methodName='runTest'*)

Bases: `unittest.case.TestCase`

TestCase class for the module *ElementParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Tear down method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

pymcxray.FileFormat.Results.test_Intersections module

Tests for module *Intersections*.

```
class pymcxray.FileFormat.Results.test_Intersections.TestIntersections (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *Intersections*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()
    Setup method.
```

```
tearDown ()
    Teardown method.
```

```
testSkeleton ()
    First test to check if the testcase is working with the testing framework.
```

```
test_extractFromLines ()
    Tests for method extractFromLines.
```

pymcxray.FileFormat.Results.test_MicroscopeParameters module

Tests for module *MicroscopeParameters*.

```
class pymcxray.FileFormat.Results.test_MicroscopeParameters.TestMicroscopeParameters (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *MicroscopeParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()
    Setup method.
```

```
tearDown ()
    Teardown method.
```

```
testSkeleton ()
    First test to check if the testcase is working with the testing framework.
```

```
test_readFromLines ()
    Tests for method readFromLines.
```

```
pymcxray.FileFormat.Results.test_MicroscopeParameters.getLinesAndReference ()
```

pymcxray.FileFormat.Results.test_ModelParameters module

ModelParameters

```
class pymcxray.FileFormat.Results.test_ModelParameters.TestModelParameters (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *ModelParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_readFromLines()
Tests for method *readFromLines*.

`pymcxray.FileFormat.Results.test_ModelParameters.getLinesAndReference()`

pymcxray.FileFormat.Results.test_Phirhoz module

Tests for the module *Phirhoz*.

class `pymcxray.FileFormat.Results.test_Phirhoz.TestPhirhoz` (*methodName='runTest'*)
Bases: `unittest.case.TestCase`

TestCase class for the module *Phirhoz*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_readFromLines()
Tests for method *readFromLines*.

`pymcxray.FileFormat.Results.test_Phirhoz.getLinesAndReference(path)`

pymcxray.FileFormat.Results.test_PhirhozElement module

Tests for the module *PhirhozElement*.

class `pymcxray.FileFormat.Results.test_PhirhozElement.TestPhirhozElement` (*methodName='runTest'*)
Bases: `unittest.case.TestCase`

TestCase class for the module *PhirhozElement*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_readFromLine()
Tests for method *readFromLines*.

`pymcxray.FileFormat.Results.test_PhirhozElement.getLineAndReference()`

pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristic module

Tests for the module *PhirhozEmittedCharacteristic*.

class `pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristic.TestPhirhozEmittedChar`
Bases: `unittest.case.TestCase`

TestCase class for the module *PhirhozEmittedCharacteristic*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_read()
Tests for method *read*.

pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristicThinFilm module

Tests for module *PhirhozEmittedCharacteristicThinFilm*.

class `pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristicThinFilm.TestPhirhozEmi`
Bases: `unittest.case.TestCase`

TestCase class for the module *PhirhozEmittedCharacteristicThinFilm*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_getIntensity()
Tests for method *getIntensity*.

test_read()
Tests for method *read*.

pymcrray.FileFormat.Results.test_PhirhozGenerated module

Tests for the module *PhirhozGenerated*.#

```
class pymcrray.FileFormat.Results.test_PhirhozGenerated.TestPhirhozGenerated(methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module moduleName.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp()
        Setup method.

    tearDown()
        Teardown method.

    testSkeleton()
        First test to check if the testcase is working with the testing framework.

    test_read()
        Tests for method read.
```

pymcrray.FileFormat.Results.test_PhirhozGeneratedCharacteristic module

Tests for the module *PhirhozGeneratedCharacteristic*.

```
class pymcrray.FileFormat.Results.test_PhirhozGeneratedCharacteristic.TestPhirhozGeneratedCharacteristic(methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module PhirhozGeneratedCharacteristic.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp()
        Setup method.

    tearDown()
        Teardown method.

    testSkeleton()
        First test to check if the testcase is working with the testing framework.

    test_read()
        Tests for method read.
```

pymcrray.FileFormat.Results.test_PhirhozGeneratedCharacteristicThinFilm module

Tests for module *PhirhozGeneratedCharacteristicThinFilm*.

```
class pymcrray.FileFormat.Results.test_PhirhozGeneratedCharacteristicThinFilm.TestPhirhozGeneratedCharacteristicThinFilm(methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module PhirhozGeneratedCharacteristicThinFilm.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.
```

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_getIntensity()
Tests for method *getIntensity*.

test_read()
Tests for method *read*.

pymcxray.FileFormat.Results.test_PhirhozRegion module

Tests for the module *PhirhozRegion*.

class pymcxray.FileFormat.Results.test_PhirhozRegion.**TestPhirhozRegion** (*methodName='runTest'*)
Bases: unittest.case.TestCase

TestCase class for the module *PhirhozRegion*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_readFromLines()
Tests for method *readFromLines*.

pymcxray.FileFormat.Results.test_PhirhozRegion.**getLinesAndReference** (*path*)

pymcxray.FileFormat.Results.test_RegionParameters module

Tests for the module *RegionParameters*.

class pymcxray.FileFormat.Results.test_RegionParameters.**TestRegionParameters** (*methodName='runTest'*)
Bases: unittest.case.TestCase

TestCase class for the module *RegionParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

pymcxray.FileFormat.Results.test_RegionVolume module

Tests for the module *RegionVolume*.

```
class pymcxray.FileFormat.Results.test_RegionVolume.TestRegionVolume (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *RegionVolume*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
NOtest_readFromLines ()
    Tests for method readFromLines.
```

```
setUp ()
    Setup method.
```

```
tearDown ()
    Teardown method.
```

```
testSkeleton ()
    First test to check if the testcase is working with the testing framework.
```

```
pymcxray.FileFormat.Results.test_RegionVolume.getLinesAndReference ()
```

pymcxray.FileFormat.Results.test_SimulationParameters module

Tests for the module *SimulationParameters*.

```
class pymcxray.FileFormat.Results.test_SimulationParameters.TestSimulationParameters (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *SimulationParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()
    Setup method.
```

```
tearDown ()
    Teardown method.
```

```
testSkeleton ()
    First test to check if the testcase is working with the testing framework.
```

```
test_readFromLines ()
    Tests for method readFromLines.
```

```
pymcxray.FileFormat.Results.test_SimulationParameters.getLinesAndReference ()
```

pymcxray.FileFormat.Results.test_Spectra module

Tests for the module *Spectra*.

```
class pymcxray.FileFormat.Results.test_Spectra.TestSpectra (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *Spectra*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()
    Setup method.

tearDown ()
    Teardown method.

testSkeleton ()
    First test to check if the testcase is working with the testing framework.

test__extractElementHeader ()
    Tests for method _extractElementHeader.

test__extractRegionHeader ()
    Tests for method _extractRegionHeader.

test_read ()
    Tests for method read.

test_readRegion ()
    Tests for method readRegion.

test_readSpecimen ()
    Tests for method readSpecimen.
```

pymcxray.FileFormat.Results.test_SpectraEDS module

Tests for module *SpectraEDS*.

```
class pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS (methodName='runTest')
    Bases: unittest.case.TestCase
```

TestCase class for the module *SpectraEDS*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()
    Setup method.

tearDown ()
    Teardown method.

testSkeleton ()
    First test to check if the testcase is working with the testing framework.

test_FindTestData ()
    Tests for method FindTestData.

test__extractTotalCounts ()
    Tests for method _extractTotalCounts.

test__isPartialSpectraReferenceSection ()
    Tests for method _isTestInputSection.

test__isRegionSpectraSection ()
    Tests for method _isRegionSpectraSection.

test__isTestInputSection ()
    Tests for method _isTestInputSection.
```

test_readPartialSpectraReferenceSection()

Tests for method *readTestInputSection*.

test_readRegionSpectraSection()

Tests for method *readRegionSpectraSection*.

test_readTestInputSection()

Tests for method *readTestInputSection*.

pymcxray.FileFormat.Results.test_Spectrum module

Tests for the module *Spectrum*

class pymcxray.FileFormat.Results.test_Spectrum.**TestSpectrum** (*methodName='runTest'*)

Bases: `unittest.case.TestCase`

TestCase class for the module *Spectrum*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Tearardown method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

pymcxray.FileFormat.Results.test_Tags module

Tests for the modules *Tags*.

class pymcxray.FileFormat.Results.test_Tags.**TestTags** (*methodName='runTest'*)

Bases: `unittest.case.TestCase`

TestCase class for the module *Tags*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

getLines()

setUp()

Setup method.

tearDown()

Tearardown method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_findTag()

Tests for method *findTag*.

pymcxray.FileFormat.Results.test_XrayIntensities module

Tests for the module *XrayIntensities*.

class pymcxray.FileFormat.Results.test_XrayIntensities.**TestXrayIntensities** (*methodName='runTest'*)

Bases: unittest.case.TestCase

TestCase class for the module *XrayIntensities*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp ()

Setup method.

tearDown ()

Tearardown method.

testSkeleton ()

First test to check if the testcase is working with the testing framework.

test_read ()

Tests for method *read*.

pymcxray.FileFormat.Results.test_XraySimulatedSpectraRegion module

description

class pymcxray.FileFormat.Results.test_XraySimulatedSpectraRegion.**TestXraySimulatedSpectraRegion**

Bases: unittest.case.TestCase

TestCase class for the module *moduleName*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp ()

Setup method.

tearDown ()

Tearardown method.

testSkeleton ()

First test to check if the testcase is working with the testing framework.

test_read ()

Tests for method *read*.

pymcxray.FileFormat.Results.test_XraySimulatedSpectraSpecimen module

class pymcxray.FileFormat.Results.test_XraySimulatedSpectraSpecimen.**TestXraySimulatedSpectraSpecimen**

Bases: unittest.case.TestCase

TestCase class for the module *XraySimulatedSpectraSpecimen*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp ()

Setup method.

tearDown()

Teardown method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_read()

Tests for method *read*.

pymcxray.FileFormat.Results.test_XraySpectraAtomEmittedDetectedLines module

Tests for the module *XraySpectraAtomEmittedDetectedLines*.

class pymcxray.FileFormat.Results.test_XraySpectraAtomEmittedDetectedLines.**TestXraySpectraAtomEmittedDetectedLines**

Bases: unittest.case.TestCase

TestCase class for the module *XraySpectraAtomEmittedDetectedLines*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Teardown method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_read()

Tests for method *read*.

pymcxray.FileFormat.Results.test_XraySpectraRegionEmitted module

pymcxray.FileFormat.Results.test_XraySpectraRegionsEmitted module

pymcxray.FileFormat.Results.test_XraySpectraSpecimen module

Tests for the module *XraySpectraSpecimen*.

class pymcxray.FileFormat.Results.test_XraySpectraSpecimen.**TestXraySpectraSpecimen** (*methodName*)

Bases: unittest.case.TestCase

TestCase class for the module *XraySpectraSpecimen*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Teardown method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_read()
Tests for method *read*.

pymcxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetected module

Tests for the module *XraySpectraSpecimenEmittedDetected*.

class pymcxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetected.**TestXraySpectraSpecimenEmittedDetected**
Bases: unittest.case.TestCase

TestCase class for the module *XraySpectraSpecimenEmittedDetected*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_read()
Tests for method *read*.

pymcxray.FileFormat.Results.tests module

Module contents

Submodules

pymcxray.FileFormat.Element module

MCXRay element input file.

class pymcxray.FileFormat.Element.**Element** (*atomicNumber=0, massFraction=1.0*)
Bases: object

atomicNumber

createLineOldVersion()

createLinesWithKey()

extractFromLineOldVersion (*line*)

extractFromLinesWithKey (*lines*)

massFraction

name

pymcxray.FileFormat.ExportedSpectrum module

Read and write exported spectrum from McXRay.

```
class pymcxray.FileFormat.ExportedSpectrum.ExportedSpectrum
    Bases: object

    getData()

    getSpectrumType()

    read(filepath)
```

pymcxray.FileFormat.FileReaderWriterTools module

description

```
pymcxray.FileFormat.FileReaderWriterTools.reduceAfterDot(value)
```

pymcxray.FileFormat.MCXRayModel module

Model type used in MCXRay.

```
class pymcxray.FileFormat.MCXRayModel.AtomCollisionModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BROWNING = 1

    TYPE_GAUVIN = 2

    TYPE_RUTHERFORD = 0

class pymcxray.FileFormat.MCXRayModel.AtomCollisionScreeningModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_HENOC_MAUURICE = 0

class pymcxray.FileFormat.MCXRayModel.AtomCrossSectionModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BROWNING = 0

    TYPE_GAUVIN_DROUIN = 1

class pymcxray.FileFormat.MCXRayModel.AtomCrossSectionScreeningModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_HENOC_MAUURICE = 0

class pymcxray.FileFormat.MCXRayModel.AtomElectronRangeModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_KANAYA_OKAYAMA = 0

class pymcxray.FileFormat.MCXRayModel.AtomEnergyLossModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BETHE = 0

class pymcxray.FileFormat.MCXRayModel.AtomMeanIonizationPotentialModel(currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_JOY_LUO = 0
```

```
class pymcxray.FileFormat.MCXRayModel.AtomScreeningModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_HENOC_MAUURICE = 0

class pymcxray.FileFormat.MCXRayModel.MCXRayModel (currentModel=None)
    Bases: object

    getModel()

    setModel(modelType)

    setModelFromString(text)

class pymcxray.FileFormat.MCXRayModel.MassAbsorptionCoefficientModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_CHANTLER2005 = 3

    TYPE_HEINRICH_DATA = 1

    TYPE_HEINRICH_PARAMETERIZATION = 2

    TYPE_HENKE = 0

class pymcxray.FileFormat.MCXRayModel.RegionEnergyLossModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BETHE = 1

    TYPE_BETHE_JOY_LUO = 0

    TYPE_BETHE_RELATIVISTIC = 2

    TYPE_JOY_LUO_KGAUVIN = 3

    TYPE_JOY_LUO_MONSEL = 4

class pymcxray.FileFormat.MCXRayModel.SampleEnergyLossModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BETHE_JOY_LUO = 0

class pymcxray.FileFormat.MCXRayModel.SpectrumInterpolationModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_COPY = 0

    TYPE_LINEAR = 1

    TYPE_LINEAR_DOUBLE = 2

    TYPE_SPLINE = 3

    TYPE_SPLINE_BATCH = 4

    TYPE_SPLINE_POINT = 5

class pymcxray.FileFormat.MCXRayModel.XRayCSBremsstrahlungModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BETHE_HEITLER = 0

    TYPE_DING = 2

    TYPE_GAUVIN = 3

    TYPE_KIRKPATRICK_WIEDMAN = 1
```

```
class pymcxray.FileFormat.MCXRayModel.XRayCSCharacteristicModel (currentModel=None)
    Bases: pymcxray.FileFormat.MCXRayModel.MCXRayModel

    TYPE_BOTE2009 = 1
    TYPE_CASTANI1982 = 0
```

pymcxray.FileFormat.MicroscopeParameters module

MCXRay microscope parameters input file.

```
class pymcxray.FileFormat.MicroscopeParameters.MicroscopeParameters
    Bases: object

    beamCurrent_A
    beamDiameter_A
    beamEnergy_keV
    beamPositionX_A
    beamPositionY_A
    beamStandardDeviation_A
    beamTilt_deg
    defaultValues()
    detectorAzimuthalAngle_deg
    detectorBFHigh_rad
    detectorBFLow_rad
    detectorChannelWidth_eV
    detectorCrystalAtomSymbol
    detectorCrystalDistance_cm
    detectorCrystalRadius_cm
    detectorCrystalThickness_cm
    detectorDFHigh_rad
    detectorDFLow_rad
    detectorDeadLayer_A
    detectorDiffusionLenght_A
    detectorHAADFHigh_rad
    detectorHAADFLow_rad
    detectorNoise_eV
    detectorPitch_deg
    detectorSurfaceQuality
    detectorTOA_deg
    read(filepath)
```



```
time_s
version
write (filepath)
```

pymcrray.FileFormat.Models module

MCXRay models file.

```
class pymcrray.FileFormat.Models.Models
    Bases: object
    getModelList ()
    modelAtomCollision
    modelAtomCrossSection
    modelAtomMac
    modelSampleEnergyLoss
    modelXrayBremsstrahlung
    modelXrayCharacteristic
    read (filepath)
    version
    write (filepath)
```

pymcrray.FileFormat.Region module

MCXRay region input file.

```
class pymcrray.FileFormat.Region.Region
    Bases: object
    clear ()
    createLinesWithVersion ()
    createLinesWithoutVersion ()
    extractFromLinesWithVersion (lines)
    extractFromLinesWithoutVersion (lines)
```

pymcrray.FileFormat.RegionDimensions module

MCXRay region dimensions input file.

```
class pymcrray.FileFormat.RegionDimensions.RegionDimensions (parameters=None)
    Bases: object
    createLineOldVersion ()
    createLineWithKey ()
    extractFromLineOldVersion (line)
```

```
    extractFromLinesWithKey (line)
    getLineFormat ()

class pymcrray.FileFormat.RegionDimensions.RegionDimensionsBox (parameters=None)
    Bases: pymcrray.FileFormat.RegionDimensions.RegionDimensions
    getLineFormat ()
    maximumX
    maximumY
    maximumZ
    minimumX
    minimumY
    minimumZ

class pymcrray.FileFormat.RegionDimensions.RegionDimensionsCylinder (parameters=None)
    Bases: pymcrray.FileFormat.RegionDimensions.RegionDimensionsSphere
    directionX
    directionY
    directionZ
    getLineFormat ()
    length

class pymcrray.FileFormat.RegionDimensions.RegionDimensionsSphere (parameters=None)
    Bases: pymcrray.FileFormat.RegionDimensions.RegionDimensions
    getLineFormat ()
    positionX
    positionY
    positionZ
    radius

pymcrray.FileFormat.RegionDimensions.createRegionDimensions (regionType)
```

pymcrray.FileFormat.RegionType module

MCXRay region type input file.

pymcrray.FileFormat.ResultsParameters module

MCXRay ResultsParameters input file.

```
class pymcrray.FileFormat.ResultsParameters.ResultsParameters
    Bases: object
    defaultValues ()
    isComputeXrayBremsstrahlung
    isComputeXrayCharacteristic
```

```
isComputeXrayPhirhoz
isComputeXraySimulatedSpectrum
read (filepath)
version
write (filepath)
```

pymcxray.FileFormat.SimulationInputs module

MCXRay simulation inputs file.

```
class pymcxray.FileFormat.SimulationInputs.SimulationInputs
    Bases: object
    getExtension (key)
    mapFilename
    microscopeFilename
    modelFilename
    read (filepath)
    resultParametersFilename
    simulationParametersFilename
    snrFilename
    specimenFilename
    title
    version
    write (filepath)
```

pymcxray.FileFormat.SimulationParameters module

MCXRay simulation parameters input file.

```
class pymcxray.FileFormat.SimulationParameters.SimulationParameters
    Bases: object
    baseFilename
    defaultValues ()
    elasticCrossSectionScalingFactor
    energyChannelWidth_eV
    energyLossScalingFactor
    numberChannels
    numberElectrons
    numberFilmsX
    numberFilmsY
```

```
numberFilmsZ
numberPhotons
numberWindows
read (filepath)
spectrumInterpolationModel
version
voxelSimplification
write (filepath)
```

pymcxray.FileFormat.SnrParameters module

MCXRay Snr input file.

```
class pymcxray.FileFormat.SnrParameters.SnrParameters
    Bases: object

    backgroundEnergyWindowsSize
    defaultValues ()
    energyEnd_keV
    energyStart_keV
    numberEnergySteps
    read (filepath)
    snrType
    spectrumEnergyWindowsSize
    write (filepath)
```

pymcxray.FileFormat.Specimen module

MCXRay specimen input file.

```
class pymcxray.FileFormat.Specimen.Specimen
    Bases: object

    clear ()
    name
    numberRegions
    read (filepath)
    regions
    version
    write (filepath)
```

pymcrray.FileFormat.Version module

MCXray version information.

```
class pymcrray.FileFormat.Version.Version (major, minor, revision)
    Bases: object

    fromString (versionString)

    key = 'Version'

    major

    minor

    readFromFile (filepath)

    revision

    toString ()

    writeLine (outputFile)
```

pymcrray.FileFormat.testUtilities module

Utilities used for testing this package.

```
pymcrray.FileFormat.testUtilities.createTempDataPath (path)
pymcrray.FileFormat.testUtilities.getSimulationTitles ()
pymcrray.FileFormat.testUtilities.removeTempDataPath (path)
```

pymcrray.FileFormat.test_Element module

Tests for module *Element*.

```
class pymcrray.FileFormat.test_Element.TestElement (methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module Element.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp ()
        Setup method.

    tearDown ()
        Teardown method.

    testSkeleton ()
        First test to check if the testcase is working with the testing framework.

    test_createLineOldVersion ()
        Tests for method createLineOldVersion.

    test_createLineWithKey ()
        Tests for method createLineOldVersion.

    test_extractFromLineOldVersion ()
        Tests for method extractFromLineOldVersion.
```

test_extractFromLineWithKey()
Tests for method *extractFromLinesWithKey*.

pymcrray.FileFormat.test_ExportedSpectrum module

Tests for module *ExportedSpectrum*.

class pymcrray.FileFormat.test_ExportedSpectrum.**TestExportedSpectrum** (*methodName='runTest'*)
Bases: unittest.case.TestCase
TestCase class for the module *moduleName*.
Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.
setUp()
Setup method.
tearDown()
Teardown method.
testSkeleton()
First test to check if the testcase is working with the testing framework.
test_read()
Tests for method *read*.

pymcrray.FileFormat.test_FileReaderWriterTools module

Tests for the module *FileReaderWriterTools*.

class pymcrray.FileFormat.test_FileReaderWriterTools.**TestFileReaderWriterTools** (*methodName='runTest'*)
Bases: unittest.case.TestCase
TestCase class for the module *FileReaderWriterTools*.
Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.
setUp()
Setup method.
tearDown()
Teardown method.
testSkeleton()
First test to check if the testcase is working with the testing framework.
test__reduceAfterDot()
Tests for method *reduceAfterDot*.

pymcrray.FileFormat.test_MCXRayModel module

Tests for module *MCXRayModel*.

class pymcrray.FileFormat.test_MCXRayModel.**TestMCXRayModel** (*methodName='runTest'*)
Bases: unittest.case.TestCase
TestCase class for the module *MCXRayModel*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_MCXRayModel()
Tests for method *MCXRayModel*.

pymcxray.FileFormat.test_MicroscopeParameters module

pymcxray.FileFormat.test_Models module

pymcxray.FileFormat.test_Region module

Tests for module *Region*.

class `pymcxray.FileFormat.test_Region.TestRegion (methodName='runTest')`
Bases: `unittest.case.TestCase`

TestCase class for the module *moduleName*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

getTestRegionLinesWithVersion (title, userMassDensity=True)

getTestRegionLinesWithoutVersion (title, userMassDensity=True)

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

test_createLinesWithVersion()
Tests for method *createLinesWithVersion*.

test_createLinesWithoutVersion()
Tests for method *createLinesWithoutVersion*.

test_extractFromLinesWithVersion()
Tests for method *read*.

test_extractFromLinesWithoutVersion()
Tests for method *read*.

pymcxray.FileFormat.test_RegionDimensions module

Tests for module *RegionDimensions*.

```
class pymcxray.FileFormat.test_RegionDimensions.TestRegionDimensions (methodName='runTest')
```

Bases: `unittest.case.TestCase`

TestCase class for the module *RegionDimensions*.

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

```
setUp ()
```

Setup method.

```
tearDown ()
```

Teardown method.

```
testSkeleton ()
```

First test to check if the testcase is working with the testing framework.

```
test_RegionDimensionsBox_createLineOldVersion ()
```

Tests for class *RegionDimensionsBox*.

```
test_RegionDimensionsBox_createLineWithKey ()
```

Tests for class *RegionDimensionsBox*.

```
test_RegionDimensionsBox_extractFromLineOldVersion ()
```

Tests for class *RegionDimensionsBox*.

```
test_RegionDimensionsBox_extractFromLinesWithKey ()
```

Tests for class *RegionDimensionsBox*.

```
test_RegionDimensionsCylinder_createLineOldVersion ()
```

Tests for class *RegionDimensionsCylinder*.

```
test_RegionDimensionsCylinder_createLineWithKey ()
```

Tests for class *RegionDimensionsCylinder*.

```
test_RegionDimensionsCylinder_extractFromLineOldVersion ()
```

Tests for class *RegionDimensionsCylinder*.

```
test_RegionDimensionsCylinder_extractFromLinesWithKey ()
```

Tests for class *RegionDimensionsCylinder*.

```
test_RegionDimensionsSphere_createLineOldVersion ()
```

Tests for class *RegionDimensionsSphere*.

```
test_RegionDimensionsSphere_createLineWithKey ()
```

Tests for class *RegionDimensionsSphere*.

```
test_RegionDimensionsSphere_extractFromLineOldVersion ()
```

Tests for class *RegionDimensionsSphere*.

```
test_RegionDimensionsSphere_extractFromLinesWithKey ()
```

Tests for class *RegionDimensionsSphere*.

```
test_createRegionDimensions ()
```

Tests for method *createRegionDimensions*.

pymcxray.FileFormat.test_RegionType module

Test for module *RegionType*.

```
class pymcxray.FileFormat.test_RegionType.TestRegionType (methodName='runTest')
```

Bases: `unittest.case.TestCase`

TestCase class for the module *RegionType*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

Setup method.

tearDown()

Tear down method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test_Constants()

Tests for method *Constants*.

pymcxray.FileFormat.test_ResultsParameters module

pymcxray.FileFormat.test_SimulationInputs module

pymcxray.FileFormat.test_SimulationParameters module

pymcxray.FileFormat.test_SnrParameters module

Tests for the module *SnrParameters*.

class pymcxray.FileFormat.test_SnrParameters.**TestSnrParameters** (*methodName='runTest'*)
Bases: unittest.case.TestCase

TestCase class for the module *SnrParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

getSnrParametersReference (*title*)

setUp()

Setup method.

tearDown()

Tear down method.

testSkeleton()

First test to check if the testcase is working with the testing framework.

test__createKeys()

Tests for method *_createKeys*.

test_read()

Tests for method *read*.

test_write()

Tests for method *write*.

pymcxray.FileFormat.test_Specimen module

pymcrray.FileFormat.test_Version module

Tests for module *Version*.

```
class pymcrray.FileFormat.test_Version.TestVersion (methodName='runTest')
    Bases: unittest.case.TestCase

    TestCase class for the module Version.

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp ()
        Setup method.

    tearDown ()
        Teardown method.

    testSkeleton ()
        First test to check if the testcase is working with the testing framework.

    test_VersionConstants ()
        Tests for method VersionConstants.

    test_comparison ()
        Test comparison operation on Version class.

    test_fromString ()
        Tests for method fromString.

    test_readFromFile ()
        Tests for method readFromFile.

    test_readFromFile_BadFile ()
        Tests for method readFromFile.

    test_toString ()
        Tests for method toString.

    test_writeLine ()
        Tests for method writeLine.
```

pymcrray.FileFormat.tests module

Module contents

pymcrray.serialization package

Submodules

pymcrray.serialization.SerializationH5py module

```
class pymcrray.serialization.SerializationH5py.SerializationNumpy (filename=None,
                                                                    ver-
                                                                   bose=True)

    Bases: pymcrray.serialization._Serialization._Serialization

    load ()
```

save (*serializedData*)

pymcxray.serialization.SerializationNumpy module

```
class pymcxray.serialization.SerializationNumpy.SerializationNumpy (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization._Serialization._Serialization  
    load()  
    save (serializedData)  
  
class pymcxray.serialization.SerializationNumpy.SerializationNumpyNPY (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization.SerializationNumpy.SerializationNumpy  
    load()  
    save (data)  
  
class pymcxray.serialization.SerializationNumpy.SerializationNumpyNPZ (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization.SerializationNumpy.SerializationNumpy  
    load()  
    save (data)  
  
class pymcxray.serialization.SerializationNumpy.SerializationNumpyTxt (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization.SerializationNumpy.SerializationNumpy  
    load()  
    save (data)  
  
class pymcxray.serialization.SerializationNumpy.SerializationNumpyTxtGz (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization.SerializationNumpy.SerializationNumpyTxt
```

pymcxray.serialization.SerializationPickle module

```
class pymcxray.serialization.SerializationPickle.SerializationPickle (filename=None,  
                                                                    ver-  
                                                                    bose=True)  
  
    Bases: pymcxray.serialization._Serialization._Serialization  
    KEY_FILE_VERSION = 'fileVersion'  
    KEY_SERIALIZED_DATA = 'serializedData'  
    load()  
    save (serializedData)
```

pymcxray.serialization.test_Serialization module

pymcxray.serialization.test_SerializationH5py module

```
class pymcxray.serialization.test_SerializationH5py.TestSerializationH5py (methodName='runTest')
    Bases: unittest.case.TestCase

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp()
        Hook method for setting up the test fixture before exercising it.

    tearDown()
        Hook method for deconstructing the test fixture after testing it.

    testSkeleton()
```

pymcxray.serialization.test_SerializationNumpy module

```
class pymcxray.serialization.test_SerializationNumpy.TestSerializationNumpy (methodName='runTest')
    Bases: unittest.case.TestCase

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp()
        Hook method for setting up the test fixture before exercising it.

    tearDown()
        Hook method for deconstructing the test fixture after testing it.

    testSkeleton()

    test_loadSaveSerializationNumpy()

    test_loadSaveSerializationNumpyNPY()

    test_loadSaveSerializationNumpyNPZ()

    test_loadSaveSerializationNumpyTxt()

    test_loadSaveSerializationNumpyTxtGz()
```

pymcxray.serialization.test_SerializationPickle module

```
class pymcxray.serialization.test_SerializationPickle.TestSerialization (methodName='runTest')
    Bases: unittest.case.TestCase

    Create an instance of the class that will use the named test method when executed. Raises a ValueError if the
    instance does not have a method with the specified name.

    setUp()
        Hook method for setting up the test fixture before exercising it.

    tearDown()
        Hook method for deconstructing the test fixture after testing it.

    testSkeleton()
```

```
test_loadSave()
```

```
test_version()
```

pymcxray.serialization.tests module

Regression testing for the project.

Module contents

pymcxray.tests package

Submodules

pymcxray.tests.test_pymcxray module

test_pymcxray

Tests for *pymcxray* module.

```
class pymcxray.tests.test_pymcxray.TestPymcxray (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

Create an instance of the class that will use the named test method when executed. Raises a `ValueError` if the instance does not have a method with the specified name.

```
    setUp()
```

Hook method for setting up the test fixture before exercising it.

```
    tearDown()
```

Hook method for deconstructing the test fixture after testing it.

```
    test_000_something()
```

Module contents

8.1.2 Submodules

8.1.3 pymcxray.AnalyzeNumberBackgroundWindows module

Analyze the number of background windows on the x-ray spectrum.

```
class pymcxray.AnalyzeNumberBackgroundWindows.AnalyzeNumberBackgroundWindows
```

```
    Bases: object
```

```
    plotData()
```

```
    plotDifference()
```

```
    readData()
```

```
pymcxray.AnalyzeNumberBackgroundWindows.run()
```

8.1.4 pymcxcray.AtomData module

MCXRay atom data.

```
pymcxcray.AtomData.getAtomSymbol(atomic_number)
pymcxcray.AtomData.getAtomicNumber(symbol)
pymcxcray.AtomData.getIonizationEnergy_keV(shell, element)
pymcxcray.AtomData.getMassDensity_g_cm3(symbol)
pymcxcray.AtomData.getShellList()
pymcxcray.AtomData.getXRayEnergy_keV(line, element)
pymcxcray.AtomData.get_atomic_weight_g_mol(atomic_number)
pymcxcray.AtomData.run()
```

8.1.5 pymcxcray.BatchFile module

MCXRay batch file creator.

```
class pymcxcray.BatchFile.BatchFile(name, numberFiles=1)
    Bases: object
    addSimulationName(simulationFilename)
    removePreviousFiles(path)
    write(path)
```

8.1.6 pymcxcray.BatchFileConsole module

MCXRay console batch file creator.

```
class pymcxcray.BatchFileConsole.BatchFileConsole(name, programName, number-
                                                    Files=1)
    Bases: object
```

The batch file is responsible to create the simulation structure with a copy of mcxcray program.

One important parameter to set is the *numberFiles*, this is the number of batch files generated and that can be run in parallel. For maximum efficiency it should be set as the number of logical processors minus 1 or 2. For example, on a computer with 12 logical processors, the *numberFiles* should be set at 10.

Parameters

- **name** (*str*) – Basename used for the batch files
- **programName** (*str*) – Name of the executable to add in the batch file
- **numberFiles** (*int*) – Number of batch files to generate and possibly to run in parallel

```
addSimulationName(simulationFilename)
```

Add a simulation in the simulation list.

Parameters **simulationFilename** (*str*) – File path of the simulation added

```
write(path)
```

Write the batch files for all simulations in the simulation list.

Parameters **path** (*str*) – Path where the batch files are written.

8.1.7 pymcrray.ComparisonModels module

Comparison of the models used by MCXray.

```
class pymcrray.ComparisonModels.ComparisonModels (dataPath)
    Bases: object

    graphicsEnergyLoss ()
    graphicsIonizationCrossSection ()
    graphicsXrayCrossSectionBremstrahlung ()
    graphicsXrayMassAbsorptionCoefficient ()
pymcrray.ComparisonModels.runVersion1_2_3 ()
pymcrray.ComparisonModels.runVersion1_4_0 ()
pymcrray.ComparisonModels.runVersion1_4_1 ()
```

8.1.8 pymcrray.DebugSimulatedSpectrum module

Debug the simulated spectrum implementation in mcxray.

```
class pymcrray.DebugSimulatedSpectrum.DebugSimulatedSpectrum
    Bases: object

    runRegion ()
    runSpecimen ()
pymcrray.DebugSimulatedSpectrum.run ()
```

8.1.9 pymcrray.ElementProperties module

```
pymcrray.ElementProperties.computeAtomicDensity_atom_cm3 (massDensity_g_cm3,
                                                         atomicMass_g_mol)
```

Compute the atomic density.

$$n_i = \frac{N_A \rho_i}{A_i}$$

where

- n_i is the atomic density in atoms/cm^3
- N_A is the Avogadro number in atoms/mole
- ρ_i is the mass density in g/cm^3
- A_i is the atomic mass in g/mole

Parameters

- **massDensity_g_cm3** (*float*) –
- **atomicMass_g_mol** (*float*) –

For element H to Lr (1-103). From: CASINO source code, DOS version.

Todo: Add units.

Atomic weight of element in atomic number order.

For element H to Sg (1-106).

Unit $g/mole$.

From: Tableau periodique des elements, Sargent-Welch scientifique Canada Limitee.

Fermi wavelength of element in atomic number order.

For element H to Lr (1–103). From: CASINO source code, DOS version.

Todo: Add units.

Mass density of element in atomic number order.

For element H to Cm (1-96).

In g/cm^3 .

From: Tableau periodique des elements, Sargent-Welch scientifique Canada Limitee.

Note: Element Z = 85 and 87 set to 1 for the calculation.

Plasmon energy of element in atomic number order.

For element H to Lr (1-103). From: CASINO source code, DOS version.

Todo: Add units.

`pymcxray.ElementProperties.getAtomicNumber` (*atomicNumber=None, name=None, symbol=None*)

`pymcxray.ElementProperties.getAtomicNumberByName` (*name*)

`pymcxray.ElementProperties.getAtomicNumberBySymbol` (*symbol*)

`pymcxray.ElementProperties.getFermiEnergy_eV(atomicNumber)`

`pymcxray.ElementProperties.getKFermi_eV(atomicNumber)`

`pymcxray.ElementProperties.getKRatioCorrection(atomicNumber)`

Get the constant k ratio correction needed by the mean ionization potential from the atomic number.

Parameters `atomic number` (*int*) – Atomic number


```

pymcrray.ElementProperties.getKRatioCorrectionMonsel (atomicNumber, workFunc-
                                                    tion_keV)
    /// K value as defined by Monsel. /// Used in DE/DS calculation. Casino uses K Gauvin, but for low energy, ///
    JR Lowney says that this one is more appropriate (and by experience, /// it is effectively better for the secondary
    yield). /// <p> NOTE : Depends on J (ionisation potential). So it must already be calculated before. /// @param
    element Element for whom we want to calculate the K value. /// @return The K value of the element passed in
    argument

pymcrray.ElementProperties.getMassDensity_g_cm3 (atomicNumber)

pymcrray.ElementProperties.getMeanIonizationEnergy_eV (atomic_number)
    Get the mean ionization potential from the atomic number.

    In eV.

    Parameters atomic_number (int) – Atomic number

pymcrray.ElementProperties.getName (atomicNumber)

pymcrray.ElementProperties.getPlasmonEnergy_eV (atomicNumber)

pymcrray.ElementProperties.getSymbol (atomicNumber)

pymcrray.ElementProperties.run ()

pymcrray.ElementProperties.runAtomicNumberSymbol ()

```

8.1.10 pymcrray.Simulation module

MCXRay simulation parameters.

```

class pymcrray.Simulation.Layer (elements, thickness_nm, mass_density_g_cm3=None)
    Bases: object

```

```

class pymcrray.Simulation.Simulation (overwrite=True)
    Bases: object

```

basename

beamDiameter_nm

beamPosition_nm

beamTilt_deg

createSimulationFiles (*path, simulationPath, hdf5_group*)

current_A

detectorAzimuthalAngle_deg

detectorChannelWidth_eV

detectorCrystalDistance_cm

detectorCrystalRadius_cm

detectorCrystalThickness_cm

detectorNoise_eV

elasticCrossSectionScalingFactor

energyLossScalingFactor

energy_keV

```
filename
generateBaseFilename ()
getFilenameSuffixes ()
getParameters ()
getProgramVersionFilepath (simulationPath)
isDone (simulationPath, hdf5_group=None)
modelXrayBremsstrahlung
name
numberContinuumWindows
numberElectrons
numberEnergyWindows
numberLayersX
numberLayersY
numberLayersZ
numberPhotons
removeInputsFiles ()
resultsBasename
setParameters (parameters)
solidAngle_sr
spectrumInterpolationModel
takeOffAngle_deg
time_s

pymcrray.Simulation.computeWeightFraction (atomicNumberRef, atomicWeights)
pymcrray.Simulation.createAlloyBoxInSubstrate (elementsParticle, elementsSubstrate, box-
Parameters_nm)
pymcrray.Simulation.createAlloyBoxInThinFilm (elementsParticle, atomicNumberSubstrate,
boxParameters_nm, filmThickness_nm)
pymcrray.Simulation.createAlloyBoxInVacuum (elements, boxParameters_nm)
pymcrray.Simulation.createAlloyBulkSample (elements, sampleName=None)
pymcrray.Simulation.createAlloyFilmOverSubstrate (film_elements, substrate_elements,
film_thickness_nm=10.0,
film_mass_density_g_cm3=None,
substrate_mass_density_g_cm3=None)
pymcrray.Simulation.createAlloyMultiVerticalLayer (elementsLayers, layerWidths_nm)
pymcrray.Simulation.createAlloyParticleInSubstrate (elementsParticle, atomicNum-
berSubstrate, particleRadius_nm,
particlePositionZ_nm=None)
```

```

pymcxray.Simulation.createAlloyParticleInThinFilm(elementsParticle, atomicNumberSubstrate, particleRadius_nm, filmThickness_nm, particlePositionZ_nm=None)

pymcxray.Simulation.createAlloyThinFilm(elements, filmThickness_nm)

pymcxray.Simulation.createAlloyThinFilm2(elements, filmThickness_nm)

pymcxray.Simulation.createBoxFeatureInSubstrate(feature_elements, substrate_elements, depth_nm, width_nm)

pymcxray.Simulation.createFilmInSubstrate(atomicNumberFilm, atomicNumberSubstrate, filmThickness_nm, filmTopPositionZ_nm)

pymcxray.Simulation.createFilmOverSubstrate(atomicNumberFilm, atomicNumberSubstrate, filmThickness_nm=10.0)

pymcxray.Simulation.createParticleInSubstrate(atomicNumberParticle, atomicNumberSubstrate, particleRadius_nm, particlePositionZ_nm=None)

pymcxray.Simulation.createParticleOnFilm(atomicNumberParticle, atomicNumberSubstrate, particleDiameter_nm, filmThickness_nm)

pymcxray.Simulation.createParticleOnSubstrate(atomicNumberParticle, atomicNumberSubstrate, particleDiameter_nm)

pymcxray.Simulation.createPhirhozSpecimens(atomicNumberTracer, atomicNumberMatrix, tracerThickness_nm, maximumThickness_nm)

pymcxray.Simulation.createPureBulkSample(atomicNumber)

pymcxray.Simulation.create_cnt_sample(body_elements, cnt_length_nm=1000.0, cnt_outside_diameter_nm=100.0, cnt_inside_diameter_nm=50.0, particle_diameter_nm=5.0)

pymcxray.Simulation.create_multi_horizontal_layer(substrate_elements, layers, substrate_mass_density_g_cm3=None)

```

Create a horizontal multi layer sample.

The substrate is the first region created. The other region are each of the element in the *layers*.

Parameters

- **substrate_elements** – list of atomic number and weight fraction pair for the composition of the substrate
- **layers** –
- **substrate_mass_density_g_cm3** –

Returns

```

pymcxray.Simulation.create_weight_fractions(weight_fraction_step, number_elements)

```

8.1.11 pymcxray.SimulationsParameters module

Simulations parameters

```

class pymcxray.SimulationsParameters.SimulationsParameters
    Bases: dict
    addCompute(parameterKey)

```

```
addFixed (parameterKey, value)
addVaried (parameterKey, values)
computeNumberXrays (experiment)
computeNumberXraysFilepath
fixedParameters
getAllSimulationParameters ()
getVariedParameterLabels ()
variedParameters

class pymcrray.SimulationsParameters.SimulationsParametersFixed
    Bases: object
    addExperiment (experiment)
    getAllSimulationParameters ()
```

8.1.12 pymcrray.Testings module

8.1.13 pymcrray.mcxray module

Base module to create and analuyze MCXRay simulations.

8.1.14 pymcrray.multipleloop module

This module provides a tool for handling computer experiments with of a set of input parameters, where each input parameter is varied in a prescribed fashion.

In short, the parameters are held in a dictionary where the keys are the names of the parameters and the values are the numerical, string or other values of the parameters. The value can take on multiple values: e.g., an integer parameter ‘a’ can have values -1, 1 and 10. Similarly, a string parameter ‘method’ can have values ‘Newton’ and ‘Bisection’. The module will generate all combination of all parameters and values, which in the mentioned example will be (-1, ‘Newton’), (1, ‘Newton’), (10, ‘Newton’), (-1, ‘Bisection’), (1, ‘Bisection’), and (10, ‘Bisection’). Particular combination of values can easily be removed.

The usage and implementation of the module are documented in the book “Python Scripting for Computational Science” (H. P. Langtangen, Springer, 2009), Chapter 12.1.

```
pymcrray.multipleloop.combine (prm_values)
```

Compute the combination of all parameter values in the prm_values (nested) list. Main function in this module.

param prm_values: nested list (parameter_name, list_of_parameter_values) or dictionary
prm_values[parameter_name] = list_of_parameter_values. return: (all, names, varied)
where

- all contains all combinations (experiments) all[i] is the list of individual parameter values in experiment no i
- names contains a list of all parameter names
- varied holds a list of parameter names that are varied (i.e. where there is more than one value of the parameter, the rest of the parameters have fixed values)

Code example:

```

>>> dx = array([1.0/2**k for k in range(2,5)])
>>> dt = 3*dx; dt = dt[:-1]
>>> p = {'dx': dx, 'dt': dt}
>>> p
{'dt': [ 0.75 , 0.375,], 'dx': [ 0.25 , 0.125 , 0.0625,]}
>>> all, names, varied = combine(p)
>>> all
[[0.75, 0.25], [0.375, 0.25], [0.75, 0.125], [0.375, 0.125],
 [0.75, 0.0625], [0.375, 0.0625]]

```

8.1.15 pymcxray.pymcxray module

8.1.16 pymcxray.test_AtomData module

Tests for the module *AtomData*.

class pymcxray.test_AtomData.**TestAtomData** (*methodName='runTest'*)

Bases: unittest.case.TestCase

TestCase class for the module *AtomData*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp ()

Setup method.

tearDown ()

Teardown method.

testConstants ()

First test to check if the testcase is working with the testing framework.

testSkeleton ()

First test to check if the testcase is working with the testing framework.

8.1.17 pymcxray.test_BatchFileConsole module

Tests for the module *BatchFileConsole*.

class pymcxray.test_BatchFileConsole.**TestBatchFileConsole** (*methodName='runTest'*)

Bases: unittest.case.TestCase

TestCase class for the module *BatchFileConsole*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp ()

Setup method.

tearDown ()

Teardown method.

testSkeleton ()

First test to check if the testcase is working with the testing framework.

8.1.18 pymcxdray.test_ComparisonModels module

Tests for the module *ComparisonModels*.

```
class pymcxdray.test_ComparisonModels.TestComparisonModels (methodName='runTest')  
    Bases: unittest.case.TestCase
```

TestCase class for the module *ComparisonModels*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()  
    Setup method.
```

```
tearDown ()  
    Teardown method.
```

```
testSkeleton ()  
    First test to check if the testcase is working with the testing framework.
```

8.1.19 pymcxdray.test_Simulation module

Tests for the module *Simulation*.

```
class pymcxdray.test_Simulation.TestSimulation (methodName='runTest')  
    Bases: unittest.case.TestCase
```

TestCase class for the module *Simulation*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

```
setUp ()  
    Setup method.
```

```
tearDown ()  
    Teardown method.
```

```
testSkeleton ()  
    First test to check if the testcase is working with the testing framework.
```

```
test_create_multi_horizontal_layer ()  
    Test the create_multi_horizontal_layer method.
```

```
test_create_weight_fractions ()  
    First test to check if the testcase is working with the testing framework.
```

8.1.20 pymcxdray.test_SimulationsParameters module

Tests for the module *SimulationsParameters*.

```
class pymcxdray.test_SimulationsParameters.TestSimulationsParameters (methodName='runTest')  
    Bases: unittest.case.TestCase
```

TestCase class for the module *SimulationsParameters*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

8.1.21 pymcxray.test_mcxray module

Tests for the module *mcxray*.

class pymcxray.test_mcxray.**Testmcxray** (*methodName='runTest'*)
Bases: unittest.case.TestCase
TestCase class for the module *mcxray*.

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()
Setup method.

tearDown()
Teardown method.

testSkeleton()
First test to check if the testcase is working with the testing framework.

8.1.22 pymcxray.tests module

8.1.23 Module contents

pymcxray.**create_path** (*path*)
Create a path from the input string if does not exists.

Does not try to distinct between file and directory in the input string. *path* = "dir1/filename.ext" => "dir1/filename.ext/" where the new directory "filename.ext" is created.

Parameters *path* (*str*) – The path input string.

Returns The path with the path separator at the end

Return type *str*

pymcxray.**find_all_files** (*root*, *patterns='**', *ignore_path_patterns=''*, *ignore_name_patterns=''*, *single_level=False*, *yield_folders=False*)

Find all files in a root folder.

From Python Cookbook section 2.16 pages 88–90

Parameters

- **root** –
- **patterns** –
- **ignore_path_patterns** –
- **ignore_name_patterns** –

- **single_level** –
- **yield_folders** –

Returns

`pymcxray.get_current_module_path(module_path, relative_path=)`

Extract the current module path and combine it with the relative path and return it.

Parameters

- **module_path** (*str*) – Pass the `__FILE__` python keyword for this parameter
- **relative_path** (*str*) – The relative path to combine with the module path

Returns The path obtained when combine the module path and relative path

Return type `str`

`pymcxray.get_mcxray_archive_name(configuration_file_path, default=None)`

Read the MCXRay archive name in the configuration file. This option allows to choose which version of MCXRay to use for the simulations.

The configuration file need to have this entry in the section [Paths]:

```
[Paths]
mcxrayArchiveName=2016-04-11_11h41m28s_MCXRay_v1.6.6.0.zip
```

Parameters

- **configuration_file_path** (*str*) – The file path of the configuration file
- **default** (*str*) – Default value to use if the entry is not found

Returns The MCXRay archive name

Return type `str`

`pymcxray.get_mcxray_archive_path(configuration_file_path, relative_path=)`

Read the MCXRay archive path in the configuration file.

The configuration file need to have this entry in the section [Paths]:

```
[Paths]
mcxrayArchivePath=D:\Dropbox\hdemers\professional\softwareRelease\mcxray
```

Parameters

- **configuration_file_path** (*str*) – The file path of the configuration file
- **relative_path** (*str*) – Relative path to add to the path read in the configuration file

Returns Path where the mcxray archive can be found.

Return type `str`

`pymcxray.get_mcxray_program_name(configuration_file_path, default=None)`

Read the MCXRay program name in the configuration file.

This option specify which executable to use in the script. The `console_mcxray_x64.exe` should be OK for most situation. If you have a 32-bit system, you have to use `console_mcxray.exe` (32-bit version).

The configuration file need to have this entry in the section [Paths]:


```
[Paths]
mcxrayProgramName=console_mcxray_x64.exe
```

Parameters

- **configuration_file_path** (*str*) – The file path of the configuration file
- **default** (*str*) – Default value to use if the entry is not found

Returns The MCXRay program name**Return type** *str*

`pymcxray.get_mcxray_program_path(configuration_file_path, relative_path=)`

Read the MCXRay program path in the configuration file.

The configuration file need to have this entry in the section [Paths]:

```
[Paths]
mcxrayProgramPath=C:\hdemers\codings\devcasino
```

Parameters

- **configuration_file_path** (*str*) – The file path of the configuration file
- **relative_path** (*str*) – Relative path to add to the path read in the configuration file

Returns Path where the mcxray program can be found.**Return type** *str*

Deprecated since version 0.1.

See also:

function `pymcxray.get_mcxray_archive_path()`

`pymcxray.get_results_mcgill_path(configuration_file_path, relative_path=)`

Read the results path for McGill in the configuration file. The results path read in the configuration file is combine with the *relative_path* and return.

The configuration file need to have this entry in the section [Paths]:

```
[Paths]
resultsMcGillPath=D:\Dropbox\hdemers\professional\results\simulations
```

Parameters

- **configuration_file_path** (*str*) – The file path of the configuration file
- **relative_path** (*str*) – Relative path to add to the path read in the configuration file

Returns Path where the simulation input and results will be written**Return type** *str*

`pymcxray.read_value_from_configuration_file(configuration_file, section_name, key_name, default=None)`

Read a value from an entry in a section from a configuration file.

Parameters

- **configuration_file** (*str*) – The file path of the configuration file
- **section_name** (*str*) – Name of the section
- **key_name** (*str*) – Name of the entry to read
- **default** – Default value of the entry if not found

Returns The value read or default value

Return type `str`

CHAPTER 9

Indices and tables

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

Python Module Index

p

`pymcxray`, 83

`pymcxray.AnalyzeNumberBackgroundWindows`, 73

`pymcxray.AtomData`, 74

`pymcxray.BatchFile`, 74

`pymcxray.BatchFileConsole`, 74

`pymcxray.ComparisonModels`, 75

`pymcxray.DebugSimulatedSpectrum`, 75

`pymcxray.ElementProperties`, 75

`pymcxray.FileFormat`, 70

`pymcxray.FileFormat.Element`, 57

`pymcxray.FileFormat.ExportedSpectrum`, 58

`pymcxray.FileFormat.FileReaderWriterTools`, 58

`pymcxray.FileFormat.MCXRayModel`, 58

`pymcxray.FileFormat.MicroscopeParameters`, 60

`pymcxray.FileFormat.Models`, 61

`pymcxray.FileFormat.Region`, 61

`pymcxray.FileFormat.RegionDimensions`, 61

`pymcxray.FileFormat.RegionType`, 62

`pymcxray.FileFormat.Results`, 57

`pymcxray.FileFormat.Results.BaseResults`, 31

`pymcxray.FileFormat.Results.BeamParameters`, 31

`pymcxray.FileFormat.Results.DetectorParameters`, 31

`pymcxray.FileFormat.Results.Dump`, 32

`pymcxray.FileFormat.Results.ElectronExistenceResults`, 32

`pymcxray.FileFormat.Results.ElectronParameters`, 33

`pymcxray.FileFormat.Results.ElectronResults`, 33

`pymcxray.FileFormat.Results.ElectronTrajectoriesResults`, 34

`pymcxray.FileFormat.Results.ElementParameters`, 35

`pymcxray.FileFormat.Results.exported`, 31

`pymcxray.FileFormat.Results.exported.DataMap`, 29

`pymcxray.FileFormat.Results.exported.test_DataMap`, 30

`pymcxray.FileFormat.Results.exported.test_XrayIntensity`, 30

`pymcxray.FileFormat.Results.exported.XrayIntensity2`, 30

`pymcxray.FileFormat.Results.Intersections`, 35

`pymcxray.FileFormat.Results.MicroscopeParameters`, 35

`pymcxray.FileFormat.Results.ModelParameters`, 35

`pymcxray.FileFormat.Results.Phirhoz`, 36

`pymcxray.FileFormat.Results.PhirhozElement`, 36

`pymcxray.FileFormat.Results.PhirhozEmittedCharacteristics`, 36

`pymcxray.FileFormat.Results.PhirhozEmittedCharacteristics2`, 37

`pymcxray.FileFormat.Results.PhirhozGenerated`, 37

`pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristics`, 37

`pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristics2`, 38

`pymcxray.FileFormat.Results.PhirhozRegion`, 38

`pymcxray.FileFormat.Results.RegionParameters`, 38

`pymcxray.FileFormat.Results.RegionVolume`, 39

`pymcxray.FileFormat.Results.SimulationParameters`, 39

pymcxray.FileFormat.Results.Spectra, 39	pymcxray.FileFormat.Results.test_Spectrum, 54
pymcxray.FileFormat.Results.SpectraEDS, 40	pymcxray.FileFormat.Results.test_Tags, 54
pymcxray.FileFormat.Results.Spectrum, 40	pymcxray.FileFormat.Results.test_XrayIntensities, 55
pymcxray.FileFormat.Results.SpectrumEDS, 40	pymcxray.FileFormat.Results.test_XraySimulatedSpectra, 55
pymcxray.FileFormat.Results.Tags, 41	pymcxray.FileFormat.Results.test_XraySimulatedSpectraAtomEmitted, 55
pymcxray.FileFormat.Results.test_BaseResponse, 44	pymcxray.FileFormat.Results.test_XraySpectraAtomEmitted, 56
pymcxray.FileFormat.Results.test_BeamParameters, 44	pymcxray.FileFormat.Results.test_XraySpectraSpecimen, 56
pymcxray.FileFormat.Results.test_DetectorParameters, 45	pymcxray.FileFormat.Results.test_XraySpectraSpecimenEmitted, 57
pymcxray.FileFormat.Results.test_Dump, 45	pymcxray.FileFormat.Results.XrayIntensities, 41
pymcxray.FileFormat.Results.test_ElectrodeParameters, 45	pymcxray.FileFormat.Results.XraySimulatedSpectraRegions, 41
pymcxray.FileFormat.Results.test_ElectrodeResults, 46	pymcxray.FileFormat.Results.XraySimulatedSpectraSpecimen, 42
pymcxray.FileFormat.Results.test_ElementParameters, 46	pymcxray.FileFormat.Results.XraySpectraAtomEmitted, 42
pymcxray.FileFormat.Results.test_Intersections, 47	pymcxray.FileFormat.Results.XraySpectraRegionEmitted, 42
pymcxray.FileFormat.Results.test_MicroscopeParameters, 47	pymcxray.FileFormat.Results.XraySpectraRegionsEmitted, 43
pymcxray.FileFormat.Results.test_ModelParameters, 47	pymcxray.FileFormat.Results.XraySpectraSpecimen, 43
pymcxray.FileFormat.Results.test_PhirhozParameters, 48	pymcxray.FileFormat.Results.XraySpectraSpecimenEmitted, 43
pymcxray.FileFormat.Results.test_PhirhozResults, 48	pymcxray.FileFormat.Results.Parameters, 62
pymcxray.FileFormat.Results.test_PhirhozSimulatedSpectrum, 49	pymcxray.FileFormat.Results.ThinFilmInputs, 63
pymcxray.FileFormat.Results.test_PhirhozSimulationParameters, 50	pymcxray.FileFormat.SimulationParameters, 63
pymcxray.FileFormat.Results.test_PhirhozSimulationParameters, 50	pymcxray.FileFormat.SimulationParameters, 64
pymcxray.FileFormat.Results.test_PhirhozSimulationParameters, 50	pymcxray.FileFormat.Specimen, 64
pymcxray.FileFormat.Results.test_PhirhozSimulationParameters, 50	pymcxray.FileFormat.test_ExportedSpectrum, 65
pymcxray.FileFormat.Results.test_PhirhozRegion, 51	pymcxray.FileFormat.test_FileReaderWriterTools, 66
pymcxray.FileFormat.Results.test_PhirhozRegionParameters, 51	pymcxray.FileFormat.test_MCXRayModel, 66
pymcxray.FileFormat.Results.test_PhirhozRegionVolume, 52	pymcxray.FileFormat.test_Region, 67
pymcxray.FileFormat.Results.test_SimulationParameters, 52	pymcxray.FileFormat.test_RegionDimensions, 67
pymcxray.FileFormat.Results.test_SpectraParameters, 52	pymcxray.FileFormat.test_RegionType, 68
pymcxray.FileFormat.Results.test_SpectraEDS, 53	pymcxray.FileFormat.test_SnrParameters, 69
	pymcxray.FileFormat.test_Version, 70

- pymcxray.FileFormat.testUtilities, 65
- pymcxray.FileFormat.Version, 65
- pymcxray.mcxray, 80
- pymcxray.multipleloop, 80
- pymcxray.pymcxray, 81
- pymcxray.serialization, 73
 - pymcxray.serialization.SerializationH5py, 70
 - pymcxray.serialization.SerializationNumpy, 71
 - pymcxray.serialization.SerializationPickle, 71
 - pymcxray.serialization.test_SerializationH5py, 72
 - pymcxray.serialization.test_SerializationNumpy, 72
 - pymcxray.serialization.test_SerializationPickle, 72
- pymcxray.serialization.tests, 73
- pymcxray.Simulation, 77
- pymcxray.SimulationsParameters, 79
- pymcxray.test_AtomData, 81
- pymcxray.test_BatchFileConsole, 81
- pymcxray.test_ComparisonModels, 82
- pymcxray.test_mcxray, 83
- pymcxray.test_Simulation, 82
- pymcxray.test_SimulationsParameters, 82
- pymcxray.tests, 73
 - pymcxray.tests.test_pymcxray, 73

A

- acquisitionTime_s (pym-
cxray.FileFormat.Results.BeamParameters.BeamParameters
attribute), 31
- addCollision() (pym-
cxray.FileFormat.Results.ElectronTrajectoriesResults.Trajectory
method), 34
- addCompute() (pym-
cxray.SimulationsParameters.SimulationsParameters
method), 79
- addExperiment() (pym-
cxray.SimulationsParameters.SimulationsParametersFixed
method), 80
- addFixed() (pymcxray.SimulationsParameters.SimulationsParameters
method), 79
- addSimulationName() (pym-
cxray.BatchFile.BatchFile method), 74
- addSimulationName() (pym-
cxray.BatchFileConsole.BatchFileConsole
method), 74
- addVaried() (pymcxray.SimulationsParameters.SimulationsParameters
method), 80
- AnalyzeNumberBackgroundWindows
(class in pym-
cxray.AnalyzeNumberBackgroundWindows),
73
- angleBetweenDetectorSpecimenNormal_deg
(pymcxray.FileFormat.Results.DetectorParameters.DetectorParameters
attribute), 31
- angleBetweenDetectorXAxis_deg (pym-
cxray.FileFormat.Results.DetectorParameters.DetectorParameters
attribute), 31
- AtomCollisionModel (class in pym-
cxray.FileFormat.MCXRayModel), 58
- atomCollisionModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomCollisionScreeningModel (class in pym-
cxray.FileFormat.MCXRayModel), 58
- atomCollisionScreeningModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomCrossSectionModel (class in pym-
cxray.FileFormat.MCXRayModel), 58
- atomCrossSectionModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomCrossSectionScreeningModel (class in
pymcxray.FileFormat.MCXRayModel), 58
- atomCrossSectionScreeningModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomElectronRangeModel (class in pym-
cxray.FileFormat.MCXRayModel), 58
- atomElectronRangeModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomEnergyLossModel (class in pym-
cxray.FileFormat.MCXRayModel), 58
- atomEnergyLossModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- atomicNumber (pym-
cxray.FileFormat.Element.Element attribute),
57
- AtomMeanIonizationPotentialModel (class in
pymcxray.FileFormat.MCXRayModel), 58
- atomMeanIonizationPotentialModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- AtomScreeningModel (class in pym-
cxray.FileFormat.MCXRayModel), 59
- atomScreeningModel (pym-
cxray.FileFormat.Results.ModelParameters.ModelParameters
attribute), 35
- azimuthalAngleRange_deg (pym-
cxray.FileFormat.Results.ElectronExistResults.ElectronDetector
attribute), 32

B

backgroundEnergyWindowsSize (pym-
 cxray.FileFormat.SnrParameters.SnrParameters
 attribute), 64

backgroundIntensities (pym-
 cxray.FileFormat.Results.Spectrum.Spectrum
 attribute), 40

backgrounds (pymcxray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen
 attribute), 43

backgrounds (pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected
 attribute), 43

backscatteredRatio (pym-
 cxray.FileFormat.Results.ElectronParameters.ElectronParameters
 attribute), 33

baseFilename (pym-
 cxray.FileFormat.SimulationParameters.SimulationParameters
 attribute), 63

basename (pymcxray.FileFormat.Results.BaseResults.BaseResults
 attribute), 31

basename (pymcxray.Simulation.Simulation attribute), 77

BaseResults (class in pym-
 cxray.FileFormat.Results.BaseResults), 31

BatchFile (class in pymcxray.BatchFile), 74

BatchFileConsole (class in pym-
 cxray.BatchFileConsole), 74

beamCurrent_A (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamDetectorDistance_cm (pym-
 cxray.FileFormat.Results.DetectorParameters.DetectorParameters
 attribute), 31

beamDiameter_A (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamDiameter_nm (pymcxray.Simulation.Simulation
 attribute), 77

beamEnergy_keV (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

BeamParameters (class in pym-
 cxray.FileFormat.Results.BeamParameters), 31

beamParameters (pym-
 cxray.FileFormat.Results.MicroscopeParameters.MicroscopeParameters
 attribute), 35

beamPosition_nm (pymcxray.Simulation.Simulation
 attribute), 77

beamPositionX_A (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamPositionY_A (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamStandardDeviation_A (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamTilt_deg (pym-
 cxray.FileFormat.MicroscopeParameters.MicroscopeParameters
 attribute), 60

beamTilt_deg (pymcxray.Simulation.Simulation at-
 tribute), 77

bremsstrahlung_1_ekeVsr (pym-
 cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
 attribute), 42

bremsstrahlungCrossSectionModel (pym-
 cxray.FileFormat.Results.ModelParameters.ModelParameters
 attribute), 35

bremsstrahlungValue_1_ekeVsr () (pym-
 cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
 method), 42

channelNumbers (pym-
 cxray.FileFormat.Results.XraySimulatedSpectraRegion.XraySimulatedSpectraRegion
 attribute), 41

channels (pymcxray.FileFormat.Results.SpectrumEDS.SpectrumEDS
 attribute), 40

characteristicCrossSectionModel (pym-
 cxray.FileFormat.Results.ModelParameters.ModelParameters
 attribute), 36

characteristic_1_ekeVsr (pym-
 cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
 attribute), 42

characteristicIntensities (pym-
 cxray.FileFormat.Results.Spectrum.Spectrum
 attribute), 40

characteristicProbability (pym-
 cxray.FileFormat.Results.SpectraEDS.SpectraEDS
 attribute), 40

characteristics (pym-
 cxray.FileFormat.Results.XraySpectraAtomEmittedDetectedLines.XraySpectraAtomEmittedDetectedLines
 attribute), 42

characteristics (pym-
 cxray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen
 attribute), 43

characteristics (pym-
 cxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected
 attribute), 43

characteristicValue_1_ekeVsr () (pym-
 cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted
 method), 42

clear () (pymcxray.FileFormat.Region.Region
 method), 61

clear () (pymcxray.FileFormat.Specimen.Specimen
 method), 64

CoPlanarParameters (class in pym-
 cxray.FileFormat.Results.ElectronTrajectoriesResults),

34

collisions (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision attribute), 34

collisionType (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision attribute), 34

combine() (in module pymcrray.multipleloop), 80

ComparisonModels (class in pymcrray.ComparisonModels), 75

computeAtomicDensity_atom_cm3() (in module pymcrray.ElementProperties), 75

computeNumberXrays() (pymcrray.SimulationsParameters.SimulationsParameters method), 80

computeNumberXraysFilepath (pymcrray.SimulationsParameters.SimulationsParameters attribute), 80

computeWeightFraction() (in module pymcrray.Simulation), 78

correctedX_A (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision attribute), 34

correctedY_A (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision attribute), 34

correctedZ_A (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision attribute), 34

countsList (pymcrray.FileFormat.Results.SpectrumEDS.SpectrumEDS attribute), 41

create_cnt_sample() (in module pymcrray.Simulation), 79

create_multi_horizontal_layer() (in module pymcrray.Simulation), 79

create_path() (in module pymcrray), 83

create_weight_fractions() (in module pymcrray.Simulation), 79

createAlloyBoxInSubstrate() (in module pymcrray.Simulation), 78

createAlloyBoxInThinFilm() (in module pymcrray.Simulation), 78

createAlloyBoxInVacuum() (in module pymcrray.Simulation), 78

createAlloyBulkSample() (in module pymcrray.Simulation), 78

createAlloyFilmOverSubstrate() (in module pymcrray.Simulation), 78

createAlloyMultiVerticalLayer() (in module pymcrray.Simulation), 78

createAlloyParticleInSubstrate() (in module pymcrray.Simulation), 78

createAlloyParticleInThinFilm() (in module pymcrray.Simulation), 78

createAlloyThinFilm() (in module pymcrray.Simulation), 79

createBoxFeatureInSubstrate() (in module pymcrray.Simulation), 79

createFilmInSubstrate() (in module pymcrray.Simulation), 79

createFilmOverSubstrate() (in module pymcrray.Simulation), 79

createLineOldVersion() (pymcrray.FileFormat.Element.Element method), 57

createLineOldVersion() (pymcrray.FileFormat.RegionDimensions.RegionDimensions method), 61

createLinesWithKey() (pymcrray.FileFormat.Element.Element method), 57

createLinesWithoutVersion() (pymcrray.FileFormat.Region.Region method), 57

createLinesWithVersion() (pymcrray.FileFormat.Region.Region method), 57

createLineWithKey() (pymcrray.FileFormat.RegionDimensions.RegionDimensions method), 61

createParticleInSubstrate() (in module pymcrray.Simulation), 79

createParticleOnFilm() (in module pymcrray.Simulation), 79

createParticleOnSubstrate() (in module pymcrray.Simulation), 79

createPhirhozSpecimens() (in module pymcrray.Simulation), 79

createPureBulkSample() (in module pymcrray.Simulation), 79

createRegionDimensions() (in module pymcrray.FileFormat.RegionDimensions), 62

createSimulationFiles() (pymcrray.Simulation.Simulation method), 77

createTempDataPath() (in module pymcrray.FileFormat.testUtilities), 65

crystalDensity_g_cm3 (pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters attribute), 32

crystalName (pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters attribute), 32

crystalRadius_cm (pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters attribute), 32

crystalThickness_cm (pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters attribute), 32

current_A (pymcrray.FileFormat.Results.BeamParameters.BeamParameters), 60

attribute), 31

current_A (pymcrray.Simulation.Simulation attribute), 77

D

data (pymcrray.FileFormat.Results.ElectronExistResults.ElectronExistResults), 33

DataMap (class in pymcrray.FileFormat.Results.exported.DataMap), 29

deadLayerThickness_A (pymcrray.FileFormat.Results.DetectorParameters.DetectorParameters), 32

DebugSimulatedSpectrum (class in pymcrray.DebugSimulatedSpectrum), 75

defaultValues () (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters method), 60

defaultValues () (pymcrray.FileFormat.ResultsParameters.ResultsParameters method), 62

defaultValues () (pymcrray.FileFormat.SimulationParameters.SimulationParameters method), 63

defaultValues () (pymcrray.FileFormat.SnrParameters.SnrParameters method), 64

depth_A (pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic), 36

depth_A (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic), 37

depth_nm (pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic), 36

depth_nm (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic), 37

depths_A (pymcrray.FileFormat.Results.Phirhoz.Phirhoz), 36

detectedIntensities (pymcrray.FileFormat.Results.XraySimulatedSpectralRegion.XraySimulatedSpectralRegion), 41

detectElectrons () (pymcrray.FileFormat.Results.ElectronExistResults.ElectronExistResults method), 32

detectorAzimuthalAngle_deg (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorAzimuthalAngle_deg (pymcrray.Simulation.Simulation attribute), 77

detectorBFHigh_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorBFLOW_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 35

detectorChannelWidth_eV (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorChannelWidth_eV (pymcrray.Simulation.Simulation attribute), 77

detectorCrystalAtomSymbol (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorCrystalDistance_cm (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorCrystalDistance_cm (pymcrray.Simulation.Simulation attribute), 77

detectorCrystalRadius_cm (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorCrystalRadius_cm (pymcrray.Simulation.Simulation attribute), 77

detectorCrystalThickness_cm (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorCrystalThickness_cm (pymcrray.Simulation.Simulation attribute), 77

detectorDeadLayer_A (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorDFHigh_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorDFLow_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorDiffusionLenght_A (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorHAADFHigh_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorHAADFHigh_rad (pymcrray.Simulation.Simulation attribute), 77

detectorHAADFLow_rad (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorNoise_eV (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

detectorNoise_eV (pymcrray.Simulation.Simulation attribute), 77

DetectorParameters (class in pymcrray.FileFormat.Results.DetectorParameters), 31

DetectorParameters (pymcrray.FileFormat.Results.MicroscopeParameters.MicroscopeParameters), 35

DetectorPitch_deg (pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters), 60

`cxray.FileFormat.MicroscopeParameters.MicroscopeParameters`
 attribute), 60
 detectorSurfaceQuality (pym- `Element` (class in `pymcxray.FileFormat.Element`), 57
`cxray.FileFormat.MicroscopeParameters.MicroscopeParameters`
 attribute), 60
 detectorTOA_deg (pym- `ElementParameters` (class in `pym-`
`cxray.FileFormat.MicroscopeParameters.MicroscopeParameters`
 attribute), 60
 diameter90_A (pym- `elements` (`pymcxray.FileFormat.Results.RegionParameters.RegionParam`
`cxray.FileFormat.MicroscopeParameters.MicroscopeParameters`
 attribute), 60
 diameter90_A (pym- `elements` (`pymcxray.FileFormat.Results.SpectraEDS.SpectraEDS`
`cxray.FileFormat.Results.BeamParameters.BeamParameters`
 attribute), 31
 diffusionLength_A (pym- `energies_keV` (`pymcxray.FileFormat.Results.SpectrumEDS.SpectrumEDS`
`cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
 attribute), 32
 directionX (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsCylinder` (pym-
 attribute), 62
 directionY (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsCylinder` (pym-
 attribute), 62
 directionZ (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsCylinder` (pym-
 attribute), 62
 drawXY () (`pymcxray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults` (pym-
 method), 34
 drawXZ () (`pymcxray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults` (pym-
 method), 34
 drawYZ () (`pymcxray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults` (pym-
 method), 34
 Dump (class in `pymcxray.FileFormat.Results.Dump`), 32
 E
 edsMaximumEnergy_keV (pym- `energies_keV` (`pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.X`
`cxray.FileFormat.Results.SimulationParameters.SimulationParameters`
 attribute), 39
 elasticCrossSectionScalingFactor (pym- `energiesReference_keV` (`pym-`
`cxray.FileFormat.SimulationParameters.SimulationParameters`
 attribute), 63
 elasticCrossSectionScalingFactor (pym- `energy_keV` (`pymcxray.FileFormat.Results.ElectronTrajectoriesResults.C`
`cxray.Simulation.Simulation` attribute), 77
 ElectronDetector (class in `pym-`
`cxray.FileFormat.Results.ElectronExistResults`),
 32
 ElectronExistResults (class in `pym-`
`cxray.FileFormat.Results.ElectronExistResults`),
 33
 ElectronParameters (class in `pym-`
`cxray.FileFormat.Results.ElectronParameters`),
 33
 electronParameters (pym- `energyEnd_keV` (`pym-`
`cxray.FileFormat.SnrParameters.SnrParameters`
 attribute), 64
 electronParameters (pym- `energyLossScalingFactor` (`pym-`
`cxray.FileFormat.SimulationParameters.SimulationParameters`
 attribute), 63
 ElectronResults (class in `pym-`
`cxray.FileFormat.Results.ElectronResults`),
 33
 ElectronTrajectoriesResults (class in `pym-`
`cxray.FileFormat.Results.ElectronTrajectoriesResults`),
 33

`crray.FileFormat.SnrParameters.SnrParameters` (class in `pymcrray.FileFormat.ExportedSpectrum`), 58
`eNetPeak` (`pymcrray.FileFormat.Results.XraySimulatedSpectraRegion` (in module `pymcrray`), 77
`eRatio` (`pymcrray.FileFormat.Results.ElectronParameters.ElectronParameters` (in module `pymcrray`), 31
`ExportedSpectrum` (class in `pymcrray.FileFormat.ExportedSpectrum`), 58
`extractFromLineOldVersion` (`pymcrray.FileFormat.Element.Element` method), 57
`extractFromLineOldVersion` (`pymcrray.FileFormat.RegionDimensions.RegionDimensions` method), 61
`extractFromLines` (`pymcrray.FileFormat.Results.Intersections.Intersections` method), 35
`extractFromLinesWithKey` (`pymcrray.FileFormat.Element.Element` method), 57
`extractFromLinesWithKey` (`pymcrray.FileFormat.RegionDimensions.RegionDimensions` method), 61
`extractFromLinesWithoutVersion` (`pymcrray.FileFormat.Region.Region` method), 61
`extractFromLinesWithVersion` (`pymcrray.FileFormat.Region.Region` method), 61
`fieldNames` (`pymcrray.FileFormat.Results.XraySpectraSpecimenEmitted` attribute), 43
`fieldNames` (`pymcrray.FileFormat.Results.XraySimulatedSpectraRegion` (in module `pymcrray`), 77
`fieldNames` (`pymcrray.FileFormat.Results.ElectronParameters.ElectronParameters` (in module `pymcrray`), 31
`fieldNames` (`pymcrray.FileFormat.Results.BaseResults.BaseResults` (in module `pymcrray`), 31
`find_all_files` (`pymcrray`), 83
`findAllTag` (`pymcrray.FileFormat.Results.Tags` (in module `pymcrray`), 41
`findTag` (`pymcrray.FileFormat.Results.Tags` (in module `pymcrray`), 41
`fixedParameters` (`pymcrray.SimulationsParameters.SimulationsParameters` (in module `pymcrray`), 80
`fractionBackscatteredElectrons` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` (in module `pymcrray`), 33
`fractionInternalElectrons` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` (in module `pymcrray`), 33
`fractionSkirtedElectrons` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` (in module `pymcrray`), 33
`fractionTransmittedElectrons` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` (in module `pymcrray`), 33
`fromString` (`pymcrray.FileFormat.Version.Version` method), 65

G

F
`fieldNames` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` (in module `pymcrray`), 33
`fieldNames` (`pymcrray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm.PhirhozEmittedCharacteristicThinFilm` (in module `pymcrray`), 36
`fieldNames` (`pymcrray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm.PhirhozEmittedCharacteristicThinFilm` (in module `pymcrray`), 37
`fieldNames` (`pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm` (in module `pymcrray`), 37
`fieldNames` (`pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm` (in module `pymcrray`), 38
`fieldNames` (`pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities` (in module `pymcrray`), 31
`fieldNames` (`pymcrray.FileFormat.Results.XraySimulatedSpectraRegion` (in module `pymcrray`), 42
`fieldNames` (`pymcrray.FileFormat.Results.XraySimulatedSpectraSpecimen` (in module `pymcrray`), 42
`fieldNames` (`pymcrray.FileFormat.Results.XraySpectraAtomEmittedDetectedLines.XraySpectraAtomEmittedDetectedLines` (in module `pymcrray`), 42
`fieldNames` (`pymcrray.FileFormat.Results.XraySpectraRegionEmittedXraySpectraRegionEmitted` (in module `pymcrray`), 43
`fieldNames` (`pymcrray.FileFormat.Results.XraySpectraSpecimen` (in module `pymcrray`), 43
`g_atomicMass_g_mol` (`pymcrray.ElementProperties` (in module `pymcrray`), 76
`g_FermiEnergy` (`pymcrray.ElementProperties` (in module `pymcrray`), 76
`g_kFermi` (`pymcrray.ElementProperties` (in module `pymcrray`), 76
`g_kFermi` (`pymcrray.ElementProperties` (in module `pymcrray`), 76
`g_kFermi` (`pymcrray.ElementProperties` (in module `pymcrray`), 76
`gaussianSigma` (`pymcrray.BeamParameters.BeamParameters` (in module `pymcrray`), 31
`generateBaseFilename` (`pymcrray.SimulationParameters.SimulationParameters` (in module `pymcrray`), 78
`get_atomic_weight_g_mol` (`pymcrray`), 74
`get_current_module_path` (`pymcrray`), 84

<code>get_mcrray_archive_name()</code> (in module <code>pymcrray</code>), 84	<code>getElementSpectra()</code> (pymcrray.FileFormat.Results.Spectra.Spectra method), 39
<code>get_mcrray_archive_path()</code> (in module <code>pymcrray</code>), 84	<code>getElementSpectrum()</code> (pymcrray.FileFormat.Results.Spectra.Spectra method), 39
<code>get_mcrray_program_name()</code> (in module <code>pymcrray</code>), 84	<code>getEnergyDistribution()</code> (pymcrray.FileFormat.Results.ElectronExistResults.ElectronExistResults method), 33
<code>get_mcrray_program_path()</code> (in module <code>pymcrray</code>), 85	<code>getExtension()</code> (pymcrray.FileFormat.SimulationInputs.SimulationInputs method), 63
<code>get_result_types()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41	<code>getFermiEnergy_eV()</code> (in module <code>pymcrray.ElementProperties</code>), 76
<code>get_results_mcgill_path()</code> (in module <code>pymcrray</code>), 85	<code>getGeneratedCharacteristicThinFilmPhirhoz()</code> (pymcrray.Simulation.Simulation method), 78
<code>get_subshells()</code> (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilmPhirhoz method), 38	<code>getIntensity()</code> (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilmPhirhoz method), 37
<code>get_symbols()</code> (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilmPhirhoz method), 38	<code>getIntensity()</code> (pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilmPhirhoz method), 38
<code>get_xray_lines()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41	<code>getIntensity()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41
<code>getAllSimulationParameters()</code> (pymcrray.SimulationsParameters.SimulationsParameters method), 80	<code>getIntensityEmitted()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41
<code>getAllSimulationParameters()</code> (pymcrray.SimulationsParameters.SimulationsParametersFixed method), 80	<code>getIntensityEmittedDetected()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41
<code>getAtomicMass_g_mol()</code> (in module <code>pymcrray.ElementProperties</code>), 76	<code>getIntensityGenerated()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41
<code>getAtomicNumber()</code> (in module <code>pymcrray.AtomData</code>), 74	<code>getIntensityGeneratedDetected()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41
<code>getAtomicNumber()</code> (in module <code>pymcrray.ElementProperties</code>), 76	<code>getIonizationEnergy_keV()</code> (in module <code>pymcrray.AtomData</code>), 74
<code>getAtomicNumberByName()</code> (in module <code>pymcrray.ElementProperties</code>), 76	<code>getKRatioCorrection()</code> (in module <code>pymcrray.ElementProperties</code>), 76
<code>getAtomicNumberBySymbol()</code> (in module <code>pymcrray.ElementProperties</code>), 76	<code>getKRatioCorrectionMonsel()</code> (in module <code>pymcrray.ElementProperties</code>), 76
<code>getAtomicNumberLineEnergySets()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41	<code>getLineAndReference()</code> (in module <code>pymcrray.FileFormat.Results.test_PhirhozElement</code>), 49
<code>getAtomSymbol()</code> (in module <code>pymcrray.AtomData</code>), 74	<code>getLineFormat()</code> (pymcrray.FileFormat.RegionDimensions.RegionDimensions method), 62
<code>getCharacteristicPhiRhoZ()</code> (pymcrray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated method), 37	<code>getLineFormat()</code> (pymcrray.FileFormat.RegionDimensions.RegionDimensionsBox method), 62
<code>getData()</code> (pymcrray.FileFormat.ExportedSpectrum.ExportedSpectrum method), 58	
<code>getDetectedIntensity()</code> (pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities method), 41	
<code>getElectronGunPositions_nm()</code> (pymcrray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults method), 34	

[getLineFormat\(\)](#) (pymcxray.FileFormat.RegionDimensions.RegionDimensionsCylinder method), 62
[getLineFormat\(\)](#) (pymcxray.FileFormat.RegionDimensions.RegionDimensionsSphere method), 62
[getLines\(\)](#) (pymcxray.FileFormat.Results.test_Tags.TestTagsSpecimenSpectrum method), 54
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_ElectronParameters), 46
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_MicroscopeParameters), 47
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_ModelParameters), 48
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_Phirhoz), 48
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_PhirhozRegion), 51
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_RegionVolume), 52
[getLinesAndReference\(\)](#) (in module pymcxray.FileFormat.Results.test_SimulationParameters), 52
[getMassDensity_g_cm3\(\)](#) (in module pymcxray.AtomData), 74
[getMassDensity_g_cm3\(\)](#) (in module pymcxray.ElementProperties), 77
[getMeanIonizationEnergy_eV\(\)](#) (in module pymcxray.ElementProperties), 77
[getModel\(\)](#) (pymcxray.FileFormat.MCXRayModel.MCXRayModel method), 59
[getModelList\(\)](#) (pymcxray.FileFormat.Models.Models method), 61
[getName\(\)](#) (in module pymcxray.ElementProperties), 77
[getParameters\(\)](#) (pymcxray.Simulation.Simulation method), 78
[getPlasmonEnergy_eV\(\)](#) (in module pymcxray.ElementProperties), 77
[getProgramVersionFilepath\(\)](#) (pymcxray.Simulation.Simulation method), 78
[getRegionParameters\(\)](#) (pymcxray.FileFormat.Results.Spectra.Spectra method), 39
[getRegionSpectrum\(\)](#) (pymcxray.FileFormat.Results.Spectra.Spectra method), 39
[getShellList\(\)](#) (in module pymcxray.AtomData), 74
[getSnrParametersReference\(\)](#) (pymcxray.FileFormat.test_SnrParameters.TestSnrParameters method), 69
[getSpectrumType\(\)](#) (pymcxray.FileFormat.ExportedSpectrum.ExportedSpectrum method), 58
[getSymbol\(\)](#) (in module pymcxray.ElementProperties), 77
[getTestRegionLinesWithoutVersion\(\)](#) (pymcxray.FileFormat.test_Region.TestRegion method), 67
[getTestRegionLinesWithVersion\(\)](#) (pymcxray.FileFormat.test_Region.TestRegion method), 67
[getVariedParameterLabels\(\)](#) (pymcxray.SimulationsParameters.SimulationsParameters method), 80
[getXRayEnergy_keV\(\)](#) (in module pymcxray.AtomData), 74
[graphicsEnergyLoss\(\)](#) (pymcxray.ComparisonModels.ComparisonModels method), 75
[graphicsIonizationCrossSection\(\)](#) (pymcxray.ComparisonModels.ComparisonModels method), 75
[graphicsXrayCrossSectionBremstrahlung\(\)](#) (pymcxray.ComparisonModels.ComparisonModels method), 75
[graphicsXrayMassAbsorptionCoefficient\(\)](#) (pymcxray.ComparisonModels.ComparisonModels method), 75
[imageName](#) (pymcxray.FileFormat.Results.exported.DataMap.DataMap attribute), 29
[incidentEnergy_keV](#) (pymcxray.FileFormat.Results.BeamParameters.BeamParameters attribute), 31
[index](#) (pymcxray.FileFormat.Results.ElectronTrajectoriesResults.TrajectoriesResults attribute), 34
[indexRegion](#) (pymcxray.FileFormat.Results.ElectronTrajectoriesResults.TrajectoriesResults attribute), 34
[intensities](#) (pymcxray.FileFormat.Results.PhirhozEmittedCharacteristics attribute), 37
[intensities](#) (pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristics attribute), 38
[intensities](#) (pymcxray.FileFormat.Results.Spectrum.Spectrum attribute), 40

intensities (`pymcxray.FileFormat.Results.XrayIntensities.XrayIntensities` attribute), 41

intensity (`pymcxray.FileFormat.Results.Phirhoz.Phirhoz` attribute), 36

internalRatio (`pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters` attribute), 33

interpolationType (`pymcxray.FileFormat.Results.SimulationParameters.SimulationParameters` attribute), 39

Intersections (class in `pymcxray.FileFormat.Results.Intersections`), 35

isComputeXrayBremsstrahlung (`pymcxray.FileFormat.ResultsParameters.ResultsParameters` attribute), 62

isComputeXrayCharacteristic (`pymcxray.FileFormat.ResultsParameters.ResultsParameters` attribute), 62

isComputeXrayPhirhoz (`pymcxray.FileFormat.ResultsParameters.ResultsParameters` attribute), 62

isComputeXraySimulatedSpectrum (`pymcxray.FileFormat.ResultsParameters.ResultsParameters` attribute), 63

isDone() (`pymcxray.Simulation.Simulation` method), 78

isIonizationShell_K (`pymcxray.FileFormat.Results.PhirhozElement.PhirhozElement` attribute), 36

isIonizationShell_L (`pymcxray.FileFormat.Results.PhirhozElement.PhirhozElement` attribute), 36

isIonizationShell_M (`pymcxray.FileFormat.Results.PhirhozElement.PhirhozElement` attribute), 36

key (`pymcxray.FileFormat.Version.Version` attribute), 65

KEY_FILE_VERSION (`pymcxray.serialization.SerializationPickle.SerializationPickle` attribute), 71

KEY_SERIALIZED_DATA (`pymcxray.serialization.SerializationPickle.SerializationPickle` attribute), 71

L

Layer (class in `pymcxray.Simulation`), 77

layerThickness_A (`pymcxray.FileFormat.Results.RegionParameters.RegionParameters` attribute), 38

length (`pymcxray.FileFormat.RegionDimensions.RegionDimensions` attribute), 62

load() (`pymcxray.serialization.SerializationH5py.SerializationH5py` method), 70

load() (`pymcxray.serialization.SerializationNumpy.SerializationNumpy` method), 71

load() (`pymcxray.serialization.SerializationNumpy.SerializationNumpy` method), 71

load() (`pymcxray.serialization.SerializationNumpy.SerializationNumpy` method), 71

load() (`pymcxray.serialization.SerializationNumpy.SerializationNumpy` method), 71

load() (`pymcxray.serialization.SerializationPickle.SerializationPickle` method), 71

M

major (`pymcxray.FileFormat.Version.Version` attribute), 65

mapFilename (`pymcxray.FileFormat.SimulationInputs.SimulationInputs` attribute), 63

MassAbsorptionCoefficientModel (class in `pymcxray.FileFormat.MCXRayModel`), 59

massFraction (`pymcxray.FileFormat.Element.Element` attribute), 57

maximumAzimuthalAngle_deg (`pymcxray.FileFormat.Results.ElectronExistResults.ElectronDetector` attribute), 32

maximumEnergy_keV (`pymcxray.FileFormat.Results.ElectronExistResults.ElectronDetector` attribute), 32

maximumLiveTime_s (`pymcxray.FileFormat.Results.SimulationParameters.SimulationParameters` attribute), 39

maximumPolarAngle_deg (`pymcxray.FileFormat.Results.ElectronExistResults.ElectronDetector` attribute), 32

maximumX (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsBase` attribute), 62

maximumY (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsBase` attribute), 62

maximumZ (`pymcxray.FileFormat.RegionDimensions.RegionDimensionsBase` attribute), 62

MCXRayModel (class in `pymcxray.FileFormat.MCXRayModel`), 59

meanAzimuthalAngleCollision_deg (`pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters` attribute), 33

meanDistanceBetweenCollisions_A (`pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters` attribute), 33

meanNumberCollisionPerElectrons (`pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters` attribute), 33

meanPolarAngleCollision_deg (`pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters` attribute), 33

attribute), 33
 MicroscopeParameters (class in pym-
 cxray.FileFormat.MicroscopeParameters),
 60
 MicroscopeParameters (class in pym-
 cxray.FileFormat.Results.MicroscopeParameters),
 35
 microscopeParameters (pym- name (pymcxray.FileFormat.Element.Element attribute),
 cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated
 attribute), 37
 microscopeFilename (pym- name (pymcxray.FileFormat.Specimen.Specimen at-
 cxray.FileFormat.SimulationInputs.SimulationInputs attribute), 64
 attribute), 63
 minimumAzimuthalAngle_deg (pym- name (pymcxray.Simulation.Simulation attribute), 78
 cxray.FileFormat.Results.ElectronExistResults.ElectronDetector
 attribute), 33
 minimumEnergy_keV (pym- NOtest_readFromLines () (pym-
 cxray.FileFormat.Results.ElectronExistResults.ElectronDetector
 attribute), 33
 minimumPolarAngle_deg (pym- cxray.FileFormat.Results.test_RegionVolume.TestRegionVolume
 cxray.FileFormat.Results.ElectronExistResults.ElectronDetector
 attribute), 33
 minimumX (pymcxray.FileFormat.RegionDimensions.RegionDimensionsBox
 attribute), 62
 minimumY (pymcxray.FileFormat.RegionDimensions.RegionDimensionsBox
 attribute), 62
 minimumZ (pymcxray.FileFormat.RegionDimensions.RegionDimensionsBox
 attribute), 62
 minor (pymcxray.FileFormat.Version.Version attribute),
 65
 modelAtomCollision (pym- numberBackscatteredElectrons (pym-
 cxray.FileFormat.Models.Models attribute),
 61
 modelAtomCrossSection (pym- cxray.FileFormat.Results.ElectronResults.ElectronResults
 cxray.FileFormat.Models.Models attribute), 34
 61
 modelAtomMac (pymcxray.FileFormat.Models.Models
 attribute), 61
 modelFilename (pym- numberChannels (pym-
 cxray.FileFormat.SimulationInputs.SimulationInputs
 attribute), 63
 ModelParameters (class in pym- numberElectrons (pym-
 cxray.FileFormat.Results.ModelParameters),
 35
 modelParameters (pym- cxray.FileFormat.SimulationParameters.SimulationParameters
 cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated
 attribute), 37
 Models (class in pymcxray.FileFormat.Models), 61
 modelSampleEnergyLoss (pym- numberElements (pym-
 cxray.FileFormat.Models.Models attribute),
 61
 modelXrayBremsstrahlung (pym- numberElements (pym-
 cxray.FileFormat.Models.Models attribute),
 61
 modelXrayBremsstrahlung (pym- numberEnergySteps (pym-
 cxray.Simulation.Simulation attribute), 78
 modelXrayCharacteristic (pym- cxray.FileFormat.SnrParameters.SnrParameters
 cxray.FileFormat.Models.Models attribute),
 61
 noiseEdsDetector_eV (pym- attribute), 64
 NOTest_readFromLines () (pym-
 cxray.FileFormat.Results.test_RegionVolume.TestRegionVolume
 attribute), 52
 numberBackscatteredElectrons (pym-
 cxray.FileFormat.Results.ElectronResults.ElectronResults
 attribute), 34
 numberChannels (pym-
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters
 attribute), 39
 numberElectrons (pym-
 cxray.FileFormat.SimulationParameters.SimulationParameters
 attribute), 63
 numberCharacteristicPeaks (pym-
 cxray.FileFormat.Results.SpectraEDS.SpectraEDS
 attribute), 40
 numberContinuumWindows (pym-
 cxray.Simulation.Simulation attribute), 78
 numberData (pymcxray.FileFormat.Results.ElectronExistResults.Electron
 attribute), 33
 numberElectronCollisions (pym-
 cxray.FileFormat.Results.ElectronResults.ElectronResults
 attribute), 34
 numberElectrons (pym-
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters
 attribute), 39
 numberElectrons (pym-
 cxray.FileFormat.SimulationParameters.SimulationParameters
 attribute), 63
 numberElectrons (pymcxray.Simulation.Simulation
 attribute), 78
 numberElements (pym-
 cxray.FileFormat.Results.RegionParameters.RegionParameters
 attribute), 38
 numberElements (pym-
 cxray.FileFormat.Results.SpectraEDS.SpectraEDS
 attribute), 40
 numberEnergySteps (pym-
 cxray.FileFormat.SnrParameters.SnrParameters
 attribute), 64

numberEnergyWindows (pym- attribute), 39
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters (pym- attribute), 39
 numberEnergyWindows (pym- attribute), 78
 cxray.Simulation.Simulation attribute), 78
 numberFilmsX (pym- attribute), 33
 cxray.FileFormat.SimulationParameters.SimulationParameters attribute), 33
 numberFilmsY (pym- attribute), 34
 cxray.FileFormat.SimulationParameters.SimulationParameters attribute), 34
 numberFilmsZ (pym- attribute), 40
 cxray.FileFormat.SimulationParameters.SimulationParameters attribute), 40
 numberIntensities (pym- attribute), 34
 cxray.FileFormat.Results.XrayIntensities.XrayIntensities attribute), 34
 numberInternalElectrons (pym- attribute), 34
 cxray.FileFormat.Results.ElectronResults.ElectronResults attribute), 34
 numberLayersX (pym- attribute), 64
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters attribute), 64
 numberLayersX (pymcxdxray.Simulation.Simulation attribute), 78
 numberLayersY (pym- attribute), 39
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters attribute), 39
 numberLayersY (pymcxdxray.Simulation.Simulation attribute), 78
 numberLayersZ (pym- attribute), 39
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters attribute), 39
 numberLayersZ (pymcxdxray.Simulation.Simulation attribute), 78
 numberPhotons (pym- attribute), 39
 cxray.FileFormat.Results.SimulationParameters.SimulationParameters attribute), 39
 numberPhotons (pym- attribute), 64
 cxray.FileFormat.SimulationParameters.SimulationParameters attribute), 64
 numberPhotons (pymcxdxray.Simulation.Simulation attribute), 78
 numberRegions (pym- attribute), 37
 cxray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm.PhirhozEmittedCharacteristicThinFilm attribute), 37
 numberRegions (pym- attribute), 37
 cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated attribute), 37
 numberRegions (pym- attribute), 38
 cxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm attribute), 38
 numberRegions (pym- attribute), 38
 cxray.FileFormat.Results.Spectra.Spectra attribute), 38

P
 path (pymcxdxray.FileFormat.Results.BaseResults.BaseResults attribute), 31
 peakToBackground (pym- attribute), 42
 cxray.FileFormat.Results.XraySimulatedSpectraRegion.XraySimulatedSpectraRegion attribute), 42
 peakToBackgroundAverage (pym- attribute), 42
 cxray.FileFormat.Results.XraySimulatedSpectraRegion.XraySimulatedSpectraRegion attribute), 42
 Phirhoz (class in pym- cxray.FileFormat.Results.Phirhoz), 36
 PhirhozElement (class in pym- cxray.FileFormat.Results.PhirhozElement), 36
 PhirhozEmittedCharacteristic (class in pym- cxray.FileFormat.Results.PhirhozEmittedCharacteristic), 36
 PhirhozEmittedCharacteristicThinFilm (class in pym- cxray.FileFormat.Results.PhirhozEmittedCharacteristicThinFilm), 37
 PhirhozGenerated (class in pym- cxray.FileFormat.Results.PhirhozGenerated), 37
 PhirhozGeneratedCharacteristic (class in pym- cxray.FileFormat.Results.PhirhozGeneratedCharacteristic), 37
 PhirhozGeneratedCharacteristicThinFilm (class in pym- cxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm), 37

38

PhirhozRegion (class in pymcxray.FileFormat.Results.PhirhozRegion), 38

phirhozs (pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic attribute), 36

phirhozs (pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic attribute), 37

pixels (pymcxray.FileFormat.Results.exported.DataMap.DataMap attribute), 29

plotData() (pymcxray.AnalyzeNumberBackgroundWindows.AnalyzeNumberBackgroundWindows method), 73

plotDifference() (pymcxray.AnalyzeNumberBackgroundWindows.AnalyzeNumberBackgroundWindows method), 73

polarAngleRange_deg (pymcxray.FileFormat.Results.ElectronExistResults.ElectronDetector attribute), 33

positionX (pymcxray.FileFormat.RegionDimensions.RegionDimensions attribute), 62

positionY (pymcxray.FileFormat.RegionDimensions.RegionDimensions attribute), 62

positionZ (pymcxray.FileFormat.RegionDimensions.RegionDimensions attribute), 62

pymcxray (module), 83

pymcxray.AnalyzeNumberBackgroundWindows (module), 73

pymcxray.AtomData (module), 74

pymcxray.BatchFile (module), 74

pymcxray.BatchFileConsole (module), 74

pymcxray.ComparisonModels (module), 75

pymcxray.DebugSimulatedSpectrum (module), 75

pymcxray.ElementProperties (module), 75

pymcxray.FileFormat (module), 70

pymcxray.FileFormat.Element (module), 57

pymcxray.FileFormat.ExportedSpectrum (module), 58

pymcxray.FileFormat.FileReaderWriterTools (module), 58

pymcxray.FileFormat.MCXRayModel (module), 58

pymcxray.FileFormat.MicroscopeParameters (module), 60

pymcxray.FileFormat.Models (module), 61

pymcxray.FileFormat.Region (module), 61

pymcxray.FileFormat.RegionDimensions (module), 61

pymcxray.FileFormat.RegionType (module), 62

pymcxray.FileFormat.Results (module), 57

pymcxray.FileFormat.Results.BaseResults (module), 31

pymcxray.FileFormat.Results.BeamParameters (module), 31

pymcxray.FileFormat.Results.DetectorParameters (module), 31

pymcxray.FileFormat.Results.Dump (module), 31

pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic (module), 36

pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic (module), 37

pymcxray.FileFormat.Results.ElectronExistResults (module), 33

pymcxray.FileFormat.Results.ElectronParameters (module), 33

pymcxray.FileFormat.Results.ElectronResults (module), 33

pymcxray.FileFormat.Results.ElectronTrajectoriesResults (module), 34

pymcxray.FileFormat.Results.ElementParameters (module), 35

pymcxray.FileFormat.Results.exported (module), 31

pymcxray.FileFormat.Results.exported.DataMap (module), 29

pymcxray.FileFormat.Results.exported.test_DataMap (module), 29

pymcxray.FileFormat.Results.exported.test_XrayIntensity (module), 30

pymcxray.FileFormat.Results.exported.XrayIntensity2 (module), 30

pymcxray.FileFormat.Results.Intersections (module), 35

pymcxray.FileFormat.Results.MicroscopeParameters (module), 35

pymcxray.FileFormat.Results.ModelParameters (module), 35

pymcxray.FileFormat.Results.Phirhoz (module), 36

pymcxray.FileFormat.Results.PhirhozElement (module), 36

pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic (module), 36

pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic2 (module), 37

pymcxray.FileFormat.Results.PhirhozGenerated (module), 37

pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic (module), 37

pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic2 (module), 38

pymcxray.FileFormat.Results.PhirhozRegion (module), 38

pymcxray.FileFormat.Results.RegionParameters (module), 38

pymcxray.FileFormat.Results.RegionVolume (module), 39

pymcxray.FileFormat.Results.SimulationParameters (module), 39

pymcxray.FileFormat.Results.Spectra (module), 39

[pymcxray.FileFormat.test_RegionType \(module\)](#), 68
[pymcxray.FileFormat.test_SnrParameters \(module\)](#), 69
[pymcxray.FileFormat.test_Version \(module\)](#), 70
[pymcxray.FileFormat.testUtilities \(module\)](#), 65
[pymcxray.FileFormat.Version \(module\)](#), 65
[pymcxray.mcxray \(module\)](#), 80
[pymcxray.multipleloop \(module\)](#), 80
[pymcxray.pymcxray \(module\)](#), 81
[pymcxray.serialization \(module\)](#), 73
[pymcxray.serialization.SerializationH5py \(module\)](#), 70
[pymcxray.serialization.SerializationNumpy \(module\)](#), 71
[pymcxray.serialization.SerializationPickle \(module\)](#), 71
[pymcxray.serialization.test_SerializationH5py \(module\)](#), 72
[pymcxray.serialization.test_SerializationNumpy \(module\)](#), 72
[pymcxray.serialization.test_SerializationPickle \(module\)](#), 72
[pymcxray.serialization.tests \(module\)](#), 73
[pymcxray.Simulation \(module\)](#), 77
[pymcxray.SimulationsParameters \(module\)](#), 79
[pymcxray.test_AtomData \(module\)](#), 81
[pymcxray.test_BatchFileConsole \(module\)](#), 81
[pymcxray.test_ComparisonModels \(module\)](#), 82
[pymcxray.test_mcxray \(module\)](#), 83
[pymcxray.test_Simulation \(module\)](#), 82
[pymcxray.test_SimulationsParameters \(module\)](#), 82
[pymcxray.tests \(module\)](#), 73, 83
[pymcxray.tests.test_pymcxray \(module\)](#), 73

R

[radius \(pymcxray.FileFormat.RegionDimensions.RegionDimensionsSphere attribute\)](#), 62
[read\(\) \(pymcxray.FileFormat.ExportedSpectrum.ExportedSpectrum method\)](#), 58
[read\(\) \(pymcxray.FileFormat.MicroscopeParameters.MicroscopeParameters method\)](#), 60
[read\(\) \(pymcxray.FileFormat.Models.Models method\)](#), 61
[read\(\) \(pymcxray.FileFormat.Results.Dump.Dump method\)](#), 32
[read\(\) \(pymcxray.FileFormat.Results.ElectronExistResults.ElectronExistResults method\)](#), 33
[read\(\) \(pymcxray.FileFormat.Results.ElectronResults.ElectronResults method\)](#), 34
[read\(\) \(pymcxray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults method\)](#), 34
[read\(\) \(pymcxray.FileFormat.Results.exported.DataMap.DataMap method\)](#), 29
[read\(\) \(pymcxray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic method\)](#), 36
[read\(\) \(pymcxray.FileFormat.Results.PhirhozEmittedCharacteristicThin.PhirhozEmittedCharacteristicThin method\)](#), 37
[read\(\) \(pymcxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated method\)](#), 37
[read\(\) \(pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic method\)](#), 38
[read\(\) \(pymcxray.FileFormat.Results.PhirhozGeneratedCharacteristicThin.PhirhozGeneratedCharacteristicThin method\)](#), 38
[read\(\) \(pymcxray.FileFormat.Results.Spectra.Spectra method\)](#), 40
[read\(\) \(pymcxray.FileFormat.Results.XrayIntensities.XrayIntensities method\)](#), 41
[read\(\) \(pymcxray.FileFormat.Results.XraySimulatedSpectraRegion.XraySimulatedSpectraRegion method\)](#), 42
[read\(\) \(pymcxray.FileFormat.Results.XraySimulatedSpectraSpecimen.XraySimulatedSpectraSpecimen method\)](#), 42
[read\(\) \(pymcxray.FileFormat.Results.XraySpectraAtomEmittedDetected.XraySpectraAtomEmittedDetected method\)](#), 42
[read\(\) \(pymcxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted method\)](#), 43
[read\(\) \(pymcxray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted method\)](#), 43
[read\(\) \(pymcxray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen method\)](#), 43
[read\(\) \(pymcxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected method\)](#), 43
[read\(\) \(pymcxray.FileFormat.Results.Parameters.ResultsParameters method\)](#), 63
[read\(\) \(pymcxray.FileFormat.SimulationInputs.SimulationInputs method\)](#), 63
[read\(\) \(pymcxray.FileFormat.SimulationParameters.SimulationParameters method\)](#), 64
[read\(\) \(pymcxray.FileFormat.SnrParameters.SnrParameters method\)](#), 64
[read\(\) \(pymcxray.FileFormat.Specimen.Specimen method\)](#), 64
[read_value_from_configuration_file\(\) \(in module pymcxray\)](#), 85
[readBackgroundPhirhozs\(\) \(pymcxray.FileFormat.Results.PhirhozRegion.PhirhozRegion method\)](#), 38
[readCharacteristicPhirhozs\(\) \(pymcxray.FileFormat.Results.PhirhozRegion.PhirhozRegion method\)](#), 38
[readDataFile\(\) \(pymcxray.AnalyzeNumberBackgroundWindows.AnalyzeNumberBackgroundWindows method\)](#), 73

`readData()` (`pymcxcray.FileFormat.Results.exported.XrayIntensityXYIntensityXY`
`method`), 30 `readPhirhozDistributions()` (`pym-`
`readElectron()` (`pym-` `cxray.FileFormat.Results.PhirhozRegion.PhirhozRegion`
`cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated`
`method`), 38
`method`), 37 `readRegion()` (`pym-`
`readElements()` (`pym-` `cxray.FileFormat.Results.Spectra.Spectra`
`cxray.FileFormat.Results.PhirhozRegion.PhirhozRegion` `method`), 40
`method`), 38 `readRegions()` (`pym-`
`readFileObject()` (`pym-` `cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated`
`cxray.FileFormat.Results.SpectraEDS.SpectraEDS` `method`), 37
`method`), 40 `readRegions()` (`pym-`
`readFilepath()` (`pym-` `cxray.FileFormat.Results.Spectra.Spectra`
`cxray.FileFormat.Results.SpectraEDS.SpectraEDS` `method`), 40
`method`), 40 `readRegionSpectraSection()` (`pym-`
`readFromFile()` (`pym-` `cxray.FileFormat.Results.SpectraEDS.SpectraEDS`
`cxray.FileFormat.Version.Version` `method`), `method`), 40
65 `readRegionVolume()` (`pym-`
`readFromLine()` (`pym-` `cxray.FileFormat.Results.PhirhozRegion.PhirhozRegion`
`cxray.FileFormat.Results.PhirhozElement.PhirhozElement` `method`), 38
`method`), 36 `readSimulationParameters()` (`pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated`
`cxray.FileFormat.Results.BeamParameters.BeamParameters`
`method`), 37
`method`), 31 `readSolidSimulationModels()` (`pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated`
`cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method`), 37
`method`), 32 `readSpecimen()` (`pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.Results.Spectra.Spectra`
`cxray.FileFormat.Results.ElectronParameters.ElectronParameters`
`method`), 40
`method`), 33 `readTestInputSection()` (`pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.Results.SpectraEDS.SpectraEDS`
`cxray.FileFormat.Results.MicroscopeParameters.MicroscopeParameters`
`method`), 35
`method`), 35 `reduceAfterDot()` (`in` `module` `pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.FileReaderWriterTools`),
`cxray.FileFormat.Results.ModelParameters.ModelParameters`
`method`), 36
`method`), 36 `Region` (`class` `in` `pymcxcray.FileFormat.Region`), 61
`readFromLines()` (`pym-` `RegionDimensions` (`class` `in` `pym-`
`cxray.FileFormat.Results.Phirhoz.Phirhoz` `cxray.FileFormat.RegionDimensions`), 61
`method`), 36 `RegionDimensionsBox` (`class` `in` `pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.RegionDimensions`), 62
`cxray.FileFormat.Results.PhirhozRegion.PhirhozRegion` `RegionDimensionsCylinder` (`class` `in` `pym-`
`method`), 38 `cxray.FileFormat.RegionDimensions`), 62
`readFromLines()` (`pym-` `RegionDimensionsSphere` (`class` `in` `pym-`
`cxray.FileFormat.Results.RegionVolume.RegionVolume` `cxray.FileFormat.RegionDimensions`), 62
`method`), 39 `RegionEnergyLossModel` (`class` `in` `pym-`
`readFromLines()` (`pym-` `cxray.FileFormat.MCXRModel`), 59
`cxray.FileFormat.Results.SimulationParameters.SimulationParameters`
`method`), 39 `RegionEnergyLossModel` (`pym-`
`cxray.FileFormat.Results.ModelParameters.ModelParameters`
`readLines()` (`pymcxcray.FileFormat.Results.SpectrumEDS.SpectrumEDS`
`method`), 41
`method`), 41 `regionID` (`pymcxcray.FileFormat.Results.PhirhozRegion.PhirhozRegion`
`readMicroscope()` (`pym-` `attribute`), 38
`cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated` (`pymcxcray.FileFormat.Results.RegionParameters.RegionParam`
`method`), 37 `attribute`), 38
`readPartialSpectraReferenceSection()` `regionID` (`pymcxcray.FileFormat.Results.RegionVolume.RegionVolume`
(`pymcxcray.FileFormat.Results.SpectraEDS.SpectraEDS` `attribute`), 39

`regionID` (`pymcxray.FileFormat.Results.SpectraEDS.SpectraEDSRegion` attribute), 40
`RegionParameters` (class in `pymcxray.FileFormat.Results.RegionParameters`), 38
`regions` (`pymcxray.FileFormat.Specimen.Specimen` attribute), 64
`RegionVolume` (class in `pymcxray.FileFormat.Results.RegionVolume`), 39
`removeInputsFiles` (`pymcxray.Simulation.Simulation` method), 78
`removePreviousFiles` (`pymcxray.BatchFile.BatchFile` method), 74
`removeTempDataPath` (in module `pymcxray.FileFormat.testUtilities`), 65
`resultParametersFilename` (`pymcxray.FileFormat.SimulationInputs.SimulationInputs` attribute), 63
`resultsBasename` (`pymcxray.Simulation.Simulation` attribute), 78
`ResultsParameters` (class in `pymcxray.FileFormat.ResultsParameters`), 62
`revision` (`pymcxray.FileFormat.Version.Version` attribute), 65
`run` (in module `pymcxray.AnalyzeNumberBackgroundWindows`), 73
`run` (in module `pymcxray.AtomData`), 74
`run` (in module `pymcxray.DebugSimulatedSpectrum`), 75
`run` (in module `pymcxray.ElementProperties`), 77
`run` (in module `pymcxray.FileFormat.Results.ElectronTrajectoriesResults`), 35
`run` (in module `pymcxray.FileFormat.Results.exported.DataMap`), 30
`run` (in module `pymcxray.FileFormat.Results.XraySpectraRegionEmitted`), 43
`run` (in module `pymcxray.FileFormat.Results.XraySpectraRegionsEmitted`), 43
`runAtomicNumberSymbol` (in module `pymcxray.ElementProperties`), 77
`runAuCThinFilm` (in module `pymcxray.FileFormat.Results.ElectronTrajectoriesResults`), 35
`runExample` (in module `pymcxray.FileFormat.Results.SpectraEDS`), 40
`runFogging` (in module `pymcxray.FileFormat.Results.ElectronTrajectoriesResults`), 35
`runSpecimen` (`pymcxray.DebugSimulatedSpectrum.DebugSimulatedSpectrum` method), 75
`runVersion1_2_3` (in module `pymcxray.ComparisonModels`), 75
`runVersion1_4_0` (in module `pymcxray.ComparisonModels`), 75
`runVersion1_4_1` (in module `pymcxray.ComparisonModels`), 75

S

`SampleEnergyLossModel` (class in `pymcxray.FileFormat.MCXRayModel`), 59
`save` (`pymcxray.serialization.SerializationH5py.SerializationNumpy` method), 70
`save` (`pymcxray.serialization.SerializationNumpy.SerializationNumpy` method), 71
`save` (`pymcxray.serialization.SerializationNumpy.SerializationNumpyN` method), 71
`save` (`pymcxray.serialization.SerializationNumpy.SerializationNumpyN` method), 71
`save` (`pymcxray.serialization.SerializationNumpy.SerializationNumpyT` method), 71
`save` (`pymcxray.serialization.SerializationPickle.SerializationPickle` method), 71
`saveImage` (`pymcxray.FileFormat.Results.exported.DataMap.DataMa` method), 29
`SerializationNumpy` (class in `pymcxray.serialization.SerializationH5py`), 70
`SerializationNumpy` (class in `pymcxray.serialization.SerializationNumpy`), 71
`SerializationNumpyNPY` (class in `pymcxray.serialization.SerializationNumpy`), 71
`SerializationNumpyNPZ` (class in `pymcxray.serialization.SerializationNumpy`), 71
`SerializationNumpyTxt` (class in `pymcxray.serialization.SerializationNumpy`), 71
`SerializationNumpyTxtGz` (class in `pymcxray.serialization.SerializationNumpy`), 71
`SerializationPickle` (class in `pymcxray.serialization.SerializationPickle`), 71
`setModel` (`pymcxray.FileFormat.MCXRayModel.MCXRayModel` method), 59
`setModelFromString` (`pymcxray.FileFormat.MCXRayModel.MCXRayModel` method), 59
`setParameters` (`pymcxray.Simulation.Simulation` method), 78

setUp () (pymcxray.FileFormat.Results.exported.test_DataMap.TestDataMap method), 30
 setUp () (pymcxray.FileFormat.Results.exported.test_XrayIntensityXY.TestXrayIntensityXY method), 30
 setUp () (pymcxray.FileFormat.Results.test_BaseResults.TestBaseResults method), 44
 setUp () (pymcxray.FileFormat.Results.test_BeamParameters.TestBeamParameters method), 44
 setUp () (pymcxray.FileFormat.Results.test_DetectorParameters.TestDetectorParameters method), 45
 setUp () (pymcxray.FileFormat.Results.test_Dump.TestDump method), 45
 setUp () (pymcxray.FileFormat.Results.test_ElectronParameters.TestElectronParameters method), 46
 setUp () (pymcxray.FileFormat.Results.test_ElectronResults.TestElectronResults method), 46
 setUp () (pymcxray.FileFormat.Results.test_ElementParameters.TestElementParameters method), 46
 setUp () (pymcxray.FileFormat.Results.test_Intersections.TestIntersections method), 47
 setUp () (pymcxray.FileFormat.Results.test_MicroscopeParameters.TestMicroscopeParameters method), 47
 setUp () (pymcxray.FileFormat.Results.test_ModelParameters.TestModelParameters method), 47
 setUp () (pymcxray.FileFormat.Results.test_Phirhoz.TestPhirhoz method), 48
 setUp () (pymcxray.FileFormat.Results.test_PhirhozElements.TestPhirhozElements method), 48
 setUp () (pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristicXrays.TestPhirhozEmittedCharacteristicXrays method), 49
 setUp () (pymcxray.FileFormat.Results.test_PhirhozEmittedCharacteristicXraysThinFilm.TestPhirhozEmittedCharacteristicXraysThinFilm method), 49
 setUp () (pymcxray.FileFormat.Results.test_PhirhozGeneratedPhirhozSerialized.test_PhirhozGeneratedPhirhozSerialized method), 50
 setUp () (pymcxray.FileFormat.Results.test_PhirhozGeneratedCharacteristicThinFilm.TestPhirhozGeneratedCharacteristicThinFilm method), 50
 setUp () (pymcxray.FileFormat.Results.test_PhirhozRegions.TestPhirhozRegions method), 51
 setUp () (pymcxray.FileFormat.Results.test_RegionParameters.TestRegionParameters method), 51
 setUp () (pymcxray.FileFormat.Results.test_RegionVolumes.TestRegionVolumes method), 52
 setUp () (pymcxray.FileFormat.Results.test_SimulationParameters.TestSimulationParameters method), 52
 setUp () (pymcxray.FileFormat.Results.test_Spectra.TestSpectra method), 53
 setUp () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS method), 53
 setUp () (pymcxray.FileFormat.Results.test_Spectrum.TestSpectrum method), 54
 setUp () (pymcxray.FileFormat.Results.test_Tags.TestTags method), 54
 setUp () (pymcxray.FileFormat.Results.test_XrayIntensities.TestXrayIntensities method), 55
 setUp () (pymcxray.FileFormat.Results.test_XraySimulatedSpectraRegion.TestXraySimulatedSpectraRegion method), 55
 setUp () (pymcxray.FileFormat.Results.test_XraySimulatedSpectraSpecimen.TestXraySimulatedSpectraSpecimen method), 55
 setUp () (pymcxray.FileFormat.Results.test_XraySpectraAtomEmittedDetectors.TestXraySpectraAtomEmittedDetectors method), 56
 setUp () (pymcxray.FileFormat.Results.test_XraySpectraSpecimen.TestXraySpectraSpecimen method), 56
 setUp () (pymcxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetectors.TestXraySpectraSpecimenEmittedDetectors method), 57
 setUp () (pymcxray.FileFormat.test_Element.TestElement method), 65
 setUp () (pymcxray.FileFormat.test_ExportedSpectrum.TestExportedSpectrum method), 66
 setUp () (pymcxray.FileFormat.test_FileReaderWriterTools.TestFileReaderWriterTools method), 66
 setUp () (pymcxray.FileFormat.test_MCXRayModel.TestMCXRayModel method), 67
 setUp () (pymcxray.FileFormat.test_MicroscopyParameters.Region.TestRegion method), 67
 setUp () (pymcxray.FileFormat.test_RegionDimensions.TestRegionDimensions method), 68
 setUp () (pymcxray.FileFormat.test_RegionType.TestRegionType method), 69
 setUp () (pymcxray.FileFormat.test_SnrParameters.TestSnrParameters method), 69
 setUp () (pymcxray.FileFormat.test_PhirhozEmittedCharacteristicXrays.TestPhirhozEmittedCharacteristicXrays method), 70
 setUp () (pymcxray.FileFormat.test_PhirhozEmittedCharacteristicXraysThinFilm.TestPhirhozEmittedCharacteristicXraysThinFilm method), 72
 setUp () (pymcxray.FileFormat.test_PhirhozGeneratedSerialized.test_PhirhozGeneratedSerialized method), 72
 setUp () (pymcxray.FileFormat.test_PhirhozGeneratedSerialized.test_PhirhozGeneratedSerialized method), 72
 setUp () (pymcxray.FileFormat.test_PhirhozGeneratedCharacteristicThinFilm.TestPhirhozGeneratedCharacteristicThinFilm method), 81
 setUp () (pymcxray.FileFormat.test_PhirhozRegions.TestPhirhozRegions method), 81
 setUp () (pymcxray.FileFormat.test_RegionParameters.ComparisonModels.TestComparisonModels method), 82
 setUp () (pymcxray.FileFormat.test_RegionVolumes.TestRegionVolumes method), 83
 setUp () (pymcxray.FileFormat.test_SimulationParameters.TestSimulationParameters method), 82
 setUp () (pymcxray.test_SimulationsParameters.TestSimulationsParameters method), 82
 setUp () (pymcxray.test_pymcxray.TestPymcxray method), 73
 setUp () (pymcxray.FileFormat.Results.Phirhoz.Phirhoz attribute), 36
 setUp () (pymcxray.FileFormat.Results.exported.DataMap.DataMap method), 29

simulatedIntensities (pym-
 cxray.FileFormat.Results.XraySimulatedSpectralRegionModel (pym-
 attribute), 42
 Simulation (class in pymcxray.Simulation), 77
 SimulationInputs (class in pym-
 cxray.FileFormat.SimulationInputs), 63
 SimulationParameters (class in pym-
 cxray.FileFormat.Results.SimulationParameters),
 39
 SimulationParameters (class in pym-
 cxray.FileFormat.SimulationParameters),
 63
 simulationParameters (pym-
 cxray.FileFormat.Results.PhirhozGenerated.PhirhozGenerated
 attribute), 37
 simulationParametersFilename (pym-
 cxray.FileFormat.SimulationInputs.SimulationInputs
 attribute), 63
 SimulationsParameters (class in pym-
 cxray.SimulationsParameters), 79
 SimulationsParametersFixed (class in pym-
 cxray.SimulationsParameters), 80
 size (pymcxray.FileFormat.Results.exported.DataMap.DataMap
 attribute), 29
 skirtRatio (pymcxray.FileFormat.Results.ElectronParameters.ElectronParameters
 attribute), 33
 snrFilename (pymcxray.FileFormat.SimulationInputs.SimulationInputs
 attribute), 63
 SnrParameters (class in pym-
 cxray.FileFormat.SnrParameters), 64
 snrType (pymcxray.FileFormat.SnrParameters.SnrParameters
 attribute), 64
 solidAngle_deg (pym-
 cxray.FileFormat.Results.DetectorParameters.DetectorParameters
 attribute), 32
 solidAngle_sr (pymcxray.Simulation.Simulation at-
 tribute), 78
 Specimen (class in pymcxray.FileFormat.Specimen), 64
 specimenFilename (pym-
 cxray.FileFormat.SimulationInputs.SimulationInputs
 attribute), 63
 Spectra (class in pym-
 cxray.FileFormat.Results.Spectra), 39
 SpectraEDS (class in pym-
 cxray.FileFormat.Results.SpectraEDS), 40
 Spectrum (class in pym-
 cxray.FileFormat.Results.Spectrum), 40
 SpectrumEDS (class in pym-
 cxray.FileFormat.Results.SpectrumEDS),
 40
 spectrumEnergyWindowsSize (pym-
 cxray.FileFormat.SnrParameters.SnrParameters
 attribute), 64
 SpectrumInterpolationModel (class in pym-
 cxray.FileFormat.MCXRayModel), 59
 XraySimulatedSpectralRegionModel (pym-
 cxray.FileFormat.SimulationParameters.SimulationParameters
 attribute), 64
 spectrumInterpolationModel (pym-
 cxray.Simulation.Simulation attribute), 78
 surfaceQualityFactor (pym-
 cxray.FileFormat.Results.DetectorParameters.DetectorParameters
 attribute), 32
 symbol (pymcxray.FileFormat.Results.Phirhoz.Phirhoz
 attribute), 36
 symbol (pymcxray.FileFormat.Results.PhirhozElement.PhirhozElement
 attribute), 36
 TagNotFoundError, 41
 takeOffAngle_deg (pym-
 cxray.Simulation.Simulation attribute), 78
 takeoffAngleEffective_deg (pym-
 cxray.FileFormat.Results.DetectorParameters.DetectorParameters
 attribute), 32
 takeoffAngleNormalIncidence_deg (pym-
 cxray.FileFormat.Results.DetectorParameters.DetectorParameters
 attribute), 32
 test_DataMap.TestDataMap (pymcxray.FileFormat.Results.exported.test_DataMap.TestD
 method), 30
 test_XrayIntensity.XrayIntensity (pymcxray.FileFormat.Results.exported.test_XrayIntensityX
 method), 30
 tearDown() (pymcxray.FileFormat.Results.test_BaseResults.TestBaseRes
 method), 44
 tearDown() (pymcxray.FileFormat.Results.test_BeamParameters.TestBea
 method), 44
 tearDown() (pymcxray.FileFormat.Results.test_DetectorParameters.Test
 method), 45
 tearDown() (pymcxray.FileFormat.Results.test_Dump.TestDump
 method), 45
 tearDown() (pymcxray.FileFormat.Results.test_ElectronParameters.Test
 method), 46
 tearDown() (pymcxray.FileFormat.Results.test_ElectronResults.TestElec
 method), 46
 tearDown() (pymcxray.FileFormat.Results.test_ElementParameters.Test
 method), 46
 tearDown() (pymcxray.FileFormat.Results.test_Intersections.TestInterse
 method), 47
 tearDown() (pymcxray.FileFormat.Results.test_MicroscopeParameters.T
 method), 47
 tearDown() (pymcxray.FileFormat.Results.test_ModelParameters.TestM
 method), 48
 tearDown() (pymcxray.FileFormat.Results.test_Phirhoz.TestPhirhoz
 method), 48
 tearDown() (pymcxray.FileFormat.Results.test_PhirhozElement.TestPhir
 method), 48
 tearDown() (pymcxray.FileFormat.Results.test_PhirhozEmittedCharacte
 method), 49

tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 49	tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 72
tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 50	tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 72
tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 50	tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 72
tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 51	tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 81
tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 51	tearDown () (pymcxray.FileFormat.Results.test_PhirhozGeneralizedCharacterization.test_SerializationNumpy.TestSerialization), 81
tearDown () (pymcxray.FileFormat.Results.test_RegionParameters.TestRegionParameters), 51	tearDown () (pymcxray.FileFormat.Results.test_RegionParameters.TestRegionParameters), 82
tearDown () (pymcxray.FileFormat.Results.test_RegionVolume.TestRegionVolume), 52	tearDown () (pymcxray.test_mcxray.Testmcxray), 83
tearDown () (pymcxray.FileFormat.Results.test_SimulationParameters.TestSimulationParameters), 52	tearDown () (pymcxray.test_SimulationParameters.TestSimulationParameters), 82
tearDown () (pymcxray.FileFormat.Results.test_Spectra.TestSpectra), 53	tearDown () (pymcxray.test_SimulationParameters.TestSimulationParameters), 83
tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53	tearDown () (pymcxray.tests.test_pymcxray.TestPymcxray), 73
tearDown () (pymcxray.FileFormat.Results.test_Spectrum.TestSpectrum), 54	tearDown () (pymcxray.test_Spectrum.TestSpectrum), 73
tearDown () (pymcxray.FileFormat.Results.test_Tags.TestTags), 54	tearDown () (pymcxray.test_Tags.TestTags), 73
tearDown () (pymcxray.FileFormat.Results.test_XrayIntensities.TestXrayIntensities), 55	tearDown () (pymcxray.test_XrayIntensities.TestXrayIntensities), 69
tearDown () (pymcxray.FileFormat.Results.test_XraySimulatedSpectraRegion.TestXraySimulatedSpectraRegion), 55	tearDown () (pymcxray.FileFormat.Results.test_Spectra.TestSpectra), 52
tearDown () (pymcxray.FileFormat.Results.test_XraySimulatedSpectraSpecimen.TestXraySimulatedSpectraSpecimen), 55	tearDown () (pymcxray.FileFormat.Results.test_Spectra.TestSpectra), 52
tearDown () (pymcxray.FileFormat.Results.test_XraySpectraAtomEmittedFlatFoiledLines.TestXraySpectraAtomEmittedFlatFoiledLines), 56	tearDown () (pymcxray.FileFormat.Results.test_Spectra.TestSpectra), 53
tearDown () (pymcxray.FileFormat.Results.test_XraySpectraSpecimen.TestXraySpectraSpecimen), 56	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 57
tearDown () (pymcxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetected.TestXraySpectraSpecimenEmittedDetected), 57	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_Element.TestElement), 65	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_ExportedSpectrum.TestExportedSpectrum), 66	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_FileReaderWriterTools.TestFileReaderWriterTools), 66	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_MCXRayModel.TestMCXRayModel), 67	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_Region.TestRegion), 67	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_RegionDimensions.TestRegionDimensions), 68	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_RegionType.TestRegionType), 69	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_SnrParameters.TestSnrParameters), 69	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_Version.TestVersion), 70	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53
tearDown () (pymcxray.FileFormat.test_Version.TestVersion), 70	tearDown () (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS), 53

`cxray.FileFormat.test_RegionType.TestRegionType`
`method), 69`
`test_create_multi_horizontal_layer()`
`(pymcxray.test_Simulation.TestSimulation`
`method), 82`
`test_create_weight_fractions()` (pym-
`cxray.test_Simulation.TestSimulation method),`
`82`
`test_createLineOldVersion()` (pym-
`cxray.FileFormat.test_Element.TestElement`
`method), 65`
`test_createLinesWithoutVersion()` (pym-
`cxray.FileFormat.test_Region.TestRegion`
`method), 67`
`test_createLinesWithVersion()` (pym-
`cxray.FileFormat.test_Region.TestRegion`
`method), 67`
`test_createLineWithKey()` (pym-
`cxray.FileFormat.test_Element.TestElement`
`method), 65`
`test_createRegionDimensions()` (pym-
`cxray.FileFormat.test_RegionDimensions.TestRegionDimensions`
`method), 68`
`test_extractFromLineOldVersion()` (pym-
`cxray.FileFormat.test_Element.TestElement`
`method), 65`
`test_extractFromLines()` (pym-
`cxray.FileFormat.Results.test_Intersections.TestIntersections`
`method), 47`
`test_extractFromLinesWithoutVersion()`
`(pymcxray.FileFormat.test_Region.TestRegion`
`method), 67`
`test_extractFromLinesWithVersion()`
`(pymcxray.FileFormat.test_Region.TestRegion`
`method), 67`
`test_extractFromLineWithKey()` (pym-
`cxray.FileFormat.test_Element.TestElement`
`method), 65`
`test_findTag()` (pym-
`cxray.FileFormat.Results.test_Tags.TestTags`
`method), 54`
`test_FindTestData()` (pym-
`cxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS`
`method), 53`
`test_fromString()` (pym-
`cxray.FileFormat.test_Version.TestVersion`
`method), 70`
`test_getIntensity()` (pym-
`cxray.FileFormat.Results.test_PhirhozEmittedCharacteristicThinFilm`
`method), 49`
`test_getIntensity()` (pym-
`cxray.FileFormat.Results.test_PhirhozGeneratedCharacteristicThinFilm`
`method), 51`
`test_init()` (pymcxray.FileFormat.Results.exported.test_DataMapTestDataMap

`method), 30`
`test_init()` (pymcxray.FileFormat.Results.test_BaseResults.TestBaseR

`method), 44`
`test_loadSave()` (pym-
`cxray.serialization.test_SerializationPickle.TestSerialization`
`method), 72`
`test_loadSaveSerializationNumpy()` (pym-
`cxray.serialization.test_SerializationNumpy.TestSerializationNum`
`method), 72`
`test_loadSaveSerializationNumpyNPY()`
`(pymcxray.serialization.test_SerializationNumpy.TestSerialization`
`method), 72`
`test_loadSaveSerializationNumpyNPZ()`
`(pymcxray.serialization.test_SerializationNumpy.TestSerialization`
`method), 72`
`test_loadSaveSerializationNumpyTxt()`
`(pymcxray.serialization.test_SerializationNumpy.TestSerialization`
`method), 72`
`test_loadSaveSerializationNumpyTxtGz()`
`(pymcxray.serialization.test_SerializationNumpy.TestSerialization`
`method), 72`
`test_loadSaveSerializationNumpyModel()` (pym-
`cxray.FileFormat.test_MCXRayModel.TestMCXRayModel`
`method), 67`
`test_path()` (pymcxray.FileFormat.Results.test_BaseResults.TestBaseR

`method), 44`
`test_read()` (pymcxray.FileFormat.Results.exported.test_DataMap.Test

`method), 30`
`test_read()` (pymcxray.FileFormat.Results.test_Dump.TestDump
`method), 45`
`test_read()` (pymcxray.FileFormat.Results.test_ElectronResults.TestEle

`method), 46`
`test_read()` (pymcxray.FileFormat.Results.test_PhirhozEmittedCharac

`method), 49`
`test_read()` (pymcxray.FileFormat.Results.test_PhirhozEmittedCharac

`method), 49`
`test_read()` (pymcxray.FileFormat.Results.test_PhirhozGenerated.Test

`method), 50`
`test_read()` (pymcxray.FileFormat.Results.test_PhirhozGeneratedChar

`method), 50`
`test_read()` (pymcxray.FileFormat.Results.test_PhirhozGeneratedChar

`method), 51`
`test_read()` (pymcxray.FileFormat.Results.test_Spectra.TestSpectra

`method), 53`
`test_read()` (pymcxray.FileFormat.Results.test_XrayIntensities.TestXra

`method), 55`
`test_read()` (pymcxray.FileFormat.Results.test_XraySimulatedSpectraF

`method), 55`
`test_read()` (pymcxray.FileFormat.Results.test_XraySimulatedSpectraS

`method), 56`
`test_read()` (pymcxray.FileFormat.Results.test_XraySpectraAtomEmitt

`method), 56`
`test_read()` (pymcxray.FileFormat.Results.test_XraySpectraSpecimen.T

test_read() (pymcxray.FileFormat.Results.test_XraySpectrumRegionEmittedDetectedTestXraySpectrumSpecOnchEmittedDetected method), 57

test_read() (pymcxray.FileFormat.test_ExportedSpectrum.TestExportedSpectrum method), 66

test_read() (pymcxray.FileFormat.test_SnrParameters.TestSnrParameters method), 69

test_readFromFile() (pymcxray.FileFormat.test_Version.TestVersion method), 70

test_readFromFile_BadFile() (pymcxray.FileFormat.test_Version.TestVersion method), 70

test_readFromLine() (pymcxray.FileFormat.Results.test_PhirhozElement.TestPhirhozElement method), 48

test_readFromLines() (pymcxray.FileFormat.Results.test_BeamParameters.TestBeamParameters method), 44

test_readFromLines() (pymcxray.FileFormat.Results.test_DetectorParameters.TestDetectorParameters method), 45

test_readFromLines() (pymcxray.FileFormat.Results.test_ElectronParameters.TestElectronParameters method), 46

test_readFromLines() (pymcxray.FileFormat.Results.test_MicroscopeParameters.TestMicroscopeParameters method), 47

test_readFromLines() (pymcxray.FileFormat.Results.test_ModelParameters.TestModelParameters method), 48

test_readFromLines() (pymcxray.FileFormat.Results.test_Phirhoz.TestPhirhoz method), 48

test_readFromLines() (pymcxray.FileFormat.Results.test_PhirhozRegion.TestPhirhozRegion method), 51

test_readFromLines() (pymcxray.FileFormat.Results.test_SimulationParameters.TestSimulationParameters method), 52

test_readPartialSpectraReferenceSection(test_version) (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS.serialization.test_SerializationPickle.TestSerialization method), 53

test_readRegion() (pymcxray.FileFormat.Results.test_Spectra.TestSpectra method), 53

test_readRegionSpectraSection() (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS method), 54

test_readSpecimen() (pymcxray.FileFormat.Results.test_Spectra.TestSpectra method), 53

test_readTestInputSection() (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS method), 54

test_toString() (pymcxray.FileFormat.test_Version.TestVersion method), 70

test_version() (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS.serialization.test_SerializationPickle.TestSerialization method), 73

test_VersionConstants() (pymcxray.FileFormat.test_Version.TestVersion method), 70

test_write() (pymcxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS method), 69

test_writeLine() (pymcxray.FileFormat.test_Version.TestVersion method), 70

TestAtomData (class in pymcxray.test_AtomData), 81

TestBaseResults (class in pymcxray.FileFormat.Results.test_BaseResults),

44		<i>cxray.FileFormat.Results.test_PhirhozElement</i>),
TestBatchFileConsole	(class in pym-	48
	<i>cxray.test_BatchFileConsole</i>), 81	TestPhirhozEmittedCharacteristic
TestBeamParameters	(class in pym-	(class in pym-
	<i>cxray.FileFormat.Results.test_BeamParameters</i>),	<i>cxray.FileFormat.Results.test_PhirhozEmittedCharacteristic</i>),
44		49
TestComparisonModels	(class in pym-	TestPhirhozEmittedCharacteristicThinFilm
	<i>cxray.test_ComparisonModels</i>), 82	(class in pym-
testConstants()	(pym-	<i>cxray.FileFormat.Results.test_PhirhozEmittedCharacteristicThinFilm</i>
	<i>cxray.test_AtomData.TestAtomData</i> method),	49
81		TestPhirhozGenerated (class in pym-
TestDataMap	(class in pym-	<i>cxray.FileFormat.Results.test_PhirhozGenerated</i>),
	<i>cxray.FileFormat.Results.exported.test_DataMap</i>),	50
30		TestPhirhozGeneratedCharacteristic
TestDetectorParameters	(class in pym-	(class in pym-
	<i>cxray.FileFormat.Results.test_DetectorParameters</i>),	<i>cxray.FileFormat.Results.test_PhirhozGeneratedCharacteristic</i>),
45		50
TestDump	(class in pym-	TestPhirhozGeneratedCharacteristicThinFilm
	<i>cxray.FileFormat.Results.test_Dump</i>), 45	(class in pym-
TestElectronParameters	(class in pym-	<i>cxray.FileFormat.Results.test_PhirhozGeneratedCharacteristicThinFilm</i>
	<i>cxray.FileFormat.Results.test_ElectronParameters</i>),	50
45		TestPhirhozRegion (class in pym-
TestElectronResults	(class in pym-	<i>cxray.FileFormat.Results.test_PhirhozRegion</i>),
	<i>cxray.FileFormat.Results.test_ElectronResults</i>),	51
46		TestPymcxray (class in pym-
TestElement	(class in pym-	<i>cxray.tests.test_pymcxray</i>), 73
	<i>cxray.FileFormat.test_Element</i>), 65	TestRegion (class in pym-
TestElementParameters	(class in pym-	<i>cxray.FileFormat.test_Region</i>), 67
	<i>cxray.FileFormat.Results.test_ElementParameters</i>),	TestRegionDimensions (class in pym-
46		<i>cxray.FileFormat.test_RegionDimensions</i>),
TestExportedSpectrum	(class in pym-	67
	<i>cxray.FileFormat.test_ExportedSpectrum</i>),	TestRegionParameters (class in pym-
66		<i>cxray.FileFormat.Results.test_RegionParameters</i>),
TestFileReaderWriterTools	(class in pym-	51
	<i>cxray.FileFormat.test_FileReaderWriterTools</i>),	TestRegionType (class in pym-
66		<i>cxray.FileFormat.test_RegionType</i>), 68
TestIntersections	(class in pym-	TestRegionVolume (class in pym-
	<i>cxray.FileFormat.Results.test_Intersections</i>),	<i>cxray.FileFormat.Results.test_RegionVolume</i>),
47		52
Testmcxray	(class in pymcxray.test_mcxray), 83	TestSerialization (class in pym-
TestMCXRayModel	(class in pym-	<i>cxray.serialization.test_SerializationPickle</i>),
	<i>cxray.FileFormat.test_MCXRayModel</i>), 66	72
TestMicroscopeParameters	(class in pym-	TestSerializationH5py (class in pym-
	<i>cxray.FileFormat.Results.test_MicroscopeParameters</i>),	<i>cxray.serialization.test_SerializationH5py</i>),
47		72
TestModelParameters	(class in pym-	TestSerializationNumpy (class in pym-
	<i>cxray.FileFormat.Results.test_ModelParameters</i>),	<i>cxray.serialization.test_SerializationNumpy</i>),
47		72
testOpenFile()	(pym-	TestSimulation (class in pymcxray.test_Simulation),
	<i>cxray.FileFormat.Results.exported.test_XrayIntensityXY.TestXrayIntensityXY</i>	80
	method), 30	TestSimulationParameters (class in pym-
TestPhirhoz	(class in pym-	<i>cxray.FileFormat.Results.test_SimulationParameters</i>),
	<i>cxray.FileFormat.Results.test_Phirhoz</i>), 48	52
TestPhirhozElement	(class in pym-	TestSimulationsParameters (class in pym-

`cxray.test_SimulationsParameters)`, 82
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.exported.test_DataMap.TestDataMap` (pym-
`method)`, 30 `method)`, 51
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.exported.test_XrayIntensityXY.TestXrayIntensityXY` (pym-
`method)`, 31 `method)`, 51
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_BaseResults.TestBaseResults` (pym-
`method)`, 44 `cxray.FileFormat.Results.test_PhirhozGeneratedCharacteristicTh`
`method)`, 51
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_BeamParameters.TestBeamParameters` (pym-
`method)`, 44 `cxray.FileFormat.Results.test_PhirhozRegion.TestPhirhozRegion`
`method)`, 52
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_DetectorParameters.TestDetectorParameters` (pym-
`method)`, 45 `cxray.FileFormat.Results.test_SimulationParameters.TestSimulationParameters`
`method)`, 52
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_Dump.TestDump` (pym-
`method)`, 45 `cxray.FileFormat.Results.test_Spectra.TestSpectra`
`method)`, 53
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_ElectronParameters.TestElectronParameters` (pym-
`method)`, 46 `cxray.FileFormat.Results.test_SpectraEDS.TestSpectraEDS`
`method)`, 53
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_ElectronResults.TestElectronResults` (pym-
`method)`, 46 `cxray.FileFormat.Results.test_Spectrum.TestSpectrum`
`method)`, 54
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_ElementParameters.TestElementParameters` (pym-
`method)`, 46 `cxray.FileFormat.Results.test_Tags.TestTags`
`method)`, 54
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_Intersections.TestIntersections` (pym-
`method)`, 47 `cxray.FileFormat.Results.test_XrayIntensities.TestXrayIntensities`
`method)`, 55
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_MicroscopeParameters.TestMicroscopeParameters` (pym-
`method)`, 47 `cxray.FileFormat.Results.test_XraySimulatedSpectraRegion.TestXraySimulatedSpectraRegion`
`method)`, 55
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_ModelParameters.TestModelParameters` (pym-
`method)`, 48 `cxray.FileFormat.Results.test_XraySimulatedSpectraSpecimen.TestXraySimulatedSpectraSpecimen`
`method)`, 56
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_Phirhoz.TestPhirhoz` (pym-
`method)`, 48 `cxray.FileFormat.Results.test_XraySpectraAtomEmittedDetectedL`
`method)`, 56
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_PhirhozElement.TestPhirhozElement` (pym-
`method)`, 48 `cxray.FileFormat.Results.test_XraySpectraSpecimen.TestXraySpectraSpecimen`
`method)`, 56
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_PhirhozEmittedCharacteristic.TestPhirhozEmittedCharacteristic` (pym-
`method)`, 49 `cxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetectedL`
`method)`, 57
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_PhirhozEmittedCharacteristicThinFilm.TestPhirhozEmittedCharacteristicThinFilm` (pym-
`method)`, 49 `cxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetectedL`
`method)`, 65
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_PhirhozGenerated.TestPhirhozGenerated` (pym-
`method)`, 50 `cxray.FileFormat.test_ExportedSpectrum.TestExportedSpectrum`
`method)`, 66
`testSkeleton()` (pym- `testSkeleton()` (pym-
`cxray.FileFormat.Results.test_PhirhozGeneratedCharacteristic.TestPhirhozGeneratedCharacteristic` (pym-
`method)`, 50 `cxray.FileFormat.test_ExportedSpectrum.TestExportedSpectrum`
`method)`, 66

`method)`, 66
`testSkeleton()` (pym- `TestTags` (class in pym-
`cxray.FileFormat.test_MCXRayModel.TestMCXRayModel` `cxray.FileFormat.Results.test_Tags)`, 54
`method)`, 67
`testSkeleton()` (pym- `TestVersion` (class in pym-
`cxray.FileFormat.test_Version.TestVersion` `cxray.FileFormat.test_Version)`, 70
`method)`, 67
`testSkeleton()` (pym- `TestXrayIntensities` (class in pym-
`cxray.FileFormat.test_Region.TestRegion` `cxray.FileFormat.Results.test_XrayIntensities)`,
`method)`, 67
`testSkeleton()` (pym- `TestXrayIntensityXY` (class in pym-
`cxray.FileFormat.test_RegionDimensions.TestRegionDimensions` `cxray.FileFormat.Results.exported.test_XrayIntensityXY)`,
`method)`, 68
`testSkeleton()` (pym- `TestXraySimulatedSpectraRegion`
`cxray.FileFormat.test_RegionType.TestRegionType` (class in pym-
`method)`, 69
`testSkeleton()` (pym- `cxray.FileFormat.Results.test_XraySimulatedSpectraRegion)`,
`cxray.FileFormat.test_SnrParameters.TestSnrParameters` 55
`method)`, 69
`testSkeleton()` (pym- `TestXraySimulatedSpectraSpecimen`
`cxray.FileFormat.test_Version.TestVersion` (class in pym-
`method)`, 70
`testSkeleton()` (pym- `TestXraySpectraAtomEmittedDetectedLines`
`cxray.serialization.test_SerializationH5py.TestSerializationH5py` (class in pym-
`method)`, 72
`testSkeleton()` (pym- `TestXraySpectraAtomEmittedDetectedLines`
`cxray.serialization.test_SerializationNumpy.TestSerializationNumpy` (class in pym-
`method)`, 72
`testSkeleton()` (pym- `TestXraySpectraSpecimen` (class in pym-
`cxray.serialization.test_SerializationPickle.TestSerializationPickle` `cxray.FileFormat.Results.test_XraySpectraSpecimen)`,
`method)`, 72
`testSkeleton()` (pym- `TestXraySpectraSpecimenEmittedDetected`
`cxray.test_AtomData.TestAtomData` (class in pym-
`method)`, 72
`testSkeleton()` (pym- `cxray.FileFormat.Results.test_XraySpectraSpecimenEmittedDetected`
`cxray.test_AtomData.TestAtomData` `method)`, 57
`81`
`testSkeleton()` (pym- `thicknessAir_um` (pym-
`cxray.test_BatchFileConsole.TestBatchFileConsole` `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method)`, 81
`testSkeleton()` (pym- `thicknessAlWindow_um` (pym-
`cxray.test_ComparisonModels.TestComparisonModels` `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method)`, 82
`testSkeleton()` (pym- `thicknessBeWindow_um` (pym-
`cxray.test_mcxdxray.Testmcxdxray` `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method)`, 83
`testSkeleton()` (pym- `thicknessH2O_um` (pym-
`cxray.test_Simulation.TestSimulation` `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method)`, 82
`testSkeleton()` (pym- `thicknessMoxtek_um` (pym-
`cxray.test_SimulationsParameters.TestSimulationsParameters` `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`method)`, 83
`TestSnrParameters` (class in pym- `thicknessOil_um` (pym-
`cxray.FileFormat.test_SnrParameters)`, 69
`TestSpectra` (class in pym- `thicknessTiWindow_um` (pym-
`cxray.FileFormat.Results.test_Spectra)`, 52
`TestSpectraEDS` (class in pym- `cxray.FileFormat.Results.DetectorParameters.DetectorParameters`
`cxray.FileFormat.Results.test_SpectraEDS)`,
53
`TestSpectrum` (class in pym- `throughRatio` (pym-
`cxray.FileFormat.Results.test_Spectrum)`,
attribute), 33

tiltAngle_deg (pym- cxray.FileFormat.MCXRayModel.XRayCSCharacteristicModel
cxray.FileFormat.Results.BeamParameters.BeamParameters attribute), 60
attribute), 31 TYPE_CHANTLER2005 (pym-
time_s (pymcxray.FileFormat.MicroscopeParameters.MicroscopeParameters attribute), 60
attribute), 60 cxray.FileFormat.MCXRayModel.MassAbsorptionCoefficientModel
attribute), 59
time_s (pymcxray.Simulation.Simulation attribute), 78 TYPE_COPY (pymcxray.FileFormat.MCXRayModel.SpectrumInterpolation
attribute), 59
title (pymcxray.FileFormat.SimulationInputs.SimulationInputs attribute), 59
attribute), 63 TYPE_DING (pymcxray.FileFormat.MCXRayModel.XRayCSBremsstrahlungModel
attribute), 59
toString() (pymcxray.FileFormat.Version.Version attribute), 59
method), 65 TYPE_GAUVIN (pymcxray.FileFormat.MCXRayModel.AtomCollisionScreeningModel
attribute), 58
total_1_ekeVsr (pym- attribute), 58
cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted FileFormat.MCXRayModel.XRayCSBremsstrahlungModel
attribute), 43 attribute), 59
totals (pymcxray.FileFormat.Results.XraySimulatedSpectraSpecimen.XraySimulatedSpectraSpecimen (pym-
attribute), 42 cxray.FileFormat.MCXRayModel.AtomCrossSectionModel
attribute), 58
totals (pymcxray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen attribute), 43
attribute), 43 TYPE_HEINRICH_DATA (pym-
attribute), 44 TYPE_HEINRICH_PARAMETERIZATION (pym-
attribute), 59
totalValue_1_ekeVsr() (pym- TYPE_HEINRICH_PARAMETERIZATION (pym-
cxray.FileFormat.Results.XraySpectraRegionEmitted.XraySpectraRegionEmitted FileFormat.MCXRayModel.MassAbsorptionCoefficientModel
method), 43 attribute), 59
Trajectory (class in pym- TYPE_HENKE (pymcxray.FileFormat.MCXRayModel.MassAbsorptionCoefficientModel
cxray.FileFormat.Results.ElectronTrajectoriesResults), attribute), 59
34 TYPE_HENOC_MAURICE (pym-
trajectoryType (pym- cxray.FileFormat.MCXRayModel.AtomCollisionScreeningModel
cxray.FileFormat.Results.ElectronTrajectoriesResults.Trajectory attribute), 58
attribute), 34 TYPE_HENOC_MAURICE (pym-
TYPE_BETHE (pymcxray.FileFormat.MCXRayModel.AtomEnergyLossModel FileFormat.MCXRayModel.AtomCrossSectionScreeningModel
attribute), 58 attribute), 58
TYPE_BETHE (pymcxray.FileFormat.MCXRayModel.RegionEnergyLossModel MAURICE (pym-
attribute), 59 cxray.FileFormat.MCXRayModel.AtomScreeningModel
attribute), 59
TYPE_BETHE_HEITLER (pym- attribute), 59
cxray.FileFormat.MCXRayModel.XRayCSBremsstrahlungModel LUO (pym-
attribute), 59 cxray.FileFormat.MCXRayModel.AtomMeanIonizationPotentialModel
attribute), 58
TYPE_BETHE_JOY_LUO (pym- attribute), 58
cxray.FileFormat.MCXRayModel.RegionEnergyLossModel JOY_LUO_KGAUVIN (pym-
attribute), 59 cxray.FileFormat.MCXRayModel.RegionEnergyLossModel
attribute), 59
TYPE_BETHE_JOY_LUO (pym- attribute), 59
cxray.FileFormat.MCXRayModel.SampleEnergyLossModel JOY_LUO_MONSEL (pym-
attribute), 59 cxray.FileFormat.MCXRayModel.RegionEnergyLossModel
attribute), 59
TYPE_BETHE_RELATIVISTIC (pym- attribute), 59
cxray.FileFormat.MCXRayModel.RegionEnergyLossModel ANAYA_OKAYAMA (pym-
attribute), 59 cxray.FileFormat.MCXRayModel.AtomElectronRangeModel
attribute), 58
TYPE_BOTE2009 (pym- attribute), 58
cxray.FileFormat.MCXRayModel.XRayCSCharacteristicModel PATRICK_WIEDMAN (pym-
attribute), 60 cxray.FileFormat.MCXRayModel.XRayCSBremsstrahlungModel
attribute), 59
TYPE_BROWNING (pym- attribute), 59
cxray.FileFormat.MCXRayModel.AtomCollisionScreeningModel TYPE_LINEAR (pymcxray.FileFormat.MCXRayModel.SpectrumInterpolation
attribute), 58 attribute), 59
TYPE_BROWNING (pym- TYPE_LINEAR_DOUBLE (pym-
cxray.FileFormat.MCXRayModel.AtomCrossSectionModel cxray.FileFormat.MCXRayModel.SpectrumInterpolationModel
attribute), 58 attribute), 59
TYPE_CASTANI1982 (pym- TYPE_RUTHERFORD (pym-

`cxray.FileFormat.MCXRayModel.AtomCollisionModel` (class in `pymcrray.FileFormat.SimulationParameters.SimulationParameters`), 58

`cxray.FileFormat.MCXRayModel.SpectrumInterpolationModel` (class in `pymcrray.FileFormat.SimulationParameters.SimulationParameters`), 59

`TYPE_SPLINE` (`pymcrray.FileFormat.MCXRayModel.SpectrumInterpolationModel` attribute), 59

`TYPE_SPLINE_BATCH` (`pymcrray.FileFormat.MCXRayModel.SpectrumInterpolationModel` attribute), 59

`TYPE_SPLINE_POINT` (`pymcrray.FileFormat.MCXRayModel.SpectrumInterpolationModel` attribute), 59

`cxray.FileFormat.Results.ElectronResults.ElectronResults` (class in `pymcrray.FileFormat.Results.ElectronResults`), 34

`cxray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults` (class in `pymcrray.FileFormat.Results.ElectronTrajectoriesResults`), 34

`cxray.FileFormat.Results.SimulationParameters.SimulationParameters` (class in `pymcrray.FileFormat.Results.SimulationParameters`), 36

`cxray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic` (class in `pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic`), 38

`cxray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic` (class in `pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic`), 38

`cxray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm` (class in `pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm`), 38

`cxray.SimulationsParameters.SimulationsParameters` (class in `pymcrray.SimulationsParameters`), 80

`Version` (class in `pymcrray.FileFormat.Version`), 65

`version` (`pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters` attribute), 61

`version` (`pymcrray.FileFormat.Models.Models` attribute), 61

`version` (`pymcrray.FileFormat.ResultsParameters.ResultsParameters` attribute), 63

`version` (`pymcrray.FileFormat.SimulationInputs.SimulationInputs` attribute), 63

`version` (`pymcrray.FileFormat.SimulationParameters.SimulationParameters` attribute), 64

`version` (`pymcrray.FileFormat.Specimen.Specimen` attribute), 64

`voxelSimplification` (`pymcrray.FileFormat.SimulationParameters.SimulationParameters` attribute), 64

`weightFraction` (`pymcrray.FileFormat.Results.PhirhozElement.PhirhozElement` attribute), 36

`write()` (`pymcrray.BatchFile.BatchFile` method), 74

`write()` (`pymcrray.BatchFileConsole.BatchFileConsole` method), 74

`write()` (`pymcrray.FileFormat.MicroscopeParameters.MicroscopeParameters` method), 61

`write()` (`pymcrray.FileFormat.Models.Models` method), 61

`write()` (`pymcrray.FileFormat.ResultsParameters.ResultsParameters` method), 63

`write()` (`pymcrray.FileFormat.SimulationInputs.SimulationInputs` method), 63

`write_hdf5()` (`pymcrray.FileFormat.SimulationParameters.SimulationParameters` method), 64

`write_hdf5()` (`pymcrray.FileFormat.Results.ElectronResults.ElectronResults` method), 34

`write_hdf5()` (`pymcrray.FileFormat.Results.ElectronTrajectoriesResults.ElectronTrajectoriesResults` method), 34

`write_hdf5()` (`pymcrray.FileFormat.Results.PhirhozEmittedCharacteristic.PhirhozEmittedCharacteristic` method), 38

`write_hdf5()` (`pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristic.PhirhozGeneratedCharacteristic` method), 38

`write_hdf5()` (`pymcrray.FileFormat.Results.PhirhozGeneratedCharacteristicThinFilm.PhirhozGeneratedCharacteristicThinFilm` method), 38

`write_hdf5()` (`pymcrray.SimulationsParameters.SimulationsParameters` method), 80

`write_hdf5()` (`pymcrray.FileFormat.Results.XrayIntensities.XrayIntensities` method), 41

`write_hdf5()` (`pymcrray.FileFormat.Results.XraySpectraRegionsEmitted.XraySpectraRegionsEmitted` method), 43

`write_hdf5()` (`pymcrray.FileFormat.Results.XraySpectraSpecimen.XraySpectraSpecimen` method), 43

`write_hdf5()` (`pymcrray.FileFormat.Results.XraySpectraSpecimenEmittedDetected.XraySpectraSpecimenEmittedDetected` method), 44

`writeLine()` (`pymcrray.FileFormat.Version.Version` method), 65

`x_A` (`pymcrray.FileFormat.Results.ElectronTrajectoriesResults.Collision` attribute), 34

`XRayCSBremsstrahlungModel` (class in `pymcrray.FileFormat.MCXRayModel`), 59

`XRayCSCharacteristicModel` (class in `pymcrray.FileFormat.MCXRayModel`), 59

`XrayIntensities` (class in `pymcrray.FileFormat.Results.XrayIntensities`), 41

`XrayIntensityXY` (class in `pymcrray.FileFormat.Results.exported.XrayIntensityXY`), 30

`XraySimulatedSpectraRegion` (class in `pymcrray.FileFormat.Results.XraySimulatedSpectraRegion`), 41

`XraySimulatedSpectraSpecimen` (class in `pymcrray.FileFormat.Results.XraySimulatedSpectraSpecimen`), 41

42

XraySpectraAtomEmittedDetectedLines
(class in pym-
cxray.FileFormat.Results.XraySpectraAtomEmittedDetectedLines),

42

XraySpectraRegionEmitted (class in pym-
cxray.FileFormat.Results.XraySpectraRegionEmitted),

42

XraySpectraRegionsEmitted (class in pym-
cxray.FileFormat.Results.XraySpectraRegionsEmitted),

43

XraySpectraSpecimen (class in pym-
cxray.FileFormat.Results.XraySpectraSpecimen),

43

XraySpectraSpecimenEmittedDetected
(class in pym-
cxray.FileFormat.Results.XraySpectraSpecimenEmittedDetected),

43

Y

y_A (pymcxray.FileFormat.Results.ElectronTrajectoriesResults.Collision
attribute), 34

Z

z_A (pymcxray.FileFormat.Results.ElectronTrajectoriesResults.Collision
attribute), 34