API Documentation

API Documentation

April 2, 2013

Contents

1 Package fabio

FabIO module

Date: 02/04/2013

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: Jerome.Kieffer@ESRF.eu

License: GPLv3+

1.1 Modules

- **fabioutils**: General purpose utilities functions for fabio
  (Section 20, p. 67)
- **file_series**:
  Authors: .....
  (Section 21, p. 79)
- **fit2dmaskimage**: Author: Andy Hammersley, ESRF Translation into python/fabio: Jon Wright, ESRF
  (Section 22, p. 88)
- **fit2dspreadsheetimage**: Read the fit2d ascii image output...
  (Section 23, p. 90)
- **kcdimage**: Authors: Jerome Kieffer, ESRF email:jerome.kieffer@esrf.fr
  (Section 24, p. 92)
- **mar345_IO**: New Cython version of mar345_io for preparing the migration to Python3
  (Section 25, p. 94)
- **mar345image**:
  Authors: .....
  (Section 26, p. 95)
- **marccdimage**:
  Authors: .....
  (Section 27, p. 97)
- **openimage**:
  Authors: Henning O.
  (Section 28, p. 100)
- **pilatusimage**:
  Authors: .....
  (Section 29, p. 101)
- **pnmimage**:
  Authors: Henning O.
  (Section 30, p. 103)
- **readbytestream**: Reads a bytestream
  (Section 31, p. 106)
- **tifimage**: FabIO class for dealing with TIFF images.
  (Section 32, p. 107)
- **xsdimage**: Authors: Jérôme Kieffer, ESRF email:jerome.kieffer@esrf.fr
  (Section 33, p. 112)

1.2 Variables

Name	Description
__status__	**Value:** `'stable'`
version	**Value:** `'0.1.2'`
__package__	**Value:** `'fabio'`

2 Module fabio.GEimage

2.1 Functions

demo()

2.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("GEimage")`
GE_HEADER_INFO	**Value:** `[('ImageFormat', 10, None), ('VersionOfStandardHeader', 2...`
__package__	**Value:** `'fabio'`

2.3 Class GEimage

object
fabio.fabioimage.fabioimage
fabio.GEimage.GEimage

2.3.1 Methods

read(*self*, *fname*, *frame*=`None`)

Read in header into self.header and the data into self.data

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*, *force_type*=<`type 'numpy.uint16'`>)

Not yet implemented

Overrides: fabio.fabioimage.fabioimage.write

getframe(*self*, *num*)

Returns a frame as a new fabioimage object

Overrides: fabio.fabioimage.fabioimage.getframe

next(*self*)

Get the next image in a series as a fabio image

Overrides: fabio.fabioimage.fabioimage.next

previous(*self*)

Get the previous image in a series as a fabio image

Overrides: fabio.fabioimage.fabioimage.previous

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.3.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

3 Module fabio.GEimage_old

```
Reads the header from a GE a-Si Angio Detector

Authors: Henning O. Sorensen & Erik Knudsen
         Center for Fundamental Research: Metal Structures in Four Dimensions
         Risoe National Laboratory
         Frederiksborgvej 399
         DK-4000 Roskilde
         email:erik.knudsen@risoe.dk

        + Jon Wright, ESRF

        The header information has been taken from the script read_GEaSi_data.py
        by
        Antonino Miceli
        Thu Jan  4 13:46:31 CST 2007
```

3.1 Variables

Name	Description
__package__	**Value:** `'fabio'`

3.2 Class GEimage

object

fabio.fabioimage.fabioimage

fabio.GEimage_old.GEimage

3.2.1 Methods

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

3.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

4 Module fabio.HiPiCimage

```
Authors: Henning O. Sorensen & Erik Knudsen
         Center for Fundamental Research: Metal Structures in Four Dimensions
         Risoe National Laboratory
         Frederiksborgvej 399
         DK-4000 Roskilde
         email:erik.knudsen@risoe.dk

        + Jon Wright, ESRF

Information about the file format from Masakatzu Kobayashi is highly appreciated
```

4.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("HiPiCimage")`
__package__	**Value:** `'fabio'`

4.2 Class HiPiCimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.HiPiCimage.HiPiCimage**

Read HiPic images e.g. collected with a Hamamatsu CCD camera

4.2.1 Methods

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

4.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

5 Module fabio.OXDimage

```
Reads Oxford Diffraction Sapphire 3 images

Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jon Wright, Jérôme Kieffer & Gaël Goret:
  European Synchrotron Radiation Facility;
  Grenoble (France)
```

5.1 Variables

Name	Description
__doc__	**Value:** "...
logger	**Value:** logging.getLogger("OXDimage")
rad2deg	**Value:** <ufunc 'rad2deg'>
deg2rad	**Value:** <ufunc 'deg2rad'>
DETECTOR_TYPES	**Value:** {0: 'Sapphire/KM4CCD (1x1: 0.06mm, 2x2: 0.12mm)', 1: 'Sap...
DEFAULT_HEADERS	**Value:** {'ASCII Section size in Byte': 256, 'Compression': 'TY1',...
__package__	**Value:** 'fabio'

5.2 Class OXDimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.OXDimage.OXDimage**

Oxford Diffraction Sapphire 3 images reader/writer class

5.2.1 Methods

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

Write Oxford diffraction images: this is still beta

Parameters

`fname`: output filename

Overrides: fabio.fabioimage.fabioimage.write

getCompressionRatio(*self*)

calculate the compression factor obtained vs raw data

checkData(*data*=`None`)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

5.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	

continued on next page

Name	Description
classname	
Inherited from object __class__	

5.3 Class Section

object
└
fabio.OXDimage.Section

Small helper class for writing binary headers

5.3.1 Methods

__init__(*self*, *size*, *dictHeader*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Parameters

`size:` size of the header section in bytes

`dictHeader:` headers of the image

Overrides: object.__init__

__repr__(*self*)

repr(x)

Overrides: object.__repr__ extit(inherited documentation)

getSize(*self*, *dtype*)

setData(*self*, *key*, *offset*, *dtype*, *default*=`None`)

Parameters

`offset:` int, starting position in the section

`key:` name of the header key

`dtype:` type of the data to insert (defines the size!)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

5.3.2 Properties

Name	Description
Inherited from object	
__class__	

6 Module fabio.TiffIO

Author: V.A. Sole - ESRF Data Analysis

6.1 Variables

Name	Description
__revision__	**Value:** 1501
DEBUG	**Value:** 0
ALLOW_MULTIPLE_STRIPS	**Value:** False
TAG_ID	**Value:** {256: 'NumberOfColumns', 257: 'NumberOfRows', 258: 'BitsP...
TAG_NUMBER_OF_COLUMNS	**Value:** 256
TAG_NUMBER_OF_ROWS	**Value:** 257
TAG_BITS_PER_SAMPLE	**Value:** 258
TAG_PHOTOMETRIC_INTERPRETATION	**Value:** 262
TAG_COMPRESSION	**Value:** 259
TAG_IMAGE_DESCRIPTION	**Value:** 270
TAG_STRIP_OFFSETS	**Value:** 273
TAG_ROWS_PER_STRIP	**Value:** 278
TAG_STRIP_BYTE_COUNTS	**Value:** 279
TAG_SOFTWARE	**Value:** 305
TAG_DATE	**Value:** 306
TAG_COLORMAP	**Value:** 320
TAG_SAMPLE_FORMAT	**Value:** 339
FIELD_TYPE	**Value:** {1: ('BYTE', 'B'), 2: ('ASCII', 's'), 3: ('SHORT', 'H'), ...
FIELD_TYPE_OUT	**Value:** {'B': 1, 'H': 3, 'I': 4, 'II': 5, 'b': 6, 'd': 12, 'f': 1...
SAMPLE_FORMAT_UINT	**Value:** 1
SAMPLE_FORMAT_INT	**Value:** 2
SAMPLE_FORMAT_FLOAT	**Value:** 3

continued on next page

Name	Description
SAMPLE_FORMAT_VOID	**Value:** `4`
SAMPLE_FORMAT_COMPLEXINT	**Value:** `5`
SAMPLE_FORMAT_COMPLEXIEEEFP	**Value:** `6`
__package__	**Value:** `'fabio'`

6.2 Class TiffIO

object
└ **fabio.TiffIO.TiffIO**

6.2.1 Methods

__init__(*self*, *filename*, *mode*=`None`, *cache_length*=`20`, *mono_output*=`False`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

getNumberOfImages(*self*)

getImageFileDirectories(*self*, *fd*=`None`)

getData(*self*, *nImage*, ***kw*)

getImage(*self*, *nImage*)

getInfo(*self*, *nImage*, ***kw*)

writeImage(*self*, *image0*, *info*=`None`, *software*=`None`, *date*=`None`)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

6.2.2 Properties

Name	Description
Inherited from object	
__class__	

7 Module fabio.adscimage

Authors: Henning O. Sorensen & Erik Knudsen
Center for Fundamental Research: Metal Structures in Four Dimensions
Risoe National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde
email:erik.knudsen@risoe.dk

+ mods for fabio by JPW

7.1 Functions

test()
testcase

7.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("adscimage")`
__package__	**Value:** `'fabio'`

7.3 Class adscimage

object
fabio.fabioimage.fabioimage
fabio.adscimage.adscimage

Read an image in ADSC format (quite similar to edf?)

7.3.1 Methods

__init__(*self*, **args*, ***kwargs*)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

read in the file

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

Write adsc format

Overrides: fabio.fabioimage.fabioimage.write

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

7.3.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

8 Module fabio.binaryimage

Authors: Gael Goret, Jerome Kieffer, ESRF, France Emails: gael.goret@esrf.fr, jerome.kieffer@esrf.fr

Binary files images are simple none-compressed 2D images only defined by their : data-type, dimensions, byte order and offset

This simple library has been made for manipulating exotic/unknown files format.

Version: 17 Apr 2012

Authors: Ga\xc3\xabl Goret, J\xc3\xa9r\xc3\xb4me Kieffer

Contact: gael.goret@esrf.fr

License: GPLv3+

8.1 Variables

Name	Description
__doc__	**Value:** `...`
logger	**Value:** `logging.getLogger("binaryimage")`
__package__	**Value:** `'fabio'`

8.2 Class binaryimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.binaryimage.binaryimage**

This simple library has been made for manipulating exotic/unknown files format.

Binary files images are simple none-compressed 2D images only defined by their : data-type, dimensions, byte order and offset

8.2.1 Methods

__init__(*self*, **args*, ***kwargs*)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

swap_needed(*endian*)

Decide if we need to byteswap

read(*self*, *fname*, *dim1*, *dim2*, *offset*=0, *bytecode*='int32', *endian*='<')

Read a binary image Parameters : fname, dim1, dim2, offset, bytecode, endian fname : file name :
str dim1,dim2 : image dimensions : int offset : size of the : int bytecode among : "int8","int16","int32","int64","uint8","uint16","uint32","uint64","float32","float64",... endian among short or long endian ("<" or ">")

Overrides: fabio.fabioimage.fabioimage.read

estimate_offset_value(*self*, *fname*, *dim1*, *dim2*, *bytecode*='int32')

Estimates the size of a file

write(*self*, *fname*)

To be overwritten - write the file

Overrides: fabio.fabioimage.fabioimage.write extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

8.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	

continued on next page

Name	Description
classname	
Inherited from object __class__	

9 Module fabio.bruker100image

9.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("bruker100image")`
__package__	**Value:** `'fabio'`

9.2 Class bruker100image

object
└
fabio.fabioimage.fabioimage
└
fabio.brukerimage.brukerimage
└
fabio.bruker100image.bruker100image

9.2.1 Methods

toPIL16(*self*, *filename*=`None`)

Convert to Python Imaging Library 16 bit greyscale image

FIXME - this should be handled by the libraries now

Overrides: fabio.fabioimage.fabioimage.toPIL16 extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

Read in and unpack the pixels (including overflow table

Overrides: fabio.fabioimage.fabioimage.read extit(inherited documentation)

Inherited from fabio.brukerimage.brukerimage(Section 10.3)

write(), write2()

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(),

make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

9.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

9.2.3 Class Variables

Name	Description
Inherited from fabio.brukerimage.brukerimage (Section 10.3)	
__headerstring__	

10 Module fabio.brukerimage

Authors: Henning O. Sorensen & Erik Knudsen
Center for Fundamental Research: Metal Structures in Four Dimensions
Risoe National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde
email:erik.knudsen@risoe.dk

Based on: openbruker,readbruker, readbrukerheader functions in the opendata module of ImageD11 written by Jon Wright, ESRF, Grenoble, France

10.1 Functions

test()

a testcase

10.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("brukerimage")`
__package__	**Value:** `'fabio'`

10.3 Class brukerimage

object
fabio.fabioimage.fabioimage
fabio.brukerimage.brukerimage

Known Subclasses: fabio.bruker100image.bruker100image

Read and eventually write ID11 bruker (eg smart6500) images

10.3.1 Methods

read(*self*, *fname*, *frame*=`None`)

Read in and unpack the pixels (including overflow table

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

```
Writes the image as EDF

FIXME:  this should call edfimage.write if that is wanted?
        obj = edfimage(data = self.data, header = self.header)
        obj.write(fname)
        or maybe something like: edfimage.write(self, fname)
```

Overrides: fabio.fabioimage.fabioimage.write

write2(*self*, *fname*)

FIXME: what is this?

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

10.3.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

10.3.3 Class Variables

Name	Description
__headerstring__	**Value:** ’ ’

11 Module fabio.byte_offset

Authors: Jerome Kieffer, ESRF Email: jerome.kieffer@esrf.eu

Cif Binary Files images are 2D images written by the Pilatus detector and others. They use a modified (simplified) byte-offset algorithm. This file contains the decompression function from a string to an int64 numpy array.

This is Cython: convert it to pure C then compile it with gcc $ cython byte_offset.pyx

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: jerome.kieffer@esrf.eu

Copyright: 2010, European Synchrotron Radiation Facility, Grenoble, France

License: GPLv3+

11.1 Variables

Name	Description
__package__	**Value:** `'fabio'`
__test__	**Value:** {}

12 Module fabio.cbfimage

```
Authors: Jérôme Kieffer, ESRF
         email:jerome.kieffer@esrf.fr

Cif Binary Files images are 2D images written by the Pilatus detector and others.
They use a modified (simplified) byte-offset algorithm.

CIF is a library for manipulating Crystallographic information files and tries
to conform to the specification of the IUCR
```

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: jerome.kieffer@esrf.eu

Copyright: European Synchrotron Radiation Facility, Grenoble, France

License: GPLv3+

12.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("cbfimage")`
DATA_TYPES	**Value:** {'signed 16-bit integer': <type 'numpy.int16'>, 'signed 3...
MINIMUM_KEYS	**Value:** ['X-Binary-Size-Fastest-Dimension', 'ByteOrder', 'Data ty...
STARTER	**Value:** '\x0c\x1a\x04\xd5'
PADDING	**Value:** 512
__package__	**Value:** 'fabio'

12.2 Class cbfimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.cbfimage.cbfimage**

Read the Cif Binary File data format

12.2.1 Methods

__init__(*self*, *data*=`None`, *header*=`None`, *fname*=`None`)

Constructor of the class CIF Binary File reader.

Parameters

`_strFilename`: the name of the file to open

(type=string)

Overrides: object.__init__

checkData(*data*=`None`)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

write the file in CBF format

Parameters

`fname`: name of the file

Overrides: fabio.fabioimage.fabioimage.write

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

12.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

12.3 Class CIF

This is the CIF class, it represents the CIF dictionary; and as a a python dictionary thus inherits from the dict built in class.

12.3.1 Methods

__init__(*self*, *_strFilename*=`None`)

Constructor of the class.

Parameters

`_strFilename`: the name of the file to open

(type=filename (str) or file object)

Return Value

`new empty dictionary`

Overrides: object.__init__

__setitem__(*self*, *key*, *value*)

x[i]=y

Overrides: dict.__setitem__ extit(inherited documentation)

pop(*self*, *key*)

If key is not found, d is returned if given, otherwise KeyError is raised

Return Value

v, remove specified key and return the corresponding value

Overrides: dict.pop extit(inherited documentation)

popitem(*self*, *key*)

2-tuple; but raise KeyError if D is empty.

Return Value

(k, v), remove and return some (key, value) pair as a

Overrides: dict.popitem extit(inherited documentation)

loadCIF(*self*, *_strFilename*, *_bKeepComment*=`False`)

Load the CIF file and populates the CIF dictionary into the object

Parameters

`_strFilename`: the name of the file to open

(type=string)

`_strFilename`: the name of the file to open

(type=string)

Return Value

None

readCIF(*self*, *_strFilename*, *_bKeepComment*=`False`)

Load the CIF file and populates the CIF dictionary into the object

Parameters

`_strFilename`: the name of the file to open

(type=string)

`_strFilename`: the name of the file to open

(type=string)

Return Value

None

isAscii(*_strIn*)

Check if all characters in a string are ascii,

Parameters

`_strIn`: input string

(type=python string)

Return Value

boolean

(type=boolean)

saveCIF(*self*, *_strFilename*=`'test.cif'`, *linesep*=`'\n'`, *binary*=`False`)

Transforms the CIF object in string then write it into the given file

Parameters

`_strFilename`: the of the file to be written

`linesep`: line separation used (to force compatibility with windows/unix)

`binary`: Shall we write the data as binary (True only for imageCIF/CBF)

`param`: *(type=string)*

tostring(*self*, *_strFilename*=`None`, *linesep*=`'\n'`)

Converts a cif dictionnary to a string according to the CIF syntax

Parameters

`_strFilename`: the name of the filename to be appended in the header of the CIF file

(type=string)

Return Value

a sting that corresponds to the content of the CIF - file.

exists(*self*, *sKey*)

Check if the key exists in the CIF and is non empty.

Parameters

`sKey`: CIF key

(type=string)

`cif`: CIF dictionary

Return Value

True if the key exists in the CIF dictionary and is non empty

(type=boolean)

existsInLoop(*self*, *sKey*)

Check if the key exists in the CIF dictionary.

Parameters

`sKey`: CIF key

(type=string)

`cif`: CIF dictionary

Return Value

True if the key exists in the CIF dictionary and is non empty

(type=boolean)

loadCHIPLOT(*self*, *_strFilename*)

Load the powder diffraction CHIPLOT file and returns the pd_CIF dictionary in the object

Parameters

`_strFilename`: the name of the file to open

(type=string)

Return Value

the CIF object corresponding to the powder diffraction

(type=dictionary)

LoopHasKey(*loop*, *key*)

Returns True if the key (string) exist in the array called loop

Inherited from dict

__cmp__(), __contains__(), __delitem__(), __eq__(), __ge__(), __getattribute__(), __getitem__(), __gt__(), __iter__(), __le__(), __len__(), __lt__(), __ne__(), __new__(), __repr__(), __sizeof__(),

clear(), copy(), fromkeys(), get(), has_key(), items(), iteritems(), iterkeys(), itervalues(), keys(), setdefault(), update(), values()

Inherited from object

__delattr__(), __format__(), __reduce__(), __reduce_ex__(), __setattr__(), __str__(), __subclasshook__()

12.3.2 Properties

Name	Description
Inherited from object	
__class__	

12.3.3 Class Variables

Name	Description
EOL	**Value:** `['\r', '\n', '\r\n', '\n\r']`
BLANK	**Value:** `[' ', '\t', '\r', '\n', '\r\n', '\n\r']`
START_COMMENT	**Value:** `['"', '\'']`
BINARY_MARKER	**Value:** `'--CIF-BINARY-FORMAT-SECTION--'`
Inherited from dict	
__hash__	

13 Module fabio.cf_io

13.1 Functions

read(...)

call the c-columnfile reading interface. The mode keyword argument is either: "a" for ascii (the default) "b" for binary

13.2 Variables

Name	Description
__package__	**Value:** `None`

14 Module fabio.compression

```
Authors: Jérôme Kieffer, ESRF
         email:jerome.kieffer@esrf.fr
```

```
FabIO library containing compression and decompression algorithm for various
```

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: jerome.kieffer@esrf.eu

Copyright: European Synchrotron Radiation Facility, Grenoble, France

License: GPLv3+

14.1 Functions

md5sum(*blob*)

returns the md5sum of an object...

endianness()

Return the native endianness of the system

decGzip(*stream*)

Decompress a chunk of data using the gzip algorithm from Python or alternatives if possible

decBzip2(*stream*)

Decompress a chunk of data using the bzip2 algorithm from Python

decZlib(*stream*)

Decompress a chunk of data using the zlib algorithm from Python

decByteOffet_python(*stream*, *size*)

Analyze a stream of char with any length of exception (2,4, or 8 bytes integers)

Parameters

`stream`: string representing the compressed data

`size`: the size of the output array (of longInts)

Return Value

1D-ndarray

decByteOffet_weave(*stream*, *size*)

Analyze a stream of char with any length of exception (2,4, or 8 bytes integers)

Parameters

`stream`: string representing the compressed data

`size`: the size of the output array (of longInts)

Return Value

1D-ndarray

decByteOffet_numpy(*stream*, *size*=`None`)

```
Analyze a stream of char with any length of exception:
            2, 4, or 8 bytes integers

@param stream: string representing the compressed data
@param size: the size of the output array (of longInts)
@return: 1D-ndarray
```

decByteOffet_cython(*stream*, *size*=`None`)

```
Analyze a stream of char with any length of exception:
            2, 4, or 8 bytes integers

@param stream: string representing the compressed data
@param size: the size of the output array (of longInts)
@return: 1D-ndarray
```

compByteOffet_numpy(*data*)

Compress a dataset into a string using the byte_offet algorithm

Parameters

`data`: ndarray

Return Value

string/bytes with compressed data

test =
numpy.array([0,1,2,127,0,1,2,128,0,1,2,32767,0,1,2,32768,0,1,2,2147483647,0,1,2,2147483648,0,1,2

decTY1(*raw_8*, *raw_16*=`None`, *raw_32*=`None`)

Modified byte offset decompressor used in Oxford Diffraction images

Parameters

`raw_8`: strings containing raw data with integer 8 bits

`raw_16`: strings containing raw data with integer 16 bits

`raw_32`: strings containing raw data with integer 32 bits

Return Value

numpy.ndarray

decKM4CCD(*raw_8*, *raw_16*=`None`, *raw_32*=`None`)

Modified byte offset decompressor used in Oxford Diffraction images

Parameters

`raw_8`: strings containing raw data with integer 8 bits

`raw_16`: strings containing raw data with integer 16 bits

`raw_32`: strings containing raw data with integer 32 bits

Return Value

numpy.ndarray

compTY1(*data*)

Modified byte offset compressor used in Oxford Diffraction images

Parameters

`data`: numpy.ndarray with the input data (integers!)

Return Value

3-tuple of strings: raw_8,raw_16,raw_32 containing raw data with integer of the given size

decPCK(*stream*, *dim1*=`None`, *dim2*=`None`, *overflowPix*=`None`)

Modified CCP4 pck decompressor used in MAR345 images

Parameters

`stream`: string or file

Return Value

numpy.ndarray (square array)

compPCK(*data*)

Modified CCP4 pck compressor used in MAR345 images

Parameters

`data`: numpy.ndarray (square array)

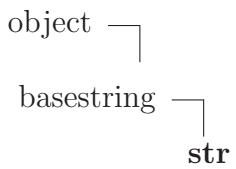
Return Value

compressed stream

14.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("compression")`
__package__	**Value:** `'fabio'`

14.3 Class str



str(object) -> string

Return a nice string representation of the object. If the argument is a string, the return value is the same object.

14.3.1 Methods

__add__(*x*, *y*)

x+y

__**contains**__(*x*, *y*)

y in x

__**eq**__(*x*, *y*)

x==y

__**format**__(*S*, *format_spec*)

default object formatter

Return Value

string

Overrides: object.__format__

__**ge**__(*x*, *y*)

x>=y

__**getattribute**__(*...*)

x.__getattribute__('name') <==> x.name

Overrides: object.__getattribute__

__**getitem**__(*x*, *y*)

x[y]

__**getnewargs**__(*...*)

__**getslice**__(*x*, *i*, *j*)

x[i:j]

Use of negative indices is not supported.

__**gt**__(*x*, *y*)

x>y

__**hash**__(*x*)

hash(x)

Overrides: object.__hash__

__**le**__(*x*, *y*)

x<=y

__**len**__(*x*)

len(x)

__**lt**__(*x*, *y*)

x<y

__**mod**__(*x*, *y*)

x%y

__**mul**__(*x*, *n*)

x*n

__**ne**__(*x*, *y*)

x!=y

__**new**__(*T*, *S*, *...*)

Return Value

a new object with type S, a subtype of T

Overrides: object.__new__

__**repr**__(*x*)

repr(x)

Overrides: object.__repr__

__**rmod**__(*x*, *y*)

y%x

__**rmul**__(*x*, *n*)

n*x

__sizeof__(*S*)

Return Value
size of S in memory, in bytes

Overrides: object.__sizeof__

__str__(*x*)

str(x)

Overrides: object.__str__

capitalize(*S*)

Return a copy of the string S with only its first character capitalized.

Return Value
string

center(*S*, *width*, *fillchar*=...)

Return S centered in a string of length width. Padding is done using the specified fill character (default is a space)

Return Value
string

count(*S*, *sub*, *start*=..., *end*=...)

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return Value
int

decode(*S*, *encoding*=..., *errors*=...)

Decodes S using the codec registered for encoding. encoding defaults to the default encoding. errors may be given to set a different error handling scheme. Default is 'strict' meaning that encoding errors raise a UnicodeDecodeError. Other possible values are 'ignore' and 'replace' as well as any other name registered with codecs.register_error that is able to handle UnicodeDecodeErrors.

Return Value
object

encode(*S*, *encoding*=..., *errors*=...)

Encodes S using the codec registered for encoding. encoding defaults to the default encoding. errors may be given to set a different error handling scheme. Default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that is able to handle UnicodeEncodeErrors.

Return Value
object

endswith(*S*, *suffix*, *start*=..., *end*=...)

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

Return Value
bool

expandtabs(*S*, *tabsize*=...)

Return a copy of S where all tab characters are expanded using spaces. If tabsize is not given, a tab size of 8 characters is assumed.

Return Value
string

find(*S*, *sub*, *start*=... , *end*=...)

Return the lowest index in S where substring sub is found, such that sub is contained within s[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

Return Value
int

format(*S*, **args*, ***kwargs*)

Return Value
string

index(*S*, *sub*, *start*=. . . , *end*=. . .)

Like S.find() but raise ValueError when the substring is not found.

Return Value
int

isalnum(*S*)

Return True if all characters in S are alphanumeric and there is at least one character in S, False otherwise.

Return Value
bool

isalpha(*S*)

Return True if all characters in S are alphabetic and there is at least one character in S, False otherwise.

Return Value
bool

isdigit(*S*)

Return True if all characters in S are digits and there is at least one character in S, False otherwise.

Return Value
bool

islower(*S*)

Return True if all cased characters in S are lowercase and there is at least one cased character in S, False otherwise.

Return Value
bool

isspace(*S*)

Return True if all characters in S are whitespace and there is at least one character in S, False otherwise.

Return Value
bool

istitle(*S*)

Return True if S is a titlecased string and there is at least one character in S, i.e. uppercase characters may only follow uncased characters and lowercase characters only cased ones. Return False otherwise.

Return Value
bool

isupper(*S*)

Return True if all cased characters in S are uppercase and there is at least one cased character in S, False otherwise.

Return Value
bool

join(*S*, *iterable*)

Return a string which is the concatenation of the strings in the iterable. The separator between elements is S.

Return Value
string

ljust(*S*, *width*, *fillchar*=...)

Return S left-justified in a string of length width. Padding is done using the specified fill character (default is a space).

Return Value
string

lower(*S*)

Return a copy of the string S converted to lowercase.

Return Value
string

lstrip(*S*, *chars*=...)

Return a copy of the string S with leading whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is unicode, S will be converted to unicode before stripping

Return Value
string or unicode

partition(*S*, *sep*)

Search for the separator sep in S, and return the part before it, the separator itself, and the part after it. If the separator is not found, return S and two empty strings.

Return Value
(head, sep, tail)

replace(*S*, *old*, *new*, *count*=...)

Return a copy of string S with all occurrences of substring old replaced by new. If the optional argument count is given, only the first count occurrences are replaced.

Return Value
string

rfind(*S*, *sub*, *start*=... , *end*=...)

Return the highest index in S where substring sub is found, such that sub is contained within s[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

Return Value
int

rindex(*S*, *sub*, *start*=... , *end*=...)

Like S.rfind() but raise ValueError when the substring is not found.

Return Value
int

rjust(*S*, *width*, *fillchar*=...)

Return S right-justified in a string of length width. Padding is done using the specified fill character (default is a space)

Return Value
string

rpartition(*S*, *sep*)

Search for the separator sep in S, starting at the end of S, and return the part before it, the separator itself, and the part after it. If the separator is not found, return two empty strings and S.

Return Value
(head, sep, tail)

rsplit(*S*, *sep*=. . . , *maxsplit*=. . .)

Return a list of the words in the string S, using sep as the delimiter string, starting at the end of the string and working to the front. If maxsplit is given, at most maxsplit splits are done. If sep is not specified or is None, any whitespace string is a separator.

Return Value
list of strings

rstrip(*S*, *chars*=. . .)

Return a copy of the string S with trailing whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is unicode, S will be converted to unicode before stripping

Return Value
string or unicode

split(*S*, *sep*=. . . , *maxsplit*=. . .)

Return a list of the words in the string S, using sep as the delimiter string. If maxsplit is given, at most maxsplit splits are done. If sep is not specified or is None, any whitespace string is a separator and empty strings are removed from the result.

Return Value
list of strings

splitlines(*S*, *keepends*=. . .)

Return a list of the lines in S, breaking at line boundaries. Line breaks are not included in the resulting list unless keepends is given and true.

Return Value
list of strings

startswith(*S*, *prefix*, *start*=..., *end*=...)

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

Return Value
bool

strip(*S*, *chars*=...)

Return a copy of the string S with leading and trailing whitespace removed. If chars is given and not None, remove characters in chars instead. If chars is unicode, S will be converted to unicode before stripping

Return Value
string or unicode

swapcase(*S*)

Return a copy of the string S with uppercase characters converted to lowercase and vice versa.

Return Value
string

title(*S*)

Return a titlecased version of S, i.e. words start with uppercase characters, all remaining cased characters have lowercase.

Return Value
string

translate(*S*, *table*, *deletechars*=...)

Return a copy of the string S, where all characters occurring in the optional argument deletechars are removed, and the remaining characters have been mapped through the given translation table, which must be a string of length 256.

Return Value
string

upper(*S*)

Return a copy of the string S converted to uppercase.

Return Value
string

zfill(*S*, *width*)

Pad a numeric string S with zeros on the left, to fill a field of the specified width. The string S is never truncated.

Return Value
string

Inherited from object

__delattr__(), __init__(), __reduce__(), __reduce_ex__(), __setattr__(), __subclasshook__()

14.3.2 Properties

Name	Description
Inherited from object	
__class__	

15 Module fabio.converters

Converter module. This is for the moment empty (populated only with almost pass through anonymous functions) but aims to be populated with more sofisticated translators ...

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: jerome.kieffer@esrf.eu

License: GPLv3+

15.1 Functions

convert_data_integer(*data*)

convert data to integer

convert_data(*inp*, *outp*, *data*)

Return data converted to the output format ... over-simplistic implementation for the moment ...

Parameters

`inp, outp:` input/output format like "cbfimage"

`data(ndarray):` the actual dataset to be transformed

convert_header(*inp*, *outp*, *header*)

return header converted to the output format

Parameters

`inp, outp:` input/output format like "cbfimage"

`header(dict):` the actual set of headers to be transformed

15.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("converter")`
CONVERSION_HEADER	**Value:** {('edfimage', 'edfimage'): <function <lambda> at 0x1ec1500>}
CONVERSION_DATA	**Value:** {('edfimage', 'OXDimage'): <function convert_data_integer...
__package__	**Value:** 'fabio'

16 Module fabio.datIO

```
Authors: Henning O. Sorensen & Erik Knudsen
         Center for Fundamental Research: Metal Structures in Four Dimensions
         Risoe National Laboratory
         Frederiksborgvej 399
         DK-4000 Roskilde
         email:erik.knudsen@risoe.dk

         and Jon Wright, ESRF
```

16.1 Variables

Name	Description
__package__	**Value:** `None`

16.2 Class fabiodata

object
└
fabio.datIO.fabiodata

Known Subclasses: fabio.datIO.columnfile

A common class for dataIO in fable Contains a 2d numpy array for keeping data, and two lists (clabels and rlabels) containing labels for columns and rows respectively

16.2.1 Methods

__init__(*self*, *data*=`None`, *clabels*=`None`, *rlabels*=`None`, *fname*=`None`)

set up initial values

Overrides: object.__init__

read(*self*, *fname*=`None`, *frame*=`None`)

To be overridden by format specific subclasses

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(),

__repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

16.2.2 Properties

Name	Description
Inherited from object	
__class__	

16.3 Class columnfile

object

fabio.datIO.fabiodata

fabio.datIO.columnfile

Concrete fabiodata class

16.3.1 Methods

read(*self*, *fname*, *frame*=`None`)

To be overridden by format specific subclasses

Overrides: fabio.datIO.fabiodata.read extit(inherited documentation)

Inherited from fabio.datIO.fabiodata(Section 16.2)

__init__()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

16.3.2 Properties

Name	Description
Inherited from object	
__class__	

17 Module fabio.dm3image

```
Authors: Henning O. Sorensen & Erik Knudsen
         Center for Fundamental Research: Metal Structures in Four Dimensions
         Risoe National Laboratory
         Frederiksborgvej 399
         DK-4000 Roskilde
         email:erik.knudsen@risoe.dk

        + Jon Wright, ESRF
```

17.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("dm3image")`
DATA_TYPES	**Value:** {2: <type 'numpy.int16'>, 3: <type 'numpy.int32'>, 4: <ty...
DATA_BYTES	**Value:** {2: 2, 3: 4, 4: 2, 5: 4, 6: 4, 7: 8, 8: 1, 9: None, 10: N...
__package__	**Value:** `'fabio'`

17.2 Class dm3image

object
fabio.fabioimage.fabioimage
fabio.dm3image.dm3image

Read and try to write the dm3 data format

17.2.1 Methods

__init__(*self*, **args*, ***kwargs*)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

To be overridden - fill in self.header and self.data

Overrides: fabio.fabioimage.fabioimage.read extit(inherited documentation)

readbytes(*self*, *bytes_to_read*, *format*, *swap*=`True`)

read_tag_group(*self*)

read_tag_entry(*self*)

read_tag_type(*self*)

read_data(*self*)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

17.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

18 Module fabio.edfimage

```
License: GPLv2+

Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jon Wright & Jérôme Kieffer:
  European Synchrotron Radiation Facility;
  Grenoble (France)
```

18.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("edfimage")`
BLOCKSIZE	**Value:** `512`
DATA_TYPES	**Value:** {'Double': <type 'numpy.float64'>, 'DoubleIEEE128': <type...
NUMPY_EDF_DTYPE	**Value:** {'float128': 'QuadrupleValue', 'float32': 'FloatValue', '...
MINIMUM_KEYS	**Value:** ['HEADERID', 'IMAGE', 'BYTEORDER', 'DATATYPE', 'DIM_1', '...
DEFAULT_VALUES	**Value:** {}
__package__	**Value:** `'fabio'`

18.2 Class Frame

object
└ **fabio.edfimage.Frame**

A class representing a single frame in an EDF file

18.2.1 Methods

__init__(*self*, *data*=`None`, *header*=`None`, *header_keys*=`None`, *number*=`None`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

parseheader(*self*, *block*)

Parse the header in some EDF format from an already open file

Parameters

`block:` string representing the header block

(type=string, should be full ascii)

Return Value

size of the binary blob

swap_needed(*self*)

Decide if we need to byteswap

getData(*self*)

Unpack a binary blob according to the specification given in the header

Return Value

dataset as numpy.ndarray

setData(*self*, *npa*=`None`)

Setter for data in edf frame

getByteCode(*self*)

setByteCode(*self*, *_iVal*)

getEdfBlock(*self*, *force_type*=`None`, *fit2dMode*=`False`)

Parameters

`force_type:` type of the dataset to be enforced like "float64" or "uint16"

(type=string or numpy.dtype)

`fit2dMode:` enforce compatibility with fit2d and starts countimg number of images at 1

(type=boolean)

Return Value

ascii header block

(type=python string with the concatenation of the ascii header and the binary data block)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

18.2.2 Properties

Name	Description
data	Unpack a binary blob according to the specification given in the header
bytecode	
Inherited from object __class__	

18.3 Class edfimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.edfimage.edfimage**

Read and try to write the ESRF edf data format

18.3.1 Methods

__init__(*self*, *data*=None, *header*=None, *header_keys*=None, *frames*=None)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

checkHeader(*header*=None)

Empty for fabioimage but may be populated by others classes

Overrides: fabio.fabioimage.fabioimage.checkHeader

read(*self*, *fname*, *frame*=None)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

swap_needed(*self*)

Decide if we need to byteswap

unpack(*self*)

Unpack a binary blob according to the specification given in the header and return the dataset

Return Value

dataset as numpy.ndarray

getframe(*self*, *num*)

returns the file numbered 'num' in the series as a fabioimage

Overrides: fabio.fabioimage.fabioimage.getframe

previous(*self*)

returns the previous file in the series as a fabioimage

Overrides: fabio.fabioimage.fabioimage.previous

next(*self*)

returns the next file in the series as a fabioimage

Overrides: fabio.fabioimage.fabioimage.next

write(*self*, *fname*, *force_type*=`None`, *fit2dMode*=`False`)

Try to write a file check we can write zipped also mimics that fabian was writing uint16 (we sometimes want floats)

Parameters

`force_type:` can be numpy.uint16 or simply "float"

Return Value

None

Overrides: fabio.fabioimage.fabioimage.write

appendFrame(*self*, *frame*=`None`, *data*=`None`, *header*=`None`)

Method used add a frame to an EDF file

Parameters

`frame:` frame to append to edf image

(type=instance of Frame)

Return Value

None

deleteFrame(*self*, *frameNb*=`None`)

Method used to remove a frame from an EDF image. by default the last one is removed.

Parameters

`frameNb:` frame number to remove, by default the last.

(type=integer)

Return Value

None

fastReadData(*self*, *filename*=`None`)

This is a special method that will read and return the data from another file ... The aim is performances, ... but only supports uncompressed files.

Return Value

data from another file using positions from current edfimage

fastReadROI(*self*, *filename*, *coords*=`None`)

Method reading Region of Interest of another file based on metadata available in current edfimage. The aim is performances, ... but only supports uncompressed files.

Return Value

ROI-data from another file using positions from current edfimage

(type=numpy 2darray)

getNbFrames(*self*)

Getter for number of frames

setNbFrames(*self*, *val*)

Setter for number of frames ... should do nothing. Here just to avoid bugs

getHeader(*self*)

Getter for the headers. used by the property header,

setHeader(*self*, *_dictHeader*)

Enforces the propagation of the header to the list of frames

delHeader(*self*)

Deleter for edf header

getHeaderKeys(*self*)

Getter for edf header_keys

setHeaderKeys(*self*, *_listtHeader*)

Enforces the propagation of the header_keys to the list of frames

Parameters

`_listtHeader:` list of the (ordered) keys in the header

(type=python list)

delHeaderKeys(*self*)

Deleter for edf header_keys

getData(*self*)

getter for edf Data

Return Value
data for current frame
(type=numpy.ndarray)

setData(*self*, *_data*)

Enforces the propagation of the data to the list of frames

Parameters
`_data`: numpy array representing data

delData(*self*)

deleter for edf Data

getCapsHeader(*self*)

getter for edf headers keys in upper case

Return Value
data for current frame
(type=dict)

setCapsHeader(*self*, *_data*)

Enforces the propagation of the header_keys to the list of frames

Parameters
`_data`: numpy array representing data

delCapsHeader(*self*)

deleter for edf capsHeader

getDim1(*self*)

setDim1(*self*, *_iVal*)

getDim2(*self*)

setDim2(*self*, *_iVal*)

getDims(*self*)

getByteCode(*self*)

setByteCode(*self*, *_iVal*)

getBpp(*self*)

setBpp(*self*, *_iVal*)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), convert(), getclassname(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

18.3.2 Properties

Name	Description
nframes	Getter for number of frames
header	property: header of EDF file
header_keys	property: header_keys of EDF file
data	property: data of EDF file
capsHeader	property: capsHeader of EDF file, i.e. the keys of the header in UPPER case.
dim1	
dim2	
dims	
bytecode	
bpp	
Inherited from fabio.fabioimage.fabioimage (Section 19.3) classname	
Inherited from object __class__	

19 Module fabio.fabioimage

Authors: Henning O. Sorensen & Erik Knudsen
Center for Fundamental Research: Metal Structures in Four Dimensions
Risoe National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde
email:erik.knudsen@risoe.dk

and Jon Wright, Jerome Kieffer: ESRF

19.1 Functions

test()

check some basic fabioimage functionality

19.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("fabioimage")`
__package__	**Value:** `'fabio'`

19.3 Class fabioimage

object
fabio.fabioimage.fabioimage

Known Subclasses: fabio.fit2dmaskimage.fit2dmaskimage, fabio.tifimage.tifimage, fabio.kcdimage.kcdim
fabio.brukerimage.brukerimage, fabio.cbfimage.cbfimage, fabio.edfimage.edfimage, fabio.GEimage.GEimag
fabio.xsdimage.xsdimage, fabio.binaryimage.binaryimage, fabio.OXDimage.OXDimage, fabio.mar345image
fabio.dm3image.dm3image, fabio.adscimage.adscimage, fabio.GEimage_old.GEimage, fabio.pnmimage.pnm
fabio.fit2dspreadsheetimage.fit2dspreadsheetimage, fabio.HiPiCimage.HiPiCimage

A common object for images in fable Contains a numpy array (.data) and dict of meta data (.header)

19.3.1 Methods

__init__(*self*, *data*=None, *header*=None)

Set up initial values

Overrides: object.__init__

checkHeader(*header*=None)

Empty for fabioimage but may be populated by others classes

checkData(*data*=None)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

getclassname(*self*)

Retrieves the name of the class

Return Value
the name of the class

getframe(*self*, *num*)

returns the file numbered 'num' in the series as a fabioimage

previous(*self*)

returns the previous file in the series as a fabioimage

next(*self*)

returns the next file in the series as a fabioimage

toPIL16(*self*, *filename*=None)

Convert to Python Imaging Library 16 bit greyscale image

FIXME - this should be handled by the libraries now

getheader(*self*)

returns self.header

getmax(*self*)

Find max value in self.data, caching for the future

getmin(*self*)

Find min value in self.data, caching for the future

make_slice(*self*, *coords*)

Convert a len(4) set of coords into a len(2) tuple (pair) of slice objects the latter are immutable, meaning the roi can be cached

integrate_area(*self*, *coords*)

Sums up a region of interest if len(coords) == 4 -> convert coords to slices if len(coords) == 2 -> use as slices floor -> ? removed as unused in the function.

getmean(*self*)

return the mean

getstddev(*self*)

return the standard deviation

add(*self*, *other*)

Add another Image - warning, does not clip to 16 bit images by default

resetvals(*self*)

Reset cache - call on changing data

rebin(*self*, *x_rebin_fact*, *y_rebin_fact*, *keep_I*=`True`)

Rebin the data and adjust dims

Parameters

`x_rebin_fact:` x binning factor
(type=int)

`y_rebin_fact:` y binning factor
(type=int)

`keep_I:` shall the signal increase ?
(type=boolean)

write(*self*, *fname*)

To be overwritten - write the file

save(*self*, *fname*)

wrapper for write

readheader(*self*, *filename*)

Call the _readheader function...

update_header(*self*, ***kwds*)

update the header entries by default pass in a dict of key, values.

read(*self*, *filename*, *frame*=`None`)

To be overridden - fill in self.header and self.data

load(*self*, **arg*, ***kwarg*)

Wrapper for read

readROI(*self*, *filename*, *frame*=`None`, *coords*=`None`)

Method reading Region of Interest. This implementation is the trivial one, just doing read and crop

convert(*self*, *dest*)

Convert a fabioimage object into another fabioimage object (with possible conversions)

Parameters

`dest:` destination type "EDF", "edfimage" or the class itself

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

19.3.2 Properties

Name	Description
classname	Retrieves the name of the class
Inherited from object	
__class__	

20 Module fabio.fabioutils

General purpose utilities functions for fabio

20.1 Functions

deprecated(*func*)

used to deprecate a function/method: prints a lot of warning messages to enforce the modifaction of the code

getnum(*name*)

try to figure out a file number # guess it starts at the back

numstem(*name*)

cant see how to do without reversing strings Match 1 or more digits going backwards from the end of the string

deconstruct_filename(**arg*, ***kw*)

decorator that deprecates the use of a function

construct_filename(*filename*, *frame*=`None`)

Try to construct the filename for a given frame

next_filename(*name*, *padding*=`True`)

increment number

previous_filename(*name*, *padding*=`True`)

decrement number

jump_filename(*name*, *num*, *padding*=`True`)

jump to number

extract_filenumber(*name*)

extract file number

isAscii(*name*, *listExcluded*=`None`)

Parameters

`name:` string to check

`listExcluded:` list of char or string excluded.

Return Value

True of False whether name is pure ascii or not

toAscii(*name*, *excluded*=`None`)

Parameters

`name:` string to check

`excluded:` tuple of char or string excluded (not list: they are mutable).

Return Value

the name with all non valid char removed

nice_int(*s*)

Workaround that int('1.0') raises an exception

Parameters

`s:` string to be converted to integer

20.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("fabioutils")`
FILETYPES	**Value:** {'cbf': ['cbf'], 'cbf.bz2': ['cbf'], 'cbf.gz': ['cbf'], '...
COMPRESSORS	**Value:** {'.bz2': 'bzip2 -dc ', '.gz': 'gzip -dc '}
dictAscii	**Value:** {None: [' ', '!', '"', '#', '$', '%', '&', '\'', '(', ')'...
lines	**Value:** 'bzip2, a block-sorting file compressor. Version 1.0.5, ...
__package__	**Value:** 'fabio'
i	**Value:** 126
key	**Value:** 'cbf'

20.3 Class FilenameObject

object
└ **fabio.fabioutils.FilenameObject**

The 'meaning' of a filename ...

20.3.1 Methods

__init__(*self*, *stem*=None, *num*=None, *directory*=None, *format*=None, *extension*=None, *postnum*=None, *digits*=4, *filename*=None)

This class can either be instanciated by a set of parameters like directory, prefix, num, extension, ...

Parameters

`stem:`	the stem is a kind of prefix (str)
`num:`	image number in the serie (int)
`directory:`	name of the directory (str)
`format:`	??
`extension:`	
`postnum:`	
`digits:`	Number of digits used to print num
	Alternative constructor:
`filename:`	fullpath of an image file to be deconstructed into directory, prefix, num, extension, ...

Overrides: object.__init__

str(*self*)

Return a string representation

__repr__(*self*)

Return a string representation

Overrides: object.__repr__

tostring(*self*)

convert yourself to a string

deconstruct_filename(*self*, *filename*)

Break up a filename to get image type and number

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

20.3.2 Properties

Name	Description
Inherited from object	
__class__	

20.4 Class StringIO

StringIO.StringIO
└
fabio.fabioutils.StringIO

just an interface providing the name and mode property to a StringIO

BugFix for MacOSX mainly

20.4.1 Methods

__init__(*self*, *data*, *fname*=`None`, *mode*=`'r'`)

Overrides: StringIO.StringIO.__init__

getSize(*self*)

setSize(*self*, *size*)

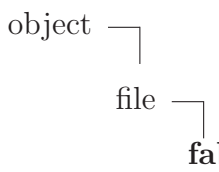
Inherited from StringIO.StringIO

__iter__(), close(), flush(), getvalue(), isatty(), next(), read(), readline(), readlines(), seek(), tell(), truncate(), write(), writelines()

20.4.2 Properties

Name	Description
size	

20.5 Class File



Known Subclasses: fabio.fabioutils.UnknownCompressedFile

wrapper for "file" with locking

20.5.1 Methods

__init__(*name*, *mode*=..., *buffering*=...)

```
        Open a file.  The mode can be 'r', 'w' or 'a' for reading (default),
        writing or appending.  The file will be created if it doesn't exist
        when opened for writing or appending; it will be truncated when
        opened for writing.  Add a 'b' to the mode for binary files.
        Add a '+' to the mode to allow simultaneous reading and writing.
        If the buffering argument is given, 0 means unbuffered, 1 means line
        buffered, and larger numbers specify the buffer size.  The preferred way
        to open a file is with the builtin open() function.
        Add a 'U' to mode to open the file for input with universal newline
        support.  Any line ending in the input file will be seen as a '
'
        in Python.  Also, a file so opened gains the attribute 'newlines';
        the value for this attribute is one of None (no newline read yet),
        '
', '
', '
' or a tuple containing all the newline types seen.

        'U' cannot be combined with 'w' or '+' mode.
```

Return Value
file object

Overrides: object.__init__

getSize(*self*)

setSize(*self*, *size*)

Inherited from file

__delattr__(), __enter__(), __exit__(), __getattribute__(), __iter__(), __new__(), __repr__(), __setattr__(), close(), fileno(), flush(), isatty(), next(), read(), readinto(), readline(), readlines(), seek(), tell(), truncate(), write(), writelines(), xreadlines()

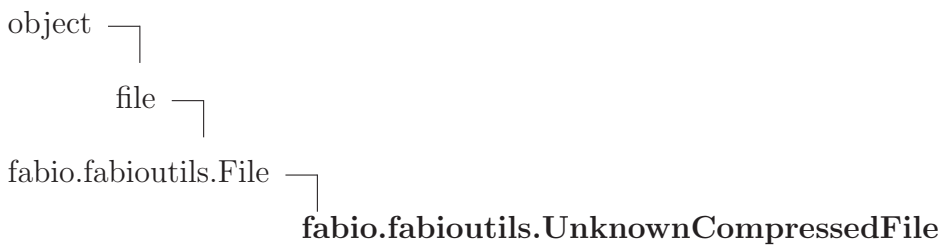
Inherited from object

__format__(), __hash__(), __reduce__(), __reduce_ex__(), __sizeof__(), __str__(), __subclasshook__()

20.5.2 Properties

Name	Description
size	
Inherited from file closed, encoding, errors, mode, name, newlines, softspace	
Inherited from object __class__	

20.6 Class UnknownCompressedFile



wrapper for "File" with locking

20.6.1 Methods

__init__(*self*, *name*, *mode*='rb', *buffering*=0)

```
        Open a file.  The mode can be 'r', 'w' or 'a' for reading (default),
        writing or appending.  The file will be created if it doesn't exist
        when opened for writing or appending; it will be truncated when
        opened for writing.  Add a 'b' to the mode for binary files.
        Add a '+' to the mode to allow simultaneous reading and writing.
        If the buffering argument is given, 0 means unbuffered, 1 means line
        buffered, and larger numbers specify the buffer size.  The preferred way
        to open a file is with the builtin open() function.
        Add a 'U' to mode to open the file for input with universal newline
        support.  Any line ending in the input file will be seen as a '
'
        in Python.  Also, a file so opened gains the attribute 'newlines';
        the value for this attribute is one of None (no newline read yet),
        '
', '
', '
' or a tuple containing all the newline types seen.

        'U' cannot be combined with 'w' or '+' mode.
```

Return Value

file object

Overrides: object.__init__ extit(inherited documentation)

Inherited from fabio.fabioutils.File(Section 20.5)

getSize(), setSize()

Inherited from file

__delattr__(), __enter__(), __exit__(), __getattribute__(), __iter__(), __new__(), __repr__(), __setattr__(), close(), fileno(), flush(), isatty(), next(), read(), readinto(), readline(), readlines(), seek(), tell(), truncate(), write(), writelines(), xreadlines()

Inherited from object

__format__(), __hash__(), __reduce__(), __reduce_ex__(), __sizeof__(), __str__(), __subclasshook__()

20.6.2 Properties

Name	Description
Inherited from fabio.fabioutils.File (Section 20.5) size	
Inherited from file closed, encoding, errors, mode, name, newlines, softspace	
Inherited from object __class__	

20.7 Class GzipFile

gzip.GzipFile
└
fabio.fabioutils.GzipFile

Just a wrapper forgzip.GzipFile providing the correct seek capabilities for python 2.5

20.7.1 Methods

__**init**__(*self*, *filename*=`None`, *mode*=`None`, *compresslevel*=`9`, *fileobj*=`None`)

Wrapper with locking for constructor for the GzipFile class.

At least one of fileobj and filename must be given a non-trivial value.

The new class instance is based on fileobj, which can be a regular file, a StringIO object, or any other object which simulates a file. It defaults to None, in which case filename is opened to provide a file object.

When fileobj is not None, the filename argument is only used to be included in the gzip file header, which may includes the original filename of the uncompressed file. It defaults to the filename of fileobj, if discernible; otherwise, it defaults to the empty string, and in this case the original filename is not included in the header.

The mode argument can be any of 'r', 'rb', 'a', 'ab', 'w', or 'wb', depending on whether the file will be read or written. The default is the mode of fileobj if discernible; otherwise, the default is 'rb'. Be aware that only the 'rb', 'ab', and 'wb' values should be used for cross-platform portability.

The compresslevel argument is an integer from 1 to 9 controlling the level of compression; 1 is fastest and produces the least compression, and 9 is slowest and produces the most compression. The default is 9.

Overrides: gzip.GzipFile.__init__

getSize(*self*)

setSize(*self*, *value*)

seek(*self*, *offset*, *whence*=0)

Move to new file position.

Argument offset is a byte count. Optional argument whence defaults to 0 (offset from start of file, offset should be >= 0); other values are 1 (move relative to current position, positive or negative), and 2 (move relative to end of file, usually negative, although many platforms allow seeking beyond the end of a file). If the file is opened in text mode, only offsets returned by tell() are legal. Use of other offsets causes undefined behavior.

This is a wrapper for seek to ensure compatibility with old python 2.5

Overrides: gzip.GzipFile.seek

Inherited from gzip.GzipFile

__del__(), __iter__(), __repr__(), close(), fileno(), flush(), isatty(), next(), read(), readline(), readlines(), rewind(), tell(), write(), writelines()

20.7.2 Properties

Name	Description
size	
closed	
Inherited from gzip.GzipFile filename	

20.7.3 Class Variables

Name	Description
Inherited from gzip.GzipFile max_read_chunk, myfileobj	

20.8 Class BZ2File

object
bz2.BZ2File
fabio.fabioutils.BZ2File

Wrapper with lock

20.8.1 Methods

__init__(*name*, *mode*='r', *buffering*=0, *compresslevel*=9)

```
            Open a bz2 file. The mode can be 'r' or 'w', for reading (default) or
            writing. When opened for writing, the file will be created if it doesn't
            exist, and truncated otherwise. If the buffering argument is given, 0 me
            unbuffered, and larger numbers specify the buffer size. If compressleve
            is given, must be a number between 1 and 9.

            Add a 'U' to mode to open the file for input with universal newline
            support. Any line ending in the input file will be seen as a '
' in
            Python. Also, a file so opened gains the attribute 'newlines'; the value
            for this attribute is one of None (no newline read yet), '
', '
',
            '
' or a tuple containing all the newline types seen. Universal
            newlines are available only when reading.
```

Return Value
file object

Overrides: object.__init__

getSize(*self*)

setSize(*self*, *value*)

Inherited from bz2.BZ2File

__delattr__(), __getattribute__(), __iter__(), __new__(), __setattr__(), close(), next(), read(), readline(), readlines(), seek(), tell(), write(), writelines(), xreadlines()

Inherited from object

__format__(), __hash__(), __reduce__(), __reduce_ex__(), __repr__(), __sizeof__(), __str__(), __subclasshook__()

20.8.2 Properties

Name	Description
size	
Inherited from bz2.BZ2File closed, mode, name, newlines, softspace	
Inherited from object __class__	

21 Module fabio.file_series

Authors:
.....

* Henning O. Sorensen & Erik Knudsen
  Center for Fundamental Research: Metal Structures in Four Dimensions
  Risoe National Laboratory
  Frederiksborgvej 399
  DK-4000 Roskilde
  email:erik.knudsen@risoe.dk
* Jon Wright, ESRF

21.1 Functions

new_file_series0(*first_object*, *first*=None, *last*=None, *step*=1)

Created from a fabio image first and last are file numbers

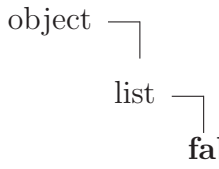
new_file_series(*first_object*, *nimages*=0, *step*=1, *traceback*=False)

```
A generator function that creates a file series starting from a a fabioimage.
Iterates through all images in a file (if more than 1), then proceeds to
the next file as determined by fabio.next_filename.

@param first_object: the starting fabioimage, which will be the first one yielded
    in the sequence
@param nimages:  the maximum number of images to consider
    step: step size, will yield the first and every step'th image until nimages
    is reached.  (e.g. nimages = 5, step = 2 will yield 3 images (0, 2, 4)
@param traceback: if True causes it to print a traceback in the event of an
    exception (missing image, etc.).  Otherwise the calling routine can handle
    the exception as it chooses
@param yields: the next fabioimage in the series.
    In the event there is an exception, it yields the sys.exec_info for the
    exception instead.  sys.exec_info is a tuple:
    ( exceptionType, exceptionValue, exceptionTraceback )
    from which all the exception information can be obtained.

Suggested usage:

::

    for obj in new_file_series( ... ):
      if not isinstance(obj, fabio.fabioimage.fabioimage ):
        # deal with errors like missing images, non readable files, etc
        # e.g.
        traceback.print_exception(obj[0], obj[1], obj[2])
```

21.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("fileseries")`
__package__	**Value:** `'fabio'`

21.3 Class file_series



Known Subclasses: fabio.file_series.numbered_file_series

```
Represents a series of files to iterate
has an idea of a current position to do next and prev

You also get from the list python superclass:
   append
   count
   extend
   insert
   pop
   remove
   reverse
   sort
```

21.3.1 Methods

__init__(*self*, *list_of_strings*)

Constructor:

Parameters

`list_of_strings`: arg should be a list of strings which are filenames

Return Value

new empty list

Overrides: object.__init__

first(*self*)

First image in series

last(*self*)

Last in series

previous(*self*)

Prev in a sequence

current(*self*)

Current position in a sequence

next(*self*)

Next in a sequence

jump(*self*, *num*)

Goto a position in sequence

len(*self*)

Number of files

first_image(*self*)

First image in a sequence

Return Value
fabioimage

last_image(*self*)

Last image in a sequence

Return Value
fabioimage

next_image(*self*)

Return the next image

Return Value
fabioimage

previous_image(*self*)

Return the previous image

Return Value
fabioimage

jump_image(*self*, *num*)

Jump to and read image

Return Value
fabioimage

current_image(*self*)

Current image in sequence

Return Value
fabioimage

first_object(*self*)

First image in a sequence

Return Value
file_object

last_object(*self*)

Last image in a sequence

Return Value
file_object

next_object(*self*)

Return the next image

Return Value
file_object

previous_object(*self*)

Return the previous image

Return Value
file_object

jump_object(*self*, *num*)

Jump to and read image

Return Value
file_object

current_object(*self*)

Current image in sequence

Return Value
file_object

Inherited from list

__add__(), __contains__(), __delitem__(), __delslice__(), __eq__(), __ge__(), __getattribute__(), __getitem__(), __getslice__(), __gt__(), __iadd__(), __imul__(), __iter__(), __le__(), __len__(), __lt__(), __mul__(), __ne__(), __new__(), __repr__(), __reversed__(), __rmul__(), __setitem__(), __setslice__(), __sizeof__(), append(), count(), extend(), index(), insert(), pop(), remove(), reverse(), sort()

Inherited from object

__delattr__(), __format__(), __reduce__(), __reduce_ex__(), __setattr__(), __str__(), __subclasshook__()

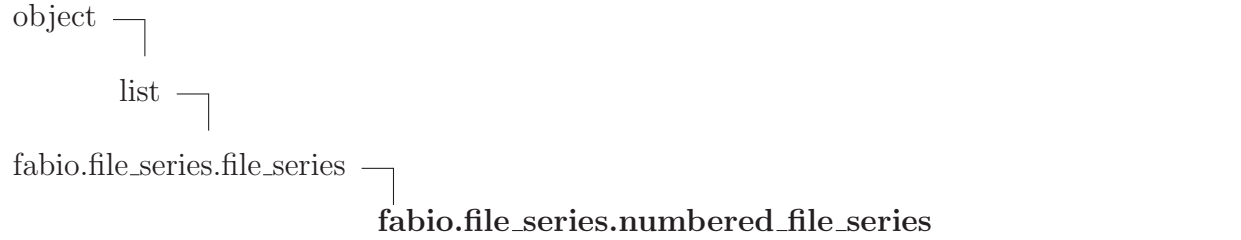
21.3.2 Properties

Name	Description
Inherited from object	
__class__	

21.3.3 Class Variables

Name	Description
Inherited from list	
__hash__	

21.4 Class numbered_file_series



mydata0001.edf = "mydata" + 0001 + ".edf" mydata0002.edf = "mydata" + 0002 + ".edf" mydata0003.edf = "mydata" + 0003 + ".edf"

21.4.1 Methods

__init__(*self*, *stem*, *first*, *last*, *extension*, *digits*=4, *padding*='Y', *step*=1)

Constructor

Parameters

`stem`: first part of the name

`step`: in case of every nth file

`padding`: possibility for specifying that numbers are not padded with zeroes up to digits

Return Value

new empty list

Overrides: object.__init__

Inherited from fabio.file_series.file_series(Section 21.3)

current(), current_image(), current_object(), first(), first_image(), first_object(), jump(), jump_image(), jump_object(), last(), last_image(), last_object(), len(), next(), next_image(), next_object(), previous(), previous_image(), previous_object()

Inherited from list

__add__(), __contains__(), __delitem__(), __delslice__(), __eq__(), __ge__(), __getattribute__(), __getitem__(), __getslice__(), __gt__(), __iadd__(), __imul__(), __iter__(), __le__(), __len__(), __lt__(), __mul__(), __ne__(), __new__(), __repr__(), __reversed__(), __rmul__(), __setitem__(), __setslice__(), __sizeof__(), append(), count(), extend(), index(), insert(), pop(), remove(), reverse(), sort()

Inherited from object

__delattr__(), __format__(), __reduce__(), __reduce_ex__(), __setattr__(), __str__(), __subclasshook__()

21.4.2 Properties

Name	Description
Inherited from object	
__class__	

21.4.3 Class Variables

Name	Description
Inherited from list	
__hash__	

21.5 Class filename_series

Much like the others, but created from a string filename

21.5.1 Methods

__init__(*self*, *filename*)

create from a filename (String)

next(*self*)

increment number

previous(*self*)

decrement number

current(*self*)

return current filename string

jump(*self*, *num*)

jump to a specific number

next_image(*self*)

returns the next image as a fabioimage

prev_image(*self*)

returns the previos image as a fabioimage

current_image(*self*)

returns the current image as a fabioimage

jump_image(*self*, *num*)

returns the image number as a fabioimage

next_object(*self*)

returns the next filename as a fabio.FilenameObject

previous_object(*self*)

returns the previous filename as a fabio.FilenameObject

current_object(*self*)

returns the current filename as a fabio.FilenameObject

jump_object(*self*, *num*)

returns the filename num as a fabio.FilenameObject

22 Module fabio.fit2dmaskimage

Author: Andy Hammersley, ESRF Translation into python/fabio: Jon Wright, ESRF

22.1 Variables

Name	Description
__package__	**Value:** `'fabio'`

22.2 Class fit2dmaskimage

object
└
fabio.fabioimage.fabioimage
└
fabio.fit2dmaskimage.fit2dmaskimage

Read and try to write Andy Hammersley's mask format

22.2.1 Methods

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

Try to write a file check we can write zipped also mimics that fabian was writing uint16 (we sometimes want floats)

Overrides: fabio.fabioimage.fabioimage.write

checkData(*data*=None)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

22.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3) classname	
Inherited from object __class__	

23 Module fabio.fit2dspreadsheetimage

```
Read the fit2d ascii image output
        + Jon Wright, ESRF
```

23.1 Variables

Name	Description
__package__	**Value:** `'fabio'`

23.2 Class fit2dspreadsheetimage

object
└
fabio.fabioimage.fabioimage
└
fabio.fit2dspreadsheetimage.fit2dspreadsheetimage

Read a fit2d ascii format

23.2.1 Methods

read(*self*, *fname*, *frame*=`None`)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3) classname	
Inherited from object __class__	

24 Module fabio.kcdimage

```
Authors: Jerome Kieffer, ESRF
         email:jerome.kieffer@esrf.fr

kcd images are 2D images written by the old KappaCCD diffractometer built by Nonius in t
Based on the edfimage.py parser.
```

24.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("kcdimage")`
DATA_TYPES	**Value:** {'u16': <type 'numpy.uint16'>}
MINIMUM_KEYS	**Value:** ['ByteOrder', 'Data type', 'X dimension', 'Y dimension', ...
DEFAULT_VALUES	**Value:** {'Data type': 'u16'}
__package__	**Value:** 'fabio'

24.2 Class kcdimage

object
└ fabio.fabioimage.fabioimage
  └ **fabio.kcdimage.kcdimage**

Read the Nonius kcd data format

24.2.1 Methods

read(*self*, *fname*, *frame*=None)

```
Read in header into self.header and
    the data   into self.data
```

Overrides: fabio.fabioimage.fabioimage.read

checkData(*data*=None)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

__init__(), add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

24.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3) classname	
Inherited from object __class__	

25 Module fabio.mar345_IO

```
New Cython version of mar345_io for preparing the migration to Python3

Compressor & decompressor for "pack" algorithm by JPA, binding to CCP4 libraries

Warning: decompressor is just a cython porting of mar345_io, but in cython so (soon) pyt

Future: make those algorithm actually generate strings not go via files;
        it will allow a broader use of the algorithm.
```

Authors: Jerome Kieffer, Gael Goret

Contact: jerome.kieffer@esrf.eu

License: LGPLv3+

25.1 Variables

Name	Description
__package__	**Value:** `'fabio'`
__test__	**Value:** {}

26 Module fabio.mar345image

```
Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jon Wright, Jérôme Kieffer & Gaël Goret:
  European Synchrotron Radiation Facility;
  Grenoble (France)
```

26.1 Variables

Name	Description
__doc__	**Value:** `...`
logger	**Value:** `logging.getLogger("mar345image")`
__package__	**Value:** `'fabio'`

26.2 Class mar345image

object

fabio.fabioimage.fabioimage

fabio.mar345image.mar345image

26.2.1 Methods

__init__(*self*, **args*, ***kwargs*)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

Read a mar345 image

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

Try to write mar345 file. This is still in beta version. It uses CCP4 (LGPL) PCK1 algo from JPA

Overrides: fabio.fabioimage.fabioimage.write

nb_overflow_pixels(*self*)

checkData(*data*=`None`)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

26.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

27 Module fabio.marccdimage

```
Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jon Wright:
  European Synchrotron Radiation Facility;
  Grenoble (France)

marccdimage can read MarCCD and MarMosaic images including header info.

JPW : Use a parser in case of typos (sorry?)
```

27.1 Functions

make_format(*c_def_string*)

Reads the header definition in c and makes the format string to pass to struct.unpack

interpret_header(*header*, *fmt*, *names*)

given a format and header interpret it

27.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("marccdimage")`
CDEFINITION	**Value:** `'\ntypedef struct frame_header_type {\n /* File/h...`
C_TO_STRUCT	**Value:** `{'INT32': 'i', 'UINT16': 'H', 'UINT32': 'I', 'char': 'c'}`
C_SIZES	**Value:** `{'INT32': 4, 'UINT16': 2, 'UINT32': 4, 'char': 1}`
MAXIMAGES	**Value:** `9`

continued on next page

Name	Description
HEADER_FORMAT	**Value:** `'IccccccccccccccccIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.`
HEADER_NAMES	**Value:** `['header_type', 'header_name[16]', 'header_name[16]', 'he...`
__package__	**Value:** `'fabio'`

27.3 Class marccdimage

object
fabio.fabioimage.fabioimage
fabio.tifimage.tifimage
fabio.marccdimage.marccdimage

Read in data in mar ccd format, also MarMosaic images, including header info

27.3.1 Methods

Inherited from fabio.tifimage.tifimage(Section 32.2)

__init__(), read(), write()

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

27.3.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	

continued on next page

Name	Description
Inherited from object	
__class__	

28 Module fabio.openimage

Authors: Henning O. Sorensen & Erik Knudsen
Center for Fundamental Research: Metal Structures in Four Dimensions
Risoe National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde
email:henning.sorensen@risoe.dk

mods for fabio by JPW

28.1 Functions

do_magic(*byts*)

Try to interpret the bytes starting the file as a magic number

openimage(*filename*, *frame*=None)

Try to open an image

openheader(*filename*)

return only the header

28.2 Variables

Name	Description
logger	**Value:** logging.getLogger("openimage")
MAGIC_NUMBERS	**Value:** [('FORMAT : 86', 'bruker'), ('MM\x00*', 'tif'), ('...
__package__	**Value:** 'fabio'

29 Module fabio.pilatusimage

```
Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jon Wright:
  European Synchrotron Radiation Facility;
  Grenoble (France)
```

29.1 Variables

Name	Description
__package__	**Value:** `'fabio'`

29.2 Class pilatusimage

object
fabio.fabioimage.fabioimage
fabio.tifimage.tifimage
fabio.pilatusimage.pilatusimage

Read in Pilatus format, also pilatus images, including header info

29.2.1 Methods

Inherited from fabio.tifimage.tifimage(Section 32.2)

__init__(), read(), write()

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

29.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

30 Module fabio.pnmimage

Authors: Henning O. Sorensen & Erik Knudsen
Center for Fundamental Research: Metal Structures in Four Dimensions
Risoe National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde
email:henning.sorensen@risoe.dk

30.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("pnmimage")`
SUBFORMATS	**Value:** `['P1', 'P2', 'P3', 'P4', 'P5', 'P6', 'P7']`
HEADERITEMS	**Value:** `['SUBFORMAT', 'DIMENSIONS', 'MAXVAL']`
P7HEADERITEMS	**Value:** `['WIDTH', 'HEIGHT', 'DEPTH', 'MAXVAL', 'TUPLTYPE', 'ENDHDR']`
__package__	**Value:** `'fabio'`

30.2 Class pnmimage

object
fabio.fabioimage.fabioimage
fabio.pnmimage.pnmimage

30.2.1 Methods

__init__(*self*, **arg*, ***kwargs*)

Set up initial values

Overrides: object.__init__ extit(inherited documentation)

read(*self*, *fname*, *frame*=`None`)

try to read PNM images

Parameters

`fname:` name of the file

`frame:` not relevant here! PNM is always single framed

Overrides: fabio.fabioimage.fabioimage.read

P1dec(*buf*, *bytecode*)

P4dec(*buf*, *bytecode*)

P2dec(*buf*, *bytecode*)

P5dec(*buf*, *bytecode*)

P3dec(*buf*, *bytecode*)

P6dec(*buf*, *bytecode*)

P7dec(*buf*, *bytecode*)

write(*self*, *filename*)

To be overwritten - write the file

Overrides: fabio.fabioimage.fabioimage.write extit(inherited documentation)

checkData(*data*=`None`)

Empty for fabioimage but may be populated by others classes, especially for format accepting only integers

Overrides: fabio.fabioimage.fabioimage.checkData extit(inherited documentation)

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

30.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

31 Module fabio.readbytestream

```
Reads a bytestream

Authors: Jon Wright     Henning O. Sorensen & Erik Knudsen
         ESRF           Risoe National Laboratory
```

31.1 Functions

readbytestream(*fil*, *offset*, *x*, *y*, *nbytespp*, *datatype*='int', *signed*='n', *swap*='n', *typeout*=<type 'numpy.uint16'>)

Reads in a bytestream from a file (which may be a string indicating a filename, or an already opened file (should be "rb")) offset is the position (in bytes) where the pixel data start nbytespp = number of bytes per pixel type can be int or float (4 bytes pp) or double (8 bytes pp) signed: normally signed data 'y', but 'n' to try to get back the right numbers when unsigned data are converted to signed (python once had no unsigned numeric types.) swap, normally do not bother, but 'y' to swap bytes typeout is the numpy type to output, normally uint16, but more if overflows occurred x and y are the pixel dimensions

TODO : Read in regions of interest

PLEASE LEAVE THE STRANGE INTERFACE ALONE - IT IS USEFUL FOR THE BRUKER FORMAT

31.2 Variables

Name	Description
logger	**Value:** `logging.getLogger("readbytestream")`
DATATYPES	**Value:** {('double', 'y', 4): <type 'numpy.float64'>, ('float', 'y...
__package__	**Value:** 'fabio'

32 Module fabio.tifimage

```
FabIO class for dealing with TIFF images.
In facts wraps TiffIO from V. Armando Solé (available in PyMca) or falls back to PIL

Authors:
........
* Henning O. Sorensen & Erik Knudsen:
  Center for Fundamental Research: Metal Structures in Four Dimensions;
  Risoe National Laboratory;
  Frederiksborgvej 399;
  DK-4000 Roskilde;
  email:erik.knudsen@risoe.dk
* Jérôme Kieffer:
  European Synchrotron Radiation Facility;
  Grenoble (France)

License: GPLv3+
```

Date: 11/07/2011

Authors: J\xc3\xa9r\xc3\xb4me Kieffer, Henning O. Sorensen, Erik Knudsen

Copyright: ESRF, Grenoble & Risoe National Laboratory

License: GPLv3+

32.1 Variables

Name	Description
__status__	**Value:** 'stable'
logger	**Value:** logging.getLogger("tifimage")
PIL_TO_NUMPY	**Value:** {'1': <type 'bool'>, 'F': <type 'numpy.float32'>, 'I': <t...
LITTLE_ENDIAN	**Value:** 1234
BIG_ENDIAN	**Value:** 3412
TYPES	**Value:** {0: 'invalid', 1: 'byte', 2: 'ascii', 3: 'short', 4: 'lon...
TYPESIZES	**Value:** {0: 0, 1: 1, 2: 1, 3: 2, 4: 4, 5: 8, 6: 1, 7: 1, 8: 2, 9:...
baseline_tiff_tags	**Value:** {256: 'ImageWidth', 257: 'ImageLength', 258: 'BitsPerSamp...
__package__	**Value:** 'fabio'

32.2 Class tifimage

object
fabio.fabioimage.fabioimage
fabio.tifimage.tifimage

Known Subclasses: fabio.marccdimage.marccdimage, fabio.pilatusimage.pilatusimage

Images in TIF format Wraps TiffIO

32.2.1 Methods

__init__(*self*, **args*, ***kwds*)

Tifimage constructor adds an nbits member attribute

Overrides: object.__init__

read(*self*, *fname*, *frame*=`None`)

Wrapper for TiffIO.

Overrides: fabio.fabioimage.fabioimage.read

write(*self*, *fname*)

Overrides the fabioimage.write method and provides a simple TIFF image writer.

Parameters

`fname`: name of the file to save the image to

Overrides: fabio.fabioimage.fabioimage.write

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

32.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

32.3 Class Tiff_header

object
└
fabio.tifimage.Tiff_header

32.3.1 Methods

__init__(*self*, *string*)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

32.3.2 Properties

Name	Description
Inherited from object	
__class__	

32.4 Class Image_File_Directory

object
└
fabio.tifimage.Image_File_Directory

32.4.1 Methods

__init__(*self*, *instring*=`None`, *offset*=`-1`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

unpack(*self*, *instring*, *offset*=`-1`)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

32.4.2 Properties

Name	Description
Inherited from object	
__class__	

32.5 Class Image_File_Directory_entry

object
└ **fabio.tifimage.Image_File_Directory_entry**

32.5.1 Methods

__init__(*self*, *tag*=`0`, *tag_type*=`0`, *count*=`0`, *offset*=`0`)

x.__init__(...) initializes x; see x.__class__.__doc__ for signature

Overrides: object.__init__ extit(inherited documentation)

unpack(*self*, *strInput*)

extract_data(*self*, *full_string*)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

32.5.2 Properties

Name	Description
Inherited from object __class__	

33 Module fabio.xsdimage

```
Authors: Jérôme Kieffer, ESRF
         email:jerome.kieffer@esrf.fr
```

```
XSDimge are XML files containing numpy arrays
```

Author: J\xc3\xa9r\xc3\xb4me Kieffer

Contact: jerome.kieffer@esrf.eu

Copyright: European Synchrotron Radiation Facility, Grenoble, France

License: GPLv3+

33.1 Variables

Name	Description
logger	**Value:** `logging.getLogger("xsdimage")`
__package__	**Value:** `'fabio'`

33.2 Class xsdimage

object

fabio.fabioimage.fabioimage

fabio.xsdimage.xsdimage

Read the XSDataImage XML File data format

33.2.1 Methods

__init__(*self*, *data*=`None`, *header*=`None`, *fname*=`None`)

Constructor of the class XSDataImage.

Parameters

`_strFilename`: the name of the file to open

(type=string)

Overrides: object.__init__

read(*self*, *fname*, *frame*=`None`)

To be overridden - fill in self.header and self.data

Overrides: fabio.fabioimage.fabioimage.read

Inherited from fabio.fabioimage.fabioimage(Section 19.3)

add(), checkData(), checkHeader(), convert(), getclassname(), getframe(), getheader(), getmax(), getmean(), getmin(), getstddev(), integrate_area(), load(), make_slice(), next(), previous(), readROI(), readheader(), rebin(), resetvals(), save(), toPIL16(), update_header(), write()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

33.2.2 Properties

Name	Description
Inherited from fabio.fabioimage.fabioimage (Section 19.3)	
classname	
Inherited from object	
__class__	

Index