jicbioimage.core Documentation

Release 0.15.0

Tjelvar Olsson and Matthew Hartley

October 31, 2016

Contents

jicimagelib documentation

CHAPTER 1

The `jicbioimage.core` Python package

The `jicbioimage.core` Python package provides the core functionality of the `jicbioimage` namespace package.

- Documentation: http://jicbioimagecore.readthedocs.io
- GitHub: https://github.com/JIC-CSB/jicbioimage.core
- PyPI: https://pypi.python.org/pypi/jicbioimage.core
- Free software: MIT License

1.1 Features

- Built in functionality for working with microscopy data
- Automatic generation of audit trails
- IPython integration
- Cross-platform: Linux, Mac and Windows are all supported
- Works with Python 2.7, 3.3 and 3.4

1.2 Related packages

1.2.1 `jicbioimage`

- Documentation: http://jicbioimage.readthedocs.io
- GitHub: https://github.com/JIC-CSB/jicbioimage

1.2.2 `jicbioimage.transform`

- Documentation: http://jicbioimagetransform.readthedocs.io
- GitHub: https://github.com/JIC-CSB/jicbioimage.transform

1.2.3 jicbioimage.segment

- Documentation: http://jicbioimagesegment.readthedocs.io
- GitHub: https://github.com/JIC-CSB/jicbioimage.segment

1.2.4 jicbioimage.illustrate

- Documentation: http://jicbioimageillustrate.readthedocs.io
- GitHub: https://github.com/JIC-CSB/jicbioimage.illustrate

CHAPTER 2

API documentation

2.1 jicbioimage.core.image

Module for managing and accessing images.

class jicbioimage.core.image.**History**(*creation=None*)
Class for storing the provenance of an image.

class Event(*function*, *args*, *kwargs*)
An event in the history of an image.

History.**add_event**(*function*, *args*, *kwargs*)
Return event added to the history.

class jicbioimage.core.image.**Image**(*shape*, *dtype=<type 'numpy.uint8'>*, *buffer=None*, *offset=0*, *strides=None*, *order=None*, *name=None*, *log_in_history=True*)
Image class.

classmethod from_file(*fpath*, *name=None*, *log_in_history=True*)
Return *jicbioimage.core.image.Image* instance from a file.

Parameters

- **fpath** – path to the image file
- **name** – name of the image
- **log_in_history** – whether or not to log the creation event in the image's history

Returns *jicbioimage.core.image.Image*

class jicbioimage.core.image.**Image3D**(*shape*, *dtype=<type 'numpy.uint8'>*, *buffer=None*, *offset=0*, *strides=None*, *order=None*, *name=None*, *log_in_history=True*)
Image3D class; in other words a 3D stack.

classmethod from_directory(*directory*)
Return *jicbioimage.core.image.Image3D* from directory.

Parameters **directory** – name of input directory

Returns *jicbioimage.core.image.Image3D*

to_directory(*directory*)
Write slices from 3D image to directory.

write(*name*)
Write slices from 3D image to disk.

Parameters **name** – name of output directory

class jicbioimage.core.image.**ImageCollection**(*fpath=None*)
Class for storing related images.

image(*index=0*)
Return image as a *jicbioimage.core.image.Image*.

Parameters **index** – list index

Returns *jicbioimage.core.image.Image*

parse_manifest(*fpath*)
Parse manifest file to build up the collection of images.

Parameters **fpath** – path to the manifest file

Raises RuntimeError

proxy_image(*index=0*)
Return a *jicbioimage.core.image.ProxyImage* instance.

Parameters **index** – list index

Returns *jicbioimage.core.image.ProxyImage*

class jicbioimage.core.image.**MicroscopyCollection**(*fpath=None*)
Collection of *jicbioimage.core.image.MicroscopyImage* instances.

channels(*s=0*)
Return list of channels in the collection.

Parameters **s** – series

Returns list of channel identifiers

image(*s=0*, *c=0*, *z=0*, *t=0*)
Return image as a *jicbioimage.core.image.Image*.

Parameters

- **s** – series
- **c** – channel
- **z** – zslice
- **t** – timepoint

Returns *jicbioimage.core.image.Image*

proxy_image(*s=0*, *c=0*, *z=0*, *t=0*)
Return a *jicbioimage.core.image.MicroscopyImage* instance.

Parameters

- **s** – series
- **c** – channel
- **z** – zslice
- **t** – timepoint

Returns *jicbioimage.core.image.MicroscopyImage*

series
Return list of series in the collection.

timepoints (*s=0*)
Return list of time points in the collection.

Parameters **s** – series

Returns list of time point identifiers

zslices (*s=0*)
Return list of z-slices in the collection.

Parameters **s** – series

Returns list of zslice identifiers

zstack (*s=0*, *c=0*, *t=0*)
Return zstack as a *jicbioimage.core.image.Image3D*.

Parameters

- **s** – series
- **c** – channel
- **t** – timepoint

Returns zstack as a *jicbioimage.core.image.Image3D*

zstack_array (*s=0*, *c=0*, *t=0*)
Return zstack as a `numpy.ndarray`.

Parameters

- **s** – series
- **c** – channel
- **t** – timepoint

Returns zstack as a `numpy.ndarray`

zstack_proxy_iterator (*s=0*, *c=0*, *t=0*)
Return zstack *jicbioimage.core.image.ProxyImage* iterator.

Parameters

- **s** – series
- **c** – channel
- **t** – timepoint

Returns zstack as a *jicbioimage.core.image.ProxyImage* iterator

class `jicbioimage.core.image.`**MicroscopyImage** (*fpath*, *metadata={}*)
Lightweight image class with microscopy meta data.

in_zstack (*s*, *c*, *t*)
Return True if I am in the zstack.

Parameters

- **s** – series
- **c** – channel
- **t** – timepoint

Returns bool

is_me(*s*, *c*, *z*, *t*)
Return True if arguments match my meta data.

Parameters

- **s** – series
- **c** – channel
- **z** – zslice
- **t** – timepoint

Returns bool

class jicbioimage.core.image.**ProxyImage**(*fpath*, *metadata={}*)
Lightweight image class.

image
Underlying *jicbioimage.core.image.Image* instance.

2.2 jicbioimage.core.io

Module for reading and writing images.

class jicbioimage.core.io.**AutoName**
Class for generating output file names automatically.

directory = None
Output directory to save images to.

classmethod name(*func*)
Return auto generated file name.

namespace = ‘’
Image file namespace.

classmethod prefix()
Return auto generated file prefix.

prefix_format = ‘{:d}_’
Image file prefix format.

class jicbioimage.core.io.**AutoWrite**
Class for writing images automatically.

on = True
Whether or not auto writing of images is enabled.

class jicbioimage.core.io.**BFConvertWrapper**(*backend*)
Class for unpacking microscopy files using bfconvert.

already_converted(*fpath*)
Return true if the file already has a manifest file in the backend.

Parameters **fpath** – potential path to the manifest file

Returns bool

manifest(*entry*)
Returns manifest as a list.

Parameters **entry** – *jicbioimage.core.io.FileBackend.Entry*

Returns *jicbioimage.core.io.Manifest*

metadata_from_fname(*fname*, *md5_hexdigest*)
Return meta data extracted from file name.

Parameters **fname** – metadata file name

Returns dictionary with meta data required by

run_command(*input_file*, *output_dir=None*)
Return the command for running bfconvert as a list.

Parameters

- **input_file** – path to microscopy image to be converted
- **ouput_dir** – directory to write output tiff files to

Returns list

split_pattern(*win32=False*)
Pattern used to split the input file.

class jicbioimage.core.io.**DataManager**(*backend=None*)
Manage *jicbioimage.core.image.ImageCollection* instances.

load(*fpath*)
Load a microscopy file.

Parameters **fpath** – path to microscopy file

class jicbioimage.core.io.**FileBackend**(*directory*)
Class for storing image files.

class Entry(*base_dir*, *fpath*)
Class representing a backend entry.

directory
Where the images are stored.

FileBackend.**directory**
Where the entries are stored.

FileBackend.**new_entry**(*fpath*)
Return a new entry; to be populated with images.

Parameters **fpath** – path to microscopy image

Returns jiciimagelib.image.FileBackend.Entry instance

class jicbioimage.core.io.**Manifest**
Class for generating backend entry manifest files.

add(*filename*, ***kwargs*)
Add an entry to the manifest.

Parameters

- **filename** – relative path to image
- **kwargs** – custom parameters, e.g. series, channel, zslice

Returns the added entry

json
Return json representation.

2.3 jicbioimage.core.transform

Module containing image transformation functions.

This module contains the function decorator *jicbioimage.core.transform.transformation()* that can be used to turn functions into image transformations.

Below is an example of how to create a transformation that inverts an image.

```
>>> import numpy as np
>>> @transformation
... def invert(image):
...     "Return the inverted image."
...     maximum = np.iinfo(image.dtype).max
...     maximum_array = np.ones(image.shape, dtype=image.dtype) * maximum
...     return maximum_array - image
...
```

jicbioimage.core.transform.**transformation**(*func*)
Function decorator to turn another function into a transformation.

2.4 jicbioimage.core.util.array

Module containing utility functions for manipulating numpy arrays.

jicbioimage.core.util.array.**check_dtype**(*array*, *allowed*)
Raises TypeError if the array is not of an allowed dtype.

Parameters

- **array** – array whose dtype is to be checked
- **allowed** – instance or list of allowed dtypes

Raises TypeError

jicbioimage.core.util.array.**color_array**(*array*, *color_dict*)
Return RGB color array.

Assigning a unique RGB color value to each unique element of the input array and return an array of shape (array.shape, 3).

Parameters

- **array** – input numpy.array
- **color_dict** – dictionary with keys/values corresponding to identifiers and RGB tuples respectively

jicbioimage.core.util.array.**dtype_contract**(*input_dtype=None*, *output_dtype=None*)
Function decorator for specifying input and/or output array dtypes.

Parameters

- **input_dtype** – dtype of input array
- **output_dtype** – dtype of output array

Returns function decorator

jicbioimage.core.util.array.**map_stack**(*array3D*, *z_function*)
Return 3D array where each z-slice has had the function applied to it.

Parameters

- **array3D** – 3D numpy.array
- **z_function** – function to be mapped to each z-slice

jicbioimage.core.util.array.**normalise**(*array*)
Return array normalised such that all values are between 0 and 1.

If all the values in the array are the same the function will return: - np.zeros(array.shape, dtype=np.float) if the value is 0 or less - np.ones(array.shape, dtype=np.float) if the value is greater than 0

Parameters **array** – numpy.array

Returns numpy.array.astype(numpy.float)

jicbioimage.core.util.array.**pretty_color_array**(*array*, *keep_zero_black=True*)
Return a RGB pretty color array.

Assigning a pretty RGB color value to each unique element of the input array and return an array of shape (array.shape, 3).

Parameters

- **array** – input numpy.array
- **keep_zero_black** – whether or not the background should be black

Returns numpy.array

jicbioimage.core.util.array.**reduce_stack**(*array3D*, *z_function*)
Return 2D array projection of the input 3D array.

The input function is applied to each line of an input x, y value.

Parameters

- **array3D** – 3D numpy.array
- **z_function** – function to use for the projection (e.g. `max()`)

jicbioimage.core.util.array.**unique_color_array**(*array*)
Return a RGB unique color array.

Assigning a unique RGB color value to each unique element of the input array and return an array of shape (array.shape, 3).

Parameters **array** – input numpy.array

Returns numpy.array

2.5 jicbioimage.core.util.color

Module for generating RGB tuples for use as colors in images.

jicbioimage.core.util.color.**identifier_from_unique_color**(*unique_color*)
Return identifier from unique RGB tuple.

Parameters **unique_color** – RGB tuple

Returns positive integer in range from 0 to 16777215 inclusive

jicbioimage.core.util.color.**pretty_color_from_identifier**(*identifier*)
Return deterministic aesthetically pleasing RGB tuple.

Returns RGB tuple

jicbioimage.core.util.color.**pretty_color_palette**(*identifiers*, *keep_zero_black=True*)
Return dictionary with pretty colors.

Parameters

- **identifiers** – set of unique identifiers
- **keep_zero_black** – whether or not the background should be black

Returns dictionary

jicbioimage.core.util.color.**random_pretty_color**()
Return random aesthetically pleasing RGB tuple.

Returns RGB tuple

jicbioimage.core.util.color.**unique_color_from_identifier**(*identifier*)
Return unique color as RGB tuple.

Useful for creating PNG images where each color is used as an identifier.

Raises TypeError if the identifier is not an integer.

Raises ValueError if the identifier is not in the range 0 to 16777215 inclusive.

Parameters **identifier** – positive integer in range from 0 to 16777215 inclusive

Raises TypeError, ValueError

Returns RGB tuple

jicbioimage.core.util.color.**unique_color_palette**(*identifiers*)
Return dictionary with unique colors.

Parameters **identifiers** – set of unique identifiers

Returns dictionary

j

A

B

C

D

F

H

I

J

L

M

N