Daffodil

Release 1.0

June 26, 2016

Contents

1 About **3**
1.1 Goals 3
1.2 License 3

2 API Reference **5**
2.1 daffodil 5

Daffodil is a D image processing Library, inspired by Pillow (The Python Imaging Library)

CHAPTER 1

About

1.1 Goals

- Simple, Extensible API
- Controllable internals with suitable defaults
- Wide format support with extensive testing
- High performance
- Support a variety of filters and transformations
- Thread Safety (pending)

1.2 License

Daffodil is licensed under the open source MIT license:

CHAPTER 2

API Reference

2.1 daffodil

The `daffodil` module provides the public interface for Daffodil.

2.1.1 color

module daffodil.color

2.1.2 image

The `daffodil.image` module exposes the `Image` class, which provides basic storage, access and conversion of images.

module daffodil.image

2.1.3 meta

The `daffodil.meta` module exposes the `MetaData` class.

module daffodil.meta

2.1.4 filter

The `daffodil.filter` module provides various filter functions that can be performed on images. filter functions differ from transformations (`daffodil.transform`) in that they cannot be performed in-place, ie. a copy of the image is required to perform the filter.

convolve

module daffodil.filter.convolve

gaussian

A gaussian filter (aka gaussian blur) is a convolution (`daffodil.filter.convolve`) using a matrix created from a gaussian distribution.

real gaussianDistribution(real x, real stDev = 1, real mean = 0)
Evaluate the gaussian/normal distribution for a given `x`, `stDev` and `mean`.

real[] gaussianMatrix(real stDev = 1, real maxDev = 3)
Create a 1D matrix of a discrete gaussian distribution with a given standard deviation and the number of standard deviations to stop generating at. The result is mirrored with guaranteed odd length.

The result can be used to convolve a image.

auto gaussianBlurred(string axis = "xy", size_t bpc)(const Image!bpc image, real stDev = 1,
Return a copy of `image` with a gaussian blur applied across axies `axis` with a given standard deviation and the number of standard deviations to stop at.

module daffodil.filter.gaussian

module daffodil.filter

2.1.5 transform

The `daffodil.transform` module provides various transformation functions that can be performed on images. Transform functions differ from filters (`daffodil.filter`) in that they can be performed in-place.

flip

void flip(string axis, size_t bpc)(Image!bpc image)
Flip `image` along `axis` in-place. `axis` may contain `x`, `y` or both.

Example:

```
auto image = load!8("daffodil.bmp");
image.flip!"x"(); // Flip the image horizontally
image.flip!"y"(); // Flip the image vertically
```

Image!bpc flipped(string axis, size_t bpc)(const Image!bpc image)
Same as *`flip`* but performs the operation on a copy of `image`. Allows for stringing operations together.

Example:

```
auto image = load!8("daffodil.bmp");

// Flip along each axis individually, making a copy each time.
auto flipped = image.flipped!"x".flipped!"y";
```

module daffodil.transform.flip

module daffodil.transform

module daffodil