

---

# Flask-IIIF Documentation

*Release 0.1.0*

**CERN**

April 28, 2015



<b>1</b>	<b>Contents</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Quickstart . . . . .	4
1.3	Configuration . . . . .	4
1.4	API . . . . .	5
1.5	Changes . . . . .	10
1.6	Contributing . . . . .	10
1.7	License . . . . .	11
1.8	Authors . . . . .	11
	<b>Python Module Index</b>	<b>13</b>



Flask-IIIF is a Flask extension permitting easy integration with the International Image Interoperability Framework (IIIF) API standards.



---

## Contents

---

- Installation
  - Requirements
- Quickstart
  - A Minimal Example
- Configuration
- API
  - Flask-IIIF
  - Cache
  - RESTful
- Changes
- Contributing
- License
- Authors
  - Contributors

## 1.1 Installation

Flask-IIIF is on PyPI so all you need is :

```
$ pip install flask-iiif
```

The development version can be downloaded from [its page at GitHub](#).

```
$ git clone https://github.com/inveniosoftware/flask-iiif.git
$ cd flask-iiif
$ python setup.py develop
$ ./run-tests.sh
```

### 1.1.1 Requirements

Flask-IIIF has the following dependencies:

- Flask
- blinker
- six

Flask-IIIF requires Python version 2.6, 2.7 or 3.3+

## 1.2 Quickstart

This part of the documentation will show you how to get started in using Flask-IIIF with Flask.

This guide assumes you have successfully installed Flask-IIIF and a working understanding of Flask. If not, follow the installation steps and read about Flask at <http://flask.pocoo.org/docs/>.

### 1.2.1 A Minimal Example

A minimal Flask-IIIF usage example looks like this.

First, let's create the application and initialise the extension:

```
from flask import Flask, session, redirect
from flask_iiif import IIIF
app = Flask("myapp")
ext = IIIF(app=app)
```

Second, let's create *Flask-RESTful* api instance and register image resource.

```
from flask_restful import Api
api = Api(app=app)
ext.init_restful(api)
```

## 1.3 Configuration

IIIF configuration.

`flask_iiif.config.IIIF_CACHE_HANDLER`

Add the preferred cache adaptor.

**See also:**

[ImageCache](#)

`flask_iiif.config.IIIF_CACHE_TIME`

How much time the image would be cached.

`flask_iiif.config.IIIF_QUALITIES`

The supported image qualities.

**See also:**

[IIIF Image API](#)

`flask_iiif.config.IIIF_CONVERTERS`

The supported image converters.

`flask_iiif.config.IIIF_VALIDATIONS`

The IIIF Image API validation.

**See also:**

[IIIF Image API v1](#) and [IIIF Image API v2](#)



## 1.4 API

This documentation section is automatically generated from Flask-IIIF's source code.

### 1.4.1 Flask-IIIF

Multimedia Image API.

**class** flask\_iiif.api.IIIFImageAPIWrapper(*image*)  
IIIF Image API Wrapper.

**apply\_api**(\*\*kwargs)  
Apply the IIIF API to the image.

Example to apply the IIIF API:

```
from flask_iiif.api import IIIFImageAPIWrapper

image = IIIFImageAPIWrapper.from_file(path)

image.apply_api(
    version=version,
    region=region,
    size=size,
    rotate=rotation,
    quality=quality
)
```

---

**Note:**

- If the version is not specified it will fallback to version 2.0.
  - Please note the `validate_api()` should be ran before `apply_api()`.
- 

**apply\_quality**(*value*)  
IIIF apply quality.

Apply `quality()`.

**apply\_region**(*value*)  
IIIF apply crop.

Apply `crop()`.

**apply\_rotate**(*value*)  
IIIF apply rotate.

Apply `rotate()`.

**apply\_size**(*value*)  
IIIF apply resize.

Apply `resize()`.

**static validate\_api**(\*\*kwargs)  
Validate IIIF Image API.

Example to validate the IIIF API:

```
from flask_iiif.api import IIIFImageAPIWrapper

IIIFImageAPIWrapper.validate_api(
    version=version,
    region=region,
    size=size,
    rotate=rotation,
    quality=quality,
    image_format=image_format
)
```

---

**Note:** If the version is not specified it will fallback to version 2.0.

---

**class** flask\_iiif.api.**MultimediaImage**(*image*)  
 Multimedia Image API.

Initializes an image api with IIIF standards. You can:

- Resize `resize()`.
- Crop `crop()`.
- Rotate `rotate()`.
- Change image quality `quality()`.

Example of editing and image and save it to disk:

```
from flask_iiif.api import MultimediaImage

image = IIIFImageAPIWrapper.from_file(path)
# Rotate the image
image.rotate(90)
# Resize the image
image.resize('300,200')
# Crop the image
image.crop('20,20,400,300')
# Make the image black and white
image.quality('grey')
# Finally save it to /tmp
image.save('/tmp')
```

Example of serving the modified image over http:

```
from flask import current_app, Blueprint
from flask_iiif.api import MultimediaImage

@blueprint.route('/serve/<string:uuid>/<string:size>')
def serve_thumbnail(uuid, size):
    \"\"\"Serve the image thumbnail.

    :param uuid: The document uuid.
    :param size: The desired image size.
    \"\"\"
    # Initialize the image with the uuid
    path = current_app.extensions['iiif'].uuid_to_path(uuid)
    image = IIIFImageAPIWrapper.from_file(path)
    # Resize it
    image.resize(size)
```

```
# Serve it
return send_file(image.serve(), mimetype='image/jpeg')
```

**crop** (*coordinates*)

Crop the image.

**Parameters** **coordinates** (*str*) – The coordinates to crop the image

---

**Note:**

- *coordinates* must have the following pattern:

- ‘x,y,w,h’: in pixels.
  - ‘pct:x,y,w,h’: percentance.
- 

**classmethod from\_file** (*path*)

Return the image object from the given path.

**Parameters** **path** (*str*) – The absolute path of the file

**Returns** a `MultimediaImage` instance

**classmethod from\_string** (*source*)

Create an `MultimediaImage` instance.

**Parameters** **source** (*BytesIO* object) – The image image string

**Returns** a `MultimediaImage` instance

**static percent\_to\_number** (*number*)

Calculate the percentance.

**quality** (*quality*)

Change the image format.

**Parameters** **quality** (*str*) – The image quality should be in (default, grey, bitonal, color)

---

**Note:** The library supports transformations between each supported mode and the “L” and “RGB” modes. To convert between other modes, you may have to use an intermediate image (typically an “RGB” image).

---

**static reduce\_by** (*nominally, dominator*)

Calculate the ratio.

**resize** (*dimensions, resample=0*)

Resize the image.

**Parameters**

- **dimensions** (*str*) – The dimensions to resize the image
  - **resample** (`PIL.Image` algorithm) – The algorithm to be used
- 

**Note:**

- *dimensions* must be one of the following:

- ‘w,’: The exact width, height will be calculated.
- ‘,h’: The exact height, width will be calculated.
- ‘pct:n’: Image percentance scale.
- ‘w,h’: The exact width and height.

–‘!w,h’: Best fit for the given width and height.

---

**rotate** (*degrees*, *mirror=False*)

Rotate the image by given degress.

**Parameters**

- **degress** (*float*) – The degrees, should be in range of [0, 360]
- **mirror** (*bool*) – Flip image from left to right

**static sanitize\_format\_name** (*value*)

Lowercase formats and make sure that jpg is written as jpeg.

**save** (*path*, *image\_format='jpeg'*, *quality=90*)

Store the image to the specific path.

**Parameters**

- **path** (*str*) – absolute path
- **image\_format** (*str*) – (gif, jpeg, pdf, png)
- **quality** (*int*) – The image quality; [1, 100]

---

**Note:** *image\_format* = jpg will not be recognized by `PIL.Image` and it will be changed to jpeg.

---

**serve** (*image\_format='png'*, *quality=90*)

Return a BytesIO object to easily serve it thought HTTP.

**Parameters**

- **image\_format** (*str*) – (gif, jpeg, pdf, png)
- **quality** (*int*) – The image quality; [1, 100]

---

**Note:** *image\_format* = jpg will not be recognized by `PIL.Image` and it will be changed to jpeg.

---

**size** ()

Return the current image size.

**Returns** the image size

**Return type** list

**class** flask\_iiif.api.**MultimediaObject**

The Multimedia Object.

## 1.4.2 Cache

Abstract simple cache definition.

All cache adaptors must at least implement `get()` and `set()` methods.

**class** flask\_iiif.cache.cache.**ImageCache**

Abstract cache layer.

**delete** (*key*)

Delete the specific key.

**flush** ()

Flush the cache.

**get** (*key*)

Return the key value.

**Parameters** **key** (*string*) – The object’s key

**set** (*key*, *value*, *timeout=172800*)

Cache the object.

**Parameters**

- **key** (*string*) – The object’s key
- **value** (*StringIO.StringIO* object) – the stored object
- **timeout** (*int*) – The cache timeout in seconds

Implement a simple cache.

**class** flask\_iiif.cache.simple.**ImageSimpleCache**

Simple image cache.

**delete** (*key*)

Delete the specific key.

**flush** ()

Flush the cache.

**get** (*key*)

Return the key value.

**Parameters** **key** (*string*) – The object’s key

**Returns** the stored object

**Return type** *BytesIO* object

**set** (*key*, *value*, *timeout=172800*)

Cache the object.

**Parameters**

- **key** (*string*) – The object’s key
- **value** (*BytesIO* object) – the stored object
- **timeout** (*int*) – The cache timeout in seconds

### 1.4.3 RESTful

Multimedia IIIF Image API.

**class** flask\_iiif.restful.**IIIFImageAPI**

IIIF API Implementation.

---

**Note:**

•**IIIF IMAGE API v1.0**

– For more infos please visit <<http://iiif.io/api/image/>>.

•**IIIF Image API v2.0**

– For more infos please visit <<http://iiif.io/api/image/2.0/>>.

•The API works only for GET requests

- The image process must follow strictly the following workflow:

- Region
- Size
- Rotation
- Quality
- Format

---

**delete** ()

delete.

**get** (*version, uuid, region, size, rotation, quality, image\_format*)

Run IIIF Image API workflow.

**head** ()

head.

**options** ()

options.

**post** ()

post.

**put** ()

put.

`flask_iiif.restful.error_handler(f)`

error handler.

## 1.5 Changes

Here you can see the full list of changes between each Flask-IIIF release.

Version 0.1.0 (released 2015-04-28)

- Initial public release.

## 1.6 Contributing

Bug reports, feature requests, and other contributions are welcome. If you find a demonstrable problem that is caused by the code of this library, please:

1. Search for [already reported problems](#).
2. Check if the issue has been fixed or is still reproducible on the latest *master* branch.
3. Create an issue with **a test case**.

If you create a feature branch, you can run the tests to ensure everything is operating correctly:

```
$ ./run-tests.sh
```

## 1.7 License

Flask-IIIF is free software; you can redistribute it and/or modify it under the terms of the Revised BSD License quoted below.

Copyright (C) 2013, 2014 CERN.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

In applying this license, CERN does not waive the privileges and immunities granted to it by virtue of its status as an Intergovernmental Organization or submit itself to any jurisdiction.

## 1.8 Authors

Flask-IIIF was originally developed for use in [Invenio](#) digital library software.

Contact us at [info@invenio-software.org](mailto:info@invenio-software.org)

### 1.8.1 Contributors

- Harris Tzovanakis <[drjova@cern.ch](mailto:drjova@cern.ch)>
- Jiri Kuncar <[jiri.kuncar@cern.ch](mailto:jiri.kuncar@cern.ch)>





## **f**

`flask_iiif.api`, 5  
`flask_iiif.cache.cache`, 8  
`flask_iiif.cache.simple`, 9  
`flask_iiif.config`, 4  
`flask_iiif.restful`, 9



## A

`apply_api()` (flask\_iiif.api.IIIFImageAPIWrapper method), 5  
`apply_quality()` (flask\_iiif.api.IIIFImageAPIWrapper method), 5  
`apply_region()` (flask\_iiif.api.IIIFImageAPIWrapper method), 5  
`apply_rotate()` (flask\_iiif.api.IIIFImageAPIWrapper method), 5  
`apply_size()` (flask\_iiif.api.IIIFImageAPIWrapper method), 5

## C

`crop()` (flask\_iiif.api.MultimediaImage method), 7

## D

`delete()` (flask\_iiif.cache.cache.ImageCache method), 8  
`delete()` (flask\_iiif.cache.simple.ImageSimpleCache method), 9  
`delete()` (flask\_iiif.restful.IIIFImageAPI method), 10

## E

`error_handler()` (in module flask\_iiif.restful), 10

## F

`flask_iiif.api` (module), 5  
`flask_iiif.cache.cache` (module), 8  
`flask_iiif.cache.simple` (module), 9  
`flask_iiif.config` (module), 4  
`flask_iiif.restful` (module), 9  
`flush()` (flask\_iiif.cache.cache.ImageCache method), 8  
`flush()` (flask\_iiif.cache.simple.ImageSimpleCache method), 9  
`from_file()` (flask\_iiif.api.MultimediaImage class method), 7  
`from_string()` (flask\_iiif.api.MultimediaImage class method), 7

## G

`get()` (flask\_iiif.cache.cache.ImageCache method), 8

`get()` (flask\_iiif.cache.simple.ImageSimpleCache method), 9  
`get()` (flask\_iiif.restful.IIIFImageAPI method), 10

## H

`head()` (flask\_iiif.restful.IIIFImageAPI method), 10

## I

`IIIF_CACHE_HANDLER` (in module flask\_iiif.config), 4  
`IIIF_CACHE_TIME` (in module flask\_iiif.config), 4  
`IIIF_CONVERTERS` (in module flask\_iiif.config), 4  
`IIIF_QUALITIES` (in module flask\_iiif.config), 4  
`IIIF_VALIDATIONS` (in module flask\_iiif.config), 4  
`IIIFImageAPI` (class in flask\_iiif.restful), 9  
`IIIFImageAPIWrapper` (class in flask\_iiif.api), 5  
`ImageCache` (class in flask\_iiif.cache.cache), 8  
`ImageSimpleCache` (class in flask\_iiif.cache.simple), 9

## M

`MultimediaImage` (class in flask\_iiif.api), 6  
`MultimediaObject` (class in flask\_iiif.api), 8

## O

`options()` (flask\_iiif.restful.IIIFImageAPI method), 10

## P

`percent_to_number()` (flask\_iiif.api.MultimediaImage static method), 7  
`post()` (flask\_iiif.restful.IIIFImageAPI method), 10  
`put()` (flask\_iiif.restful.IIIFImageAPI method), 10

## Q

`quality()` (flask\_iiif.api.MultimediaImage method), 7

## R

`reduce_by()` (flask\_iiif.api.MultimediaImage static method), 7  
`resize()` (flask\_iiif.api.MultimediaImage method), 7  
`rotate()` (flask\_iiif.api.MultimediaImage method), 8

## S

`sanitize_format_name()` (`flask_iiif.api.MultimediaImage` static method), [8](#)

`save()` (`flask_iiif.api.MultimediaImage` method), [8](#)

`serve()` (`flask_iiif.api.MultimediaImage` method), [8](#)

`set()` (`flask_iiif.cache.cache.ImageCache` method), [9](#)

`set()` (`flask_iiif.cache.simple.ImageSimpleCache` method), [9](#)

`size()` (`flask_iiif.api.MultimediaImage` method), [8](#)

## V

`validate_api()` (`flask_iiif.api.IIIFImageAPIWrapper` static method), [5](#)