# Contents

1  Configuration  
2  User’s Guide  
   2.1 Installation  
   2.2 Configuration  
   2.3 Usage  
   2.4 Customization  
   2.5 Example application  
3  API Reference  
   3.1 API Docs  
4  Additional Notes  
   4.1 Contributing  
   4.2 Changes  
   4.3 License  
   4.4 Authors  

Python Module Index
Invenio 3 module to connect Invenio to Archivematica

This is an experimental developer preview release.

Archivematica is an open-source software used to create digital archives. The goal of this module is to create a link between Invenio and Archivematica, so Invenio can send records to Archivematica to archive them.
CHAPTER 1

Configuration

• setup the Invenio URL in the automation-tools
• setup the Archivematica URL in the invenio-archivematica config
• setup the spaces and locations in Archivematica dashboards / storage

Further documentation is available on https://invenio-archivematica.readthedocs.io/
This part of the documentation will show you how to get started in using Invenio-Archivematica.

## 2.1 Installation

Invenio-Archivematica is on PyPI so all you need is:

```bash
$ pip install invenio-archivematica
```

## 2.2 Configuration

Invenio 3 module to connect Invenio to Archivematica.

```python
invenio_archivematica.config.ARCHIVEMATICA_BASE_TEMPLATE = 'invenio_archivematica/base.html'
    # Default base template for the demo page.

invenio_archivematica.config.ARCHIVEMATICA_DASHBOARD_API_KEY = 'change me'
    # The API key to use with the user above.

invenio_archivematica.config.ARCHIVEMATICA_DASHBOARD_URL = 'http://localhost:81'
    # The URL to Archivematica Dashboard.

invenio_archivematica.config.ARCHIVEMATICA_DASHBOARD_USER = 'invenio'
    # The user to connect to Archivematica Dashboard.

invenio_archivematica.config.ARCHIVEMATICA_ISARCHIVABLE_FACTORY = 'invenio_archivematica.factories.is_archivable_default'
    # The factory that is used to know if the sip should be archived or not.

See invenio_archivematica.factories.is_archivable_default().

invenio_archivematica.config.ARCHIVEMATICA_ORGANIZATION_NAME = 'CERN'
    # Organization name setup in Archivematica’s dashboard.
```
invenio-archivematica Documentation, Release 0.1.0.dev20170825

invenio_archivematica.config.ARCHIVEMATICA_STORAGE_API_KEY = 'change me'
    The API key to use with the user above.

invenio_archivematica.config.ARCHIVEMATICA_STORAGE_URL = 'http://localhost:8001'
    The URL to Archivematica Storage.

invenio_archivematica.config.ARCHIVEMATICA_STORAGE_USER = 'invenio'
    The user to connect to Archivematica Storage.

invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FACTORY = 'invenio_archivematica.factories.transfer_cp'
    The factory to do the transfers of files to the dashboard.
    See invenio_archivematica.factories.transfer_cp() and invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FOLDER for more information.

invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FOLDER = '.'
    The transfer folder setup in the dashboard.
    If you use a custom factory to do the transfer, you can put whatever you want here, it will be passed to your factory. See invenio_archivematica.factories.transfer_cp() and invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FACTORY.

2.3 Usage

Invenio 3 module to connect Invenio to Archivematica.

2.4 Customization

2.4.1 Factories

Factories used to customize the behavior of the module.

invenio_archivematica.factories.create_accession_id(ark)
    Create an accession ID to store the sip in Archivematica.
    Parameters ark (invenio_archivematica.models.Archive) – the archive
    Returns the created ID: SERVICE-SIP_UUID
    Return type str

invenio_archivematica.factories.is_archivable_default(sip)
    Tell if the given sip should be archived or not.
    If this function returns True, the sip will be archived later. Otherwise, the sip will never get archived.
    This function returns the archived flag on the SIP.

invenio_archivematica.factories.is_archivable_none(sip)
    Archive no sip.

invenio_archivematica.factories.transfer_cp(uuid, config)
    Transfer the files contained in the sip to a local destination.
    The transfer is done with a simple copy of files.
    This method is automatically called by the module to transfer the files. Depending on your installation, you may want to have a different behavior (copy among servers...). Then, you can create your own factory and link it into the config variable invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FACTORY.
Parameters

• **uuid** (*str*) – the id of the sip containing files to transfer

• **config** – can be empty. It will have the content of the variable `invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FOLDER`. However, it will use the export folder set in `invenio_sipstore.config.SIPSTORE_ARCHIVE_LOCATION_NAME`.

`invenio_archivematica.factories.transfer_demo(uuid, config)`

Transfer the files contained in the sip to the destination.

Very similar to the rsync transfer. However, because of time, I use the VERY UNSECURE sshpass package for rsync authentication. DO NOT USE IN PROD!!!

Parameters

• **uuid** (*str*) – the id of the sip containing files to transfer

• **config** (*dict*) – here config must be a dict with the following keys: - user - the SSH user
  - password_file - a path where the password is stored - remote - the URL or IP of the remote
  - remote_path - where to store files on the remote - args - the args for rsync

`invenio_archivematica.factories.transfer_rsync(uuid, config)`

Transfer the files contained in the sip to the destination.

The transfer is done with a rsync. If transfer to remote, you need a valid ssh setup.

This method is automatically called by the module to transfer the files. Depending on your installation, you may want to have a different behavior (copy among servers...). Then, you can create your own factory and link it into the config variable `invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FACTORY`.

The config needs to include at least the destination folder. If transfer to remote, it needs to include the user and the server. In either cases, you can include usual rsync parameters. See `invenio_archivematica.config.ARCHIVEMATICA_TRANSFER_FOLDER`:

```python
ARCHIVEMATICA_TRANSFER_FOLDER = {
    'server': 'localhost',
    'user': 'invenio',
    'destination': '/tmp',
    'args': '-az'
}
```

Parameters

• **uuid** (*str*) – the id of the sip containing files to transfer

• **config** – the config for rsync

### 2.4.2 Tasks

Tasks used by invenio-archivematica.

```sh
.... autotask:: invenio_archivematica.tasks.oais_start_transfer .... autotask:: invenio_archivematica.tasks.oais_process_transfer .... autotask:: invenio_archivematica.tasks.oais_finish_transfer .... autotask:: invenio_archivematica.tasks.oais_fail_transfer .... autotask:: invenio_archivematica.tasks.archive_new_sips
```

### 2.5 Example application
If you are looking for information on a specific function, class or method, this part of the documentation is for you.

### 3.1 API Docs

Invenio 3 module to connect Invenio to Archivematica.

```python
class invenio_archivematica.ext.InvenioArchivematica(app=None)
    Invenio-Archivematica extension.
    Extension initialization.
    init_app(app)
        Flask application initialization.
    init_config(app)
        Initialize configuration.
    init_listeners()
        Register the listener to invenio_sipstore’s signals.
```

### 3.1.1 API

API for Invenio 3 module to connect Invenio to Archivematica.

```python
invenio_archivematica.api.change_status_func = {<ArchiveStatus.FAILED: ‘FAILED’>: <function fail_transfer>, ...
    Dictionary that maps status to functions used to change the status.
invenio_archivematica.api.fail_transfer(sip, accession_id='', archivematica_id=None)
    Fail the archive process for a sip.
    Fail the transfer of the sip in asynchronous mode. See invenio_archivematica.tasks
    Parameters
    • sip(invenio_sipstore.api.SIP) – the sip to archive
```
• `accession_id (str)` – the accession_id

• `archivematica_id (str)` – the ID of the created AIP in Archivematica

`invenio_archivematica.api.finish_transfer(sip, accession_id='', archivematica_id=None)`

Finish the transfer of the sip in asynchronous mode. See `invenio_archivematica.tasks`

Parameters

• `sip (invenio_sipstore.api.SIP)` – the sip to archive

• `accession_id (str)` – the accession_id

• `archivematica_id (str)` – the ID of the created AIP in Archivematica

`invenio_archivematica.api.process_aip(sip, accession_id='', archivematica_id=None)`

Create the archive for a sip.

Process the aip of the sip in asynchronous mode. See `invenio_archivematica.tasks`

Parameters

• `sip (invenio_sipstore.api.SIP)` – the sip to archive

• `accession_id (str)` – the accession_id

• `archivematica_id (str)` – the ID of the AIP in Archivematica

`invenio_archivematica.api.process_transfer(sip, accession_id='', archivematica_id=None)`

Create the archive for a sip.

Process the transfer of the sip in asynchronous mode. See `invenio_archivematica.tasks`

Parameters

• `sip (invenio_sipstore.api.SIP)` – the sip to archive

• `accession_id (str)` – the accession_id

• `archivematica_id (str)` – the ID of the AIP in Archivematica

`invenio_archivematica.api.start_transfer(sip, accession_id, archivematica_id=None)`

Start the transfer of the sip in asynchronous mode. See `invenio_archivematica.tasks`

Parameters

• `sip (invenio_sipstore.api.SIP)` – the sip to archive

• `accession_id (str)` – the accessioned ID in archivematica. You can compute it from `invenio_archivematica.factories.create_accession_id()`

• `archivematica_id (str)` – the ID in Archivematica

### 3.1.2 Models

Archive models.

```python
class invenio_archivematica.models.Archive(**kwargs)

Registers the status of a sip: archived or not.
```

The status is a member of `invenio_archivematica.models.ArchiveStatus`. 
A sip can have only one archive, and an archive applies to only one sip.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

**accession_id**
Accessioned ID of the AIP in Archivematica.

**archivematica_id**
ID of the AIP in Archivematica.

**classmethod create**(sip, accession_id=None, archivematica_id=None)
Create a new Archive object and add it to the session.

The new Archive object will have a NEW status

**Parameters**

- **sip**(invenio_sipstore.models.SIP) – the sip attached to the archive
- **accession_id**(str) – the accession ID of the AIP
- **archivematica_id**(str) – The UUID of the AIP

**classmethod get_from_accession_id**(accession_id)
Return the Archive object associated to the given accession_id.

If the accession_id is not in the table, it returns None.

**Parameters**

- **accession_id**(str) – the accession_id of the Archive object.

**Return type**
invenio_archivematica.models.Archive or None

**classmethod get_from_sip**(uuid)
Return the Archive object associated to the given sip.

It tries to get the Archive object associated to the sip. If it exists, it returns it, otherwise it returns None.

**Parameters**

- **uuid**(str) – the uuid of the sip

**Return type**
invenio_archivematica.models.Archive or None

**id**
ID of the Archive object.

**sip**
Relationship with SIP.

**sip_id**
SIP related with the Archive.

**status**
Status of the archive.

**class** invenio_archivematica.models.ArchiveStatus
Constants for possible status of any given Archive object.

**DELETED** = ‘DELETED’
The archive has been deleted.

**FAILED** = ‘FAILED’
The sip has not been archived because of an error.
**IGNORED** = 'IGNORED'
   The sip won’t be archived.

**NEW** = 'NEW'
   The sip has been created or updated, but not yet archived.

**PROCESSING_AIP** = 'PROCESSING_AIP'
   The sip is currently being processed as an AIP (final step).

**PROCESSING_TRANSFER** = 'PROCESSING_TRANSFER'
   The sip is currently being processed as a transfer (first step).

**REGISTERED** = 'REGISTERED'
   The sip has been archived.

**WAITING** = 'WAITING'
   The sip has been transferred, and is waiting for processing.

**title**
Return human readable title.

invenio_archivematica.models.status_converter(status, aip_processing=False)
Convert a status given by Archivematica into an ArchiveStatus.

**Parameters**
- **status** *(str)* – a status returned by an Archivematica API
- **aip_processing** *(bool)* – tells if it is processing the AIP or the transfer

### 3.1.3 Views

Invenio-Archivematica views.
CHAPTER 4

Additional Notes

Notes on how to contribute, legal information and changes are here for the interested.

4.1 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

4.1.1 Types of Contributions

Report Bugs

If you are reporting a bug, please include:

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

Fix Bugs

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

Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.
Write Documentation

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

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/inveniosoftware/invenio-archivematica/issues.

If you are proposing a feature:

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

4.1.2 Get Started!

Ready to contribute? Here’s how to set up invenio-archivematica for local development.

1. Fork the inveniosoftware/invenio-archivematica repo on GitHub.
2. Clone your fork locally:

   ```bash
   $ git clone git@github.com:your_name_here/invenio-archivematica.git
   ```

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

   ```bash
   $ mkvirtualenv invenio-archivematica
   $ cd invenio-archivematica/
   $ pip install -e .[all]
   ```

4. Create a branch for local development:

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

   Now you can make your changes locally.

5. When you’re done making changes, check that your changes pass tests:

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

   The tests will provide you with test coverage and also check PEP8 (code style), PEP257 (documentation), flake8 as well as build the Sphinx documentation and run doctests.

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

   ```bash
   $ git add .
   $ git commit -s
   -m "component: title without verbs"
   -m "* NEW Adds your new feature."
   -m "* FIX Fixes an existing issue."
   -m "* BETTER Improves and existing feature."
   -m "* Changes something that should not be visible in release notes."
   $ git push origin name-of-your-bugfix-or-feature
   ```

7. Submit a pull request through the GitHub website.
4.1.3 Pull Request Guidelines

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

1. The pull request should include tests and must not decrease test coverage.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring.
3. The pull request should work for Python 2.7, 3.3, 3.4 and 3.5. Check https://travis-ci.org/inveniosoftware/invenio-archivematica/pull_requests and make sure that the tests pass for all supported Python versions.

4.2 Changes

Version 0.1.0 (released TBD)

- Initial public release.

4.3 License

MIT License

Copyright (C) 2017-2019 CERN.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

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.

4.4 Authors

Invenio 3 module to connect Invenio to Archivematica

- CERN <info@inveniosoftware.org>
- Rémi Ducceschi <remi.ducceschi@gmail.com>

4.2. Changes
Python Module Index

invenio_archivematica, 6
invenio_archivematica.api, 9
invenio_archivematica.config, 5
invenio_archivematica.ext, 9
invenio_archivematica.factories, 6
invenio_archivematica.models, 10
invenio_archivematica.tasks, 7
invenio_archivematica.views, 12
Index

A
accession_id (invenio_archivematica.models.Archive attribute), 11
Archive (class in invenio_archivematica.models), 10
ARCHIVEMATICA_BASE_TEMPLATE (in module invenio_archivematica.config), 5
ARCHIVEMATICA_DASHBOARD_API_KEY (in module invenio_archivematica.config), 5
ARCHIVEMATICA_DASHBOARD_URL (in module invenio_archivematica.config), 5
ARCHIVEMATICA_DASHBOARD_USER (in module invenio_archivematica.config), 5
archivematica_id (invenio_archivematica.models.Archive attribute), 11
ARCHIVEMATICA_ISARCHIVABLE_FACTORY (in module invenio_archivematica.config), 5
ARCHIVEMATICA_ORGANIZATION_NAME (in module invenio_archivematica.config), 5
ARCHIVEMATICA_STORAGE_API_KEY (in module invenio_archivematica.config), 5
ARCHIVEMATICA_STORAGE_URL (in module invenio_archivematica.config), 6
ARCHIVEMATICA_STORAGE_USER (in module invenio_archivematica.config), 6
ARCHIVEMATICA_TRANSFER_FACTORY (in module invenio_archivematica.config), 6
ARCHIVEMATICA_TRANSFER_FOLDER (in module invenio_archivematica.config), 6
ArchiveStatus (class in invenio_archivematica.models), 11

C
change_status_func (in module invenio_archivematica.api), 9
create() (invenio_archivematica.models.Archive class method), 11
create_accession_id() (in module invenio_archivematica.factories), 6

D
DELETED (invenio_archivematica.models.ArchiveStatus attribute), 11

F
fail_transfer() (in module invenio_archivematica.api), 9
FAILED (invenio_archivematica.models.ArchiveStatus attribute), 11
finish_transfer() (in module invenio_archivematica.api), 10

G
get_from_accession_id() (invenio_archivematica.models.Archive class method), 11
get_from_sip() (invenio_archivematica.models.Archive class method), 11

I
id (invenio_archivematica.models.Archive attribute), 11
IGNORED (invenio_archivematica.models.ArchiveStatus attribute), 11
init_app() (invenio_archivematica.ext.InvenioArchivematica method), 9
init_config() (invenio_archivematica.ext.InvenioArchivematica method), 9
init_listeners() (invenio_archivematica.ext.InvenioArchivematica method), 9
invenio_archivematica (module), 6
invenio_archivematica.api (module), 9
invenio_archivematica.config (module), 5
invenio_archivematica.ext (module), 9
invenio_archivematica.factories (module), 6
invenio_archivematica.models (module), 10
invenio_archivematica.tasks (module), 7
invenio_archivematica.views (module), 12
InvenioArchivematica (class in invenio_archivematica.ext), 9
is_archivable_default() (in module invenio_archivematica.factories), 6

19
is_archivable_none() (in module invenio_archivematica.factories), 6

NEW
NEW (invenio_archivematica.models.ArchiveStatus attribute), 12

process_aip() (in module invenio_archivematica.api), 10
process_transfer() (in module invenio_archivematica.api), 10
PROCESSING_AIP (invenio_archivematica.models.ArchiveStatus attribute), 12
PROCESSING_TRANSFER (invenio_archivematica.models.ArchiveStatus attribute), 12

REGISTERED (invenio_archivematica.models.ArchiveStatus attribute), 12

sip (invenio_archivematica.models.Archive attribute), 11
sip_id (invenio_archivematica.models.Archive attribute), 11
start_transfer() (in module invenio_archivematica.api), 10
status (invenio_archivematica.models.Archive attribute), 11
status_converter() (in module invenio_archivematica.models), 12

title (invenio_archivematica.models.ArchiveStatus attribute), 12
transfer_cp() (in module invenio_archivematica.factories), 6
transfer_demo() (in module invenio_archivematica.factories), 7
transfer_rsync() (in module invenio_archivematica.factories), 7

WAITING (invenio_archivematica.models.ArchiveStatus attribute), 12