invenio-workflows-ui Documentation

Release 0.1.0.dev20160000

CERN

October 20, 2016

Contents

Invenio module which acts as a UI layer for *invenio-workflows*.

This is an experimental developer preview release. While this module has been primarily developed for the INSPIRE-HEP High-Energy Physics use case, it is customizable enough to be used for any use case.

- Free software: GPLv2 license
- Documentation: https://pythonhosted.org/invenio-workflows-ui/

CHAPTER 1

User's Guide

This part of the documentation will show you how to get started in using Invenio-Workflows-UI.

1.1 Installation

1.2 Usage

In order to integrate your existing "invenio-workflows" with the user interface of *invenio-workflows-ui*, you need to configure your search (elasticsearch) mappings for the data structures in your workflows, add some additional properties to the workflow definitions and configure the UI and REST endpoints.

As an example lets add integration for incoming books, videos and article metadata. We will use *invenio-workflows-ui* as an interface to view, and accept or reject incoming items.

1.2.1 Setting up search indices

First we need to setup search indexes with the appropriate data model mappings. Similar to the integration of *invenio-search* and *invenio-records*. We add indexes with common alias "incoming" and several different data types underneath:

1. Create a folder called *mappings* somewhere in your overlay with your search mapping configuration, like this structure:

```
mappings/
    __init__.py
    incoming/
        book.json
        video.json
        article.json
```

Note, for *invenio-workflows-ui* to work correctly, your search mappings should define the extra fields specified under *invenio_workflows_ui/mappings/workflows/record.json*. This is used for filtering on status, created/modified date. Basically fields added under *_workflow.**.

2. Add the new mappings to the *invenio_search.mappings* entry-point, if necessary:

```
'invenio_search.mappings': [
    'incoming = youroverlay.mappings',
]
```

3. *pip install* your overlay again and recreate indexes:

```
pip install .
youroverlay index init
```

1.2.2 Adding configuration and workflow properties

Now we are ready to setup the configuration and extra properties in the workflow definitions.

In your workflow definitions, make sure you add the `data_type` attribute:

```
class MyBookWorkflow(object):
    data_type = "book"
    workflow = [a, b, c]
```

Then map the *data_type* value as the key to the search index and search type.

```
WORKFLOWS_UI_DATA_TYPES = dict(
    book=dict(
        search_index='incoming-book',
        search_type='book',
    ),
    video=dict(
        search_index='incoming-video',
        search_type='video',
    ),
    article=dict(
        search_index='incoming-article',
        search_type='article',
    ),
)
```

1.2.3 Configuring the API

Finally, you need to setup your UI and REST API endpoint configuration. We will have a UI endpoint called */incoming* under the root URL, e.g. http://localhost:5000/incoming

Then the REST API will be available under http://localhost:5000/api/incoming

```
WORKFLOWS_UI_URL = "/incoming"
WORKFLOWS_UI_API_URL = "/api/incoming/"
WORKFLOWS_UI_REST_ENDPOINT = dict(
    workflow_object_serializers={
        'application/json': ('invenio_workflows_ui.serializers'
                             ':json_serializer'),
    },
    search_serializers={
        'application/json': ('invenio_workflows_ui.serializers'
                             ':json_search_serializer'),
    },
    action_serializers={
        'application/json': ('invenio_workflows_ui.serializers'
                             ':json_action_serializer'),
    },
    bulk_action_serializers={
        'application/json': ('invenio_workflows_ui.serializers'
                             ':json_action_serializer'),
```

```
    },
    list_route='/incoming/',
    item_route='/incoming/<object_id>',
    search_index="incoming",    # <- main search alias for all "incoming" indices
    default_media_type='application/json',
    max_result_window=10000,
)
```

Note that the `search_index` value should be the same as the folder name containing the mappings for invenio-workflows-ui, e.g. "incoming".

1.3 Known Issues

1.3.1 Errors while building assets

In order to build assets correctly, you need to modify your instance `settings.js` file and add the following to the list of `paths`:

```
{
  hgn: "node_modules/requirejs-hogan-plugin/hgn",
  hogan: "node_modules/hogan.js/web/builds/3.0.2/hogan-3.0.2.amd",
  text: "node_modules/requirejs-hogan-plugin/text",
  flight: "node_modules/flightjs/build/flight"
}
```

If you use *invenio-theme*, you may need to adjust the settings.js file contained within the module. Or, alternatively, roll your own bundle with an updated *settings.js* file.

CHAPTER 2

API Reference

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

2.1 API Docs

CHAPTER 3

Additional Notes

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

3.1 Contributing

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

3.1.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/inspirehep/invenio-workflows-ui/issues.

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-workflows-ui could always use more documentation, whether as part of the official invenio-workflows-ui 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/inspirehep/invenio-workflows-ui/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 :)

3.1.2 Get Started!

Ready to contribute? Here's how to set up *invenio* for local development.

1. Fork the *invenio* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/invenio-workflows-ui.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:

```
$ mkvirtualenv invenio-workflows-ui
$ cd invenio-workflows-ui/
$ pip install -e .[all]
```

4. Create a branch for local development:

```
$ 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:

```
$ ./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:

```
$ git add .
$ git commit -s -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

3.1.3 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.com/inspirehep/invenio-workflows-ui/pull_requests and make sure that the tests pass for all supported Python versions.

3.2 Changes

Version 0.1.0 (released TBD)

- Initial public release.

3.3 License

3.4 Authors

Invenio module which acts as a UI layer for invenio-workflows.

- Guillaume Lastecoueres <PX9e@gmx.fr>
- Jan Aage Lavik <jan.age.lavik@cern.ch>