# Contents

1 User's Guide 3  
1.1 Installation ........................................ 3  
1.2 Configuration ........................................ 3  
1.3 Example application .................................. 4  
2 API Reference 5  
2.1 API Docs ............................................. 5  
3 Additional Notes 9  
3.1 Contributing .......................................... 9  
3.2 Changes ............................................. 11  
3.3 License ............................................. 11  
3.4 Contributors ......................................... 12  

Python Module Index 13  

Index 15
User profiles module for Invenio.

Features:

- Editing and managing user profiles (username, full name).
- Extends the user registration form provided by Invenio-Accounts.
- Resending user verification emails.
- Administration interface for user profiles.

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

## 1.1 Installation

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

```bash
$ pip install invenio-userprofiles
```

## 1.2 Configuration

Default configuration.

```python
invenio_userprofiles.config.USERPROFILES = True
   # Enable or disable module extensions.

invenio_userprofiles.config.USERPROFILES_BASE_TEMPLATE = None
   # Base templates for user profile module.

invenio_userprofiles.config.USERPROFILES_EMAIL_ENABLED = True
   # Include the user email in the profile form.

invenio_userprofiles.config.USERPROFILES_EXTEND_SECURITY_FORMS = False
   # Extend the Invenio-Accounts user registration forms.

invenio_userprofiles.config.USERPROFILES_PROFILE_TEMPLATE = 'invenio_userprofiles/settings/profile.html'
   # Default profile template.

invenio_userprofiles.config.USERPROFILES_PROFILE_URL = '/account/settings/profile/'
   # Default profile URL endpoint.

invenio_userprofiles.config.USERPROFILES_SETTINGS_TEMPLATE = None
   # Settings base templates for user profile module.
```
1.3 Example application

First install Invenio-UserProfiles, setup the application and load fixture data by running:

```
$ pip install -e .[all]
$ cd examples
$ ./app-setup.sh
$ ./app-fixtures.sh
```

You should also have the Redis running on your machine. To know how to install and run redis, please refer to the redis website.

Next, start the development server:

```
$ export FLASK_APP=app.py FLASK_DEBUG=1
$ flask run
```

and open the example application in your browser:

```
$ open http://127.0.0.1:5000/
```

Choose a user to login:

- user info@inveniosoftware.org password 123456
- user another@inveniosoftware.org password 123456

You can check the administration page opening the page:

```
$ open http://127.0.0.1:5000/admin
```

Note that, as defined in our fixtures, only info@inveniosoftware.org user can enter.

To uninstall and purge the example app, run:

```
$ ./app-teardown.sh
```
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

2.1.1 Extension

User profiles module for Invenio.

```python
class invenio_userprofiles.ext.InvenioUserProfiles(app=None)
   Invenio-UserProfiles extension.
   Extension initialization.
   init_app(app)
      Flask application initialization.
   init_config(app)
      Initialize configuration.
```

2.1.2 Admin

2.1.3 API

API for user profiles.

```python
invenio_userprofiles.api.current_userprofile
   Proxy to the user profile of the currently logged in user.
```
### 2.1.4 Models

Database models for user profiles.

```python
class invenio_userprofiles.models.AnonymousUserProfile
    Anonymous user profile.

    is_anonymous
    Return whether this UserProfile is anonymous.

class invenio_userprofiles.models.UserProfile(**kwargs)
    User profile model.

    Stores a username, display name (case sensitive version of username) and a full name for a user.
    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.

    full_name
    Full name of person.

classmethod get_by_userid(user_id)
    Get profile by user identifier.

    Parameters user_id – Identifier of a User.

    Returns A UserProfile instance or None.

classmethod get_by_username(username)
    Get profile by username.

    Parameters username – A username to query for (case insensitive).
```

```python
is_anonymous
    Return whether this UserProfile is anonymous.

user
    User relationship.

user_id
    Foreign key to User.

username
    Get username.
```

```python
invenio_userprofiles.models.on_user_init(target, args, kwargs)
    Provide hook on User initialization.

    Automatically convert a dict to a UserProfile instance. This is needed during e.g. user registration where
    Flask-Security will initialize a user model with all the form data (which when Invenio-UserProfiles is enabled
    includes a profile key). This will make the user creation fail unless we convert the profile dict into a
    UserProfile instance.
```

### 2.1.5 Forms

Forms for user profiles.
class invenio_userprofiles.forms.EmailProfileForm(formdata=<object object>, **kwargs)
   Form to allow editing of email address.

class invenio_userprofiles.forms.ProfileForm(formdata=<object object>, **kwargs)
   Form for editing user profile.

   validate_username(field)
   Wrap username validator for WTForms.

class invenio_userprofiles.forms.VerificationForm(formdata=<object object>, **kwargs)
   Form to render a button to request email confirmation.

invenio_userprofiles.forms.confirm_register_form_factory(Form)
   Factory for creating a confirm register form.

invenio_userprofiles.forms.current_user_email(form, field)
   Field validator to stop validation if email wasn’t changed.

invenio_userprofiles.forms.register_form_factory(Form)
   Factory for creating an extended user registration form.

invenio_userprofiles.forms.strip_filter(text)
   Filter for trimming whitespace.

   Parameters text – The text to strip.

   Returns The stripped text.

2.1.6 Validators

Validators for user profiles.

invenio_userprofiles.validators.USERNAME_RULES = l’Username must start with a letter, be at least three characters long and only contain alphanumeric characters, dashes and underscores.’
   Description of username validation rules.

   Note: Used for both form help text and for form validation error.

invenio_userprofiles.validators.username_regex = re.compile('^[a-zA-Z][a-zA-Z0-9-\_]{2}[a-zA-Z0-9-\_]*$')
   Username rules.

invenio_userprofiles.validators.validate_username(username)
   Validate the username.

   See username_regex to know which rules are applied.

   Parameters username – A username.

   Raises ValueError – If validation fails.

2.1.7 Views

Invenio module that adds userprofiles to the platform.

invenio_userprofiles.views.handle_profile_form(form)
   Handle profile update form.

invenio_userprofiles.views.handle_verification_form(form)
   Handle email sending verification form.
invenio_userprofiles.views.init_common(app)
    Post initialization.

invenio_userprofiles.views.profile()
    View for editing a profile.

invenio_userprofiles.views.profile_form_factory()
    Create a profile form.

invenio_userprofiles.views.userprofile(value)
    Retrieve user profile for a given user id.
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


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-UserProfiles could always use more documentation, whether as part of the official Invenio-UserProfiles 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-userprofiles/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-userprofiles for local development.

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

   ```
   $ git clone git@github.com:your_name_here/invenio-userprofiles.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-userprofiles
   $ cd invenio-userprofiles/
   $ 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 "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.
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/inveniosoftware/invenio-userprofiles/pull_requests and make sure that the tests pass for all supported Python versions.

3.2 Changes

Version 1.1.1 (released 2020-05-11)
- Minimum version of Invenio-Accounts bumped to v1.2.1 due WTForms moving the email validation to an optional dependency

Version 1.1.0 (released 2020-03-11)
- Change Flask dependency management to invenio-base
- drop Python 2.7 support

Version 1.0.1 (released 2018-05-25)
- Flask v1.0 support.

Version 1.0.0 (released 2018-03-23)
- Initial public release.

3.3 License

MIT License

Copyright (C) 2015-2018 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.

Note: 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.
3.4 Contributors

• Alizée Pace
• Bruno Cuc
• Diego Rodriguez
• Eamonn Maguire
• Harri Hirvonsalo
• Jacopo Notarstefano
• Jiri Kuncar
• Lars Holm Nielsen
• Leonardo Rossi
• Nicola Tarocco
• Orestis Melkonian
• Paulina Lach
• Sami Hiltunen
• Sebastian Witowski
• Tibor Simko
• Vaibhav Gupta
invenio_userprofiles.api, 5
invenio_userprofiles.config, 3
invenio_userprofiles.ext, 5
invenio_userprofiles.forms, 6
invenio_userprofiles.models, 6
invenio_userprofiles.validators, 7
invenio_userprofiles.views, 7
Index

A
AnonymousUserProfile (class in invenio_userprofiles.models), 6

C
confirm_register_form_factory() (in module invenio_userprofiles.forms), 7
current_user_email() (in module invenio_userprofiles.forms), 7
current_userprofile (in module invenio_userprofiles.api), 5

E
EmailProfileForm (class in invenio_userprofiles.forms), 6

F
full_name (invenio_userprofiles.models.UserProfile attribute), 6

G
get_by_userid() (invenio_userprofiles.models.UserProfile method), 6
get_by_username() (invenio_userprofiles.models.UserProfile method), 6

H
handle_profile_form() (in module invenio_userprofiles.views), 7
handle_verification_form() (in module invenio_userprofiles.views), 7

I
init_app() (invenio_userprofiles.ext.InvenioUserProfiles method), 5
init_common() (in module invenio_userprofiles.views), 8

O
on_user_init() (in module invenio_userprofiles.models), 6

P
profile() (in module invenio_userprofiles.views), 8
profile_form_factory() (in module invenio_userprofiles.views), 8
ProfileForm (class in invenio_userprofiles.forms), 7

R
register_form_factory() (in module invenio_userprofiles.forms), 7

S
strip_filter() (in module invenio_userprofiles.forms), 7
U
user (invenio_userprofiles.models.UserProfile attribute), 6
user_id (invenio_userprofiles.models.UserProfile attribute), 6
username (invenio_userprofiles.models.UserProfile attribute), 6
username_regex (in module invenio_userprofiles.validators), 7
USERNAME_RULES (in module invenio_userprofiles.validators), 7
UserProfile (class in invenio_userprofiles.models), 6
userprofile() (in module invenio_userprofiles.views), 8
USERPROFILES (in module invenio_userprofiles.config), 3
USERPROFILES_BASE_TEMPLATE (in module invenio_userprofiles.config), 3
USERPROFILES_EMAIL_ENABLED (in module invenio_userprofiles.config), 3
USERPROFILES_EXTEND_SECURITY_FORMS (in module invenio_userprofiles.config), 3
USERPROFILES_PROFILE_TEMPLATE (in module invenio_userprofiles.config), 3
USERPROFILES_PROFILE_URL (in module invenio_userprofiles.config), 3
USERPROFILES_SETTINGS_TEMPLATE (in module invenio_userprofiles.config), 3

V
validate_username() (in module invenio_userprofiles.validators), 7
validate_username() (invenio_userprofiles.forms.ProfileForm method), 7
VerificationForm (class in invenio_userprofiles.forms), 7