INSPIRE for developers

INSPIRE is built on the free digital library software Invenio. All our software is free software.

Back to main pages

Development principles

Most of our code will be contributed back into the Invenio software, so please read carefully the standard coding guidelines and Python PEP8 standards. These rules will be enforced by maintainers and integrators.

In order to work in a really agile manner, the basic principles are rather simple:

- make things configurable; do not hard-code INSPIRE specific information or tags;
- make branches integration-friendly at any time; if a feature is not ready, then protect it by a switch, and turn it off by default, so that an eventual merge would not disturb anyone;
- write test cases and use ondemand jenkins builds to signal branch kwalitee: a tested branch with green light is on a faster track than an untested branch with no light.

- Tibor Simko

Environment

Typically we work in Debian/Ubuntu based environment, but any flavor of GNU/Linux works.

Before you move on, you should make sure that your computer is up-to-date and ready to be used and configured to your liking. See also our list of commonly suggested tools, if you are interested.

Exercises and tutorials

During your first week with us, you will be going through a series of exercises and lessons to get you into how we do development here at the INSPIRE team, with the hope of smoothen the introduction to the Invenio software suite you will be working with.

Please complete these exercises in order:

Git + guide

Special set of exercises to get you acquainted with our workflow with Git: Start

Please also read the Git Flow guide on GitHub to understand better the way we provide contributions to Invenio with GitHub.

Extra reading material:

- https://help.github.com/articles/using-pull-requests
- http://invenio-software.org/wiki/Tools/Git/
- http://invenio-software.org/wiki/Tools/Git/Workflow (informative!)
Invenio v.1.x.x guide

This exercise gives you an introduction to Invenio v.1.x.x. It is recommended that you install Invenio v.1.x.x first.

Invenio introduction (master branch): Start

Invenio v.2.x.x guide

Invenio introduction (next/pu branch): Start

Invenio: creating a new module in pu: (in development)

See also

- Python
  - http://invenio-software.org/wiki/Tools/Python/GettingStarted
  - Python style guide

Contributing code

Right now, most of our code repositories are hosted mainly at Github, either on Invenio Software or INSPIRE-HEP. Github is the primary place to post issues/bugs and ask for your work to be merged via pull requests.

We also have a discussion forum running here and we are on Hangout (ask for invite), Gitter and IRC (#invenio @ freenode.net).

Before you start

Before starting coding on Invenio, you should also have a look at these guides:

- INSPIRE design principles mandatory for front-end development
- Coding style
- Testing (see also this book)

Again, bear in mind the basic principles of contributing to Invenio software.

Invenio repositories

An important distinction is the difference between Invenio official repositories and our own INSPIRE repositories.

The core Invenio software sources are located at GitHub. This repository contain several branches.

See also other related packages of Invenio on GitHub

INSPIRE repositories

INSPIRE has its own overlay repository, which basically contains customizations of the core Invenio software (templates, scripts, formats etc.). This is also where you will find the INSPIRE demo site.
For Invenio master branches, the overlay is located here.

For Invenio pu branch, the overlay is here.

If you are working on content ingestion workflows (such as content flows from APS, Elsevier etc.) you will also find the Python module harvesting-kit interesting.

Our full list of repositories (including master overlay mirror) is available on our GitHub pages.

Finally, for the INSPIRE production code we have a repository (or really a special branch) called ops (or OPS), with a branch called "prod".

**NOTE:** The ops repository is only used by operators to deploy code to production. It is not for development.

## Installation

And once you feel confident with all the previous steps, go ahead and install Invenio by following the guides below.

We recommend that you have Python 2.7.5+ installed with newest pip and virtualenv + virtualenvwrapper. This way you can install Invenio inside a virtual Python environment tailored for development.

### Installation of master (Invenio v.1.x.x)

Here is an updated guide to install Invenio master branch with virtualenv: installation guide.

### Installation of pu (Invenio v.2.x.x)

Current docs: https://invenio.readthedocs.org

For installation of pu follow the INSTALL file in the repo, or this guide.

### A note on Invenio's new framework (branch pu)

If you are going to be working on Invenio's next or pu branch, which makes use of Flask, SQLAlchemy and Jinja2, please read:

- Flask - Invenio integration
- Flask - getting started
- SQLAlchemy - getting started
- Jinja - getting started

Usually one wants to emulate the structure of documents when migrating/creating modules for pu.

## Common tools

In addition to the required tools we use, such as git, we also have a range of common programs many of our developers are using among themselves:

- iPython (recommended) (url): interactive shell for the Python programming language that offers enhanced introspection, additional shell syntax, tab completion and rich history.
Invenio Devscripts (url): contains a collection of scripts useful when hacking on Invenio.

virtualenv (url): tool to create Python isolated environments.

Terminator: GPL terminal emulator.

Sublime Text (url): fast, sophisticated text editor for code, markup and prose. We have licenses available (contact javier.martin.montull@cernNOSPAMPLEASE.ch)

Atom (url) - The new trendy editor from the GitHub guys. (in beta/development)

PyCharm (url): Intelligent Python IDE with unique code assistance and analysis, for productive Python development. Open source community edition now available!

Emacs/Vim: The classic text editors / environments we all love (or hate).

Sublime Text has a plethora of plugins available. First of all, install Sublime Package Control.

Then use it to install the plugins you want. Here is a list of suggestions:

- Anaconda: All-in-one package for Python development. Remember to amend the project settings to add the right Python executable in your virtualenv.
- Python Checker: Alternative to Anaconda. Will highlight PEP8 issues and lints.

You should also personalize your editor settings, for example to auto-remove white-space, spell-checking etc. For example:

```
$ cat ~/.config/sublime-text-3/Packages/User/Preferences.sublime-settings
{
    "always_show_minimap_viewport": true,
    "auto_complete_triggers":
    [
        {
            "characters": ".",
            "selector": "source.python - string - comment - constant.numeric"
        }
    ],
    "bold_folder_labels": true,
    "caret_style": "solid",
    "dictionary": "Packages/Language - English/en_US.dic",
    "draw_minimap_border": true,
    "enable_telemetry": false,
    "ensure_newline_at_eof_on_save": true,
    "folder_exclude_patterns":
    [
        ".svn",
        ".git",
        ".hg",
        "CVS",
        "__pycache__"
    ],
    "font_options":
    [
        "subpixel_antialias"
    ],
    "font_size": 10,
    "highlight_line": true,
    "highlight_modified_tabs": true,
```
"ignored_packages":
  [
    "Vintage"
  ],
"indent_guide_options":
  [
    "draw_active",
    "draw_normal"
  ],
"indent_to_bracket": true,
"line_padding_bottom": 0,
"line_padding_top": 0,
"rulers":
  [
    79
  ],
"shift_tab_unindent": true,
"show_panel_on_build": false,
"spell_check": true,
"translate_tabs_to_spaces": true,
"trim_trailing_white_space_on_save": true,
"wrap_width": 80
}