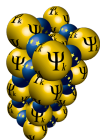




The AiiDA plugin ecosystem

Leopold Talirz



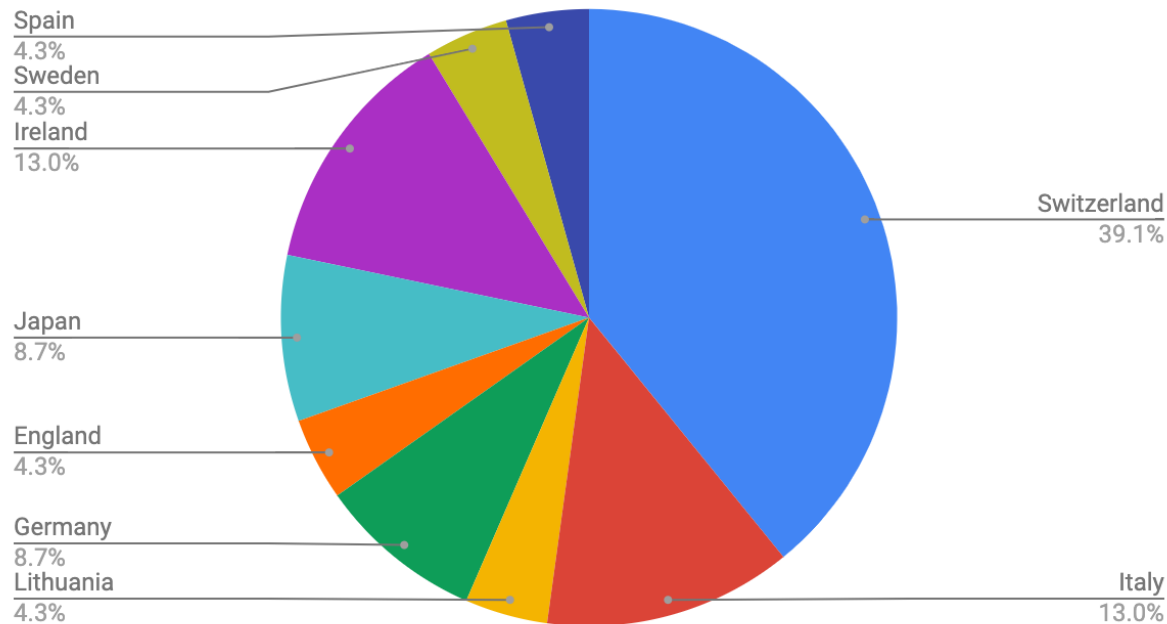
swissuniversities



AiiDA-powered research projects

- Questionnaire on AiiDA mailing list last week
 - 27 research projects
 - 25 plugins

Country of affiliation



Outline

- ① What's an AiiDA plugin?
- ② Python entry points
or: How AiiDA discovers plugins
- ③ Example: aiida-diff demo plugin
- ④ Developing a plugin

What's an AiiDA plugin?

AiiDA plugin = python package that extends **aiida-core**

aiida-core

- database model
- provenance graph
- workflow engine
- REST API
- ...

What's an AiiDA plugin?

AiiDA plugin = python package that extends **aiida-core**

aiida-core

- database model
- provenance graph
- workflow engine
- REST API
- ...

aiida-<plugin>

- calculation plugins
- parser plugins
- workflows
- data types
- verdi commands
- schedulers
- transports
- db importers/exporters
- future: REST endpoints

AiiDA plugin registry

AiiDA registry of plugins

[\[View on GitHub/register your plugin\]](#)

Global summary of the AiiDA plugin registry

Total number of entries: 36

Calculations	62 plugins in 25 entries
Parsers	55 plugins in 25 entries
Data	24 plugins in 13 entries
Workflows	53 plugins in 11 entries
Other	40 plugins in 12 entries

Available plugins (alphabetically sorted)

[alloy: aiida-alloy package](#)

Current state: development

[Plugin code homepage](#) (hosted on github.com)

Documentation: Documentation not provided by the plugin author

[Show plugin details](#)

[ase: aiida-ase package](#)

The official AiiDA plugin for ASE

Current state: stable

[Plugin code homepage](#) (hosted on github.com)

Documentation: [Go to plugin documentation](#)

Calculations 1 Parsers 1

[Show plugin details](#)

Python Entry Points

The problem

- aiida-siesta provides a new StmCalculation
Q: Where to import it from?
- aiida-vasp provides a new VaspPotcarData
Q: How can I add a verdi command for it?
- How do I list all installed workflows?
- ...

Goal

Plugins can **extend** standard AiiDA interfaces.

What is a python entry point?

Python entry points associate **python objects** with free-form **identifiers**

aiida-diff plugin - setup.json file

```
"entry_points": {
  "aiida.data": [
    "diff = aiida_diff.data:DiffParameters"
  ],
  "aiida.calculations": [
    "diff = aiida_diff.calculations:DiffCalculation"
  ],
  "aiida.parsers": [
    "diff = aiida_diff.parsers:DiffParser"
  ],
  "aiida.cmdline.data": [
    "diff = aiida_diff.cli:data_cli"
  ]
},
```

group **identifier** **Object (class, function, variable, ...)**

groups are defined by aiida-core

How to load an entry point

Loading an entry point

```
In [1]: import pkg_resources
```

```
In [2]: DiffCalculation = pkg_resources.load_entr  
....: y_point(dist='aiida-diff', group='aiida.c  
....: alculations', name='diff')
```

```
In [3]: print(DiffCalculation)  
<class 'aiida_diff.calculations.DiffCalculation'>
```

`CalculationFactory('diff')` does this

What can AiiDA plugins provide?

`aiida.calculations`

`aiida.parsers`

`aiida.workflows`

`aiida.data`

`aiida.cmdline`

`aiida.schedulers`

`aiida.transports`

`aiida.tools.dbimporters`

`aiida.tools.dbexporters`

`aiida-<plugin>`

- calculation plugins
- parser plugins
- workflows
- data types
- verdi commands
- schedulers
- transports
- db importers/exporters
- future: REST endpoints

List using: `verdi plugin list`

aiida-core uses entry points as well!

```
"aiida.transports": [  
    "local = aiida.transports.plugins.local:LocalTransport",  
    "ssh = aiida.transports.plugins.ssh:SshTransport"  
],  
"aiida.schedulers": [  
    "direct = aiida.schedulers.plugins.direct:DirectScheduler",  
    "lsf = aiida.schedulers.plugins.lsf:LsfScheduler",  
    "pbspro = aiida.schedulers.plugins.pbspro:PbsproScheduler",  
    "sge = aiida.schedulers.plugins.sge:SgeScheduler",  
    "slurm = aiida.schedulers.plugins.slurm:SlurmScheduler",  
    "torque = aiida.schedulers.plugins.torque:TorqueScheduler"  
],
```

E.g. `verdi plugin list aiida.transports`

Full list in [setup.json](#) file in the aiida_core repository

pkg_resources and reentry

- `pkg_resources` (part of `setuptools`)
provides registry and lookup for entry points
- `pkg_resources` does integrity checks => can be slow
- we use entry points in the `verdi cli` => should be fast
- `reentry` package
 - separate entry point cache
 - 10x faster
 - but: need to refresh cache using `reentry scan`
after installing a new aiida plugin

aiida-diff demo plugin

diff: aiida-diff package

AiiDA demo plugin that computes the difference between two files.
Current state: development

[Plugin code homepage](#) (hosted on github.com)

Documentation: [Go to plugin documentation](#)

Calculations 1 Parsers 1 Data 1 Other (Cmdline data) 1

[Show plugin details](#)

- A demo plugin for getting started with plugin development
 - wraps the `diff` executable
 - `diff` is available on almost every UNIX system
 - `diff` has command line options (e.g. `-i`)
 - `diff` takes 2 input files & produces 1 output file

```
diff file1 file1 > file3
```
- Covers a number of use cases already

DEMO - aiida-diff

Branch: master ▾ **aiida-diff** / aiida_diff /

Create new file Upload files Find file History

 **Italirz** transform get_computer and get_code into fixtures (#22) ... Latest commit 2ae3ac1 2 hours ago

..

data	improve tests of data cli (#21)	a day ago
tests	transform get_computer and get_code into fixtures (#22)	2 hours ago
__init__.py	transform get_computer and get_code into fixtures (#22)	2 hours ago
calculations.py	transform get_computer and get_code into fixtures (#22)	2 hours ago
cli.py	improve tests of data cli (#21)	a day ago
helpers.py	transform get_computer and get_code into fixtures (#22)	2 hours ago
parsers.py	improve tests of data cli (#21)	a day ago

Free time investment advice:
Learn how to write tests (with `pytest`)!

Plugin design guidelines

1. **Start simple.**

Use existing classes like `Dict`, `SinglefileData`, ...

Write only what is necessary to pass information from/to AiiDA.

2. **Don't break data provenance.**

Store *at least* what is needed for full reproducibility.

3. **Parse what you want to query for.**

Make a list of which information to:

1. parse into the database for querying (`Dict`, ...)
2. store in files for safe-keeping (e.g. `SinglefileData`, ...)
3. leave on the remote computer (`RemoteData`, ...)

4. **Expose the full functionality of your code.**

Don't artificially limit the power of a code you are wrapping.

If the code can do it, there should be *some* way to do it with your plugin.

Transition to aiida-core 1.0

- Many API changes from aiida 0.12 to 1.0
- AiiDA API **should remain stable** from aiida-core 1.0.0b2

Videos

- [aiida-core 1.0.0b2 \(plugin migration lectures\)](#)
- [aiida-core 0.9.0 \(lectures\)](#)
- [aiida-core 0.8.0 \(short videos\)](#)
- [aiida-core 0.6.0 \(lectures\)](#)

aiida-tutorials.readthedocs.io

Useful resources

- [AiiDA release roadmap](#)
- [AiiDA plugin registry](#)
- [Git cheatsheet](#)
- [How to migrate from BitBucket to GitHub](#)
- [RST tutorial for GitHub](#)
- [AiiDA 1.0 plugin migration guide](#)

github.com/aiidateam/aiida_core/wiki

Transition to aiida-core 1.0



Itairlz commented 27 days ago • edited ▾

Member



This is a list of plugins that are compatible with `aiida-core` 1.0.0b2 (or later) and can therefore be included in the `develop` branch of the AiiDA lab:

- ☐ aiida-castep: in progress in branch [aiida_1.0_compatible](#)
- ☒ aiida-cp2k: new release v1.0.0b1
- ☒ aiida-diff: new release v1.0.0a1
- ☐ aiida-fleur: in progress in [develop](#) branch
- ☐ aiida-lammps: in progress in master branch
- ☐ aiida-quantum_espresso: release v3.0.0a3 works for `pw.x` and already a few more codes with aiida-core 1.0.0b3; in progress in branch [migrate_aiida1.0](#);
- ☒ aiida-qeq: new release v1.0.0a1
- ☐ aiida-siesta: progress in branch [workflow-migration](#)
- ☐ aiida-vasp: in progress in branch [migrate_beta](#)
- ☐ aiida-yambo: in progress in branch [dev_aiida1.x](#)
- ☒ aiida-zeopp: new release v1.0.0a2

Is your plugin of interest on this list? If not => Group B

github.com/aiidalab/aiidalab-metapkg/issues/10

Practical guidelines

1. Check on the [aiida plugin registry](#) that your desired plugin name is still available
2. Use the [AiiDA plugin cutter](#) to jumpstart your plugin:

```
pip install cookiecutter
cookiecutter https://github.com/aiidateam/aiida-plugin-cutter.git
# follow instructions ...
cd aiida-mycode
```

3. Get your new plugin registered on the plugin registry
4. Write **tests** for your plugin
5. Take advantage of free **continuous integration** platforms

Group B

Start writing a plugin for your code

Material to be covered

- aiida-plugin-cutter
- get your new plugin registered
- how to **test** plugins
- Advanced example: aiida-siesta
- pre-commit hooks & IDEs
- **aiida-siesta**: Intro & plugin structure

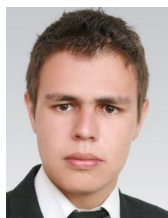
Tutors

- Leopold Talirz
- Oscar Arbelaes
- Daniele Tomerini
- Alberto Garcia

Recap

- aiida plugins are python packages that **extend** existing AiiDA interfaces through python entry points
- when developing a new plugin
 1. start simple
 2. don't start from scratch
 3. use git for version control & github for collaboration
 4. write tests!

AiiDA & Materials Cloud Teams



Oscar D.
Arbelaez
(EPFL)



Marco
Borelli
(EPFL)



Valeria
Granata
(EPFL)



Sebastiaan
P. Huber
(EPFL)



Leonid
Kahle
(EPFL)



Boris
Kozinsky
(BOSCH)



Snehal P.
Kumbhar
(EPFL)



Nicola
Marzari
(EPFL)



Elsa
Passaro
(EPFL)



Giovanni
Pizzi
(EPFL)



Thomas
Schulthess
(ETHZ,CSCS)



Berend
Smit
(EPFL)



Leopold
Talirz
(EPFL)



Daniele
Tomerini
(EPFL)



Joost
VandeVondele
(ETHZ,CSCS)



Casper
Welzel
(EPFL)



Aliaksandr
Yakutovich
(EPFL)

Contributors for the 30+ plugins: [Quantum ESPRESSO](#), [Wannier90](#), [CP2K](#), [FLEUR](#), [YAMBO](#), [SIESTA](#), [VASP](#), ...

Contributors to `aiida_core` and former AiiDA team members — Valentin Bersier, Jocelyn Boullier, Jens Broeder, Andrea Cepellotti, Fernando Gargiulo, Dominik Gresch, Conrad Johnston, Rico Häuselmann, Eric Hontz, Christoph Koch, Espen Flage-Larsen, Antimo Marrazzo, Andrius Merkys, Nicolas Mounet, Tiziano Müller, Riccardo Sabatini, Ole Schütt, Phillippe Schwaller, Andreas Stamminger, Martin Uhrin, Spyros Zoupanos

The CSCS support teams

