eprc Documentation

Release 0.1.0

Marco Neumann

May 20, 2015

Contents

This is an **experimental** approach to fix one of the most fundamental issues with pip: The lack of a dependency resolver.

> **Warning:** Do **not** use this on production systems. It is highly experimtal and may contain security bugs!

CHAPTER 1

Requirements

To use eprc, you need the following components:

- Python 2.7 (for eprc and the extractors)
- Java 7 or 8 (for the solver)
- Redis (for caching/storage)

CHAPTER 2

Usage

Make sure that Redis is up and running and that the project that you want to install has a *setup.py*. Now lets generate a *requirements.txt* file that has a closed and optimal set of packages:

```
eprc calc path/to/project
```

Hint: This might take a while because eprc fetches all versions of all somehow required packages and tries to extras meta information. This is only done once because Redis caches this meta data for future sessions. There are plans to set up a public caching server.

Caution: In case that eprc is not able to find a requirements set without conflicts, you will only get a message and no *requirements.txt* file will be written. Sadly there won't be any information about the packages that resulted in the conflict.

In case that eprc was able to find a requirements set without conflicts, you can now install this set and your project:

```
pip install requirements.txt
cd path/to/project && python setup.py install
```

Tip: Use *eprc calc –help* to get more information about the different options and how to calculate a requirements set for multiple projects simultaneously.

CHAPTER 3

Contents

3.1 Database

class eprc.database.**Database**(*host*, *port*, *db*)

all_versions(*name*)

get(*name*, *version*)

static name_version_to_key(*name*, *version*)

set(*name*, *version*, *data*)

3.2 Extractor

class eprc.extractor.**Extractor**(*virtualenv*, *tmpdir*, *pypi*, *extractors_path='/var/build/user_builds/eprc/checkouts/latest/eprc/extractors'*)

from_native(*db*, *name*)

from_path(*path*, *db*, *name=None*, *version=None*)

from_pypi(*db*, *name*, *version*)

3.3 PyPi

class eprc.pypi.**PyPi**

package_releases(*name*)
Use weird PIP system instead of the official PyPi API.

They sometimes provide different results. But because eprc is intended to be used with PIP, we accept this buggy system here.

real_name(*package_name*, *timeout=None*)
Replaces buggy pkgtools.pypi.real_name.

release_urls(*name*, *version*)

3.4 Scheduler

class eprc.scheduler.**Scheduler**(*db*, *extractor*, *pypi*, *verbosity=1*)

add_todos_from_db(*name*, *version*, *extra=''*)

blacklist_version(*name*, *version*)

done_with_all_versions(*name*, *extra*)

get()

is_version_blacklisted(*name*, *version*)

process_cached(*name*, *extra*)

process_extract(*name*, *extra*)

3.5 Solver

class eprc.solver.**VariableRegister**

VIRTUAL_VERSION = ('*virtual', '*final')

get_virtual_variable()

register_set(*name*, *versions*, *extras*)

register_single(*name*, *version*, *extras*)

eprc.solver.**solve**(*scheduler*, *db*, *must_satisfy*, *tmpdir*, *solver*, *outpath*, *include_starting_points=False*)

3.6 Utils

exception eprc.utils.**HandledError**(*msg*, **args*, ***kwargs*)

eprc.utils.**TemporaryDirectory**(**args*, ***kwds*)

eprc.utils.**normalize**(*string*)

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

e