collective.table Documentation

Release 0.2alpha

August 29, 2011

CONTENTS

Project title collective.table

Latest version 0.2alpha

URL http://pypi.python.org/pypi/collective.table

Summary

This is a Plone 4 add-on project for storing tabular data inside Plone.

CHAPTER

ONE

TABLE OF CONTENTS

1.1 Definitions of basic terms

TODO Add some glossary entries.

1.2 Developer conventions

1.2.1 Quick start

If you have a Linux or OS X system already capable of running Plone then a quick start with collective.table is as follows:

```
$ git co git@github.com:nzupan/collective.table.git
$ cd collective.table/
$ virtualenv -p python2.6 --no-site-packages ./
$ bin/python bootstrap.py
$ bin/buildout

# start Zope
$ bin/instance fg
```

1.2.2 Commit checklist

Before every commit you must:

- run all *Unit tests*.
- perform *Syntax validation*.
- add an entry to *Changelog* (if applicable).
- add an entry to *Sphinx documentation* (if applicable).

You can run all syntax check and all test with a single command:

```
$ ./pre-commit-check.sh
```

1.2.3 Unit tests

Un-tested code is broken code.

For every feature you add to the codebase you must also add tests for it.

You can run tests like this:

```
.. sourcecode:: bash
```

    $ bin/test -s collective.table

1.2.4 Syntax validation

All Python source code should be *PEP-8* valid and checked for syntax errors. Tools for checking this are *pep8* and *pyflakes*.

If possible make your editor run *pep8* and *pyflakes* on your current file every time you save it. Useful links:

- http://github.com/ppierre/python-pep8-tmbundle
- http://www.arthurkoziel.com/2008/06/28/pyflakes-installation-and-textmate-integration/

Alternatively you can use these two commands to check style manually:

```
$ bin/pyflakes collective
$ bin/pep8 collective
```

1.2.5 Changelog

We track all feature-level changes to code inside `docs/HISTORY.txt`. Examples:

- added feature X
- removed Y
- fixed bug Z

1.2.6 Sphinx documentation

Un-documented code is broken code.

For every feature you add to the codebase you must also add documentation for it in `docs/sphinx/`.

After adding documentation, re-build *Sphinx* and check how it is displayed:

```
.. sourcecode:: bash
```

    $ bin/sphinxbuilder $ open docs/html/index.html

1.2.7 Sorting imports

As a stylistic guide: Imports of code from other modules should always be alphabetically sorted with no empty lines between imports. The only exception to this rule is to keep one empty line between a group of `from x import y` and a group of `import y` imports.

1.3 API

CHAPTER

TWO

CHANGELOG

2.1 0.2 (2011-07-20)

- Renamed README.txt to README.rst and added README as a soft-link. [zupo]

2.2 0.1 (2011-07-19)

- Initial release. [zupo]

CHAPTER

THREE

ROADMAP

3.1 Current status

At this point, collective.table offers only basic functionalities and is somewhat usable. It's in alpha stage so storage API *will* change and there will be *no migration path*. Do not use for storing important data just yet!

3.2 Milestone *Sauna Sprint*

Both Nejc Zupan (GSoC student) and Martijn Pieters (mentor) will be attending the Plone Sauna Sprint 2011. During that week we will focus on the following:

- improve ZODB storage (sorting, filtering, scalability and such)
- improve UI
- add more tests (test coverage is already above 85%, btw)
- unit-tests for Javascript

3.3 Milestone *End of GSoC*

- battle-hardening what we produce during the Sauna Sprint
- selenium tests, ran regularly with help of jenkins.plone.org
- improve user and developer documentation
- export data to csv, etc
- screencast usage demo

3.4 Milestone *future*

- column types (numeric, date, boolean, vocabularies, etc.)
- add SQLAlchemy storage, based on SQLite by default
- import from csv
- insertion of rows in the middle, together with concurrent editing

CHAPTER

FOUR

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*