niteoweb.click2sell Documentation

Release 0.4.4

June 03, 2012

CONTENTS

A Plone add-on that integrates Click2Sell digital products retailer system with Plone to enable paid memberships on your site.

- Source code @ GitHub
- Releases @ PyPI
- Documentation @ ReadTheDocs
- Continuous Integration @ Travis-CI

CHAPTER

ONE

HOW IT WORKS

1. Visitor comes to `yoursite.com/order` (or similar) and clicks *Order* link.
2. Visitor is sent to Click2Sell's order form (on `http://click2sell.eu`), where he enters his personal information and performs payment.
3. Click2sell calls-back a special view on your plone site (`/@@click2sell`), which reads POST data from Click2Sell, verifies it against your `Secret Key` and creates a new member.
4. The following information is stored in member data for later use:

   `product_id` Click2Sell's *Product ID* of the purchased item.

   `product_name` Click2Sell's *Product Name* of the purchased item.

   `affiliate` Affiliate who referred the buyer.

   `last_purchase_id` Click2Sell's *Receipt ID* of the last purchase. This field gets updated on every recurring payment.

   `last_purchase_timestamp` Exact timestamp of the last purchase. This field gets updated on every recurring payment.
5. Upon creating a new member, Plone sends an email with login password.
6. An `IMemberCreateEvent` is emitted upon creating a new member.
7. The new member can now login and use the site.
8. It is possible to create a `product_id` to `group_name` mapping in Plone Control Panel. This means that if a member purchased a product which is listed in this mapping, the member will also be added to a group mapped to this product.

Note: If a member already exists in Plone, then the `@@click2sell` view simply updates `last_purchase_id` and `last_purchase_timestamp` member fields. The member will also be added to the new product group, but also kept in the old.

CHAPTER

TWO

DEMO

You can see this product in action at BigContentSearch.

CHAPTER

THREE

INSTALLATION

To install `niteoweb.click2sell` you simply add `niteoweb.click2sell` to the list of eggs in your buildout, run buildout and restart Plone. Then, install *niteoweb.click2sell* using the Add-ons control panel.

CHAPTER

FOUR

CONFIGURATION

4.1 Click2Sell

Go to Click2Sell.eu and use `For Merchant` -> `Add Product` to add a new *Product*.

Then click on your new Product and select `API Settings`. For *URL to notify* set `http://yoursite.com/@@click2sell` and also choose a *Secret Key*. Check `Enable/Disable remote server's notification` to enable server notifications and consequently member auto-registering.

4.2 Plone

Go to `Site Setup` -> `click2sell` control panel form and configure the `Secret Key` field by pasting in the *Secret Key* you defined above.

You can also configure the `product_id` to `group_name` mapping. This comes in effect when member purchases a product which is listed in this mapping, the member will also be added to a group mapped to this product.

For example, imagine you have the following in your mapping:

```
1|basic-members
2|premium-members
```

Members purchasing the product with id `1` will be added to the `basic-members` group, whose purchasing `2` will be added to the `premium-members` group. For others, nothing will be done.

When switching products, an updated member will be added to new product group, but also kept in the old group. No information is removed/deleted.

CHAPTER

FIVE

TEST IT

You are now ready to do a test buy! Go back to `My Products` and click `Test BUY`. Before you finish the transaction, you need to set up your Plone site to receive Click2Sell server notifications.

Confirm by logging-in to Click2Sell and checking to see if there were any purchases. Also check if your receive an email with username and password for accessing your site and try to login with them.

CHAPTER

SIX

KNOWN ISSUES

The following known issues exist:

- If members stop paying for monthly or yearly subscriptions, you have to manually delete them from your Plone site.
- The same as above goes for any chargebacks or refunds. You have to manage them manually.

CHAPTER

SEVEN

DEVELOPER DOCUMENTATION

7.1 Tips & Tricks

7.1.1 Mocked request

If you want to mock a request from Click2Sell in your local development environment, run something along these lines:

```
$ curl -d "buyer_email=test@niteoweb.com&buyer_name=John&buyer_surname=Smith&product_id=1&product_nar
```

7.2 Conventions

7.2.1 Line length

All Python code in this package should be PEP8 valid. However, we don't enforce the 80-char line length rule. It is encouraged to have your code formatted in 80-char lines, but somewhere it's just more readable to break this rule for a few characters. Long and descriptive test method names are a good example of this.

Note: Configuring your editor to display a line at 80th column helps a lot here and saves time.

Note: The line length rules also applies to non-python source files, such as documentation .rst files.

7.2.2 About imports

1. Don't use * to import *everything* from a module.
2. Don't use commas to import multiple stuff on a single line.
3. Don't use relative paths.

```
from collective.table.local import add_row
from collective.table.local import delete_rows
from collective.table.local import update_cell
```

instead of

```
from collective.table.local import *
from collective.table.local import add_row, delete_rows
from .local import update_cell
```

7.2.3 Sort imports

As another imports 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.

```
from collective.table.tests.base import TableIntegrationTestCase
from plone.app.testing import login

import os
```

instead of

```
import os

from plone.app.testing import login
from collective.table.tests.base import TableIntegrationTestCase
```

7.2.4 Commit checklist

Before every commit you should:

- Run *Unit tests*.
- Run *Syntax validation*.
- Add an entry to *Changelog* (if applicable).
- Add/modify *Sphinx documentation* (if applicable).

Note: All syntax checks and all tests can be run with a single command:

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

7.2.5 Unit tests

Un-tested code is broken code.

For every feature you add to the codebase you must also add tests for it. Also write a test for every bug you fix to ensure it doesn't crop up again in the future.

You run tests like this:

```
$ bin/test -s niteoweb.click2sell
```

7.2.6 Syntax validation

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

To validate your source code, run the following two commands:

```
$ bin/pyflakes src/niteoweb/click2sell
$ bin/pep8 --ignore=E501 src/niteoweb/click2sell
$ for pt in `find src/niteoweb/click2sell/ -name "*.pt"` ; do bin/zptlint $pt; done
```

Note: It pays off to invest a little time to make your editor run *pep8* and *pyflakes* on a file every time you save that file. Saves lots of time in the long run.

7.2.7 Changelog

Feature-level changes to code are tracked inside `docs/HISTORY.txt`. Examples:

- added feature X
- removed Y
- fixed bug Z

Add an entry every time you add/remove a feature, fix a bug, etc.

7.2.8 Sphinx documentation

Un-documented code is broken code.

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

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

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

Documentation is automatically generated from these source files every time you push your code to GitHub. The post-commit hook is handled by ReadTheDocs and the results are visible at http://readthedocs.org/docs/niteowebclick2sel.

7.3 Releasing a new version

Releasing a new version of *niteoweb.click2sell* involves the following steps:

1. Create a git tag for the release.
2. Push the git tag upstream to GitHub.
3. Generate a distribution file for the package.
4. Upload the generated package to Python Package Index (PyPI).

7.3.1 Checklist

Before every release make sure that:

1. You have documented your changes in the `HISTORY.rst` file.
2. You have modified the version identifier in the `version.txt` to reflect the new release.

3. You have confirmed that the package description (generated from `README.rst` and others) renders correctly by running `bin/longtest`.
4. You have committed all changes to the git repository and pushed them upstream.
5. You have the working directory checked out at the revision you wish to release.

7.3.2 Actions

For help with releasing we use `jarn.mkreleaser`. It's listed as a dependency in `setup.py` and should already be installed in your local bin:

```
$ bin/mkrelease -d pypi -pq ./
```

Note: In order to push packages to PyPI you need to have the appropriate access rights to the package on PyPI and you need to configure your PyPI credentials in the `~/.pypirc` file, e.g.:

```
[distutils]
index-servers =
  pypi

[pypi]
username = fred
password = secret
```

7.3.3 Example

In the following example we are releasing version 0.1 of *niteoweb.click2sell*. The package has been prepared so that `version.txt` contains the version `0.1`, this change has been committed to git and all changes have been pushed upstream to GitHub:

```
# Check that package description is rendered correctly
$ bin/longtest

# Make a release and upload it to PyPI
$ bin/mkrelease -d pypi -pq ./
Releasing niteoweb.click2sell 0.1
Tagging niteoweb.click2sell 0.1
To git@github.com:niteoweb/niteoweb.click2sell.git
* [new tag]         0.1 -> 0.1
running egg_info
running sdist
warning: sdist: standard file not found: should have one of README, README.txt
running register
Server response (200): OK
running upload
warning: sdist: standard file not found: should have one of README, README.txt
Server response (200): OK
done
```

Note: Please ignore the sdist warning about README file above. PyPI does not depend on it and it's just a bug in setupools (reported and waiting to be fixed).

CHAPTER

EIGHT

CHANGELOG

8.1 0.4.4 (2012-06-03)

- Previous release was swallowing errors. Re-thinked how handling and displaying errors should be done and rewrote most of it. Also added more tests to assert error messages in response bodies and in the Zope log. [zupo]

8.2 0.4.3 (2012-06-01)

- More verbose POST error handling. [zupo]
- Revert making click2sell key non-required so we can edit the mapping without always supplying the key. [zupo]

8.3 0.4.2 (2012-04-29)

- Made `click2sell key` non-required so we can modify the product_id to group_name mapping without always supplying the `click2sell key`. [zupo]
- Even more work on groups edge-cases. [zupo]

8.4 0.4.1 (2012-04-22)

- More tests for groups edge-cases. [zupo]
- Updating package with latest best practices. [zupo]

8.5 0.4 (2012-04-21)

- Site admins can now map C2S `product_id` to groups. This causes new members to be added to the group their `product_id` maps to. [zupo]
- Store configuration in *plone.app.registry* rather than in a local utility. [zupo]

8.6 0.3 (2012-01-28)

- Updated the package with latest best practices, added support for *plone.app.testing*, moved to GitHub. [zupo]

8.7 0.2.2 (2011-08-16)

- Support for Plone 4.1. [zupo]

8.8 0.2.1 (2010-10-06)

- Fixed updating an already existing member. [zupo]
- Added Uninstall profile. [zupo]

8.9 0.2 (2010-10-06)

- Polishing, adding tests. [zupo]

8.10 0.1 (2010-09-30)

- Initial release. [zupo]

CHAPTER

NINE

LICENSE (3-CLAUSE BSD)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of NiteoWeb Ltd. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

CHAPTER

TEN

INDICES AND TABLES

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