Mixpanel-Celery Documentation

Release 0.8.0.

Wes Winham

September 30, 2015

Contents

Contents:

CHAPTER 1

mixpanel-celery - Asynchronous event tracking for Mixpanel

Version 0.8.0.

1.1 Introduction

mixpanel-celery helps you use Celery to asynchronously track your Mixpanel events. You want to perform your tracking asynchronously, because waiting for HTTP requests to Mixpanel to complete every time you want to record something important isn't ideal when you've already worked so hard to tune your performance.

mixpanel-celery works great with Django, but because Celery works with just python, so does mixpanel-celery.

1.2 Installation

The easiest way to install the current development version of mixpanel-celery is via pip

1.2.1 Installing The Stable Version

```
$ pip install mixpanel-celery
```

1.3 Running The Test Suite

We use Tox to test across all of our supported environments.

```
$ pip install tox
$ tox
```

If you'd just like to test for the version of python and Celery that you use, install the appropriate requirements listed in the `requirements` folder, and then run your tests. eg.

```
$ pip install -r requirements/test_celery_default.txt
$ pip install -r requirements/test_django_default.txt
$ python setup.py test
```

Right now, the test suite requires Django, but we'd love a pull request to remove that requirement.

It is also possible to run specific tests using `nosetests` directly.

1.4 Configuration

For easy test usage with Django, set your Mixpanel api token in your project's `settings.py` file with the `MIXPANEL_API_TOKEN` variable. Then set:

```
CELERY_ALWAYS_EAGER = True
```

So that all of your Celery tasks will run in-line for now.

Note: Obviously you'll want to actually configure Celery using one of the many available backends for actual production use. Celery has great documentation on that.

1.4.1 With Celery 3.1 and above

If you're not using `django-celery`, you must add the `mixpanel.tasks` module to your `include`. Otherwise, Celery won't know about the `mixpanel-celery` tasks.

Your configuration should look something like:

```
celery = Celery(
    'myproject',
    broker=settings.REDIS_URL,
    include=['myproject.tasks', 'mixpanel.tasks'],
)
```

1.4.2 With `django-celery` and Celery 3.0 or lower

If you're using an older version of Celery along with the now-deprecated combination of `django-celery` and a call to `djcelery.setup_loader()`, just add `mixpanel` to your list of `INSTALLED_APPS`.

1.5 Usage

Basic python example tracking an event called `my_event`

```
from mixpanel.tasks import EventTracker

result = EventTracker.delay(
    'my_event',
    {'distinct_id': 1},
    token='YOUR_API_TOKEN',
)
result.wait()
```

Example usage in a Django view

```
from mixpanel.tasks import EventTracker
from django.shortcuts import render

def test_view(request, template='test/test_view.html'):
    """
    Show user a test page.
    """
    # We should record that the user hit this page
```

```
    EventTracker.delay('hit_test_view', {'distinct_id': request.user.pk})

    return render(template)
```

To pass the API key to your templates where you probably use the Mixpanel Javascript API, add the context_processor to your settings file

```
TEMPLATE_CONTEXT_PROCESSORS = (
    # ...
    'mixpanel.context_processors.api_key',
    # ...
)
```

Now in your templates you can access the API key like this

```
mixpanel.init("{{ MIXPANEL_API_TOKEN }}");
```

1.5.1 People Tracker Usage

mixpanel-celery also supports the People Tracker API which allows you store user profiles in Mixpanel's People Analytics product. The API for this is based on the Mixpanel JavaScript People API. Three calls are supported at this time: `set`, `add`, and `track_charge`. The `add` command is the `mixpanel.people.increment` in the JavaScript API.

To set profile property values using the `set` event:

```
from mixpanel.tasks import PeopleTracker

result = PeopleTracker.delay(
    'set',
    {
        'distinct_id': 1,
        'Plan': 'Premium',
        # you can set many properties in one call
        'discount end': '2013-01-01'

    },
    token='YOUR_API_TOKEN',
)
result.wait()
```

The above would set the `Plan` property to `Premium` for the profile with the mixpanel distinct id of 1. To increment profile property values using the `add` event:

```
from mixpanel.tasks import PeopleTracker

result = PeopleTracker.delay(
    'add',
    {
        'distinct_id': 1,
        # differs for JS API. You must provide
        # an increment value. There is no default
        'games played': 1,
        'points earned: 500,
        # subtract by providing a negative value
        'credits remaining': -34
    },
    token='YOUR_API_TOKEN',
```

```
)
result.wait()
```

Since some tasks are done separate from user interaction when updating their associated Person in mixpanel, you can set the `$ignore_time` special property by setting `'ignore_time'` to `True` in the `properties` dictionary:

```
from mixpanel.tasks import PeopleTracker

result = PeopleTracker.delay(
    'set',
    {
        'distinct_id': 1,
        'Plan': 'Premium',
        'ignore_time': True,
    },
    token='YOUR_API_TOKEN',
)
result.wait()
```

This bypasses the automatic re-setting of the "Last Seen" date property on the Person as described in Mixpanel's People HTTP Specification.

You can also track charges using the `track_charge` event:

```
from mixpanel.tasks import PeopleTracker

result = PeopleTracker.delay(
    'track_charge',
    {
        'distinct_id': 1,
        # this value is required
        'amount': 100,
        # optionally can have other properties
        'order_id': 6543
    },
    token='YOUR_API_TOKEN',
)
result.wait()

result = PeopleTracker.delay(
    'track_charge',
    {
        'distinct_id': 1,
        # use negative value for refund
        'amount': -50,
    },
    token='YOUR_API_TOKEN',
)
result.wait()
```

The `track_charge` event differs from the JS API in that you can't override the time of the transaction.

1.6 Building the Documentation

mixpanel-celery uses sphinx for documentation. To build the HTML docs

```
$ pip install sphinx
$ pip install sphinxtogithub
$ cd /path/to/mixpanel-celery/docs
$ make html
```

1.7 Bug Tracker

If you have feedback about bugs, features or anything else, the github issue tracking is a great place to report them: http://github.com/winhamwr/mixpanel-celery/issues

1.8 License

This software is licensed under the `New BSD License`. See the `LICENSE` file in the top distribution directory for the full license text.

1.9 Versioning

This project uses Semantic Versioning.

CHAPTER 2

Configuring mixpanel-celery

Mixpanel-celery's configuration looks a lot like Celery's configuration. All of the configuration directives will go in the same file you're using to configure Celery.

If you're using celery in a Django project these settings should be defined in your projects `settings.py` file.

In a regular Python environment using the default loader you must create the `celeryconfig.py` module and make sure it is available on the Python path.

Full instructions on settings up your Celery configuration are located at the Celery Configuration Docs page.

2.1 Configuration Options

- **MIXPANEL_API_TOKEN** As the name suggests, this is the Mixpanel api token for your Mixpanel account. All events that you track/register will take place on this account.

CHAPTER 3

Contributing to mixpanel-celery

Contribution of all types is very welcome via github pull requests. Have something you think might possibly be useful to other folks? Please send a pull request!

We'll definitely add you to the `AUTHORS` file, unless you request otherwise, so if you'd like to use contact info other than your github profile, let us know in your pull request.

3.1 Contribution Tips

Some general tips for making your pull request easy to accept:

3.1.1 1. Detailed Description

Provide a detailed description in your pull request so we know exactly what you're trying to accomplish. That way, if it's 90% there, it will be obvious how we can get it that extra 10%.

3.1.2 2. Follow PEP8

Code should be pep8 compatible. We recommend using flake8.

3.1.3 3. Include Tests

Bug fixes and features should come with additional tests and all existing tests should be passing. If you're not sure how to test your change, feel free to submit it anyway with a comment indicating so.

3.1.4 4. Documentation

Any new features should include documentation for those features.

CHAPTER 4

Module API Reference

Release 0.8.0

Date September 30, 2015

4.1 Empty Django Models: mixpanel - mixpanel.models

4.2 Event Tracker Tasks: mixpanel - mixpanel.tasks

4.3 Mixpanel Settings: mixpanel - mixpanel.conf

4.4 Configuration: mixpanel - mixpanel.conf.settings

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

m

M