django-admin-charts

Release 0.24.1

Dec 28, 2021

Contents

1 Introduction **3**

2 Requirements **7**

3 Installation **9**
- 3.1 Basic setup for `django-admin` . . . 9

4 Special configurations **11**
- 4.1 Update from `django-admin-tools-stats` . . . 11
- 4.2 Installation of javascript libraries with `django-bower` . . . 11
- 4.3 Usage with `django-admin-tools` . . . 12
- 4.4 Usage on DB that doesn't support JSONFields . . . 13

5 Running demo **15**

6 Development **17**
- 6.1 Dependencies . . . 17
- 6.2 Contributing . . . 17
- 6.3 Debugging charts . . . 17
- 6.4 Documentation . . . 17
- 6.5 License . . . 18

7 Developer Documentation **19**
- 7.1 Prerequisites . . . 19
- 7.2 Objects Description . . . 19
- 7.3 Django-admin-tools-stats Modules . . . 20
- 7.4 Test Case Descriptions . . . 20

8 Indices and tables **21**

Release 0.24.1

Date Dec 28, 2021

Keywords django, python, plot, graph, nvd3, d3, dashboard

Author Arezqui Belaid, Petr Dlouhý

Description Django-admin-tools-stats is a Django admin module that allow you to create easily charts on your dashboard based on specific models and criterias.

Contents:

CHAPTER 1

Introduction

Version 0.24.1

Date Dec 28, 2021

Keywords django, python, plot, graph, nvd3, d3, dashboard

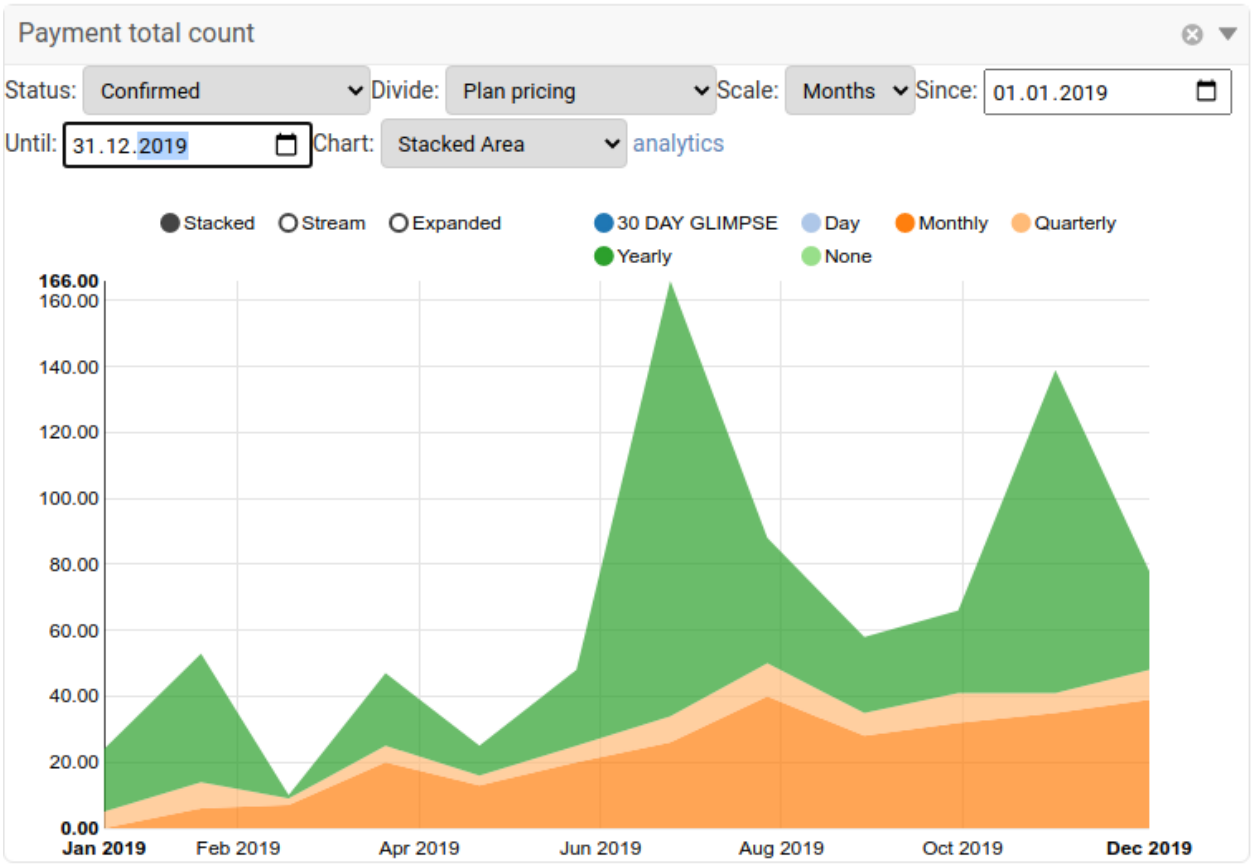
Author Arezqui Belaid, Petr Dlouhý

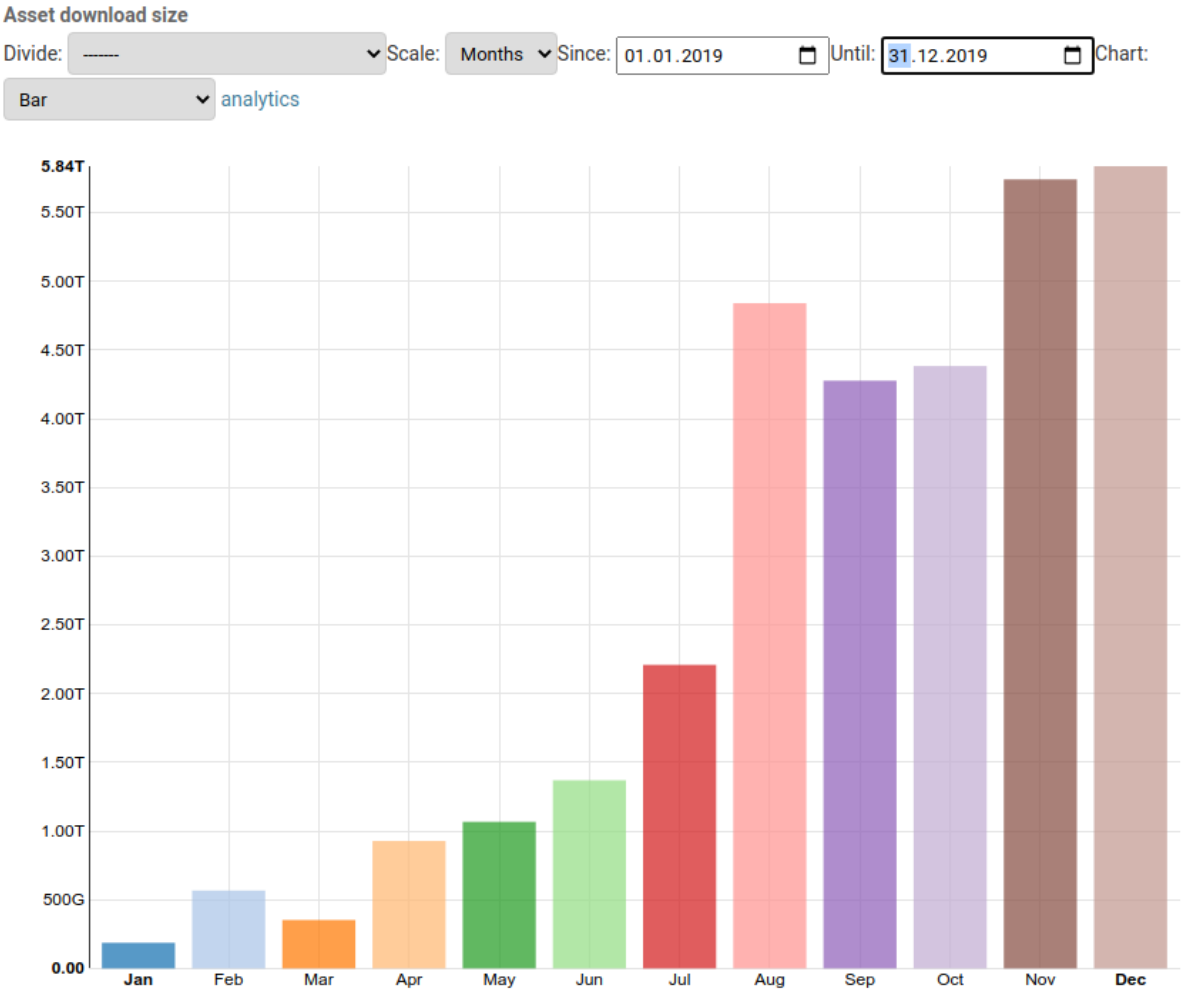
License MIT

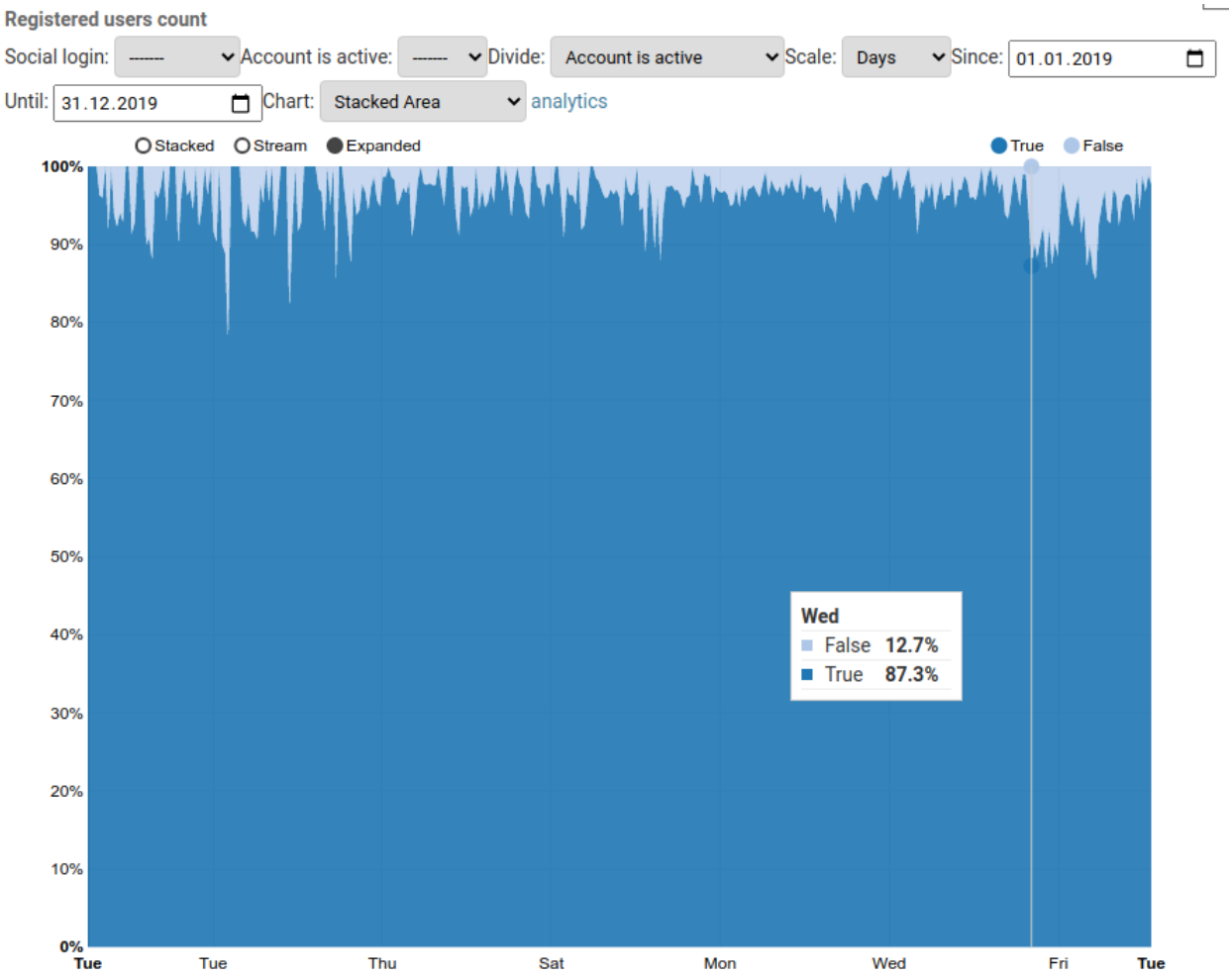
–

Create beautiful configurable charts from your models and display them on the `django-admin` index page or on `django-admin-tools` dashboard. The charts are based on models and criterias defined through admin interface and some chart parameters are configurable in live view.

This is application is fork of django-admin-tools-stats which has been reworked to display all charts through Ajax and made work with plain `django-admin`. The `django-admin-tools` are supported but not needed.


Payment total count
Status: Confirmed
Divide: Plan pricing
Scale: Months
Since: 01.01.2019
Until: 31.12.2019
Chart: Stacked Area
analytics
Stacked
Stream
Expanded
30 DAY GLIMPSE
Day
Monthly
Quarterly
Yearly
None
166.00
160.00
140.00
120.00
100.00
80.00
60.00
40.00
20.00
0.00
Jan 2019
Feb 2019
Apr 2019
Jun 2019
Aug 2019
Oct 2019
Dec 2019


Asset download size
Divide: ------
Scale: Months
Since: 01.01.2019
Until: 31.12.2019
Chart:
Bar
analytics
5.84T
5.50T
5.00T
4.50T
4.00T
3.50T
3.00T
2.50T
2.00T
1.50T
1.00T
500G
0.00
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Registered users count
Social login: ------
Account is active: ------
Divide: Account is active
Scale: Days
Since: 01.01.2019
Until: 31.12.2019
Chart: Stacked Area
analytics
Stacked
Stream
Expanded
True
False
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Wed
False 12.7%
True 87.3%
Tue
Tue
Thu
Sat
Mon
Wed
Fri
Tue

CHAPTER 2

Requirements

- `Django>=2.0`
- `Python>3.6`
- PostgreSQL (MySQL is experimental, other databases probably not working but PRs are welcome)
- `simplejson` for charts based on `DecimalField` values

CHAPTER 3

Installation

Install django-admin-charts with these commands:

```
$ pip install django-admin-charts
```

3.1 Basic setup for `django-admin`

Add `admin_tools_stats` (the Django admin charts application) & `django_nvd3` into INSTALLED_APPS in settings.py:

```
INSTALLED_APPS = (
    'admin_tools_stats',  # this must be BEFORE 'admin_tools' and 'django.contrib.
↪admin'
    'django_nvd3',
    ...
    'django.contrib.admin',
)
```

Install the `nvd3==1.7.1` and `d3==3.3.13` javascript libraries. For installation with `django-bower` see section *Installation of javascript libraries with django-bower*. Set library paths if they differ from the `django-bower` defaults:

```
ADMIN_CHARTS_NVD3_JS_PATH = 'bow/nvd3/build/nv.d3.js'
ADMIN_CHARTS_NVD3_CSS_PATH = 'bow/nvd3/build/nv.d3.css'
ADMIN_CHARTS_D3_JS_PATH = 'bow/d3/d3.js'
```

Register chart views in your `urls.py`:

```
from django.urls import include, path
urlpatterns = [
    path('admin_tools_stats/', include('admin_tools_stats.urls')),
]
```

Ensure, you have `default` cache set up: https://docs.djangoproject.com/en/3.2/topics/cache/#memcached

Run migrations:

```
$ python manage.py migrate
```

Open admin panel, configure `Dashboard Stats Criteria` & `Dashboard Stats` respectively

CHAPTER 4

Special configurations

4.1 Update from `django-admin-tools-stats`

Uninstall `django-admin-tools-stats`.

Follow `django-admin-charts` installation according to previous section. Especially pay attention to these steps:
- Move `admin_tools_stats` in `INSTALLED_APPS` before `admin_tools` and `django.contrib.admin`.
- Configure `urls.py`.

Change `DashboardCharts` to `DashboardChart` in dashboard definition (this is recomended even if dummy class is left for compatibility reasons).

Check any overridden template from `admin_tools_stats` or `DashboardChart(s)` class that might interfere with the changes.

4.2 Installation of javascript libraries with `django-bower`

Add `django-bower` to INSTALLED_APPS in settings.py:

```
INSTALLED_APPS = (
    ...
    'djangobower'
)
```

Add the following properties to you settings.py file:

```
# Specifie path to components root (you need to use absolute path)
BOWER_COMPONENTS_ROOT = os.path.join(PROJECT_ROOT, 'components')


BOWER_INSTALLED_APPS = (
    'd3#3.3.13',
```

(continues on next page)

(continued from previous page)

```
    'nvd3#1.7.1',
)
```

Add django-bower finder to your static file finders:

```
STATICFILES_FINDERS = (
    ...
    'djangobower.finders.BowerFinder',
)
```

Run the following commands. These will download nvd3.js and its dependencies using bower and throw them in to you static folder for access by your application:

```
$ python manage.py bower_install
$ python manage.py collectstatic
```

4.3 Usage with `django-admin-tools`

Configure `admin_tools`

Add following code to dashboard.py:

```
from admin_tools_stats.modules import DashboardChart, get_active_graph

# append an app list module
self.children.append(modules.AppList(
    _('Dashboard Stats Settings'),
    models=('admin_tools_stats.*', ),
))

# Copy following code into your custom dashboard
# append following code after recent actions module or
# a link list module for "quick links"
if context['request'].user.has_perm('admin_tools_stats.view_dashboardstats'):
        graph_list = get_active_graph()
    else:
        graph_list = []

for i in graph_list:
    kwargs = {}
    kwargs['require_chart_jscss'] = True
    kwargs['graph_key'] = i.graph_key

    for key in context['request'].POST:
        if key.startswith('select_box_'):
            kwargs[key] = context['request'].POST[key]

    self.children.append(DashboardChart(**kwargs))
```

You may also need to add some includes to your template admin base, see an example on the demo project:

```
demoproject/demoproject/templates/admin/base_site.html
```

4.4 Usage on DB that doesn't support JSONFields

You can add following line to your settings in order to use JSONField from *django-jsonfield* instead of native Django JSONField:

```
ADMIN_CHARTS_USE_JSONFIELD = False
```

This can become handy, when deploying on MySQL<5.7 (Like AWS RDS Aurora)

CHAPTER 5

Running demo

Run following commands:

```
pip install -r requirements
python manage.py migrate
python manage.py loaddata demoproject/fixtures/auth_user.json
python manage.py loaddata demoproject/fixtures/test_data.json
python manage.py bower install
python manage.py runserver
```

And log in with username *admin* and password *admin* to the *localhost:8000/admin* site.

CHAPTER 6

Development

6.1 Dependencies

django-admin-charts is a django based application, the major requirements are:

- django-jsonfield
- django-nvd3
- django-bower

6.2 Contributing

If you've found a bug, add a feature or improve django-admin-charts and think it is useful then please consider contributing. Patches, pull requests or just suggestions are always welcome!

Source code: http://github.com/PetrDlouhy/django-admin-charts

Bug tracker: https://github.com/PetrDlouhy/django-admin-charts/issues

6.3 Debugging charts

For chart data view (/admin_tools_stats/chart_data/payments/) the URL query parameter *&debug=True* can be added, in order to get Django debug page or Django debug toolbar.

6.4 Documentation

Documentation is available on 'Read the Docs': http://readthedocs.org/docs/django-admin-charts/

6.5 License

django-admin-charts is licensed under MIT, see `MIT-LICENSE.txt`.

CHAPTER 7

Developer Documentation

Contents:

7.1 Prerequisites

To fully understand this project, developers will need to have an advanced knowledge of:

- Django : http://www.djangoproject.com/
- Python : http://www.python.org/

7.2 Objects Description

7.2.1 `DashboardStatsCriteria`

To configure criteria for dashboard graphs

Attributes:

- `criteria_name` - Unique word .
- `criteria_fix_mapping` - JSON data key-value pairs.
- `dynamic_criteria_field_name` - Dynamic criteria field.
- `criteria_dynamic_mapping` - JSON data key-value pairs.
- `created_date` - record created date.
- `updated_date` - record updated date.

Name of DB table: dash_stats_criteria

7.2.2 DashboardStats

To configure graphs for dashboard

Attributes:

- graph_key - unique graph name.
- graph_title - graph title.
- model_app_name - App name of model.
- model_name - model name.
- date_field_name - Date field of model_name.
- criteria - many-to-many relationship.
- is_visible - enable/disable.
- created_date - record created date.
- updated_date - record updated date.

Name of DB table: dashboard_stats

7.3 Django-admin-tools-stats Modules

7.3.1 DashboardChart

Dashboard module with user registration charts.

7.3.2 DashboardCharts

Right now only dummy module with deprecation warning. We don't use module Groups any more.

7.4 Test Case Descriptions

7.4.1 How to run tests

1. Run full test suite:

```
$ python manage.py test --verbosity=2
```

2. Run AdminToolsStatsAdminInterfaceTestCase:

```
$ python manage.py test admin_tools_stats.AdminToolsStatsAdminInterfaceTestCase --
↪verbosity=2
```

7.4.2 Test Case

AdminToolsStatsAdminInterfaceTestCase

Test cases for django-admin-tools-stats Admin Interface.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search