

---

# **Django Powerbank Documentation**

***Release 0.4.2***

**Janusz Skonieczny**

**Jun 05, 2020**



---

## Contents

---

<b>1</b>	<b>Django Powerbank</b>	<b>3</b>
1.1	Work in progress . . . . .	3
1.2	Features . . . . .	3
1.3	Demo . . . . .	4
1.4	Quickstart . . . . .	4
1.5	Running Tests . . . . .	4
1.6	Credits . . . . .	4
<b>2</b>	<b>Installation</b>	<b>5</b>
2.1	Stable release . . . . .	5
2.2	From sources . . . . .	5
<b>3</b>	<b>Usage</b>	<b>7</b>
<b>4</b>	<b>Api docs</b>	<b>9</b>
4.1	django_powerbank package . . . . .	9
<b>5</b>	<b>Contributing</b>	<b>19</b>
5.1	Types of Contributions . . . . .	19
5.2	Get Started! . . . . .	20
5.3	Pull Request Guidelines . . . . .	21
5.4	Tips . . . . .	21
<b>6</b>	<b>Credits</b>	<b>23</b>
6.1	Development Lead . . . . .	23
6.2	Contributors . . . . .	23
<b>7</b>	<b>History</b>	<b>25</b>
7.1	0.2.1 (2017-07-14) . . . . .	25
<b>8</b>	<b>Indices and tables</b>	<b>27</b>
	<b>Python Module Index</b>	<b>29</b>
	<b>Index</b>	<b>31</b>



Contents:



# CHAPTER 1

---

## Django Powerbank

---

Extra power for included batteries

- Free software: MIT license
- Documentation: <https://django-powerbank.readthedocs.io>.

### 1.1 Work in progress

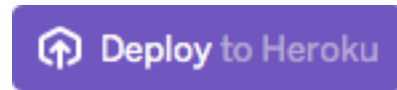
This package was created with [wooyek/cookiecutter-django-app](#) project template. It may fool you by having a lot of cookie filling just ready from the start. Be warned! It may not be ready for eating!

### 1.2 Features

- Pending :D

## 1.3 Demo

To run an example project for this django reusable app, click the button below and start a demo server on Heroku



## 1.4 Quickstart

Install Django Powerbank:

```
pip install django-powerbank
```

Add it to your *INSTALLED\_APPS*:

```
INSTALLED_APPS = (
    ...
    'django_powerbank.apps.DjangoPowerbankConfig',
    ...
)
```

Add Django Powerbank's URL patterns:

```
from django_powerbank import urls as django_powerbank_urls

urlpatterns = [
    ...
    url(r'^$', include(django_powerbank_urls)),
    ...
]
```

## 1.5 Running Tests

Does the code actually work?

```
$ pipenv install --dev
$ pipenv shell
$ tox
```

We recommend using [pipenv](#) but a legacy approach to creating virtualenv and installing requirements should also work. Please install *requirements/development.txt* to setup virtual env for testing and development.

## 1.6 Credits

This package was created with [Cookiecutter](#) and the [wooyek/cookiecutter-django-app](#) project template.



### 2.1 Stable release

To install Django Powerbank, run this command in your terminal:

```
$ pip install django-powerbank
```

This is the preferred method to install Django Powerbank, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

### 2.2 From sources

You can either clone the public repository:

```
$ git clone git://github.com/wooyek/django-powerbank
```

Or download the download source from [project website](#). Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```



## CHAPTER 3

---

### Usage

---

To use Django Powerbank in a project, add it to your *INSTALLED\_APPS*:

```
INSTALLED_APPS = (  
    ...  
    'django_powerbank.apps.DjangoPowerbankConfig',  
    ...  
)
```

Add Django Powerbank's URL patterns:

```
from django_powerbank import urls as django_powerbank_urls  
  
urlpatterns = [  
    ...  
    url(r'^$', include(django_powerbank_urls)),  
    ...  
)
```



## 4.1 django\_powerbank package

### 4.1.1 Subpackages

django\_powerbank.core package

Submodules

django\_powerbank.core.validators module

```
class django_powerbank.core.validators.MaxCurrentYearValidator
    Bases: object

    deconstruct ()
        Return a 3-tuple of class import path, positional arguments, and keyword arguments.

    message = 'Ensure this value is less than or equal to %(limit_value)s.'
```

```
class django_powerbank.core.validators.MaxTodayDateValidator
    Bases: object

    deconstruct ()
        Return a 3-tuple of class import path, positional arguments, and keyword arguments.

    message = 'Ensure this value is less than or equal to %(limit_value)s.'
```

```
class django_powerbank.core.validators.MsisdnValidator (code=None,                in-
                                                         verse_match=None,
                                                         flags=None)
    Bases: django.core.validators.RegexValidator
```

```
class django_powerbank.core.validators.SmartUrlValidator
    Bases: object
```

**deconstruct** ()

Return a 3-tuple of class import path, positional arguments, and keyword arguments.

### Module contents

#### django\_powerbank.db package

#### Subpackages

#### django\_powerbank.db.models package

#### Subpackages

#### django\_powerbank.db.models.fields package

#### Submodules

#### django\_powerbank.db.models.fields.pl module

**class** django\_powerbank.db.models.fields.pl.**PlNipField** (\*args, \*\*kwargs)

Bases: django.db.models.fields.CharField

**formfield** (\*\*kwargs)

Return a django.forms.Field instance for this field.

**to\_python** (value)

Convert the input value into the expected Python data type, raising django.core.exceptions.ValidationError if the data can't be converted. Return the converted value. Subclasses should override this.

**class** django\_powerbank.db.models.fields.pl.**PlRegonField** (\*args, \*\*kwargs)

Bases: django.db.models.fields.CharField

**formfield** (\*\*kwargs)

Return a django.forms.Field instance for this field.

**to\_python** (value)

Convert the input value into the expected Python data type, raising django.core.exceptions.ValidationError if the data can't be converted. Return the converted value. Subclasses should override this.

### Module contents

**class** django\_powerbank.db.models.fields.**AutoSlugField** (source\_field=None,  
keep\_existing=False,  
source\_fallback=False,  
\*args, \*\*kwargs)

Bases: *django\_powerbank.db.models.fields.SourceFieldMixin*, django.db.models.fields.SlugField

**get\_slug\_value** (model\_instance)

**get\_source\_value** (model\_instance)

**pre\_save** (model\_instance, add)

Return field's value just before saving.

```

class django_powerbank.db.models.fields.BinaryMaskEnum
    Bases: django_powerbank.db.models.fields.ChoicesIntEnum

    An enumeration.

    get_display = <bound method BinaryMaskEnum.get_display of <enum 'BinaryMaskEnum'>>

class django_powerbank.db.models.fields.ChoicesIntEnum
    Bases: enum.IntEnum

    Extends IntEnum with django choices generation capability

    capitalize = <bound method ChoicesIntEnum.capitalize of <enum 'ChoicesIntEnum'>>

    choices = <bound method ChoicesIntEnum.choices of <enum 'ChoicesIntEnum'>>

    values = <bound method ChoicesIntEnum.values of <enum 'ChoicesIntEnum'>>

class django_powerbank.db.models.fields.JSONField(*args, **kwargs)
    Bases: django.db.models.fields.TextField

    Simple JSON field that stores python structures as JSON strings on database.

    from_db_value (value, expression, connection, context)

    get_prep_value (value)
        Convert value to JSON string before save

    to_python (value)
        Convert the input JSON value into python structures, raises django.core.exceptions.ValidationError if the
        data can't be converted.

    validate (value, model_instance)
        Check value is a valid JSON string, raise ValidationError on error.

    value_from_object (obj)
        Return the value of this field in the given model instance.

    value_to_string (obj)
        Return value from object converted to string properly

class django_powerbank.db.models.fields.MarkDownField(source_field=None, *args,
                                                         **kwargs)
    Bases: django_powerbank.db.models.fields.SourceFieldMixin, django.db.models.
    fields.TextField

    contribute_to_class (cls, name, private_only=False)
        Register the field with the model class it belongs to.

        If private_only is True, create a separate instance of this field for every subclass of cls, even if cls is not an
        abstract model.

    from_db_value (value, expression, connection)

    pre_save (model_instance, add)
        Return field's value just before saving.

    to_python (value)
        Convert the input value into the expected Python data type, raising django.core.exceptions.ValidationError
        if the data can't be converted. Return the converted value. Subclasses should override this.

class django_powerbank.db.models.fields.PhoneField(*args, **kwargs)
    Bases: django.db.models.fields.CharField

    default_validators = [<django_powerbank.core.validators.MsisdnValidator object>]

```

**formfield** (*\*\*kwargs*)

Return a `django.forms.Field` instance for this field.

**to\_python** (*value*)

Convert the input value into the expected Python data type, raising `django.core.exceptions.ValidationError` if the data can't be converted. Return the converted value. Subclasses should override this.

```
class django_powerbank.db.models.fields.SecretField(source_field=None, *args,  
                                                    **kwargs)
```

Bases: `django_powerbank.db.models.fields.SourceFieldMixin`, `django.db.models.fields.CharField`

**pre\_save** (*model\_instance, add*)

Return field's value just before saving.

```
class django_powerbank.db.models.fields.SourceFieldMixin(source_field=None,  
                                                         *args, **kwargs)
```

Bases: `object`

**check** (*\*\*kwargs*)

```
class django_powerbank.db.models.fields.UniqueSlugField(source_field=None,  
                                                         keep_existing=False,  
                                                         *args, **kwargs)
```

Bases: `django_powerbank.db.models.fields.AutoSlugField`, `django.db.models.fields.SlugField`

**get\_slug\_value** (*model\_instance*)

## Submodules

### django\_powerbank.db.models.base module

```
class django_powerbank.db.models.base.BaseModel (*args, **kwargs)
```

Bases: `django.db.models.base.Model`

**class Meta**

Bases: `object`

**abstract** = `False`

**get\_field\_name** (*field\_name*)

**get\_field\_names** ()

**populate** (*\*\*kwargs*)

Populate an instance from keyword arguments.

Each keyword argument will be used to set a corresponding field. Keywords must refer to valid property name. This is similar to passing keyword arguments to the `Model` constructor.

**to\_dict** (*include=None, exclude=None*)

Return a dict containing the entity's property values.

#### Parameters

- **include** – Optional set of property names to include, default all.
- **exclude** – Optional set of property names to skip, default none. A name contained in both include and exclude is excluded.



## django\_powerbank.db.models.query module

```
class django_powerbank.db.models.query.ApproxQuerySet (model=None, query=None,  
                                                    using=None, hints=None)  
  
    Bases: django.db.models.query.QuerySet  
  
    count ()  
        Perform a SELECT COUNT() and return the number of records as an integer.  
  
        If the QuerySet is already fully cached, return the length of the cached results set to avoid multiple SELECT  
        COUNT(*) calls.  
  
    classmethod wrap_query (qry)  
  
class django_powerbank.db.models.query.TableStatusQuerySet (model=None,  
                                                            query=None,  
                                                            using=None,  
                                                            hints=None)  
  
    Bases: django_powerbank.db.models.query.ApproxQuerySet
```

## Module contents

## Module contents

## django\_powerbank.forms package

## Submodules

## django\_powerbank.forms.fields module

```
class django_powerbank.forms.fields.DateRangeField (input_formats=None, *args,  
                                                    **kwargs)  
  
    Bases: django.forms.fields.DateField  
  
    to_python (value)  
        Validate that the input can be converted to a date. Return a Python datetime.date object.  
  
    widget  
        alias of django.forms.widgets.Input  
  
class django_powerbank.forms.fields.PhoneField (*, max_length=None,  
                                                    min_length=None, strip=True,  
                                                    empty_value="", **kwargs)  
  
    Bases: django.forms.fields.CharField  
  
    widget  
        alias of django_powerbank.forms.widgets.PhoneInput  
  
class django_powerbank.forms.fields.TypeaheadField (*, max_length=None,  
                                                    min_length=None, strip=True,  
                                                    empty_value="", **kwargs)  
  
    Bases: django.forms.fields.CharField  
  
    widget  
        alias of django_powerbank.forms.widgets.Typeahead
```

**django\_powerbank.forms.widgets module**

```
class django_powerbank.forms.widgets.PhoneInput (attrs=None)
```

```
    Bases: django.forms.widgets.TextInput
```

```
    input_type = 'tel'
```

```
    media
```

```
class django_powerbank.forms.widgets.SelectizeBase (attrs=None, url=None, allow_create=False, value_field='text', label_field='text', search_field='text', plugins=[], close_after_select=True)
```

```
    Bases: django.forms.widgets.Input
```

```
    build_attrs (base_attrs, extra_attrs=None, **kwargs)
```

```
        Build an attribute dictionary.
```

```
    get_context (name, value, attrs)
```

```
    media
```

```
class django_powerbank.forms.widgets.SelectizeSelect (attrs=None, url=None, allow_create=False, value_field='text', label_field='text', search_field='text', plugins=[], close_after_select=True)
```

```
    Bases: django_powerbank.forms.widgets.SelectizeBase, django.forms.widgets.Select
```

```
    A selectize.js field
```

It requires selectize.js and headjs to be available in the browser. See a template below to see why. You can provide your own template to use selectize.js in a different way.

```
    media
```

```
    optgroup (name, value, attrs=None)
```

```
        Return a list of optgroups for this widget.
```

```
    template_name = 'django_powerbank/forms/widgets/selectize/select.html'
```

```
class django_powerbank.forms.widgets.SelectizeTags (attrs=None, url=None, allow_create=False, value_field='text', label_field='text', search_field='text', plugins=[], close_after_select=True)
```

```
    Bases: django_powerbank.forms.widgets.SelectizeBase
```

```
    media
```

```
    template_name = 'django_powerbank/forms/widgets/selectize/tags.html'
```

```
class django_powerbank.forms.widgets.Typeahead (attrs=None, url=None)
```

```
    Bases: django.forms.widgets.Input
```

```
    build_attrs (base_attrs, extra_attrs=None, **kwargs)
```

```
        Build an attribute dictionary.
```

**media**

## Module contents

### django\_powerbank.testing package

#### Submodules

#### django\_powerbank.testing.base module

```
class django_powerbank.testing.base.AdminUserTestCase
    Bases: django_powerbank.testing.base.AssertionsMx

    setUp()

class django_powerbank.testing.base.AssertionsMx
    Bases: object

    assertNoFormErrors(response, form_context_key='form')

class django_powerbank.testing.base.MigrationsCheck(methodName='runTest')
    Bases: django_powerbank.testing.base.MigrationsCheckMx, unittest.case.
    TestCase

class django_powerbank.testing.base.MigrationsCheckMx
    Bases: object

    setUp()

    tearDown()

    test_missing_migrations()

class django_powerbank.testing.base.StaffUserTestCase
    Bases: django_powerbank.testing.base.AssertionsMx

    setUp()

class django_powerbank.testing.base.UserTestCase
    Bases: django_powerbank.testing.base.AssertionsMx

    setUp()
```

#### django\_powerbank.testing.factories module

```
class django_powerbank.testing.factories.UserFactory
    Bases: factory.django.DjangoModelFactory

    email = <factory.faker.Faker object>

    first_name = <factory.faker.Faker object>

    is_active = True

    is_staff = False

    last_name = <factory.faker.Faker object>

    username = <factory.declarations.Sequence object>
```

## django\_powerbank.testing.utils module

django\_powerbank.testing.utils.**model\_to\_request\_data\_dict** (*model*)

Removes fields with None value. Test client will serialize them into 'None' strings that will cause validation errors.

## Module contents

### django\_powerbank.views package

#### Submodules

### django\_powerbank.views.auth module

**class** django\_powerbank.views.auth.**AbstractAccessView** (*\*\*kwargs*)

Bases: *django\_powerbank.views.ExceptionResponseView*

Allows you to handle authorization before dispatch is called

**check\_authorization** (*\*args, \*\*kwargs*)

**dispatch** (*request, \*args, \*\*kwargs*)

**class** django\_powerbank.views.auth.**AbstractAuthorizedView** (*\*\*kwargs*)

Bases: *django\_powerbank.views.auth.AuthenticatedView*

**check\_authorization** (*\*args, \*\*kwargs*)

**forbidden\_message** = 'You are not authorized to view this page'

**get\_forbidden\_message** ()

**handle\_forbidden** ()

**is\_authorized** (*\*args, \*\*kwargs*)

django\_powerbank.views.auth.**AccessMixin**

alias of *django\_powerbank.views.auth.AuthenticatedView*

**class** django\_powerbank.views.auth.**AuthenticatedView** (*\*\*kwargs*)

Bases: *django\_powerbank.views.auth.AbstractAccessView*

redirects unauthenticated users to login

**check\_authorization** (*\*args, \*\*kwargs*)

**handle\_anonymous** (*\*args, \*\*kwargs*)

**is\_authenticated** (*\*args, \*\*kwargs*)

**class** django\_powerbank.views.auth.**StaffRequiredMixin** (*\*\*kwargs*)

Bases: *django\_powerbank.views.auth.AbstractAuthorizedView*

**is\_authorized** (*\*args, \*\*kwargs*)

### django\_powerbank.views.generic module

**class** django\_powerbank.views.generic.**FilterMixin**

Bases: *django.views.generic.list.MultipleObjectMixin*

**get\_queryset()**

Return the list of items for this view.

The return value must be an iterable and may be an instance of *QuerySet* in which case *QuerySet* specific behavior will be enabled.

## django\_powerbank.views.mixins module

**class** django\_powerbank.views.mixins.**ReturnUrlMx**(\*\*kwargs)

Bases: django.views.generic.base.ContextMixin, django.views.generic.base.View

**dispatch**(request, \*args, \*\*kwargs)

Does request processing for return\_url query parameter and redirects with it's missing

We can't do that in the get method, as it does not exist in the View base class and child mixins implementing get do not call super().get

**get\_context\_data**(\*\*kwargs)

**get\_success\_url**()

## Module contents

**exception** django\_powerbank.views.**ExceptionResponse**(response)

Bases: Exception

Generic exception for signalling that instead of default error handling we should use attached response

**class** django\_powerbank.views.**ExceptionResponseView**(\*\*kwargs)

Bases: django.views.generic.base.View

**dispatch**(request, \*args, \*\*kwargs)

**exception** django\_powerbank.views.**Http302**(to, \*args, \*\*kwargs)

Bases: *django\_powerbank.views.ExceptionResponse*

Wraps a redirect shortcut call into the ResponseException

**exception** django\_powerbank.views.**Http400**(response)

Bases: *django\_powerbank.views.ExceptionResponse*

**exception** django\_powerbank.views.**Http401**(response)

Bases: *django\_powerbank.views.ExceptionResponse*

**exception** django\_powerbank.views.**Http403**(response)

Bases: django.core.exceptions.PermissionDenied, *django\_powerbank.views.ExceptionResponse*

A convenience wrapper around :py:class:`~django.core.exceptions.PermissionDenied`

## 4.1.2 Submodules

### 4.1.3 django\_powerbank.app\_settings module

### 4.1.4 django\_powerbank.apps module

**class** django\_powerbank.apps.**DjangoPowerbankConfig**(app\_name, app\_module)

Bases: django.apps.config.AppConfig

```
name = 'django_powerbank'

ready()
    Override this method in subclasses to run code when Django starts.

verbose_name = 'Django Powerbank'

django_powerbank.apps.setup_app_settings()
```

#### 4.1.5 Module contents

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

## 5.1 Types of Contributions

### 5.1.1 Report Bugs

Report bugs at <https://github.com/wooyek/django-powerbank/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

### 5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

### 5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

### 5.1.4 Write Documentation

Django Powerbank could always use more documentation, whether as part of the official Django Powerbank docs, in docstrings, or even on the web in blog posts, articles, and such.

### 5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/wooyek/django-powerbank/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

## 5.2 Get Started!

Ready to contribute? Here's how to set up *django-powerbank* for local development.

1. Fork the *django-powerbank* repo on github.com
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-powerbank.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv django-powerbank
$ cd django-powerbank/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 django-powerbank tests
$ tox -e check
$ python setup.py test or py.test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.



## 5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python versions mentioned in tox.ini file. Check [https://travis-ci.org/wooyek/django-powerbank/pull\\_requests](https://travis-ci.org/wooyek/django-powerbank/pull_requests) and make sure that the tests pass for all supported Python versions.

## 5.4 Tips

To run a subset of tests:

```
$ pytest tests.test_models
```



### 6.1 Development Lead

- @wooyek: Janusz Skonieczny <[js+pypi@bravelabs.pl](mailto:js+pypi@bravelabs.pl)>

### 6.2 Contributors

None yet. Why not be the first?



#### 7.1 0.2.1 (2017-07-14)

- First release on PyPI.



## CHAPTER 8

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`





### a

`django_powerbank.app_settings`, 17  
`django_powerbank.apps`, 17

### c

`django_powerbank.core`, 10  
`django_powerbank.core.validators`, 9

### d

`django_powerbank`, 18  
`django_powerbank.db`, 13  
`django_powerbank.db.models`, 13  
`django_powerbank.db.models.base`, 12  
`django_powerbank.db.models.fields`, 10  
`django_powerbank.db.models.fields.pl`, 10  
`django_powerbank.db.models.query`, 13

### f

`django_powerbank.forms`, 15  
`django_powerbank.forms.fields`, 13  
`django_powerbank.forms.widgets`, 14

### t

`django_powerbank.testing`, 16  
`django_powerbank.testing.base`, 15  
`django_powerbank.testing.factories`, 15  
`django_powerbank.testing.utils`, 16

### v

`django_powerbank.views`, 17  
`django_powerbank.views.auth`, 16  
`django_powerbank.views.generic`, 16  
`django_powerbank.views.mixins`, 17



## A

[abstract](#) ([django\\_powerbank.db.models.base.BaseModel.Meta](#) attribute), 12  
[AbstractAccessView](#) (class in [django\\_powerbank.views.auth.AbstractAccessView](#) method), 16  
[AbstractAuthorizedView](#) (class in [django\\_powerbank.views.auth.AbstractAuthorizedView](#) method), 16  
[AccessMixin](#) (in module [django\\_powerbank.views.auth](#)), 16  
[AdminUserTestCase](#) (class in [django\\_powerbank.testing.base](#)), 15  
[ApproxQuerySet](#) (class in [django\\_powerbank.db.models.query](#)), 13  
[AssertionsMx](#) (class in [django\\_powerbank.testing.base](#)), 15  
[assertNoFormErrors\(\)](#) ([django\\_powerbank.testing.base.AssertionsMx](#) method), 15  
[AuthenticatedView](#) (class in [django\\_powerbank.views.auth](#)), 16  
[AutoSlugField](#) (class in [django\\_powerbank.db.models.fields](#)), 10

## B

[BaseModel](#) (class in [django\\_powerbank.db.models.base](#)), 12  
[BaseModel.Meta](#) (class in [django\\_powerbank.db.models.base](#)), 12  
[BinaryMaskEnum](#) (class in [django\\_powerbank.db.models.fields](#)), 10  
[build\\_attrs\(\)](#) ([django\\_powerbank.forms.widgets.SelectizeBase](#) method), 14  
[build\\_attrs\(\)](#) ([django\\_powerbank.forms.widgets.Typeahead](#) method), 14

## C

[capitalize](#) ([django\\_powerbank.db.models.fields.ChoicesIntEnum](#) attribute), 11

[check\(\)](#) ([django\\_powerbank.db.models.fields.SourceFieldMixin](#) method), 12  
[check\\_authorization\(\)](#) ([django\\_powerbank.views.auth.AbstractAccessView](#) method), 16  
[check\\_authorization\(\)](#) ([django\\_powerbank.views.auth.AbstractAuthorizedView](#) method), 16  
[check\\_authorization\(\)](#) ([django\\_powerbank.views.auth.AuthenticatedView](#) method), 16  
[choices](#) ([django\\_powerbank.db.models.fields.ChoicesIntEnum](#) attribute), 11  
[ChoicesIntEnum](#) (class in [django\\_powerbank.db.models.fields](#)), 11  
[contribute\\_to\\_class\(\)](#) ([django\\_powerbank.db.models.fields.MarkDownField](#) method), 11  
[count\(\)](#) ([django\\_powerbank.db.models.query.ApproxQuerySet](#) method), 13

## D

[DateRangeField](#) (class in [django\\_powerbank.forms.fields](#)), 13  
[deconstruct\(\)](#) ([django\\_powerbank.core.validators.MaxCurrentYearValidator](#) method), 9  
[deconstruct\(\)](#) ([django\\_powerbank.core.validators.MaxTodayDateValidator](#) method), 9  
[deconstruct\(\)](#) ([django\\_powerbank.core.validators.SmartUrlValidator](#) method), 9  
[default\\_validators](#) ([django\\_powerbank.db.models.fields.PhoneField](#) attribute), 11  
[dispatch\(\)](#) ([django\\_powerbank.views.auth.AbstractAccessView](#) method), 16  
[dispatch\(\)](#) ([django\\_powerbank.views.ExceptionResponseView](#) method), 17  
[dispatch\(\)](#) ([django\\_powerbank.views.mixins.ReturnUrlMx](#) method), 17  
[django\\_powerbank](#) (module), 18

[django\\_powerbank.app\\_settings \(module\)](#), 17  
[django\\_powerbank.apps \(module\)](#), 17  
[django\\_powerbank.core \(module\)](#), 10  
[django\\_powerbank.core.validators \(module\)](#), 9  
[django\\_powerbank.db \(module\)](#), 13  
[django\\_powerbank.db.models \(module\)](#), 13  
[django\\_powerbank.db.models.base \(module\)](#), 12  
[django\\_powerbank.db.models.fields \(module\)](#), 10  
[django\\_powerbank.db.models.fields.pl \(module\)](#), 10  
[django\\_powerbank.db.models.query \(module\)](#), 13  
[django\\_powerbank.forms \(module\)](#), 15  
[django\\_powerbank.forms.fields \(module\)](#), 13  
[django\\_powerbank.forms.widgets \(module\)](#), 14  
[django\\_powerbank.testing \(module\)](#), 16  
[django\\_powerbank.testing.base \(module\)](#), 15  
[django\\_powerbank.testing.factories \(module\)](#), 15  
[django\\_powerbank.testing.utils \(module\)](#), 16  
[django\\_powerbank.views \(module\)](#), 17  
[django\\_powerbank.views.auth \(module\)](#), 16  
[django\\_powerbank.views.generic \(module\)](#), 16  
[django\\_powerbank.views.mixins \(module\)](#), 17  
[DjangoPowerbankConfig \(class in django\\_powerbank.apps\)](#), 17

## E

[email \(django\\_powerbank.testing.factories.UserFactory attribute\)](#), 15  
[ExceptionResponse](#), 17  
[ExceptionResponseView \(class in django\\_powerbank.views\)](#), 17

## F

[FilterMixin \(class in django\\_powerbank.views.generic\)](#), 16  
[first\\_name \(django\\_powerbank.testing.factories.UserFactory attribute\)](#), 15  
[forbidden\\_message \(django\\_powerbank.views.auth.AbstractAuthorizedView attribute\)](#), 16  
[formfield\(\) \(django\\_powerbank.db.models.fields.PhoneField method\)](#), 11  
[formfield\(\) \(django\\_powerbank.db.models.fields.pl.PlNipField method\)](#), 10  
[formfield\(\) \(django\\_powerbank.db.models.fields.pl.PlRegonField method\)](#), 10

[from\\_db\\_value\(\) \(django\\_powerbank.db.models.fields.JSONField method\)](#), 11  
[from\\_db\\_value\(\) \(django\\_powerbank.db.models.fields.MarkDownField method\)](#), 11

## G

[get\\_context\(\) \(django\\_powerbank.forms.widgets.SelectizeBase method\)](#), 14  
[get\\_context\\_data\(\) \(django\\_powerbank.views.mixins.ReturnUrlMx method\)](#), 17  
[get\\_display \(django\\_powerbank.db.models.fields.BinaryMaskEnum attribute\)](#), 11  
[get\\_field\\_name\(\) \(django\\_powerbank.db.models.base.BaseModel method\)](#), 12  
[get\\_field\\_names\(\) \(django\\_powerbank.db.models.base.BaseModel method\)](#), 12  
[get\\_forbidden\\_message\(\) \(django\\_powerbank.views.auth.AbstractAuthorizedView method\)](#), 16  
[get\\_prep\\_value\(\) \(django\\_powerbank.db.models.fields.JSONField method\)](#), 11  
[get\\_queryset\(\) \(django\\_powerbank.views.generic.FilterMixin method\)](#), 16  
[get\\_slug\\_value\(\) \(django\\_powerbank.db.models.fields.AutoSlugField method\)](#), 10  
[get\\_slug\\_value\(\) \(django\\_powerbank.db.models.fields.UniqueSlugField method\)](#), 12  
[get\\_source\\_value\(\) \(django\\_powerbank.db.models.fields.AutoSlugField method\)](#), 10  
[get\\_success\\_url\(\) \(django\\_powerbank.views.mixins.ReturnUrlMx method\)](#), 17

## H

[handle\\_anonymous\(\) \(django\\_powerbank.views.auth.AuthenticatedView method\)](#), 16  
[handle\\_forbidden\(\) \(django\\_powerbank.views.auth.AbstractAuthorizedView method\)](#), 16  
[Http302](#), 17  
[Http400](#), 17  
[Http401](#), 17  
[Http403](#), 17

## I

[input\\_type \(django\\_powerbank.forms.widgets.PhoneInput attribute\)](#), 14  
[is\\_active \(django\\_powerbank.testing.factories.UserFactory attribute\)](#), 15

[is\\_authenticated\(\)](#)  
     ([django\\_powerbank.views.auth.AuthenticatedView](#)  
     [method](#)), 16

[is\\_authorized\(\)](#) ([django\\_powerbank.views.auth.AbstractAuthorizedView](#)  
[method](#)), 16

[is\\_authorized\(\)](#) ([django\\_powerbank.views.auth.StaffRequiredMixin](#)  
[method](#)), 16

[is\\_staff](#) ([django\\_powerbank.testing.factories.UserFactory](#)  
[attribute](#)), 15

**J**

[JSONField](#) (class in [django\\_powerbank.db.models.fields](#)), 11

**L**

[last\\_name](#) ([django\\_powerbank.testing.factories.UserFactory](#)  
[attribute](#)), 15

**M**

[MarkdownField](#) (class in [django\\_powerbank.db.models.fields](#)), 11

[MaxCurrentYearValidator](#) (class in [django\\_powerbank.core.validators](#)), 9

[MaxTodayDateValidator](#) (class in [django\\_powerbank.core.validators](#)), 9

[media](#) ([django\\_powerbank.forms.widgets.PhoneInput](#)  
[attribute](#)), 14

[media](#) ([django\\_powerbank.forms.widgets.SelectizeBase](#)  
[attribute](#)), 14

[media](#) ([django\\_powerbank.forms.widgets.SelectizeSelect](#)  
[attribute](#)), 14

[media](#) ([django\\_powerbank.forms.widgets.SelectizeTags](#)  
[attribute](#)), 14

[media](#) ([django\\_powerbank.forms.widgets.Typeahead](#) [attribute](#)), 14

[message](#) ([django\\_powerbank.core.validators.MaxCurrentYearValidator](#)  
[attribute](#)), 9

[message](#) ([django\\_powerbank.core.validators.MaxTodayDateValidator](#)  
[attribute](#)), 9

[MigrationsCheck](#) (class in [django\\_powerbank.testing.base](#)), 15

[MigrationsCheckMx](#) (class in [django\\_powerbank.testing.base](#)), 15

[model\\_to\\_request\\_data\\_dict\(\)](#) (in module [django\\_powerbank.testing.utils](#)), 16

[MsisdnValidator](#) (class in [django\\_powerbank.core.validators](#)), 9

**N**

[name](#) ([django\\_powerbank.apps.DjangoPowerbankConfig](#)  
[attribute](#)), 17

**O**

[optgroupgroups\(\)](#) ([django\\_powerbank.forms.widgets.SelectizeSelect](#)  
[method](#)), 14

**P**

[PhoneField](#) (class in [django\\_powerbank.db.models.fields](#)), 11

[PhoneField](#) (class in [django\\_powerbank.forms.fields](#)), 13

[PhoneInput](#) (class in [django\\_powerbank.forms.widgets](#)), 14

[PlNipField](#) (class in [django\\_powerbank.db.models.fields.pl](#)), 10

[PlRegonField](#) (class in [django\\_powerbank.db.models.fields.pl](#)), 10

[populate\(\)](#) ([django\\_powerbank.db.models.base.BaseModel](#)  
[method](#)), 12

[pre\\_save\(\)](#) ([django\\_powerbank.db.models.fields.AutoSlugField](#)  
[method](#)), 10

[pre\\_save\(\)](#) ([django\\_powerbank.db.models.fields.MarkDownField](#)  
[method](#)), 11

[pre\\_save\(\)](#) ([django\\_powerbank.db.models.fields.SecretField](#)  
[method](#)), 12

**R**

[ready\(\)](#) ([django\\_powerbank.apps.DjangoPowerbankConfig](#)  
[method](#)), 18

[ReturnUrlMx](#) (class in [django\\_powerbank.views.mixins](#)), 17

**S**

[SecretField](#) (class in [django\\_powerbank.db.models.fields](#)), 12

[SelectizeBase](#) (class in [django\\_powerbank.forms.widgets](#)), 14

[SelectizeSelect](#) (class in [django\\_powerbank.forms.widgets](#)), 14

[SelectizeTags](#) (class in [django\\_powerbank.forms.widgets](#)), 14

[setUp\(\)](#) ([django\\_powerbank.testing.base.AdminUserTestCase](#)  
[method](#)), 15

[setUp\(\)](#) ([django\\_powerbank.testing.base.MigrationsCheckMx](#)  
[method](#)), 15

[setUp\(\)](#) ([django\\_powerbank.testing.base.StaffUserTestCase](#)  
[method](#)), 15

[setUp\(\)](#) ([django\\_powerbank.testing.base.UserTestCase](#)  
[method](#)), 15

[setup\\_app\\_settings\(\)](#) (in module [django\\_powerbank.apps](#)), 18

[SmartUrlValidator](#) (class in [django\\_powerbank.core.validators](#)), 9

[SourceFieldMixin](#) (class in [django\\_powerbank.db.models.fields](#)), 12

StaffRequiredMixin (class in [value\\_to\\_string\(\)](#)  
[django\\_powerbank.views.auth](#)), 16 (django\_powerbank.db.models.fields.JSONField  
StaffUserTestCase (class in [method](#)), 11  
[django\\_powerbank.testing.base](#)), 15 values (django\_powerbank.db.models.fields.ChoicesIntEnum  
[attribute](#)), 11  
[verbose\\_name](#) (django\_powerbank.apps.DjangoPowerbankConfig  
[attribute](#)), 18  
T  
TableStatusQuerySet (class in [method](#)), 13  
[django\\_powerbank.db.models.query](#)), 13  
tearDown() (django\_powerbank.testing.base.MigrationsCheckMx  
[method](#)), 15 widget (django\_powerbank.forms.fields.DateRangeField  
[attribute](#)), 13  
template\_name (django\_powerbank.forms.widgets.SelectizeSelectattribute), 13  
[attribute](#)), 14 widget (django\_powerbank.forms.fields.PhoneField at-  
[attribute](#)), 13  
template\_name (django\_powerbank.forms.widgets.SelectizeTags [tribute](#)), 13  
[attribute](#)), 14 widget (django\_powerbank.forms.fields.TypeaheadField  
[attribute](#)), 13  
test\_missing\_migrations() [Map\\_query\(\)](#) (django\_powerbank.db.models.query.ApproxQuerySet  
[method](#)), 15 class method), 13  
to\_dict() (django\_powerbank.db.models.base.BaseModel  
[method](#)), 12  
to\_python() (django\_powerbank.db.models.fields.JSONField  
[method](#)), 11  
to\_python() (django\_powerbank.db.models.fields.MarkDownField  
[method](#)), 11  
to\_python() (django\_powerbank.db.models.fields.PhoneField  
[method](#)), 12  
to\_python() (django\_powerbank.db.models.fields.pl.PlNipField  
[method](#)), 10  
to\_python() (django\_powerbank.db.models.fields.pl.PlRegonField  
[method](#)), 10  
to\_python() (django\_powerbank.forms.fields.DateRangeField  
[method](#)), 13  
Typeahead (class in [method](#)), 14  
[django\\_powerbank.forms.widgets](#)), 14  
TypeaheadField (class in [method](#)), 13  
[django\\_powerbank.forms.fields](#)), 13  
U  
UniqueSlugField (class in [method](#)), 12  
[django\\_powerbank.db.models.fields](#)), 12  
UserFactory (class in [method](#)), 15  
[django\\_powerbank.testing.factories](#)), 15  
username (django\_powerbank.testing.factories.UserFactory  
[attribute](#)), 15  
UserTestCase (class in [method](#)), 15  
[django\\_powerbank.testing.base](#)), 15  
V  
validate() (django\_powerbank.db.models.fields.JSONField  
[method](#)), 11  
value\_from\_object() (django\_powerbank.db.models.fields.JSONField  
[method](#)), 11