Easy-select2 Documentation

Release 1.2.2

Lobanov Stanislav aka asyncee

April 29, 2014

Contents

Contents:

CHAPTER 1

Installation

1. install this package as usual, using `python setup.py install`, `pip install django-easy-select2` or download sources and install to your python path.
2. add `easy_select2` to `INSTALLED_APPS` in your settings.py
3. Use `python manage.py collectstatic` or manually copy easy_select2's static directory to your project's static directory (if you serve your static with nginx, for example).
4. Modify your admin.py.
5. Check out admin in browser.

CHAPTER 2

Quickstart

In your admin.py:

```
from django.contrib import admin
from easy_select2 import select2_modelform
from polls.models import Poll

PollForm = select2_modelform(Poll, attrs={'width': '250px'})

class PollAdmin(admin.ModelAdmin):
    form = PollForm
```

Thats all. All your choice widgets are select2 widgets 250px wide.

CHAPTER 3

Configuration

`django-easy-select2` bundles jQuery and Select2 static files. You can use them, or specify your own files to include in widget.

To use bundled static, just install an application.

To use your custom static files, you can specify next settings in your settings.py:

- `SELECT2_JS` - path to `select2.js` file. Specify path without static directory, because full URL will be interpolated using `static` function from `staticfiles` application. Default: `easy_select2/vendor/select2/select2.min.js`
- `SELECT2_CSS` - path to `select2.css` file. Default: `easy_select2/vendor/select2/select2.min.css`
- `SELECT2_USE_BUNDLED_JQUERY` - default is `True`. Set to `False` if your already have included custom jQuery.

Usage

There are `Select2` and `Select2Multiple` widget classes for choice fields and `Select2TextInput` for non-choice fields which can prodive a list of pre-set choices, or can accept arbitrary input.

You can use `Select2` and `Select2Multiple` on any form field, as usual django widget:

```
class Form(forms.Form):
    field = forms.ModelChoiceField(queryset=qs, widget=Select2())
```

or:

```
class Form(forms.Form):
    field = forms.ModelChoiceField(queryset=qs, widget=Select2Multiple(
        select2attrs={'width': 'auto'}
    ))
```

`Select2` and `Select2Multiple` is simple classes build with `Select2Mixin`:

```
class Select2Multiple(Select2Mixin, forms.SelectMultiple):
    pass

class Select2(Select2Mixin, forms.Select):
    pass
```

`Select2Mixin` is a simple widget mixin with predefined `Media` class and custom render method, which applies *$.fn.select2()* method on html input.

To use `Select2TextInput` do NOT set a choices attribute on the model field, but DO supply a `data` attribute to `select2attrs` that contains a list of dictionaries each having at least an `id` and `text` terms like so:

```
form.fields['myfield'].widget = Select2TextInput(
    select2attrs={
        'data': [ {'id': 'your data', 'text': 'your data'}, ... ],
    },
)
```

`Select2TextInput` will be rendered as combo-box widget that can accept arbitrary input, but also has some default choices for user.

Note, that `select2attrs` can accept not only dicts, but also strings or any other objects that can be converted to unicode with unicode() builtin. If dictionary is passed, it will be dumped to json.

If you want to use it with all form fields automatically, without specifying each field, you can create your `ModelForm` class with `Meta` class constructed by custom `Meta` factory:

```
from easy_select2 import select2_modelform_meta

class SomeModelForm(forms.ModelForm):
    Meta = select2_modelform_meta(SomeModel)
```

`select2_modelform_meta()` is a simple factory, that produces a `Meta` class with model attribute set to specified model and `widgets` attribute set to dictionary, containing all selectable fields on model. Every selectable field will be converted from standard widget to `Select2` or `Select2Multiple` widget.

If you are lazy, you can use `ModelForm` factory to build ready-to-use ModelForm for model with `select2_modelform()`:

```
from easy_select2 import select2_modelform

MyModelForm = select2_modelform(MyModel)
```

is the same like:

```
class MyModelForm(forms.ModelForm):
    Meta = select2_modelform_meta(models.SomeModelForm)
```

MyModelForm is an instance of ModelForm with `model` attribute set to `MyModel`, and appropriate `Meta` class.

There is also an `apply_select2()` function that dynamically creates new widget class mixed with `Select2Mixin`.

Usage, for example:

```
class SomeModelForm(admin.ModelForm):
    class Meta:
        widgets = {
            'field': apply_select2(forms.Select),
        }
```

So, `apply_select2(forms.Select)` will return new class, named Select2Select, mixed with Select2Mixin.

CHAPTER 5

Reference

5.1 Widgets

5.2 Utils

CHAPTER 6

Sampleproject

Sample project useful for testing django applications and other utility needs.

6.1 Installation

1. git clone https://github.com/asyncee/django-easy-select2.git
2. cd django-easy-select2/sampleproject
3. ./bootstrap.sh - creates virtualenv and installs django (from requirements.txt)
4. source env/bin/activate
5. ./manage.py syncdb –migrate

6.2 Configuration

Project ships with sane defaults (settings are pretty verbose):

- sqlite3 database
- filebased cache in /tmp
- cached sessions
- console email backend
- non-cached template loaders
- *css/js/img* default static dirs
- default *templates* directory
- nice default loggers

6.3 Usage

After you bootstrapped a project, you can fill it with some data and play with Note model admin.

Changelog

7.1 Version 1.2.3

- Python 3.3 support, thanks to *dzerrenner*

7.2 Version 1.2.2

- Rendering select2attrs as unicode or json based on type

Now, if select2attrs is instance of basestring (str or unicode), it will be casted to unicode, else it will be turned to json string.

7.3 Version 1.2.1

- Extended package-level imports with Select2TextInput

7.4 Version 1.2.0

- added Select2TextInput, thanks to *mkoistinen*

7.5 Version 1.1.1

- issue#1 fix (django-admin-sortable compatibility), thanks to @mkoistinen

CHAPTER 8

Introduction

This is django application that brings select2 widget to select inputs in admin.

8.1 How it looks

Select one of existing values with single-valued choice field (ForeignKeyField, for example): Easily select 1 or more "categories" for your project, you can also add a new one in the normal, Django-Admin manner by using the green + button with multiple-valued choice field (ManyToManyField): Don't see the "mood" you want? No problem, just type in a new one. It will be there as a choice for the next time too (text input). Continue to *Installation*.

CHAPTER 9

Indices and tables

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