Django Cms Bootstrap Grid Builder Documentation

Release 0.1.2

FrankHood Business Solutions srl

Nov 12, 2021

CONTENTS

Contents:

CHAPTER

ONE

DJANGO CMS BOOTSTRAP GRID BUILDER

This tool offers the preliminary drafting of a grid layout consisting of containers, rows and columns, allows you to correctly define the spaces assigned to each page content, and to map these spaces for the different responsive steps.

1.1 Documentation

The full documentation is at https://django-cms-bootstrap-grid-builder.readthedocs.io/en/latest/.

1.1.1 Quickstart

> **warning** ATTENTION !!! This package requires **django-cms** already installed.

Install Django CMS bootstrap grid builder:

```
pip install django-cms-bootstrap-grid-builder
```

Add it to your *INSTALLED_APPS*:

```
INSTALLED_APPS = (
    ...
    'bootstrap_grid_builder',
    ...
)
```

The first thing to do is to override the page creation wizard with the one offered by the package.

```
# project/urls.py

from cms.cms_wizards import cms_page_wizard, cms_subpage_wizard
from bootstrap_grid_builder.cms_wizards import cms_wizards
from cms.wizards.wizard_pool import wizard_pool

# OVERRIDE CMS WIZARD
wizard_pool.unregister(cms_page_wizard)
wizard_pool.unregister(cms_subpage_wizard)
```

(continues on next page)

(continued from previous page)

```
wizard_pool.register(cms_wizards.cms_page_wizard)
wizard_pool.register(cms_wizards.cms_subpage_wizard)
```

A variable must be defined to specify the name of the placeholder that will contain the plugins generated by the page creation wizard.

```
GRID_PLUGIN_STRUCTURE_PLACEHOLDER = "grid_placeholder"
```

Add the placeholder name inside your home.html template like this:

```
{% load cms_tags sekizai_tags %}
<html>
    <head>
        <title>{% page_attribute "page_title" %}</title>
        {% render_block "css" %}
    </head>
    <body>
        {% cms_toolbar %}
        {% placeholder "grid_placeholder" %}
        {% render_block "js" %}
    </body>
</html>
```

Then run migrate to apply package migrations:

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

1.1.2 HowTo customize Grid Plugins & Grid Plugin Models

You can customize the wizard-generated plugins to add functionality or fields to the basic implementation of the tool.

The plugins registered by the package are:

- GridContainerPlugin
- GridRowPlugin
- GridColPlugin

To modify one of these plugins you need:

- Create an associated model in which to add your own field.
- Create a plugin in which to insert the field previously added to the model and make the unregister of the base plugin and the register of the plugin just created.

```
# your_app/models.py

class MyCustomGridContainerPluginModel(GridContainerPluginAbstractModel):
    my_field = models.CharField("My Field", max_length=255)

    class Meta:
        verbose_name = _("My Custom grid container plugin")
        verbose_name_plural = _("My Custom grid container plugins")
```

(continues on next page)

(continued from previous page)

```
# your_app/cms_plugins.py

plugin_pool.unregister_plugin(GridContainerPlugin)

@plugin_pool.register_plugin
class MyCustomGridContainerPlugin(GridContainerPlugin):
    model = MyCustomGridContainerPluginModel
    module = _("Custom")
    name = _("Custom Grid Container")
    render_template = 'path/to/my/custom/template.html'

    fieldsets = (
        (None, {"fields": (
            ("variant_class", "tag_type",),
            ("my_field",),
        )}),
    )
```

Following these changes it is necessary to set variables in the settings.py file to specify the name of the plugin that must be generated by the wizard instead of the base plugin.

```
# project/settings.py

GRID_CONTAINER_PLUGIN = "MyCustomGridContainerPlugin"
# this are the others plugins variables
GRID_COL_PLUGIN = ""
GRID_ROW_PLUGIN = ""
```

After models creation run makemigration & migrate to create yours models in database.

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

1.1.3 Running Tests

```
source <YOURVIRTUALENV>/bin/activate
(myenv) $ pip install tox
(myenv) $ tox
```

1.1.4 Development commands

```
# Back-end
$ pip install -r requirements_dev.txt
$ pre-commit install
$ python manage.py migrate
$ python manage.py runserver
```

1.2 Frontend

This is a Vue.js application for creating custom bootstrap grids throughout an intuitive interface and draggable elements

1.2.1 Browser Compatibility

The page-layout-builder component is compatible with modern browsers such as Chrome, Firefox, Safari, Opera, and Edge. It also supports Internet Explorer 11 but with limited performance.

1.2.2 Frontend source folder ascii tree

```
/django-cms-bootstrap-grid-builder/src
├─ main.js  //Entrypoint for build
├─ index.js  //Entrypoint for development
├─ components
│   ├─ CustomDragElement.vue
│   ├─ page-layout-builder.vue
│   ├─ GridItem.vue
│   ├─ GridLayout.vue
│   └─ index.js
└─ helpers
    ├─ DOM.js
    ├─ draggableUtils.js
    ├─ responsiveUtils.js
    └─ utils.js
```

1.2.3 How it works

The informations obtained from the interface configuration are serialized into a JSON object and sent to the backend wich replicates the desired grid structure with Django plugins templates

1.2.4 Development commands

```
# Front-end
$ npm i -g yarn rimraf @vue/cliz
$ yarn install
$ yarn serve (for development, lauches local live reloading server)
$ yarn build (for production build, creates dist at django-cms-bootstrap-grid-builder/
↪bootstrap_grid_builder/static/cms_plugin_structure/dist)
```

1.2.5 Credits

Tools used in rendering this package:

- Cookiecutter
- cookiecutter-djangopackage
- element-resize-detector
- interactjs
- vue-drag-drop
- google-palette
- bootstrap

CHAPTER

TWO

INSTALLATION

At the command line:

```
$ easy_install django-cms-bootstrap-grid-builder
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv django-cms-bootstrap-grid-builder
$ pip install django-cms-bootstrap-grid-builder
```

CHAPTER

THREE

USAGE

To use Django Cms Bootstrap Grid Builder in a project, add it to your *INSTALLED_APPS*:

```
INSTALLED_APPS = (
    ...
    'bootstrap_grid_builder.apps.CmsBootstrapGridBuilderConfig',
    ...
)
```

Add Django Cms Bootstrap Grid Builder's URL patterns:

```
from bootstrap_grid_builder import urls as bootstrap_grid_builder_urls


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

CHAPTER

FOUR

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/frankhood/django-cms-bootstrap-grid-builder/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.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/frankhood/django-cms-bootstrap-grid-builder/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 :)

4.2 Get Started!

Ready to contribute? Here's how to set up *django-cms-bootstrap-grid-builder* for local development.

1. Fork the *django-cms-bootstrap-grid-builder* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-cms-bootstrap-grid-builder.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-cms-bootstrap-grid-builder
$ cd django-cms-bootstrap-grid-builder/
$ 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 bootstrap_grid_builder tests
$ python setup.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.

4.3 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 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/frankhood/django-cms-bootstrap-grid-builder/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_bootstrap_grid_builder
```

CHAPTER

FIVE

CREDITS

5.1 Development Lead

- FrankHood Business Solutions srl <info@frankhood.it>

5.2 Contributors

None yet. Why not be the first?

CHAPTER

SIX

HISTORY

6.1 0.1.0 (2021-11-04)

- First release on PyPI.

6.2 0.1.1 (2021-11-05)

- Renamed LICENSE file in LICENSE.md for publication on PyPI.

6.3 0.1.2 (2021-11-08)

- Added dist files removing dist from .gitignore and updated README.rst.