
django-admirarchy Documentation

Release 1.2.2

Igor ‘idle sign’ Starikov

Dec 18, 2021

Contents

1	Description	3
2	Requirements	5
3	Table of Contents	7
3.1	Quickstart	7
3.2	Advanced usage	8

<https://github.com/idlesign/django-admirarchy>

CHAPTER 1

Description

Django Admin addon to navigate through hierarchies.

Have you ever wanted Django Admin to be able to navigate through hierarchies?

Without existing models modifications? Yeah!

Admirarchy does it in an old-school way, just like Norton Commander and Co - one level at a time.

Hierarchies described as adjacency lists and nested sets are supported.

CHAPTER 2

Requirements

1. Python 3.6+
2. Django 1.8+
3. Django Admin contrib

3.1 Quickstart

Note: Make sure `admirarchy` is listed in `INSTALLED_APPS` in settings file of your project (usually ‘`settings.py`’).

With a few minor changes...

```
# admin.py of your application
from django.contrib import admin

from admirarchy.toolbox import HierarchicalModelAdmin

from .models import MyModel  # Let's say this model represents a hierarchy.

# Inherit from HierarchicalModelAdmin instead of admin.ModelAdmin
@admin.register(MyModel)
class MyModelAdmin(HierarchicalModelAdmin):

    hierarchy = True  # This enables hierarchy handling.

    # and other code as usual...
```



...your admin...

Select Category to change

Action: <input type="text" value="-----"/>		Go	0 of 4 selected
Action checkbox	Category		
<input type="checkbox"/>	No children		
<input type="checkbox"/>	This one is inside of another one		
<input type="checkbox"/>	And another one		
<input type="checkbox"/>	Root		
4 Categories			

... turns into something similar to this:

Select Category to change

Action: <input type="text" value="-----"/>		Go	0 of 3 selected
	Action checkbox	Category	
			
	<input type="checkbox"/>	No children	
	<input type="checkbox"/>	And another one	
2 Objects inside: 1			

3.2 Advanced usage

Here are some words on more advanced `admirarchy` usage.

3.2.1 Adjacency lists

For hierarchies described through adjacency lists you can explicitly define a name of a field in your model containing parent item identifier (defaults to `parent`):

```

from django.contrib import admin

from admirarchy.toolbox import HierarchicalModelAdmin, AdjacencyList

from .models import MyModel

@admin.register(MyModel)
class MyModelAdmin(HierarchicalModelAdmin):

    hierarchy = AdjacencyList('upper') # That says MyModel uses `upper` field to_
    ↪store parent ID.

```

3.2.2 Nested sets

For hierarchies described through nested sets you can explicitly define names of fields containing left and right set limits, and nesting level (defaults to lft, rgt and level respectively):

```

from django.contrib import admin

from admirarchy.toolbox import HierarchicalModelAdmin, NestedSet

from .models import MyModel

@admin.register(MyModel)
class MyModelAdmin(HierarchicalModelAdmin):

    # That says MyModel uses has 'left_border', 'right_border', 'depth' to describe_
    ↪nesting.
    hierarchy = NestedSet('left_border', 'right_border', 'depth')

```