



[Django](#) [Views](#) [Model](#) [Template](#) [Forms](#) [Jinja](#) [Python SQLite](#) [Flask](#) [Json](#) [Postman](#) [Interview Ques](#)

Django ModelFormSets

Last Updated : 09 Jan, 2020

ModelFormsets in a Django is an advanced way of handling multiple forms created using a model and use them to create model instances. In other words, ModelFormsets are a group of forms in Django. One might want to initialize multiple forms on a single page all of which may involve multiple POST requests, for example

```
class GeeksModel(models.Model):  
    title = models.CharField(max_length = 200)  
    description = models.TextField()
```

Now if one wants to create a modelformset for this model, modelformset_factory needs to be used. A formset is a layer of abstraction to work with multiple forms on the same page. It can be best compared to a data grid.

```
from django.forms import formset_factory  
GeeksFormSet = modelformset_factory(GeeksModel)
```

Creating and using Django ModelFormsets

Illustration of **Rendering Django ModelFormsets manually** using an Example. Consider a project named `geeksforggeeks` having an app named `geeks`.

Refer to the following articles to check how to create a project and an app in Django.

- [How to Create a Basic Project using MVT in Django?](#)
- [How to Create an App in Django ?](#)

In your geeks app make a new file called models.py where you would be making all your models. To create a Django model you need to use Django Models. Let's demonstrate how,
In your models.py Enter the following,

```
# import the standard Django Model
# from built-in library
from django.db import models

# declare a new model with a name "GeeksModel"
class GeeksModel(models.Model):

    # fields of the model
    title = models.CharField(max_length = 200)
    description = models.TextField()

    # renames the instances of the model
    # with their title name
    def __str__(self):
        return self.title
```

Let's explain what exactly is happening, left side denotes the name of the field and to right of it, you define various functionalities of an input field correspondingly. A field's syntax is denoted as

Syntax :

```
Field_name = models.FieldType(attributes)
```

Now to create a simple formset of this form, move to views.py and create a formset_view as below.

```
from django.shortcuts import render

# relative import of forms
from .forms import GeeksForm

# importing formset_factory
from django.forms import formset_factory

def formset_view(request):
    context = {}

    # creating a formset
```

```

GeeksFormSet = modelformset_factory(GeeksForm)
formset = GeeksFormSet()

# Add the formset to context dictionary
context['formset'] = formset
return render(request, "home.html", context)

```

To render the formset through HTML, create a html file “home.html”. Now let’s edit templates > home.html

```

<form method="POST" enctype="multipart/form-data">
    {% csrf_token %}
    {{ formset.as_p }}
    <input type="submit" value="Submit">
</form>

```

All set to check if our formset is working or not let’s visit <http://localhost:8000/>.

The screenshot shows a web browser at localhost:8000 displaying a Django formset. The page contains two identical forms stacked vertically. Each form has a 'Title' text input field and a 'Description' text area. The first form's title is 'Naveen' and the second's is 'GFG is Best'. Each form has a 'Submit' button. The browser's address bar shows 'localhost:8000'.

Our modelformset is working completely. Let’s learn how to modify this formset to use extra features of this formset.

How to create Multiple forms using Django ModelFormsets

Django formsets are used to handle multiple instances of a form. One can create multiple forms easily using extra attribute of Django Formsets. In `geeks/views.py`,

```

from django.shortcuts import render

```

```

# relative import of forms
from .models import GeeksModel

# importing formset_factory
from django.forms import modelformset_factory

def modelformset_view(request):
    context = {}

    # creating a formset and 5 instances of GeeksForm
    GeeksFormSet = modelformset_factory(GeeksModel, fields=['title', 'description'])
    formset = GeeksFormSet()

    # Add the formset to context dictionary
    context['formset'] = formset
    return render(request, "home.html", context)

```

The keyword argument `extra` makes multiple copies of same form. If one wants to create 5 forms enter `extra = 5` and similarly for others. Visit <http://localhost:8000/> to check if 5 forms are created.

The screenshot shows a web browser window with the address bar displaying 'localhost:8000'. The page content includes a 'Submit' button at the top. Below it, there are five identical form instances stacked vertically. Each form consists of a 'Title:' label followed by a text input field, and a 'Description:' label followed by a larger text area. The first form's description field contains the text 'GFG is Best'. At the bottom of the page, there is another 'Submit' button and a long alphanumeric string: '>>>>> 177726cb0667a203bad6744cf646db8ba0df'.

Handling Multiple forms using Django Formsets

Creating a form is much easier than handling the data entered into those fields at the back end. Let's try to demonstrate how one can easily use the data of a model formset in a view. When trying to handle formset, Django formsets required one extra argument `{{ formset.management_data }}`. To know more

about Management data, [Understanding the ManagementForm](#).

In templates/home.html,

```
<form method="POST" enctype="multipart/form-data">

    <!-- Management data of formset -->
    {{ formset.management_data }}

    <!-- Security token -->
    {% csrf_token %}

    <!-- Using the formset -->
    {{ formset.as_p }}

    <input type="submit" value="Submit">
</form>
```

Now to check how and what type of data is being rendered edit **formset_view** to print the data. In `geeks/view.py`,

```
from django.shortcuts import render

# relative import of forms
from .forms import GeeksForm

# importing formset_factory
from django.forms import formset_factory

def formset_view(request):
    context = {}

    # creating a formset and 5 instances of GeeksForm
    GeeksFormSet = formset_factory(GeeksForm, extra = 3)
    formset = GeeksFormSet(request.POST or None)

    # print formset data if it is valid
    if formset.is_valid():
        for form in formset:
            print(form.cleaned_data)

    # Add the formset to context dictionary
    context['formset'] = formset
    return render(request, "home.html", context)
```

Now let's try to enter data in the formset through <http://localhost:8000/>

local:localhost:8000

<<<<<< HEAD

Submit

=====

Title:

Description:

Description:

Title:

Description:

Description:

Title:

Description:

Description:

Submit

>>>>>> f777a26cbd66f7a203bad6744cf646d8e8ba0df

Hit submit and data will be saved in GeeksModel where server is running. One can use this data in any manner conveniently now.

Select geeks model to change

ADD GEEKS MODEL +

Action: Go

0 of 3 selected

<input type="checkbox"/>	GEEKS MODEL
<input type="checkbox"/>	Title 3
<input type="checkbox"/>	Title 2
<input type="checkbox"/>	Title 1

3 geeks models

Are you ready to elevate your web development skills from foundational knowledge to advanced expertise? Explore our [Mastering Django Framework - Beginner to Advanced Course](#) on GeeksforGeeks, designed for aspiring developers and experienced programmers. This comprehensive course covers everything you need to know about Django, from the basics to advanced features. Gain practical experience through **hands-on projects** and real-world applications, mastering essential Django principles and techniques. Whether you're just starting or looking to refine your skills, this course will empower you to build sophisticated web applications efficiently. Ready to enhance your web development journey? Enroll now and unlock your potential with Django!

N Nave...



10

Previous Article

How to create a form using Django Forms ?

Next Article

Django ModelForm - Create form from Models

Similar Reads

Adding Tags Using Django-Taggit in Django Project

Django-Taggit is a Django application which is used to add tags to blogs, articles etc. It makes very easy for us to make adding the tags functionality to our django...

2 min read

How to customize Django forms using Django Widget Tweaks ?

Django forms are a great feature to create usable forms with just few lines of code. But Django doesn't easily let us edit the form for good designs. Here, we...

3 min read

Styling Django Forms with django-crispy-forms

Django by default doesn't provide any Django form styling method due to which it takes a lot of effort and precious time to beautifully style a form. django-crispy...

1 min read

Integrating Django with Reactjs using Django REST Framework

In this article, we will learn the process of communicating between the Django Backend and React js frontend using the Django REST Framework. For the sake...

15+ min read

How to Fix Django "ImportError: Cannot Import Name 'six' from..."

After Django 3.0, django.utils.six was removed. The "ImportError: Cannot Import Name 'six' from 'django.utils'" error occurs because of the dependency issue. Thi...

3 min read

Handling Ajax request in Django

Introduction This tutorial explains how to carry out an ajax request in the Django web framework. We will create a simple post-liking app as a part of the exampl...

4 min read

Django Introduction | Set 2 (Creating a Project)

Note- This article is in continuation of Django introduction. Popularity of Django Django is used in many popular sites like as: Disqus, Instagram, Knight...

3 min read

now - Django Template Tags

A Django template is a text document or a Python string marked-up using the Django template language. Django being a powerful Batteries included...

2 min read

Python | User groups with Custom permissions in Django

Let's consider a trip booking service, how they work with different plans and packages. There is a list of product which subscriber gets on subscribing to...

4 min read

Python | Django Admin Interface

In this article, we delve into the capabilities and advantages of the Django Admin Interface, exploring how its customizable features and streamlined workflows...

4 min read

Article Tags : [Python](#) [Django-forms](#) [Python Django](#)

Practice Tags : [python](#)



Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305) | Registered Address:- K 061,
Tower K, Gulshan Vivante Apartment,
Sector 137, Noida, Gautam Buddh
Nagar, Uttar Pradesh, 201305



Company

About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Computer Science

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

DevOps

Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

Inteview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved