



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

UpdateView – Class Based Views Django

Last Updated : 20 Jul, 2020

UpdateView refers to a view (logic) to update a particular instance of a table from the database with some extra details. It is used to update entries in the database, for example, updating an article at geeksforgeeks. We have already discussed basics of Update View in [Update View – Function based Views Django](#). Class-based views provide an alternative way to implement views as Python objects instead of functions. They do not replace function-based views, but have certain differences and advantages when compared to function-based views:

- Organization of code related to specific HTTP methods (GET, POST, etc.) can be addressed by separate methods instead of conditional branching.
- Object oriented techniques such as mixins (multiple inheritance) can be used to factor code into reusable components.

Class based views are simpler and efficient to manage than function-based views. A function based view with tons of lines of code can be converted into a class based views with few lines only. This is where Object Oriented Programming comes into impact.

Django UpdateView – Class Based Views

Illustration of **How to create and use UpdateView** using an Example. Consider a project named geeksforgeeks 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 ?](#)

After you have a project and an app, let's create a model of which we will be creating instances through our view. In `geeks/models.py`,

```
# 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
```

After creating this model, we need to run two commands in order to create Database for the same.

Python `manage.py makemigrations`

Python `manage.py migrate`

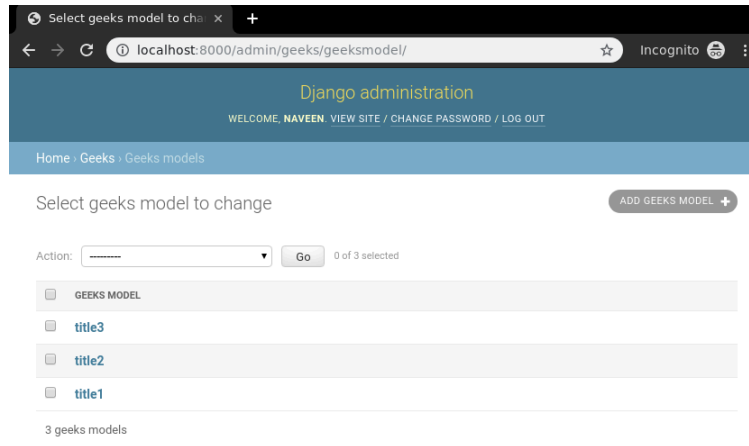
Now let's create some instances of this model using shell, run from bash,

Python `manage.py shell`

Enter following commands

```
>>> from geeks.models import GeeksModel
>>> GeeksModel.objects.create(
            title="title1",
            description="description1").save()
>>> GeeksModel.objects.create(
            title="title2",
            description="description2").save()
>>> GeeksModel.objects.create(
            title="title2",
            description="description2").save()
```

Now we have everything ready for back end. Verify that instances have been created from <http://localhost:8000/admin/geeks/geeksmodel/>



Class Based Views automatically setup everything from A to Z. One just needs to specify which model to create UpdateView for, then Class based UpdateView will automatically try to find a template in `app_name/modelname_form.html`. In our case it is `geeks/templates/geeks/geeksmodel_form.html`. Let's create our class based view. In `geeks/views.py`,

```
# import generic UpdateView
from django.views.generic.edit import UpdateView

# Relative import of GeeksModel
from .models import GeeksModel

class GeeksUpdateView(UpdateView):
    # specify the model you want to use
    model = GeeksModel

    # specify the fields
    fields = [
        "title",
        "description"
    ]

    # can specify success url
    # url to redirect after successfully
    # updating details
    success_url = "/"
```

Now create a url path to map the view. In `geeks/urls.py`,

```
from django.urls import path

# importing views from views..py
from .views import GeeksUpdateView
urlpatterns = [
    # <pk> is identification for id field,
    # <slug> can also be used
    path('<pk>/update', GeeksUpdateView.as_view()),
]
```

Create a template in `templates/geeks/geeksmodel_form.html`,

```
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <input type="submit" value="Save">
</form>
```

Let's check what is there on <http://localhost:8000/1/update/>



The screenshot shows a web browser window with the address bar displaying 'localhost:8000/1/update'. The page content includes a form with the following elements:

- A text input field labeled 'Title:' containing the text 'title1'.
- A text area labeled 'Description:' containing the text 'description1'.
- A 'Save' button located below the description field.

N Nave...



11

Previous Article

DetailView - Class Based Views Django

Next Article

FormView - Class Based Views Django

Similar Reads

Class Based vs Function Based Views - Which One is Better to Use in Django?

Django...We all know the popularity of this python framework all over the world. This framework has made life easier for developers. It has become easier for...

7 min read

Createview - Class Based Views Django

Create View refers to a view (logic) to create an instance of a table in the database. We have already discussed basics of Create View in Create View –...

3 min read

ListView - Class Based Views Django

List View refers to a view (logic) to display multiple instances of a table in the database. We have already discussed the basics of List View in List View –...

4 min read

DetailView - Class Based Views Django

Detail View refers to a view (logic) to display one instances of a table in the database. We have already discussed basics of Detail View in Detail View –...

3 min read

DeleteView - Class Based Views Django

Delete View refers to a view (logic) to delete a particular instance of a table from the database. It is used to delete entries in the database for example, deleting a...

3 min read

FormView - Class Based Views Django

FormView refers to a view (logic) to display and verify a Django Form. For example, a form to register users at Geeksforgeeks. Class-based views provide ...

3 min read

Class based views - Django Rest Framework

Class-based views help in composing reusable bits of behavior. Django REST Framework provides several pre-built views that allow us to reuse common...

12 min read

Django Class Based Views

Django is a Python-based web framework that allows you to quickly create web applications. It has a built-in admin interface which makes it easy to work with it...

10 min read

How to Use permission_required Decorators with Django Class-Based Views

In Django, permissions are used to control access to views and resources. When working with function-based views (FBVs), decorators like `@permission_require...`

4 min read

Create View - Function based Views Django

Create View refers to a view (logic) to create an instance of a table in the database. It is just like taking an input from a user and storing it in a specified...

3 min read

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

Languages

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

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

Data Science & ML

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

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial

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

Computer Science

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

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design

Interview Preparation

Competitive Programming
Top DS or Algo for CP

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

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