Django form view

ways to send information to server[¶](https://learnbatta.com/course/django/form-view/#ways-to-send-information-to-server)

* Can send the data in HTTP headers
* Can send the data in HTTP request params
* Can send the data in HTTP request body.

What is a form?[¶](https://learnbatta.com/course/django/form-view/#what-is-a-form)

* A form is used to send the information to the server
* GET and POST are the only HTTP methods to use when dealing with forms.

Django forms[¶](https://learnbatta.com/course/django/form-view/#django-forms)

* Django provides a way to create the form fields in a pythonic way with/without form context
* Django forms validates the data when a data is submitted to the server.
* Django forms also helpful in rendering the error or validation messages if submitted data is not in the required format.

django contact form[¶](https://learnbatta.com/course/django/form-view/#django-contact-form)

* Let's create a simple django contact form.
* Create a forms.py file in the *django app* my\_app and add the below code.

my\_app/forms.py

from django import forms

class ContactForm(forms.Form):

first\_name = forms.CharField(max\_length=10)

last\_name = forms.CharField(max\_length=10)

email = forms.EmailField(max\_length=100)

django form as HTML[¶](https://learnbatta.com/course/django/form-view/#django-form-as-html)

* Can generate HTML from the above django form.
* Now, open django shell generate HTML from django form with below code.

form = ContactForm()

print(form.as\_div())

* It will generate below HTML

<div>

<label for="id\_first\_name">First name:</label>

<input type="text" name="first\_name" maxlength="10" required id="id\_first\_name">

</div>

<div>

<label for="id\_last\_name">Last name:</label>

<input type="text" name="last\_name" maxlength="10" required id="id\_last\_name">

</div>

<div>

<label for="id\_email">Email:</label>

<input type="email" name="email" maxlength="10" required id="id\_email">

</div>

* we can use same HTML if the django template just like {{ form.as\_div }}
* similarly, we can use other form methods like form.as\_p(), form.as\_ul() and form.as\_table()
* we render the template with form fields for GET request.

Django Form for data validation[¶](https://learnbatta.com/course/django/form-view/#django-form-for-data-validation)

* Django forms can be used to validate the form data as well.
* Look at the below code.

data = {}

form = ContactForm(data)

print(form.is\_valid())

# output: False

print(form.errors)

# output: <ul class="errorlist"><li>first\_name<ul class="errorlist"><li>This field is required.</li></ul></li><li>last\_name<ul class="errorlist"><li>This field is required.</li></ul></li><li>email<ul class="errorlist"><li>This field is required.</li></ul></li></ul>

print(form.errors.as\_json())

# output: {"first\_name": [{"message": "This field is required.", "code": "required"}], "last\_name": [{"message": "This field is required.", "code": "required"}], "email": [{"message": "This field is required.", "code": "required"}]}

* form.is\_valid() method is used to check whether the form data is valid or not.
* form.errors, form.errors.as\_json() are used for form errors.
* Let's use the valid data

data = {"email": "john@example.com", "first\_name": "John", "last\_name": "Doe"}

form = ContactForm(data)

print(form.is\_valid())

# output: True

print(form.cleaned\_data)

# output: {'first\_name': 'John', 'last\_name': 'Doe'}

* form.cleaned\_data returns the validated data.

Django form view[¶](https://learnbatta.com/course/django/form-view/#django-form-view)

function based form view[¶](https://learnbatta.com/course/django/form-view/#function-based-form-view)

* we can use the forms with function based views for simple functionality.
* lets write a simple form view to get the contact information.
* we will be using the ContactForm mentioned in this article.
* look at the below code

from django.http import HttpResponse

from django.shortcuts import render

from my\_app.forms import ContactForm

def my\_form\_view(request):

if request.method.lower() == "POST":

form = ContactForm(request.POST)

if form.is\_valid():

data = form.cleaned\_data

print(data)

return HttpResponse("successfully submitted the form")

else:

form = ContactForm()

context = {"form": form}

return render(request, 'form\_template.html', context)

form\_template.html

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

{% csrf\_token %}

{{ form.as\_table }}

<input type="submit" value="submit" />

</form>

class based form view[¶](https://learnbatta.com/course/django/form-view/#class-based-form-view)

from django.http import HttpResponse

from django.views.generic.edit import FormView

from my\_app.forms import ContactForm

class MyFormView(FormView):

template\_name='form\_template.html'

form\_class=ContactForm

def form\_valid(self, form):

data = form.cleaned\_data

print(data)

return HttpResponse("successfully submitted the form")

* FormView is generic class provided by django.
* It provides the generic functionality for form views.
* It comes with methods like form\_valid and form\_invalid to handle the requests.
* Only use it if your functionality matches with FormView implementation.

configure urls[¶](https://learnbatta.com/course/django/form-view/" \l "configure-urls" \o "Permanent link)

* open my\_app/urls.py and add below code to it.

from django.urls import path

from my\_app import views

urlpatterns = [

# ...

path('fbv\_form\_view', views.my\_form\_view, name='fmy\_form\_view'),

path('cbv\_form\_view', views.MyFormView.as\_view(), name='cmy\_form\_view'),

# ...

]

* Access the url <http://localhost:8000/my_app/fbv_form_view> to see the function based form view
* Access the url <http://localhost:8000/my_app/cbv_form_view> to see the class based form view
* Once open the form urls, fill out the form and submit the form if the form is valid then we can see the response "successfully submitted the form" otherise it shows the form with error messages.

References[¶](https://learnbatta.com/course/django/form-view/#references)

* <https://docs.djangoproject.com/en/dev/topics/forms/>
* <https://docs.djangoproject.com/en/dev/ref/forms/fields/>
* <https://docs.djangoproject.com/en/dev/topics/class-based-views/generic-editing/>