

**By Vitor Freitas**

I'm a passionate software developer and researcher from Brazil, currently living in Finland. I write about Python, Django and Web Development on a weekly basis.

[Read more.](#)



Get \$10 Free Credit on DigitalOcean to Get Started.

ads via Carbon

TUTORIAL

How to Use Bootstrap 4 Forms With Django

📅 Aug 13, 2018 ⌚ 7 minutes read 💬 33 comments 👁 111,958 views



(Picture: <https://www.pexels.com/photo/people-apple-desk-technology-89724/>)

This is a quick tutorial to get you start with [django-crispy-forms](#) and never look back. Crispy-forms is a great application that gives you control over how you render Django forms, without breaking the default behavior. This tutorial is going to be tailored towards Bootstrap 4, but it can also be used with older Bootstrap versions as well as with the [Foundation framework](#).

The main reason why I like to use it on my projects is because you can simply render a Django form using `` and it will be nicely rendered with Bootstrap 4, with very minimal setup. It's a really life saver.

Installation

Install it using pip:

```
pip install django-crispy-forms
```

Add it to your `INSTALLED_APPS` and select which styles to use:

settings.py

```
INSTALLED_APPS = [  
    ...  
    'crispy_forms',  
]  
  
CRISPY_TEMPLATE_PACK = 'bootstrap4'
```

Setup Bootstrap

You can either download the latest Bootstrap 4 version at [getbootstrap.com](#). In that case, go to download page and get the **Compiled CSS and JS** version.

Or you can use the hosted Bootstrap CDN:

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js">
```

For simplicity, I will be using the CDN version. Here is my **base.html** template that will be referenced in the following examples:

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
    <title>Django People</title>
  </head>
  <body>
    <div class="container">
      <div class="row justify-content-center">
        <div class="col-8">
          <h1 class="mt-2">Django People</h1>
          <hr class="mt-0 mb-4">
          {% block content %}
          {% endblock %}
        </div>
      </div>
    </div>
  </body>
</html>
```

I only added the CSS file because we won't be using any JavaScript feature.

Basic Usage

Suppose we have a model named `Person` as follows:

models.py

```
from django.db import models

class Person(models.Model):
```

```
name = models.CharField(max_length=130)
email = models.EmailField(blank=True)
job_title = models.CharField(max_length=30, blank=True)
bio = models.TextField(blank=True)
```

Let's say we wanted to create a view to add new `Person` objects. In that case we could use the built-in `CreateView`:

views.py

```
from django.views.generic import CreateView
from .models import Person

class PersonCreateView(CreateView):
    model = Person
    fields = ('name', 'email', 'job_title', 'bio')
```

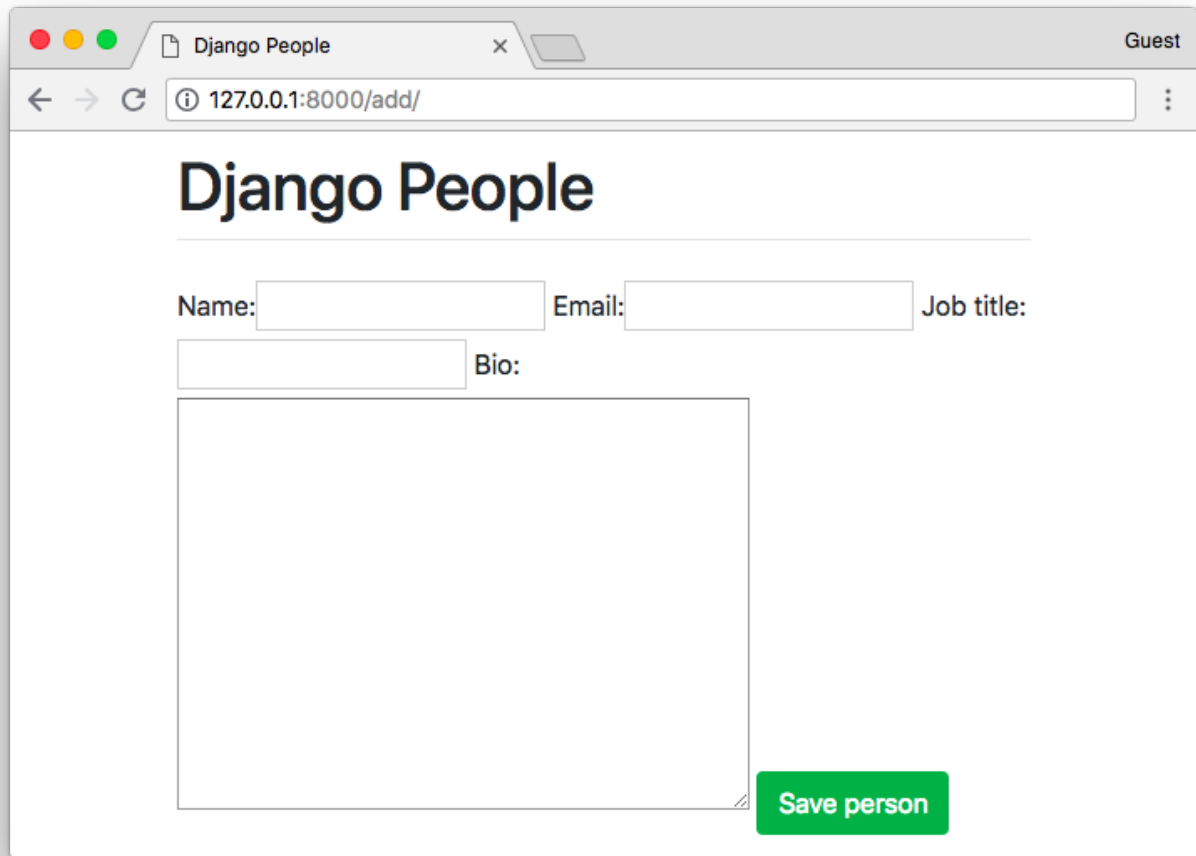
Without any further change, Django will try to use a template named `people/person_form.html`. In that case "people" is the name of my Django app:

people/person_form.html

```
{% extends 'base.html' %}

{% block content %}
<form method="post">
    {% csrf_token %}
    {{ form }}
    <button type="submit" class="btn btn-success">Save person</button>
</form>
{% endblock %}
```

This is a very basic form rendering, and as it is, Django will render it like this, with no style, just plain form fields:



The screenshot shows a web browser window with the title 'Django People' and a tab icon. The address bar shows '127.0.0.1:8000/add/'. The page content includes a large heading 'Django People'. Below the heading, there are four form fields: 'Name:', 'Email:', 'Job title:', and 'Bio:'. The 'Bio:' field is a large text area. A green button labeled 'Save person' is located at the bottom right of the form.

To render the same form using Bootstrap 4 CSS classes you can do the following:

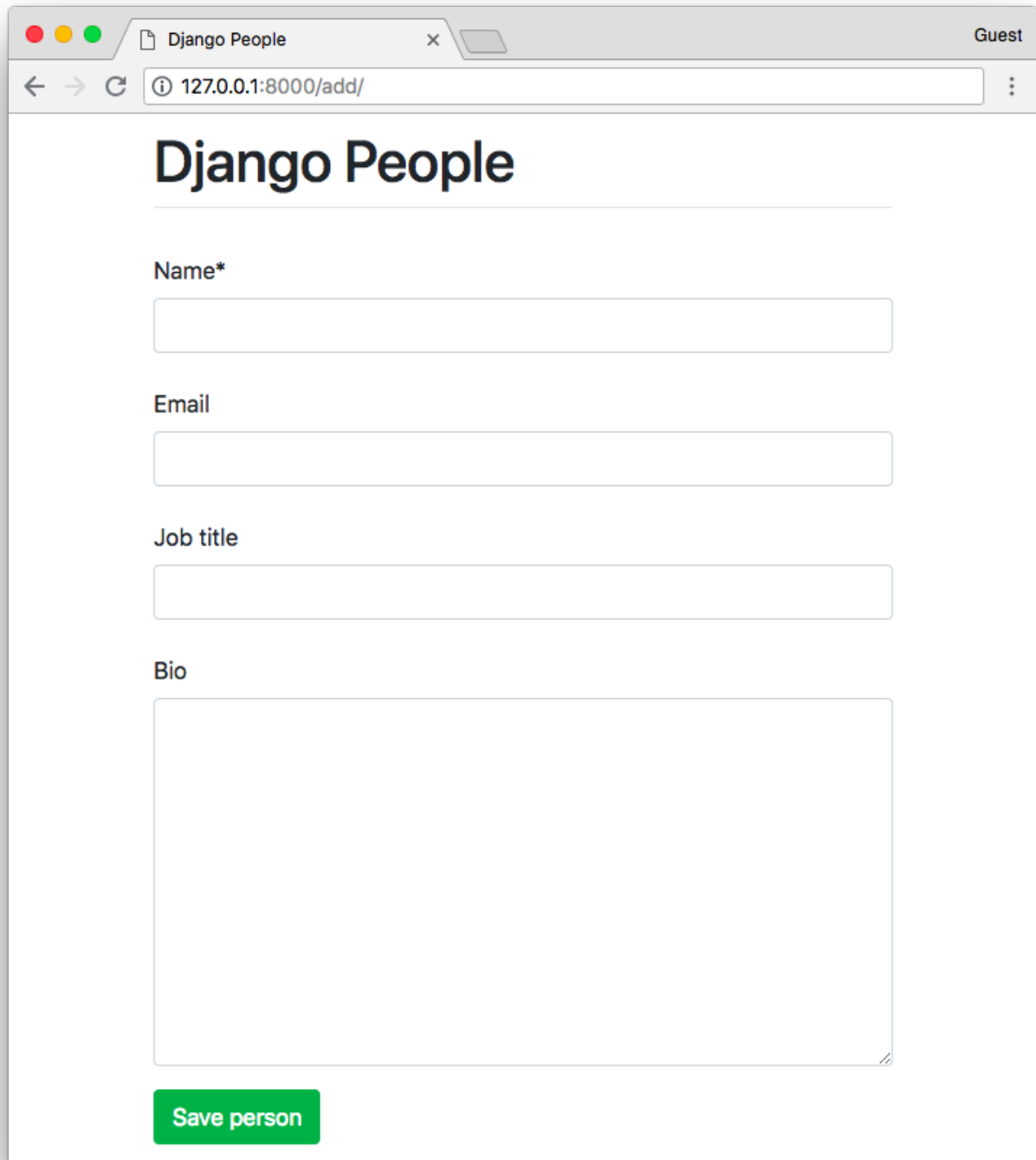
people/person_form.html

```
{% extends 'base.html' %}

{% load crispy_forms_tags %}

{% block content %}
    <form method="post" novalidate>
        {% csrf_token %}
        {{ form|crispy }}
        <button type="submit" class="btn btn-success">Save person</button>
    </form>
{% endblock %}
```

Now the result, much better:



The screenshot shows a web browser window with the title 'Django People' and a tab icon. The address bar shows '127.0.0.1:8000/add/'. The page content includes a large heading 'Django People' followed by four form fields: 'Name*' (a single-line text input), 'Email' (a single-line text input), 'Job title' (a single-line text input), and 'Bio' (a multi-line text area). At the bottom of the form is a green button labeled 'Save person'.

There are some cases where you may want more freedom to render your fields. You can do so by rendering the fields manually and using the `as_crispy_field` template filter:

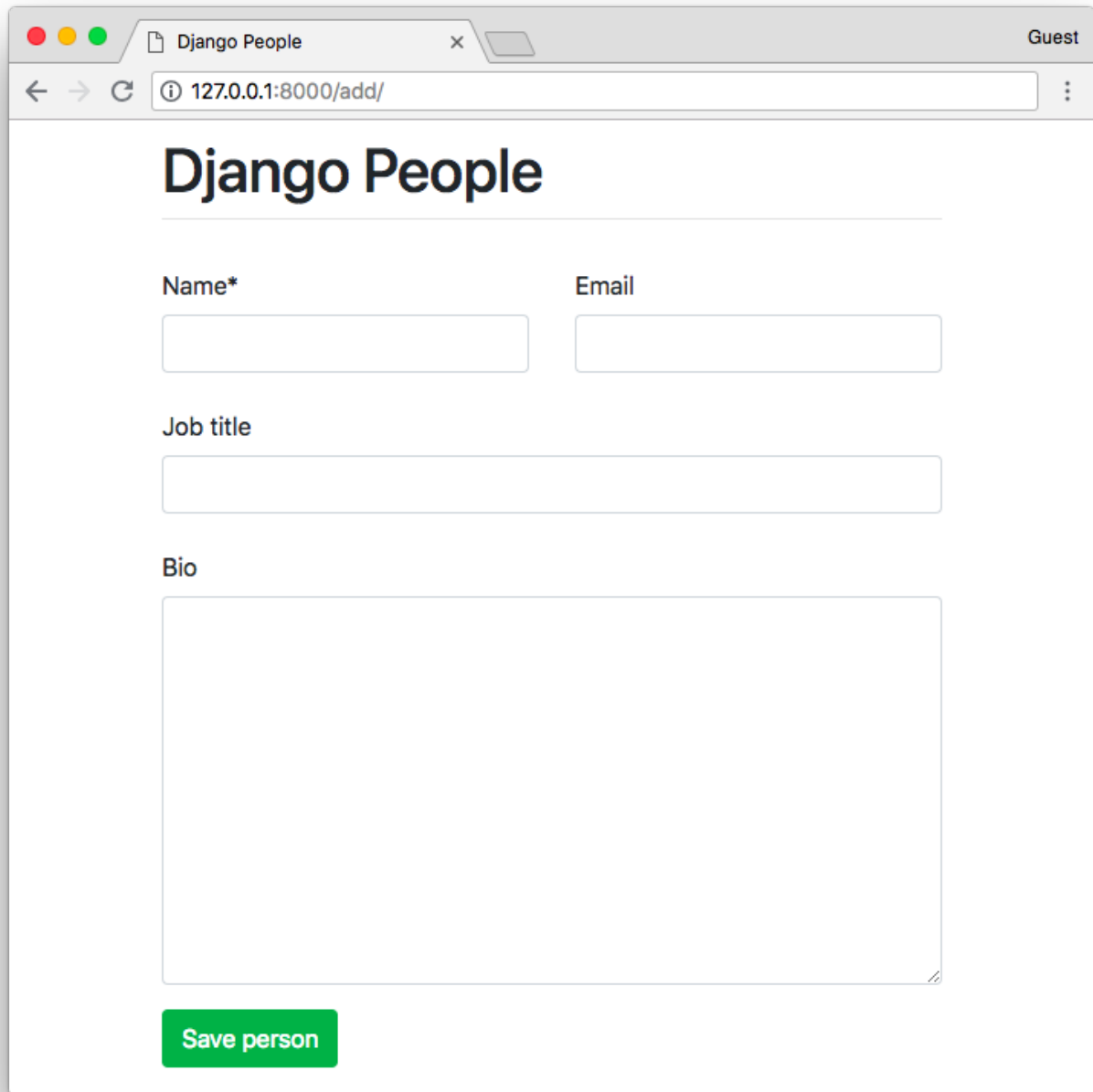
```
{% extends 'base.html' %}

{% load crispy_forms_tags %}

**people/person_form.html**
```

```
{% block content %}
<form method="post" novalidate>
  {% csrf_token %}
  <div class="row">
    <div class="col-6">
      {{ form.name|as_crispy_field }}
    </div>
    <div class="col-6">
      {{ form.email|as_crispy_field }}
    </div>
  </div>
  {{ form.job_title|as_crispy_field }}
  {{ form.bio|as_crispy_field }}
  <button type="submit" class="btn btn-success">Save person</button>
</form>
{% endblock %}
```

And the result is something like the screen shot below:



The screenshot shows a web browser window with the title 'Django People' and a tab icon. The address bar shows '127.0.0.1:8000/add/'. The page content is a form titled 'Django People' with a horizontal line below the title. The form contains four fields: 'Name*' (a text input), 'Email' (a text input), 'Job title' (a text input), and 'Bio' (a large text area). At the bottom of the form is a green button labeled 'Save person'.

Form Helpers

The django-crispy-forms app has a special class named `FormHelper` to make your life easier and to give you complete control over how you want to render your forms.

Here is an example of an update view:

forms.py


```
from django import forms
from crispy_forms.helper import FormHelper
from crispy_forms.layout import Submit
from people.models import Person

class PersonForm(forms.ModelForm):
    class Meta:
        model = Person
        fields = ('name', 'email', 'job_title', 'bio')

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.helper = FormHelper()
        self.helper.form_method = 'post'
        self.helper.add_input(Submit('submit', 'Save person'))
```

The job is done inside the `__init__()` method. The rest is just a regular Django model form. Here I'm defining that this form should handle the request using the **POST** method and the form should have an submit button with label "Save person".

Now our view, just regular Django code:

views.py

```
from django.views.generic import UpdateView
from people.models import Person
from people.forms import PersonForm

class PersonUpdateView(UpdateView):
    model = Person
    form_class = PersonForm
    template_name = 'people/person_update_form.html'
```

Then in our template:

people/person_update_form.html

```
{% extends 'base.html' %}

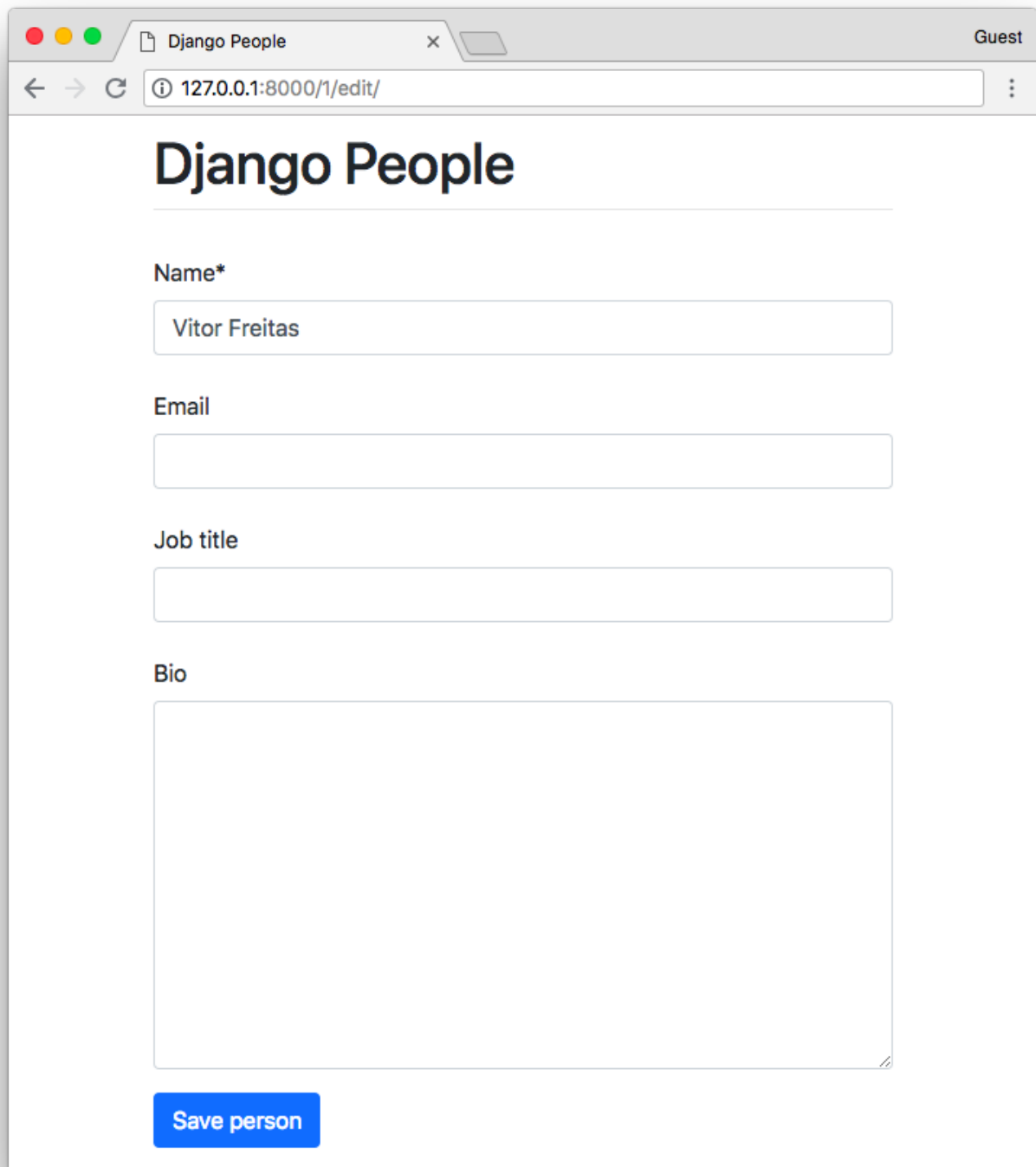
{% load crispy_forms_tags %}

{% block content %}
```

```
{% crispy form %}  
{% endblock %}
```

Here we can simply call the `{% crispy %}` template tag and pass our form instance as parameter.

And that's all you need to render the form:



The screenshot shows a web browser window with the title 'Django People' and a guest user. The address bar shows '127.0.0.1:8000/1/edit/'. The page content includes a heading 'Django People' followed by four form fields: 'Name*' (containing 'Vitor Freitas'), 'Email', 'Job title', and 'Bio' (a large text area). A blue 'Save person' button is located at the bottom of the form.

Conclusions

That's pretty much it for the basics. Honestly that's about all that I use. Usually I don't even go for the `FormHelper` objects. But there are much more about it. If you are interested, you can check their official documentation: django-crispy-forms.readthedocs.io.

If you are not sure about where you should create a certain file, or want to explore the sample project I created for this tutorial, you can grab the source code on GitHub at github.com/sibtc/bootstrap-forms-example.

Related Posts



[How to Use Date Picker with Django](#)



[How to Implement Grouped Model Choice Field](#)



[Advanced Form Rendering with Django Crispy Forms](#)

[django](#)[forms](#)[bootstrap](#)[bs4](#)[crispy-forms](#)

Share this post



33 Comments

Simple is Better Than Complex

 Login ▾

 Recommend 4

 Tweet

 Share

Sort by Best ▾



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS 

Name



Paul • a year ago

Best django blog I've ever seen . Great job Vitor.

9 ^ | v • Reply • Share >



Adam Bishop • a year ago

Ok, cool, but what about [urls.py](#), how do I register class Person(CreateView), because I have no idea, and my Django server also and why Your form is under address 127.0.0.1:8000/add, not under /person_form? I stuck in basic view version.

2 ^ | v • Reply • Share >



duub qnnp ➔ Adam Bishop • 8 months ago • edited

in this tutorial it's not clear how he manage the url paths... but in the source code you can see that it's not like he says "Without any further change, Django will try to use a template named" here the link to the [urls.py](#) file:

<https://github.com/sibtc/bo...>

thanks for the tutorial Vitor!

^ | v • Reply • Share >



Adriano Silva • a year ago

Obrigado Vitor. Ótimo artigo.

2 ^ | v • Reply • Share >



real mah47 • 3 months ago • edited

Thank you ^-^

1 ^ | v • Reply • Share >



A. N. • a year ago

Actually can you have an additional blog on just Form Helpers. I also understand one can programmatically build a form in code, how to do that?

1 ^ | v • Reply • Share >



Zoid ➔ A. N. • a year ago

Hey, I hope this example is useful: <https://github.com/django-c...>

^ | v • Reply • Share >



Harry Moreno • 17 days ago

Can we get a followup that covers bootstrap 4 theming and auto reload in django? would be great.

^ | v • Reply • Share >



Thakib Kayodé Adéchinan Salami • a month ago

thanks

^ | v • Reply • Share >



Mohammad Shahnawaz • a month ago



Hello,

I am new for the Anaconda,Python, and Django I am getting error while installing the django-crispy-form i am getting following error.

Solving environment: failed with initial frozen solve. Retrying with flexible solve.

Please help me out from this situation. I explore the google but, i can't find out the solution yet.

Your first response make me pleased

^ | v • Reply • Share >



Messi • 6 months ago

Please you can make a tutorial for fluent_comments.?

^ | v • Reply • Share >



Karan Negi • 6 months ago • edited

the way you explain things reflects the meaning of "**simple is better than complex**"

^ | v • Reply • Share >



InfiltrateThem • 8 months ago • edited

I can tell you I don't have money. But what I do have are a very particular set of skills; skills I have acquired over a very long career. Skills that make me thank people like you. If you let me find you , that'll be the end of it, I'll give you a big massive kiss!.

^ | v • Reply • Share >



Kimberly Bird • 8 months ago

Third or fourth time your blog has helped me! Thank you!

^ | v • Reply • Share >



Mostafa Gafer • 9 months ago

Hi I have an issue with it can anyone help me with it . I asked aquestion on <https://stackoverflow.com/q...> and I would be grateful if any one gave me a hand

^ | v • Reply • Share >



Mostafa Gafer ➔ Mostafa Gafer • 9 months ago

The bootstrap 4.2.1 is not working with crispy 1.7.2 and django 2.1.5

^ | v • Reply • Share >



Ananda Bayu • a year ago

anyone can help me?

I try to add this code in [forms.py](#), it works for update form but not for add form

```
self.helper.layout = Layout(
    Field('name', placeholder='Your name', css_class="is-invalid"),
)
```

^ | v • Reply • Share >



Ananda Bayu • a year ago

How to use another class from bootstrap template?

like free template from internet.

^ | v • Reply • Share >

  • Reply • Share >**ardin** • a year ago

Hi Vitor, it's great blog and tutorial. Thank you.

I'm new in django, so can anyone tell me is django-crispy-forms better than django-bootstrap4 ? And what with django-widget-tweak? When I should use which one? I would learn good habits..

  • Reply • Share >**Zoid** • a year ago

Awesome job. Nice post, Vitor.

  • Reply • Share >**Muhammad Habib Mobolaji** • a year ago

Can i use crispy_forms for any other frontend framework other than bootstrap like materializecss

  • Reply • Share >**Zoid** → Muhammad Habib Mobolaji • a year ago

There is a library that uses crispy forms and Semantic UI, and you could totally make one that uses Materialize CSS: <https://github.com/acidjunk...>

  • Reply • Share >**AlmostAkuma665** • a year ago • edited

Thank you for the wonderful tutorial but the documentation is hard to read... so let me ask a question. How to render the form without labels?

EDIT: nevermind, I just need to search and I found the answer

  • Reply • Share >**JJ** • a year ago • edited

Vitor, thanks for your great work! Yes, the best django blog.

  • Reply • Share >**Ian** • a year ago

I tried crispy-forms over the last few days and it seemed like a miracle, but the bootstrap 4 support seems patchy and very tricky to get working. It kept inserting rows and other classes where they weren't needed which messed up a lot of formatting, I really hope they fix these issues as its the only thing holding me back using it.

  • Reply • Share >**Zoid** → Ian • a year ago

Hi [@Ian](#) . We've tried to improve support for Bootstrap 4, but please feel free to make any issues so we could track fixing and improving the Bootstrap 4 support.

  • Reply • Share >**Marcelo Ribeiro da Silva** • a year ago

This blog was a jewel found on the internet.

  • Reply • Share >



Oleg Kleschynov • a year ago

Hey, Vitor. Would be great to see a review on your blog about you 'colossus' project.

^ | v • Reply • Share >



Vitor Freitas Mod ➔ Oleg Kleschynov • a year ago

Thanks Oleg! I'm working on it :-)

1 ^ | v • Reply • Share >



Keania Ledum • a year ago

Thanks Vitor...great tutorial

^ | v • Reply • Share >



Shahmir Alam • a year ago

love your work <3

^ | v • Reply • Share >



Leonardo Henrique Steil • a year ago

This is the best django blog in the web.
I've learned a lot here.

Thank you so much.

^ | v • Reply • Share >



David Johns • a year ago

Hi Vitor, A superb tutorial. I haven't done much django for quite a few months but reading this tutorial has rekindled my interest. Thank you.

^ | v • Reply • Share >

ALSO ON SIMPLE IS BETTER THAN COMPLEX

Django Authentication Video Tutorial

11 comments • a year ago



Vitor Freitas — Next video coming up I will be adding bootstrap 4 to the forms :D

How to Use JWT Authentication with

How to Create Custom Django Management Commands

17 comments • a year ago



Vitor Freitas — Thanks buddy :-)

How to Use Celery and RabbitMQ with

✈ Subscribe to our Mailing List

Receive updates from the Blog!

Popular Posts



[How to Extend Django User Model](#)



[How to Setup a SSL Certificate on Nginx for a Django Application](#)



[How to Deploy a Django Application to Digital Ocean](#)

© 2015-2019 simple is better than complex cc by-nc-sa 3.0 // [about](#) [contact](#) [faq](#) [cookies](#) [privacy](#)
[policy](#)