Django Markdown Tutorial

Markdown is a popular text-to-HTML conversion tool for web writers. It is far easier to use than plain old HTML. Many/most static site generators provide a built-in way to write posts using Markdown, however adding it to a Django website–such as a blog–takes an extra step or two. In this tutorial, I'll demonstrate how to quickly add Markdown functionality to any Django website.

This post presumes knowledge of Django and how to build a blog. If you need a detailed overview of the process, check out my book Django for Beginners which walks through building 5 progressively more complex web apps, including a blog!

But for now, let's start on the command line with the usual command to install Django, create a new project called config, setup the initial database using `config`, and starting the local webserver using the runserver command. I've added steps to install this code on the Desktop, which is convenient for Mac users, but the code directory can live anywhere on your computer.

```
$ cd ~/Desktop
$ mkdir markdown && cd markdown
$ pipenv install django~=3.1.0
$ pipenv shell
(markdown) $ django-admin startproject config .
(markdown) $ python manage.py migrate
(markdown) $ python manage.py runserver
```

Update the `config/urls.py` file to include our blog app which will have the URL path of '', the empty string.

```
# config/urls.py
from django.contrib import admin
from django.urls import path, include # new

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blog.urls')), # new
]
```

Models, URLs, Views, Templates At this point we need four updates within our blog app:

models.py for the database model

- urls.py for the URL route
- views.py for our logic
- `templates/post_list.html` for our template

The order in which we make these changes does not matter; we need all of them before the functionality will work. However, in practice, starting with models, then URLS, then views, and finally templates is the approach I usually take.

Start with blog/models.py.

```
# blog/models.py
from django.db import models


class Post(models.Model):
    title = models.CharField(max_length=200)
    body = models.TextField()

    def __str__(self):
        return self.title
```