پایتون از پایه : ساخت وبسایت ایستا

لینک مطلب :

http://code.tutsplus.com/articles/python-from-scratch-create-a-dynamic-website--net-22787

ما کمی از پایتون رو در آموزش های قبلی مرور کردیم . امروز ، میریم هر چیزی رو که یاد گرفتیم ادغام کنیم و یک وبسایت ایستا بسازیم .

پس ، چطور برای ساخت وبسایت با پایتون شروع کنیم ؟ خوب ، همش رو خودتون میتونید انجام بدید ، و یک برنامه بنویسید ک در سرور های وب اجرا بشه ، و تمام درخواست های صفحه و جواب هایی با فرم اچ تی ام ال و منابع دیگه رو قبول کنه . اگر چه ، این کار خیلی زیادی هستش ، پس چرا این همه کار رو انجام بدیم وقتی که ابزار هایی برای این کار هستند ؟ این ابزار فرم ورک نام دارند ، و این چیز است که برای ساخت وبسایتمون استفاده میکنیم .

فریم ورک های پایتون

تعداد کمی برای پایتون فرم ورک هست ، اما در زیر چند تا از بهترین ها رو ذکر میکنیم :

جنگو – ما امروز از این استفاده میکنیم . جنگو خصوصیات زیادی داره ، اما کاراییش آسونه . مستنداتش هم همچنین عالی هستش ، پس اگر گیر کردید ، به راحتی میتونید مشکلاتتون رو حل کنید .

گروک – یک فرم ورک دیگر با دسته ای از خصوصیات نزدیک به جنگو . اگر تصمیم گرفتید از جنگو استفاده نکنید ، این جایگزین خوبی است .

وب پای – یک فرم ورک خیلی سبک . این رم ورک خصوصیات زیادی نداره ، هر چند یک وقتی داشت !

توربو گیرز – اگر چه قبلا مستنداتش خیلی بد بود ، به صورت چشم گیری در سال اخیر پیشرفت کرده و بهتر شده .

لیست جامع تری در سایت پایتون قابل مشاهده است برای گزینه های بیشتر . امروز ما جنگو رو برای یک ماشین محلی نصب میکنیم ، و یک وبلاگ ساده میسازیم . ما همچنین مروری بر نصب آن روی یک وب سرور میکنیم .

نصب جنگو

ما بیشتر کار هامون رو در ترمینال انجام میدهیم . و این کاملا باید در مک و لینوکس کار کنه . اگر چه ، اگر ویندوز استفاده میکنید ، کار ها کمی متفاوت هستند . آشنایی با خط فرمان ضروری نیست اگر شما فقط پایتون مینویسید ، اگرچه ، اگر قصط استفاده از جنگو رو دارید ، یا ساخت یک وبسایت ایستا ، یاد گیری خط فرمان با ارزش هستش .

آموزش های ترمینال

این آموزش ها رو برای پیشروی در ترمینال و یادگیری آن انجام دهید . (دل به خواه)

.مقدمه ای برای استفاده از ترمینال که شامل گیت و گیت هاب هم میشه

.10 نکته برای کارایی سریع تر در ترمینال

در زیر فرمان هایی که برای نصب جنگو نیاز دارید هستند . اگر با پایتون 3 مطابق نیستند ، پس باید پایتون نسخه 2.7 یا کمتر رو برای اجرا نصب کنید .

```
wget http://www.djangoproject.com/download/1.3.1/tarball/
tar xzvf Django-1.3.1.tar.gz
cd Django-1.3.1
python setup.py install
```

بعد ، میتونید فایل های نصب رو پاک کنید .

```
cd ..
rm Django-1.3.1.tar.gz
```

این باید بس باشه ! بیایید آزمایشش کنیم .

```
python
from django import get_version
get_version()
```

شما باید 1.3.1 رو ببینید ، همه چیز کار میکنه و جنگو روی سیستم شما نصب هستش .

تبریک ! ما آماده ساخت وبسایت خود هستیم !

ساخت وبسایتمون

امروز میریم که یک سیستم بلاگ بسازیم ، چون راه خوبی برای یادگیری مقدمات هستش . اول ، ما باید یک پروژه جنگو بسازیم .

```
cd ~/Documents/Projects
django-admin.py startproject FirstBlog
cd FirstBlog
ls
```

هر کدوم از این فایل ها چیکار میکنند ؟

. __init__.py

فایل بالا که نوشتم به پایتون میگه که این پوشه یک پکیج پایتون هستش . ما درباره این چیزا در درس سوم یاد گرفتیم . این به پایتون اجازه وارد کردن تمام اسکریپت های اون پوشه رو به عنوان ماژول میده .

. manage.py

فایل بالا قسمتی از وبسایت شما نیست . این یک اسکریپت کمکی هستش که شما از طریق خط فرمان اجراش میکنید . این فایل دارای زنجیره ای از توابع برای مدیریت سایتتون هست .

. settings.py

فایل بالا دارای تنظیمات سایت شماست . جنگو از فایل های xml برای تنظیمات استفاده نمیکنه . همه چیز پایتون هستش . این فایل تعدادی از متغیر هاست برای معرفی تنظیمات سایت شما .

. urls.py

این فایل لینک های صفحه رو مشخص میکنه . برای مثال ، میتونه yourwebsite.com/about رو به صفحه **درباره** سایت شما هدایت کنه .

جنگو به خودش میگه فرم ورک MTV که مخفف عبارت Model Template View میشه .

برنامه ها

اگر چه هیچ کدوم از این فایل ها به تنهایی یک وبسایت کاربردی رو نمیتنه بسازه . برای همین ، ما به برنامه ها احتیاج داریم . برنامه ها جایی هستند که ما کد هامون رو مینویسیم تا وبسایتمون کار کنه ، اما قبل از این که نگاهی به آن ها بیاندازیم ، ما باید کمی به شناخت ساختار جنگو بپردازیم .

اول ، جنگو یک فرم ورک MVC هستش که مخفف عبارت Model View Controller هست . جنگو به خودش میگه فرم ورک MTV که مخفف عبارت Model View Template هست . تقریبا با MVC فرق داره ولی در پایه به م شبیه هستند . پس ، MVC یک پترن مهندسی

هستش که یک متد رو برای مرتب سازی پروژه های شما ارائه میده . این کدی رو که برای پروسه داده ها هستش رو با کدی که برای مدیریت محیط کاربری استفاده میشه رو جدا میکنه .

جنگو به DRY معروفه یا Don't Repeat Yourself .

دوم ، جنگو به DRY یا Don't Repeat Yourself معروفه ، که یعنی شما نباید کدی رو که یک کاری رو انجام بده بیشتر از یکبار بنویسید . برای مثال ، در وبسایت ما ، اگر ما یک خصوصیتی رو که یک مقاله تصادفی از آرشیو برداره رو بنویسیم ، و از این خصیصه در صفحه های مختلف استفاده کنیم ، ما هر وقت که بهش لازم داشتیم دیگه دوباره اون رو نمی نویسیم . ما یک بار کدش رو می نویسیم و در هر صفحه استفاده میکنیم .

پس این چطور به برنامه ها مربوط میشه ؟ خوب ، برنامه ها به شما امکان نوشتن وبسایتتون رو به صورت DRY میده . هر پروژه ، مثل اینی که ما داریم ، میتونه دارای چندین برنامه باشه . و همین طور هر برنامه میتونه قسمتی از چندین پروژه باشه . با استفاده از مثال قبلی ، این یعنی اینکه اگر ما یک وبسایت دیگه میخواستیم بسازیم که همون خصیصه صفحه تصادفی (که در بالا ذکر شد) رو بخواد ، ما نباید دوباره اون خصیصه رو بنویسیم . ما به راحتی میتونیم برنامه رو از پروژه دیگر وارد کنیم . برای همین ، این مهمه که هر برنامه کار مخصوص به خودش رو انجام بده . اگر شما تمام کار های وبسایتتون رو در یک برنامه بنویسید ، و بعدا قسمتی از اون رو جایی دیگه خواستید استفاده کنید ، باید همش رو وارد کنید . اگر شما یک وبسایت e-Commerce خواستید بسازید برای مثال ، شما نمیخواهید تمام خصوصیات این وبسایت رو وارد کنید . اگرچه ، اگر یک برنامه برای خصوصیت تصادفی و یک برنامه برای سیستم انتشار سایت بسازید شما میتونید قسمت هایی رو که لازم دارید انتخاب و دوباره استفاده کنید .

این همچنین یعنی که با سایت ، کد کاملا مرتب هستش . اگر خواستید یک خصیصه رو تغییر دهید ، دیگه لازم نیست که در یک فایل حجیم جست و جو کنید ، شما به جای اینکار میتونید برنامه مورد نظر این خصیصه رو تغییر داده بدون نگرانی درباره خرابی قسمت های دیگه سایت .

```
python mangage.py startapp blog
cd blog
ls
```

دوباره ، ما یک –init.py– داریم تا به پکیج تبدیلش کنه ، و سه فایل دیگر ، tests ، models و views . ما به فایل tests فعلا کاری نداریم ، اما دوتای دیه مهم اند . Models و views دو

قسمت از MVC رو تشکیل میدن .

در models ما ساختار داده هایمان رو تعریف میکنیم .

اگر قبلا با php کار کرده باشید ، شاید با phpmyAdmin برای ساخت جدول MySQL خود استفاده کرده باشید ، و بعد نوشتن کوئری های اس کیو ال خود به صورت دستی در اسکریپت های PHP . در جنگو ، خیلی راحت ترید . ما تمام ساختار داده های خود را در این فایل models تعریف میکنیم ، بعد یک فرمان رو اجرا کرده و تمام دیتابیس های مورد نیاز برای ما ساخته میشن .

وقتی خواستید به آن داده دسترسی داشته باشید ، شما با این مدل ها و صدا زدن متد روی آن ها کار میکنید ، به جای اجرای کوئری های خام . این خیلی کمک میکنه ، چون جنگو میتونه چندین برنامه دیتابیس رو استفاده کنه . ما از MySQL در امروز استفاده میکنیم ، چون قوی ترین هستش ، و بیشتر هاست ها ارائه میدنش . اما اگر خواستیم که به یک دیتابیس دیگه بریم در آینده ، همه ی کد های ما هنوز هم درست هستن . در زبان های دیگر ، اگر بخواهید به SQLite یا یکی دیگه شبیه آن تغییر دهید ، باید دوباره کدی رو برای دسترسی به دیتابیس بنویسید .

در فایل ویو (views) کدی هست که صفحات وب رو میسازه . این تمام قسمت های دیگر رو به هم متصل میکنه . وقتی کاربر یک URL رو تایپ میکنه ، با اسکریپت urls که قبلا دیدیم به اسکریپت views فرستاده میشه ، که بعدا داده های مرتبط رو از مدل (models) میگیره ، اون رو پروسه میکنه و بعد میفرسته به template ، که منجر به صفحه ای که کاربر میبینه میشه . کمی بعد نگاهی به template ها می اندازیم . اون ها آسون ترین قسمت هستند — مخصوصا HTML .

برای بلاگ ، ما به یک جدول پُست ها نیاز داریم ، با چندین فیلد برای تیتر ، متن بدنه ، نویسنده ، زمانی که نوشته شده ، و غیره . یک بلاگ واقعی قسمت نظرات داره ، اما این رو بهتون آموزش نمیدیم و فراتر از آموزش امروز هست .

```
from django.db import models

class posts(models.Model):
    author = models.CharField(max_length = 30)
    title = models.CharField(max_length = 100)
    bodytext = models.TextField()
    timestamp = models.DateTimeField()
```

MySQL

این مدل ها فقط یک توضیحات هستند . ما باید یک دیتابیس واقعی ازشون بسازیم . اول ، اگر چه ، نیاز به اجرای MySQL روی سیستم داریم . روی یک وب سرور واقعی ، این مشکلی نیست ، چون اون ها از قبل نصب دارندش . خوشبختانه ، با مدیریت پکیج ، نصب اون آسونه . اول ، باید Homebrew و Eazy install رو نصب کنید .

```
brew install mysql
easy_install mysql-python

mysqld_safe --skip-grant-tables #let anyone have full permissions
mysql -u root
UPDATE mysql.user SET Password=PASSWORD('nettuts') WHERE User='root'; #give the
user 'root' a password
FLUSH PRIVILEGES;

mysql -u root -p #log in with our password 'nettuts'
CREATE DATABASE firstblog;
quit

python2.6 manage.py runserver
```

وقتی سیستم رو دوباره راه اندازی میکنید ، MySQL اجرا نمیشه ، پس هر وقت در آینده خواستید این کار رو انجام دهید ، mysqld رو اجرا کنید تا سرور شورع به کار کنه . شما بعدش میتونید python2.6 manage.py runserver رو روی یک تب دیگه اجرا کنید تا توسعه سرور شروع بشه .

این فرمان هنوز سرور رو اجرا نمیکنه ، فقط یک اررور میده . این به دلیل اینه که ما باید تنظیماتمون رو مشخص کنیم . بیایید نگاهی به settings.py بیاندازیم .

باید اول تنظیمات دیتابیسرو تغییر دهیم . این از خط ۱۲ شروع میشه .

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql', # Add 'postgresql_psycopg2',
'postgresql', 'mysql', 'sqlite3' or 'oracle'.
        'NAME': 'firstblog',                  # Or path to database file if using sqlite3.
        'USER': 'root',                       # Not used with sqlite3.
```

```
            'PASSWORD': 'nettuts',              # Not used with sqlite3.
            'HOST': '',                 # Set to empty string for localhost. Not used with
sqlite3.
            'PORT': '',                 # Set to empty string for default. Not used with sqlite3.
        }
}
```

اگر سعی در اجرا دوباره سرور کنید ، باید کار کنه ، شما MySQL رو با موفقیت نصب کردید .

اگر 127.0.0.1.8000 رو در مرورگر وب خود دیدید ، شما باید صفحه پیشفرض جنگو رو مشاهده کنید .

حالا بیایید سایت جنگو خود رو به یک بلاگ تبدیل کنیم . اول ، باید از مدل هامون برای ساخت جدول در دیتابیس استفاده کنیم با اجرای فرمان زیر :

```
python2.6 manage.py syncdb
```

هر وقت شما مدل هاتون رو تغییر میدید ، شما باید این فرمان رو برای بروز رسانی دیتابیس اجرا کنید . توجه کنید که این نمیتونه فیلد های موجود رو تغییر بده ، شاید فقط فیلد های جدید اضافه کنه . پس اگه میخواهید فیلد ها رو پاک کنید ، شما باید این کار رو به صورت دستی با چیزی مثل PhpMyAdmin انجام بدید . چون این اولین بازی است که فرمان رو اجرا میکنیم ، جنگو ساختمان پیشفرض رو در جدول ها برای چیز هایی مثل مدیریت سیستم نصب میکنه . فقط yes را تایپ کرده و مشخصات خود رو وارد کنید .

حال بیایید فایل urls.py را نصب کنیم . اولین خط رو در قسمت مثال ها از حالت کامنت در بیارید ، و به این تغییرش دهید :

```
url(r'^$', 'FirstBlog.blog.views.home', name='home')
```

بیایید فایل views رو برای جواب گویی به این درخواست ها بسازید :

```
1  from django.shortcuts import render_to_response
2
3  from blog.models import posts
4
5  def home(request):
6      return render_to_response('index.html')
```

Templates

این index.html هنوز وجود نداره . پس بیایید بسازیمش . یک پوشه بسازید با نام templates در برنامه بلاگ و یک فایل درش ذخیره کنید به نام index.html ، که میتونه برای حال فقط یک "Hello World" رو شامل بشه . بعد ، باید فایل settings رو ویرایش کنیم تا جنگو بفهمه که این template کجا قرار داره .

خط 105 قسمتی هستش برای مشخص کردن پوشه ها پس بیایید مثل زیر تنظیمش کنیم :

```
TEMPLATE_DIRS = (
	"blog/templates",
	# Put strings here, like "/home/html/django_templates" or
"C:/www/django/templates".
	# Always use forward slashes, even on Windows.
	# Don't forget to use absolute paths, not relative paths.
)
```

اگر دوباره سرور رو اجرا کنید و صفحه رو در مرورگر خود رفرش کنید ، شما باید پیام "Hello world" رو مشاهده کنید . ما الان میتونیم بلاگ خودمون رو چیدمان بندی کنیم . ما چندتا boilerplate HTML برای صفحه خانه اضافه میکنیم .

```
<!DOCTYPE html>

<html lang="en">

<head>
	<meta charset="utf-8" />
	<link rel="stylesheet" href="css/style.css">
	<link href="images/favicon.ico" rel="shortcut icon">
	<title>First Blog</title>
</head>

<body>

<div class="container">
	<h1>First Blog</h1>
	<h2>Title</h2>
	<h3>Posted on date by author</h3>
	<p>Body Text</p>

</div>

</body>

</html>
```

اگر ذخیره و رفرش کنید ، میتونید ببینید که صفحه با این محتوای جدید جایگزین شده . قدم بعدی اضافه کردن محتوا ایستا از دیتابیس هست . برای این کار ، جنگو یک زبان تمپلیتینگ داره که به شما اجازه میده متغییر ها رو در آکولاد بنویسید . قسمت وسط صفحه خود رو عوض کنی تا مثل کد زیر بشه :

```
<div class="container">
	<h1>First Blog</h1>
	<h2>{{ title }}</h2>
	<h3>Posted on {{ date }} by {{ author }}</h3>
```

```
	<p>{{ body }}</p>

</div>
```

ما بعد میتونیم مقدار هایی رو در این {} ها که نگه دارنده جای متغیر ها هستند از فایل views.py وارد کنیم با ساخت دیکشنری از مقدار ها .

```
from django.shortcuts import render_to_response

from blog.models import posts

def home(request):
	content = {
		'title' : 'My First Post',
		'author' : 'Giles',
		'date' : '18th September 2011',
		'body' : 'Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam cursus tempus dui, ut vulputate nisl eleifend eget. Aenean justo felis, dapibus quis vulputate at, porta et dolor. Praesent enim libero, malesuada nec vestibulum vitae, fermentum nec ligula. Etiam eget convallis turpis. Donec non sem justo.',
	}
	return render_to_response('index.html', content)
```

ذخیره و رفرش ، و الان باید ببینید که شما محتوا رو به تمپلیت از فایل ویو خود وارد میکنید .

قدم آخر گرفتن داده از دیتابیمون و وارد کردن اون به جاش هستش . خوشبختانه ، ما میتونیم تمام این کار ها رو بدون کوئری های SQL انجام بدیم ، با استفاده از مدل های جنگو . ما با تغییر یک تنظیمات دیگه باید برنامه بلاگ خود رو به پروژه اولین بلاگ خود وارد کنیم . به قسمت INSTALED_APPS در خط 112 رفته و این رو به لیست اضافه کنید :

```
'FirstBlog.blog',
```

بعد فایل views.py رو برای اضافه کردن داده از دیتابیس تغییر دهید .

```
from django.shortcuts import render_to_response

from blog.models import posts

def home(request):
	entries = posts.objects.all()[:10]
	return render_to_response('index.html', {'posts' : entries})
```

بعد ، template رو آپدیت کنید برای دسترسی به داده :

```
<div class="container">
	<h1>First Blog</h1>
	<hr />
	{% for post in posts %}
		<div class="post">
		<h2>{{ post.title }}</h2>
		<h3>Posted on {{ post.timestamp }} by {{ post.author }}</h3>
		<p>{{ post.bodytext }}</p>
```

```
            </div>
            <hr />
        {% endfor %}
</div>
```

حال میتونیم به تمام داده های جدول در فایل views.py دسترسی پیدا کنیم ، بعد فقط 10 ورودی اول رو انتخاب میکنیم . این داده ها رو به template وارد میکنیم ، داخل ورودی ها چرخ میزنیم و داده رو با اچ تی ام الِ سایت خود نشان میدهیم . این هنوز کار نمیکنه ، چون هیچی در دیتابیس نیست . سرور رو قطع کنید و این رو اجرا کنید :

```
python2.6 manage.py syncdb
```

این جدول جدید برای پست های ما رو به دیتابیس اضافه میکنه . بعد ، یک تب جدید باز کنید و تایپ کنید :

```
mysql -u root -p
```

پسورد خود را تایپ کنید ، اینتر را بفشارید ، اجراش کنید :

```
INSERT INTO blog_posts (author, title, bodytext) values ('Bob', 'Hello World', 'Lorem
Ipsum');
```

به تب قبلی برید و سرور رو دوباره اجرا کنید . صفحه را رفرش کنید و حال باید یک پست با محتوا احماقانه ای که اضافه کردید ببینید . اگر فرمان MySQL رو چند دفعه دیگه اجرا کنید ، باید پست های بیشتری بعد از رفرش کردن صفحه ببینید .

سیستم ادمین جنگو

آخرین کاری که امروز میخواهیم انجام بدیم بررسی سیستم ادمین جنگو هستش . این یک سیستم قدرتمندی برای مدیریت سایت شما بدون نوشتن کد بیشتر هستش ، که اگر سایتی را از پایه میساختید باید کد مینوشتید . برای فعال سازی ، ما باید کمی از تنظیمات رو تغییر دهیم . اول ، در فایل urls.py خط های 4 و 5 و 13 تا 16 رو از حالت کامنت خارج کنید ، پس کاملا بتونید به صفحه ادمین دسترسی پیدا کنید . بعد ، به قسمت INSTALLED_APPS در فایل settings.py برید و 'django.contrib.admin' , و 'django.contrib.admindocs' , را از حالت کامنت خارج کنید . برای اجازه دادن به ادمین برای کنترل جدول پست ها ، یک فایل جدید با نام admin.py در پوشه بلاگ بسازید و کدهای زیر رو درش وارد کنید :

```
from django.contrib import admin
from blog.models import posts

admin.site.register(posts)
```

خط پایین رو دوباره اجرا کنید برای اضافه کردن جداول به قسمت ادمین و سرور رو ریستارت

کنید :

```
python2.6 manage.py syncdb
```

اگر حال *127.0.0.1:8000/admin* رو در مرورگر خود دیدید ، باید یک صفحه ورود ببینید . اطلاعاتی رو که قبلا در اولین اجرای فرمان syncdb نوشته بودید رو برای ورود بنویسید . شما باید یک قسمت به نام بلاگ ببینید ، با یک زیر نویس برای جدول پست ها . شما از این میتونید برای ساخت ، ویرایش و حذف پست های بلاگ با یک محیط ساده استفاده کنید . این همه کاری بود که باید انجام میدادید . شما یک بلاگ کامل و کار آمد ساختید . برای اتمام این درس ، ما نگاهی به نصب جنگو روی وب سرور می اندازیم .

نصب روی وب سرور

دو نوع وب هاستینگ هست ، و کدوم رو دارید در توانایی استفاده از جنگو تاثیر میگذاره . اگر هاستینگ قسمت شده (shared hosting) دارید ، هاست درست را دارید .

خیلی از هاست های ارزان قیمت پایتون را پشتیبانی نمیکنند . وقتی Php تقریبا تضمین شده است ، پشتیبانی از زبان های دیگر اینطور نیست . شما باید کنترل پنل را چک کنید که آیا پایتون (جنگو) موجود هست یا خیر . مشخصاً پروسه در هاست های مختلف متفاوت هست . تقریبا تام هسات ها apache رو اجرا میکنند ، و میتونیم از اون برای هاست کردن جنگو استفاده کنیم ، با استفاده از ماژول های mod_w wsgi یا mod_python در apache .

بیشتر هاست ها اسکریپت های تعداد زیادی زبان برنامه نویسی رو با استفاده از CGI اجرا میکنند . جنگو میتونه روی FastCGI اجرا بشه ، و همچنین ، به صورت تئوری روی CGI ، اما این کاملا پشتیبانی نمیشه و برای یک وبسایت خیلی کند عمل میکنه . شما باید چک کنید اگر اینها نصب هستند . اونها معمولا زیر heading پیدا میشن ، مثل "CGI and scripting language support"

اگر هاست VPS دارید ، یا آنقدر خوشبخت هستید که یک سرور واگذار شده داشته باشید ،

کارتان بسیار راحت تر هست . شما فقط باید قدم هایی رو که رفتیم رو طی کنید تا جنگو رو اجرا کنید . اگر پایتون ندارید ، میتونید با یک مدیریت پکیج نصبش کنید . سیستم شما شاید با جنگو از پیش نصب شده بیاد .

```
ssh root@example.com

wget http://www.djangoproject.com/download/1.3.1/tarball/
tar xzvf Django-1.3.1.tar.gz
cd Django-1.3.1
python setup.py install
```

وقتی جنگو رو روی سرور خودتون نصب کردید ، سایتی رو که ساختید رو با یک کلاینت آپلود کنید . شما میتونید فایل ها رو هر جایی قرار دهید ، اما اونها رو از پوشه public بیرون نگه دارید ، مگر نه هر کسی میتونه سورس سایت شما رو ببینه . من از home/ برای تمام پروژه هام استفاده میکنم .

بعد ، یک دیتابیس MySQL بسازید ، با نام firstblog روی سرور و دوباره دستور syncdb رو اجرا کنید . شما باید حساب خود را برای کنترل پنل ادمین دوباره بسازید ، اما این رو فقط یک دفعه بسازید بسه .

اگر سعی کنید و اجراش کنید ، شاید یک خطا دریافت کنید ، و این به دلیل این هستش که تنظیمات سرور با تنظیمات روی کامپیوتر شما متفاوت هست . شاید لازم باشد تا رمز خود را در فایل settings.py عوض کنید ، اما بسته به تنظیمات سرورتون شاید به مشکلات دیگه هم بر بخورید . در این مواقع گوگل دوست شماست !

برای اجرا سرور ، فرمان کمی متفاوت هست . شما باید یک آی پی و پورت مشخص کنید تا به سایت در انترنت دسترسی داشته باشید .

```
python manage.py runserver 0.0.0.0:8000
```

اگر سایتتون رو روی یک مرورگر وب مشاهده کنید ، روی پورت 8000 ، باید سایتتون رو ببینید .

جمع بندی

و این بود درس امروز --- و سری ما . امیدوارم طی این 5 تا درس قبلی مطالب مفیدی آموخته باشید ، و آماده یادگیری حتی پایتون بیشتر در آینده باشید . اگر از جنگو خوشتان آمده ، و دوست دارید اطلاعاتتون رو درباره این فریم ورک افزایش دهید ، در زیر چند آموزش دیگر در این

رابطه هست (البته این ها رو هم بعدا ترجمه میکنم نگران نباشید دوستان) .

Diving into Django
Making A To-do List
10 Useful Django Tips

مثل همیشه ، خوش حال میشم درباره هر سوالی درباره این آموزش یا کلا پایتون در قسمت نظرات با شما بحث کنم . مرسی که این مطلب رو مطالعه کردید .

مترجم : علی مرادی

ایمیل : adeadmarshal@gmail.com