django

It worked!

Tutorial Django 1.8

Régis da Silva @rg3915
e
Jayme Neto @kalkehcoisa

Grupy-SP

22 de setembro de 2015

Se tiver pressa...

```
$ git clone https://github.com/rg3915/django1.8.git
$ virtualenv -p python3 django1.8
$ cd django1.8
$ source bin/activate
$ make initial
$ ./manage.py runserver
```

... senão, leia o tutorial.

Ementa

- MTV e ORM
- 1 min de Python
- Instalação
- Criar o ambiente
- Criar o projeto e a App
- Deploy no Heroku

Objetivo

- Criar uma lista de filmes
- Retornar o filme de maior bilheteria
- Criar um formulário
- Ver os detalhes de cada filme

O que é Django?

Segundo Django Brasil,

> *Django é um framework web de alto nível escrito em Python que estimula o desenvolvimento rápido e limpo.*

O que é Django?

- adota o padrão MTV
- possui ORM
- admin
- herança de templates e modelos
- open source

Quem usa Django?

www.djangosites.org

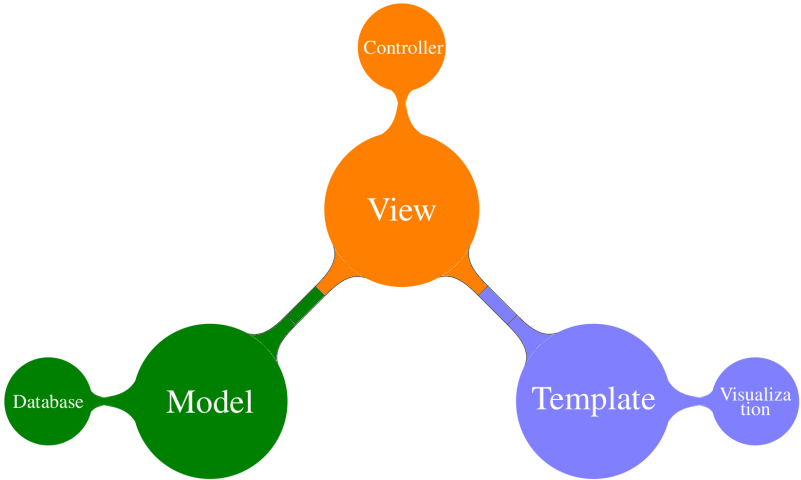
Sites

1. www.djangoproject.com/
2. www.djangobrasil.org/ (desatualizado)
3. www.djangopackages.com/
4. www.groups.google.com/forum/django-brasil
5. www.pythonclub.com.br/
6. www.github.com/rg3915/django-basic-apps
7. www.realpython.com/blog/categories/django/
8. www.marinamele.com/taskbuster-django-tutorial

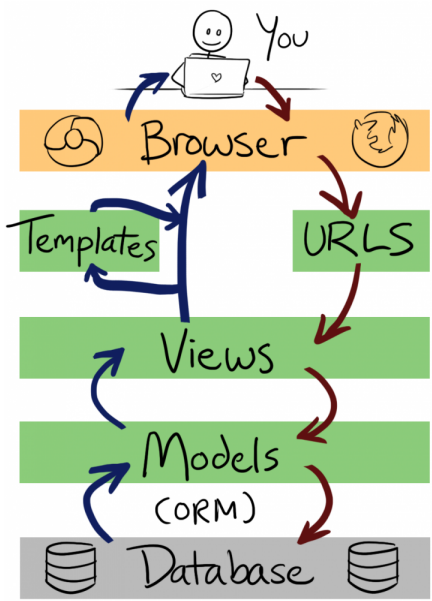
MVC x MTV

- **Model** - é o modelo, a camada de abstração do banco de dados, onde acontece o ORM
- **View** - é o controlador, onde acontece as regras de negócio e a comunicação entre a base de dados e o navegador
- **Templates** - é a camada de apresentação, são as páginas html

MVC x MTV



MVC x MTV



ORM

- Modelo (Classe) = Tabela
- Atributos = Colunas
- Objetos = Tuplas (Registros)

1 min de Python

```
public static void main (String[] args){
    System.out.println("Desculpa");
} # oops

print("Python") # simples assim

def soma(a, b):
    return a + b

soma(25,9)
```

1 min de Python

```
lista = ['a', 10, 5.5]
for i in lista:
    print(i)

for i in range(10):
    print(i)
```

O que você precisa?

- Python (2 ou 3)
- Pip
- VirtualEnv

Instalando Python no Windows

- Download do python: https://www.python.org/downloads/windows/
- Configurar as variáveis de ambiente (PATH)

Leia: *Instalando e Configurando o Python e Django no Windows* - Thiago Corôa
http://pythonclub.com.br/instalacao-python-django-windows.html

Pip

- Gerenciador de pacotes do python
- https://pip.pypa.io/en/latest/installing.html#install-pip

Instalando no Linux

- **Pip** http://pip.readthedocs.org/en/latest/

Primeira opção

```
$ wget https://bootstrap.pypa.io/get-pip.py
$ sudo python get-pip.py
```

Segunda opção

```
$ sudo apt-get install -y python-pip
```

- **VirtualEnv** https://virtualenv.pypa.io/en/latest/

Digite

```
$ sudo pip install virtualenv
$ # ou
$ sudo apt-get install -y virtualenv
```

O que vamos considerar no nosso projeto?

- Ambiente: venv
- Projeto: myproject
- App: core

Criando o ambiente

Vamos criar um ambiente usando o Python 3, então digite

```
$ virtualenv -p /usr/bin/python3 venv
```

onde *venv* é o nome do ambiente.

Entre na pasta

```
$ cd venv
```

e ative o ambiente

```
$ source bin/activate
```

Obs: todos os pacotes instalados com o ambiente ativado serão instalados dentro do ambiente e visíveis somente nele.

Dica: No Linux, edite o arquivo *~/.bashrc*

```
alias sa='source bin/activate;'
```

Assim você cria atalhos para ativar seus ambientes:

```
$ sa
```

Dica: Para diminuir o caminho do prompt digite

```
$ PS1="(`basename \"$VIRTUAL_ENV\"`):/\W$ "
```

O caminho vai ficar assim

```
(venv):/venv$
```

Onde *(venv)* é o nome do ambiente e *:/venv$* é a pasta atual.

Para desativar o ambiente digitamos

```
(venv):/venv$ deactivate
```

Instalando Django 1.8 + django-bootstrap-form

```
$ pip install django==1.8.4 django-bootstrap-form
```

https://github.com/tzangms/django-bootstrap-form

Vendo o que foi instalado

```
$ pip freeze
Django==1.8.4
django-bootstrap-form==3.2
```

Crie o *requirements.txt* (os ingredientes do bolo)

```
$ pip freeze > requirements.txt
```

Criando o projeto e a App

https://docs.djangoproject.com/en/1.8/intro/tutorial01/

Para criar o **projeto** digite

```
$ django-admin.py startproject myproject .
```

repare no ponto final do comando, isto permite que o arquivo *manage.py* fique na pasta ”principal”, pasta *venv*.

Criando a **app**

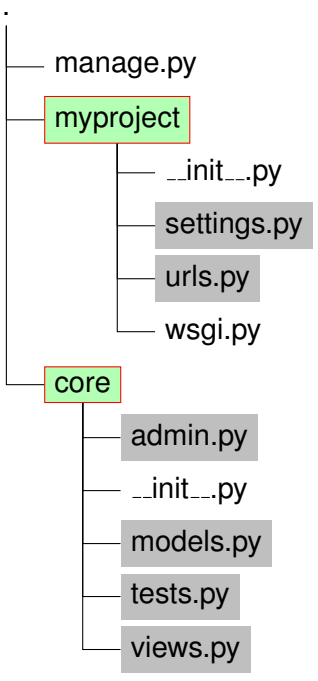
```
$ python manage.py startapp core
ou
$ ./manage.py startapp core
ou
$ manage startapp core
```

Dica: para funcionar o último comando você deve editar o *~/.bashrc*

```
$ alias manage='python $VIRTUAL_ENV/manage.py'
```

O que temos até aqui?

```
$ tree myproject; tree core
```


.
manage.py
myproject
__init__.py
settings.py
urls.py
wsgi.py
core
admin.py
__init__.py
models.py
tests.py
views.py

Django funcionando em nível 0

Criando a primeira migração

```
$ python manage.py migrate
```

Obs: o comando `migrate` se chamava `syncdb` e só era capaz de criar novas tabelas no banco de dados. Já o `migrate` consegue remover e alterar tabelas. Criado baseado nas funcionalidades do Django South.

Rodando o projeto

```
$ python manage.py runserver
```

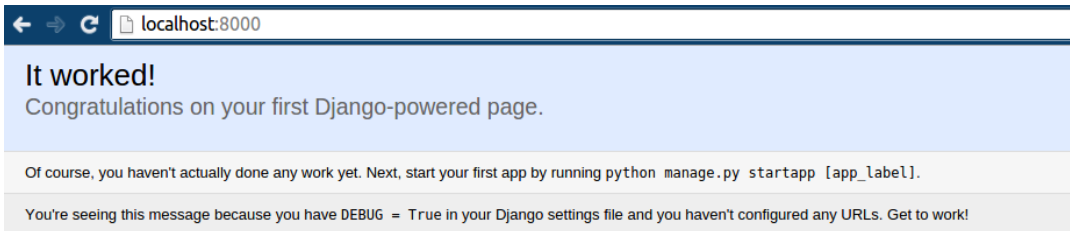
Por padrão ele está rodando na porta 8000

http://localhost:8000/ ou http://127.0.0.1:8000/

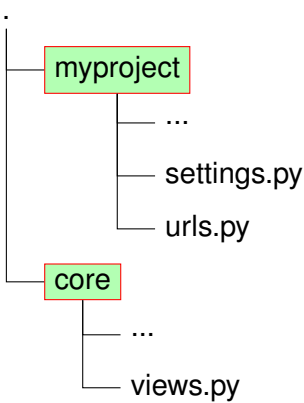
ou

```
$ python manage.py runserver <PORTA>
$ python manage.py runserver 8080
```

http://localhost:8080/


localhost:8000
It worked!
Congratulations on your first Django-powered page.
Of course, you haven't actually done any work yet. Next, start your first app by running python manage.py startapp [app_label].
You're seeing this message because you have DEBUG = True in your Django settings file and you haven't configured any URLs. Get to work!

O mínimo - nível 1: settings, views, urls



Editando settings.py

```
INSTALLED_APPS = (
    ...
    'core',
)
```

Editando views.py

```
# -*- coding: utf-8 -*-
# from django.shortcuts import render
from django.http import HttpResponse

def home(request):
    return HttpResponse('<h1>Django</h1>
                         <h3>Bem vindo ao Grupy-SP</h3>')
```

Editando urls.py

```
from django.conf.urls import include, url

urlpatterns = [
    url(r'^$', 'core.views.home'),
    url(r'^admin/', include(admin.site.urls)),
]
```

Ou

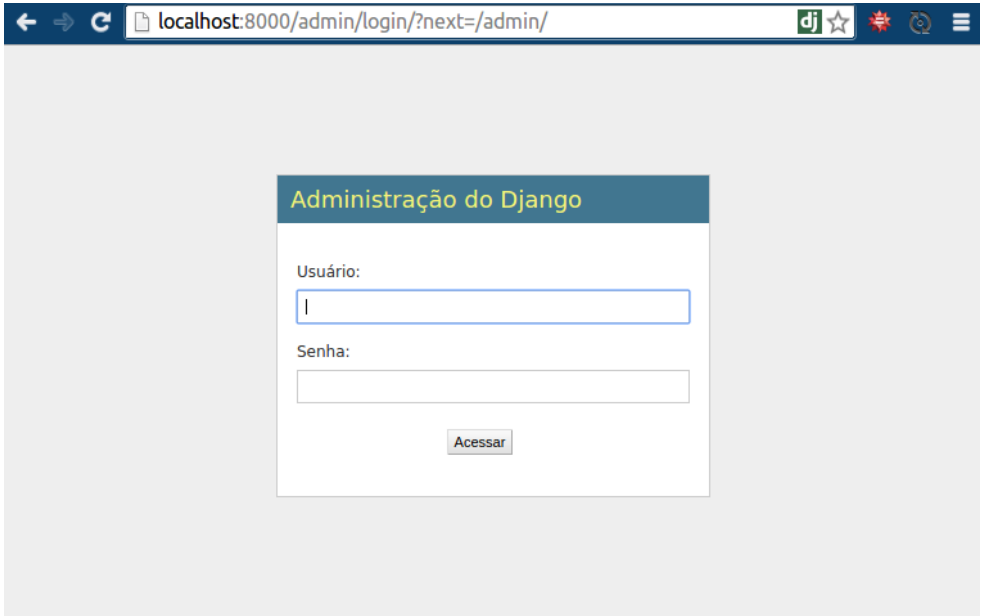
```
from django.conf.urls import patterns, include, url

urlpatterns = patterns(
    'core.views',
    url(r'^$', 'home'),
    url(r'^admin/', include(admin.site.urls)),
)
```

Django

Bem vindo ao Grupy-SP

Admin

```
$ python manage.py createsuperuser --username='admin' --email=''
```


localhost:8000/admin/login/?next=/admin/
Administração do Django
Usuário:
Senha:
Acessar

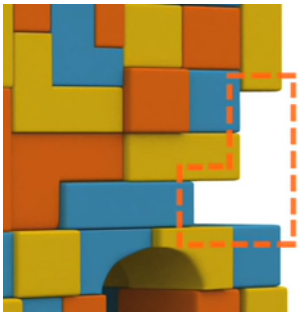

localhost:8000/admin/
Administração do Django
Bem-vindo(a), admin. Ver o site / Alterar senha / Encerrar sessão
Administração do Site
Autenticação e Autorização
Grupos
Adicionar
Modificar
Usuários
Adicionar
Modificar
Core
Categorias
Adicionar
Modificar
Diretores
Adicionar
Modificar
Filmes
Adicionar
Modificar
Ações Recentes
Minhas Ações
Nenhum disponível

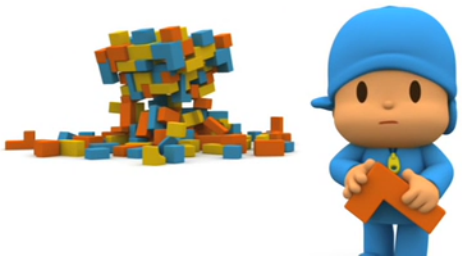
Tocando o barco

Editando settings.py

```
LANGUAGE_CODE = 'pt-br'

TIME_ZONE = 'America/Sao_Paulo'

LOGIN_URL = '/admin/login'
```

Testes





Teste: Verificar se existe a página *index.html*.

```
from django.test import TestCase

class HomeTest(TestCase):

    def setUp(self):
        self.resp = self.client.get('/')

    def test_get(self):
        ''' get / deve retornar status code 200. '''
        self.assertEqual(200, self.resp.status_code)

    def test_template(self):
        ''' Home deve usar template index.html '''
        self.assertTemplateUsed(self.resp, 'index.html')
```

Leia: *pytest: escreva menos, teste mais* - Erick Wilder de Oliveira - https://goo.gl/8E9FB1

Editando views.py

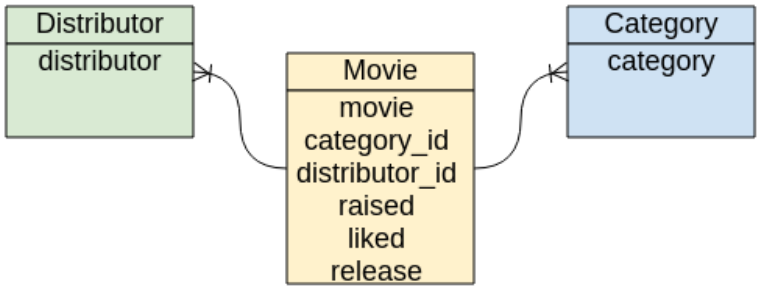
```
from django.shortcuts import render
# from django.http import HttpResponse

# def home(request):
#     return HttpResponse('<h1>Django</h1><h3>Bem vindo ao Grupy-

def home(request):
    return render(request, 'index.html')
```

Criando o index.html

Estando na pasta `venv` digite

```
$ mkdir -p core/templates
$ echo "<html><body><h1>Tutorial Django</h1>
        <h3>Bem vindo ao Grupy-SP</h3></body>
        </html>" > core/templates/index.html
```

Editando models.py - Filmes



Editando models.py

```
# -*- coding: utf-8 -*-
from django.db import models

class Distributor(models.Model):
    distributor = models.CharField('distribuidor',
                                   max_length=50, unique=True)

    class Meta:
        ordering = ['distributor']
        verbose_name = 'distribuidor'
        verbose_name_plural = 'distribuidores'

    def __str__(self): # ou __unicode__ no Python 2
        return self.distributor
```

```
class Category(models.Model):
    category = models.CharField('categoria',
                                max_length=50, unique=True)

    class Meta:
        ordering = ['category']
        verbose_name = 'categoria'
        verbose_name_plural = 'categorias'

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

```
class Movie(models.Model):
    movie = models.CharField('filme', max_length=100)
    category = models.ForeignKey(
        'Category', verbose_name='categoria',
        related_name='movie_category')
    distributor = models.ForeignKey(
        'Distributor', verbose_name='distribuidor',
        related_name='movie_distributor')
    raised = models.DecimalField(
            'arrecadou', max_digits=4, decimal_places=3)
    liked = models.BooleanField('gostou', default=True)
    release = models.DateTimeField(u'lancamento')
    class Meta:
        ordering = ['-release']
        verbose_name = 'filme'
        verbose_name_plural = 'filmes'
    def __str__(self):
        return self.movie
```

Tipos de campos

- BooleanField
- CharField
- DateField
- DateTimeField
- DecimalField
- DurationField
- EmailField
- FileField
- FloatField
- ImageField
- NullBooleanField
- PositiveIntegerField
- PositiveSmallIntegerField
- SlugField
- SmallIntegerField
- TextField
- TimeField
- ForeignKeyField
- ManyToManyField
- OneToOneField

https://docs.djangoproject.com/en/1.8/ref/models/fields/

Atualizando o banco

```
$ python manage.py makemigrations
$ python manage.py migrate
```

shell

Explorando um pouco as queryset.

```
$ python manage.py shell
Python 3.4.0 (default, Jun 19 2015, 14:18:46)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more informa
(InteractiveConsole)
>>>
```

Precisamos importar o models.

```
>>> from core.models import Distributor, Category, Movie
```

Todos os comandos estão em `shell/shell.py`

shell

```
$ manage shell < shell/distributors.py
$ manage shell < shell/movies.py
```

https://docs.djangoproject.com/en/1.8/ref/models/querysets/
https://pt.wikipedia.org/wiki/Lista_de_filmes_de_maior_bilheteria

Admin

```
from django.contrib import admin
from .models import Distributor, Category, Movie

admin.site.register(Distributor)
admin.site.register(Category)
admin.site.register(Movie)
```

Criando os templates

```
$ mkdir core/templates/core
$ touch core/templates/{base.html,menu.html}
$ touch core/templates/core/{movie_list.html,
                             movie_detail.html,
                             movie_form.html}
```