

Centro Universitário Barão de Mauá

Aluno

Willem Allan Rigo Ferreira

Framework de Desenvolvimento Web Django e Suas Principais Características

Orientador

Prof. Dr. Lucas Baggio Figueira



Estrutura do Trabalho

1. INTRODUÇÃO
2. APRESENTAÇÃO
 - FRAMEWORK
 - PARADIGMA MVC
 - DJANGO
3. DESENVOLVIMENTO DO PORTAL EDUCACIONAL
 - CONFIGURANDO O AMBIENTE DE TRABALHO
 - INICIANDO O PROJETO
 - CRIANDO UMA APP
 - MODELS
 - TEMPLATES
 - VIEWS
 - FORMULÁRIOS
 - ADMIN
4. CONCLUSÃO
5. REFERÊNCIAS BIBLIOGRÁFICAS



Introdução

- Atualmente, tornou-se vital que empresas e até pessoas disponham de um site para divulgação de sua marca, produtos e serviços oferecidos.
- O trabalho será composto por dois capítulos. O primeiro capítulo discorrerá sobre os conceitos teóricos e a apresentação geral do framework, enquanto o segundo demonstrará os passos dados na instalação e desenvolvimento da aplicação do portal, explicando trechos de códigos importantes.

Objetivos

Apresentar as características do Framework Django:

- Prazos curtos para entrega do site
- Facilidade em manutenção do código
- Padronização no código
- Facilidade em separar as tarefas para o trabalho em equipe



Apresentação

O Django é um novo Framework que vem ampliando os seus domínios, de forma crescente, dentro do cenário de desenvolvimento web.

O Django foi criado na linguagem Python, o que efetivamente agradou a utilizá-lo ao invés de outros Frameworks como o Ruby on Rails.



Framework

Um framework é uma aplicação que possui um padrão na estrutura dos projetos, o qual é separado em camadas, dividindo em arquivos e diretórios pré-definidos, que visam melhorar o seu desenvolvimento. Também possui um conjunto de bibliotecas definidas e outros componentes para ajudar o desenvolvedor a criar um projeto com o menor esforço possível.

Framework é uma aplicação quase completa, mas que possui partes que precisam ser escritas pelo desenvolvedor.

As principais características de um Framework

- Fácil de usar
- Reutilizável
- Bem documentado
- Contendo funcionalidades abstratas onde o desenvolvedor as implementa de acordo com a sua necessidade
- Seguro
- Completo para atender todos os problemas propostos



Frameworks que utilizam o paradigma MVC

- RoR (Ruby On Rails)
- Grails (Groovy on Rails)
- Django
- CakePHP

Paradigma MVC (model-view-controller)

- É um padrão de arquitetura de software que tende a separar a lógica de negócio da lógica de apresentação dos dados (design), permitindo um desenvolvimento ágil, isolando os testes e manutenção.
- Permite dividir as funcionalidades de seu sistema em camadas. Essa divisão é feita para facilitar a solução de problemas.

As três camadas do MVC

- Model: é a camada responsável por manipular as informações dos dados, onde esses dados e informações são provenientes de um banco de dados ou de arquivo XML.
- View: é a camada responsável por todo conteúdo da aplicação visualizado pelo usuário.
- Controller: é a camada responsável pelo controle do fluxo de informação que passa pelo sistema. É a controladora que executa a regra de negócio do modelo (model) e repassa a informação para a visualização (view).

Django

Django é um framework MVC de desenvolvimento web, escrito na linguagem Python, que foi desenvolvido pelo grupo editorial "The World Company" no intuito de disponibilizar uma versão web dos seus jornais. Posteriormente, em 2005, foi liberado sob a licença Berkeley Software Distribution (BSD), tornando-se assim um software de código aberto. O framework foi nomeado Django em homenagem ao famoso músico de jazz Django Reinhardt.

Atualmente está disponível na versão 1.3



Objetivo do Django

Resolver problemas complexos em prazos curtos, com isso prevenindo os desenvolvedores a não ultrapassassem os prazos estipulados.

Django

Assim como diversos frameworks ágeis de desenvolvimento web, o Django utiliza o conceito DRY Don't repeat yourself (Não se repita), que trabalha com a ideia de reutilização de código.

Django é considerado um superframework, pois, é composto de vários menores frameworks (componentes) .



Alguns componentes do Django

- Object-Relational Mapping
- Interface Administrativa
- Formulários
- URL Dispatcher
- Templates
- Cache
- Internacionalização

Object-relational mapping (ORM)

Permite que o programador crie sua aplicação, utilizando objetos sem se preocupar com a persistência desses dados no seu banco de dados.

Essa camada de software abstrai toda a comunicação com seu banco de dados, permitindo que o desenvolvedor manipule seus objetos sem o uso da linguagem SQL.

Object-relational mapping (ORM)

```
1 from django.db import models
2
3 class Aluno(models.Model):
4     nome = models.CharField(max_length=100, null=False)
5     login = models.CharField(max_length=45, null=False, unique=True)
6     senha = models.CharField(max_length=45, null=False)
7
8     class Meta:
9         db_table = u'aluno'
10
11     def __unicode__(self):
12         return str(self.id) + ' - ' + self.nome
```

```
CREATE aluno (
  id int(11) NOT NULL AUTO_INCREMENT,
  nome varchar(100) NOT NULL,
  login varchar(45) NOT NULL,
  senha varchar(45) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY login (login)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```


Interface Administrativa

Django é um dos poucos Frameworks web que disponibilizou uma interface administrativa automática em um design pronto.

Framework lê no seu modelo de metadados para fornecer uma interface agradável.

URL Dispatcher

URL dispatcher cuida do processamento das urls do sistema, executando funções especificadas pelo desenvolvedor e possibilitando o uso de urls amigáveis ao usuário.

Templates

Os programadores da linguagem PHP, que misturam o código PHP com o código HTML, precisam considerar que o sistema de templates do Django não é simplesmente um código Python embutido em HTML. Esse sistema, basicamente, recebe variáveis, onde estão os dados que serão exibidos, dados estes que serão tratados na view e depois demonstrados na página.

O sistema de templates é usado para exibir a apresentação dos dados, não a lógica do programa, onde pode gerar qualquer formato baseado em texto (HTML, XML, JSON e etc.).



Cache

Ele que cuida das requisições de um usuário ao site, isto é muito importante, pois, cada vez que um usuário acessa uma página, o servidor web faz todo o tipo de cálculo, como consultas a banco de dados, renderização de templates e lógica de negócio para criação da página que o seu visitante precisa ver. Este procedimento tem um custo significativo de processamento.

Qual seria o cenário se mais de cem usuários tentassem acessar simultaneamente a página?



Internacionalização

```
27 LOGIN_URL = "/login/"
28 LOGOUT_URL = "/logout/"
29 LOGIN_REDIRECT_URL = "/"
30
31 TIME_ZONE = 'America/Sao_Paulo'
32 LANGUAGE_CODE = 'pt-br'
33
34 SITE_ID = 1
35
36 USE_I18N = False
37 USE_L10N = True
38
39 MEDIA_ROOT = os.path.join(ROOTDIR, 'media')
40 MEDIA_URL = '/media/'
```

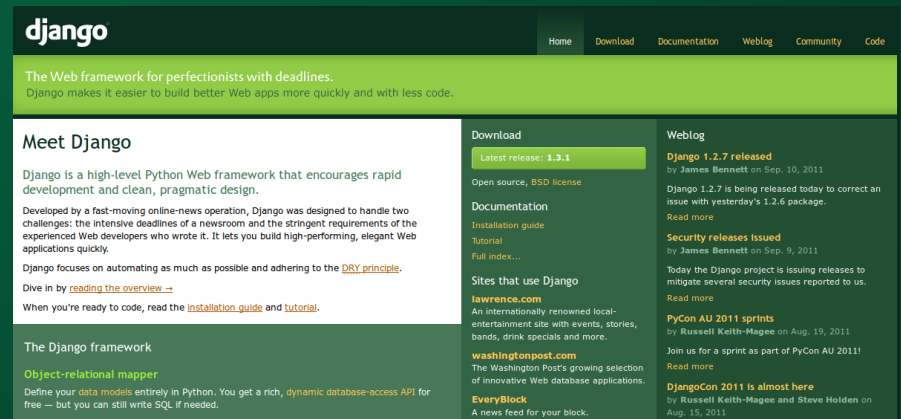
Desenvolvimento do Portal Educacional

Com a criação de um portal Educacional, é possível desenvolver uma aplicação web com alto nível de complexidade, utilizando diversos recursos interessantes do Framework.

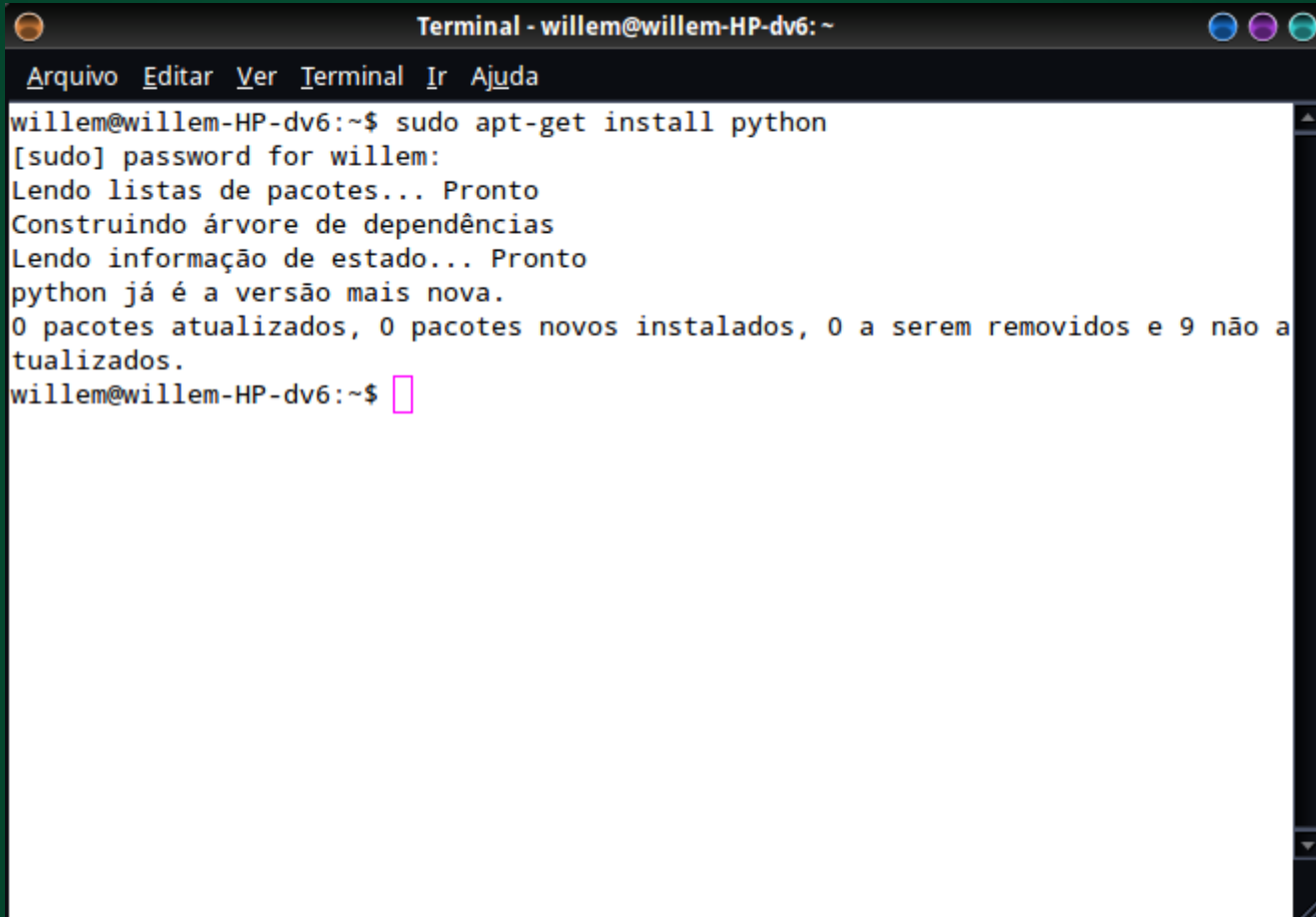


Configurando o ambiente de trabalho

- Instalar o Python (windows)
- Configurar a variável ambiente (windows)
- Instalar o Django (windows, mac, linux)

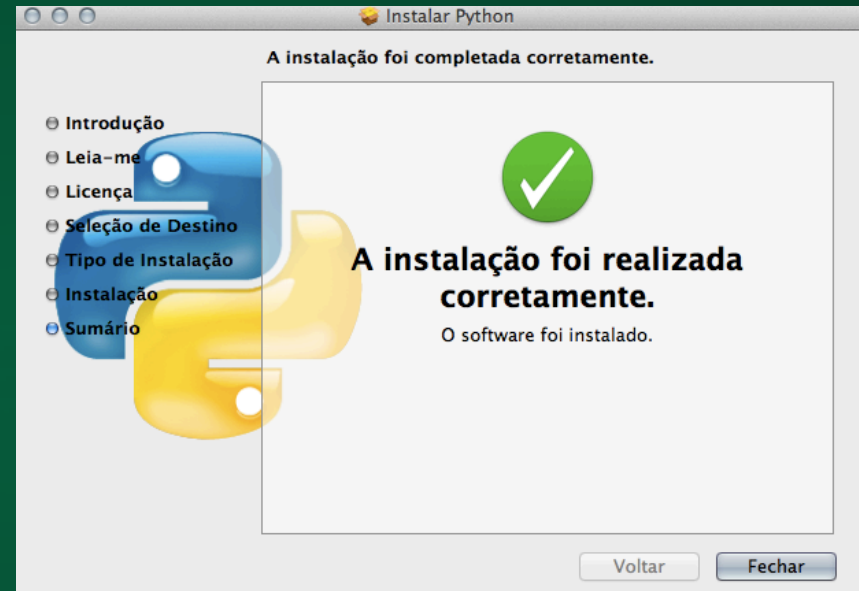


Instalação do python no Linux (ubuntu)

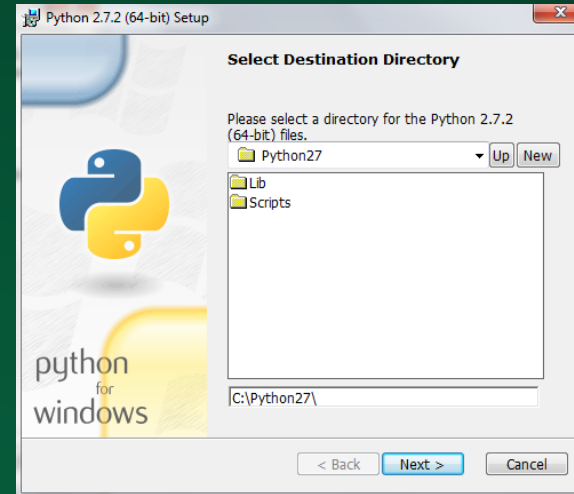
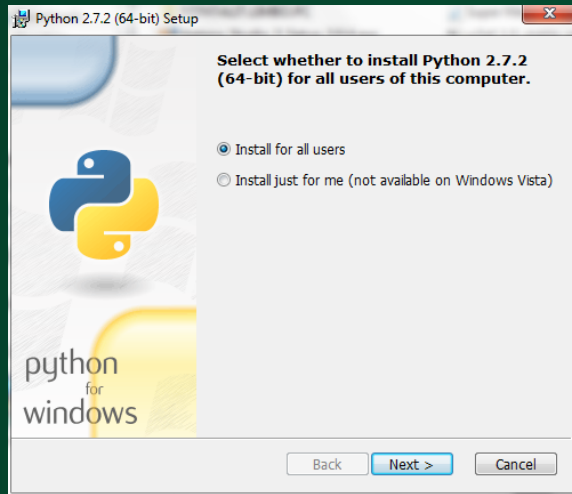


```
Terminal - willem@willem-HP-dv6: ~
Arquivo  Editar  Ver  Terminal  Ir  Ajuda
willem@willem-HP-dv6:~$ sudo apt-get install python
[sudo] password for willem:
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
python já é a versão mais nova.
0 pacotes atualizados, 0 pacotes novos instalados, 0 a serem removidos e 9 não a
tualizados.
willem@willem-HP-dv6:~$
```


Instalação do python no Mac Os X



Instalação do python no Windows



Configurando a variável ambiente no Windows

1. Configurações avançadas do sistema

Exibir informações básicas sobre o computador

Windows Edition

Windows 7 Home Premium

Copyright © 2009 Microsoft Corporation. Todos os direitos reservados.

Service Pack 1

Obtenha mais recursos com a nova edição do Windows 7

2. Para tirar o máximo proveito destas alterações, é preciso ter feito login como administrador.

3. Variáveis de Ambiente...

4. Path

5. Editar...

6. Nome da variável: Path

7. Valor da variável: (86)\Windows Live\Shared;C:\Python27\Scripts

8. OK

9. OK

Propriedades do Sistema

Nome do Computador Hardware

Avançado Proteção do Sistema Remoto

Desempenho

Efeitos visuais, agendamento de processador, uso de memória e memória virtual

Configurações...

Perfis de Usuário

Configurações da área de trabalho relativas ao seu login

Configurações...

Inicialização e Recuperação

Informações sobre inicialização do sistema, falha do sistema e depuração

Configurações...

Variáveis de Ambiente

Variáveis de usuário para willem

Variável	Valor
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

Novo... Editar... Excluir

Variáveis do sistema

Variável	Valor
NUMBER_OF_P...	4
OS	Windows NT
Path	C:\Program Files\Common Files\Microsof...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;...

Novo... Editar... Excluir

Editar Variável de Sistema

Nome da variável: Path

Valor da variável: (86)\Windows Live\Shared;C:\Python27\Scripts

OK Cancelar

Instalação do Django

A instalação do Django é quase igual em comparação aos outros sistemas operacionais, bastando descompactar o arquivo, abrir o terminal ou prompt de comando e navegar pelo terminal até atingir o diretório descompactado.

Se estiver utilizando Linux ou Mac, basta digitar o comando *sudo python setup.py install*, no windows apenas é necessário tirar o sudo, ficando assim o comando *python setup.py install*.



Instalação do Django

```
C:\Windows\system32\cmd.exe
12/09/2011 02:24 <DIR> Desktop
30/08/2011 16:03 <DIR> Documents
12/09/2011 02:45 <DIR> Downloads
12/09/2011 01:01 <DIR> Dropbox
08/09/2011 15:33 <DIR> duhelper
09/09/2011 12:09 <DIR> Favorites
19/08/2011 21:16 <DIR> Links
19/08/2011 21:16 <DIR> Music
02/09/2011 12:06 <DIR> My Documents
28/08/2011 02:03 <DIR> Pictures
02/09/2011 12:19 <DIR> portal_edu
19/08/2011 21:16 <DIR> Saved Games
19/08/2011 21:16 <DIR> Searches
12/09/2011 01:01 <DIR> Tracing
19/08/2011 21:16 <DIR> Videos
23/08/2011 17:02 <DIR> VirtualBox VMs
02/09/2011 16:09 <DIR> will
                2 arquivo(s)      76.824 bytes
                28 pasta(s)    468.506.636.288 bytes disponíveis

C:\Users\will>cd Down*
C:\Users\will\Downloads>cd Djan*
C:\Users\will\Downloads\Django-1.3.1>python setup.py install
```

```
C:\Windows\system32\cmd.exe
byte-compiling C:\Python27\Lib\site-packages\django\db\models\options.py to options.pyc
byte-compiling C:\Python27\Lib\site-packages\django\db\models\sql\query.py to query.pyc
byte-compiling C:\Python27\Lib\site-packages\django\http\multipartparser.py to multipartparser.pyc
byte-compiling C:\Python27\Lib\site-packages\django\http\__init__.py to __init__.pyc
byte-compiling C:\Python27\Lib\site-packages\django\test\client.py to client.pyc
byte-compiling C:\Python27\Lib\site-packages\django\utils\cache.py to cache.pyc
byte-compiling C:\Python27\Lib\site-packages\django\__init__.py to __init__.pyc
running install_scripts
copying build\scripts-2.7\django-admin.py -> C:\Python27\Scripts
running install_data
copying django\contrib\admin\media\css\forms.css -> C:\Python27\Lib\site-packages\django\contrib\admin\media\css
copying django\contrib\admin\media\css\rtl.css -> C:\Python27\Lib\site-packages\django\contrib\admin\media\css
copying django\contrib\gis\tests\data\texas.dbf -> C:\Python27\Lib\site-packages\django\contrib\gis\tests\data
running install_egg_info
Writing C:\Python27\Lib\site-packages\Django-1.3.1-py2.7.egg-info
C:\Users\will\Downloads\Django-1.3.1>
```



Iniciando o Projeto

Primeiro passo para iniciar o projeto, é executar o comando para criação da aplicação. Assim, utiliza-se o comando *Django-admin.py startproject portal_edu*. O arquivo *Django-admin.py* é o responsável por toda área administrativa do Django, linhas de comando, que descreve tudo o que se pode fazer com o framework.

Após executar este comando é criada uma pasta *portal_edu* com arquivos padrões de configuração do Framework.



Iniciando o Projeto

```
willem@willem-HP-dv6: ~/portal_edu
willem@willem-HP-dv6:~$ dir
Aptana\ Studio\ 3\ Workspace  Documents  Imagens  Público  will
Área\ de\ Trabalho           Downloads  Modelos  run.sh
Documentos                   Dropbox   Música   Vídeos
willem@willem-HP-dv6:~$ django-admin.py startproject portal_edu
willem@willem-HP-dv6:~$ ls
Aptana Studio 3 Workspace  Documents  Imagens  portal_edu  Vídeos
Área de Trabalho          Downloads  Modelos  Público    will
Documentos                Dropbox   Música   run.sh
willem@willem-HP-dv6:~$ cd portal_edu
willem@willem-HP-dv6:~/portal_edu$ ls
__init__.py  manage.py  settings.py  urls.py
willem@willem-HP-dv6:~/portal_edu$
```

O diretório terá três arquivos, eles são: manage.py, settings.py e urls.py

Criando uma APP

O framework possui um código pré-definido na criação da app.

O comando que deve ser executado é o *python manage.py startapp nome_da_aplicação*, um exemplo na utilização seria *python manage.py startapp aluno*.

```
MacBook-de-Willem:portal_edu willem$ python manage.py startapp aluno
MacBook-de-Willem:portal_edu willem$ python manage.py startapp professor
MacBook-de-Willem:portal_edu willem$ python manage.py startapp curso
MacBook-de-Willem:portal_edu willem$ python manage.py startapp disciplina
MacBook-de-Willem:portal_edu willem$ python manage.py startapp disciplinaAluno
MacBook-de-Willem:portal_edu willem$ python manage.py startapp equipamento
MacBook-de-Willem:portal_edu willem$ python manage.py startapp home
MacBook-de-Willem:portal_edu willem$ python manage.py startapp material
MacBook-de-Willem:portal_edu willem$ python manage.py startapp professorEquipamento
MacBook-de-Willem:portal_edu willem$ python manage.py startapp turma
MacBook-de-Willem:portal_edu willem$ ls
__init__.py          disciplina            manage.py            settings.py
__init__.pyc         disciplinaAluno      material             settings.pyc
aluno                equipamento          professor            turma
curso                home                 professorEquipamento  urls.py
MacBook-de-Willem:portal_edu willem$
```



Criando uma APP

```
MacBook-de-Willem:portal_edu willem$ cd aluno
MacBook-de-Willem:aluno willem$ ls
__init__.py      models.py        tests.py         views.py
MacBook-de-Willem:aluno willem$
```

Após a criação das app do projeto é preciso adicionar aos pacotes instalados dentro do settings.py

```
INSTALLED_APPS = (
    ....'django.contrib.auth',
    ....'django.contrib.contenttypes',
    ....'django.contrib.sessions',
    ....'django.contrib.admin',
    ....'django.contrib.sites',
    ....'django.contrib.messages',
    ....'django.contrib.staticfiles',
    ....'home',
    ....'curso',
    ....'turma',
    ....'aluno',
    ....'professor',
    ....'disciplina',
    ....'equipamento',
    ....'material',
    ....'disciplinaAluno',
    ....'professorEquipamento',
)
```

Models

```
1  # -*- coding: utf-8 -*-
2  from django.db import models
3
4  class Material(models.Model):
5      nome = models.CharField(max_length=100, null=False)
6      descricao = models.TextField()
7      data = models.DateTimeField(auto_now_add=True)
8      arquivo = models.FileField(upload_to='material')
9      disciplina_id = models.ForeignKey('disciplina.Disciplina')
10
11      class Meta:
12          db_table = u'material'
13          ordering = ["data"]
14          verbose_name = u"Material"
15          verbose_name_plural = u"Materiais"
16
17      def __unicode__(self):
18          return str(self.id) + ' - ' + self.nome
```

Relacionamentos

O framework possibilita ao programador utilizar os três tipos de relacionamento:

- um-para-um (OneToOneField)
- muitos-para-um (ForeignKey)
- muitos-para-muitos (ManyToManyField)






















Templates

O framework Django oferece um sistema de templates para criação de páginas. Este sistema é encarregado de transformar as variáveis do arquivo em valores que são obtidos pela view. O sistema de templates possui diversas estruturas de controle que o transformam em um tipo de linguagem de programação.



Templates

Primeiro passo é criar o diretório na pasta raiz do projeto *portal_edu*

Nome	Data de modificaç...	Tipo	Tamanho
 aluno	20/10/2011 21:15	Pasta de arquivos	
 curso	20/10/2011 20:57	Pasta de arquivos	
 disciplina	20/10/2011 21:15	Pasta de arquivos	
 disciplinaAluno	25/10/2011 03:39	Pasta de arquivos	
 equipamento	20/10/2011 21:15	Pasta de arquivos	
 home	20/10/2011 20:57	Pasta de arquivos	
 material	20/10/2011 21:15	Pasta de arquivos	
 media	13/10/2011 03:02	Pasta de arquivos	
 professor	27/10/2011 00:36	Pasta de arquivos	
 professorEquipamento	27/10/2011 00:36	Pasta de arquivos	
 static	06/10/2011 23:56	Pasta de arquivos	
 templates	06/10/2011 23:32	Pasta de arquivos	
 turma	20/10/2011 20:57	Pasta de arquivos	
 __init__.py	06/10/2011 23:31	Arquivo PY	0 KB
 __init__.pyc	20/10/2011 20:57	Compiled Python ...	1 KB
 manage.py	06/07/2011 02:38	Arquivo PY	1 KB
 portaledu.db	27/10/2011 00:40	Data Base File	212 KB
 settings.py	27/10/2011 00:35	Arquivo PY	3 KB
 settings.pyc	27/10/2011 00:36	Compiled Python ...	4 KB
 urls.py	13/10/2011 03:17	Arquivo PY	2 KB
 urls.pyc	20/10/2011 20:57	Compiled Python ...	3 KB

Templates

Segundo passo é configurar o arquivo settings.py e definir o diretório de templates.

```
1 import os
2
3 ROOTDIR = os.path.dirname(__file__)
```

```
63 TEMPLATE_DIRS = (
64     os.path.join(ROOTDIR, 'templates'),
65 )
```

Templates

Primeiro template: Base.html

```
1 <html>
2 ..<head>
3 ....{% block cabecalho %}
4 ....<title>Portal EDU</title>
5 ....<link rel="stylesheet" type="text/css" href="{% STATIC_URL %}css/estilo.css" />
6 ....{% endblock %}
7 ..</head>
8 ..<body>
9 ....<h1>Portal EDU</h1>
10 ....{% block corpo %}
11 ....{% endblock %}
12 ..</body>
13 </html>
```

Templates

Primeiro template: Base.html

```
1 <html>
2 ..<head>
3 ....{% block cabecalho %}
4 ....<title>Portal EDU</title>
5 ....<link rel="stylesheet" type="text/css" href="{{ STATIC_URL }}css/estilo.css" />
6 ....{% endblock %}
7 ..</head>
8 ..<body>
9 ....<h1>Portal EDU</h1>
10 ....{% block corpo %}
11 ....{% endblock %}
12 ..</body>
13 </html>
```


Templates

Segundo template: index.html

```
1 {% extends 'base.html' %}
2
3 {% block corpo %}
4
5 <div class="opcao"><a href="/aluno/">Aluno</a></div>
6 <div class="opcao"><a href="/professor/">Professor</a></div>
7 <div class="opcao"><a href="/admin/">Administrador</a></div>
8
9 {% endblock %}
```

Templates

Configurando o settings

```
37 MEDIA_ROOT = os.path.join(ROOTDIR, 'media')
38 MEDIA_URL = '/media/'
39
40 STATIC_ROOT = os.path.join(ROOTDIR, 'public')
41 STATIC_URL = '/static/'
42
43 ADMIN_MEDIA_PREFIX = STATIC_URL + 'admin/'
44
45 STATICFILES_DIRS = (
46     os.path.join(ROOTDIR, 'static'),
47 )
```

Templates

Arquivo css: estilo.css

```
1 body,div,p,td {
2   — font-family: Verdana, Arial, sans-serif;
3   — font-size: 12px;
4 }
5
6 .opcao{
7   *.border: #CCCCCC solid 1px;
8   *.background: #F2F2F2;
9   *.color:#000000;
10  *.margin:5px;
11  *.padding:5px;
12 }
13
14 a{
15   *.color: black;
16   *.text-decoration: none;
17   *.font-weight:bold;
18 }
```

Views

Próximo passo é criar a view, que se encarregará de enviar as informações para o template poder exibi-los.

“python manage.py startapp home”

Para a visualização da página principal do portal é necessário alterar a views.py, que se encontra dentro do diretório home. O autor criou a função index que será chamada no arquivo urls.py, onde cuida de todas as urls do portal.

```
1 from django.shortcuts import render_to_response
2 from django.template import RequestContext
3
4 def index(request):
5     return render_to_response('index.html', {}, context_instance=RequestContext(request))
```

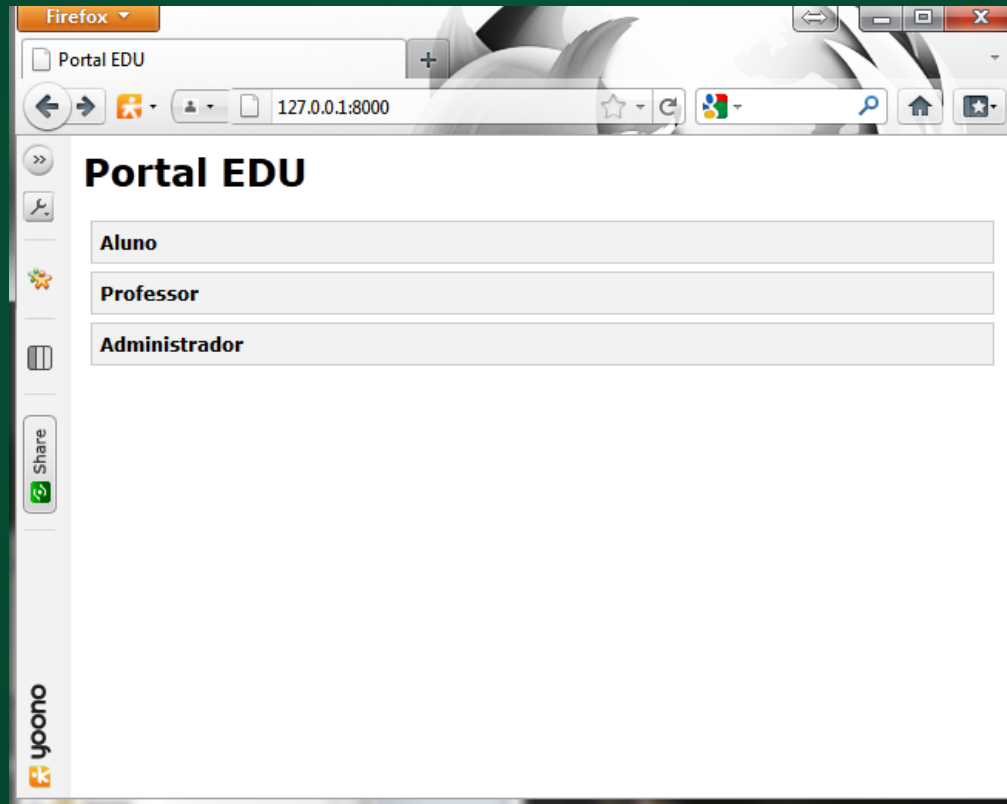


URL Dispatcher

```
1 from django.conf.urls.defaults import *
2
3 urlpatterns = patterns('',
4     — (r'^$', 'home.views.index'),
5 )
```

Iniciar o servidor

python manage.py runserver



Admin

Configurando o settings

```
78 INSTALLED_APPS = (  
79     ....'django.contrib.auth',  
80     ....'django.contrib.contenttypes',  
81     ....'django.contrib.sessions',  
82     ....'django.contrib.admin',  
83     ....'django.contrib.sites',  
84     ....'django.contrib.messages',  
85     ....'django.contrib.staticfiles',  
86     ....'home',  
87     ....'curso',  
88     ....'turma',  
89     ....'aluno',  
90     ....'professor',  
91     ....'disciplina',  
92     ....'equipamento',  
93     ....'material',  
94     ....'disciplinaAluno',  
95     ....'professorEquipamento',  
96 )
```

Admin

Configurando o URL Dispatcher

```
from django.conf.urls.defaults import *
from django.contrib import admin

admin.autodiscover()

urlpatterns = patterns('',
    (r'^$', 'home.views.index'),
    (r'^aluno/$', 'aluno.views.login'),
    (r'^aluno/logout/$', 'aluno.views.logout'),
    (r'^aluno/disciplina/$', 'disciplinaAluno.views.disciplinasAluno'),
    (r'^professor/$', 'professor.views.login'),
    (r'^professor/logout/$', 'professor.views.logout'),
    (r'^professor/material/$', 'material.views.material'),
    (r'^professor/material/adiciona/$', 'material.views.adiciona'),
    (r'^professor/equipamento/$', 'equipamento.views.equipamento'),
    (r'^professor/disciplina/$', 'disciplina.views.disciplinasProfessor'),
    (r'^professor/disciplina/(?P<id_disciplina>\d+)/$', 'disciplinaAluno.views.disciplinasAlunos'),
    (r'^professor/nota/aluno/(?P<id_disciplinaAluno>\d+)/$', 'disciplinaAluno.views.alterarNotaAluno'),
    (r'^professor/equipamento/(?P<id_equipamento>\d+)/$', 'professorEquipamento.views.reservarEquipamento'),
    ....(r'^admin/', include(admin.site.urls)),
)
```

Para concluir a criação do admin automático, execute o comando `python manage.py syncdb`



Admin

Só isso?



Admin

Administrativo foi criado, porém o administrativo não possui nenhuma tela dos modelos criados pelo desenvolvedor.

```
1 from aluno.models import Aluno
2 from django.contrib import admin
3
4 admin.site.register(Aluno)
```

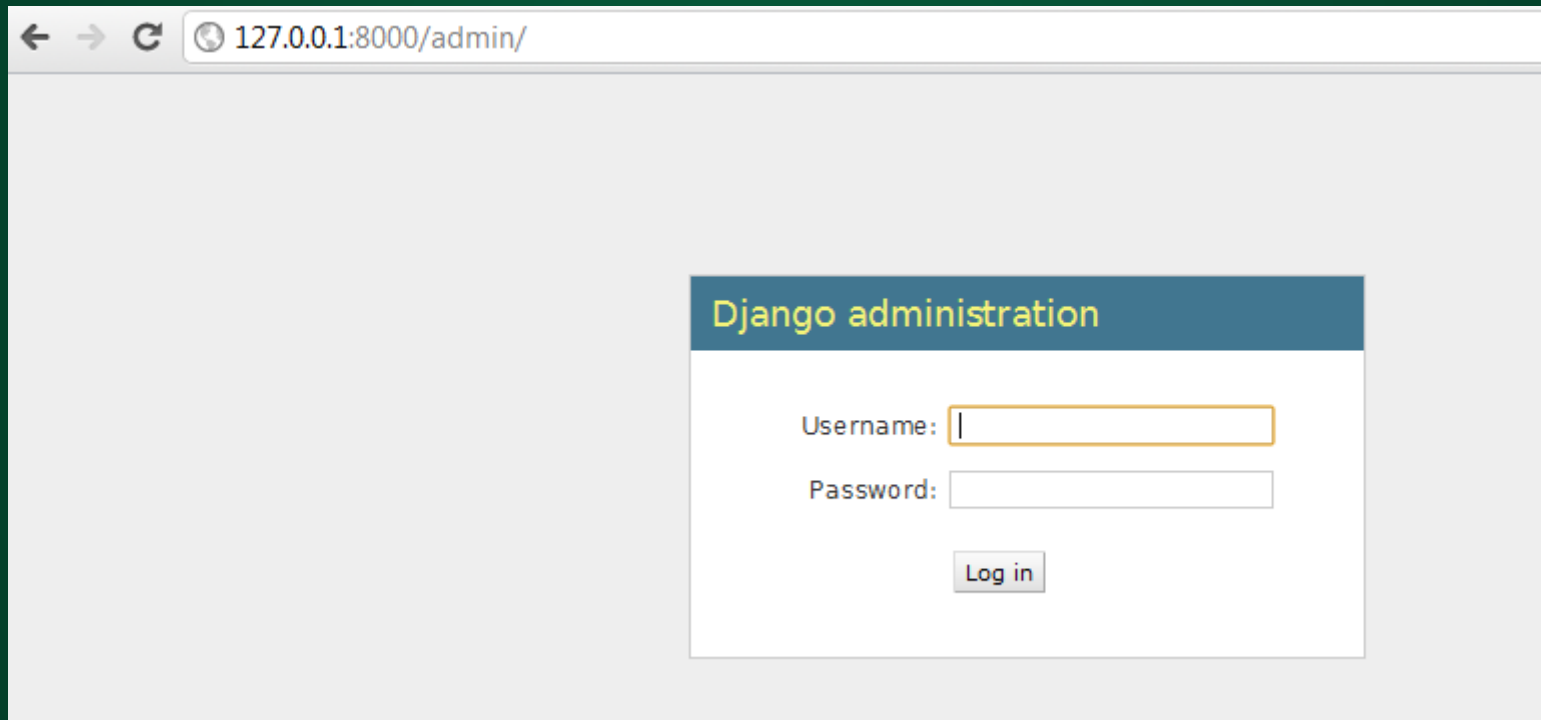
```
1 from turma.models import Turma
2 from django.contrib import admin
3
4 admin.site.register(Turma)
```

```
1 from professor.models import Professor
2 from django.contrib import admin
3
4 admin.site.register(Professor)
```

Admin

É bastante simples criar o administrativo do Django.

Para acessá-lo basta abrir o navegador e digitar o endereço `http://127.0.0.1:8000/`.



Admin

← → ↻ 127.0.0.1:8000/admin/ ☆ 📶 🐛 🔧

Django administration Welcome, **admin**. Change pass word / Log out

Site administration

Aluno	
Alunos	+ Add Change
Auth	
Groups	+ Add Change
Users	+ Add Change
Curso	
Cursos	+ Add Change
Disciplina	
Disciplinas	+ Add Change
Disciplinaaluno	
Disciplina alunos	+ Add Change
Equipamento	
Equipamentos	+ Add Change
Material	
Materiais	+ Add Change
Professor	
Professors	+ Add Change
Professorequipamento	
Professor equipamentos	+ Add Change
Sites	
Sites	+ Add Change
Turma	
Turmas	+ Add Change

Recent Actions

My Actions

- [+ 2 - Projetor 2](#)
Equipamento
- [+ 1 - Projetor](#)
Equipamento
- [will](#)
User
- [+ will](#)
User

Admin

127.0.0.1:8000/admin/disciplina/disciplina/

Esta página está em **inglês** Deseja traduzi-la? Traduzir Não Opções

Django administration Welcome, **admin**. Change pass word / Log out

Home > Disciplina > Disciplinas

Select disciplina to change Add disciplina +

Action: ----- Go 0 of 2 selected

<input type="checkbox"/>	Disciplina
<input type="checkbox"/>	2 - Eletrônica Digital
<input type="checkbox"/>	1 - Métodos Numéricos

2 disciplinas

127.0.0.1:8000/admin/disciplina/disciplina/add/

Esta página está em **inglês** Deseja traduzi-la? Traduzir Não Opções

Django administration Welcome, **admin**. Change pass word / Log out

Home > Disciplina > Disciplinas > Add disciplina

Add disciplina

Nome:

Professor id: ----- +

Curso id: ----- +

- 1 - Ciência da Computação
- 2 - Farmácia
- 3 - Direito
- 4 - Medicina

Save and continue editing Save and add another Save

Admin

← → ↻ 127.0.0.1:8000/admin/professorEquipamento/professorequipamento/add/ ☆ 📶 🐛 🔧

Esta página está em inglês ▼ Deseja traduzi-la? Traduzir Não Opções ▼ ✕

Django administration Welcome, **admin**. Change pass word / Log out

Home > ProfessorEquipamento > Professor equipamentos > Add professor equipamento

Add professor equipamento

Professor id: ▼ +

Equipamento id: ▼ +

Data inicial: Date: Today | 📅 Time: Now | 🕒

Data final: Date: Today | 📅 Time: Now | 🕒

Sala:

Save and continue editing Save and add another **Save**

November 2011						
S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			
Yesterday Today Tomorrow						
Cancel						

Admin

← → ↻ 127.0.0.1:8000/admin/professorEquipamento/professorequipamento/add/ ☆ 📶 🐛 🔧

Esta página está em inglês ▼ Deseja traduzi-la? Traduzir Não Opções ▼ ✕

Django administration Welcome, **admin**. Change pass word / Log out

Home > ProfessorEquipamento > Professor equipamentos > Add professor equipamento

Add professor equipamento

Professor id: ▼ +

Equipamento id: ▼ +

Data inicial: Date: Today | 📅 Time: Now | 🕒

Data final: Date: Today | 📅 Time: Now | 🕒

Sala:

Save and continue editing Save and add another **Save**

November 2011						
S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			
Yesterday Today Tomorrow						
Cancel						

Conclusão

- Python não foi programado para uso na web, mas o Django possibilitou-lhe a criação de aplicações web.
- Dois aspectos que favoreceram o uso do framework: sistema administrativo muito prático; segurança do URL dispatcher.
- Uma vez aprendido, o framework mostra-se rápido, dinâmico e eficiente, gerando grande economia de tempo, qualidades que caracterizam o Django e os demais frameworks.

Conclusão

- Diversos deles são atualmente adotados por empresas de desenvolvimento de software.
- Tendo a destacar que uma maior disponibilidade de tempo seria necessária para que se pudesse melhor explorar os recursos ofertados pelo framework escolhido, dentre os quais podem ser mencionados o de internacionalização, paginação, comentários etc.

Referências Bibliográficas

- OFICINA DA NET. O que é Model-view-controller (MVC). 2011. Disponível em: <http://www.oficinadanet.com.br/artigo/desenvolvimento/o_que_e_model-view-controller_mvc/>. Acesso em: 04 set. 2011.
- PENSO TI. O que é MVC. 2011. Disponível em: <<http://www.pensoti.com.br/2011/desenvolvimento/o-que-e-mvc/>>. Acesso em: 10 ago. 2011.
- UFCG TI. O que é um framework. Disponível em: <<http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/frame/oque.htm>>. Acesso em: 05 ago. 2011.
- CODE IGNITER BRASIL. MVC (Model – View – Controller). Disponível em: <<http://codeigniterbrasil.com/passos-iniciais/mvc-model-view-controller/>>. Acesso em: 07 set. 2011.
- DJANGO. Quick install guide. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/intro/install/>>. Acesso em: 03 ago. 2011.



Referências Bibliográficas

- DJANGO. Models. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/db/models/>>. Acesso em: 12 set. 2011.
- DJANGO. Model field reference. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/ref/models/fields/>>. Acesso em: 12 set. 2011.
- DJANGO. Model Meta options. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/ref/models/options/>>. Acesso em: 12 set. 2011.
- DJANGO. Working with forms. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/forms/>>. Acesso em: 14 set. 2011.
- DJANGO. Django settings. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/settings/>>. Acesso em: 06 set. 2011.
- DJANGO. Django-admin.py and manage.py. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/ref/Django-admin/>>. Acesso em: 06 set. 2011.



Referências Bibliográficas

- DJANGO. Django's cache framework. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/cache/>>. Acesso em: 07 set. 2011.
- DJANGO. The Django admin site. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/ref/contrib/admin/>>. Acesso em: 03 set. 2011.
- DJANGO. File Uploads. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/http/file-uploads/>>. Acesso em: 03 set. 2011.
- DJANGO BRASIL. Trabalhando com formulários. 2011. Disponível em: <<http://docs.djangobrasil.org/topics/forms/index.html#topics-forms-index>>. Acesso em: 03 set. 2011.
- DJANGO BRASIL. Models. 2011. Disponível em: <<http://docs.djangobrasil.org/topics/db/models.html>>. Acesso em: 03 set. 2011.
- SANTANA, Osvaldo; GALESI, Thiago. PYTHON e DJANGO – Desenvolvimento ágil de aplicações web. Novatec Editora Ltda. São Paulo, 2010.



Referências Bibliográficas

- DJANGO. URL dispatcher. 2011. Disponível em: <<https://docs.djangoproject.com/en/1.3/topics/http/urls/>>. Acesso em: 25 out. 2011.
- OFICINA DA NET. MVC – O padrão de arquitetura de software. 2011. Disponível em: <http://www.oficinadanet.com.br/artigo/1687/mvc_-_o_padrao_de_arquitetura_de_software>. Acesso em: 09 set. 2011.