# Django Unchained

PyTexas 2014 Tutorial by Douglas Mendizábal

# Installation

$ pip install django

# Create a new project called “bookstore”

$ django-admin.py startproject bookstore

$ cd bookstore/

## Run the development web server

$ python manage.py runserver

Django is now serving your project at <http://localhost:8000>

# Create a new application “inventory”

$ python manage.py startapp inventory

## Add the new application to your project

In **bookstore/bookstore/settings.py** change **INSTALLED\_APPS** to include **‘inventory’.**

INSTALLED\_APPS = (

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'inventory',

)

# Initial URL mapping

In **bookstore/bookstore/urls.py** add a new url pattern:

url(r'^$', 'inventory.views.home'),

# Simple View

In **bookstore/inventory/views.py** add a new view:

a) Import Django’s HTTPResponse

from django.http import HttpResponse

b) Add the **home** view we previously mapped in the URLConf

def home(request):

return HttpResponse(

'<html><body><h1>Django Unchained!</h1></body></html>'

)

We should now have a working home page that says “Django Unchained!”

# Models

For copy/paste see this GitHub Gist: <TODO: Gist URL>

In **bookstore/inventory/models.py** add the following two models:

class Author(models.Model):

first\_name = models.CharField(max\_length=256)

last\_name = models.CharField(max\_length=256)

class Book(models.Model):

author = models.ForeignKey(Author)

title = models.CharField(max\_length=256)

release\_date = models.DateField(null=True)

isbn = models.CharField(max\_length=13)

list\_price = models.DecimalField(

max\_digits=8, decimal\_places=2

)

on\_hand = models.IntegerField()

## Create and apply the first migration

$ python manage.py makemigraitons

$ python manage.py migrate

# Object Relational Mapper (ORM)

Now that we have a database, we can start saving data. Start an interactive python session:

$ python manage.py shell

>>> from inventory.models import \*

>>> mark = Author(first\_name='Mark', last\_name='Twain')

>>> mark.first\_name

>>> mark.last\_name

>>> mark.pk

>>> mark.pk is None

True

>>> mark.save()

>>> mark.pk

1

## Get some better output from the repl

Add \_\_str\_\_ methods to the Models

# Using Models in your view

Let’s design a view that will give you book details when you navigate to a URL that has the book’s ISBN

url(r'^books/(\d+)/$', 'inventory.views.detail'),

## Adding the detail view

a) In **bookstore/inventory/views.py** let’s add some imports

from django.http import HttpResponse, Http404

from inventory.models import Book

b) Add the detail view function

def detail(request, isbn):

matches = Book.objects.filter(isbn\_\_exact=isbn)

if not matches:

raise Http404

else:

book = matches[0]

return render(request, 'detail.html', {'book': book})

# Templates

a) Create a new directory for templates at **bookstore/inventory/templates/**

b) add a new **base.html** template

<!DOCTYPE html>

<html>

<head>

<title>{% block title %}{% endblock %}</title>

</head>

<body>

{% block content %}

{% endblock %}

</html>

c) add a new detail.html template

{% block title %}Django Bookstore{% endblock %}

{% block content %}

<div class="book">

<h1>{{ book.title }}</h1>

<h2>by {{ book.author.first\_name }}

{{ book.author.last\_name }}</h2>

{% if book.release\_date %}

<p>Released on {{ book.release\_date }}</p>

{% endif %}

</div>

{% endblock %}

# Forms

Let’s add a new model to save form data

class Order(models.Model):

first\_name = models.CharField(max\_length=256)

last\_name = models.CharField(max\_length=256)

email = models.EmailField()

book = models.ForeignKey(Book)

quantity = models.IntegerField(default=1)

$ python manage.py makemigrations

$ python manage.py migrate

Now let’s make a Form from the model

a) Add a new file **bookstore/inventory/forms.py**

b) Import Django forms and our models

from django import forms

from inventory import models

class OrderForm(forms.Form):

first\_name = forms.CharField(max\_length=256)

last\_name = forms.CharField(max\_length=256)

email = forms.EmailField()

book = forms.ModelChoiceField(

queryset=models.Book.objects.all()

)

quantity = forms.IntegerField()

Now let’s wire up some new URLs and a couple of views to process form data.

a) In **bookstore/bookstore/urls.py** add

url(r'^books/order/$', 'inventory.views.order'),

url(r'^books/thanks/$', 'inventory.views.thanks'),

b) In **bookstore/inventory/views.py** add

from django.http import HttpResponse, Http404, \

HttpResponseRedirect

from inventory.forms import OrderForm

from inventory.models import Book, Order

def thanks(req):

return HttpResponse(

'<html><body><h1>Thanks!</h1></body></html'

)

def order(req):

if req.method == 'POST':

form = OrderForm(req.POST)

if form.is\_valid():

order = Order()

order.first\_name = form.cleaned\_data['first\_name']

order.last\_name = form.cleaned\_data['last\_name']

order.book = form.cleaned\_data['book']

order.quantity = form.cleaned\_data['quantity']

order.save()

return HttpResponseRedirect('/books/thanks/')

else:

form = OrderForm()

return render(req, 'order.html', {'form': form})

c) Add a new template in **bookstore/inventory/templates/order.html**

{% block title %}Order a book{% endblock %}

{% block content %}

<form action="/books/order/" method="post">

{% csrf\_token %}

{{ form.as\_p }}

<input type=submit>

</form>

{% endblock %}

# Admin Site

First let’s add a new user

$ python manage.py createsuperuser

Now let’s check out the admin site at <http://localhost:8000/admin>

## Add your models to the Admin site

In **bookstore/inventory/admin.py** add

from inventory import models

admin.site.register(models.Book)

admin.site.register(models.Author)

admin.site.register(models.Order)