

# PyLadies Stockholm

Build your own blog with Django



November 19, 2013

# Goals

- Have fun
- Learn about web development
- Create something



# Layout

- Quick intro to Python
- Overview of Django framework
- Code together



# Setup

Either use one of the USBs

-or-

Navigate here: [gist.io/7525444](https://gist.io/7525444)



# Vocab

- shell/terminal/console
- Python
- Django
- Text Editor



# Know your prompts!

- Terminal prompt starts with “\$” or “C:\  
• Python shell prompt starts with “>>>”

```
C:\Users\lynn>
```

```
$
```

```
vagrant@pyladies: ~$
```

```
>>>
```

# Vocab

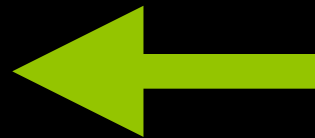
- Terminal prompt starts with “\$” or “C:\  
• Python shell prompt starts with “>>>”

C:\Users\lynn>

\$

vagrant@pyladies: ~\$

>>>



**Windows prompt**

# Vocab

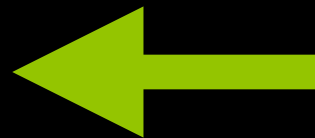
- Terminal prompt starts with “\$” or “C:\  
• Python shell prompt starts with “>>>”

C:\Users\lynn>

\$

vagrant@pyladies: ~\$

>>>



**Mac/Linux prompt**



# Vocab

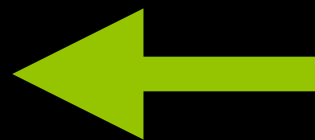
- Terminal prompt starts with “\$” or “C:\  
• Python shell prompt starts with “>>>”

C:\Users\lynn>

\$

vagrant@pyladies: ~\$

>>>



**Linux/VM prompt**

# Vocab

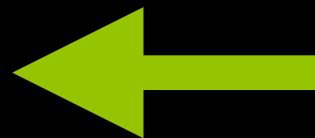
- Terminal prompt starts with “\$” or “C:\  
Python shell prompt starts with “>>>”

C:\Users\lynn>

\$

vagrant@pyladies: ~\$

>>>



**Python prompt**

# Vocab

- Data types & variables
- Functions/methods
- Classes
- Modules & Packages



# NOTE!!!

Everything from now on is in the **Virtual Machine!**

Within your “pyladies” folder, to get into the Virtual Machine, type:

**vagrant ssh**



# Let's try Python

- SSH into the virtual machine
- Type `python` into the shell:

```
$ vagrant ssh
vagrant@pyladies: ~$ python
Python 2.7.3 (default, Sep 26 2013, 20:03:06)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license"
for more information.
>>>
```

# Let's try Python

```
>>> x = 1
>>> type(x)
<type 'int'>
>>> y = 2
>>> x + y
3
>>> def add_numbers(x, y):
...     return x + y
...
>>> add_numbers(4, 5)
9
>>>
```

# Let's try Python

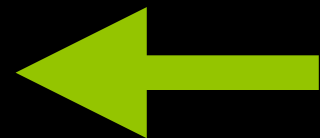
```
>>> x = 1 ← x is a variable
>>> type(x)
<type 'int'>
>>> y = 2
>>> x + y
3
>>> def add_numbers(x, y):
...     return x + y
...
>>> add_numbers(4, 5)
9
>>>
```

# Let's try Python

```
>>> x = 1
```

```
>>> type(x)
```

```
<type 'int'>
```



**x has a data type of “int”  
or integer**

```
>>> y = 2
```

```
>>> x + y
```

```
3
```

```
>>> def add_numbers(x, y):
```

```
...     return x + y
```

```
...
```

```
>>> add_numbers(4, 5)
```

```
9
```

```
>>>
```



# Let's try Python

```
>>> x = 1
>>> type(x)
<type 'int'>
```

```
>>> y = 2
>>> x + y
```

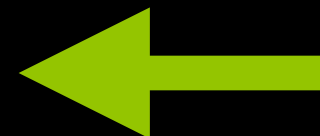
```
3
```

```
>>> def add_numbers(x, y):
...     return x + y
...
```

```
>>> add_numbers(4, 5)
```

```
9
```

```
>>>
```



**functions begin with  
“def”**

# Let's try Python

```
>>> x = 1
>>> type(x)
<type 'int'>
>>> y = 2
>>> x + y
3
>>> def add_numbers(x, y):
...     return x + y
...
>>> add_numbers(4, 5)
9
>>>
```

**Note that there are 2 spaces of indentation on the second line**



# Let's try Python

```
>>> pyladies = { 'San Francisco': 'Lynn', 'Stockholm': 'Oxana' }
>>> type(pyladies)
<type 'dict'>
>>> pyladies['New York'] = 'Katherine'
>>> pyladies
{'San Francisco': 'Lynn', 'Stockholm': 'Oxana', 'New York':
'Katherine'}
>>> pyladies.keys()
['San Francisco', 'Stockholm', 'New York']
>>> pyladies.values()
['Lynn', 'Oxana', 'Katherine']
>>>
```

# Let's try Python

```
>>> pyladies.values()
['Lynn', 'Oxana', 'Katherine']
>>> names = pyladies.values()
>>> names
['Lynn', 'Oxana', 'Katherine']
>>> type(names)
<type 'list'>
>>> names[0]
'Lynn'
>>> type(names[0])
<type 'str'>
```

# Let's try Python

```
>>> exit()
```

# Let's get started

```
vagrant@pyladies: ~$ pip freeze
vagrant@pyladies: ~$ python
>>> import django
ImportError: No module named django
>>> exit()
vagrant@pyladies: ~$ mkvirtualenv test
(test) vagrant@pyladies: ~$ pip install django
(test) vagrant@pyladies: ~$ pip freeze
(test) vagrant@pyladies: ~$ python
>>> import django
>>> exit()
(test) vagrant@pyladies: ~$ deactivate
vagrant@pyladies: ~$ rmvirtualenv test
```

# Commands to remember within terminal

- To create a new virtual environment:  
**syntax:** `$ mkvirtualenv $VIRTUALENV_NAME`  
**example:** `$ mkvirtualenv MyBlog`
- To activate a previously-made virtual environment:  
**syntax:** `$ workon $VIRTUALENV_NAME`  
**example:** `$ workon MyBlog`
- To install python packages/modules within the activated virtual environment:  
**syntax:** `$ pip install $PACKAGE_NAME`  
**example:** `$ pip install django`
- To turn off/deactivate a virtual environment that is currently in use:  
**command:** `$ deactivate`
- To remove a virtual environment entirely:  
**syntax:** `$ rmvirtualenv $VIRTUALENV_NAME`  
**example:** `$ rmvirtualenv MyBlog`

# Starting a Django Project

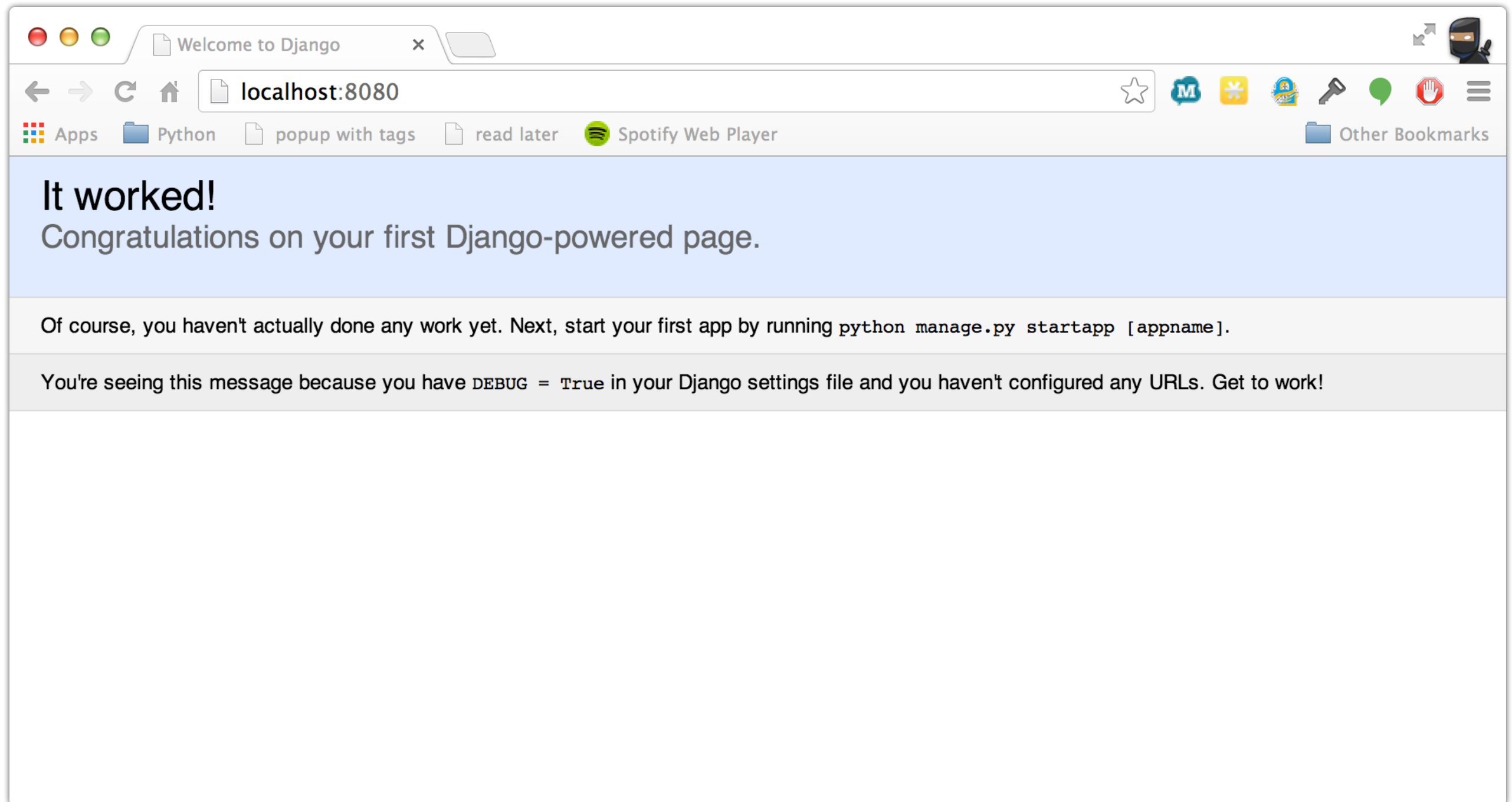
```
v@pyl:~$ cd DjangoProj
v@pyl:~DjangoProj/$ mkvirtualenv Blog
(Blog)v@pyl:~DjangoProj/$ pip install django
(Blog)v@pyl:~DjangoProj/$ django-admin.py startproject MySite
(Blog)v@pyl:~DjangoProj/$ ls
MyBlog
(Blog)v@pyl:~DjangoProj/$ ls MySite
manage.py      MyBlog
(Blog)v@pyl:~DjangoProj/$ ls MySite/MySite
__init__.py    settings.py    urls.py       wsgi.py
```



# Running the Django Server

```
(Blog)v@pyl:~DjangoProj$ cd MySite  
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py runserver 0.0.0.0:8000
```

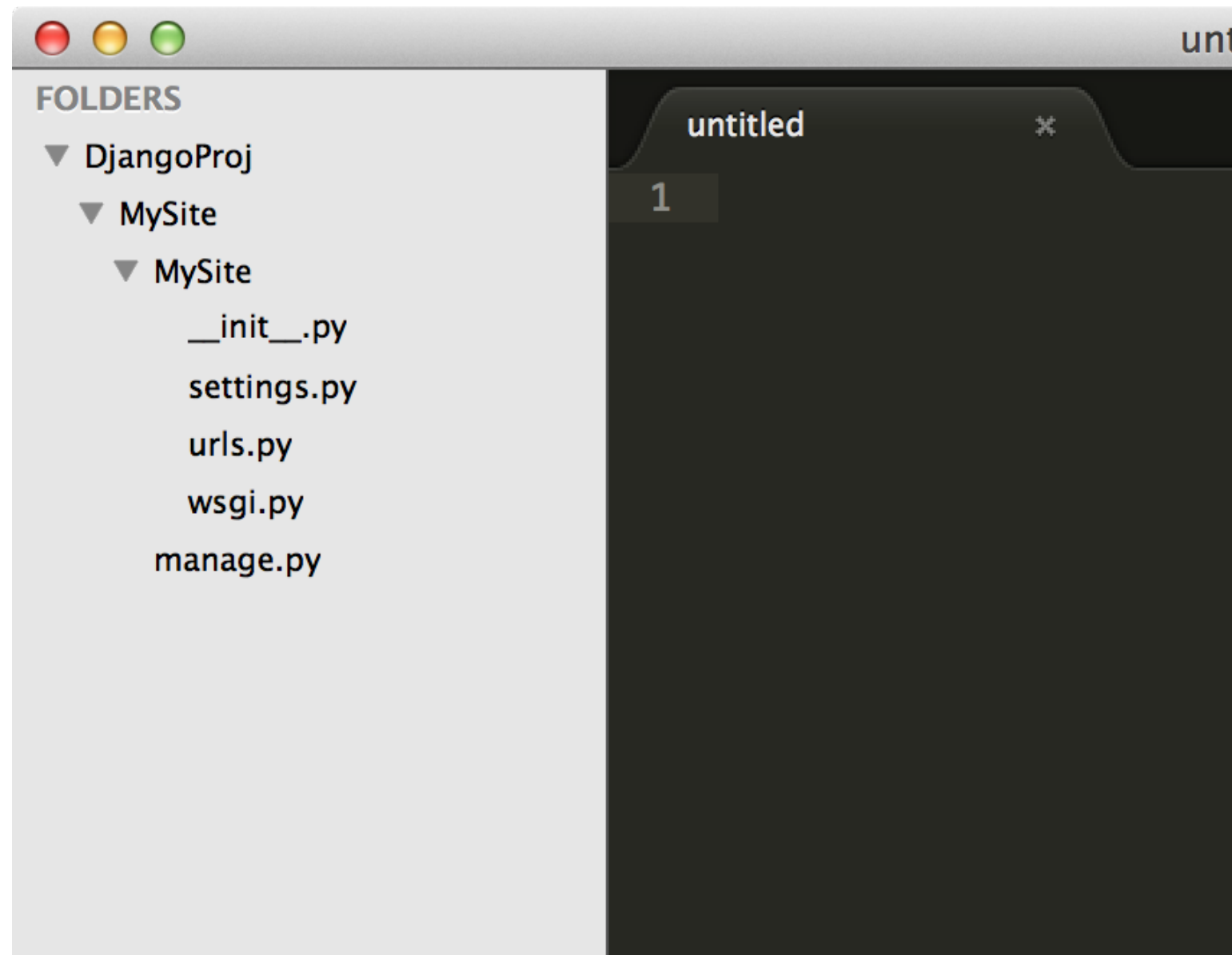
# Running the Django Server



# Editing files locally

- On your local machine (not the virtual machine), open up the “DjangoProj” folder in Sublime Text (or other text editor).
- When you expect the “DjangoProj” folder in Sublime Text, you should see “MySite” and the other files.

# Editing files locally



# Take a look at settings.py (no need to remember all of this)

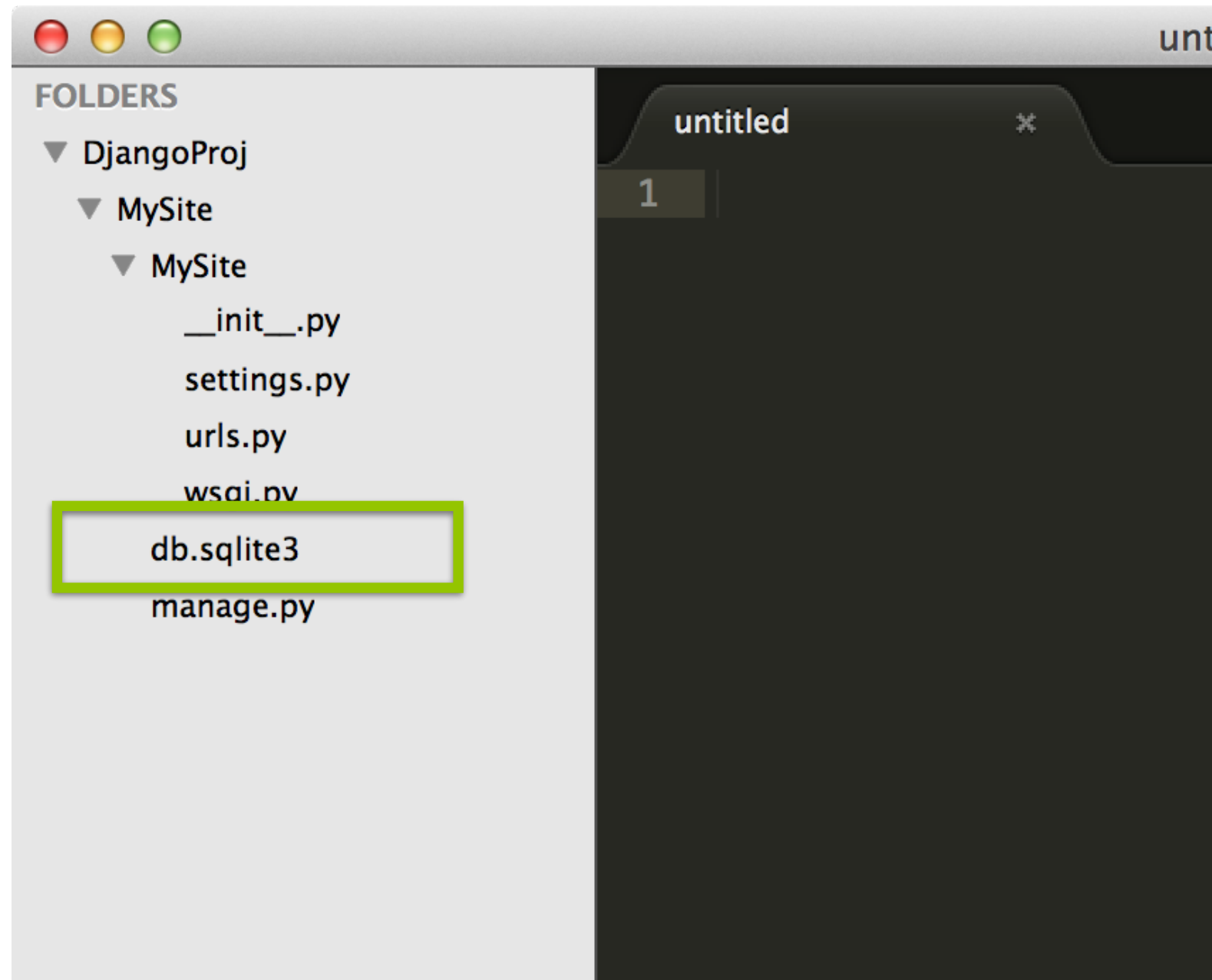
- imports the “os” module, and builds a BASE\_DIR off of it
- has a super-duper secret SECRET\_KEY made up of a string of characters
- a few variables for debugging
- a tuple of INSTALLED\_APPS
- another tuple for MIDDLEWARE\_CLASSES
- URL configuration with ROOT\_URLCONF
- WSGI configuration with WSGI\_APPLICATION
- a dictionary for DATABASES (for instance, you can have one for development, testing, and production)
- language & internationalization configuration
- where Django can find static assets (e.g. HTML, CSS files)

# Sync the database

```
(Blog)v@pyl:~DjangoProj/MyBlog$ python manage.py syncdb
```

Follow the prompts

# Sync the database

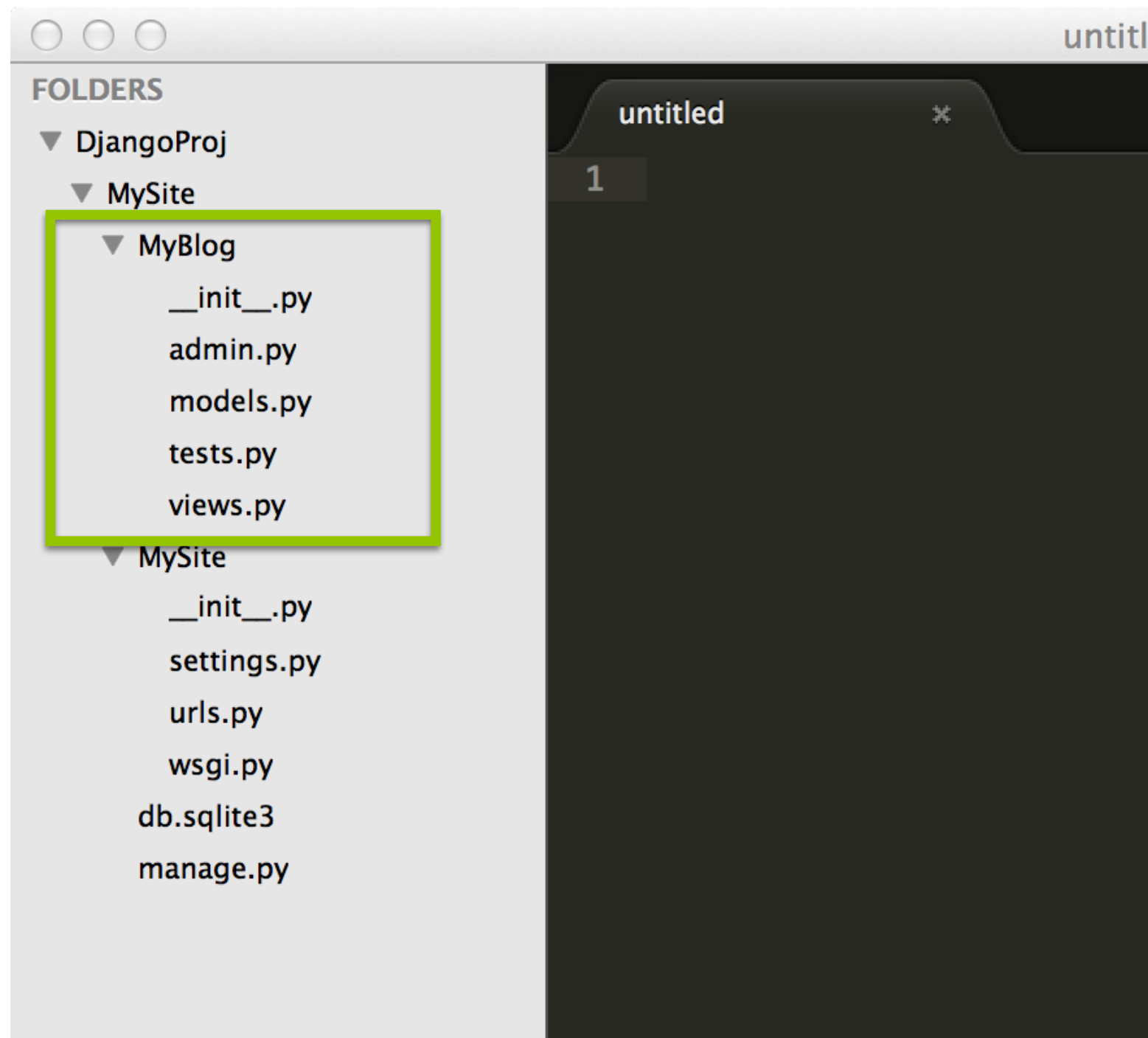


# Startapp

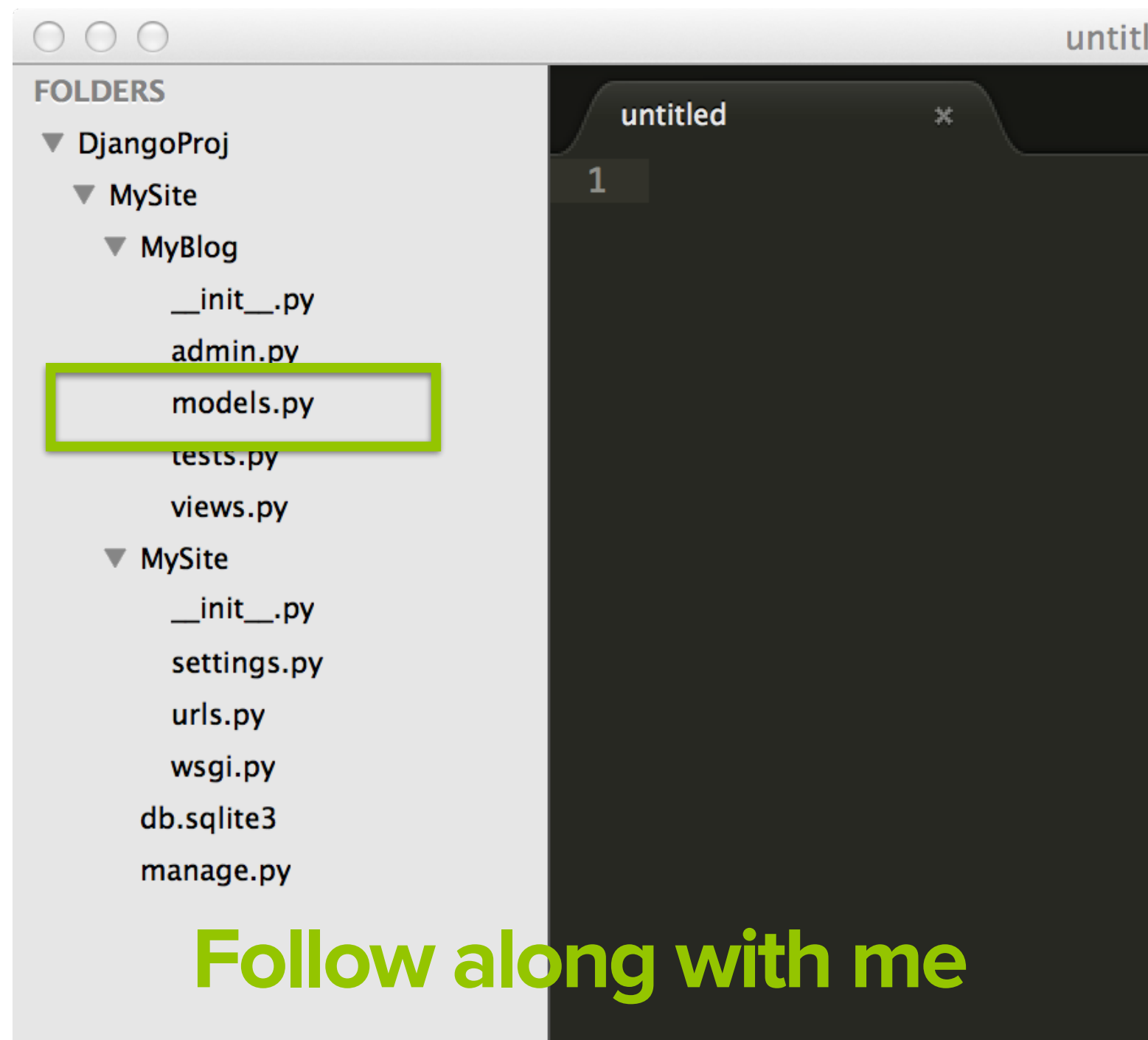
```
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py startapp MyBlog  
(Blog)v@pyl:~DjangoProj/MySite$ cd MyBlog  
(Blog)v@pyl:~DjangoProj/MySite/MyBlog$ ls
```



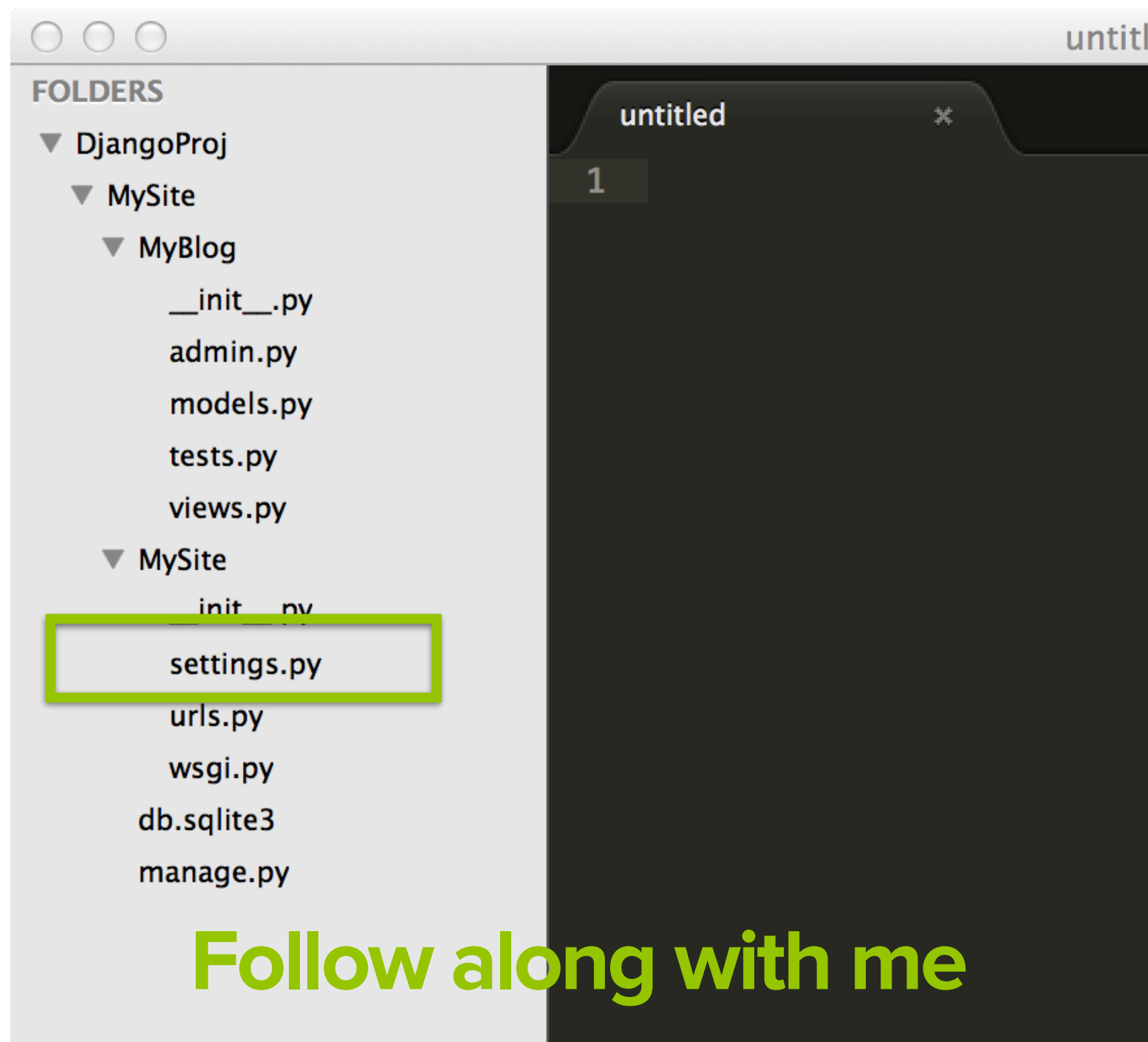
# Startapp



# Open up models.py



# Return to settings.py

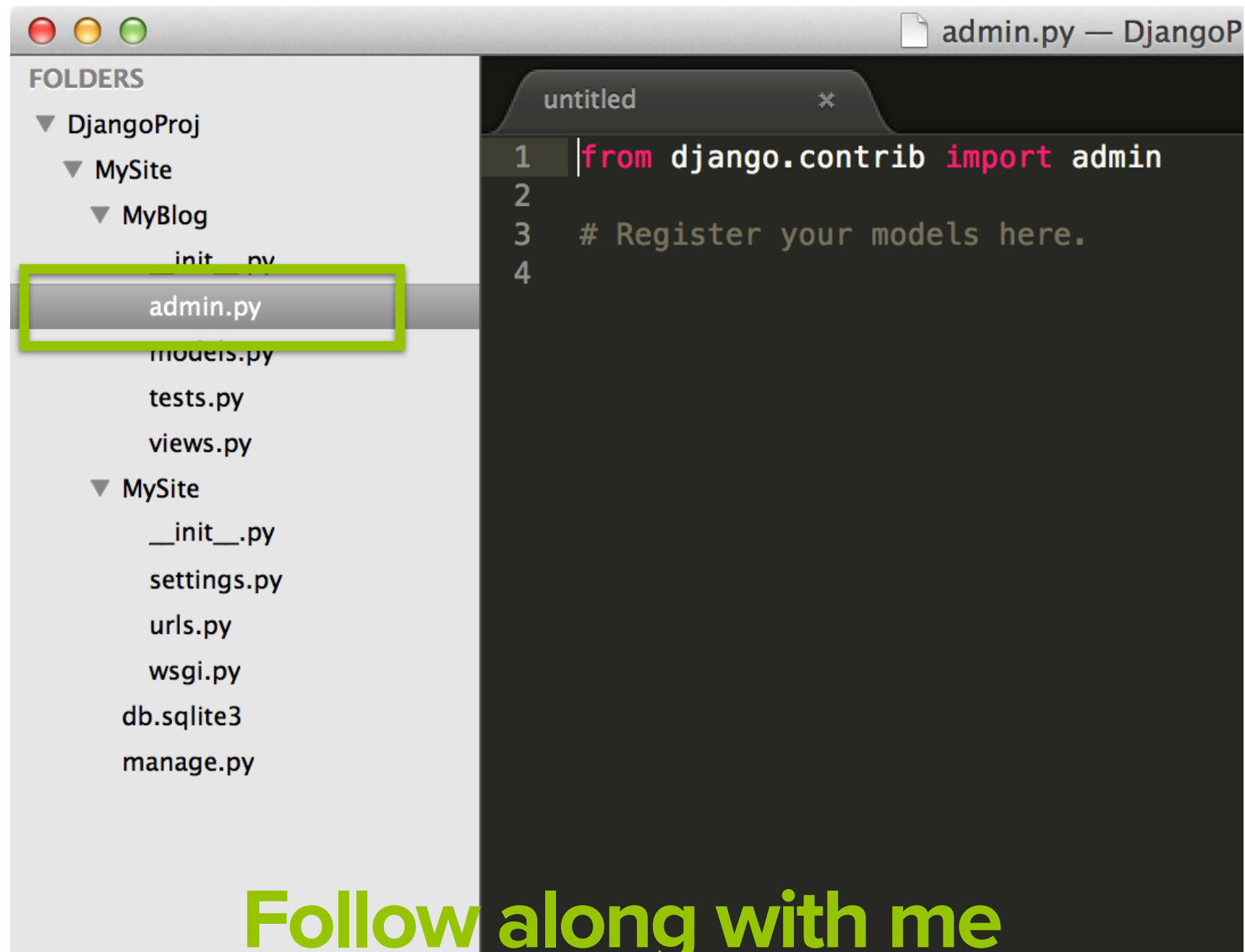


# Syncing the database with our new models

```
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py syncdb  
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py shell  
>>>
```

Follow along with me

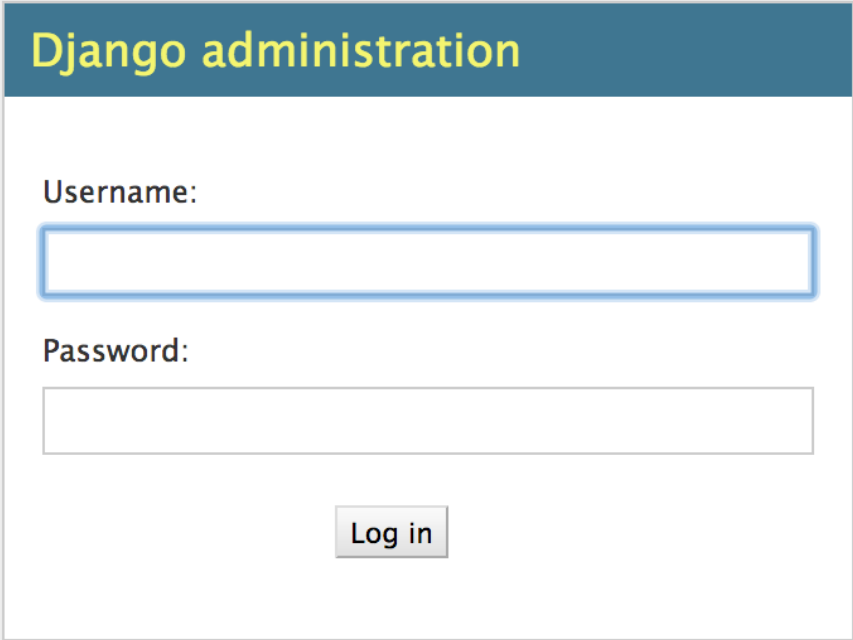
# Creating the admin



# Checking out the admin

```
(Blog)v@pyl:~DjangoProj/MySite/MyBlog$ cd ..  
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py runserver 0.0.0.0:8000
```

# Navigate to your Admin site again



The image shows a screenshot of the Django administration login page. It features a dark blue header with the text "Django administration" in yellow. Below the header, there are two input fields: "Username:" and "Password:". The "Username:" field is highlighted with a blue border. Below the "Password:" field, there is a "Log in" button.

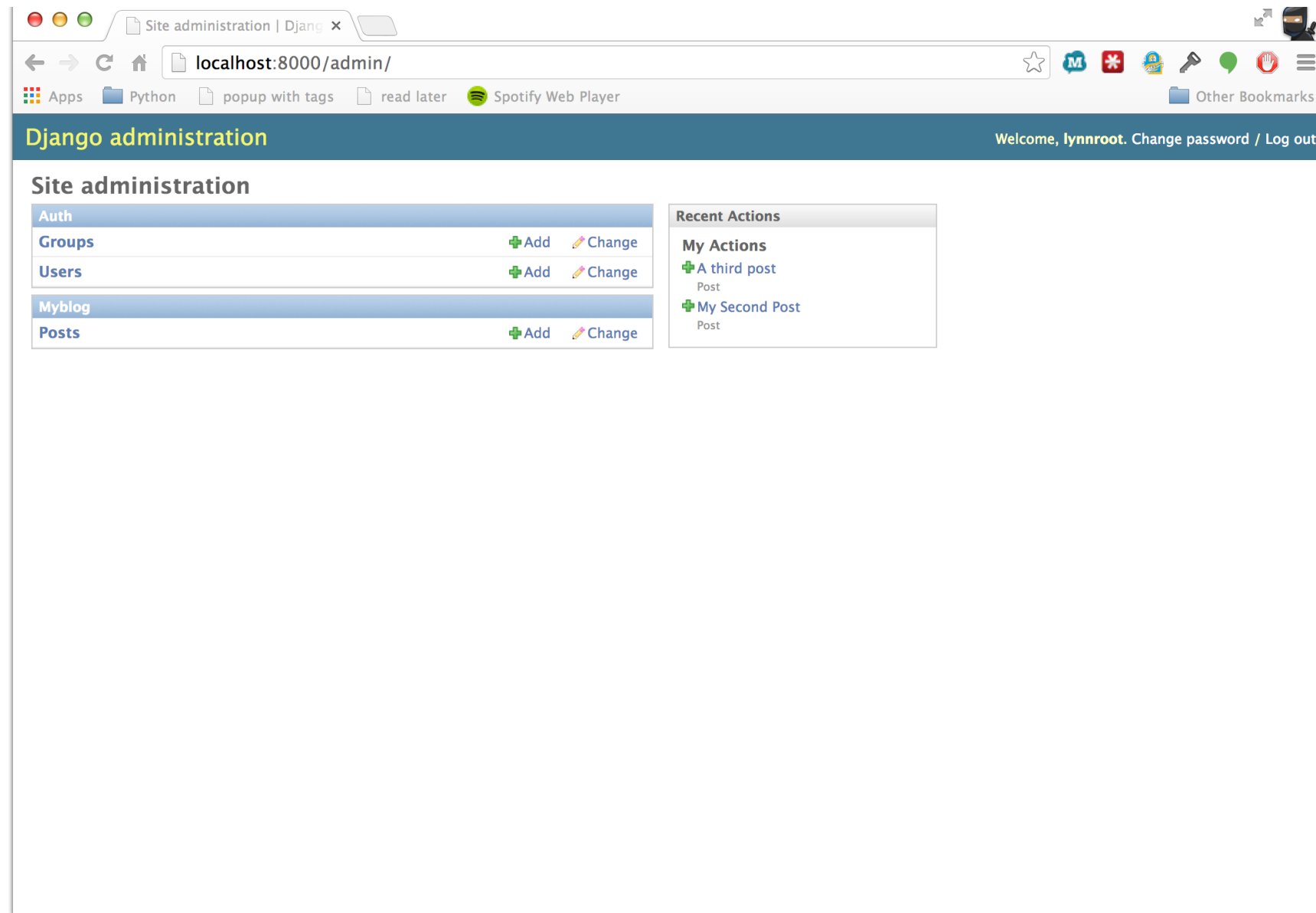
Django administration

Username:

Password:

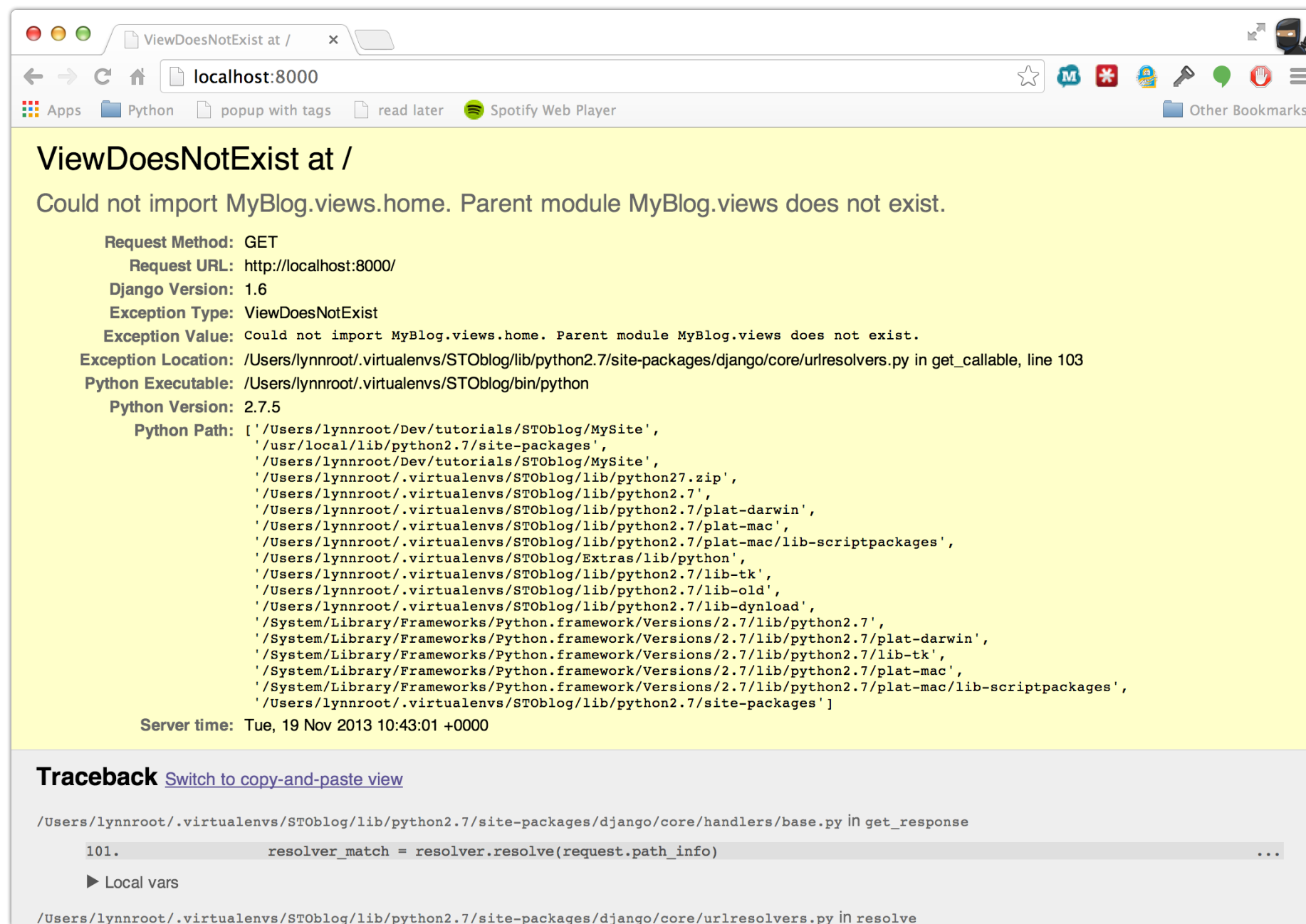
Log in

# Navigate to your Admin site again

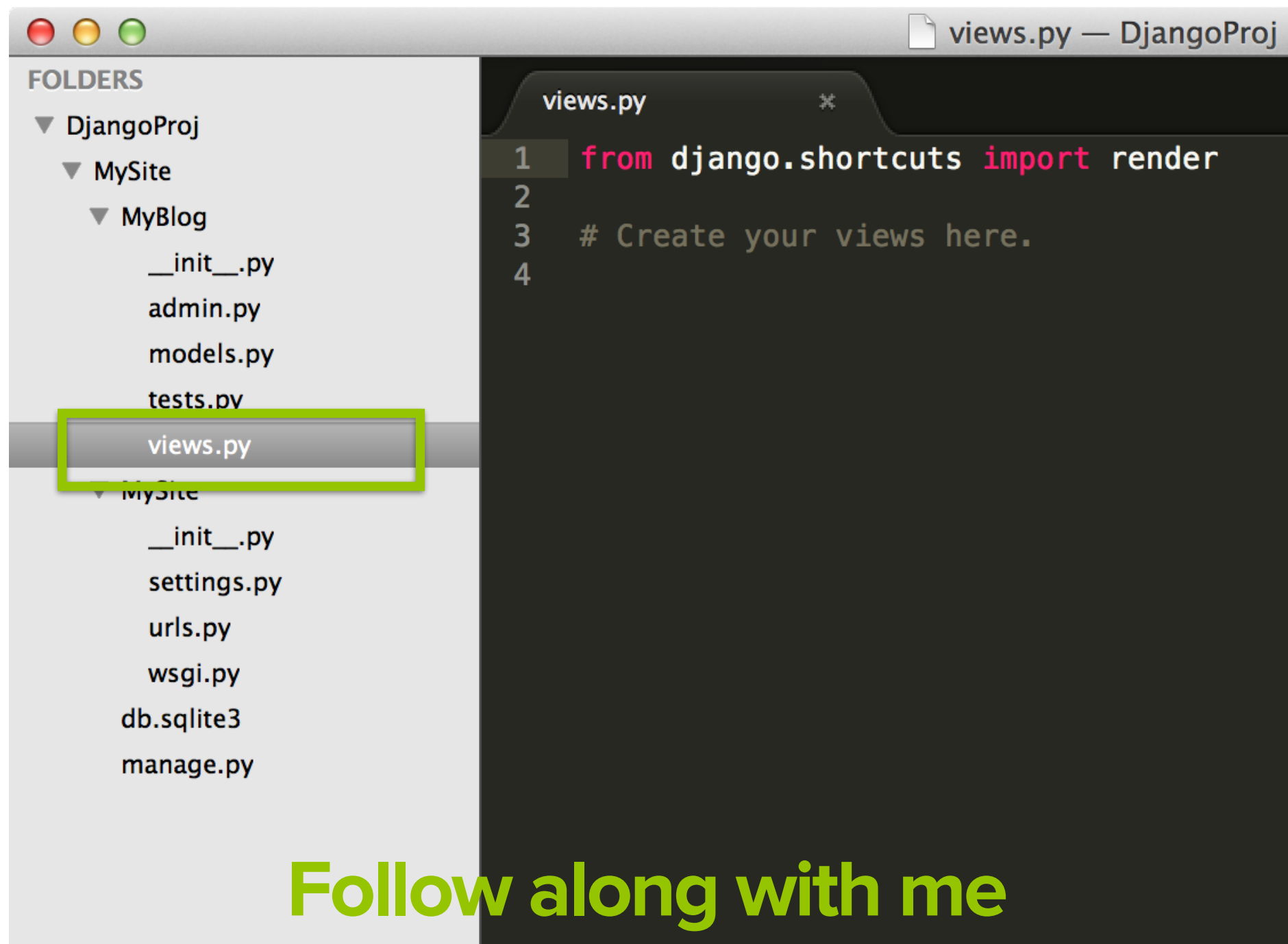




# Navigate to your the Website



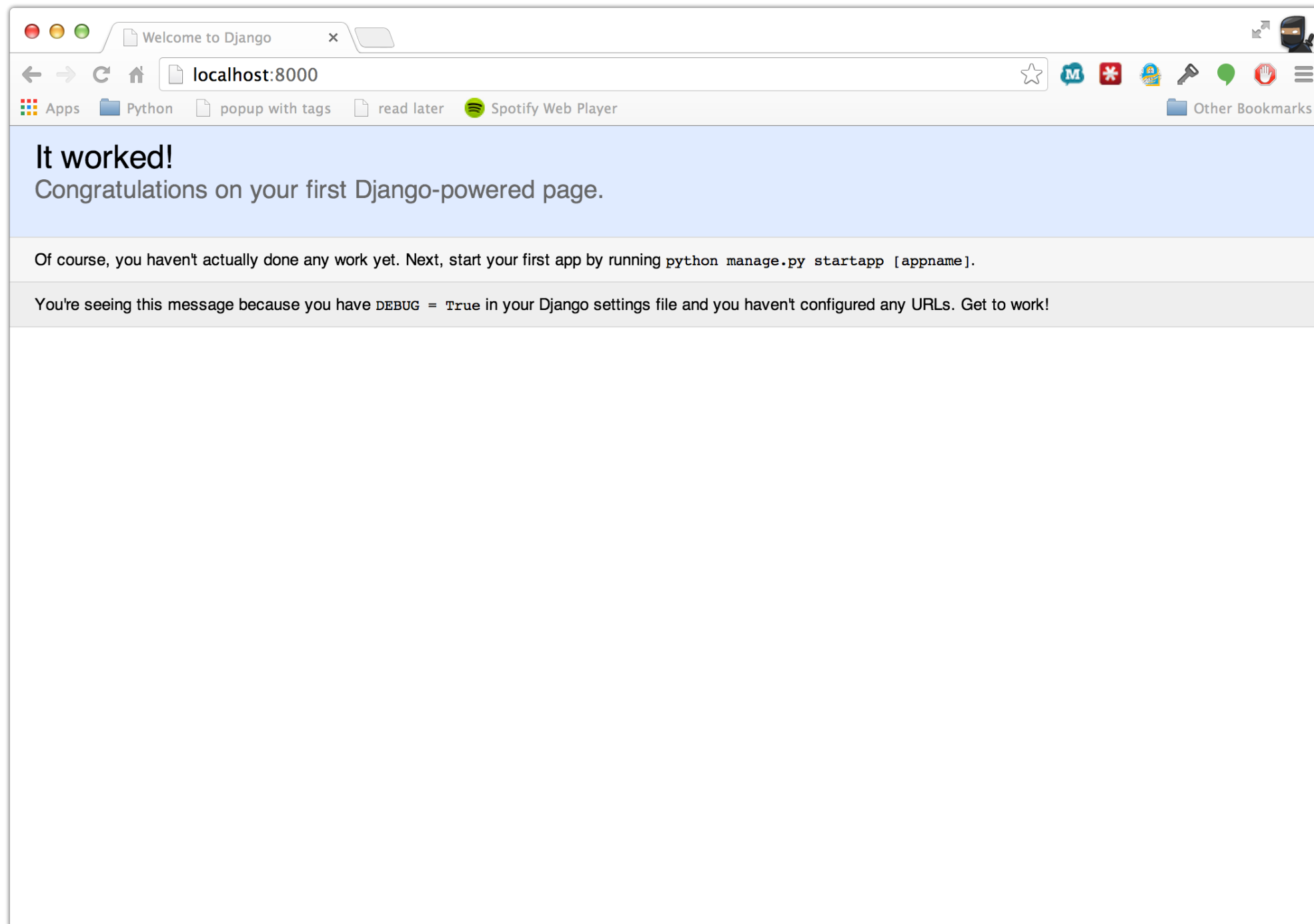
# Open up views.py



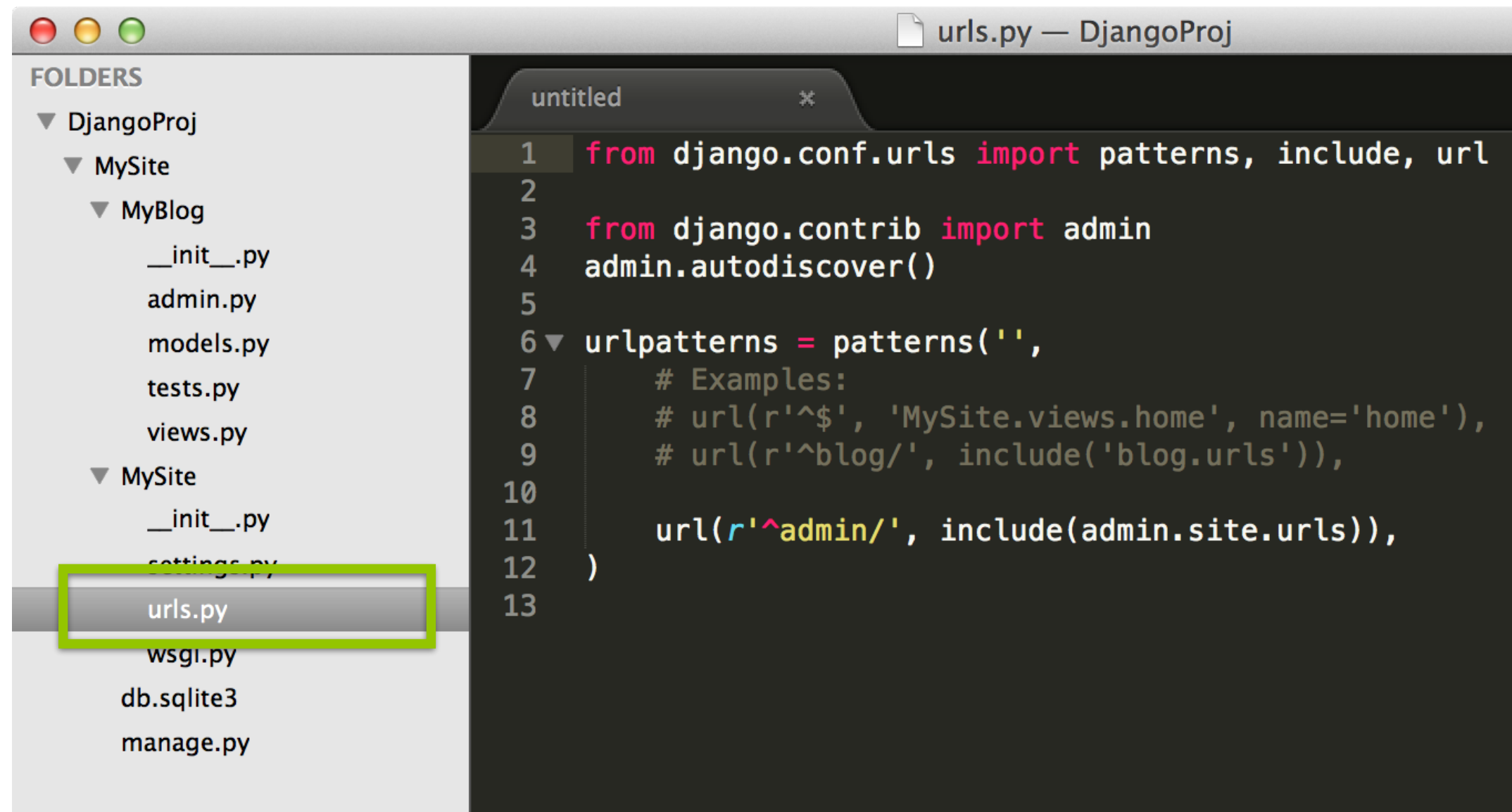
# Kick up the server again!

```
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py runserver 0.0.0.0:8000
```

# Navigate to your the Website



# Open up urls.py

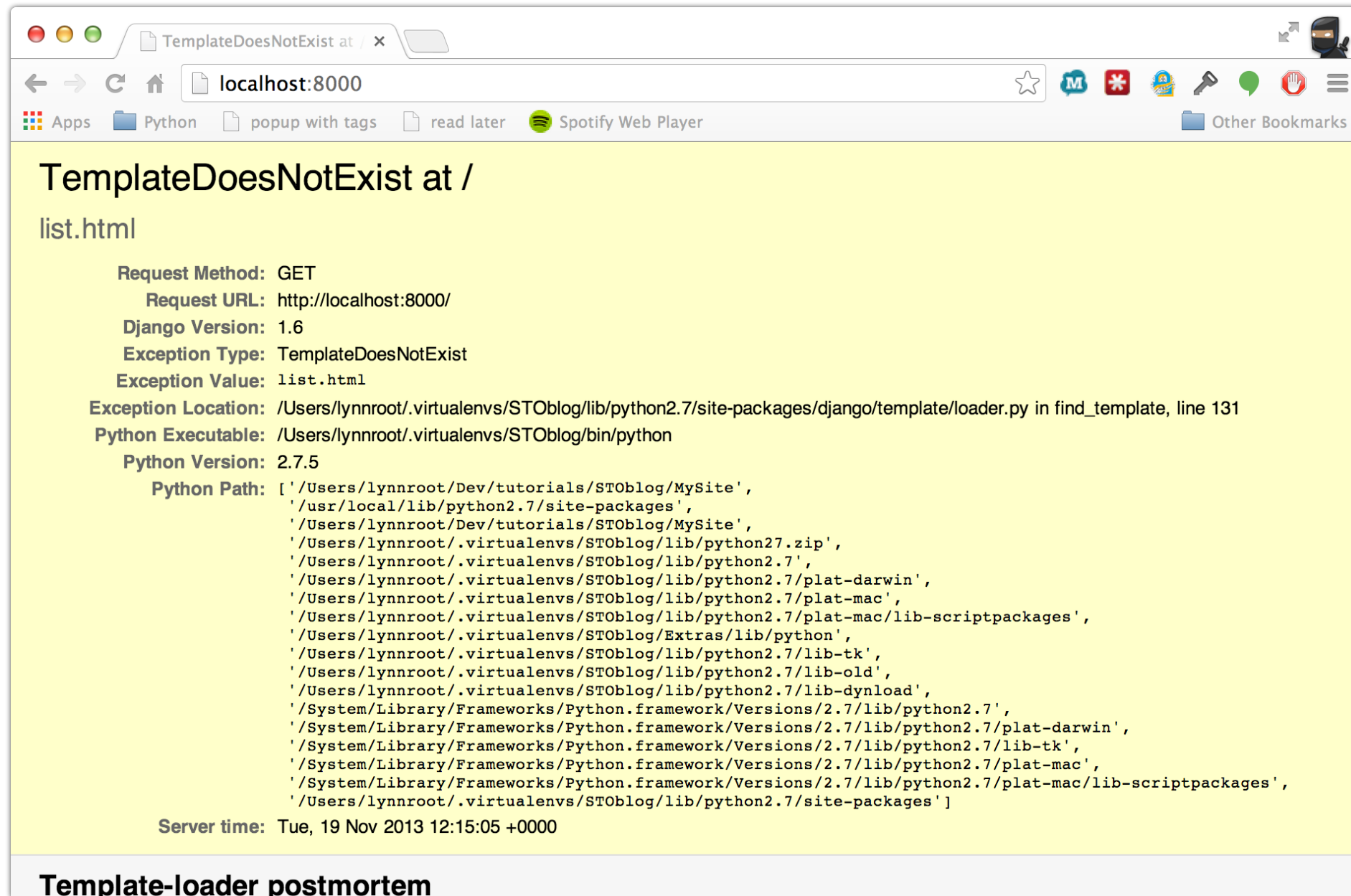


Follow along with me

# Kick up the server again!

```
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py runserver 0.0.0.0:8000
```

# Navigate to your the Website



# Making MOAR directories

```
(Blog)v@pyl:~DjangoProj/MySite/MyBlog$ mkdir static  
(Blog)v@pyl:~DjangoProj/MySite/MyBlog$ mkdir templates
```



# Getting template files and CSS files

- Navigate here: <http://git.io/ocBtDw>
- For the first file “**base.html**” save it under “**MySite/MyBlog/templates**”
- For the second file “**list.html**” save it under “**MySite/MyBlog/templates**”
- For the third file “**post.html**” save it under “**MySite/MyBlog/templates**”
- For the last file “**style.css**” save it under “**MySite/MyBlog/static**”

# Kick up the server again!

```
(Blog)v@pyl:~DjangoProj/MySite$ python manage.py runserver 0.0.0.0:8000
```

# Navigate to your the Website

