Web Programming in Python with Django

IAP 2013

Student Information Processing Board
Luke O'Malley '14

Overview

1. Install Django
2. Website Architecture
3. The “10,000 foot view” of Django
4. Running a Django Server Locally
5. Django Project Structure
6. Adding a Web Page, Creating a Model, and Developing an API
7. Creating a Django App
8. Admin Capabilities, Templating, and Form Generation
9. scripts.mit.edu

Class Material

mit.edu/omalley1/django

Install Django

SIPB: Django Web Programming, IAP 2013

Django is a Python module:

```
<your terminal>$ pip install django
```

```
<your terminal>$ easy_install django
```

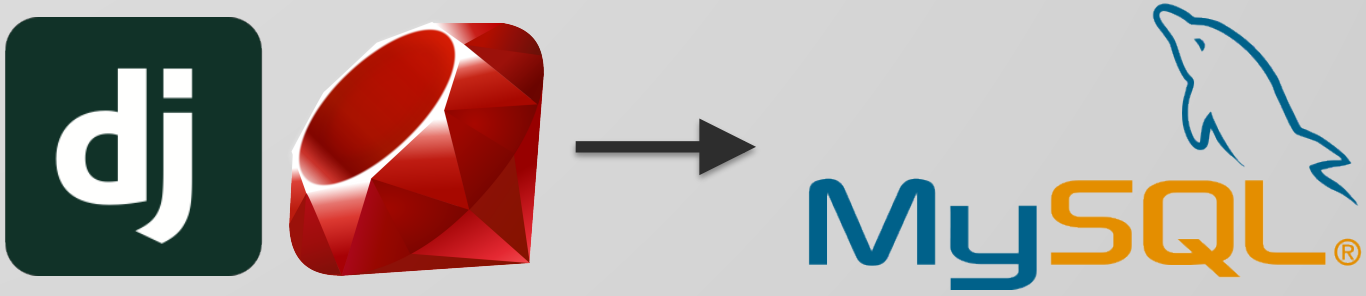
Websites and Their Pieces

Front-end:

- Broadly, it is what the user interacts with
- Where data is entered and displayed
- Often sends information to backend for processing and storage

Back-end:

- Broadly, it receives data from front-end and processes and stores it
- Responsible for “serving” content
- Often composed of a database and management layer



frameworks database

The “10,000 foot view” of Django

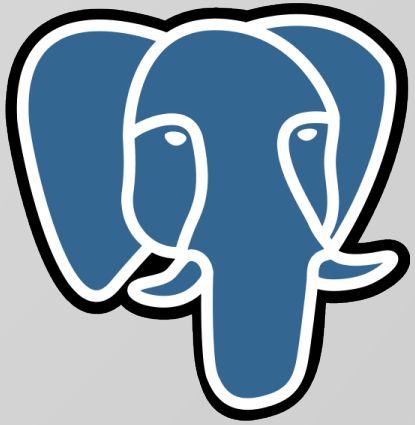
What is Django?

What is Django?

- Bridges the gap between what the user sees and the database
- Databases are hard, but Django makes it easy to work with them
- Databases include:

MySQL®

ORACLE®



Why Django?

Why Django?

- Python, easy to read and understand
- Don’t Repeat Yourself (DRY) Philosophy!
- The community has done a lot of the thinking for you, including security

How Does Django Work?

Model-Template-View (MTV)

- Model
  - Anything dealing with data and its representation (i.e. a user or a car or even data validation)
- Template
  - How data is displayed, what it looks like, this is the presentation layer
- View
  - What data is presented to the template, the control logic, bridging models and template

Django gives you:

- Object-relational mapper
  - Define Python classes
  - Rich way of interacting with database
- Automatic admin interface
  - Don’t waste your time creating an admin page, Django does this for you
- Elegant URL design
  - Regex matching
- Templating system
  - Fill in web pages on the fly!

Diving in!

SIPB: Django Web Programming, IAP 2013

In your terminal type, from the website directory:

```
<your terminal>$ python manage.py runserver

Validating models...

0 errors found
Django version 1.4, using settings 'rsvp.settings'
Development server is running at http://
127.0.0.1:8000/
Quit the server with CONTROL-C
```

Open your browser and visit:

```
localhost:8000
```

SIPB: Django Web Programming, IAP 2013

To start your own project later on:

```
<your terminal>$ django-admin.py startproject <name>
```

Creates following project structure:

```
<name>/
    manage.py
    <name>/
        __init__.py
        settings.py
        urls.py
        wsgi.pyls
```

SIPB: Django Web Programming, IAP 2013

Our Django project:

```
event/ <- developer added app
    __init__.py
    models.py
    test.py
    views.py
manage.py
rsvp/
    __init__.py
    settings.py
    static/ <- developer added
    templates/ <- developer added
    urls.py
    wsgi.pyls
```

SIPB: Django Web Programming, IAP 2013

Adding a module:

```
<your terminal>$ python manage.py startapp <name>
```

I added events by typing (I ran ‘ls’ just to show results):

```
<your terminal>$ python manage.py startapp event
<your terminal>$ ls event/
__init__.py
models.py
tests.py
views.py
```

From URL to Page Render

What happens when a user enters this in the browser?

```
www.your-site-domain.com
```

SIPB: Django Web Programming, IAP 2013

What happens when a user enters this in the browser?

```
www.your-site-domain.com
```

```
event/ <- developer added app
    __init__.py
    models.py
    test.py
    views.py
manage.py
rsvp/
    __init__.py
    settings.py
    static/ <- developer added
    templates/ <- developer added
    urls.py
    wsgi.pyls
```

SIPB: Django Web Programming, IAP 2013

‘rsvp/settings.py’ stores location of our URL config file

```
ROOT_URLCONF = 'rsvp.urls'
```

We then reference ‘rsvp/urls.py’

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

urlpatterns = patterns('',
    url(r'^$', 'event.views.home', name='home'),
)
```

```
www.your-site-domain.com <- matches r’^$’
```

We then reference ‘events/views.py’

```
from django.shortcuts import render

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

Django knows where to find template directory because of ‘rsvp/settings.py’

```
TEMPLATE_DIRS = (
    os.path.join(PROJECT_ROOT, 'templates/'),
)
```

What we've covered so far:

- How to run a server locally
- How to start a Django project
- How to add an app
- How Django goes from URL to web page

Work with Data

SIPB: Django Web Programming, IAP 2013

Define a model for events:

```
from django.db import models

class Event(models.Model):
    title = models.CharField(max_length=128)
    description   = models.TextField()
    date = models.DateTimeField()

    def __unicode__(self):
        return self.title
```

Manually manipulate data using Django shell:

```
<your terminal>$ python manage.py shell
    >>> from event.models import Event
    >>> Event.objects.all()
    []
    >>> e = Event(title="Birthday", date="2013-01-13
    07:00")
    >>> e.save()
```

SIPB: Django Web Programming, IAP 2013

Render a template with data:

```
# View to return all events
def all_events(request):
    events = Event.objects.all()
    return render(request, 'events.html', {'events':
events})
```

Create a template for the events:

```
{% extends "base.html" %}

{% block content %}
      <h1>All Events</h1>

      {% for event in events %}
      <h3>{{ event.title }}</h3>
      <p>{{ event.description }}</p>
      {% endfor %}

{% endblock%}
```

Adding Admin Capabilities

What next?

- work on personal projects
- learn algorithms (CLRS)
  - performance and memory management
- grab a Python book
- active online community, dive in!
- SIPB, you can get involved with out projects!