Introduction to Django and Getting Started

Chapter 1

Objectives

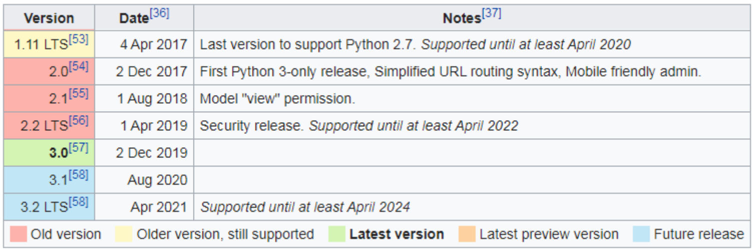
- What is Django?
- Initial setup with PythonAnywhere
- What is virtual environment?
- Understanding Django’s project structure
- Having your django site live on the Internet

What is Django?

- The world of Python web frameworks is full of choices. Django, Flask, Pyramid, Tornado, Bottle, Diesel, Pecan, Falcon, and many more are competing for developer mindshare.
- **Django** is a free and open source web framework, written in Python that encourages rapid development.
- Django is pronounced JANG-oh. The "D" is silent.
- Basically, it follows the MVC (Model-view-controller) pattern, with its own modification to be called the MTV framework (Model-Template-View)
- Django provides all basic features that are part of a generic web application: authentication, security and data management.
- Includes ORM that supports many databases – Postgresql, MySQL, Oracle, SQLite.

What is Django? (cont'd)

- Named after famous Guitarist "Django Reinhardt"
- Developed by Adrian Holovaty and Simon Willison
- Open sourced in 2005
- First Version released September 3, 2008



Version	Date[36]	Notes[37]
1.11 LTS[53]	4 Apr 2017	Last version to support Python 2.7. *Supported until at least April 2020*
2.0[54]	2 Dec 2017	First Python 3-only release, Simplified URL routing syntax, Mobile friendly admin.
2.1[55]	1 Aug 2018	Model "view" permission.
2.2 LTS[56]	1 Apr 2019	Security release. *Supported until at least April 2022*
3.0[57]	2 Dec 2019	
3.1[58]	Aug 2020	
3.2 LTS[58]	Apr 2021	*Supported until at least April 2024*

🟥 Old version 🟨 Older version, still supported 🟩 **Latest version** 🟧 Latest preview version 🟦 Future release

- The textbook uses Django(2.0)and Python (3.6x) for demonstrations.

Django

- After nearly 14 years of growth, Django now not only runs news publishing companies like the Washington Post, but is also running all or part of major global enterprises like Instagram and Bitbucket.
- For a list of websites powered by Django, can visit https://djangosites.org/ .

Common tasks supported by Django

- Django supports for common tasks in web development:
  - user authentication
  - templates, routes, and views
  - admin interface
  - robust security
  - support for multiple database backends
  - and much much more

PythonAnywhere

- We will be using PythonAnywhere
- PythonAnywhere is a tool for us to host, run and code Python in the cloud.
- You can register a free beginner's account, the name of which will be used for your blog's URL in the form yourusername.pythonanywhere.com.
- **Please use your student ID, P18XXXXX as the account name so that your blog's URL will take the form, P18XXXXX.pythonanywhere.com.**
- Refer to the details of Lab 1 on the steps to setup PythonAnywhere to have a django site live and on the Internet, **yourusername.pythonanywhere.com** from a browser.

The Bash Console (PythonAnywhere)

- The Bash console is a *textual* way to interact with the system, just as the 'desktop', in conjunction with your mouse, is the *graphical* way to interact your system.
- Some common commands:
  - cd (change down a directory)
  - cd .. (change up a directory)
  - ls (list files in your current directory)
  - mkdir (make directory)
  - zip -r myzipfile my_folder_name (to create a zip file)

What is virtual environment?

- You might be running several Python applications that require a different version to run. For example, you want to switch to the new version of Django, but still want to maintain your Django 1.11 project.
- The solution is to use virtual environments.
- Virtual environments (virtualenv or venv) allow multiple installations of Python and their relevant packages to exist together in harmony.
- In the Bash console, the command to create a virtual environment called django2 and install Django in it is as follows:

```
mkvirtualenv django2 --python=/usr/bin/python3.6
pip install django
```

Creating a Django project

- You should now see parentheses on your bash console with the name of virtual environment activated. For example, something like

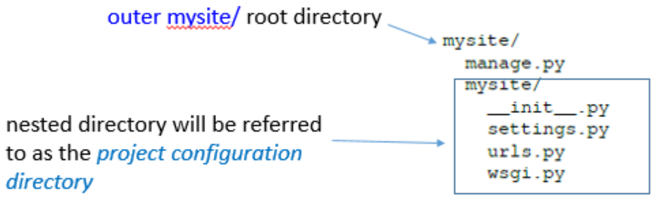
```
(django2)17:18 ~ $
```

- This means it’s working! Create a new Django project called mysite with the following command.

```
(django2)17:18 ~/django_projects $ django-admin startproject mysite
```

Django project structure

- If you just run django-admin startproject mysite then by default Django will create the following directory structure.



outer mysite/ root directory

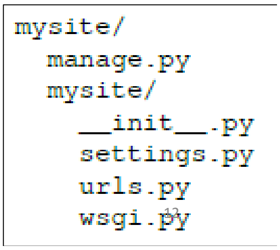
nested directory will be referred to as the *project configuration directory*

```
mysite/
  manage.py
  mysite/
    __init__.py
    settings.py
    urls.py
    wsgi.py
```

- See how it creates a new directory mysite and then within it a manage.py file and a mysite directory.

Django project structure (cont'd)

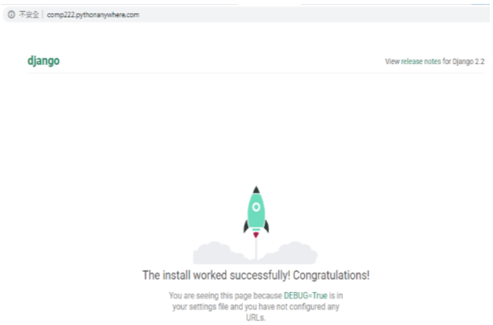
- The outer mysite/ root directory is a container for your project.
  - manage.py, a command-line utility that lets you interact with your Django project.
  - The inner mysite/ directory is the Python package for your project. It's the name you will use to import anything inside it (for example, mysite.urls).
    - mysite/__init__.py, an empty file that tells Python that this directory should be considered a Python package.
    - mysite/settings.py, settings/configuration for this Django project.
    - mysite/urls.py, the URL declarations for this Django project.
    - mysite/wsgi.py, an entry-point for WSGI-compatible web servers to serve your project



```
mysite/
  manage.py
  mysite/
    __init__.py
    settings.py
    urls.py
    wsgi.py
```

Your django site live on the Internet

- Refer to the details of Lab 1 on the steps to setup PythonAnywhere to have a django site live and on the Internet, **yourusername.pythonanywhere.com** from a browser.


不安全 | comp222.pythonanywhere.com
django
View release notes for Django 2.2

The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.