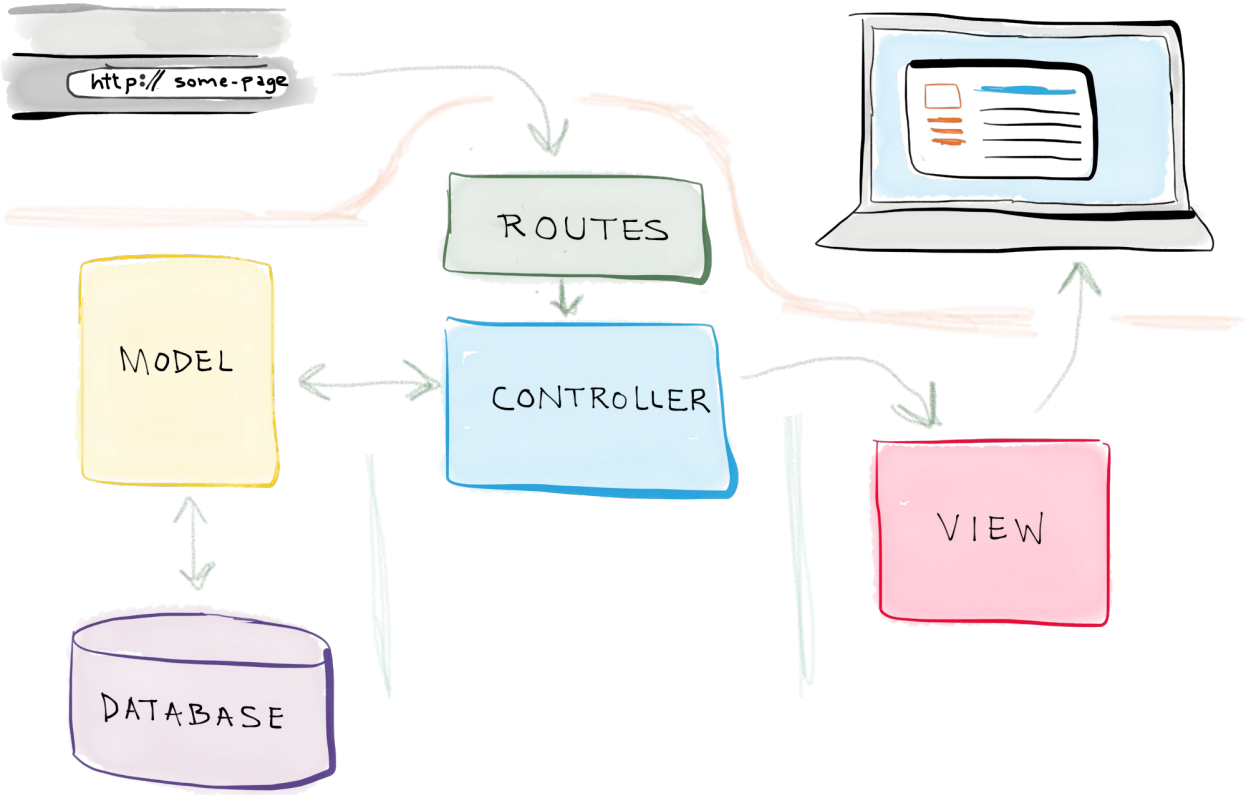
Up and Running in Python: Easy Mode

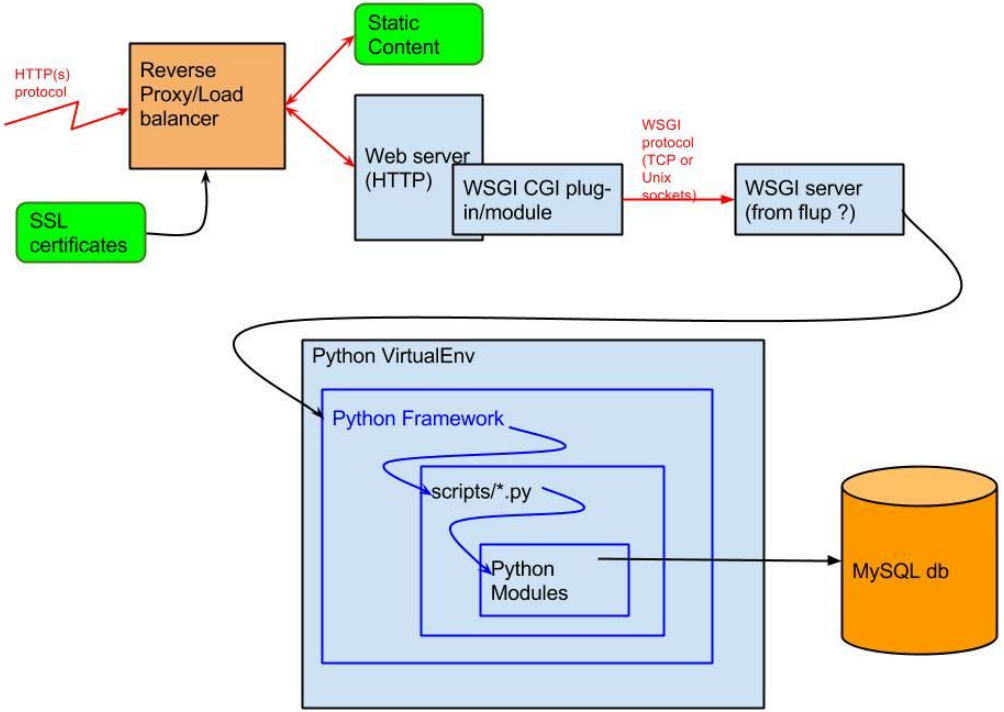
Mary (Nagle)

What you know as a novice...



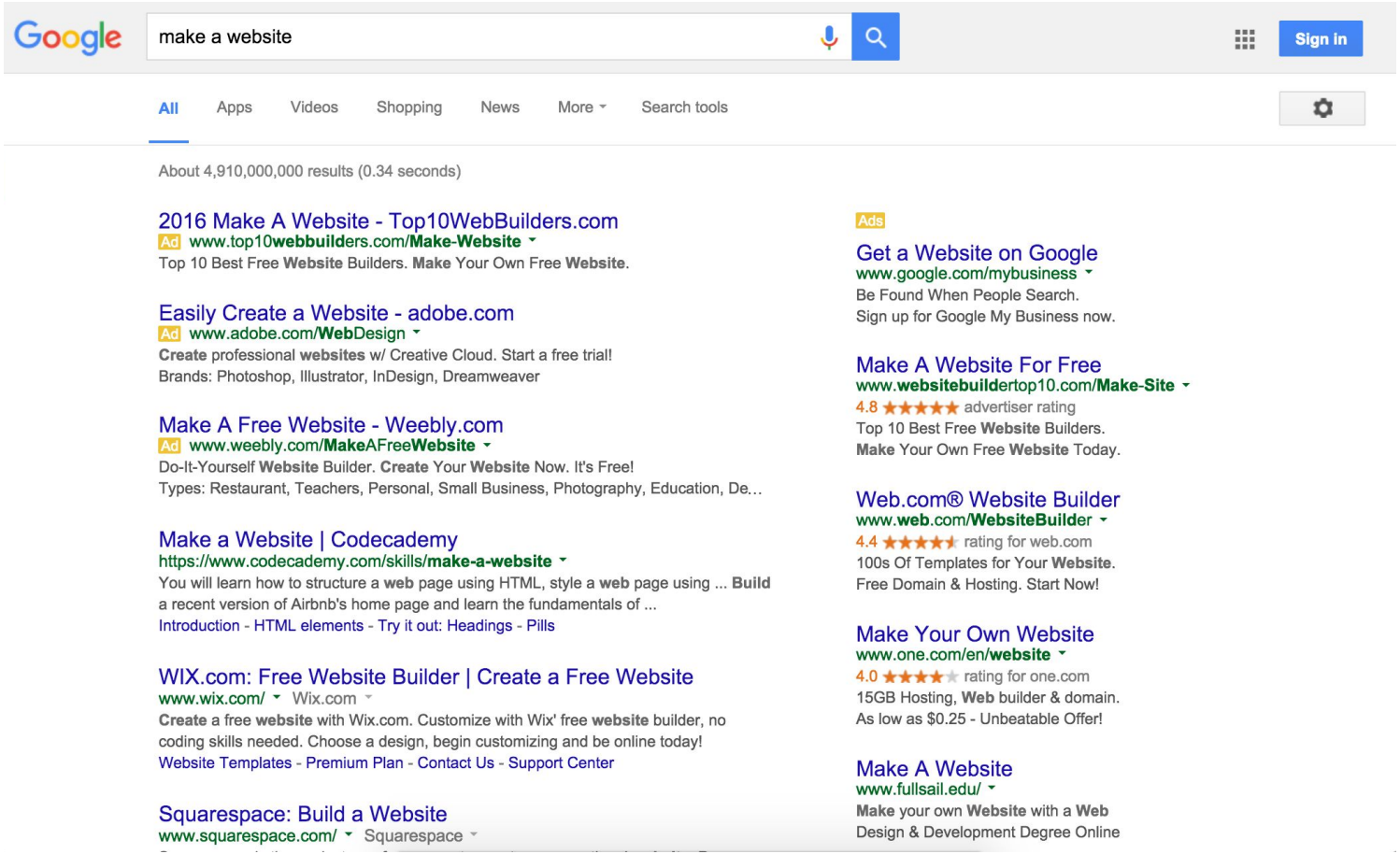
https://realpython.com/images/blog_images/mvc_diagram_with_routes.png

What's actually going on, sorta



Python Modules = Most of the last slide

http://superuser.com/questions/822385/how-does-the-python-web-application-stack-work


Google
make a website
Sign in
All
Apps
Videos
Shopping
News
More
Search tools
About 4,910,000,000 results (0.34 seconds)
2016 Make A Website - Top10WebBuilders.com
Ad www.top10webbuilders.com/Make-Website
Top 10 Best Free Website Builders. Make Your Own Free Website.
Easily Create a Website - adobe.com
Ad www.adobe.com/WebDesign
Create professional websites w/ Creative Cloud. Start a free trial!
Brands: Photoshop, Illustrator, InDesign, Dreamweaver
Make A Free Website - Weebly.com
Ad www.weebly.com/MakeAFreeWebsite
Do-It-Yourself Website Builder. Create Your Website Now. It's Free!
Types: Restaurant, Teachers, Personal, Small Business, Photography, Education, De...
Make a Website | Codecademy
https://www.codecademy.com/skills/make-a-website
You will learn how to structure a web page using HTML, style a web page using ... Build a recent version of Airbnb's home page and learn the fundamentals of ...
Introduction - HTML elements - Try it out: Headings - Pills
WIX.com: Free Website Builder | Create a Free Website
www.wix.com/ Wix.com
Create a free website with Wix.com. Customize with Wix' free website builder, no coding skills needed. Choose a design, begin customizing and be online today!
Website Templates - Premium Plan - Contact Us - Support Center
Squarespace: Build a Website
www.squarespace.com/ Squarespace
Ads
Get a Website on Google
www.google.com/mybusiness
Be Found When People Search.
Sign up for Google My Business now.
Make A Website For Free
www.websitebuildertop10.com/Make-Site
4.8 advertiser rating
Top 10 Best Free Website Builders.
Make Your Own Free Website Today.
Web.com® Website Builder
www.web.com/WebsiteBuilder
4.4 rating for web.com
100s Of Templates for Your Website.
Free Domain & Hosting. Start Now!
Make Your Own Website
www.one.com/en/website
4.0 rating for one.com
15GB Hosting, Web builder & domain.
As low as $0.25 - Unbeatable Offer!
Make A Website
www.fullsail.edu/
Make your own Website with a Web Design & Development Degree Online

Google build a python site from scratch

Sign in

All Videos Shopping Images News More Search tools

About 19,200,000 results (0.40 seconds)

Python Development Tools - VisualStudio.com
Ad www.visualstudio.com/**Python**-IDE
Cpython, PyPy, IronPython & More Free **Python** Tools for Visual Studio

Python from Scratch - Create a Dynamic Website - Envato ...
code.tutsplus.com/.../**python**-from-**scratch**-create-a-dynamic-**website**--net...
Nov 19, 2011 - **Python** from **Scratch**: Object Oriented Programming. We've covered ... So, how do you get started **creating websites** with **Python**? Well, you ...

Python from Scratch - Creating a Dynamic Website - YouTube


https://www.youtube.com/watch?v=bRnm8f6Wavk
Nov 19, 2011 - Uploaded by Tuts+ Code
We've covered a lot of **Python** in the last 4 tutorials. Today, we're going to use everything we've learned so ...

Python and Django Tutorials Building Websites from Scratch ...


https://www.youtube.com/watch?v=HcDcl5RNM90
Jul 30, 2015 - Uploaded by CodeGeek
Python Programmer T-shirt: https://hii.to/Vk5PE7KMg Create projects using Raspberry Pi and **Python** http ...

How to use Python to make websites - Quora
https://www.quora.com/How-do-you-use-**Python**-to-make-**websites** Quora
Become frustrated when Django (even with the available add-on libraries) doesn't fit your intended type of **site** exactly (you aren't **building** a CMS, after all), and ...

Exercise 50: Your First Website - Learn Python The Hard Way

Ads

Make A WordPress Site
www.wordpress.com/**Website**
4.4 ★★★★½ rating for wordpress.com
Create Your Own Premium **Website**.
Superior Live Support. Get Started!

Build a Website for 2016
www.thetop10**sites**.com/**Build_Website**
Compare Software to **Build Website** .
Start Your Own **Site** for Only $0.99!

Build A Site
www.**buildyoursite**.com/**build-website**
Easily **Build** & Manage a **Website**.
Includes Everything - Save 15% Now!

Product Python Developers
resources.kuliza.com/
9+ years 70+ success stories
Get in touch to get free consulting

Build a Webpage: Web.com®
www.web.com/**BuildWebpage**
100s Of Templates for Your **Webpage**.
Free Domain & Hosting. Start Now!

Hire Python Programmers

Say What??

“SQLite is a C library that provides a *lightweight* **disk-based** database that doesn’t require a **separate server process** and allows accessing the database using a *nonstandard variant* of the SQL query language. Some applications can use SQLite for **internal data storage**. It’s also possible to prototype an application using SQLite and then ***port the code*** to a larger database such as PostgreSQL or Oracle.”

https://docs.python.org/2/library/sqlite3.html

Me


#GOALS

Where R Ü Now?

(You)

Your operating system

PyPI - the Python Package Index

pip vs. easy_install

But what is setuptools?

- Support for project dependencies and configuration
- Included in Python

28.1. `distutils` — Building and installing Python modules

...

Most Python users will *not* want to use this module directly, but instead use the cross-version tools maintained by the Python Packaging Authority. In particular, setuptools is an enhanced alternative to `distutils` that provides:

- support for declaring project dependencies
- additional mechanisms for configuring which files to include in source releases (including plugins for integration with version control systems)
- the ability to declare project "entry points", which can be used as the basis for application plugin systems
- the ability to automatically generate Windows command line executables at installation time rather than needing to prebuild them
- consistent behaviour across all supported Python versions

The recommended pip installer runs all `setup.py` scripts with `setuptools`, even if the script itself only imports `distutils`. Refer to the Python Packaging User Guide for more information.

easy_install

- Found in setuptools distribution*
- Given a name, searches for a distribution with that name
- Once found, it downloads the distribution using `setup.py install`
- Checks whether newly installed distribution requires other libraries that are not installed; if so, finds them and installs them
- Installs packages into the running version of python’s site-packages directory

easy_install

- Modifies sys.path - a list of strings that specifies the search path for the currently installed modules
  - sys package = System specific parameters and functions
  - Populated from PYTHONPATH variable
- Allows simultaneous installation of different versions of the same package into a single environment
- More easily installs .egg files

Several stages of `pip install`

1. Identify base requirements. The user supplied arguments are processed here.
2. Resolve dependencies. What will be installed is determined here.
    - Generates metadata on package dependencies using setuptools's setup.py
    - Uses pkg_resources or setup.py `egg_info` to parse and read distribution and version requirements from this metadata
    - Assumes latest version that satisfies given constraints is best

https://pip.pypa.io/en/stable/reference/pip_install/#usage

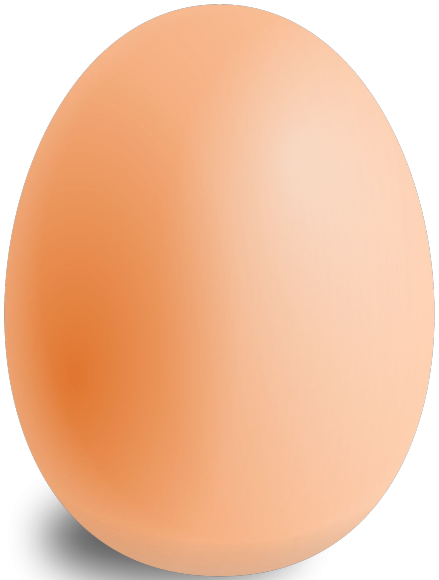
Several stages of `pip install` con't

3. Build wheels. All the dependencies that can be are built into wheels.
    - Wheels or .whl files are intended to replace eggs
4. Install the packages (and uninstall anything being upgraded/replaced).
    - Installs dependencies before their dependents
    - Installs using option `--single-version-externally-managed`, which forces setuptools to install it in a more flat manner than it would with easy_install -- i.e. eggs are a bit more difficult

Pip perk: Allows you to generate a requirements.txt file that specifies all required packages and their corresponding versions

https://pip.pypa.io/en/stable/reference/pip_install/#usage

Eggs vs. Wheels



.egg

vs.

.whl

Wheels

- Versioned
- Consistent installation
- Performance optimizations
- Better caching
- Continuous integration
- **A single wheel can be both Python 2 and 3 compatible**

Several stages of `pip install` con't

3. Build wheels. All the dependencies that can be are built into wheels.
    - Wheels or .whl files are intended to replace eggs
4. Install the packages (and uninstall anything being upgraded/replaced).
    - Installs dependencies before their dependents
    - Installs using option `--single-version-externally-managed`, which forces setuptools to install it in a more flat manner than it would with easy_install -- i.e. eggs are a bit more difficult

Pip perk: Allows you to generate a requirements.txt file that specifies all required packages and their corresponding versions

https://pip.pypa.io/en/stable/reference/pip_install/#usage

Ok, great!

Your operating system

App_1

Table

ge

query.order_by(“age desc”)

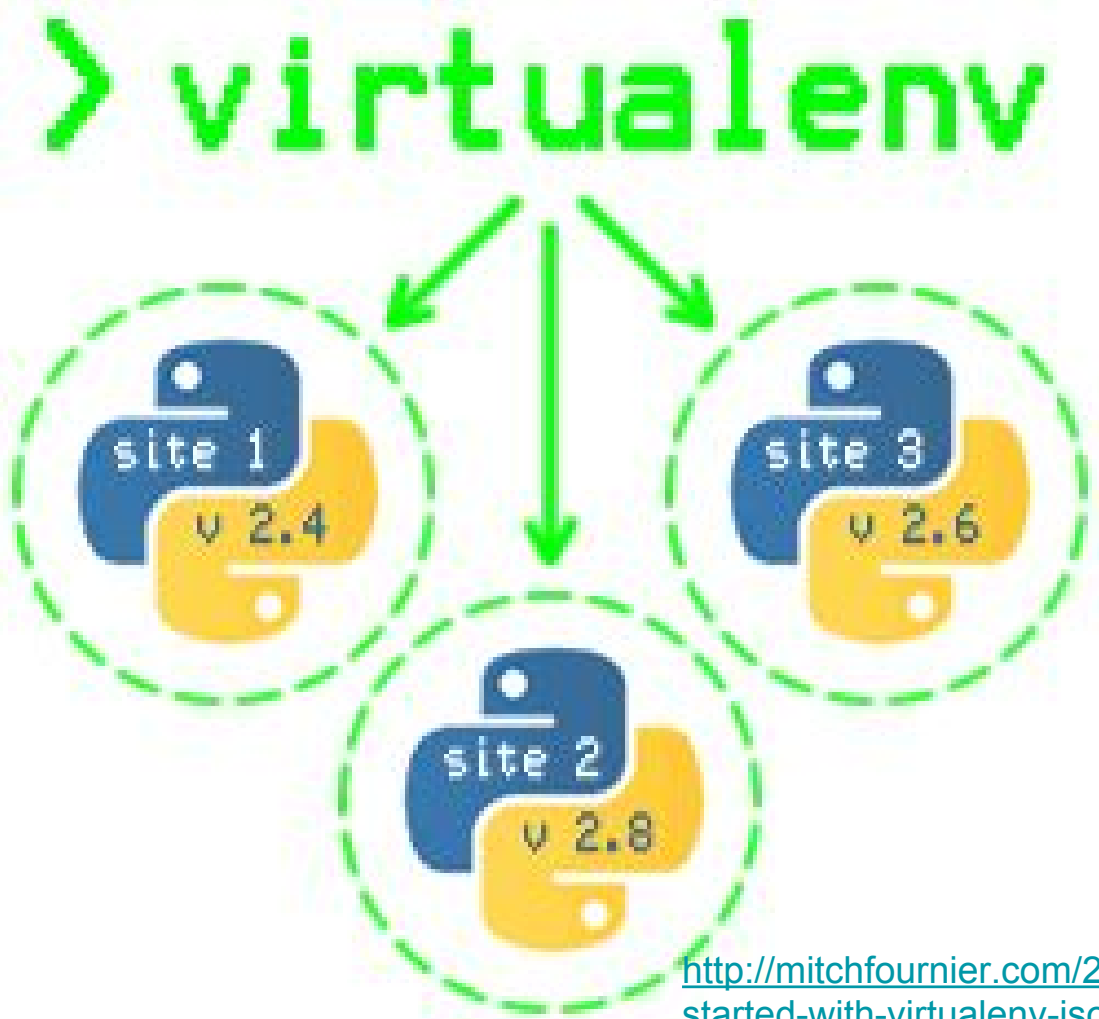
App_2

SQLAlchemy v 1.1.0

```
query = session.query(Rings)
    .filter_by(purpose=’to rule them all’)
```

query.one_or_none()

Virtual Environments!!!!!



http://mitchfournier.com/2010/06/25/getting-started-with-virtualenv-isolated-python-

A bit more granular...

Hooray!

virtualenv

- Works on any system that has python installed
- ~/ENV/lib/ and ~/ENV/include/
  - Contain supporting library files for a new virtualenv python.
- ~/ENV/lib/pythonX.X/site-packages/
  - Where packages installed in this environment will live
- ENV/bin
  - Where executables live - noticeably the copy of python
  - Running a script with #! /path/to/ENV/bin/python would run that script under this virtualenv's python.
- Map chosen directory to your PATH variable in your bash_profile

Virtualenvwrapper, etc.

- Installs pip, python and setuptools upon creation
- Shout out: virtualenvwrapper
  - ex. `workon env_name` vs. `source ~/ENV/env_name/bin/activate`
- venv and pyvenv are built-ins of Python 3 that do essentially the same things as virtualenv

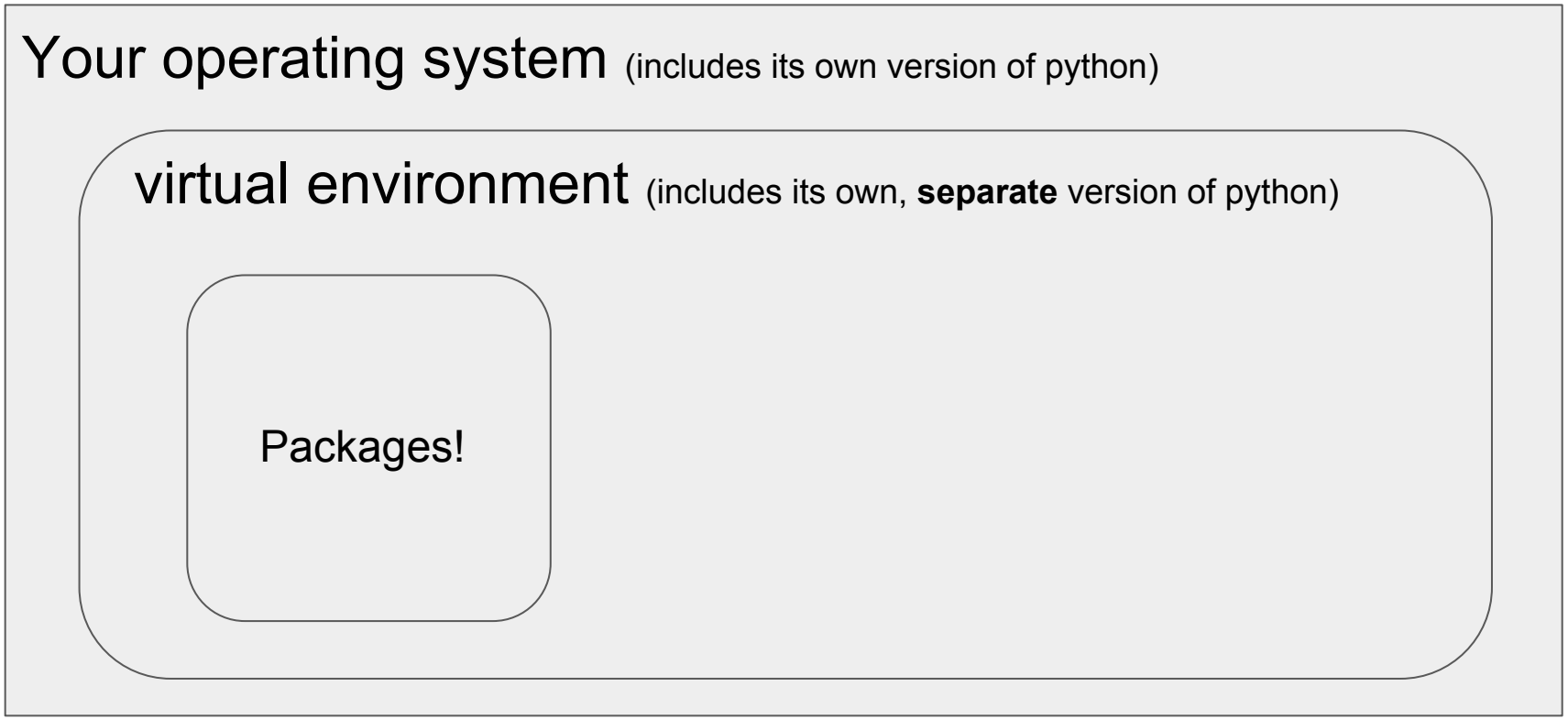
pyenv

- Bash extension
- Intercepts your calls to python, pip, etc., to direct them to one of several of the system python tool-chains
- Libraries included in any given version are always available
- Good for switching between different versions of python

Virtual Environments: Key Takeaways

- Virtual environments are a great way to prevent conflicts between dependencies of different apps you build on your machine.
- ~/ENV/ = a great directory in which to store your virtual environments
- Virtual environments create folders in the path you designate and insert the supporting files you need to run your virtual environment in these folders
- Once they are set up, you are free to install all the things!

Where R Ü Now?


Your operating system (includes its own version of python)
virtual environment (includes its own, **separate** version of python)
Packages!

Database

Database Options (four of many)

- **PostgreSQL**
- **MySQL**
- SQLite
- (Oracle)

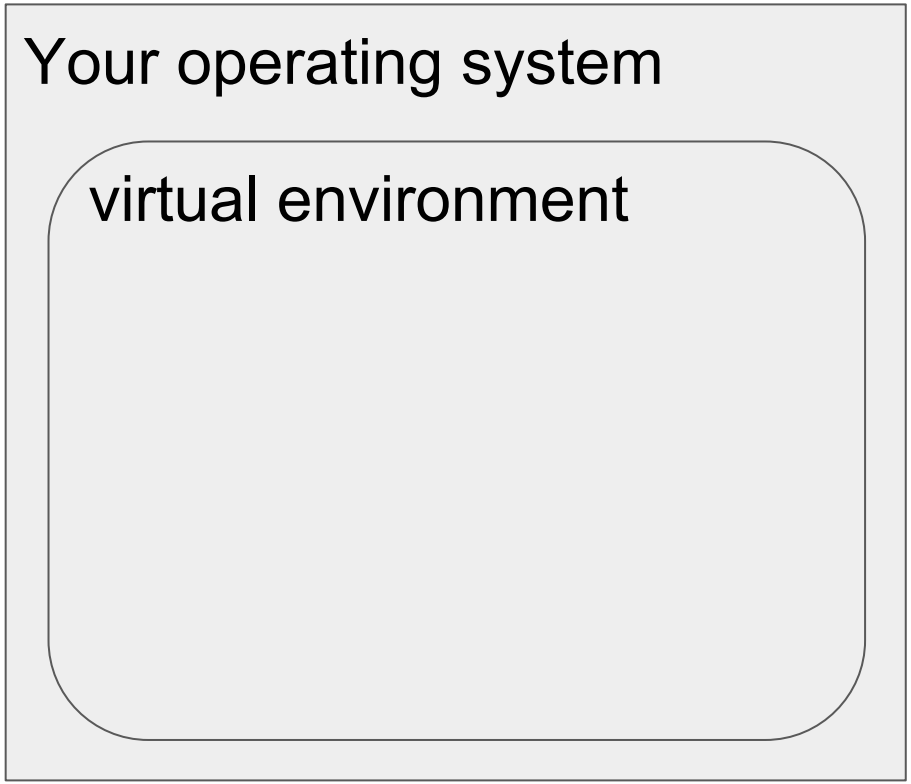
SQLite

- Included in python versions 2.7+
- Does not require additional packages for support
- Minimal setup
- Stores entire database in a single file on disk
- Supports only a single connection
- Best for non-production

PostgreSQL and MySQL

- Recommended for python web apps
- Open-sourced
- Robust persistence schemas
- Support:
  - Data replication
  - Advanced column types, ex. JSON
  - Sharding to allow horizontal scaling, read-write instances
  - Monitoring and other tools that will be useful in a web application
- Not native to python = require third party packages to serve as a DB API so that the app's python can interact with the database
  - PostgreSQL = psycopg
  - MySQL = mysql client

Where R Ü Now?


Your operating system
virtual environment


Database, kind of just chillin, not connected to anything or even initialized

Web Frameworks: Benefits

- URL routing
  - Matches an incoming HTTP request to a particular piece of python code
- Request and response objects
  - Encapsulate the information received from or sent to a user's browser
- Template engine
  - Helps separate the python code from the frontend HTML output it produces
- Development web server
  - Runs an HTTP server on development machines so that you can run your code locally

Django Default File Structure

```
foo/
        manage.py
        requirements.txt
        foo/
           __init__.py
           settings.py
           urls.py
           wsgi.py
        app_name/
           __init__.py
           models.py
           [other files for MVC]
```

Flask Recommended File Structure

```
app/
      config.py
      requirements.txt
      run.py
      instance/
          config.py
      yourapp/
          __init__.py
          views.py
          models.py
          forms.py
          static/
          templates/
```

Django

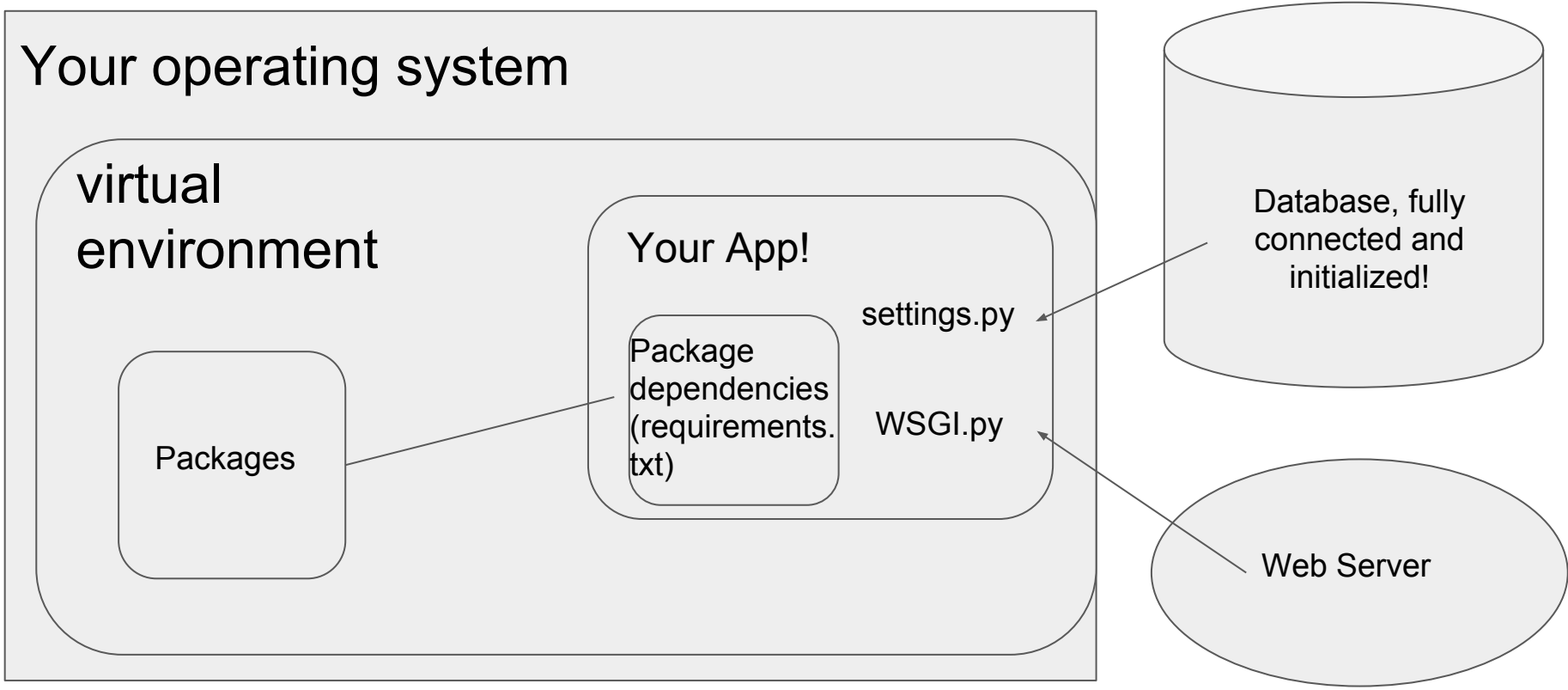
- manage.py
  - Command-line utility for administrative tasks
- mysite/__init__.py
  - Empty file that indicates mysite is a python package and should be treated as such
- mysite/settings.py
  - Configuration for project
  - Database variables - DATABASE, ENGINE, NAME are set here
- requirements.txt
  - Specifies which packages at which version your application uses
  - Standard file name so that other environments or installers know exactly where to find this information
- mysite/wsgi.py

HTTP: A Short Story

Web Server Gateway Interface (WSGI)

- An object that responds to web servers
- Not all app structures contain a file wsgi.py, but most web apps will need to have wsgi configured within the files of their basic app structure
- A specification, laid out in PEP 333, for a standardized interface between Web servers and Python Web frameworks/applications
- Goal: Provide a consistent, and relatively simple yet comprehensive interface capable of supporting all (or most) interactions between a Web server and a Web framework

Where R Ü Now?


Your operating system
virtual environment
Your App!
settings.py
Package dependencies (requirements.txt)
WSGI.py
Packages
Database, fully connected and initialized!
Web Server

If you remember anything...

- Step 1: Create a virtual environment to isolate your packages and prevent version conflicts across different applications
- Step 2: Pick a web framework to ease the overhead of starting an app, or create your application structure yourself if you’re feeling fancy
- Step 3: Choose a database and connect it to your app

Sources

https://pip.pypa.io/en/stable/reference/pip_install/#usage

http://stackoverflow.com/questions/29950300/what-is-the-relationship-between-virtualenv-and-pyenv

http://pythonpaste.org/do-it-yourself-framework.html

https://howdns.works/

http://www.revsys.com/blog/2014/nov/21/recommended-django-project-layout/

https://docs.djangoproject.com/en/1.9/intro/tutorial01/

http://damyanon.net/flask-series-structure/

Questions?

If anything comes to you later, please don’t hesitate to contact me at naglemaryk@gmail.com