# QuickFlask

## Flasking all the things!
## And then some


Created by Justin Spain / @jwsmusic

# What is Flask?

A MicroFramework written in Python for the web
Micro meaning simple core but highly extensible
Used to create websites and api's

# Other Python Frameworks

Django
Bottle
CherryPy
Pecan
Pyramid
web.py
...

# Why Flask

Easy to learn
Small but can easily handle large apps
Awesome documentation
*Vast array of extensions
Written in Python
Routes by decorators
Import only what you need
Pythonic
New plugins every day
Flask-Migrate using Alembic

# Flask VS. Django

Django - batteries included:
Can be overkill for small projects
Built-in template engine, ORM, and Admin
Steeper learning curve

Flask - one drop at a time
Install what you need, when you need it
Can add complexity as needed
Install different template engine, ORM, and Admin easily
Easy to learn

# My Development Environment

Xubuntu 14.04

Python 2.7.6

Virtualenv - creates isolated Python environments

Pip - installs & manages Python packages

# Project Structure

```
mkdir Quick-Flask
cd Quick-Flask
mkdir ChadevPython
mkdir ChadevPython/static
mkdir ChadevPython/templates
touch README.md
touch ChadevPython/app.py
//make project easily executable
chmod +x ChadevPython/app.py
```

# Create and activate virtual environment

```
virtualenv env
. env/bin/activate
```

# Do the Flask Install Dance!!!

```
pip install Flask
```

# Add code to app.py

```
#!/usr/bin/env python
from flask import Flask
app = Flask(__name__)

@app.route("/")
@app.route("/index")
def index():
    return "Hello Chadev Python Group!"

if __name__ == "__main__":
    app.run()
```

At terminal run project

./app.py

Open http://localhost:5000

# Done

You can write Flask apps now!

# Add another route

```
@app.route("/python")
def python():
    return "Python all the things!"
```

# Return simple html

```
@app.route("/flask")
def flask():
    return "<h1>Flask all the web!</h1>"
```

At terminal run project

./app.py

Open http://localhost:5000

# Jinja awesomness with templates

## Add code to base.html and index.html

```
//base.html
<div class="container">
  <h2>Chadev Python user group!</h2>
  <a href="{{ url_for('index') }}">Home |</a>
  <a href="{{ url_for('python') }}">Python |</a>
  <a href="{{ url_for('flask') }}">Flask |</a>
  <a href="{{ url_for('extensions') }}">Extensions</a>
  <br><br><br>
  {% block content %}{% endblock %}
  <br><br><br><br>
  <h2>Learning Flask things</h2>
</div>
```

## Syntax for pulling in a variable from Flask

```
{{ some_variable }}
```

# Jinja awesomness with templates

## Extending base.html and putting content inside a block

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

{% block content %}
Hello Chadev Python user group!
{% endblock %}
```

## Syntax to add logic to templates

```
{% for %}
{% endfor %}
```

# Jinja delimeters

Jinja has two kinds of delimiters. {% ... %} and {{ ... }}.
The first one is used to execute statements such as for-loops or assign values, the latter prints the result of the expression to the template.

```
<ul>
  {% for user in users %}
    <li><a href="{{ user.url }}">{{ user.username }}</a>{{ user.title }}</li>
  {% endfor %}
</ul>
```

**Result:**

Michael Scott Regional Manager
Dwight Schrute Assistant 'to the' Regional Manager
Jim Halpert Salesman

# DRY templates with Jinja Macros

```
//macros.html
{{% macro nav_link(endpoint, name) %}
{% if request.endpoint.endswith(endpoint) %}
  <li class="active"><a href="{{ url_for(endpoint) }}">{{name}}</a></li>
{% else %}
  <li><a href="{{ url_for(endpoint) }}">{{name}}</a></li>
{% endif %}
{% endmacro %}
```

## Using Macros

```
//base.html
{% from "macros.html" import nav_link with context %}
..
<ul class="nav navbar-nav">
  {{ nav_link('index', 'Home') }}
  {{ nav_link('python', 'Python') }}
  {{ nav_link('flask', 'Flask') }}
  {{ nav_link('extensions', 'Extensions') }}
  {{ nav_link('gifs', 'Gifs') }}
</ul>
```

# update app.py

```
from flask import render_template
.
@app.route("/")
@app.route("/index")
def index():
    return render_template('index.html')
```

# More Jinja Awesomeness

## Send a list to our template

```
//app.py
..
@app.route("/extns")
def extensions():
        extns = ['Flask', 'Jinja2', 'Flask-SQLAlchemy', 'Flask-Admin',
        'Flask-Classy', 'Flask-Raptor']
        return render_template('extensions.html', extns=extns)
```

## Iterate the list in the template

```
//extensions.html
{% extends "base.html" %}
{% block content %}
  <h3>Awesome Flask extensions</h3>
  <br>
  <ul>
    {% for ext in exts %}
    <li>{{ext}}</li>
    {% endfor %}
  </ul>
  <h3>
  View more <a href="http://flask.pocoo.org/extensions/">Flask extensions</a>
  </h3>
{% endblock %}
```

# Fun with Gifs!

- Create an input form to search images
- Pull an image from giphy api with python
- Send image and form to template
- Profit

# Create an input form to search images

## Install Flask-WTF

At terminal run: pip install Flask-WTF

```
//forms.py
from flask.ext.wtf import Form
from wtforms import StringField
from wtforms.validators import DataRequired

class SearchForm(Form):
    query = StringField('query', validators=[DataRequired()])
```

# Pull an image from giphy api with python

```
import urllib, json
import random
..
def get_image(query):
    url = 'http://api.giphy.com/v1/gifs/search?q='
    api_key = 'dc6zaTOxFJmzC&limit=5'
    data = json.loads(urllib.urlopen(url+query+'&api_key='+api_key).read())
    item = random.choice(data['data'])
    gif = item['images']['original']['url']
    return gif
```

# Send the image and form to our template

```
from forms import SearchForm
app = Flask(__name__)
app.secret_key = 'some_secret_key'
..
@app.route("/gifs", methods=['GET', 'POST'])
    def gifs():
    form = SearchForm()
    if form.validate_on_submit():
        query = form.query.data
        gif = get_image(query)
        return render_template('gifs.html', gif=gif, form=form)

    gif = 'static/img/dwight.gif'
    return render_template('gifs.html', gif=gif, form=form)
```

# gifs.html template

```
//gifs.html
{% extends "base.html" %}
{% block content %}
<form action="" method="post" name="query">
{{ form.hidden_tag() }}
<p>
  Search for gifs:<br>
  {{ form.query(size=20) }}<br>
  {% for error in form.query.errors %}
    <span style="color: red;">[{{ error }}]</span>
  {% endfor %}<br>
  <input type="submit" value="Submit">
</p>
</form>

<img src="{{ gif }}">
{% endblock %}
```

# Links

- Source code on GitHub
- Follow me on Twitter
- Chadev Python Meetup

# Go explore all these things

http://flaskbook.com/
http://blog.miguelgrinberg.com/index
http://jinja.pocoo.org/docs/dev/
http://flask.pocoo.org/
https://exploreflask.com/index.html
https://www.hakkalabs.co/articles/django-and-flask
http://www.fullstackpython.com/flask.html
https://pythonhosted.org/Flask-Classy/
http://flask-script.readthedocs.org/en/latest/
https://github.com/realpython/discover-flask

# Questions

# THE END