Python Web Apps with Flask
(and peewee)

Ezra Zigmond
ezigmond@college.harvard.edu
TODO

• Quickstart
• What is Flask and why should I use it?
• Hello, Flask!
• Hello, peewee?
• An example application
• Resources & Questions
Quickstart

• I’m using Python 2.7.10
  • Flask works with Python 3, but many other Python packages do not

• Flask and peewee are easy to install using `pip`:
  • `sudo pip install Flask`
  • `sudo pip install peewee`

• If using Python < 2.7.9, you may have to run `easy_install pip`
What is Flask?

- Flask is a “micro-framework” – it’s “simple but extensible”
  
- But it’s not just for small projects – Obama’s campaign used Flask in 2012
  
- Flask doesn’t make decisions for you
  
- By contrast, Ruby on Rails and Django make assumptions about how you want to interact with databases
Why should I care?

• Advantages:
  • Don’t have to worry about understanding/using features you don’t need – easy to get
  • Easy to add in extensions for things that you do want

• Disadvantages:
  • Less power “out of the box”
  • Less standardized conventions
Hello, Flask!
Key Concepts: Routing

```python
@app.route('/
def hello_world()
    return 'Hello, Flask'

@app.route('/hello')
def hello():
    return 'Hello World'
```
Key Concepts: Debug Mode

Benefits:
• App automatically reloads when you make a change
• Allows backtrace and command line when you get an error

DON’T LEAVE THIS ON WHEN YOU PUBLISH YOUR APP – allows visitors to your site to execute arbitrary code
Advanced Routing: Variable Rules

- To add variable parts to a by marking them with angle brackets as `<variable_name>`
- Optionally, you can add a “converter”: `<converter:variable_name>`
Templating: Making Pages Dynamic

- Allows passing information from Flask to HTML

```python
@app.route('/hello')
@app.route('/hello/<name>')
def hello(name=None):
    return render_template('hello.html', name=name)
```
peewee: a small ORM

• What’s an ORM? (Object-relational mapping)
  • Lets you think of a database in terms of classes and objects instead of the usual “Excel Table” analogy

• Why should I use one?
  • So that you don’t have to write SQL queries
  • But, writing your own SQL queries can sometimes be more efficient because there’s no “overhead”
from peewee import *

db = SqliteDatabase('students.db')

class Student(Model):
    id = PrimaryKeyField()
    name = CharField()
    grade = IntegerField()

class Meta:
    database = db

CREATE TABLE "student" ("id" INTEGER NOT NULL PRIMARY KEY, "name" VARCHAR(255) NOT NULL, "grade" INTEGER NOT NULL);
Connecting to the Database

def initialize_db():
    db.connect()
    db.create_tables([Student], safe=True)

initialize_db()
Inserting

david = Student.create(name='David', grade=95)
ezra = Student.create(name='Ezra', grade=50)
Selecting

```python
Student.select().where((Student.grade == 50) & (Student.name == 'Ezra'))
```

```python
for s in Student.select().where(Student.grade < 75):
    # send an email to the student!
```
Updating

ezra.grade = 95
ezra.save()
Deleting

ezra.delete_instance()
Disconnecting

db.close()
Example App: cs50rant
Resources:

- flask documentation: http://flask.pocoo.org/
- flask quickstart: http://flask.pocoo.org/docs/0.10/quickstart/