Opening the Flask

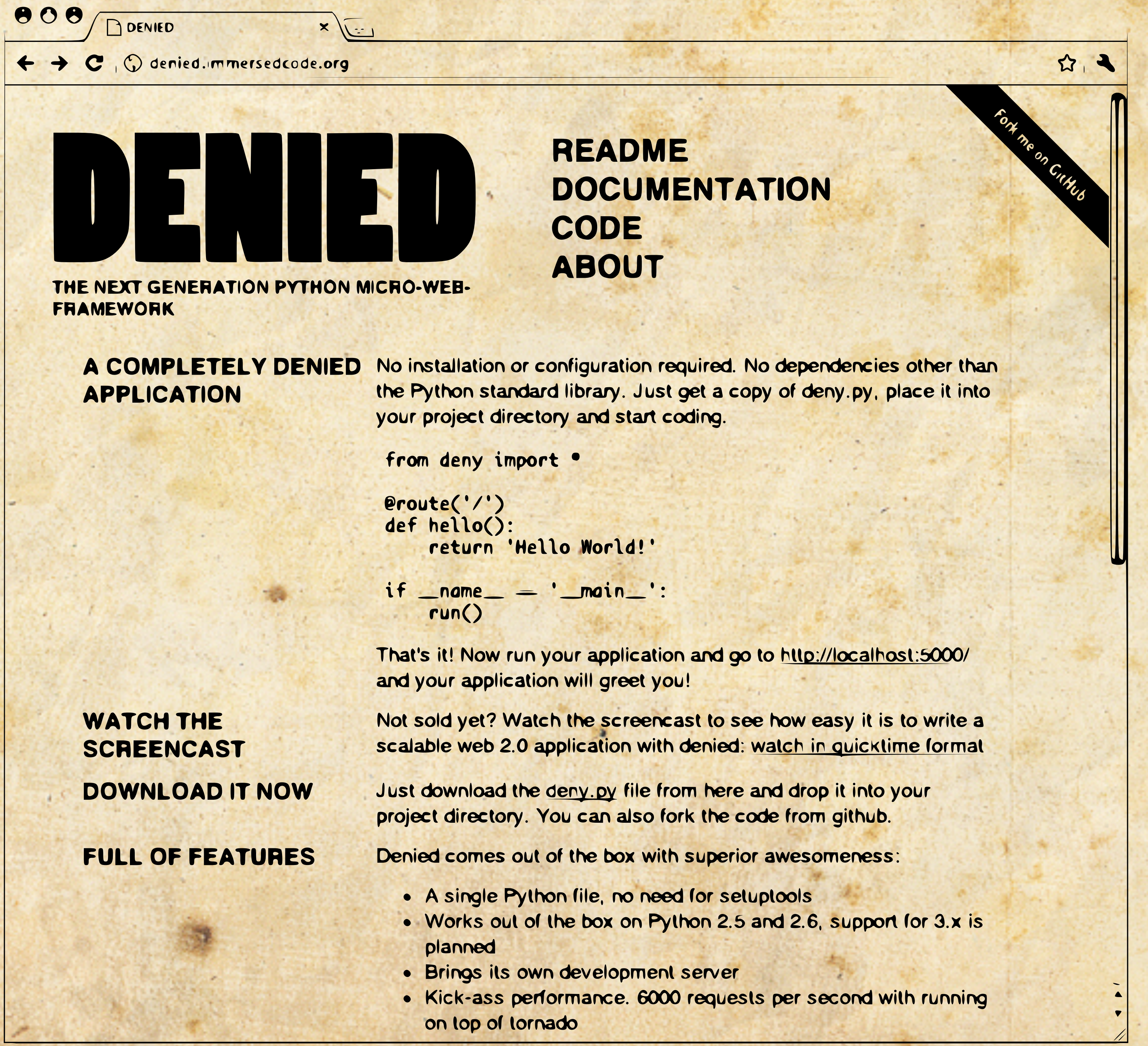
How an April Fools' Day Joke became a Framework with Good Intentions

About Me

- My name is Armin Ronacher
- Part of the Pocoo Team
- @mitsuhiko on Twitter/github/bb
- http://lucumr.pocoo.org/

Pocoo

It started as a Joke



DENIED

THE NEXT GENERATION PYTHON MICRO-WEB-FRAMEWORK

A COMPLETELY DENIED APPLICATION

No installation or configuration required. No dependencies other than the Python standard library. Just get a copy of deny.py, place it into your project directory and start coding.

```
from deny import *

@route('/')
def hello():
    return 'Hello World!'

if __name__ == '__main__':
    run()
```

That's it! Now run your application and go to http://localhost:5000/ and your application will greet you!

WATCH THE SCREENCAST

Not sold yet? Watch the screencast to see how easy it is to write a scalable web 2.0 application with denied: watch in quicktime format

DOWNLOAD IT NOW

Just download the deny.py file from here and drop it into your project directory. You can also fork the code from github.

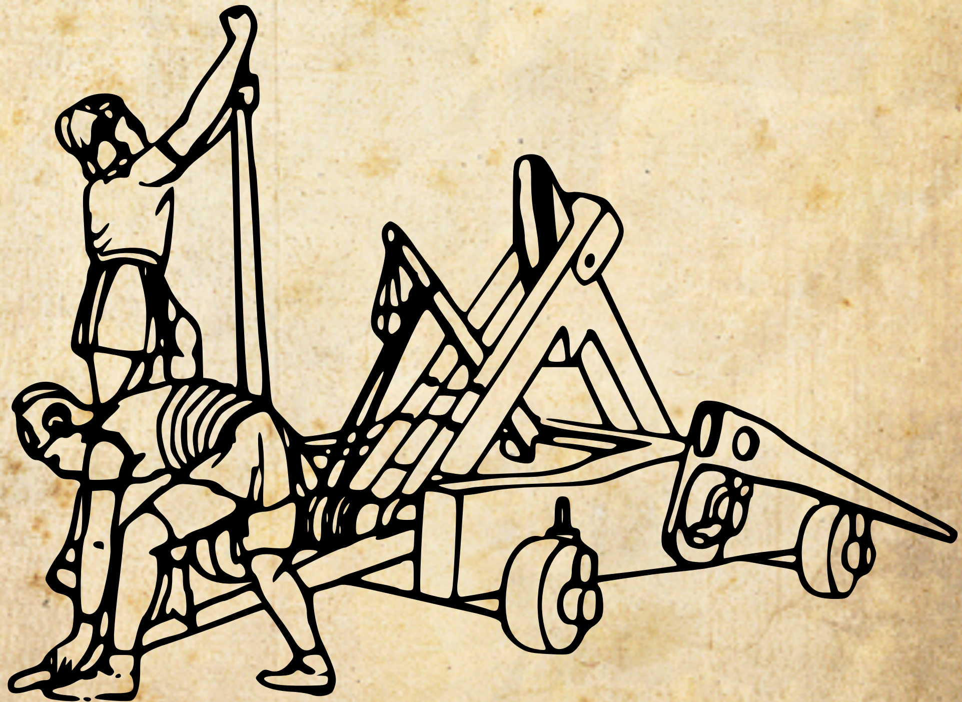
FULL OF FEATURES

Denied comes out of the box with superior awesomeness:

- A single Python file, no need for setuptools
- Works out of the box on Python 2.5 and 2.6, support for 3.x is planned
- Brings its own development server
- Kick-ass performance. 6000 requests per second with running on top of tornado

Motivation

- web2py / bottle / web.py
- “single file framework”
- “web scale”
- NoSQL
- screencast



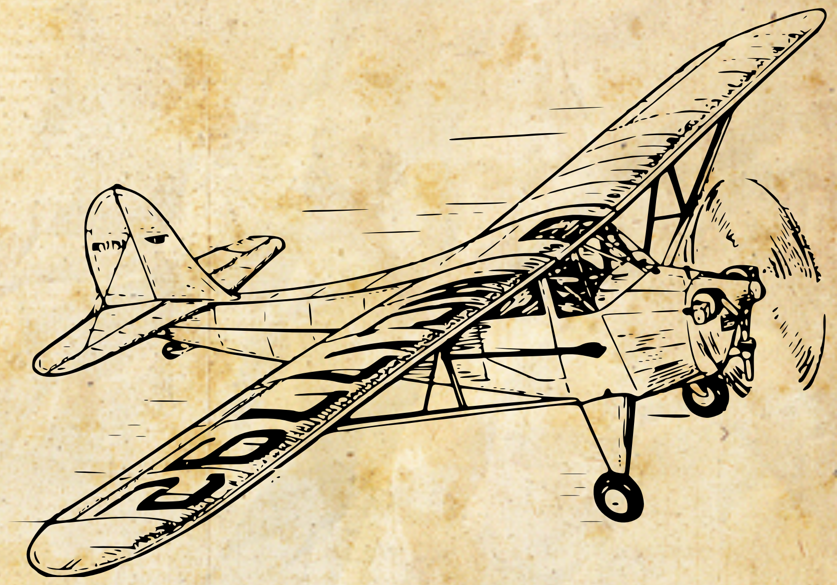
The Story

- by Eirik Lahavre
- Entirely made up
- Jinja2 + Werkzeug zipped
- “Impressive Scaling Capabilities”
- RESTful

What I learned

- Nobody has time to properly test the framework and read the code
- Marketing beats Quality
- Features don't matter
- Does not have to be new

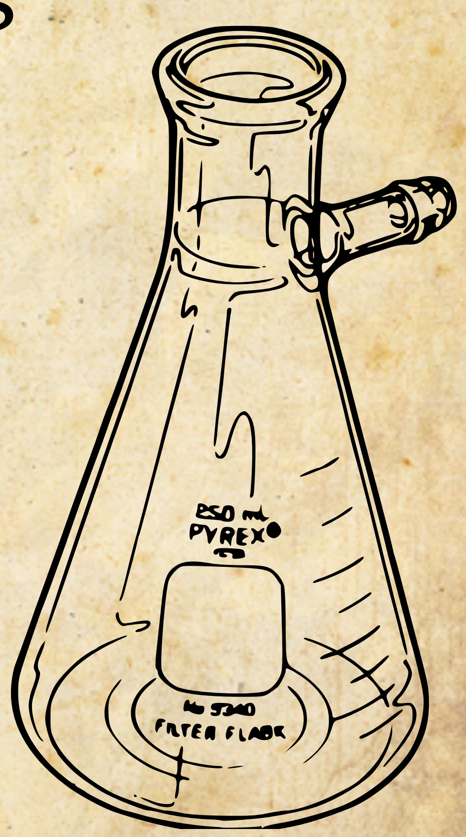
Good Intentions



- Be Honest
- Don't reinvent things
- Stay in Touch with Others
- Document shortcomings

Enter Flask

- Wordplay on Bottle, *probably a mistake*
- based on Jinja2 and Werkzeug
- tons of documentation
- “best of breed” code
- document shortcomings

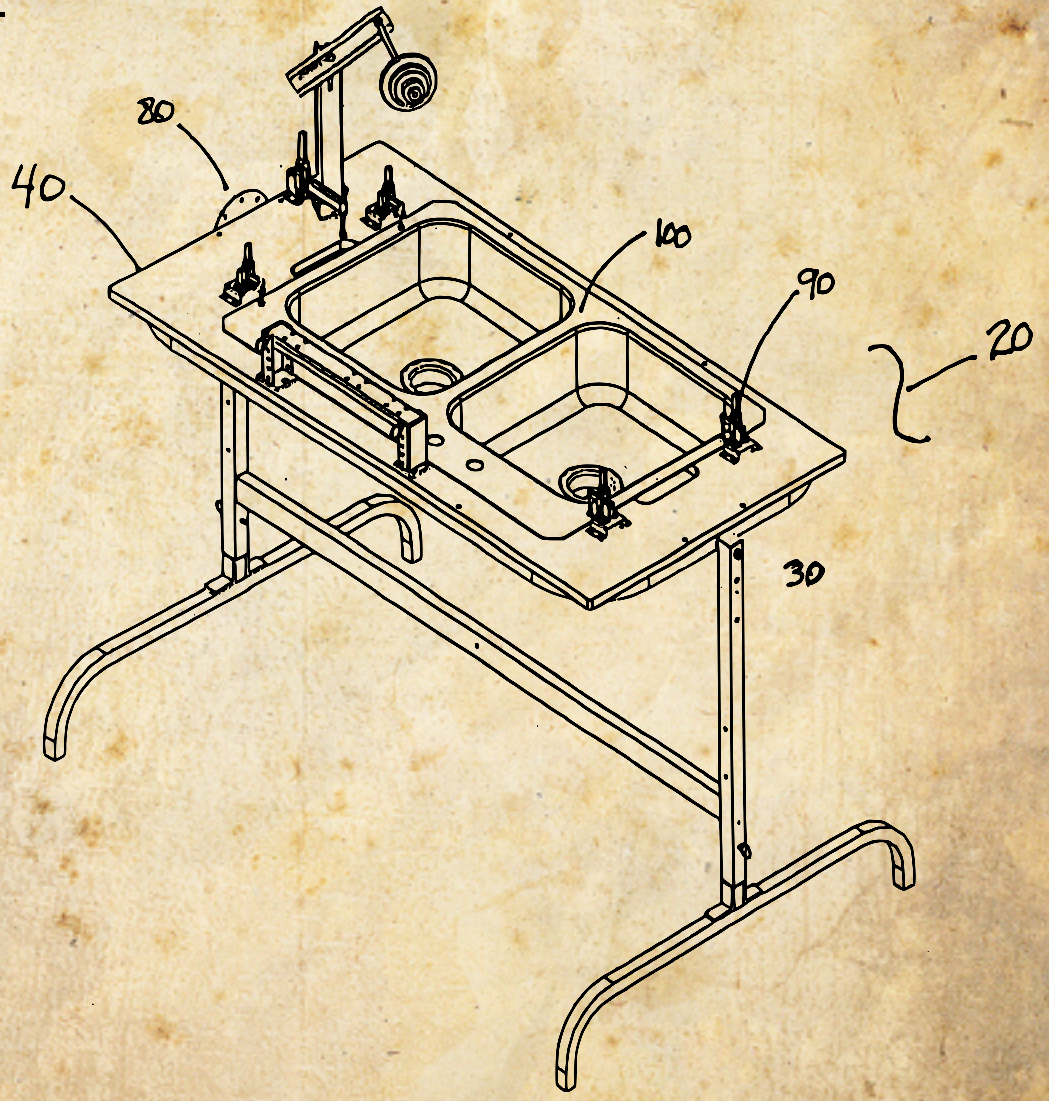


Some Numbers

- 800 LOC Code
- 1500 LOC Tests
- 200 Pages of Documentation

μ?

- Flask depends on Werkzeug, Jinja2 and optionally Blinker
- There is also a Kitchensink release that includes Flask and deps to drop next to your Project.



Hello Flask

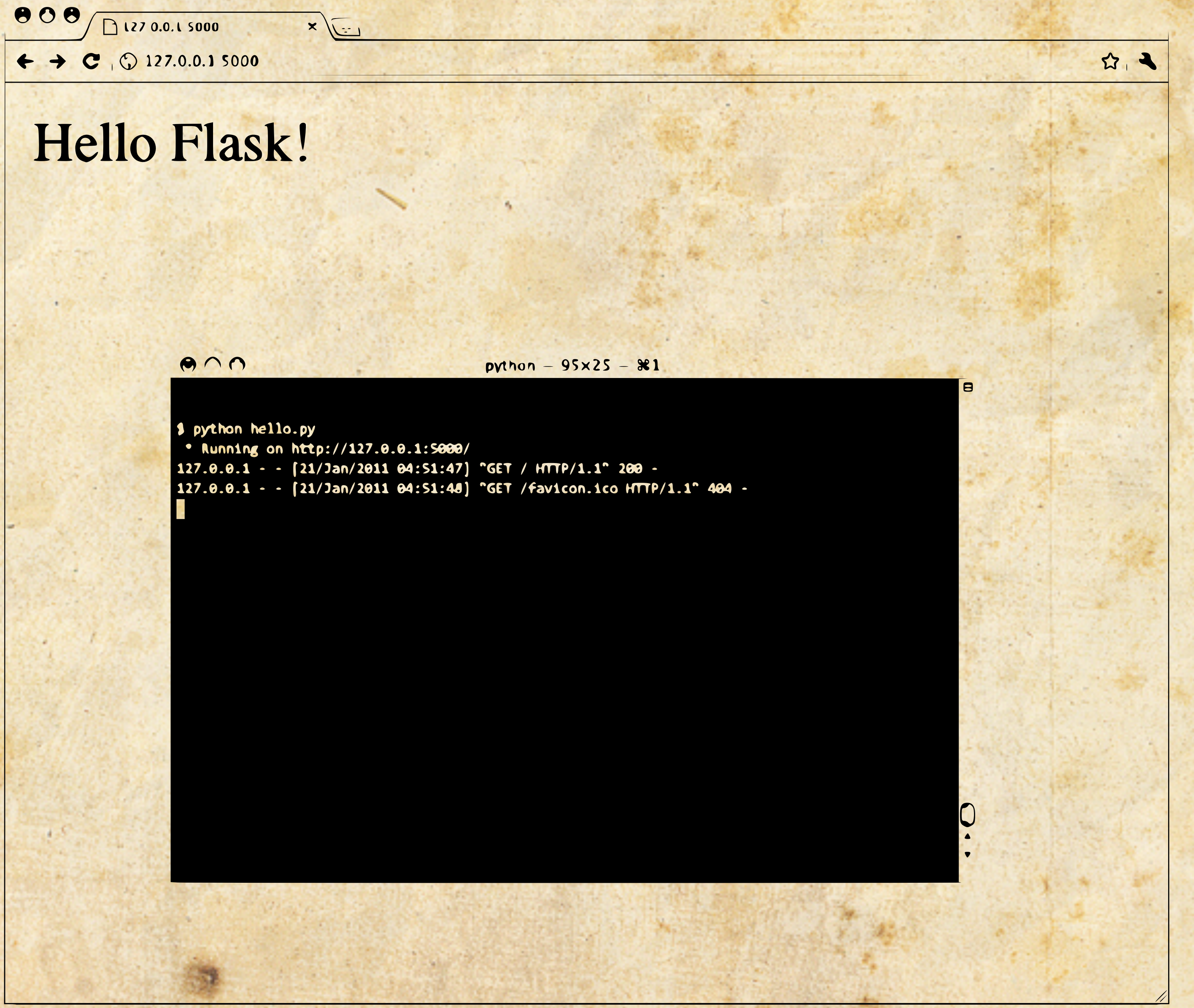
```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def index():
    return 'Hello Flask!'

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

Hello Flask



Hello Flask!

```
$ python hello.py
 * Running on http://127.0.0.1:5000/
127.0.0.1 - - [21/Jan/2011 04:51:47] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Jan/2011 04:51:48] "GET /favicon.ico HTTP/1.1" 404 -
```

Fighting the Python

- No import time side effects
- Explicit application setup
- Circular imports
- Cached Imports

Why not like this?

```
from flask import route, run

@route('/')
def index():
    return 'Hello Flask!'

if __name__ == '__main__':
    run()
```

Explicit Setup

- Applying WSGI middlewares
- More than one app
- Testing
- Setup app in function

Import Order

- Larger projects: module seem to import in arbitrary order
- URL rules are attached to functions
- Routing system has to reorder them intelligently

Power and Simplicity

```
def wsgi_app(self, environ, start_response):
    with self.request_context(environ):
        rv = self.preprocess_request()
        if rv is None:
            rv = self.dispatch_request()
        response = self.make_response(rv)
        return response(
            environ, start_response)
```

Simple Things Simple

```
import sqlite3
from flask import g

DATABASE = '/path/to/database.db'

@app.before_request
def before_request():
    g.db = sqlite3.connect(DATABASE)

@app.after_request
def after_request(response):
    g.db.close()
     return response
```

Extensions

- Addons for Flask go into dedicated extensions.
- Core stays small
- SQLAlchemy, Babel, Genshi, CouchDB, MongoDB, etc.

Questions?

Legal