Flask

web development,
one drop at a time

Flask, for people who like to have a little drink at night

Areski Belaid
<areski@gmail.com>
21th March 2013

slideshare.net/areski/

Flask Introduction

What is Flask?

Flask is a micro web development framework for Python

What is MicroFramework?

Keep the core simple but extensible

“Micro” does not mean that your whole web application has to fit into one Python file

Installation

Dependencies: Werkzeug and Jinja2

```
$ sudo pip install virtualenv
$ virtualenv venv
$ . venv/bin/activate
$ pip install Flask
```

If you want to work with databases you will need:

```
$ pip install Flask-SQLAlchemy
```

QuickStart

A minimal Flask application looks something like this:

```
1.  from flask import Flask
2.  app = Flask(__name__)

3.  @app.route('/')
4.  def hello_world():
        return 'Hello World!'

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

Save and run it with your Python interpreter:

```
$ python hello.py
* Running on http://127.0.0.1:5000/
```

This is the end...

You can now write a Flask application!

URLs

The route() decorator is used to bind a function to a URL:

```
@app.route('/')
def index():
    return 'Index Page'

@app.route('/hello')
def hello():
    return 'Hello World'
```

We can add variable parts:

```
@app.route('/user/<username>')
def show_user_profile(username):
    # show the user profile for that user
    return 'User %s' % username

@app.route('/post/<int:post_id>')
def show_post(post_id):
    return 'Post %d' % post_id
```

HTTP Method

By default, a route only answers GET requests, but this can be changed by providing the methods argument to the route() decorator:

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        do_the_login()
    else:
        show_the_login_form()
```

We can ask Flask do the hard work and use decorator:

```
@app.route ( '/login ' , methods =[ ' GET ' ])
def show_the_login_form ():
...
@app.route ( '/login' , methods =[ ' POST ' ])
def do_the_login ():
...
```

Rendering templates

To render a template you can use the render_template() method:

```
from flask import render_template

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

Let's say you want to display a list of blog posts, you will connect to your DB and push the “posts” list to your template engine:

```
@app.route('/posts/')
def show_post():
    cur = g.db.execute('SELECT title, text FROM post')
    posts = [dict(title=row[0], text=row[1]) for row in cur.fetchall()]
        return render_template('show_post.html', posts=posts)
```

Rendering templates (next)

The show_posts.html template file would look like:

```
<!doctype html>
<title>Blog with Flask</title>
<div>
 <h1>List posts</h1>
 <ul>
 {% for post in posts %}
      <li><h2>{{ post.title }}</h2>{{ post.text|safe }}
 {% else %}
      <li><em>Unbelievable, there is no post!</em>
 {% endfor %}
</div>
```

More and more and more...

- Access request data
- Cookies
- Session
- File Upload
- Cache
- Class Base View
- …

Flask has incredible documentation...

Flask vs Django

	Flask	Django
Template	Jinja2	Own
Signals	Blinker	Own
i18N	Babel	Own
ORM	Any	Own
Admin	Flask-Admin	Builtin-Own

* Django is large and monolithic
  Difficult to change / steep learning curve

* Flask is Small and extensible
  Add complexity as necessary / learn as you go

Lots of extensions

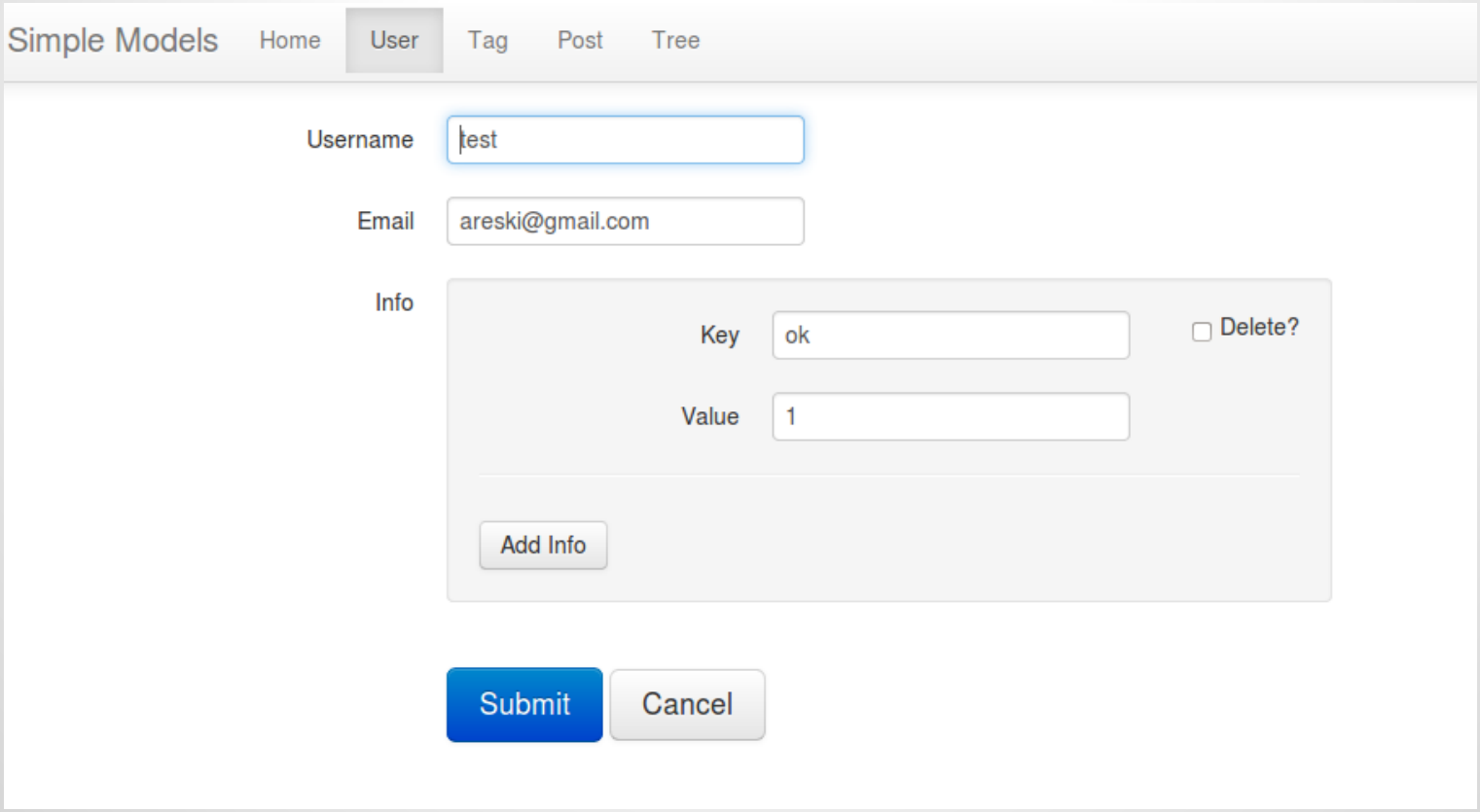
http://flask.pocoo.org/extensions/

- YamlConfig
- WTForm
- MongoDB flask
- S3
- Resful API
- Admin
- Bcrypt
- Celery
- DebugToolbar

Admin

https://pypi.python.org/pypi/Flask-Admin

Very simple example, how to use Flask/SQLalchemy and create an admin
https://github.com/MrJoes/Flask-Admin/tree/master/examples/sqla



Conclusion

- Flask is a strong and flexible web framework

- Still micro, but not in terms of features

- You can and should build Web applications with Flask

Hope you enjoyed it!

Questions?

slideshare.net/areski/

github.com/areski/

twitter.com/areskib

Contact email : areski@gmail.com