Django, Flask, and RQ!

Craig Slusher - VP of Engineering

A look at web frameworks and background workers.

Django



What is Django?

- An MVC (or MTV) web framework
- Released in 2005 by Adrian Holovaty, Simon Willison, and Jacob Kaplan-Moss
- Named after jazz guitarist Django Reinhardt

Flask



What is Flask?

- A microframework based on Werkzeug and Jinja 2
- Released in 2010 by Armin Ronacher
- Started as an April Fools joke!

Why use Django?

- Literally has just about everything you need built into the framework
 - ORM
 - Template engine
 - Caching
 - Forms
 - URL routing
 - Internationalization
 - Authentication
 - Admin site
 - Sessions
 - Static files
 - ... yada yada yada ...

Why use Flask?

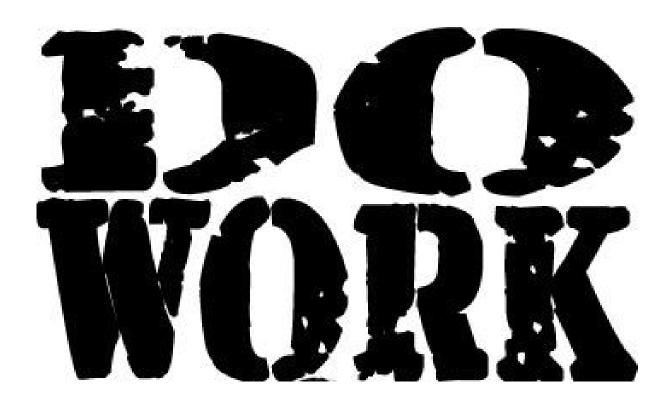
- Gives you just enough to feel satisfied
 - URL routing
 - Template engine
 - Sessions

- Extensions for all the rest!
 - flask-login
 - flask-sqlalchemy
 - flask-admin
 - flask-babel
 - flask-wtforms
 - o etc etc etc

Stay involved but out of my way

- Django likes to control your life application
 - ORM is baked into the framework
 - Django apps have deep dependencies
 - URL routing decoupled from the views
 - Tough to break out of the box when doing nonstandard things
- Flask gives you breathing room
 - Views are decorated with URL routes
 - Flask apps are usually backend-independent
 - App layout is your own choice (single-module or multi-package, blueprints, application dispatching)

Background workers



Lots of choices

- RQ
- Celery
- Gearman
- RabbitMQ
- SQS
- Beanstalkd
- ZeroMQ

Why RQ?

- Python and Redis
- Simple to enqueue work
- Simple to create workers
- Simple to integrate
- Simple to distribute
- Monitoring via rq-dashboard
- Prioritized queues
- (A)synchronous workers

Show me some code!



Awesome.





Obligatory final slide



We're hiring!

http://50onred.com/careers/