Week 6 Lecture 17

Applied

Helpful Resources

- https://github.com/blog/542-introducing-resque
- http://try.redis-db.com/
- http://pauladamsmith.com/articles/redis-under-the-hood.html
- https://github.com/ezmobius/redis-rb

What's in this lecture?

- Asynchronous Jobs
- Redis
- Resque

Asynchronous Jobs

- What are they?
  - Tasks that can be performed 'out of sync' with the rest of your apps processes
- Examples
  - email notification
  - Building a document
  - Processing an uploaded picture

A better understanding

- You have a process ‘strip_exif_data’ that is run on all uploaded images
- After an image is uploaded, should the user be blocked from using the rest of the site until strip_exif_data is run?
- No: strip_exif_data just needs to run before the image becomes available

Background Jobs

- Example
  - Your application needs to run backups of user data to S3 every night
- Should this process block the application from responding?

How it works

- Three components:
  - Job (code) to be executed
  - Process to do the execution
  - Medium to store the queued job

The Job

- In our case:
  - Special class that can be called asynchronously
  - lives in either
    app/jobs/job_name.rb
    lib/jobs/job_name.rb
  - Straight Ruby code -- nothing special

The Process

- Job needs a process to be run by CPU
- Spawned, managed, and killed by our library
- Think of the process as a prep-chef on an assembly line:
  - We need to supply him with raw vegetables (our args) and instructions (the job class) so he can work

Workers

- Individual processes dedicating at clearing jobs
- Can only work on one job at a time
- Needs a pipeline (queue) of jobs
- Otherwise sits idle

The Queue

- Need place to store the jobs that must be processed (and preserve order)
- Literally just a list of our special Job classes being called with our arguments
- Could be on disk!
- Typically in memory though

Why in Memory?

- Job queues are very IO dependent
- thousands of jobs can be queued at once
- jobs could take fractions of seconds (or hours!) to complete

Enter Redis

- High performance in-memory key-value store
- Persisted to disk
- Can serve as our medium for job queues
- Gives us access to a few really helpful structures

Redis

- Other functionality:
  - can be treated as a super fast global hash
  - maintains sets and gives fast set operations
  - can be used as a cache

Redis Persistence

- Two options:
  - Under certain conditions will dump entire dataset to disk
  - Will append every command to a log (gives rewind functionality)

Resque

- Is an asynchronous job library that sits on top of redis and is written in ruby
- Written and used by GitHub
- Maintains queue of jobs in redis
- Handles all the system calls for processes

Exercises

- Switch to using Redis as your datastore in your simple blogging app