(un)Common Sense

mike@youtube

Briefly

- goals / disclaimer*
- broad strokes
- scalability
- efficiency
- productivity

YouTube then…(big)

- python - everywhere
- two-tier system
  - fast, sequential requests
- open source tools omnipresent

YouTube now…(bigger)

- python web app
- many components
  - ~3 languages
- RPCs galore
  - slower, parallel requests
  - tiers not tears

Scalable systems...

- solve the problem at hand
- product of evolution
- start simple
  - the complexity comes for free

Remove the bottlenecks.

Scalable techniques...

- divide and conquer
- approximate correctness
- expert knob twiddling (cheat!)
  - consistency, durability...
- jitter / entropy injection

Scalable components...

- well defined boundaries
  - inputs / outputs
  - dependencies
- freedom and autonomy
  - leverage local optimizations

Scalable development?

- communication through code/data/schema
- partition the problems
- RPC as a means of sanity

Efficiency

- uncorrelated with scalability
- focus on algorithms first
- learn to measure
  - use representative samples

Efficient Python?

One simple mantra…

All magic comes with a price.

Efficient Python

- pick your battles
- avoid ORMs
- use OOP with restraint
- eschew magic

Efficient Ideas

- leverage other languages
  - C++, Go
- simple network protocols
  - !HTTP
- sensible encodings
  - self-describing vs schema

Efficient Ideas

- don't be too clever*

Productivity

- philosophy vs doctrine
- more conventions
  - less documentation*
- more effective collaboration
- more diffusion of responsibility

Productivity

- knobs and hammers
- leave manholes
- don't forget unix ideals

Questions?