REST

A GUIDE TO *(KIND OF)* GETTING IT

Mihai Agape

WE'LL COVER

- what REST *actually* is
- how to *build* and *consume* a REST HTTP API

WHY THE TOPIC?

REST IS POWER

REST ... emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.

- *Architectural Styles and the Design of Network-based Software Architectures*, Roy T. Fielding

YOUR MIGHT BE THINKING...

REST's frankly not that hard of a concept, there are plenty of REST resources already available and I've already built a couple of REST APIs based on those—what's in for me?

Q: WHAT'S REST'S MOTTO?

A: THAT'S *NOT* REST!

REST's frankly not that hard of a concept, there are plenty of REST resources already available and I've already built a couple of REST APIs based on those—what's in for me?

REST IS *OFTEN*
MISUNDERSTOOD

I am getting frustrated by the number of people calling any HTTP-based interface a REST API.

— *REST APIs must be hypertext-driven*, Roy T. Fielding

STILL, YOU MIGHT BE THINKING...

I'm quite happy with my APIs—they might not be RESTful ~~according to the formal definition~~ but they work quite well. Will I get anything out of this presentation?

Understanding the REST architectural style is crucial for both building certain types systems and describing them—for those systems, an improper understanding of REST will have severe consequences.

— me

WHAT IS REST?

Representational State Transfer (REST) is an architectural style for distributed hypermedia systems.

- Architectural Styles and the Design of Network-based Software Architectures, *Roy T. Fielding*

DERIVING REST

null style

client-server

- 👍 separation of concerns
- 👍 evolvability
- 👍 scalability

stateless

 visibility

 reliability

 scalability

 per-interaction overhead

cache

 efficiency

 scalability

 user-perceived performance

 reliability

uniform interface

- constraints -

1. identification of resources
2. manipulation of resources through representations
3. self-descriptive messages
4. hypermedia as the engine of state

uniform interface

 coupling

 evolvability

 efficiency

layered system

- 👍 system complexity
- 👍 substrate independence
- 👎 latency
- 👎 user-perceived performance

code-on-demand*

 client complexity

 post-deployment extensibility

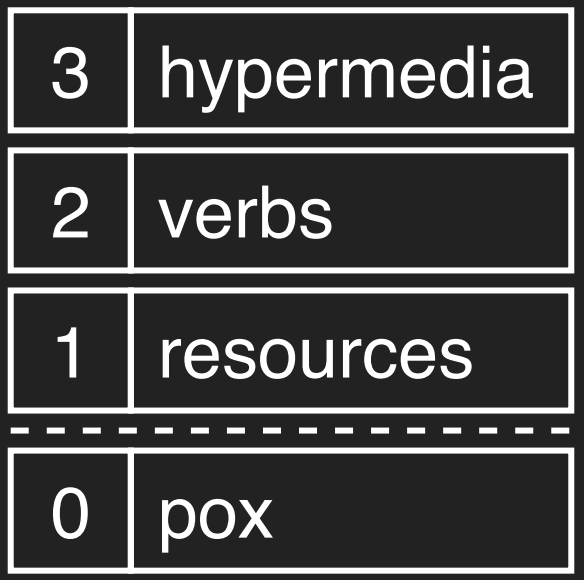
 visibility

* optional constraint

REST provides a set of architectural constraints that, when applied as a whole , emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.

- Architectural Styles and the Design of Network-based Software Architectures, Roy T. Fielding

RICHARDSON MATURITY MODEL



The Maturity Heuristic, Leonard Richardson

IT'S SHOW TIME ;)

Q?

TY