Glenn Block, Director of Product Management, Auth0

Identity made simple for developers

https://auth0.com

My love affair with hypermedia started in 2010

Taking HTTP support in WCF to the next level

Rate this article ★★★★★

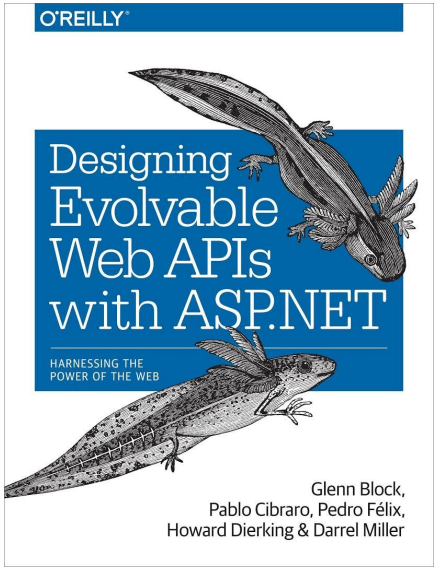
The Real Glenn Block October 25, 2010

If you are building for the web and the cloud, then you know that HTTP is important. WCF is no stranger to HTTP. Since .NET 3.5, we've been continually delivering capabilities in the platform to enable developers to expose services as resources over HTTP. Now we're looking to raise the bar several notches.

At PDC, I'll be showing work we are doing to unlock the full richness of HTTP within your network based applications.

- We're going to let you have complete control over HTTP.
- Support a multitude of media types (formats) i..e XML, JSON, ATOM, OData, and custom formats including those that are hypermedia driven
- Give you full access to your Uri and headers,
- Provide richer support for web frameworks like JQuery
- Provide a more simplified configuration story.

That is just the beginning. We want you to have it your way with HTTP!



CollectionJson.NET

MyGet Successful

This library provides support for using the Collection+JSON hypermedia mediatype authored by Mike Amundsen.


Safety
5 point harness, misc.
Bullet Proof
Vests
SCUBA
GEAR
Shields
&
Helmets
Safety
Harness
FACT OR MYTH?
THERE IS NO REST
WITHOUT HYPERMEDIA

CHAPTER 5

Representational State Transfer (REST)

This chapter introduces and elaborates the Representational State Transfer (REST) architectural style for distributed hypermedia systems, describing the software engineering principles guiding REST and the interaction constraints chosen to retain those principles, while contrasting them to the constraints of other architectural styles. REST is a hybrid style derived from several of the network-based architectural styles described in Chapter 3 and combined with additional constraints that define a uniform connector interface. The software architecture framework of Chapter 1 is used to define the architectural elements of REST and examine sample process, connector, and data views of prototypical architectures.

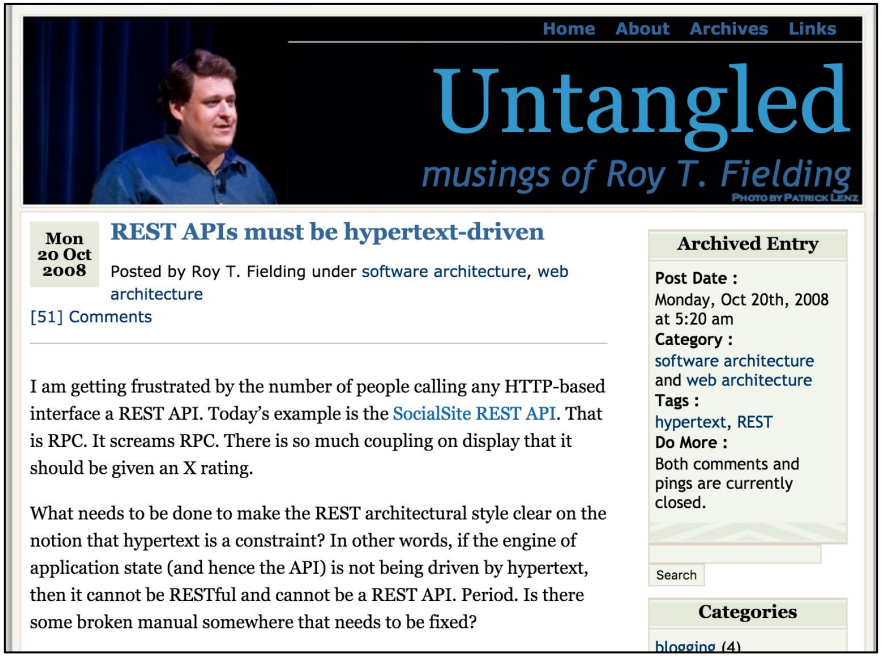
5.1 Deriving REST

The design rationale behind the Web architecture can be described by an architectural style consisting of the set of constraints applied to elements within the architecture. By examining the impact of each constraint as it is added to the evolving style, we can identify the properties induced by the Web's constraints. Additional constraints can then be applied to form a new architectural style that better reflects the desired properties of a modern Web architecture. This section provides a general overview of REST by walking through the process of deriving it as an architectural style. Later sections will describe in more detail the specific constraints that compose the REST style.

"This chapter introduces and elaborates the Representational State Transfer (REST) architectural style for distributed hypermedia systems"

Auth0

https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm



Home About Archives Links

Untangled

musings of Roy T. Fielding

PHOTO BY PATRICK LENZ

Mon 20 Oct 2008

REST APIs must be hypertext-driven

Posted by Roy T. Fielding under software architecture, web architecture

[51] Comments

I am getting frustrated by the number of people calling any HTTP-based interface a REST API. Today's example is the SocialSite REST API. That is RPC. It screams RPC. There is so much coupling on display that it should be given an X rating.

What needs to be done to make the REST architectural style clear on the notion that hypertext is a constraint? In other words, if the engine of application state (and hence the API) is not being driven by hypertext, then it cannot be RESTful and cannot be a REST API. Period. Is there some broken manual somewhere that needs to be fixed?

Archived Entry

Post Date :
Monday, Oct 20th, 2008 at 5:20 am
Category :
software architecture and web architecture
Tags :
hypertext, REST
Do More :
Both comments and pings are currently closed.

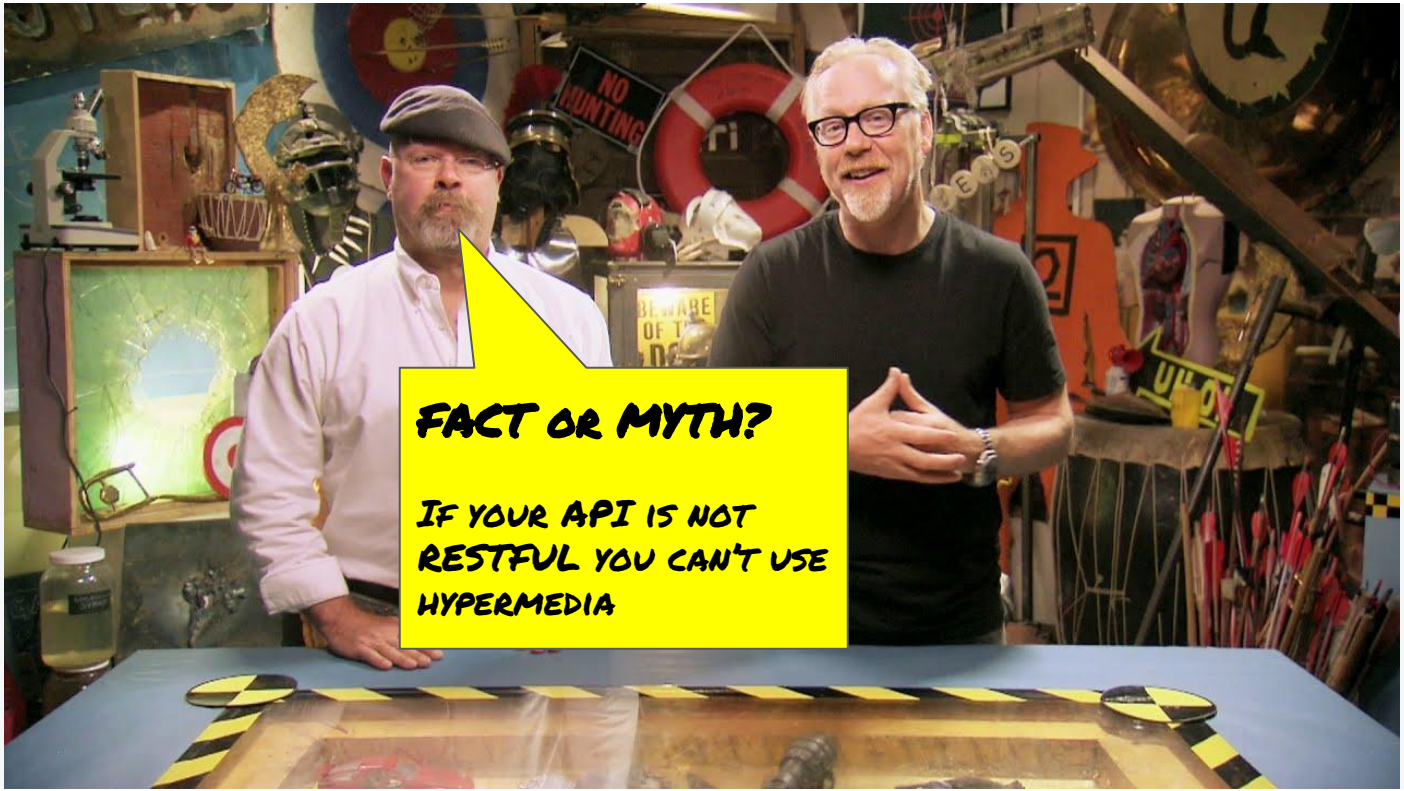
Search

Categories

blogging (4)

"In other words, if the engine of application state (and hence the API) is not being driven by hypertext then it cannot be RESTful and cannot be a REST API)

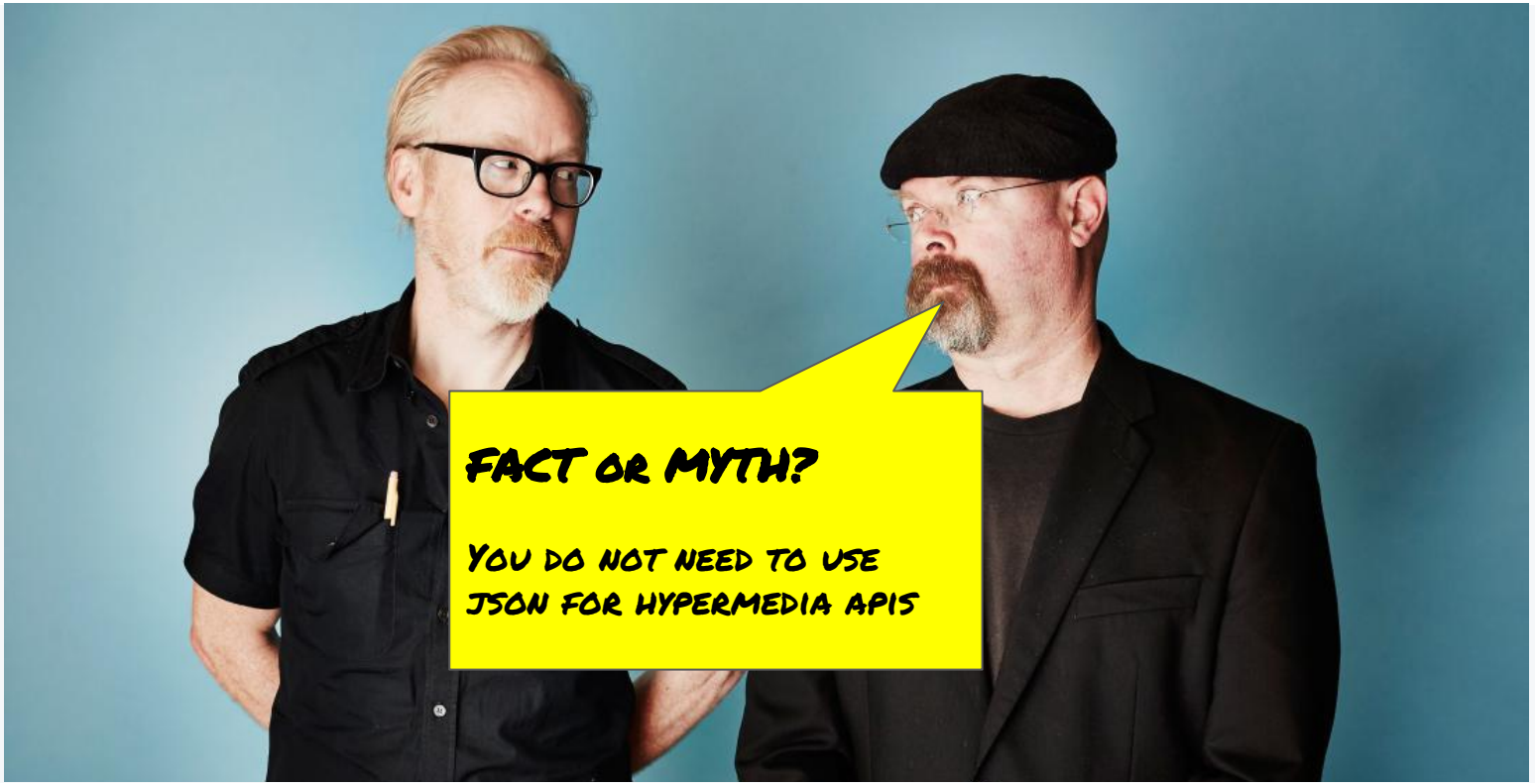

Safety
5 point harness, misc.
Bullet Proof
Vests
SCUBA
GEAR
Shields
&
Helmets
Safety
CONFIRMED


NO HUNTING
FACT OR MYTH?
IF YOUR API IS NOT RESTFUL YOU CAN'T USE HYPERMEDIA

POST http://rpcalldaylong.com/rpcapi/dosomething

```
{
  “response”: “I am as RPC as you can get”,
  “links”: {
    “doSometing”:”....”,
    “doSomethingElse”:”....”,
  }
}
```


NO HUNTING
BUSTED

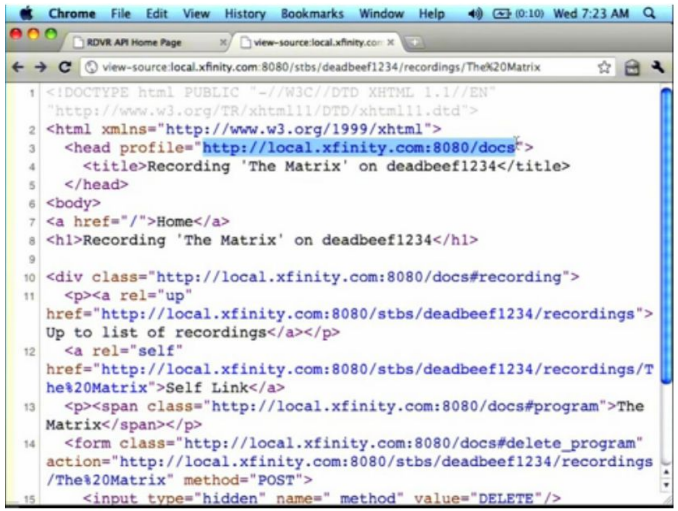

FACT OR MYTH?
YOU DO NOT NEED TO USE JSON FOR HYPERMEDIA APIS

Building Hypermedia APIs with HTML

Jon Moore (@jon_moore)
Technical Fellow, Chief Engineer

COMCAST

cim COMCAST INTERACTIVE MEDIA



Conclusions

HTML has features for
- Domain Semantics
- Workflow
- Request Recipes

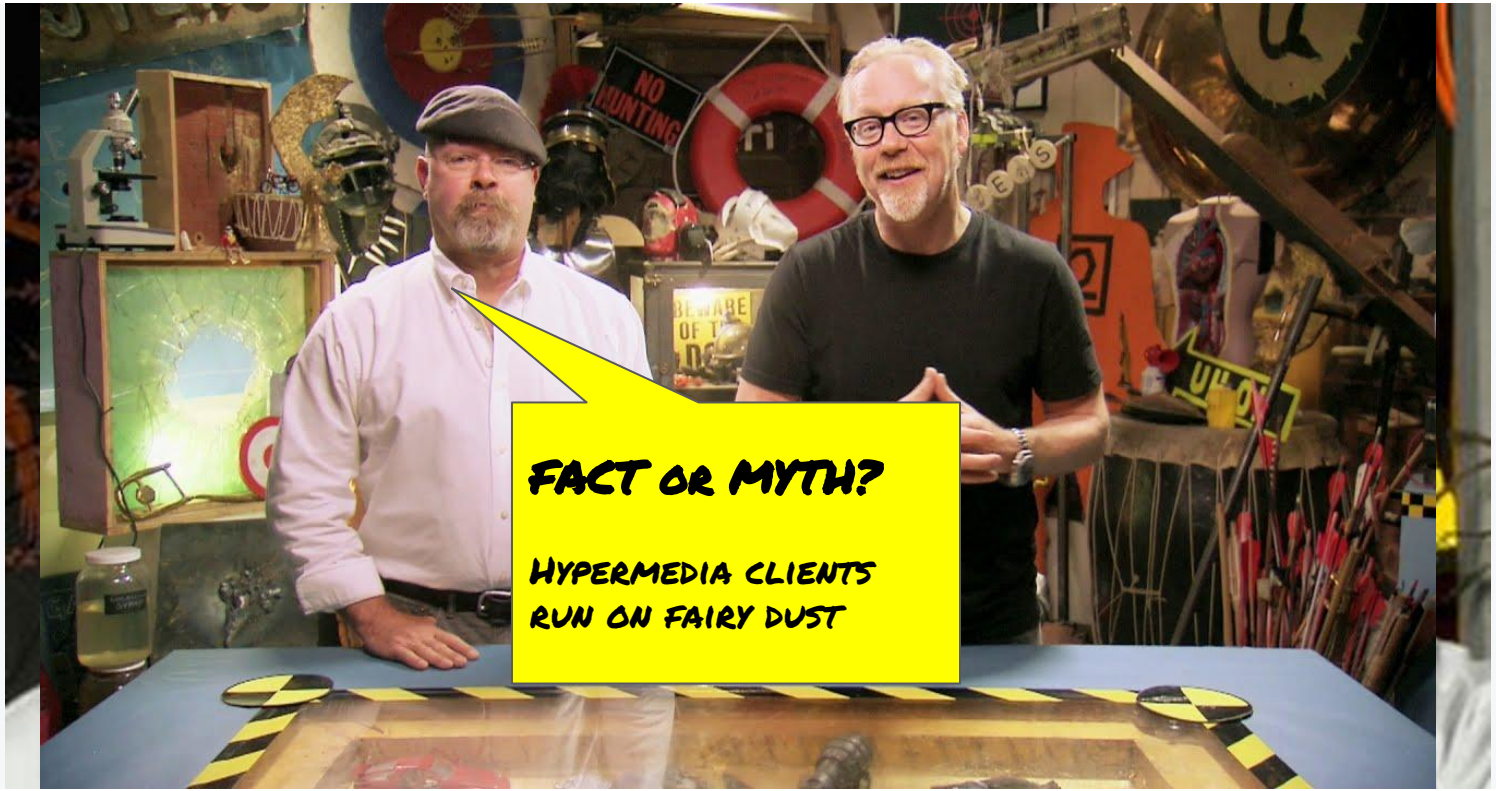
HTML APIs ***accessible*** to
- Programs
- Developers
- Other humans

Web accessible to
- Humans
- Programs?

COMCAST

cim COMCAST INTERACTIVE MEDIA

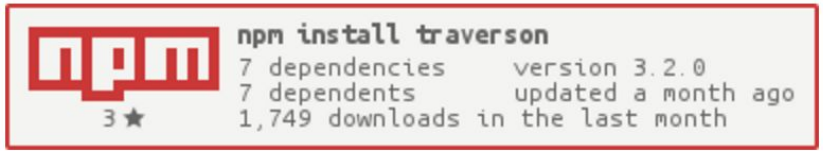

CONFIRMED


NO HUNTING
FACT OR MYTH?
HYPERMEDIA CLIENTS RUN ON FAIRY DUST


Faye

Traverson

A Hypermedia API/HATEOAS Client for Node.js and the Browser



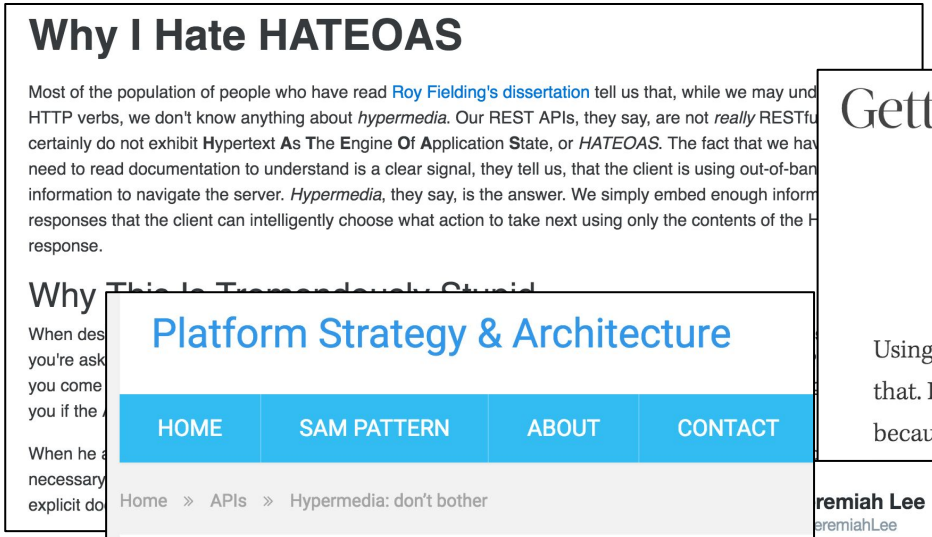
File Size (browser build)	KB
minified & gzipped	13
minified	44

JSON-LD

JSON-LD is a lightweight Linked Data format. It is easy for humans to read and write. It is based on the already successful JSON format and provides a way to help JSON data interoperate at Web-scale. JSON-LD is an ideal data format for programming environments, REST Web services, and unstructured databases such as CouchDB and MongoDB.


NO HUNTING
BUSTED


FACT OR MYTH?
HYPERMEDIA PROVIDES ACTUAL BUSINESS VALUE



https://je

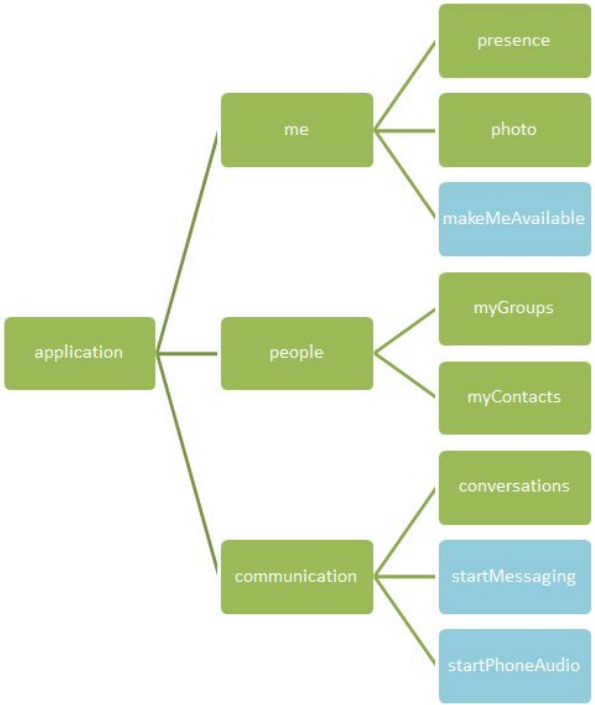
http://www.ebpml.org/blog15/2015/04/hypermedia-dont-bother/

r-about-hypermedia-apis

https://twitter.com/JeremiahLee/status/493830026983247876

Microsoft's Unified Communication Web API (UCWA)

Value - Discoverability



The Microsoft Unified Communications Web API 2.0 uses hypermedia links to discover and dynamically navigate the features that are available at any given time.

The client application starts with application creation and is presented with the features that are available in the returned application resource.

Some links are temporary and reflect the capabilities that the user has at a particular moment in time. For example, when a meeting attendee is in the lobby, an admit capability will appear. After the user is admitted, the admit link will no longer appear.

https://msdn.microsoft.com/en-us/skype/ucwa/hypermedia

Github API

Value - Version tolerance

Hypermedia

All resources may have one or more `*_url` properties linking to other resources. These are meant to provide explicit URLs so that proper API clients don't need to construct URLs on their own. It is highly recommended that API clients use these. Doing so will make future upgrades of the API easier for developers. All URLs are expected to be proper RFC 6570 URI templates.

You can then expand these templates using something like the uri_template gem:

```
>> tmpl = URITemplate.new('/notifications{?since,all,participating}')
>> tmpl.expand
=> "/notifications"

>> tmpl.expand :all => 1
=> "/notifications?all=1"

>> tmpl.expand :all => 1, :participating => 1
=> "/notifications?all=1&participating=1"
```

Auth0

https://developer.github.com/v3/#hypermedia

Guardian API

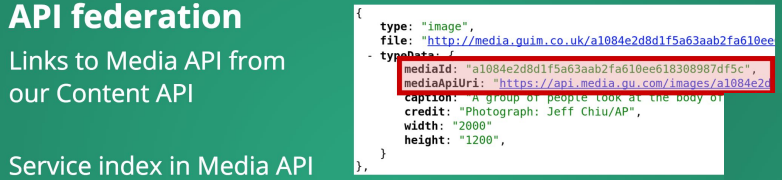
Value - Discoverability / Evolvability / Agility

Discoverable

API federation

Links to Media API from our Content API

Service index in Media API



Evolveable

1 Persistent URIs

3 Clients rely on links, not out-of-band config

Server configures clients

Only config is API root

No URL construction

Change link targets transparently

Extensible

Add data properties

Add links & form fields

Ignored if unsupported

Composeable

Add new services via links and forms

Shared media-type helps

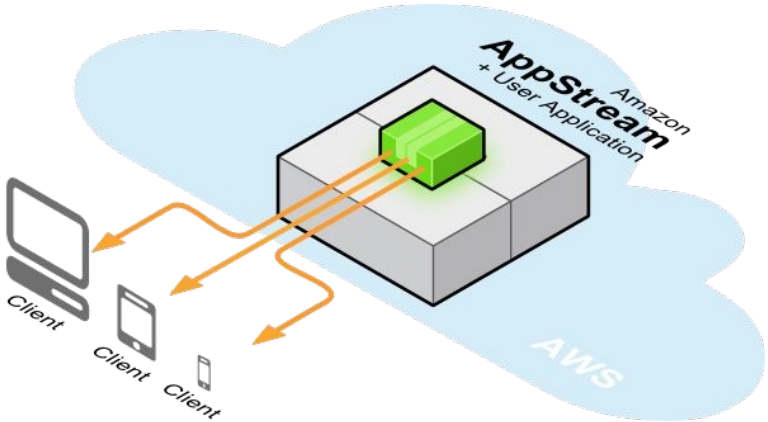
Avoids monolithic library

Auth0

http://slides.com/theefer/hypermedia-apidays

Amazon App Streaming API

Value - Discoverability



Link Relations

The Amazon AppStream API provides the following link relations that you can use to access and modify Amazon AppStream resources.

Topics

- appstream:applications
- application:by-id
- application:create
- application:update
- application:archive
- application:reactivate
- application:delete
- application:status
- application:errors
- application:sessions
- session:by-id
- session:entitle
- session:status
- session:terminate
- Common Link Relations

http://docs.aws.amazon.com/appstream/latest/developerguide/rest-api-link-relations.html

Gmail

Value - Discoverability

JSON-LD

JSON-LD is an easy-to-use JSON-based linked data format that defines the concept of `context` to specify the vocabulary for types and properties. Gmail supports JSON-LD data embedded in HTML documents with the `@context` of `schema.org`, as in the following example:

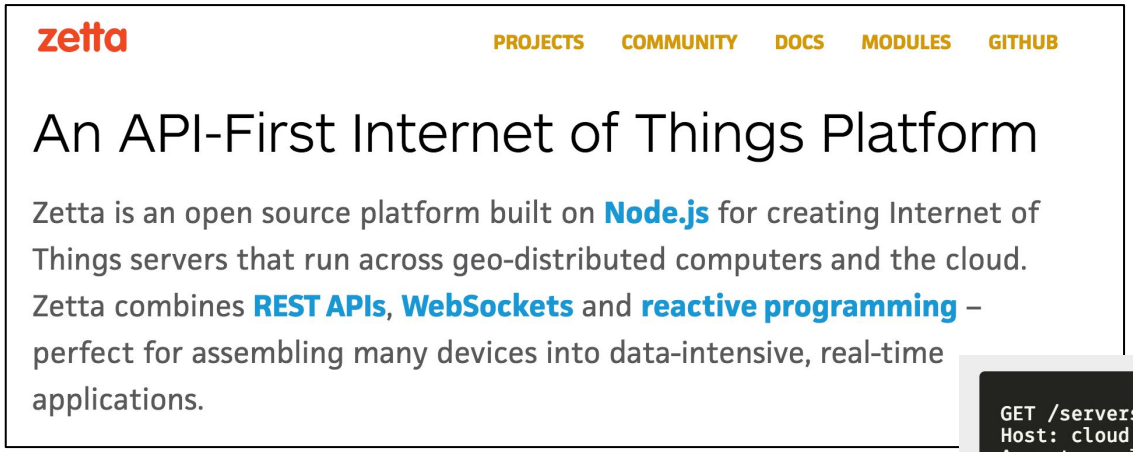
```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "Person",
  "name": "John Doe",
  "jobTitle": "Graduate research assistant",
  "affiliation": "University of Dreams",
  "additionalName": "Johnny",
  "url": "http://www.example.com",
  "address": {
    "@type": "PostalAddress",
    "streetAddress": "1234 Peach Drive",
    "addressLocality": "Wonderland",
    "addressRegion": "Georgia"
  }
}
</script>
```

Note: the `@type` key is a reserved key name and should contain either a full type URI or a URI fragment (in which case a `http://schema.org/` prefix, derived from the supplied data-context attribute, is assumed).

The full specifications and requirements for the JSON-LD syntax are available on json-ld.org, and you can also use our Schema Validator tool to try out JSON-LD and debug your markup.

Zetta

Value - Discoverability / Device management

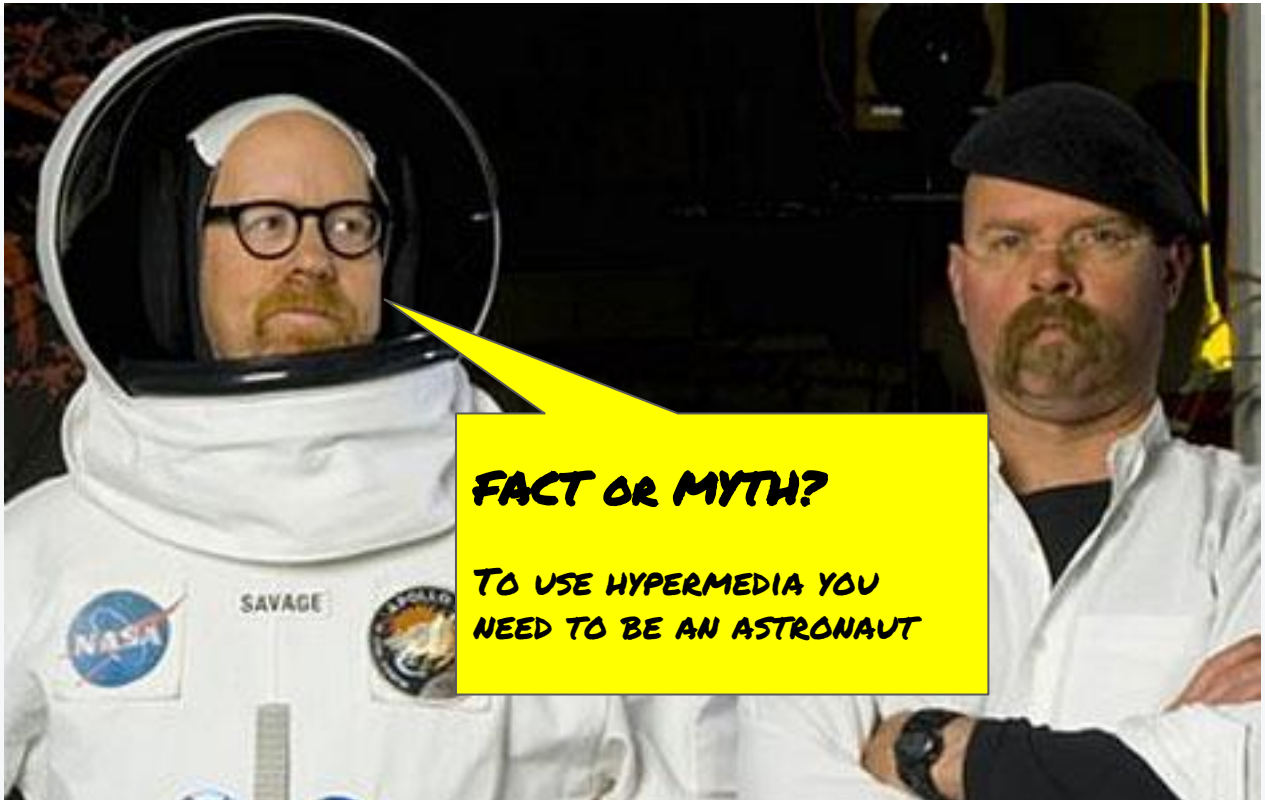


http://zettajs.org

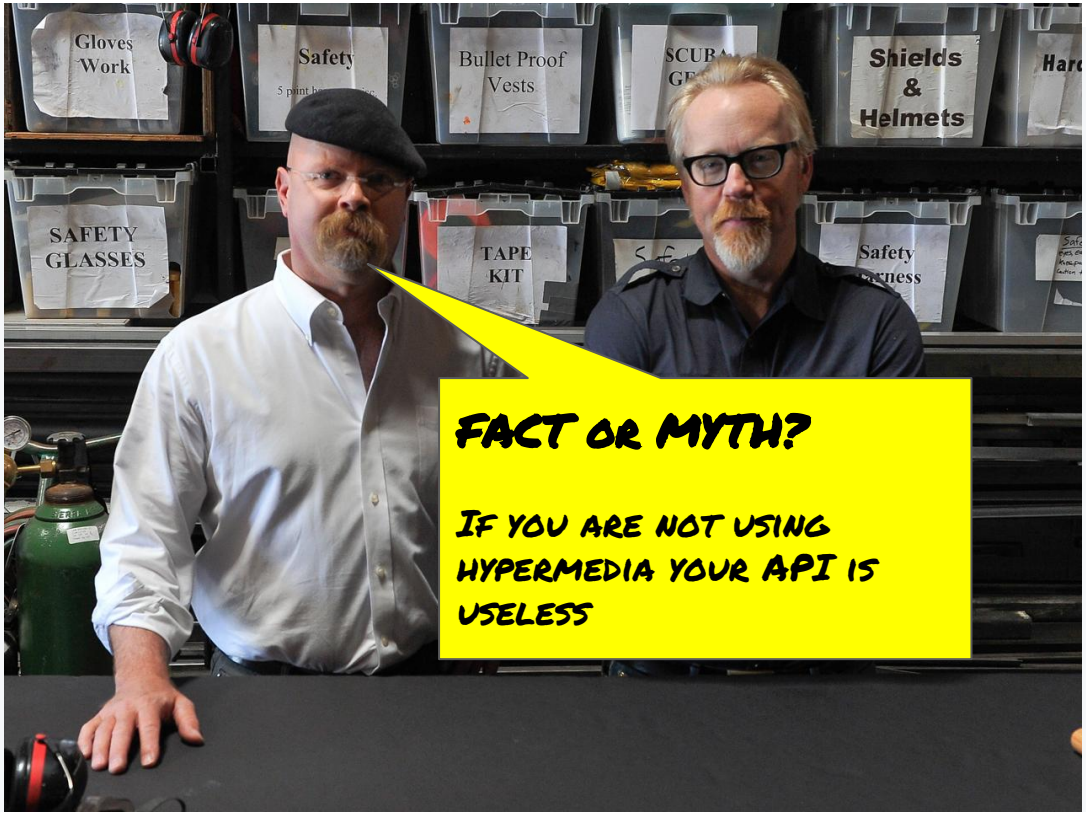
```
GET /servers/321/devices/123
Host: cloud.herokuapp.com
Accept: application/vnd.siren+json

{
  "class": [
   "device"
  ],
  "properties": {
   "id": "123",
   "type": "arm",
   "name": "Robot Arm",
   "state": "standby"
  },
  "actions": [{
   "name": "move-claw",
   "method": "POST",
```


CONFIRMED


SAVAGE
NASA
FACT OR MYTH?
TO USE HYPERMEDIA YOU NEED TO BE AN ASTRONAUT


BUSTED
SAVAGE
NASA


Gloves Work
Safety
Bullet Proof Vests
Shields & Helmets
SAFETY GLASSES
TAPE KIT
FACT OR MYTH?
IF YOU ARE NOT USING HYPERMEDIA YOUR API IS USELESS


Gloves
Work
Safety
Bullet Proof
Vests
Shields
&
Helmets
SAFETY
GLASSES
TAPE
KIT
BUSTED

Auth0

Takeaways

Hypermedia adoption is rising

Hypermedia does provide business value

You do not have to build a hypermedia API

More information on hypermedia

Mike Amundsen's books - http://www.oreilly.com/pub/au/1192

Steve Klabnik - https://www.infoq.com/articles/implementing-hypermedia

Darrel Miller - http://www.bizcoder.com/

JSON-LD - http://json-ld.org/

Glenn Block, Director of Product Management, Auth0