

RESTful Web Services and Drupal

Greg Hines

ping**VISION**TM

What is REST?

Well, It's Complicated

Well, It's Complicated

REST itself isn't complicated.
Defining REST is.

REST is...

Representation **S**tate **T**ransfer

REST is...

Representation State Transfer

“... an architectural style for distributed hypermedia systems.”

REST is...

Representation State Transfer

“... an architectural style for distributed hypermedia systems.”

A set of constraints for designing and implementing web services.

REST means using the fundamental web technologies (URI, HTTP, and hypermedia) idiomatically as the basis for web services.

REST doesn't necessarily mean "web."

REST's natural home is the web.

REST is...

- Addressability (URI)

Addressability

Every resource has at least one URI;
every URI points to a resource.

<http://example.com/>

<http://example.com/blog>

<http://example.com/blog/2008/05/14/restful-web-services-and-drupal>

<http://example.com/users>

<http://example.com/users/john-smith>

<http://example.com/links>

<http://example.com/links?offset=20>

<http://example.com/links/http%3A%2F%2Fexample.org%2Fexample%2Flink>

REST is...

- Addressability (URI)
- Interface Uniformity (HTTP)

Interface Uniformity

- GET
- POST

Interface Uniformity

- GET
- POST
- PUT
- DELETE
- HEAD
- OPTIONS
- TRACE
- CONNECT

Interface Uniformity

CRUD	HTTP
Create	PUT or POST
Read	GET, HEAD, or OPTIONS
Update	PUT
Delete	DELETE

REST is...

- Addressability (URI)
- Interface Uniformity (HTTP)
- Statelessness (HTTP)

Statelessness

- HTTP is a stateless protocol.
 - Every HTTP request happens in isolation.

Statelessness

- HTTP is a stateless protocol.
 - Every HTTP request happens in isolation.
- Possible server states *are* resources.
 - /search/term
 - /search/term?offset=10

Statelessness

- HTTP is a stateless protocol.
 - Every HTTP request happens in isolation.
- Possible server states *are* resources.
 - /search/term
 - /search/term?offset=10
- Stateless requests are infinitely repeatable.

Statelessness

- Resources can have state, which resides on the server.

Statelessness

- Resources can have state, which resides on the server.
- The application may have state, which resides on the client.

Statelessness

- Resources can have state, which resides on the server.
- The application may have state, which resides on the client.
- The relationship between the client and server is stateless.

Statelessness

- Resources can have state, which resides on the server.
- The application may have state, which resides on the client.
- The relationship between the client and server is stateless.
- By making a request (say, using PUT to update a resource), the application transfers its state to the resource.

Statelessness

- Resources can have state, which resides on the server.
- The application may have state, which resides on the client.
- The relationship between the client and server is stateless.
- By making a request (say, using PUT to update a resource), the application transfers its state to the resource.
- **Representation State Transfer**

REST is...

- Addressability (URI)
- Interface Uniformity (HTTP)
- Statelessness (HTTP)
- Connectedness (Hypermedia)

Connectedness

Resources link to each other:

```
<?xml version="1.0" encoding="UTF-8"?>  
<resource xmlns="http://example.com/ns/resource">  
  <content>...</content>  
  <nextResource href="http://example.com/resource/3" />  
  <prevResource href="http://example.com/resource/1" />  
</resource>
```

Connectedness

Resources link to each other:

```
<?xml version="1.0" encoding="UTF-8"?>
<resource xmlns="http://example.com/ns/resource">
  <content>...</content>
  <nextResource href="http://example.com/resource/3" />
  <prevResource href="http://example.com/resource/1" />
</resource>
```

```
<link rel="alternate" type="application/atom+xml"
  href="http://example.com/resource/2.atom" />
<a href="http://example.com/resource/3">...</a>
```

REST is...

- Addressability (URI)
- Interface Uniformity (HTTP)
- Statelessness (HTTP)
- Connectedness (Hypermedia)

URI + HTTP + Hypermedia

The Programmable Web

Look Familiar?

URI + HTTP + HTML

The Human Web

Every Web Site Is RESTful

“The Read-Only Web”

What REST Isn't

REST isn't...

- A Specification

REST isn't...

- A Specification
- A Data Format

REST isn't...

- A Specification
- A Data Format
- A Protocol

REST isn't...

- A Specification
- A Data Format
- A Protocol
- A Religion

Why use REST?

Simplicity

No Special Tools

The Web Was Designed For It

The Web Was Designed For It

(Sort of.)

Appeal to Authority

An appeal to authority or argument by authority is a type of argument in logic consisting on basing the truth value of an assertion on the authority, knowledge, expertise, or position of the person asserting it.

–Wikipedia

Roy T. Fielding, Ph.D.

- Co-founder, chairman of Apache Software Foundation
- Co-founder of Apache HTTP Project (a.k.a. httpd—the Apache web server)
- On IETF working groups for HTTP, URI, and HTML
 - Architectural principal on HTTP working group
 - First name on HTTP spec

Sam Ruby

- Board member, Apache Software Foundation
- Co-creator of Web Feed Validator
- Originator of Atom
- Secretary of IETF's AtomPub working group
- Co-author of "RESTful Web Services"

Greg Hines

- Currently giving a presentation about REST.

Designing a RESTful Web Service

Example Service

Digg Clone

Users submit links with descriptions, other users can comment on the link and add an up/down vote.

Users

/users

GET returns list of users.

/users/{username}

PUT creates a new user.

PUT updates a user's profile.

GET returns an existing user's profile.

DELETE removes the user.

/users/{username}/links

GET returns a list of links the user has submitted.

/users/{username}/comments

GET returns a list of links the user has commented on.

/users/{username}/votes

GET returns list of links the user has voted on.

Links

/links

/links?offset=20

GET retrieves a weighted list of recent links.

/links/{link-uri}

PUT creates the link resource with description.

PUT updates the link description.

GET returns information about the link, including comments and votes.

DELETE removes the link.

Comments

`/links/{link-uri}/comments`

GET retrieves all of the link's comments.

POST creates a new comment.

`/links/{link-uri}/comments/{comment-id}`

PUT updates the comment.

DELETE removes the comment.

Votes

`/link/{link-uri}/votes`

POST creates a new vote.

`/link/{link-uri}/votes/{vote-id}`

PUT updates the vote.

DELETE removes the vote.

REST and Drupal

Free Code

- User Management

Free Code

- User Management
- Resource Management

Free Code

- User Management
- Resource Management
- Routing

Drupal Isn't Perfect

Drupal is lacking...

- Controller

Drupal is lacking...

- Controller
- Multiple Representations for Resources

Drupal is lacking...

- Controller
- Multiple Representations for Resources
- Content Negotiation

Where Do We Go From Here?

REST API Module

Changes to Core

(Eventually.)