

Integrating Restfully

Srihari Srinivasan

ThoughtWorks

Web 2.0

Platform for software development



unlock data.
remix content.
unleash productivity.

IBM Mashup Center
Learn more.

- Home
- News
- Mashups
- APIs
- Verticals
- How-To
- Contests
- Members
- Dashboard
- Register
- Tour

Subscribe

[ProgrammableWeb](#)

Search

996 APIs

3456 Mashups

first time here?
take a tour

Topics

Social Platforms

Social Platforms

Topics

Keeping you up to date with mashups, APIs and the Web as platform. [Learn more »](#)

Find	Track	Do	Share
API Directory	Industry News	Learn	Sign-up
Mashup Directory	Market Trends	Search Code	Members
Most Popular	Major Players	Win Contests	Add Links
Mashup Matrix	Tag Cloud	Discuss	Talk to us

Latest News Updates [more news »](#)

Multimap Releases Free Map API »

Multimap, the UK-based online mapping provider that was acquired almost a year ago by Microsoft, has released a [new API](#) that allows users to embed customized Multimap maps in web pages using a single line of HTML code. This new functionality is similar to the existing custom map embedding functionality provided by [Google My Maps](#) and third party integrator [Map Channels](#). » Posted October 27, 2008. [Continue reading »](#)



ProgrammableWeb Sponsors

IBM Mashup Center
Unlock. Remix. Unleash.

Learn more.

APIs
from Orange
[find out how to use them ▶](#)

SnapLogic™
Integrate SaaS Today

OPEN SOURCE GET IT TODAY

Dada.net
ADD RINGTONES & MP3s TO YOUR SITE
[CLICK TO GET PAID ▶](#)

Subscribe
[All New APIs](#)

API Directory
Total APIs
996

Past 7 Days: 12
 Past 30 Days: 49

Our Sponsors

APIs from Orange

find out how to use them [▶](#)

Popular Directory Searches
[Celebrity Mashups](#)
[Video Mashups](#)
[Popular New](#)


API Dashboard

APIs news, how-to, contests and comprehensive database of API resources

October 27, 2008

Featured API »

[bkkeepr](#)



★★★★★

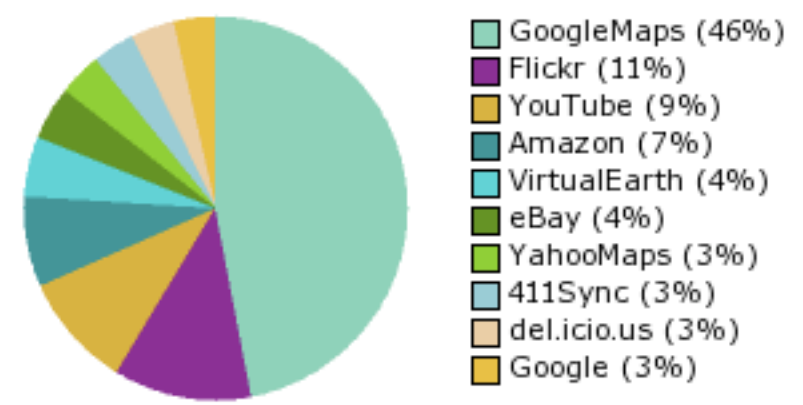
Newest APIs »

- [bkkeepr](#)
- [Best Buy Remix](#)
- [FlightStats](#)
- [Praized](#)
- [FinCRN](#)

[All APIs](#) | [By Category](#) | [By Mashups](#) | [Add More](#) | [How-To Guide](#)

Top APIs for Mashups »

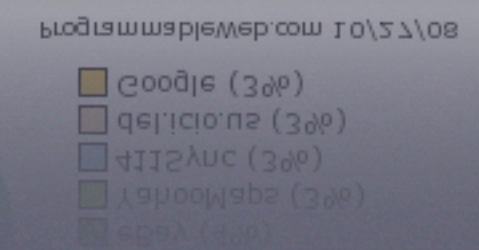
Last 14 days **All**



ProgrammableWeb.com 10/27/08

[Popular New](#)
[Video Mashups](#)
[Celebrity Mashups](#)

[Popular Directory](#)



Sort by: **Name** Date Popularity Category

Tags:

Category:

Company:

Protocol: SOAP

Date: All

Viewing 1 to 223 of 223 APIs

Viewing 1 to 223 of 223 APIs

Search

Sort by: **Name** Date Popularity Category

Tags:

Category:

Company:

Protocol: REST

Date: All

Viewing 1 to 624 of 624 APIs

Viewing 1 to 624 of 624 APIs

Search

|

:

3

Primer on REST

- REST - Representational State Transfer
- Term coined by *Roy Fielding* in his Ph.d thesis

UNIVERSITY OF CALIFORNIA, IRVINE

**Architectural Styles and
the Design of Network-based Software Architectures**

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

Ροϋ Τhomas Fielding

ρλ

in Information and Computer Science

Principled Architecture / Design

of

Large Scale Distributed Systems

Primer on REST

- Architectural style followed by the Web
- Its Generic, can be used to build other web-like systems

Qualities

- Performance, Scalability, Interoperability
- Simplicity, Extensibility
- Visibility of Interactions
- Reliability

REST intentionally places

Constraints on the system to achieve Desired Properties

REST - What it is not

- Not a standard, an architectural style
- Standards based on this style - HTTP, ATOM
- Ignores implementation details of components and protocol syntax

REST - What it is not

- Not the **ONLY** style for governing network interactions.
 - Event Driven Architectures
 - Message Oriented Architecture
 - and good old RPC...
- Most large systems have hybrid styles, one style may dominate though!

REST Constraints

REST Constraints

- Client-Server
- Statelessness
- Caching
- Layered System
- Uniform Interface
- Code on Demand (optional)

Client-Server Style of Interactions

- Based on the principle of separation of concerns
- Client and Server can be developed, maintained and scaled independently
- Desired Properties : Simplicity, Scalability, Portability of user interfaces

Uniform Interface

- The central/distinguishing feature of REST
- Most Controversial !

Uniform Interfaces are opposed to Specialized Interfaces

Specialized Interfaces

Employee Service

createEmployee(...)

updateEmployee(...)

getEmployeeDetails(identifier)

transferEmployee(identifier, to, ...)

deleteEmployee(identifier)

What happens when lots of such special interfaces
float in a distributed system ?

Specialization Consequences

- Clients have to know
 - Method names, parameters
 - Semantics of the methods
 - Possibly the order of method calls
- Each service interface is a new custom application protocol
- Clients cannot be generic

Specialization Consequences

- RPC style tries to map local computing notions to distribute operations
- Leads to data specialization
- Therefore couples format of the data to the interface

How does the Uniform Interface constraint help ?

Consequences of Specializing

- RPC/Service style interfaces promote specialization along two dimensions
 - Operations
 - Data

More Constraints

REST discourages sending ad hoc messages,
invoking special purpose functions

Resources

- Key Abstraction, Basic Unit of information
- Can be anything interesting
 - spreadsheet, image, xml file, transaction etc
- Uniquely identifiable via a URI

Resources

- Manipulated using representations - "Pass-by-value" semantics
- All resources manipulated via a limited set of standard functions/operations
 - In HTTP we have GET, POST, PUT & DELETE with well defined semantics for each

Resources

- Representations can be in any format
- Self describing messages help specify preferred formats
- Clients can even negotiate on formats with servers

How can a limited number of interfaces suffice
to perform many
domain specific operations ?

Resources

- REST promotes replacing the "do something" concept with a "make something so" concept
- “What” and not “How”
- Think Declarative, instead of Imperative

Hypermedia as the engine of application state

Hypermedia

- Representations must contain next steps/transitions a resource can make from the current state
- Use hyperlinks to define these transitions

Resource Orientation

- HTTP GET from <http://server.com/employees/name>
- HTTP POST to <http://server.coms/employees>
- HTTP PUT to <http://server.com/employees/name>
- HTTP POST to <http://server.com/employees/name/transfer>
- HTTP DELETE to <http://server.com/employees/name>

Resource Orientation

- Model Resources, instead of services
- Assign identifiers to resources
- Expose uniform operations on resources based on your chosen protocol/framework
- Support multiple formats - “One size does not fit all”
- Leverage hypermedia to traverse through finite states

Summary

- REST tries to reveal the distribution in distributed systems
- Popularizes HTTP outside the browser
- Reminds us that HTTP is not a transport protocol
- The ideas of REST have merit even if you are doing classic SOA
- Not easy to grok, requires unlearning and relearning!

```
</presentation>
```