Designing REST Services

Scott Seely
http://www.pluralsight.com/

Outline

- Service design styles
- Understanding REST
- Resource Oriented Architectures (ROA)
- Enabling Ajax/JSON Integration
- Syndication Programming Model
- OData/WCF Data Services

Service Design Styles

- **Remote Procedure Call**
  - Send some data, get a response
  - Interpretation of what to do expressed in URI, contents of message
  - Found in XML-RPC
  - Use POST to bypass HTTP caching
- **REST**
  - Spend lots of time designing resource structure
  - Define a uniform interface to interact with resources
  - Define which subset of verbs are available for each resource

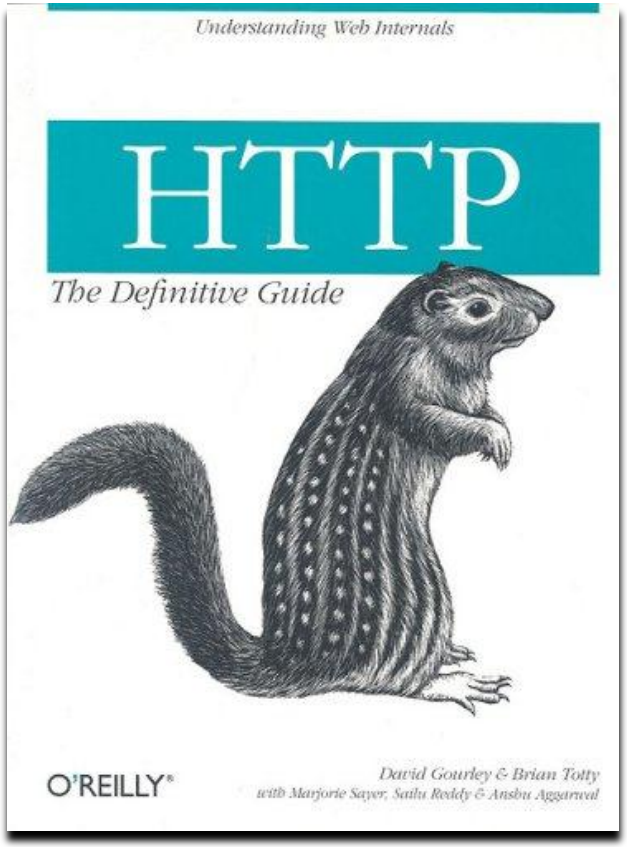
Understanding REST

- **Roy Fielding's Thesis:**
  - http://roy.gbiv.com/pubs/dissertation/top.htm -- Chapter 5
- **URL identifies resource, Method defines operations on the resource**
- **Methods on Resources**
  - POST : Create
  - GET | HEAD: Retrieve
  - PUT: Update
  - DELETE: Delete
  - OPTIONS: The list of methods (from above) that the resource supports.

Responses

- 2xx: Request worked
- 3xx: Resource moved
- 4xx: Client error
- 5xx: Server error

Just get a copy:



A Word on Behavior of HTTP Methods

- **Idempotent: operation can be applied multiple times, resource stays the same.**
- **POST: Non-cacheable. Used to create a resource. On success, response should contain the location of the newly created resource OR place to check on the status of the resource.**
- **PUT: Non-cacheable, idempotent. Used to modify a resource. Response may include the modified, complete object. Apply a modification more than once? Same as doing it once.**
- **GET: Cacheable based on cache headers**
  - Cache-Control: max-age. A time to live header indicating how long, in seconds, until the document goes stale.
  - Expires: Date Time. Absolute time that the document goes stale.
  - Can also use other headers to only request an update if the resource has changed: If-Modified-Since to check against date, If-None-Match to check against tags.

A Word on Behavior of HTTP Methods

- **HEAD: Same as GET, only don't return HTTP body (only metadata)**
- **DELETE: Idempotent. Delete once or a million times, result is the same.**
  - Handle a DELETE of an item that can't be found: 200 OK

Horror Story: MOOT vs. TIME

REFRESH DATA

Rank	Name	Avg. Rating	Total Votes
1	moot	90	16,794,368
2	Anwar Ibrahim	47	2,316,378
3	Rick Warren	45	1,902,383
4	Baitullah Mehsud	45	1,902,162
5	Larry Brilliant	44	2,005,310
6	Eric Holder	43	1,808,663
7	Carlos Slim	41	1,852,506
8	Angela Merkel	41	1,634,488
9	Kobe Bryant	39	1,976,880
10	Evo Morales	39	1,477,789
11	Alexander Lebedev	38	824,073
12	Lil' Wayne	37	939,993
13	Sheikh Ahmed bin Zayed Al Nahyan	36	838,578
14	Odell Barnes	35	916,836
15	Tina Fey	33	897,045
16	Hu Jintao	32	928,400
17	Eric Cantor	32	833,208
18	Gamal Mubarak	31	830,677
19	Ali al-Naimi	30	878,743
20	Muqtada al-Sadr	29	810,573
21	Elizabeth Warren	28	2,320,902
22	Manny Pacquiao	27	20,391,818
23	Rain	18	12,762,228
24	Paul Kagame	18	3,890,609

http://musicmachinery.com/2009/04/27/moot-wins-time-inc-loses/

Mapping HTTP Methods to WCF Contracts

- **UriTemplate: Defines how to map a URL to a method. Takes the place of Action in an OperationContract.**
  - Structure: "/fixed/{argument}/fixed/{argument}?value={argument}&value={argument}"
  - Query string parameters are matched by name/value pair.
  - Path parameters matched by location.

```
[WebGet(UriTemplate = "/Demo/{name}")]
public string SayHello(string name)
```

  - Parameter type must convert to and from string.
- **Attributes to use:**
  - GET: WebGet
  - Everything else: WebInvoke(Method = "*method*", etc…)
  - Both allow for RequestFormat/ResponseFormat properties to set response to XML or JSON. In 4.0, this can be figured out by reading request, mapping as appropriate.

Resource Oriented Architecture

- **Resource definition**
  - Single unit of addressable, updatable content.
  - Has relationship to other content.
  - Can be viewed, modified, or deleted via a uniform interface
- **A resource is addressable**
  - A resource can be accessed using a URL
  - May have multiple URLs that point to the same resource. Ex:
    - http://www.pluralsight.com/modules/Designing-Rest/v1.0
    - http://www.pluralsight.com/modules/Designing-Rest/current
- **A resource may have multiple representations**
  - XML: For XML→Object mapping tools
  - JSON: For browsers
  - Spanish/English/Japanese/etc. for different translations
  - Etc.

Resource Oriented Architecture

- **Creating address structures**
  - What entities do you have?
  - What entities do you want to access independently?
  - How do the entities relate?
  - Name the entities
- **Example: Map**
  - Define points:
    - Postal Address
    - Latitude/Longitude
  - Define viewable area:
    - Rectangle in terms of 2 points
    - Rectangle centered on single point
  - Define scale:
    - X miles/inch
    - X km/cm

Enabling Ajax/JSON Integration

- **Goal: Create services that work with just your web pages**
- **What you get:**
  - Automatic script generation
  - Automatic callback support
  - Easy integration
- **Factory=**

  "System.ServiceModel.Activiation.WebScriptServiceHostFactory"

  in a .svc file.

Syndication Programming Model

- **Create/Consume Syndication formats:**
  - RSS: Really Simple Syndication
  - Atom
- **WCF provides abstraction over syndication**
  - Use WebGet
  - Return SyndicationFeedFormatter
- **SyndicationFeed: Container for SyndicationItem**
  - Set name, description, location, and image
- **SyndicationItem: Container for data being syndicated**
  - Content: Holds HTML, XML, URL, or plaintext
  - Categories: Links to the categories
  - Authors, Contributors, Copyright, and other feed data.

Using Syndication

- **Use for read-only data**
- **Data tends to be published once, never modified**
  - Blogs
  - Reports
    - Status of system, warehouse, sales, etc. on [date]
    - Activity by user, by date
- **Code can also consume RSS and Atom**
  - SyndicationFeed.Load
  - Full access to SyndicationItem

OData

- **OData allows for**
  - Querying data
  - Updating data
- **Builds on**
  - HTTP
  - Atom Publishing Protocol
  - JSON
- **Lots of libraries for other platforms, so highly portable.**
- **Works without libraries too, it's XML, JSON, and APP**

OData Terms

- **Feed: Collection of typed entries**
- **Entry: structured record with**
  - A key
  - List of properties of primitive and complex types
- **Entries may express associations to other entries and feeds through links**
- **May expose service operations: service specific functions that accept input and return entries or complex/primitive values**
- **Discovery: supports Service Metadata Document**
- **Abstract Data Model described using Conceptual Schema Definition Language: CSDL**
- **Security: Supports HTTPS and other HTTP based authentication schemes**

OData on .NET

- **Entity Data Model describes structure.**
- **IQueryable<T> properties are exported**
- **DataService<T> hosts the service**
  - Describes security
  - Intercepts calls
  - Adds service operations
- **DataServiceKey: Use this to identify which properties on your exposed objects are keys.**

```
[DataServiceKey("ID")]
public class SomeClass
```

- **Hosting a DataService: use System.Data.Services.DataServiceHostFactory**

OData Querying

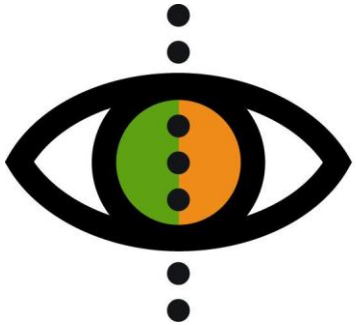
- Rich mechanism to find items by field
- Provides for paging of results
- Can extract metadata and more.

Summary

- REST allows for a large pool of consumers because it build on HTTP and the Web
- WCF has support for REST: control UriTemplate and methods, rely on RSS and Atom
- OData and WCF Data Services provides a rich mechanism for producing REST applications, links between entities, and querying data at minimal effort for the developer

- OData Introduction: http://www.pluralsight-training.net/microsoft/olt/Course.aspx?n=odata-introduction

For more in-depth online developer training visit



pluralsight

see what you can learn

on-demand content from authors you trust