# Cannonical REST Service

This document describes one model for creating a RESTful service that returns a collection of entities. This form can be implemented with a variety of technologies. This version was influenced by the capabilities of the [WCF WebApi](http://wcf.codeplex.com/wikipage?title=WCF%20HTTP) project

## Cannonical URI Scheme

Base URI: <http://tempuri.org/Resource> where Resource is the name of the REST collection (i.e. Customers, Orders etc.)

Web Formats XML and JSON are supported for all request/response messages

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Method URI Template | HTTP Method | Description | Responses | | |
| **Status** |  | **Response Content** |
| /{key} | GET | Gets a resource by key | 200 | OK | Serialized resource |
| 304 | Not Modified | Empty |
| 400 | Bad Request | Empty or Serialized error message |
| 404 | Not Found | Empty |
| / | GET | Gets a list of resources | 200 | OK | Serialized resources |
| 400 | Bad Request | Empty or Serialized error message |
| / | POST | Adds a resource to the collection | 200 | OK | Serialized resource |
| 204 | No Content | Empty |
| 400 | Bad Request | Empty or Serialized error message |
| 409 | Conflict | Empty or Serialized error message |
| /{key} | PUT | Adds or Updates a resource identified by {key}. Some services might not allow add with PUT.  If you return the updated resource, return 200. If you don't return 204. | 200 | OK | Updated resource |
| 204 | No Content | Empty |
| 400 | Bad Request | Empty or Serialized error message |
| 404 | Not Found | Empty or Serialized error message |
| 409 | Conflict | Empty or Serialized error message |
| /{key} | DELETE | Removes the resource identified by {key}. If you return the resource removed, return 200. If you don't return 204. | 200 | OK | Deleted Resource |
| 204 | No Content | Empty |
| 400 | Bad Request | Empty or Serialized error message |
| 404 | Not Found | Empty or Serialized error message |
| 409 | Conflict | Empty or Serialized error message |

## GET Tests

GET Spec <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.3>

Tests for template **GET** **/{key}**

1. GET MUST return a resource given a key if the resource with that key exists
2. GET MUST return 400-BadRequest if the key is invalid
3. GET MUST return 404-NotFound if the key is not found
4. GET MUST return 304-NotModified if Conditional GET conditions are met using If-None-Match
5. GET SHOULD return an ETag header

Tests for template **GET** **/**

1. GET MUST return resources in the collection
2. GET SHOULD support [**OData URI Conventions**](http://www.odata.org/developers/protocols/uri-conventions)

## POST Tests

POST Spec <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.5>

Tests **POST /**

1. POST MAY append a valid resource to the resource collection using a server generated key.
2. POST MAY return 201 – Created if it wants to let the client know that it created a new resource.
3. POST MAY return 202 - Accepted if it accepts a request for later processing
4. If POST returns 201 - Created, The response MUST include a location header, entity tag and entity body
5. POST MUST return 400-Bad Request if the entity is invalid
6. POST MUST return 409-Conflict if the entity conflicts with another entity
7. POST MUST ignore writes to entity fields the server considers read only

## PUT Tests

PUT Spec <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.6>

Tests **PUT /{key}**

1. PUT MUST Update the entity identified by the URI if it exists and return 200-OK with the modified entity and ETag header
2. PUT MAY Add a new entity using the key provided in the URI and return 201-Created with entity location and ETag
3. PUT MUST respect the Precondition If-Match
4. PUT MUST be Idempotent
5. PUT MUST NOT alter the key of the entity so that it does not match the key of the URI
6. PUT MUST return 400-BadRequest if the entity is invalid
7. PUT MUST return 400-BadRequest if the key is invalid
8. PUT MUST ignore writes to entity fields the server considers read only
9. PUT MUST return 404-NotFound if the server does not allow new entities to be added with PUT

## Delete TESTS

DELETE Spec <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.7>

Tests DELETE **/{key}**

1. DELETE SHOULD delete an entity that exists and return 200-OK with the deleted entity or 204-No Content if the response does not include the entity
2. DELETE SHOULD be idempotent
3. DELETE SHOULD return with 412-PreconditionFailed if no matching entity for If-Match ETag
4. DELETE SHOULD succeed if matching entity for If-Match ETag
5. DELETE SHOULD succeed if wildcard used in If-Match ETag
6. DELETE SHOULD return 202-Accepted if the request to delete has not been enacted
7. DELETE SHOULD return 400-BadRequest if the key is invalid