Understand OData

OData (Open Data Protocol) is an [OASIS standard](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=odata) that defines the best practice for building and consuming RESTful APIs. OData helps you focus on your business logic while building RESTful APIs without having to worry about the approaches to define request and response headers, status codes, HTTP methods, URL conventions, media types, payload formats and query options etc. OData also guides you about tracking changes, defining functions/actions for reusable procedures and sending asynchronous/batch requests etc. Additionally, OData provides facility for extension to fulfil any custom needs of your RESTful APIs.

OData RESTful APIs are easy to consume. The OData metadata, a machine-readable description of the data model of the APIs, enables the creation of powerful generic client proxies and tools. Some of them can help you interact with OData even without knowing anything about the protocol.

Requesting Data

OData services support requests for data via HTTP GET requests.

### Requesting Entity Collections

The request below returns the the collection of Person People.  
GET serviceRoot/People   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"@odata.nextLink": "serviceRoot/People?%24skiptoken=8",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694BD49A0F11"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404796846280400

},

......

,

{

"@odata.id": "serviceRoot/People('keithpinckney')",

"@odata.etag": "W/"08D1694BD49A0F11"",

"@odata.editLink": "serviceRoot/People('keithpinckney')",

"UserName": "keithpinckney",

"FirstName": "Keith",

"LastName": "Pinckney",

"Emails": [

"Keith@example.com",

"Keith@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404796846280400

}

]

}

### Requesting an Individual Entity by ID

The request below returns an individual entity of type Person by the given UserName "russellwhyte"  
GET serviceRoot/People('russellwhyte')   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People/$entity",

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694BF26D2BC9"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404797346655200

}

### Requesting an Individual Property

To address an entity property clients append a path segment containing property name to the URL of the entity. If the property has a complex type, properties of that value can be addressed by further property name composition.  
First let's take a look at how to get a simple property. The request below returns the Name property of an Airport.

GET serviceRoot/Airports('KSFO')/Name   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#Airports('KSFO')/Name",

"value": "San Francisco International Airport"

}

Then let's see how to get a property value of a complex type. The request below returns the Address of the complex type Location in an Airport.  
GET serviceRoot/Airports('KSFO')/Location/Address   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#Airports('KSFO')/Location/Address",

"value": "South McDonnell Road, San Francisco, CA 94128"

}

### Requesting an Individual Property Raw Value

To address the raw value of a primitive property, clients append a path segment containing the string $value to the property URL. The request below returns the raw value of property Name of an Airport.  
GET serviceRoot/Airports('KSFO')/Name/$value   
  
Response Payload

San Francisco International Airport

## Querying Data

OData supports various kinds of query options for querying data. This section will help you go through the common scenarios for these query options.

### System Query Option $filter

The $filter system query option allows clients to filter a collection of resources that are addressed by a request URL. The expression specified with $filter is evaluated for each resource in the collection, and only items where the expression evaluates to true are included in the response.

#### Basic predicates, built-in functions

There are several kinds of basic predicates and built-in functions for $filter, including logical operators and arithmetic operators. For more detailed information, please refer to [OData V4 URL Conventions Document](http://docs.oasis-open.org/odata/odata/v4.0/odata-v4.0-part2-url-conventions.html). The request below using $filter to get people with FirstName "Scott".  
GET serviceRoot/People?$filter=FirstName eq 'Scott'  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('scottketchum')",

"@odata.etag": "W/"08D1694C7E510464"",

"@odata.editLink": "serviceRoot/People('scottketchum')",

"UserName": "scottketchum",

"FirstName": "Scott",

"LastName": "Ketchum",

"Emails": [

"Scott@example.com"

],

"AddressInfo": [

{

"Address": "2817 Milton Dr.",

"City": {

"CountryRegion": "United States",

"Name": "Albuquerque",

"Region": "NM"

}

}

],

"Gender": "Male",

"Concurrency": 635404799693620400

}

]

}

#### Filter on Complex Type

$filter can also work on complex type. The request below returns airports with "San Francisco" contained in its Address. And Address is property of complex type Location.

GET serviceRoot/Airports?$filter=contains(Location/Address, 'San Francisco')  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#Airports",

"value": [

{

"@odata.id": "serviceRoot/Airports('KSFO')",

"@odata.editLink": "serviceRoot/Airports('KSFO')",

"IcaoCode": "KSFO",

"Name": "San Francisco International Airport",

"IataCode": "SFO",

"Location": {

"Address": "South McDonnell Road, San Francisco, CA 94128",

"City": {

"CountryRegion": "United States",

"Name": "San Francisco",

"Region": "California"

},

"Loc": {

"type": "Point",

"coordinates": [

-122.374722222222,

37.6188888888889

],

"crs": {

"type": "name",

"properties": {

"name": "EPSG:4326"

}

}

}

}

}

]

}

#### Filter on Enum Properties

The request below returns all female People of entity type Person. The Gender is a property of Enum type.  
GET serviceRoot/People?$filter=Gender eq Microsoft.OData.SampleService.Models.TripPin.PersonGender'Female'   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"@odata.nextLink": "serviceRoot/People?%24filter=Gender+eq+Microsoft.OData.SampleService.Models.TripPin.PersonGender%27Female%27&%24skiptoken=8",

"value": [

{

"@odata.id": "serviceRoot/People('elainestewart')",

"@odata.etag": "W/"08D1694CCFB34453"",

"@odata.editLink": "serviceRoot/People('elainestewart')",

"UserName": "elainestewart",

"FirstName": "Elaine",

"LastName": "Stewart",

"Emails": [

"Elaine@example.com",

"Elaine@contoso.com"

],

"AddressInfo": [],

"Gender": "Female",

"Concurrency": 635404801059013800

},

......

,

{

"@odata.id": "serviceRoot/People('ursulabright')",

"@odata.etag": "W/"08D1694CCFB34453"",

"@odata.editLink": "serviceRoot/People('ursulabright')",

"UserName": "ursulabright",

"FirstName": "Ursula",

"LastName": "Bright",

"Emails": [

"Ursula@example.com",

"Ursula@contoso.com"

],

"AddressInfo": [],

"Gender": "Female",

"Concurrency": 635404801059013800

}

]

}

#### Nested Filter in Expand

OData V4 supports nested filters in $expand. The request below return People and all their trips with Name "Trip in US".  
GET serviceRoot/People?$expand=Trips($filter=Name eq 'Trip in US')  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"@odata.nextLink": "serviceRoot/People?%24expand=Trips(%24filter%3dName+eq+%27Trip+in+US%27)&%24skiptoken=8",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694D0DBBD9DB"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404802099763700,

"Trips@odata.context": "serviceRoot/$metadata#People('russellwhyte')/Trips",

"Trips": [

{

"TripId": 1001,

"ShareId": "9d9b2fa0-efbf-490e-a5e3-bac8f7d47354",

"Description": "Trip from San Francisco to New York City",

"Name": "Trip in US",

"Budget": 3000,

"StartsAt": "2014-01-01T00:00:00Z",

"EndsAt": "2014-01-04T00:00:00Z",

"Tags": [

"business",

"New York meeting"

]

}

]

},

......

,

{

"@odata.id": "serviceRoot/People('keithpinckney')",

"@odata.etag": "W/"08D1694D0DBBD9DB"",

"@odata.editLink": "serviceRoot/People('keithpinckney')",

"UserName": "keithpinckney",

"FirstName": "Keith",

"LastName": "Pinckney",

"Emails": [

"Keith@example.com",

"Keith@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404802099763700,

"Trips@odata.context": "serviceRoot/$metadata#People('keithpinckney')/Trips",

"Trips": []

}

]

}

### System Query Option $orderby

The $orderby system query option allows clients to request resources in either ascending order using asc or descending order using desc. If asc or desc not specified, then the resources will be ordered in ascending order. The request below orders Trips on property EndsAt in descending order.

GET serviceRoot/People('scottketchum')/Trips?$orderby=EndsAt desc  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People('scottketchum')/Trips",

"value": [

{

"TripId": 2004,

"ShareId": "f94e9116-8bdd-4dac-ab61-08438d0d9a71",

"Description": "Trip from Shanghai to Beijing",

"Name": "Trip in Beijing",

"Budget": 11000,

"StartsAt": "2014-02-01T00:00:00Z",

"EndsAt": "2014-02-04T00:00:00Z",

"Tags": [

"Travel",

"Beijing"

]

},

{

"TripId": 2002,

"ShareId": "9d9b2fa0-efbf-490e-a5e3-bac8f7d47354",

"Description": "Trip from San Francisco to New York City",

"Name": "Trip in US",

"Budget": 5000,

"StartsAt": "2014-01-01T00:00:00Z",

"EndsAt": "2014-01-04T00:00:00Z",

"Tags": [

"business",

"New York meeting"

]

}

]

}

### System Query Option $top and $skip

The $top system query option requests the number of items in the queried collection to be included in the result. The $skip query option requests the number of items in the queried collection that are to be skipped and not included in the result.  
The request below returns the first two people of the People entity set.

GET serviceRoot/People?$top=2   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694D4AA4A8D1"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404803121654000

},

{

"@odata.id": "serviceRoot/People('scottketchum')",

"@odata.etag": "W/"08D1694D4AA4A8D1"",

"@odata.editLink": "serviceRoot/People('scottketchum')",

"UserName": "scottketchum",

"FirstName": "Scott",

"LastName": "Ketchum",

"Emails": [

"Scott@example.com"

],

"AddressInfo": [

{

"Address": "2817 Milton Dr.",

"City": {

"CountryRegion": "United States",

"Name": "Albuquerque",

"Region": "NM"

}

}

],

"Gender": "Male",

"Concurrency": 635404803121654000

}

]

}

The request below returns people starting with the 19th people of the entity set People

GET serviceRoot/People?$skip=18   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('genevievereeves')",

"@odata.etag": "W/"08D1694D8B408D60"",

"@odata.editLink": "serviceRoot/People('genevievereeves')",

"UserName": "genevievereeves",

"FirstName": "Genevieve",

"LastName": "Reeves",

"Emails": [

"Genevieve@example.com",

"Genevieve@contoso.com"

],

"AddressInfo": [],

"Gender": "Female",

"Concurrency": 635404804205612400

},

{

"@odata.id": "serviceRoot/People('kristakemp')",

"@odata.etag": "W/"08D1694D8B408D60"",

"@odata.editLink": "serviceRoot/People('kristakemp')",

"UserName": "kristakemp",

"FirstName": "Krista",

"LastName": "Kemp",

"Emails": [

"Krista@example.com"

],

"AddressInfo": [],

"Gender": "Female",

"Concurrency": 635404804205612400

}

]

}

### System Query Option $count

The $count system query option allows clients to request a count of the matching resources included with the resources in the response.  
The request below returns the total number of people in the collection.

GET serviceRoot/People?$count=true  
  
Response Payload

20

### System Query Option $expand

The $expand system query option specifies the related resources to be included in line with retrieved resources. The request below returns people with navigation property Friends of a Person

GET serviceRoot/People('keithpinckney')?$expand=Friends   
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People/$entity",

"@odata.id": "serviceRoot/People('keithpinckney')",

"@odata.etag": "W/"08D1694E2BB4317A"",

"@odata.editLink": "serviceRoot/People('keithpinckney')",

"UserName": "keithpinckney",

"FirstName": "Keith",

"LastName": "Pinckney",

"Emails": [

"Keith@example.com",

"Keith@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404806897545600,

"Friends": [

{

"@odata.id": "serviceRoot/People('clydeguess')",

"@odata.etag": "W/"08D1694E2BB4317A"",

"@odata.editLink": "serviceRoot/People('clydeguess')",

"UserName": "clydeguess",

"FirstName": "Clyde",

"LastName": "Guess",

"Emails": [

"Clyde@example.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404806897545600

},

{

"@odata.id": "serviceRoot/People('marshallgaray')",

"@odata.etag": "W/"08D1694E2BB4317A"",

"@odata.editLink": "serviceRoot/People('marshallgaray')",

"UserName": "marshallgaray",

"FirstName": "Marshall",

"LastName": "Garay",

"Emails": [

"Marshall@example.com",

"Marshall@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404806897545600

}

]

}

### System Query Option $select

The $select system query option allows the clients to requests a limited set of properties for each entity ~~or complex type~~. The request below returns Name and IcaoCode of all Airports.

GET serviceRoot/Airports?$select=Name, IcaoCode  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#Airports(Name,IcaoCode)",

"value": [

{

"@odata.id": "serviceRoot/Airports('KSFO')",

"@odata.editLink": "serviceRoot/Airports('KSFO')",

"Name": "San Francisco International Airport",

"IcaoCode": "KSFO"

},

......

,

{

"@odata.id": "serviceRoot/Airports('KJFK')",

"@odata.editLink": "serviceRoot/Airports('KJFK')",

"Name": "John F. Kennedy International Airport",

"IcaoCode": "KJFK"

}

]

}

### System Query Option $search

The $search system query option restricts the result to include only those entities matching the specified search expression. The definition of what it means to match is dependent upon the implementation. The request below get all People who has 'Boise' in their contents.

serviceRoot/People?$search=Boise  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694E90C0914C"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404808592855400

}

]

}

### Lambda Operators

OData defines two operators any and all that evaluate a Boolean expression on a collection. They can work on either collection properties or collection of entities.

The request below returns People with Emails containing "ll@contoso.com". The Emails is a collection of primitive type string.

GET serviceRoot/People?$filter=Emails/any(s:endswith(s, 'contoso.com'))  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694ECB6F8E7D"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404809577402000

},

{

"@odata.id": "serviceRoot/People('marshallgaray')",

"@odata.etag": "W/"08D1694ECB6F8E7D"",

"@odata.editLink": "serviceRoot/People('marshallgaray')",

"UserName": "marshallgaray",

"FirstName": "Marshall",

"LastName": "Garay",

"Emails": [

"Marshall@example.com",

"Marshall@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404809577402000

}

]

}

The request below returns the friends of Me who have friends using "Scott" as their FirstName.

GET serviceRoot/Me/Friends?$filter=Friends/any(f:f/FirstName eq 'Scott')  
  
Response Payload

{

"@odata.context": "serviceRoot/$metadata#People",

"value": [

{

"@odata.id": "serviceRoot/People('russellwhyte')",

"@odata.etag": "W/"08D1694EE92CB5C3"",

"@odata.editLink": "serviceRoot/People('russellwhyte')",

"UserName": "russellwhyte",

"FirstName": "Russell",

"LastName": "Whyte",

"Emails": [

"Russell@example.com",

"Russell@contoso.com"

],

"AddressInfo": [

{

"Address": "187 Suffolk Ln.",

"City": {

"CountryRegion": "United States",

"Name": "Boise",

"Region": "ID"

}

}

],

"Gender": "Male",

"Concurrency": 635404810076337700

},

{

"@odata.id": "serviceRoot/People('ronaldmundy')",

"@odata.etag": "W/"08D1694EE92CB5C3"",

"@odata.editLink": "serviceRoot/People('ronaldmundy')",

"UserName": "ronaldmundy",

"FirstName": "Ronald",

"LastName": "Mundy",

"Emails": [

"Ronald@example.com",

"Ronald@contoso.com"

],

"AddressInfo": [],

"Gender": "Male",

"Concurrency": 635404810076337700

}

]

}

## Data Modification

Updatable OData services support Create, Update and Delete operation for some or all exposed entities.

### Create an Entity

To create an entity in a collection, the client sends a POST request to that collection's URL. The POST body MUST contain a single valid entity representation. The request below creates a Person which contains complex type and collection property.

POST serviceRoot/People  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json

{  
"@odata.type" : "Microsoft.OData.SampleService.Models.TripPin.Person",  
"UserName": "teresa",  
"FirstName" : "Teresa",  
"LastName" : "Gilbert",  
"Gender" : "Female",  
"Emails" : ["teresa@example.com", "teresa@contoso.com"],  
"AddressInfo" : [  
{  
"Address" : "1 Suffolk Ln.",  
"City" :  
{  
"CountryRegion" : "United States",  
"Name" : "Boise",  
"Region" : "ID"  
}  
}]  
}  
  
  
Response Payload

HTTP/1.1 201 Created

Content-Length: 468

Content-Type: application/json;odata.metadata=minimal;odata.streaming=true;IEEE754Compatible=false;charset=utf-8

Location: serviceRoot/People('teresa')

OData-Version: 4.0

{

"@odata.context":"serviceRoot/$metadata#People/$entity",

"@odata.id":"serviceRoot/People('teresa')",

"@odata.editLink":"serviceRoot/People('teresa')",

"UserName":"teresa",

"FirstName":"Teresa",

"LastName":"Gilbert",

"Emails":["teresa@example.com","teresa@contoso.com"],

"AddressInfo":[{"Address":"1 Suffolk Ln.","City":{"CountryRegion":"United States","Name":"Boise","Region":"ID"}}],

"Gender":"Female"

}

### Remove an Entity

The request below deletes the Person with UserName 'vincentcalabrese'.  
DELETE serviceRoot/People('vincentcalabrese')   
  
Response Payload

HTTP/1.1 204 No Content

OData-Version: 4.0

### Update an Entity

The OData services SHOULD support PATCH as the preferred means of updating an entity. But also services MAY additionally support PUT. The request below update the Emails of a person using PATCH.

PATCH serviceRoot/People('russellwhyte')  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json

{  
"@odata.type" : "Microsoft.OData.SampleService.Models.TripPin.Person",  
"Emails" : ["Russell@example.com", "Russell@contoso.com", "newRussell@contoso.com"]  
}  
  
  
Response Payload

HTTP/1.1 204 No Content

### Relationship Operations

Relationships from one entity to another are represented as navigation properties.

#### Link to Related Entities

A successful POST request to a navigation property's references collection adds a relationship to an existing entity. The request below adds 'vincentcalabrese' to friends of 'scottketchum'.  
  
POST serviceRoot/People('scottketchum')/Friends/$ref  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json

{  
"@odata.id": "serviceRoot/People('vincentcalabrese')"  
}  
  
  
Response Payload

HTTP/1.1 204 No Content

#### Change a Link

A successful PUT request to a single-valued navigation property’s reference resource changes the related entity. The request below change the Airline of a Flight.  
PUT serviceRoot/People('russellwhyte')/Trips(1001)/PlanItems(11)/Microsoft.OData.SampleService.Models.TripPin.Flight/Airline/$ref  
  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json

{  
"@odata.id": "serviceRoot/Airlines('FM')"  
}  
  
  
Response Payload

HTTP/1.1 200 OK

{

"@odata.context":"serviceRoot/TripPinServiceRW/$metadata#$ref",

"@odata.id":"serviceRoot/People('russellwhyte')/Trips(1001)/PlanItems(11)/Microsoft.OData.SampleService.Models.TripPin.Flight/Airline"

}

### Functions and Actions

OData supports custom operations (Actions and Functions). Functions are operations exposed by an OData service that MUST return data and MUST have no observable side effects. Actions are operations exposed by an OData service that MAY have side effects when invoked. Functions and actions both MAY bound to an entity type, primitive type, complex type, or a collection.

#### Invoking Unbound Functions

The function below returns the nearest airport with the input geography point.  
GET serviceRoot/GetNearestAirport(lat = 33, lon = -118)  
  
Response Payload

Response 200 OK

{

"@odata.context": "serviceRoot/$metadata#Airports/$entity",

"IcaoCode": "KLAX",

"Name": "Los Angeles International Airport",

"IataCode": "LAX",

"Location": {

"Address": "1 World Way, Los Angeles, CA, 90045",

"City": {

"CountryRegion": "United States",

"Name": "Los Angeles",

"Region": "California"

},

"Loc": {

"type": "Point",

"coordinates": [

-118.408055555556,

33.9425

],

"crs": {

"type": "name",

"properties": {

"name": "EPSG:4326"

}

}

}

}

}

#### Invoking Bound Functions

The request below returns the favorite airline of a person, in TripPin service, "favorite airline" means airline which user choose most times. The function GetFavoriteAirline() is bound to Person.  
GET serviceRoot/People('russellwhyte')/Microsoft.OData.SampleService.Models.TripPin.GetFavoriteAirline()  
  
Response Payload

Response 200 OK

{

"@odata.context": "serviceRoot/$metadata#Airlines/$entity",

"@odata.id": "serviceRoot/Airlines('AA')",

"@odata.editLink": "serviceRoot/Airlines('AA')",

"AirlineCode": "AA",

"Name": "American Airlines"

}

#### Invoking Unbound Actions

TripPin currently has no scenario supported for unbound actions.

#### Invoking Bound Actions

The action below shares a trip to one of his friend. In TripPin service, by "share a trip" we mean that the owner and his friend now both have the trip and the trip share the same ShareId property.  
POST serviceRoot/People('russellwhyte')/Microsoft.OData.SampleService.Models.TripPin.ShareTrip  
  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json

{  
"userName": "scottketchum",  
"tripId": 1001  
}  
  
  
Response Payload

HTTP/1.1 204 No Content

### ETag

OData V4 supports ETag for Data Modification Request and Action Request. This section demonstrates how to operate on entity with ETag enabled. Please be noted that the ETag value below may be out-of-date, so when you try these requests, please first use the GET request to get the ETag of specified entity.

#### Update Entity with ETag

The request below shows how to update Person which has ETag supported. Please be noted that the value for If-Match should be the same with the ETag value of the specified Person.  
  
PATCH serviceRoot/People('clydeguess')  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json  
If-Match: W/"08D15F3DD9126D84"

{  
"@odata.type": "#Microsoft.OData.SampleService.Models.TripPin.Person",  
"FirstName" : "CLYDE"  
}  
  
  
Response Payload

HTTP/1.1 204 No Content

#### Delete Entity with ETag

The request below shows how to delete Person which has ETag supported. Please be noted that the value for If-Match should be the same with the ETag value of the specified Person.  
  
DELETE serviceRoot/People('vincentcalabrese')  
OData-Version: 4.0  
Content-Type: application/json;odata.metadata=minimal  
Accept: application/json  
If-Match: W/"08D15F3DD9A61539"  
  
  
Response Payload

HTTP/1.1 204 No Content