Vendor Client Setup for Paragon's OpenMLS RESO API

Introduction

We recommend reviewing any current RESO specifications on their GitHub site. This will give you an understand of how RESO API works, as it's a strict subset of Microsoft Web API. Any RESTful client can be used, but **we do not recommend** using services that store credential and data in third party storage. For example, Postman requires an account to save requests and collections.

Authentication Setup

Parameter/Setting	Value
Authorization Type	OAuth 2.0
Grant Type	Client Credentials with Basic Authentication
Access Token URL	https://MLSid.paragonrels.com/OData/MLSid/identity/connect/token
Client ID	Provided by Organization (MLS, Board, or Association)
Client Secret	Provided by Organization
Scope	OData

Time sensitive bearer tokens are issued through our identity server, does not support Auto-Refresh feature. The token in the response will look like this.

```
{"access_token":"token_string","expires_in":####,"token_type":"Bearer"}
```

Tokens should be sent in the `--header`. We do not support passing tokens passed in the URL.

Managing Credentials

Paragon OpenMLS credentials are managed by the Organization. Representatives of the Organization should be contacted regarding access to all resources in RESO RETS and API. Each feed (IDX, VOW, BBO etc.) will require its own feed and set of credentials.

Each Organizations may have different RESO Data Dictionary resources available. For the list of available resource, please contact your Organization.

Service Document

OData and RESO require a Service Document accessible from the ServiceRoot. Any metadata related to the data can be found there. In all examples SystemName and DataSet are interchangeable. Each organization or customer may have different DataSets they'll provide access to.

```
GET https://MLSid.paragonrels.com/Odata/MLSid/SystemName/$metadata
```

Per the specifications by the corresponding governing authorities, here are *some* of the pieces you'll find in the OpenMLS Service Document.

Entry	Use in OpenMLS
Edmx	Entity framework. Holds OData version number. Currently 4.0.
EntityContainer	Holds EntitySets or AssocationSets, and has a unique name
EntitySet	Holds $sort and $filter restrictions in records, and has unique EntityType
EntityType	Defines RESO define resources. Holds a key and property records for each field.
Type	Defines the Edm type
Nullable	True/False, could be null
MaxLength	Maximum length of the field
Annotation	Provides additional data, like LookupName value
Function	Allowable functions that can be used

Paging Through Results

By default, OpenMLS will return 25 records for each resource, and 250 for the Lookup resource. @odata.nextLink are included by default. It will show as long as the number of requested records is larger than the specified $skip. It will not be generated when $top is less than the `maxpagesize`. The requested set of data has been delivered, and there are no further pages of data.

It's not uncommon for vendors to need access to a lot more listings than what the `maxpagesize` is set to. This is set per resource by the Organization. The default is 1,000, 2,500 or 5,000 depending on the resource. OpenMLS does not have a separate replication endpoint. Please contact the Organization if you would like to specify a larger batch size to pull for replication.

No $top and no $skip:

```
"@odata.nextLink":"https://mlsid.paragonrels.com/OData/mlsid/DD1.7/Member?$s
elect=MemberKeyNumeric&$skip=25"
```

No $top and $skip=500:

```
"@odata.nextLink":"https://mlsid.paragonrels.com/OData/mlsid/DD1.7/Member?$s
elect=MemberKeyNumeric&$skip=525"
```

$top=150000 and $skip=160000 (`maxpagesize`=20k):

```
"@odata.nextLink":"https://mlsid.paragonrels.com/OData/mlsid/DD1.7/Property?
$select=ListingId&$top=130000&$skip=180000"
```

Processing Media

Like the other endpoints, Media also follows RESO's Data Dictionary standard. The media files are stored on our CDN, and we limit the number of connections per five minutes to 5,000. The HTTP/HTTPS HEAD requests don't work. Only use HTTPS/HTTPS GET requests on the CDN. Media on the CDN's should be pulled and stored locally.

At this time we do not support image URL's for Agent or Office, nor Documents. These will have to be obtained via RETS with GetObject Location=1.

Operators

Some additional operators available to pass with request.

Operator	Description
eq	Equal
ne	Not equal
gt	Greater than
lt	Less than
ge	Greater than or equal
le	Less than or equal
and	Logical and
or	Logical or
not	Logical not

Parameters

Some of the parameters or fields that may be passed to narrow your results.

Item	Type	Description
ListingKey(Numeric)	string/number	The listing key, available on the /Properties resource. Numeric fields are removed in DD 2.0.
MemberKey(Numeric)	string/number	The member key, available on the /Members resource. Numeric fields are removed in DD 2.0.
OfficeKey(Numeric)	string/number	The office key, available on the /Offices resource. Numeric fields are removed in DD 2.0.
$skip	number	Skips this number of results.
$select	string	Select the fields to be returned.
$unselect	string	Select the fields to be excluded.
$filter	string	Filter the results to be returned.
$top	number	Limits the size of the result set.
$orderby	string	Response field to sort query by (either "desc" or "asc")

Query Functionality

The following are queries to help search through data.

Function	Description
any	Search listings where all of the Heating is Electric: `/Property?$filter=Heating/any(a: a eq 'Electric')`
all	Search listings where all of the flooring is hardwood: `/Property?$filter=Flooring/all(a: a eq 'Hardwood')`
date	Search listings with a specific date: `/Property?$filter=date(ModificationTimestamp) eq 2017-08-29`
time	Search listings with a specific time: `/Property?$filter=time(ModificationTimestamp) eq 17:03:04`
year	Search listings with a specific year: `/Property?&$filter=year(ModificationTimestamp) eq 2017`
month	Search listings with a specific month: `/Property?$filter=month(ModificationTimestamp) eq 12`
day	Search listings with a specific day: `/Property?$filter=day(ModificationTimestamp) eq 23`
hour	Search listings with a specific hour: `/Property?$filter=hour(ModificationTimestamp) eq 17`
now()	Search listings within the current timestamp: `/Property?$filter=ModificationTimestamp eq now()`

Handling Time Zones

One special note needs to be made regarding time zones. RESO's Web Core API 2.0.0 specifications require precision of 27 to support ISO 8601 format. To simplify Organizations being in multiple time zones, our system is configured to only accept a value of 'Z' and not +/-00:00.
ie: 2021-05-21T06:28:34Z

Additional Support

Please visit our Vendor Support page for additional resources or contact methods.