MRI API Integrations Guide

Table of Contents

Calling RESTful APIs Created with APIDesign

APIs created with APIDesign follow the REST architecture style, which means that read and write operations are performed on the same URL. The root URL will be provided to you by your MRI system administrator. These APIs are accessible over standard Hypertext Transfer Protocol (HTTP).

The URL used to invoke an API over an HTTP request is shown below, where **[MRIWebDomain]** is the domain name of your MRI installation and **[APIName]** is the name of the API you want to invoke:

https://[MRIWebDomain]/MRIAPIServices/api.asp?$api=[APIName]

The following is a minimal request, using the GET HTTP verb to read data:

```
GET http://[MRIWebDomain]/mriapiservices/api.asp?$api=vendors
HTTP/1.1
Host: [MRIWebDomain]
Authorization:
```
Refer to the "Authorization" section on page 4.

Authorization

Every API call requires credentials, which must be provided in the **Authorization** header. The **Authorization** header combines the user name and password into a string; encodes that string using the RFC2045-MIME variant of Base64, with the exception of the 76 characters per line limitation; and then places **Basic** before the encoded string.

Example

Your client ID is **C1**, your Web Services user name is **U1**, your Web Services user password is **Password**, your MRI application database name is **D1**, and your assigned MIX partner key is **P1**.

With the above information, your API user name would be **C1/D1/U1/P1**, your password would be **Password**, and the **Authorization** header would be the following:

```
Authorization: Basic QzEvRDEvVTEvUDE6UGFzc3dvcmQ=
```

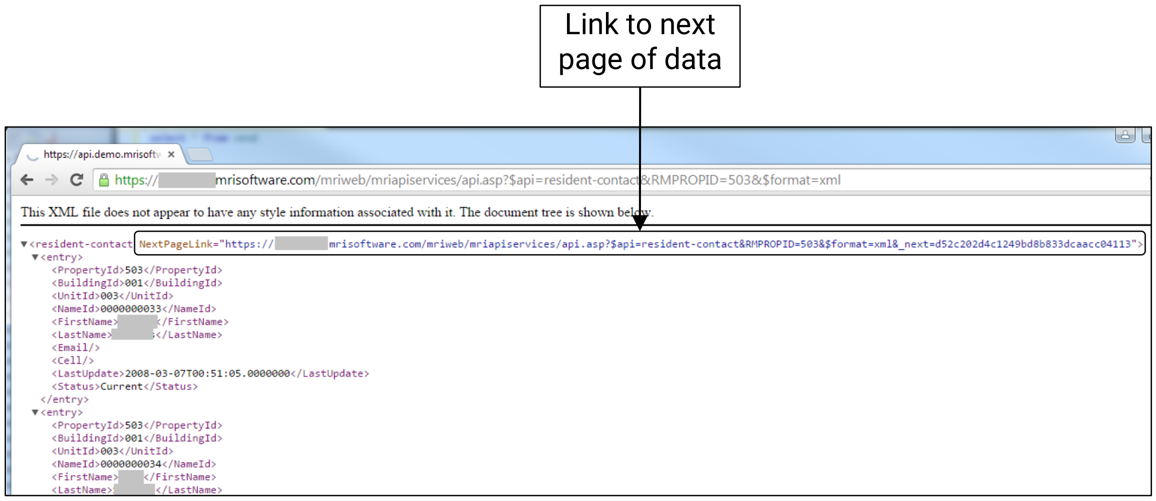
Reading Entries

To read data from an MRI API, use the HTTP GET verb.

For testing purposes, you can visit the API in any browser. You will be prompted for your credentials that you would normally provide with the **Authorization** header. For more information, refer to the “Authorization” section on page 4.

Paging

If the API call resulted in a very large set of data, results will be paged, and you will only get the first page of data. You will receive a **NextPageLink** URL at the top of each page of data (Figure 1), unless you are on the final page of data. Fetch the next page of data by invoking that URL in the same way that you invoked the original URL, providing the same headers.



https://[redacted]mrisoftware.com/mriweb/mriapiservices/api.asp?$api=resident-contact&RMPROPID=503&$format=xml

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<resident-contact NextPageLink="https://[redacted]mrisoftware.com/mriweb/mriapiservices/api.asp?$api=resident-contact&RMPROPID=503&$format=xml&_next=d52c202d4c1249bd8b833dcaacc04113">
  ▼<entry>
     <PropertyId>503</PropertyId>
     <BuildingId>001</BuildingId>
     <UnitId>003</UnitId>
     <NameId>0000000033</NameId>
     <FirstName>[redacted]</FirstName>
     <LastName>[redacted]</LastName>
     <Email/>
     <Cell/>
     <LastUpdate>2008-03-07T00:51:05.0000000</LastUpdate>
     <Status>Current</Status>
   </entry>
  ▼<entry>
     <PropertyId>503</PropertyId>
     <BuildingId>001</BuildingId>
     <UnitId>003</UnitId>
     <NameId>0000000034</NameId>
     <FirstName>[redacted]</FirstName>
     <LastName>[redacted]</LastName>
```

Figure 1. Results with Link to Next Page

Discovering the Structure of the API

You can learn about the structure of any API without invoking it by visiting its metadata endpoint. You can see the names and data types of all of the properties and relationships exposed by the API. This information can help you reliably build your side of the integration.

To see a summary of the structure of the data, go to the following URL:

http://example.com/mriapiservices/api.asp?$api=[API name]&**$metadata**

Note

Do not put an equal sign after **$metadata**.

Shaping Responses

When sending data into the API, use the **Content-Type** header to indicate the shape of the data that you are sending, and use the **Accept** header to indicate the shape of the data that you want returned.

You can specify that you want your response formatted as AtomPub, JSON, Plain XML, or CSV. If transformation is not specified, OData AtomPub feed is the default response.

AtomPub

You can request AtomPub in the following ways:

- Send an **Accept: application/atom+xml** header
- Append the **$format=atom** query string parameter

JSON

You can request JSON in the following ways:

- Send an **Accept: application/json** header
- Append the **$format=JSON** query string parameter

Plain XML

You can request plain XML in the following ways:

- Send an **Accept: application/xml** header
- Append the **$format=xml** query string parameter

CSV

You can request CSV in the following ways:

- Send an **Accept: text/csv** header

  Note

  If you want to omit CSV headers, send an **Accept: text/csv;header=absent** header.

- Append the **$format=csv** query string parameter

Creating New Entries

According to OData conventions, the POST HTTP verb is used to signify that the entry coming in must be created from scratch and cannot already exist in the database.

Your entry cannot have blank lines between headers, and there should be one blank line between headers and body text. You must provide a content type. For more information about content types, refer to the “Shaping Responses” section on page 6.

```
POST http://example.com/mriapiservices/api.asp?$api=myapi
      HTTP/1.1
Host: example.com
Authorization: Basic
      UDEyMzQ1Ni9BTUJfTVVFTlNURVIvYWJleW5lbnNvbi8xMjM0NTptcmk=
Content-Type: application/atom+xml;type=feed
Content-Length: 1063

<?xml version="1.0" encoding="utf-8"?>
<feed xml:base="http://example.com/mriapiservices/myapi/income-
      categories" xmlns="http://www.w3.org/2005/Atom"
      xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservi
      ces"
      xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservi
      ces/metadata" xmlns:georss="http://www.georss.org/georss"
      xmlns:gml="http://www.opengis.net/gml">
  <id>income-categories</id>
  <title />
  <updated>2014-03-03T01:59:52Z</updated>
  <entry>
    <category term="mri.income-categories" scheme=
      "http://schemas.microsoft.com/ado/2007/08/dataservices/sche
      me" />
    <id />
    <title />
    <updated>2014-03-03T01:59:52Z</updated>
    <author>
      <name />
    </author>
    <content type="application/xml">
      <m:properties>
        <d:IncomeCategory>942</d:IncomeCategory>
        <d:Description>nine 42</d:Description>
        <d:PriorityCode m:type="Edm.Decimal">13</d:PriorityCode>
        <d:BillingFrequency>F</d:BillingFrequency>
      </m:properties>
    </content>
  </entry>
</feed>
```

Merging with Existing Entries

According to OData conventions, the PUT HTTP verb is used to signify that the entry coming in is a patch and should be merged with existing data. If the entry does not exist, it will be created. This operation will always produce the same result.

Your entry cannot have blank lines between headers, and there should be one blank line between headers and body text. You must provide a content type.

```
PUT http://example.com/mriapiservices/api.asp?$api=income-
      categories HTTP/1.1
Host: example.com
Authorization: Basic
      UDEyMzQ1Ni9BTUJfTVVFTlNURVIvYWJleW5lbnNvbi8xMjM0NTptcmk=
Content-Type: application/atom+xml;type=feed
Content-Length: 1063

<?xml version="1.0" encoding="utf-8"?>
<feed
 xml:base="http://slnd073151045.mrisoftware.net/mriapiservices/my
 api/income-categories" xmlns="http://www.w3.org/2005/Atom"
 xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
 xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/m
 etadata" xmlns:georss="http://www.georss.org/georss"
 xmlns:gml="http://www.opengis.net/gml">
  <id>income-categories</id>
  <title />
  <updated>2014-03-03T01:59:52Z</updated>
  <entry>
    <category term="mri.income-categories"
      scheme="http://schemas.microsoft.com/ado/2007/08/dataservic
      es/scheme" />
    <id />
    <title />
    <updated>2014-03-03T01:59:52Z</updated>
    <author>
      <name />
    </author>
    <content type="application/xml">
      <m:properties>
        <d:IncomeCategory>942</d:IncomeCategory>
        <d:Description>nine 42</d:Description>
        <d:PriorityCode m:type="Edm.Decimal">13</d:PriorityCode>
        <d:BillingFrequency>F</d:BillingFrequency>
      </m:properties>
    </content>
  </entry>
</feed>
```

Note

If you are blocked from issuing a PUT or MERGE request by the web server, you can use the method override functionality by using a POST request and specifying the actual method in a header.

Example

```
POST http://example.com/mriapiservices/api.asp?$api=income-
      categories HTTP/1.1
Host: example.com
Authorization: Basic
      UDEyMzQ1Ni9BTUJfTVVFTlNURVIvYWJleW5lbnNvbi8xMjM0NTptcmk=
Content-Type: application/atom+xml;type=feed
Content-Length: 1063
X-HTTP-Method-Override: PUT

<?xml version="1.0" encoding="utf-8"?>
<feed
 xml:base="http://slnd073151045.mrisoftware.net/mriapiservices/my
 api/income-categories" xmlns="http://www.w3.org/2005/Atom"
 xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
 xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/m
 etadata" xmlns:georss="http://www.georss.org/georss"
 xmlns:gml="http://www.opengis.net/gml">
  <id>income-categories</id>
  <title />
  <updated>2014-03-03T01:59:52Z</updated>
  <entry>
    <category term="mri.income-categories"
      scheme="http://schemas.microsoft.com/ado/2007/08/dataservic
      es/scheme" />
    <id />
    <title />
    <updated>2014-03-03T01:59:52Z</updated>
    <author>
      <name />
    </author>
    <content type="application/xml">
      <m:properties>
        <d:IncomeCategory>942</d:IncomeCategory>
        <d:Description>nine 42</d:Description>
        <d:PriorityCode m:type="Edm.Decimal">13</d:PriorityCode>
        <d:BillingFrequency>F</d:BillingFrequency>
      </m:properties>
    </content>
  </entry>
</feed>
```

Error Reporting

You will receive all submitted entries after the POST and PUT requests have been saved. Entries with errors will include a **400 Bad Request** HTTP status as one of their properties.

Below are examples of how an entry with an error would look under different response-shaping options.

AtomPub

```
<entry>
  ...
  <content type="application/xml">
    <m:properties>
      <d:IncomeCategory>%RL</d:IncomeCategory>
      <d:Description>Percentage Rent in Lieu</d:Description>
      <d:Fasb13>N</d:Fasb13>
      <d:Error m:type="MRI.EntryError">
        <d:Message>BAD</d:Message>
      </d:Error>
    </m:properties>
  </content>
</entry>
```

JSON

```
{
  "IncomeCategory":"WTR",
  "Description":"Water",
  "Fasb13":"N",
  ...
  "Error":{
    "Message":"BAD"
  }
}
```

Plain XML

```
<entry>
  <Error>
    <Message>BAD</Message>
  </Error>
  <IncomeCategory>%KI</IncomeCategory>
  <Description>Kiosk % Rent</Description>
  <Fasb13>N</Fasb13>
</entry>
```

CSV

```
IncomeCategory,Description,Fasb13,ErrorMessage
KI,Kiosk % Rent,Y,N,
RL,Percentage Rent in Lieu,Y,N,
RN,Percentage Rent,Y,N,
ACR,Assoc Tax Assessment - U,Y,N,BAD
```

Troubleshooting

Before you begin calling APIs, make sure you have allowed outbound Internet access to MRI Home, as described in the “Allowing Access to the MRI Home Service” section in the *MRI Property Management System Administration Guide*.

HTTP Status Codes

When an API fails to execute, you may receive one of the errors listed below. The error response often includes a description of the problem in the response body. If no description is given, use Table 1 to find potential solutions.

Note

For a more extensive list of HTTP status codes and what they mean, refer to http://www.restapitutorial.com/httpstatuscodes.html.

Table 1. Error Codes and Solutions

Error	Solutions
400 Bad Request	Make sure you have indicated the shape of the data that you are sending in the **Content-Type** header. Make sure you are not missing any primary fields. Make sure you can call this API to save data.
401 Unauthorized	Make sure the user name in the request contains the appropriate client ID, database name, user name, and partner key, separated by forward slashes. **Example** User name: clientID/databasename/username/partnerkey In the **Authorization** header, make sure the type is **Basic** followed by the **username:password**. To see the user name and password that were provided, use a Base64 decoder. For more information, refer to the “Authorization” section on page 4. If the partner key is limited to certain clients, make sure the client is one of the authorized clients. If the partner key is limited to certain APIs, make sure the API being called is one of the authorized APIs, or that the key is a **DEVELOPER** key. Make sure the user that is calling the API has an API license and access to the API.

Error	Solutions
405 Method Not Allowed	Make sure you are using one of the following supported HTTP methods: ▪ GET ▪ POST ▪ PUT ▪ MERGE **Note** If you are blocked from issuing a PUT or MERGE request by the web server, you can use the method override functionality and route through POST, specifying the actual method in a header. For an example of this override, refer to the note in the "Merging with Existing Entries" section on page 8.
406 Not Acceptable	In the **Accept** or **Content-Type** headers, specify a valid format of **AtomPub**, **JSON**, **XML**, or **CSV**. For more information on specifying a valid format, refer to the "Shaping Responses" section on page 6.
500 Internal Server Error **"Unknown status while authorizing partner:"** **"Failed to authorize partner. Status:"** **"Failed to validate API. Status:"**	The manifest in the current system may have expired or failed to load. If you are an on-premise client, refer to the *MRI Property Management System Administration Guide* for how to refresh the manifest.
503 "Could not find more information with that token."	You may have waited too long before attempting to access the next page of continued data. You will need to reissue your original request.
503 Internal Timeout Exception	The API may be taking too long to return the requested information. Try to call the API again later. If the problem persists, contact the API developer or Client Support for troubleshooting assistance.

Support Case Recommendations

If you were unable to resolve your issue after reviewing the "Troubleshooting" section on page 11, create a Support case that includes the following information:

- The ID of the client with whom you are working
- The name of the database that you are attempting to reach
- Whether the environment is hosted on-premise or in SaaS
- If it is a SaaS environment, the version of MRI that is installed
- The name of the API that you are attempting to call
- An attachment containing the full XML or JSON request and response

Example

Support Case

Below is an example of the information to include when creating a case with MRI Client Support.

Subject: Cannot connect to client's database

Description: I'm trying to connect to client ID ABC3999 using the MRI_S-PMCM_OpenCharges API. The database name is PJB_TEST in SaaS version X.1. The full XML request and response is attached.

Attachment

Below is an example of the API request and response that should be included in a file and attached to the case. With the endpoint URL and information about how the call is being made, this file is essential for troubleshooting.

Caution!
Do **not** send Web Services credentials in regular or case emails. Credentials should only be sent in secure, encrypted emails.

Request

```
Using async GET
RequestUri:
'https://mrix1api.saas.mrisoftware.com/mriapiservices/api.asp?$api=MRI_
S-PMCM_OpenCharges&$format=xml', Version: 1.1, Headers:
{
  Authorization: Basic [encrypted text]
  Accept: application/xml
}
```

Response

```
{StatusCode: 401, ReasonPhrase: 'Unauthorized', Version: 1.1, Content:
System.Net.Http.StreamContent, Headers:
{
  X-UA-Compatible: IE=edge
  Cache-Control: no-store, no-cache, private
  Date: Wed, 27 Dec 2017 22:29:17 GMT
  Set-Cookie:
f5avrbbbbbbbbbbbbbbbb=IDNKCJHMFIFMELGCNNNFINFELANCHACHBDPEFBOBDOLKFEIPC
LAMECHDENPCDNHAOAMDBECLCOINEMMOLINALCGKHBIDBBDGNCNNMBDCBPCDAAOCKNKLFKBC
JOLPGJOE; HttpOnly; secure
  Server: Microsoft-IIS/8.5
  WWW-Authenticate: Basic realm="MRI Data Services. Unable to login as
web services user. Verify that the database exists in this
environment."
  X-Powered-By: ASP.NET
  Content-Length: 0
  Content-Type: text/html
  Expires: Wed, 01 May 1996 00:00:00 GMT
}
}
```