

# Mobile Researcher API Guide

## Overview

Mobile Researcher supports a read-only [RESTful](#) API (Application Programming Interface). The API leverages [Microsoft ADO .NET Data Services](#).

The goal of the ADO.NET Data Services framework is to facilitate the creation of flexible data services that are naturally integrated with the web. As such, ADO.NET Data Services use URIs to point to pieces of data and simple, well-known formats to represent that data, such as JSON and ATOM (XML-based feed format). This results in the data service being surfaced as a REST-style resource collection that is addressable with URIs and that agents can interact with using standard HTTP verbs such as GET, POST, PUT or DELETE. \*

The Mobile Researcher implementation of Microsoft ADO .NET Data Services can be located at the following URI: <https://www.researchconsole.com/secure/api.svc/>

## API Entities

To review entities accessible through the Mobile Researcher API you first need to log into your [research console](#) and then access the API URI specified above. The following steps outline the procedure to do this utilising Internet Explorer 7+. Please note that the steps include configuring your browser to disable feed reading view since this browser feature obfuscates the data returned by the API.

1. Launch Internet Explorer
2. Once launched, click the Tools menu option (if no menu is shown, click ALT+T), then select Internet Options.
3. Navigate to the Content tab
4. Under the "Feeds and Web Slices" section, click Settings
5. Un-check "Turn on feed reading view" (if you utilise this capability you can turn it back on after reviewing the API)
6. Click OK
7. Close and re-launch your browser
8. Navigate to [www.researchconsole.com](http://www.researchconsole.com) and login
9. Copy and paste this into your browser address bar <https://www.researchconsole.com/secure/api.svc/>

\* Mobile Researcher's implementation of the ADO .NET Data Services does not support POST, PUT and DELETE. The API is read-only.

After following these steps you should be able to review a list of entities returned by the API. The following entities should be returned:

- Devices
- Models
- Countries
- Manufacturers
- Surveys
- Studies
- Submissions
- Responses
- ResponseOptions
- Questions
- DeviceSurveyRelationships
- SubmissionsWithLocation

If you wish to review data returned for any particular entity, you would access the entity as follows: <https://www.researchconsole.com/secure/api.svc/{Entity}> – please note that {Entity} should be replaced with any particular entity you are interested in reviewing, also note that the API is case-sensitive. For example, if you wished to review all Question entities for your console, you would navigate to <https://www.researchconsole.com/secure/api.svc/Questions>.

Please be aware that when accessing records for a particular entity in the fashion described above a query to retrieve all data (i.e. no filtering is applied) will be produced. If you access the Response or Submission entities in this manner, all Responses/Submissions will be returned. This may cause your browser to freeze or crash as there are usually a large number of these entities accessible through your console. Appropriately, ADO .NET Data Services provides a rich query protocol which allows you to search and filter the data you are interested in to restrict the information downloaded.

For example, to review the first 10 Submissions for 2010, you could do so as follows:

[https://www.researchconsole.com/secure/api.svc/Submissions?\\$stop=10&\\$filter=year\(uploaddate\) eq 2010&\\$orderby=uploaddate](https://www.researchconsole.com/secure/api.svc/Submissions?$stop=10&$filter=year(uploaddate) eq 2010&$orderby=uploaddate)

If you wished to include the actual responses for each of the first 10 Submissions for 2010, you could do so as follows:

[https://www.researchconsole.com/secure/api.svc/Submissions?\\$stop=10&\\$filter=year\(uploaddate\) eq 2010&\\$orderby=uploaddate&\\$expand=Responses](https://www.researchconsole.com/secure/api.svc/Submissions?$stop=10&$filter=year(uploaddate) eq 2010&$orderby=uploaddate&$expand=Responses)

The default data serialisation format for data returned by the API is [ATOM](#) (XML-based feed format). When accessing the API programmatically, you may specify that the data be returned in [JSON](#) format. To change the default serialisation format to JSON, you would simply need to set the “Accept” HTTP header to “application/json” when accessing the API.

## Query String Options

Please find a summary of the available query string options below:

Option	Description	Example
expand	The 'expand' option allows you to embed one or more sets of related entities in the results. For example, if you want to display a Submission and its Responses, you could execute two requests, one for /Submissions(guid'42d4de6e-38b9-4b88-bcb1-df70227aff0a') and one for /Submissions(guid'42d4de6e-38b9-4b88-bcb1-df70227aff0a')/Responses. The 'expand' option on the other hand allows you to return the related entities in-line with the response of the parent in a single HTTP request. You may specify multiple navigation properties to expand by separating them with commas, and you may traverse more than one relationship by using a dot to jump to the next navigation property.	<p>A Submission with related Responses: /Submissions(guid'42d4de6e-38b9-4b88-bcb1-df70227aff0a')?\$expand=Responses.</p> <p>A Submission with related Responses and Response Options related to each Response: /Submissions(guid'42d4de6e-38b9-4b88-bcb1-df70227aff0a')?\$expand=Responses/ResponseOptions.</p> <p>A Submission with related Responses and Survey: Orders with related /Submissions(guid'42d4de6e-38b9-4b88-bcb1-df70227aff0a')?\$expand=Responses,Survey</p>
orderby	Sort the results by the criteria given in this value. Multiple properties can be indicated by separating them with a comma. The sort order can be controlled by using the "asc" (default) and "desc" modifiers.	<p>/Submissions?\$orderby=uploaddate</p> <p>/Submissions?\$orderby=uploaddate desc</p> <p>/Submissions?\$orderby=uploaddate,survey_id desc</p>
skip	Skip a given number of rows when returning results. This is useful in combination with "top" to implement paging (e.g. if using 10-entity pages, saying \$skip=30&top=\$10 would return the fourth page). NOTE: Skip only makes sense on sorted sets; if an orderby option is included, 'skip' will skip entities in the order given by that option. If no orderby option is given, 'skip' will sort the entities by their primary key and then perform the skip operation.	<p>Return all Submissions except the first 10: /Submissions?\$skip=10</p> <p>Return the 4th page where each page shows only 10 Submissions: /Submissions?\$skip=30&amp;\$top=10</p>
top	Restrict the maximum number of entities to be returned. This option is useful both by itself and in combination with skip, where it can be used to implement paging as discussed in the description of 'skip'.	<p>Top 10 Submissions: /Submissions?\$top=10</p> <p>Get the last 10 Submissions: /Submissions?\$orderby=uploaddate desc&amp;\$top=10</p>
filter	Restrict the entities returned from a query by applying the expression specified in this operator to the entity set identified by the last segment of the URI path.	<p>Get all Submissions for Survey with Id 101 : /Submissions?\$filter=survey_id eq 101</p> <p>Get all Questions where the question name contains "health": /Questions?\$filter=substringof('health', name)</p>

## Expression Syntax

The simple expression language that is used in filter operators (and also supported in orderby operations) supports references to columns and literals. The literal values can be strings enclosed in single quotes, numbers and boolean values (true or false) or any of the additional literal representations shown in the 'Data Type Literal Representations' section below. The operators in the expression language use abbreviations of the names rather than symbols to reduce the amount of escaping necessary in the URL.

### Logical Operators:

Operator	Description	Example
eq	Equal	/Questions?filter=name eq 'Health district'
ne	Not equal	/Questions?filter=name ne 'Health district'
gt	Greater than	/Submissions?\$filter=year(uploaddate) gt 2009
ge	Greater than or equal	/Submissions?\$filter=year(uploaddate) ge 2010
lt	Less than	/Submissions?\$filter=year(uploaddate) lt 2009
le	Less than or equal	/Submissions?\$filter=year(uploaddate) le 2010
and	Logical and	/Submissions?\$filter=year(uploaddate) le 2010 and survey_id = 101
or	Logical or	/Submissions?\$filter=year(uploaddate) eq 2008 or year(uploaddate) eq 2010
not	Logical negation	/Questions?\$filter=not endswith(name,'test.')

### Arithmetic Operators:

Operator	Description	Example
add	Addition	/Submissions?\$filter=hour(uploaddate) add 5 gt 10
sub	Subtraction	/Submissions?\$filter=hour(uploaddate) sub 5 gt 3
mul	Multiplication	/Submissions?\$filter=minute(uploaddate) mul 0.5 lt 3
div	Division	/Submissions?\$filter=minute(uploaddate) div 2 lt 3
mod	Modulo	/Submissions?\$filter=minute(uploaddate) mod 2 eq 3

### Grouping Operators:

Operator	Description	Example
()	Precedence grouping	/Responses?filter=(year(Submission/uploaddate) sub 5) lt 2005

*String Functions:*

String functions
bool substringof(string p0, string p1)
bool endswith(string p0, string p1)
bool startswith(string p0, string p1)
int length(string p0)
int indexof(string arg)
string insert(string p0, int pos, string p1)
string remove(string p0, int pos)
string remove(string p0, int pos, int length)
string replace(string p0, string find, string replace)
string substring(string p0, int pos)
string substring(string p0, int pos, int length)
string tolower(string p0)
string toupper(string p0)
string trim(string p0)
string concat(string p0, string p1)

*Date Functions:*

Date Functions
int day(DateTime p0)
int hour(DateTime p0)
int minute(DateTime p0)
int month(DateTime p0)
int second(DateTime p0)
int year(DateTime p0)

Please note that it is possible to apply expressions to related entities as per the following example:

[/Responses?\\$filter=year\(Submission/uploaddate\) gt 2010](#)

To perform a date query for a specific date please review the example below:

[/Submissions?\\$filter=survey\\_id eq 123 and uploaddate gt datetime'2009-12-10T14:00:00.000'](#)

This query above returns all submissions for survey Id '123' where the submission was received after 2PM 10th December 2009.

**Math functions**

double round(double p0)

decimal round(decimal p0)

double floor(double p0)

decimal floor(decimal p0)

double ceiling(double p0)

decimal ceiling(decimal p0)

## Authentication

The API should always be accessed via the SSL/HTTPS endpoint. This ensures that any data accessed through the API is encrypted. The API supports [basic HTTPAuthentication](#).

To access the API programmatically, you will need to set the “Authorization” HTTP header when executing the web request to the relevant API endpoint. Please note that the credentials used should be that of a valid user setup in your console. To access the Submission and Response data, the user should be setup with the “View responses” permission.

Please note that if the user account used to access the API is linked to more than one console, a custom HTTP header is required to be set. The custom HTTP header is “AccountID”. The value of this header should be set to the unique identifier for the account/console you are targeting. If you are unsure what your unique console/account ID is, please send an email requesting your account ID to support@mobileresearcher.com.

## Supported Data Formats

The API currently supports exchanging entities in JSON and Atom (an XML- based feed format). The mechanism used to specify in which format information is sent to a data service is the “Content-Type” HTTP header. In terms of specifying what format to receive data in, the "Accept" HTTP header should be set.

Requested Mime Type	Response Mime Type	Serialization Format
Grouping Media Types		
*/*	application/atom+xml	AtomPub
text/*	Not supported	N/A
application/*	Not supported	N/A
Individual Media Types		
text/xml	text/xml	AtomPub
application/xml	application/xml	AtomPub
application/atom+xml	application/atom+xml	AtomPub
application/json	application/json	JSON

## References

- Overview: Microsoft ADO .NET Data Services: <http://msdn.microsoft.com/en-us/library/cc956153.aspx>
- Using Microsoft ADO .NET Data Services : <http://msdn.microsoft.com/en-us/library/cc907912.aspx>