
Web API Specifications for PerfectMind

Table of Contents

Convention.....	2
Overview	3
HTTP Headers.....	3
Host	3
Methods.....	4
<i>Status</i>	<i>4</i>
<i>Organizations</i>	<i>4</i>
<i>Locations</i>	<i>4</i>
<i>Appointments</i>	<i>6</i>
<i>Objects</i>	<i>11</i>
<i>Fields.....</i>	<i>11</i>
<i>Query.....</i>	<i>12</i>
<i>Records</i>	<i>12</i>
<i>Login.....</i>	<i>15</i>
Specifications	15
<i>Data Types</i>	<i>15</i>
Glossary	16
<i>Terminology</i>	<i>16</i>
<i>HTTP Status Codes</i>	<i>16</i>
Event REST API.....	17
Steps to Demonstrate PerfectMind REST API.....	18
<i>Install Postman add-on for Chrome</i>	<i>18</i>
<i>Run Postman</i>	<i>18</i>
<i>Add Headers and Enter HTTP Command</i>	<i>18</i>
<i>Result</i>	<i>18</i>

Fetching Events Using REST API.....	19
<i>General Format</i>	<i>19</i>
<i>XML Response Format</i>	<i>21</i>
<i>Custom Filtering</i>	<i>22</i>
.....	24
Create/Update/Delete Records Via API	24
<i>Creating Records</i>	<i>24</i>
<i>Updating Existing Records.....</i>	<i>25</i>
<i>Deleting Existing Records</i>	<i>25</i>

Convention

- For each method, most of the possible responses to a REST call are listed under 'Responses'. Only one response is issued per request
- All responses are in JSON (exception: Get Status)
- All request parameters are mandatory unless explicitly marked [OPT]
- All requests must be made using https protocol

For example:

```
http://{host_address}

Here you need to replace {host_address} with a server address:

http://{host_address} = http://live.perfectmind.com
```

NOTE: In this document, placeholder text appears within curly braces {}. You must replace the placeholder text with your data.

For example:

```
...?CalendarName=My Calendar is not allowed.

Use ...?CalendarName=My%20Calendar instead.
```

All request parameters should be URL encoded, and JSON body (if the request body is in JSON format) needs to be JSON encoded.

Overview

This document describes the REST API and resources provided by PerfectMIND. REST APIs are for developers who want to integrate PerfectMIND into their application, and for administrators who want to script interactions with the PerfectMIND database.

HTTP Headers

These headers are required for all REST method calls below. Only an X-Access-Key or an X-Auth-Token is required. If an X-Auth-Token is provided in the request header there is no need for an X-Access-Key. To get a new X-Auth-Token the 'Login' method should be used. Requests with no valid X-Auth-Token can only access public data. All requests must use https protocol.

Type	Parameters	Data Type	Description
HEAD	X-Access-Key	string	A unique key assigned to API user
HEAD	X-Client-Number	string	Integer assigned by PerfectMIND uniquely identifying Client
HEAD	X-Auth-Token	string	OAuth Token acquired from PerfectMIND OAuth Server *
HEAD	X-Username	String	Perfectmind user username
HEAD	X-Password	String	Perfectmind user password

Note: PerfectMIND will provide an OAuth Server for Clients, to obtain authentication tokens in real time. In the interim, PerfectMIND provides tokens manually with expiration.

Host

Type	Host Address
Internal	newinternal.perfectmid.com
Alpha	alpha.perfectmind.com
Beta	beta.perfectmind.com
Live	[PortalSubdomain].perfectmind.com [PortalSubdomain] should be replaced with the value for the PortalSubdomain of the related organization. Portal subdomain value is included in the result of the GET Organizations request Example: myOrgSubdomain.perfectmind.com

Methods

Status

GET /api/2.0/Status

RESPONSE:

Return	Status	
	200	"Service running"
	503	"Service unavailable"

For Example:

GET <https://live.perfectmind.com/api/2.0/Status/200> HTTP/1.1

Organizations

Organizations acceptable by Web API Client

GET /api/2.0/Organizations

For Example:

GET <https://live.perfectmind.com/api/2.0/Organizations/> HTTP/1.1

RESPONSE:

Status	
400	"Access is not granted"
200	{ "AvailableOrganizations": [{ "Number":21546, "PortalSubdomain":"mobileapptest" }] }

An Organization may have multiple locations

GET /api/2.0/Organizations/{orgNum}/Locations

Locations

If a location's TimeZone 'null' it is the same as the Organization's time zone.

Returns all location information for specified Organization.

Type	Parameters	Values	Required
URL_PATH	orgNum	Integer	Yes

For Example:

GET https://live.perfectmind.com/api/2.0/Organizations/000000/Locations HTTP/1.1

RESPONSE:

Status	
200	<pre>{ "Organization" : { "ID" : "00x00000-xxox-0x00-00x0-0xxx00xx000x", "Name" : "Community Center", "Phone" : "5554443333", "LogoURI" : "http://logovault.net/00x00x00xxxx00xxx0000.png", "Number" : 18054, "LegalName" : "Community Center", "Website" : "www.communitycenter.com", "Address" : { "Country" : "Canada", "City" : "Vancouver", "Street" : "100-2000 Main Street", "PostalCode" : "ala 2f2", "StateProv" : "British Columbia" }, "Locale" : "en-CA", "TimeZone" : { "DisplayName" : "(GMT-08:00) Pacific Time (US & Canada)", "DaylightName" : "Pacific Daylight Time", "StandardName" : "Pacific Standard Time" }, "Currency" : "\$", "Locations" : [{ "ID" : "0xx0x0x0-xxxx-0x00-x00x-x000000x0xx", "Name" : "Delhi", "Phone" : "13342", "Fax" : "5554443333", "Email" : "org@email.com", "Website" : "www.somewebsite.com", "Description" : "This is description", "Address" : { "Country" : "India", "CountryAlpha2" : "IN", "City" : "delhi", "Street" : "Main Street",</pre>

	<pre> "PostalCode" : "aba 3g3", "StateProv" : "Delhi" }, "Locale" : "en-029", "Currency" : "USD", "TimeZone" : null]] } } </pre>
404	Not Found

Appointments

Manage Contact bookings for an Organization

Returns available public Appointments for an Organization.

GET /api/2.0/Organizations/[orgNum]/Appointments

Type	Parameters	Values	Required
URL_PATH	orgNum	Integer	Yes
URL_PARAM	startTime	Datetime	Yes
URL_PARAM	endTime	Datetime	Yes
URL_PARAM	page	Integer, Default=0, 0 Indexed	No
URL_PARAM	pageSize	Integer, Default=10	No

For Example:

GET <https://live.perfectmind.com/api/2.0/Organizations/000000/Appointments?startTime=2017-06-02T17:30:00Z &endTime=2017-06-02T18:30:00Z> HTTP/1.1

```

X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx0000000
X-Username: username
X-Password: pssst123
Accept: application/json
Content-Type: application/json
[
  {
    "StartTime": "2017-02-25T08:30:00-08:00",
    "EndTime": "2017-02-25T10:30:00-08:00",
  }
]

```

RESPONSE:

Status	
200	Response is a list of all occurrences of a series: <pre> { "TotalCount" : 84, "TotalPages" : 84, </pre>

	<pre> "PageNumber" : 1, "Result" : [[{ "ID" : "0x0x00xx-x0x0-0000-x0x0-00000x00x0x0", "ObjectID" : "00xx000x-xx00-00x0-xx0x-0000x0x000x0", "StartTime" : "2017-06-02T17:30:00Z", "EndTime" : "2017-06-02T18:00:00Z", "Subject" : "My Event", "Type" : "Class", "Description" : "Add description here.", "Location" : { "Country" : "Canada", "City" : "North Vancouver", "Street" : "123 Main Street", "PostalCode" : "a2a 3b3", "StateProv" : "British Columbia" }, "InstructorName" : "Francis Brook", "Taxable" : false, "Price" : "0.00", "Capacity" : "25", "Remaining" : "24", "Program" : "", "LocationIDs" : "0x0xx000-00x0-0x00-0000-0xxx00000x00" }]] </pre> ID – ID of the event ObjectID – ID of the event Object
403	
400	<pre> { "Message" : "The request is invalid.", "MessageDetail" : "Message Detail will appear here" } </pre>
400	<pre> { "Message" : "Please ensure the difference between start and end is less than 31 days" } </pre>

Note: ‘Free’ and ‘onetime fee’ events with \$0 are not distinguished. Both have a \$0 price.

RESPONSE:

Status	
200	Response is a list of all occurrences of a series: <pre> { "TotalCount" : 84, "TotalPages" : 84, "PageNumber" : 1, "Result" : [[{ "ID" : "0x0x00xx-x0x0-0000-x0x0-00000x00x0x0", "ObjectID" : "00xx000x-xx00-00x0-xx0x-0000x0x000x0", "StartTime" : "2017-06-02T17:30:00Z", "EndTime" : "2017-06-02T18:00:00Z", </pre>

	<pre> "Subject" : "My Event", "Type" : "Class", "Description" : "", "Location" : { "Country" : "Canada", "City" : "Vancouver", "Street" : "123 Main Street", "PostalCode" : "a3b 3d4", "StateProv" : "British Columbia" }, "InstructorName" : "Francis Brook", "Taxable" : false, "Price" : "0.00", "Capacity" : "25", "Remaining" : "24", "Program" : "", "LocationIDs" : "0x0xx000-00x0-0x00-0000-0xxx00000x00" } } </pre>
	ID – ID of the event ObjectID – ID of the event Object
403	
400	<pre> { "Message" : "The request is invalid.", "MessageDetail" : "Message Detail will appear here" } </pre>

GET /api/2.0/Organizations/[orgNum]/Locations/[LocationIDs]/Appointments

Returns available public Appointments for an Organization and Location.

Type	Parameters	Values	Required
URL_PATH	orgNum	Integer	Yes
URL_PATH	LocationIDs	Guid	Yes
URL_PARAM	startTime	Datetime	Yes
URL_PARAM	endTime	Datetime	Yes
URL_PARAM	page	Integer, Default=0, 0 Indexed	No
URL_PARAM	pageSize	Integer, Default=10	No

For Example:

POST <https://live.perfectmind.com/api/2.0/Organizations/000000/Locations/x000x000-x000-0000-x000-0x00x000xxxx/Appointments?startTime=2017-06-02T17:30:00Z&endTime=2017-06-02T18:30:00Z>
HTTP/1.1

X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx0000000
X-Username: username
X-Password: pssst123
Accept: application/json
Content-Type: application/json

```
[
{
"CalendarSetupId": "00000000-x00x-000x-x00x-x00x0x00xx00",
"StartTime": "2017-02-25T08:30:00-08:00",
"EndTime": "2017-02-25T10:30:00-08:00",
"Subject": "Created via API",
"LocationId": "x000x000-x000-0000-x000-0x00x000xxxx"
}
]
```

POST /api/2.0/Organizations/[orgNum]/Appointments

Book Contact to a single class for an Organization. If a user has already been created, the contactID is required and there is no need for firstName, lastName, and email. If the contactID is not provided then firstName, lastName, and email are required.

Type	Parameters	Values	Required
URL_PATH	orgNum	Integer	Yes
BODY	eventide	Guid	Yes
BODY	eventObjectID	Guid	Yes
BODY	startTime	Datetime	Yes
BODY	firstName	String	Yes
BODY	lastName	String	Yes
BODY	Email	String	Yes
BODY	ContactID	Guid	Yes

REQUEST BODY:

```
Content-Type: application/json; charset=UTF-8
{
  "eventObjectID" : "xx00000x-xxx0-x000-00x0-x0x0x0xx0xxx",
  "startTime" : "2017-01-01T17:30:00Z ",
  "firstName" : "Joe",
  "lastName" : "Smith",
  "email" : JoeSmith@test.com,
  "contactId": "00x0000x-0x0x-00x0-0000-0000x00x0xx"
}
```

For Example:

POST

<https://live.perfectmind.com/api/2.0/Organizations/000000/Appointments?eventObjectID=xx00000x-xxx0-x000-00x0-x0x0x0xx0xxx&startTime=2017-06-02T17:30:00Z&endTime=2017-06-02T18:30:00Z&firstName=Joe&lastName=Smith&Email=JoeSmith@test.com&ContactID=00x0000x-0x0x-00x0-0000-0000x00x0xx> HTTP/1.1

RESPONSE:

Status	
200	{ "EventID" : "xx00000x-xxx0-x000-00x0-x0x0x0xx0xxx", "ContactID" : "0xx000x0-0x00-0000-000x-xx0000x0000x" }

NOTE: When a Contact is created for an Organization, it is linked to an external B2B Partner contactID. If email, last name, or first name, are at the external B2B Partner's Customer account, the two accounts will be disconnected unless a contactID is provided.

IE: If email and/or last name and/or first name are updated with an external B2B Partner Customer account; new booking generates new Contact if contactID is not provided.

NOTE: The EventID provided in the result is required for cancelling the appointment. Booking the same service for the same contact multiple times will not override the existing one. If there is an existing booking the system will not create a new one and will return the 'eventID' and 'contactID' in response. This will enable you to check the status of the booking with the same request. If the response code is 'ok' then the booking still exists, otherwise, depending on the response, the booking might have been canceled, or the class was removed, or other possible scenarios.

DELETE /api.2.0/Organizations/[orgNum]/Appointments

Cancels Appointment for a Contact made with an Organization

Type	Parameters	Values	Required
URL_PATH	orgNum	Integer	Yes
BODY	eventID	Guid	Yes
BODY	contactID	Guid	Yes
BODY	eventObjectID	Guid	Yes

For Example:

DELETE <https://live.perfectmind.com/api/2.0/Organizations/000000/Appointments?eventID=xx00000x-xxx0-x000-00x0-x0x0x0xx0xxx&contactID=0xx000x0-0x00-0000-000x-xx0000x0000x&eventObjectID=xx00000x-xxx0-x000-00x0-x0x0x0xx0xxx> HTTP/1.1

RESPONSE:

Status	
200	OK
400	Bad request

Objects

Will provide name and ID of custom objects.

GET /api/2.0/B2C/Objects

For Example:

GET [https://live.perfectmind.com/api/2.0/B2C/Objects?TestObject=000x00x0-x0x0-0xxx-0x0x-0xx00x00x00&Setup_Hidden=00000000-0000-0000-0000-000000000000&CMSPageObject=0000000000000000&Facility=00000000-0000-0000-xxxx-0000x0x0000x](https://live.perfectmind.com/api/2.0/B2C/Objects?TestObject=000x00x0-x0x0-0xxx-0x0x-0xx00x00x00&Setup_Hidden=00000000-0000-0000-0000-000000000000&CMSPageObject=0000000000000000&Facility=00000000-0000-0000-xxxx-0000x0x0000x HTTP/1.1) HTTP/1.1

RESPONSE:

Status	
200	{ "TestObject": "000x00x0-x0x0-0xxx-0x0x-0xx00x00x00", "Setup_Hidden": "00000000-0000-0000-0000-000000000000", "CMSPageObject": "0000000000000000": "x000xx00-0000-0x0x-xx00", "Facility": "00000000-0000-0000-xxxx-0000x0x0000x" }
400	"Access is not granted"

Fields

GET /api/2.0/B2C/Fields

Provide metadata of object fields.

Type	Parameters	Values	Required
URL_PARAM	id	Guid	Yes

For Example:

GET [https://live.perfectmind.com/api/2.0/B2C/Fields?id=xx000xxxx](https://live.perfectmind.com/api/2.0/B2C/Fields?id=xx000xxxx HTTP/1.1) HTTP/1.1

RESPONSE

Status	
200	Object metadata in json
404	Not Found

Query

POST /api/2.0/B2C/Query

Will execute the sql statement and return the result. Only *select* statements are allowed.

Type	Parameters	Values	Required
HEAD	Content-Type	application/json	Yes
BODY	QueryString	String	Yes

For Example:

POST <https://live.perfectmind.com/api/2.0/B2C/Query?Content-Type=application&QueryString=HTTP/1.1>

RESPONSE

Status	
200	Query result
404	Not Found

Records

GET /api/2.0/B2C/ObjectRecords

Get records of a specific object.

Type	Parameters	Values	Required
URL_PATH	tableName	String	Yes

For Example:

GET <https://live.perfectmind.com/api/2.0/B2C/ObjectRecords?tableName=CalendarEvent> HTTP/1.1

RESPONSE:

Status	
200	Response is a list of all records of a custom table/object
403	
400	{ "Message" : "The request is invalid.", "MessageDetail" : "Message Detail will appear here" }

The 'runWorkFlow' input parameter is used to determine whether running the workflows is required. For example, in the case of adding records, if its value is true all workflows triggered by adding a new record of the specified object/table will be executed. The default is 'true'.

POST /api/2.0/B2C/AddRecord

Add new record to a custom table.

'runWorkFlow' is used to determine if running workflows is required or not. In this case if true all workflows triggered by adding new record of the specified object/table will be executed

Type	Parameters	Values	Required
URL_PARAM	tableName	string	Yes
HEAD	Content-Type	application/json	Yes
URL_PARAM	runWorkFlow	True/False: default is true	No
BODY	inFields	Json: field values, keys should be 'codename' of field	Yes

For Example:

POST <https://live.perfectmind.com/api/2.0/B2C/UpdateRecord?tableName=CalendarEvent&Content-Type=application&runWorkFlow=true &inFields=codename> HTTP/1.1

```
X-Access-Key: xxxxxxxxxxxx^*&^ (*&xxxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx0000000
X-Username: username
X-Password: pssst123
Accept: application/json
Content-Type: application/json
[
  {
    "CalendarSetupId": "00000000-x00x-000x-x00x-x00x0x00xx00",
    "StartTime": "2017-02-25T08:30:00-08:00",
    "EndTime": "2017-02-25T10:30:00-08:00",
    "Subject": "Created via API",
    "LocationId": "x000x000-x000-0000-x000-0x00x000xxxx"
  }
]
```

RESPONSE:

Status	
200	New record has been added
403	"Unable to add the new record"
400	"The received data is not valid"

Type	Parameters	Values	Required
URL_PARAM	tableName	String :table name	Yes
HEAD	Content-Type	application/json	Yes
URL_PARAM	id	Guid :record id	Yes
URL_PARAM	runWorkFlow	True/False: default is true	No
BODY	inFields	Json : fields to be updated, keys are field 'codename'	Yes

PUT /api/2.0/B2C/UpdateRecord

Update an existing record of a custom object.

For Example:

PUT <https://live.perfectmind.com/api/2.0/B2C/UpdateRecord?tableName=CalendarEvent&Content-Type=application&runWorkFlow=true &inFields=codename> HTTP/1.1

```
X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxx000xxxxxx0000xxxxxxxxxxxxxxxxxxxx0000000
X-Username: username
X-Password: pssst123
Accept: application/json
Content-Type: application/json
[
  {
    "CalendarSetupId": "00000000-x00x-000x-x00x-x00x0x00xx00",
    "StartTime": "2017-02-25T08:30:00-08:00",
    "EndTime": "2017-02-25T10:30:00-08:00",
    "Subject": "Created via API",
    "LocationId": "x000x000-x000-0000-x000-0x00x000xxxx"
  }
]
```

RESPONSE:

Status	
200	Record has been modified
400	"Unable to update the record"
400	"The received data is not valid"

DELETE /api/2.0/B2C/DeleteRecord

Delete a custom object record.

Type	Parameters	Values	Required
URL_PARAM	tableName	String :table name	Yes
URL_PARAM	id	Guid :record id	Yes
URL_PARAM	runWorkFlow	True/False: default is true	No

Status	
200	Record has been deleted
400	"Unable to delete the Item."
400	"The received data is not valid"

For Example:

DELETE

<https://live.perfectmind.com/api/2.0/B2C/DeleteRecord?tableName=CalendarEvent&ids=x0x0xx00-0000-0000-x000-000000xxxx00&runWorkFlow=True> HTTP/1.1

Login

This method is used to authenticate a user and create a session on the server. If the user authentication is successful a Session ID and Authentication Token cookie will be returned. Requests to other methods can use the session ID and Auth Token is passed in as a cookie, like a web browser.

GET or POST /api/2.0/B2C/Login

Type	Parameters	Values	Required
URL_PARAM	orgId	Guid	Yes
URL_PARAM	username	string	Yes
URL_PARAM	password	string	Yes

For Example: GET or POST <https://live.perfectmind.com/api/2.0/B2C/Login?orgID=&username=JoeSmith&password=pssst123> HTTP/1.1

RESPONSE:

Status	
200	{ "Success" : true, "ErrorMessage" : null, "UserID" : "00xx00x0-x0xx-0x0x-x000-0000x00x000x", "IsMemberSession" : true }
200	{ "Success" : false, "ErrorMessage" : "Invalid username or password", "UserID" : null, "IsMemberSession" : false }
404	Missing required parameters

Specifications

Data Types

Type	Format
Datetime	YYYY-MM-DDThh:mm:ss
Date	YYYY-MM-DD (UTC)
guid	32 hexadecimal digits separated by hyphens (D) : 00000000-0000-0000-0000-000000000000

Glossary

Terminology

Client	An external Company integrating with PerfectMIND for B2B communications. Organizations accessible to a Client are managed by PerfectMIND according to business agreements with them.
Client Number	An external, visible Identifier assigned to a Client by PerfectMIND.
Client ID	An internal identifier (guid) assigned to a Partner by PerfectMIND.
Organization	A PerfectMIND Customer.
orgNum	An integer identifier assigned to an Organization by PerfectMIND for use by external Clients.
Location	A PerfectMIND Organization (Customer) may have multiple site locations.
LocationIDs	An internal identifier (guid) assigned to a Location.
ContactID	A guid identifier uniquely identifying a Customer for an Organization. Obtain via Signup request.
Event	A class a Contact can register in and attend.
Series	Events can re-occur as part of a series.
Exception	An occurrence in the Series which is somehow different from the other Events in the Series.
EventID	The ID of a single event or exception, or a series, which contains the event or exception. eventID is included in the result of Appointments requests. (Id)
EventObjectID	The ID of object to which an event belongs. eventObjectID is included in result of Appointments request (ObjectID).
StartTime	Regarding the booking of a contact. This is the start time for a single class of a series of classes. startTime is included in the result of Appointments request.
HEAD	REST interface argument passed via HTTP HEADER.
BODY	REST interface argument passed via message body.
URL_PATH	REST interface argument passed via {id} segment within URL path.
URL_PARAM	REST interface argument passed via URI encoding.
BODY	REST interface argument passed via HTTP BODY.
X-Access-Key	Client OAuth token sent with all Client requests, used to authenticate sender.
X-Client-Number	Identifiable unique integer assigned to external Clients by PerfectMIND and sent with all Client requests.
Status	HTTP status code of response.

HTTP Status Codes

All status codes are standard HTTP status codes. Below are codes used in this API:

- [2XX](#) Success of some kind
- [4XX](#) Error occurred in Client
- [5XX](#) Error occurred in Server

Status Code	Description
200	OK
201	Created
202	Accepted (Request accepted and queued for execution)
400	Bad Request
401	Authentication failure
403	Forbidden
404	Resource not found
405	Method Not Allowed
409	Conflict
412	Precondition Failed
413	Request Entity Too Large
500	Internal Server Error
501	Not Implemented
503	Service Unavailable

Event REST API

This API is useful for retrieving event data for screen display systems throughout your organization, using a framework such as Scala or Capital Networks.

Before proceeding with API documentation, it is recommended that you get acquainted with HTTP and JSON standards, to understand how requests to API endpoint are formed. We can recommend these sources:

- <http://www.json.org/index.html>
- <https://developer.mozilla.org/en-US/docs/Web/HTTP>.
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>

If you need to test it outside of your code you can use tools like Fiddler, Advanced Rest Client, Curl, Postman, etc.

All requests are required to have an X-Access-Key header and one of the following: either an X-Client-Number or an X-Username and X-Password. Ask admin to provide you with an access key. The X-Client-Number is the client database number.

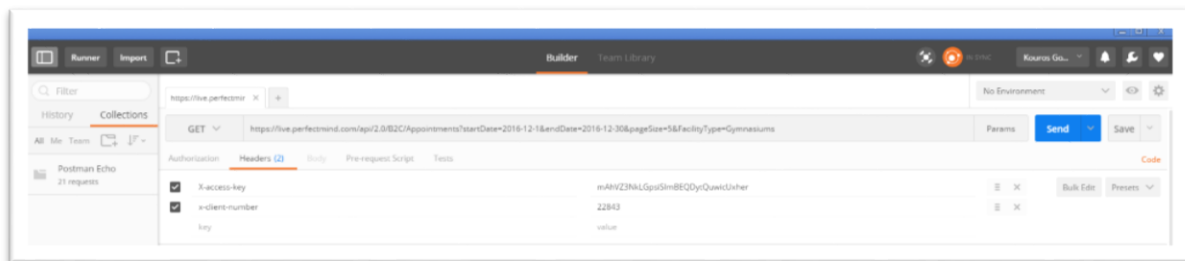
Steps to Demonstrate PerfectMind REST API

Install Postman add-on for Chrome

Get Postman from this [link](#) and add it to the Chrome Add-ons.

Run Postman

Launch the Postman App by going to this [link](#) and clicking the Launch App button.



Add Headers and Enter HTTP Command

Under the Headers section add the following keys:

Type	Parameters	Data Type	Description
HEAD	X-Access-Key	string	A unique key assigned to API user
HEAD	X-Client-Number	string	Integer assigned by PerfectMIND uniquely identifying Client
HEAD	X-Auth-Token	string	OAuth Token acquired from PerfectMIND OAuth Server *
HEAD	X-Username	String	Perfectmind user username
HEAD	X-Password	String	Perfectmind user password

Make sure the main command is “GET” and the URL is as follows:

<https://live.perfectmind.com/api/2.0/B2C/Appointments?StartDate=2016-12-1&EndDate=2016-12-30&pageSize=5&FacilityType=Gymnasiums>

After clicking Send, the body of the result will get populated with the list of all returned values.

Result

The result is paginated and the API documentation provides more examples for values returned. The result filters applicable as well as some sample code.

You can choose JSON or XML. This sample excerpt shows JSON:

```
{
  "TotalCount": 22653,
  "TotalPages": 4531,
  "PageNumber": 0,
  "Result": [
    {
      "ID": "00000xx0-0000-0x00-xx00-x00xxxx00x00",
      "StartTime": "2017-12-01T00:00:00-07:00",
```

```
"EndTime": "2017-12-01T00:00:00-07:00",
"Subject": "PerfectMind",
"Description": "",
"Location": {
  "City": "VancouverHighlands Ranch",
  "Street": "1234 Main Street ",
  ...
}
```

Fetching Events Using REST API

General Format

<https://{host}/api/2.0/B2C/Appointments?{parameter name}={parameter value}&{parameter name}={parameter value}...>

List of possible query parameters:

Name	URL name	Required	Type/Expected value
Search range start date	StartDate	Y	ISO 8601 date string in yyyy-dd-MM format
Search range end date	EndDate	Y	ISO 8601 date string in yyyy-dd-MM format
Calendar category name	CalendarCategory	N	string
Calendar name	CalendarName	N	string
Service name	ServiceName	N	string
Calendar type	CalendarType	N	One of the following strings: Appointments; SingleBooking; Courses; FacilityBooking
Instructor full name	Instructor	N	string
Facility name	Facility	N	string
Show to	ShowTo	N	One of the following strings: Staff; Members; Public
Location name	Location	N	String
Additional fields to return	AdditionalFields	N	List of strings, i.e. AdditionalFields=1&AdditionalFields=2....
List of Custom fields	AdditionalFilters	N	
List of Facility IDs to filter by	FacilityIDs	N	List of guids (FacilityIDs=....&FacilityIDs=...)
List of Location IDs to filter by	LocationIDs	N	List of guids (LocationIDs=....&LocationIDs=...)

To use PerfectMind Events REST Event API you need to make a request to API endpoint:

Sample Requests:

GET <https://{host}/api/2.0/B2C/Appointments?StartDate={StartTime}&EndDate={endTime}>

X-Access-Key: asldfjkahfjh^*&^(*&afjha987qvakfhq3489crjfhLKUWDCFNUIAFY9034340

X-Username: {login}

X-Password: {pass}

EndDate must be later than StartDate. The difference between them must be < 31 days

Response:

```
{
  "TotalCount": 107,
  "TotalPages": 2,
  "PageNumber": 0,
  "Result": [
    {
      "ID": "0x00000x-000x-0x0x-x0x0-000x0000000x",
      "StartTime": "2017-01-30T08:30:00-08:00",
      "EndTime": "2017-01-30T09:30:00-08:00",
      "Subject": "Sizzling Seniors",
      "Description": "REGISTER IN CLASS: This dynamic, non-impact, aerobic class is for age 50 Plus. It encourages the active senior to develop cardio & respiratory conditioning with endurance, along with muscular balance.",
      "Location": {
        "City": "Metroville",
        "Street": "Elm St.",
        "PostalCode": "90001",
        "Country": "United States",
        "StateProv": "California"
      },
      "InstructorName": "John Smithe",
      "Capacity": null,
      "Remaining": null,
      "Program": null,
      "LocationIDs": "x000x000-x000-0000-x000-0x00x000xxxx",
      "CalendarCategory": "55 Plus",
      "CalendarName": "55 Plus Calendar",
      "CalendarType": "SingleBooking",
      "ShowTo": "Public",
      "OnlineDisplayDate": "2017-12-07T00:00:00-08:00",
      "RegistrationStartDate": null,
      "RegistrationEndDate": null,
      "Image": "",
      "FacilityID": "xx0xx000-0000-0x0x-x0xx-xxx000x00000",
      "Facility": "Youth Rec Center - Room B",
      "Session": null,
      "AdditionalFacilities": null,
      "RequestedFields": {
        "Booked": 2,
        "Details": "REGISTER IN CLASS: This dynamic, non-impact, on-going aerobic class is for age 50 Plus. It encourages the active senior to develop cardiovascular/ respiratory conditioning with endurance, along with muscular balance, strength, flexibility and coordination. ",
      }
    }
  ]
}
```

Example:

GET <https://live.perfectmind.com:443/api/2.0/B2C/Appointments?location=00000x00-0x0x-00x0-x000-00xx000x0x0&StartDate=2017-01-30&EndDate=2017-01-31&AdditionalFields=Details&AdditionalFields=Booked> HTTP/1.1

X-Access-Key: 0xx0xxxxxxxxxx0xxxxxxxxxxxxx
X-Client-Number: 00000

As you can see, the request returned fields such as "TotalCount": 107, "TotalPages": 2, "PageNumber": 0. To change the page, pass the "page" request parameter. To change the number of records on a page, pass "pageSize" parameter.

The request also returned general information regarding the number of records in the resulting set, and an array with actual data. The array consists of a predefined set of fields and may include the "RequestedFields" array with all the data for the fields that you included in the "AdditionalFields" parameter of the request.

For example:

GET [https://live.perfectmind.com/api/2.0/B2C/Appointments?StartDate=2017-01-30&EndDate=2017-01-31&AdditionalFilters\[0\].Field=FacilityID&AdditionalFilters\[0\].Operator=NotNull&AdditionalFilters\[1\].Field=InstructorId&AdditionalFilters\[1\].Operator=IsNull](https://live.perfectmind.com/api/2.0/B2C/Appointments?StartDate=2017-01-30&EndDate=2017-01-31&AdditionalFilters[0].Field=FacilityID&AdditionalFilters[0].Operator=NotNull&AdditionalFilters[1].Field=InstructorId&AdditionalFilters[1].Operator=IsNull) HTTP/1.1

X-Access-Key: xxxxxxxxxxxx^*&^ (*&xxxxxx000xxxxxxxx0000xxxxxxLKUWDCFNUIAFY9000000
X-Username: {login}
X-Password: {pass}
Accept: application/xml

Change the response format to xml by adding Accept: application/xml request header.

XML Response Format

To receive results in XML format, add the following header:

GET <https://{host}/api/2.0/B2C/Appointments?StartDate={startTime}&EndDate={endTime}>

Example:

GET <https://live.perfectmind.com:443/api/2.0/B2C/Appointments?StartDate=2017-01-30&EndDate=2017-01-31> HTTP/1.1

Custom Filtering

To fetch events with empty FacilityIDs, add a custom filter to the query:

```
GET
https://{host}/api/2.0/B2C/Appointments?StartDate={startTime}&EndDate={endTime}&AdditionalFilters[0].Field=FacilityIDs&AdditionalFilters[0].Operator=IsNull
```

For example:

```
X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx00000000
X-Username: {login}
X-Password: {pass}
```

You may add any filter. For example, fetch only events which are not empty.

Use FacilityIDs by changing AdditionalFilters[0].Operator=IsNull to AdditionalFilters[0].Operator=NotNull. You can fetch events that have specific FacilityIDs by changing AdditionalFilters[0].Operator=IsNull&AdditionalFields=FacilityIDs to AdditionalFilters[0].Operator=EqualTo&AdditionalFields=FacilityIDs&AdditionalFilters[0].Value={value}.

Add any filter by adding a new one with a different filter index (i.e. AdditionalFilters[1] instead of AdditionalFilters[0]).

To filter events by multiple facility IDs or location IDs, add the following filters to the query:

- 1) Locations: LocationIDs=....&LocationIDs=
- 2) Facilities: FacilityIDs=....&FacilityIDs=...

Your request will look like this:

```
GET
https://{host}/api/2.0/B2C/Appointments?StartDate={startTime}&EndDate={endTime}&AdditionalFilters[0].Field=TeamsidelineID&AdditionalFilters[0].Operator=IsNull&LocationIDs={LocationIDs1}&LocationIDs={LocationIDs2}&FacilityIDs={FacilityIDs1}&FacilityIDs={FacilityIDs2}

X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx00000000
X-Username: {login}
X-Password: {pass}
```

You can filter a resulting set by using the custom filters feature, such as the "AdditionalFilters" query parameter.

{usual set of headers}

You can add multiple filters at once. They all will be combined by AND operator (so only records satisfying all conditions will be returned).

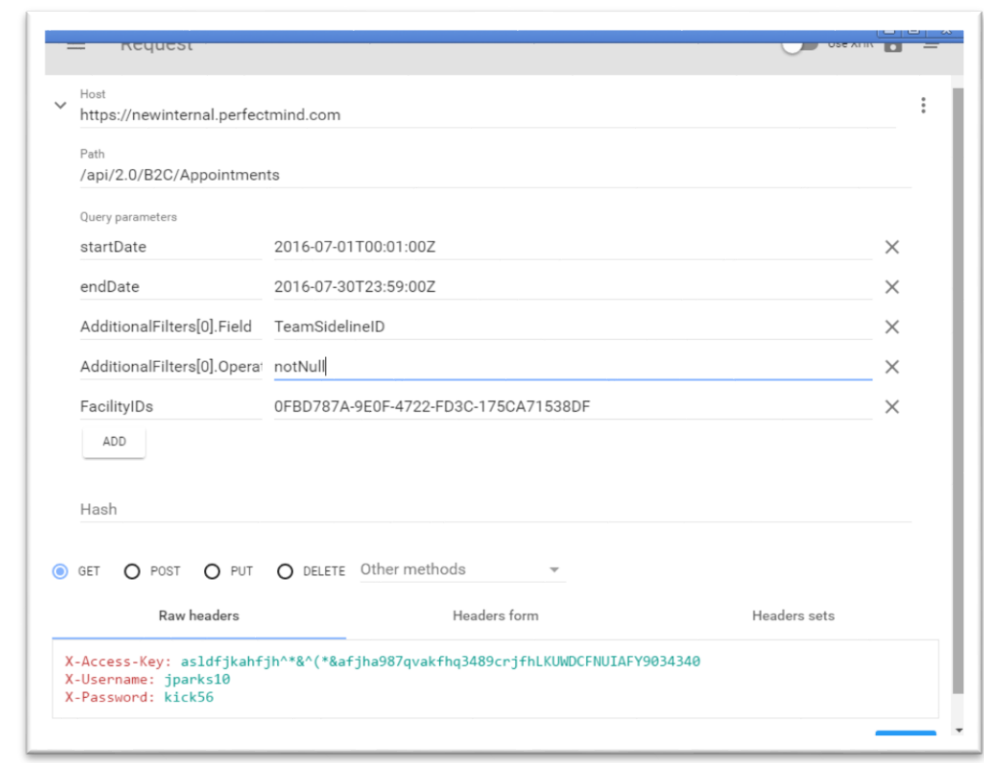
If you need to compare field value with exact value you can use AdditionalFilters[{filter index}].Value.

{usual set of headers}

List of possible filter operators:

EqualTo
NotEqualTo
GreaterThan
LessThan
GreaterThanOrEqualTo
LessThanOrEqualTo
StartsWith
EndsWith
IsNull
NotNull

Sample request using Chrome extension, called Advanced REST client:



For example:

GET [https://live.perfectmind.com/api/2.0/B2C/Appointments?StartDate=2017-01-30&EndDate=2017-01-31&AdditionalFilters\[0\].Field=FacilityID&AdditionalFilters\[0\].Operator=EqualTo&AdditionalFilters\[0\].Value=332BE2B3-3FF3-4BD9-9FDC-7F60293A4D87](https://live.perfectmind.com/api/2.0/B2C/Appointments?StartDate=2017-01-30&EndDate=2017-01-31&AdditionalFilters[0].Field=FacilityID&AdditionalFilters[0].Operator=EqualTo&AdditionalFilters[0].Value=332BE2B3-3FF3-4BD9-9FDC-7F60293A4D87) HTTP/1.1

For Example:

POST <https://live.perfectmind.com/api/2.0/B2C/Record?tableName=Amenities> HTTP/1.1

X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxx000xxxxxxxx0000xxxxxxxxLKUWDCFNUIAFY0000000

X-Username: OrgName

X-Password: pssst123

Content-Type: application/json

```
[
  {
    "RecordName": "this was created via API"
  },
  {
    "RecordName": "this was created via API 2"
  }
]
```

Create/Update/Delete Records Via API

To create/update you must include the Content-Type: application/json header. You can change it to application/xml, but you would have to send XML data instead of JSON.

Creating Records

POST https://{host}/api/2.0/B2C/Records?tableName={table_name} HTTP/1.1

{headers}

{json encoded request body}

Anything can be added. As a result, those two amenities will be created. Please note, endpoint expects an array so you can create any number of records at once. They are not created in transaction, so if something goes wrong, the result may be partially saved. Also, it is important to fill out the required data for the object, otherwise it will return an error. Here is how to create new event in the system:

POST <https://live.perfectmind.com/api/2.0/B2C/Records?tableName=CalendarEvent> HTTP/1.1

X-Access-Key: xxxxxxxxxxxx^*&^(*&xxxxx000xxxxxxxx0000xxxxxxxxxxxxxxxxxxxxxxxx0000000

X-Username: username

X-Password: pssst123

Accept: application/json

Content-Type: application/json

```
[
  {
    "CalendarSetupId": "00000000-x00x-000x-x00x-x00x0x00xx00",
    "StartTime": "2017-02-25T08:30:00-08:00",
    "EndTime": "2017-02-25T10:30:00-08:00",
    "Subject": "Created via API",
    "LocationId": "x000x000-x000-0000-x000-0x00x000xxxx"
  }
]
```

Updating Existing Records

PUT https://{host}/api/2.0/B2C/Records?tableName={table_name} HTTP/1.1

{headers}

{json encoded request body}

For updating it is not enough to only pass key value data. You need to pass data in format:

```
{ "record id": { updated data for the record }, "record 2 id" : {...} }
```

All other rules for creating events are also applicable here.

For Example:

```
PUT https://live.perfectmind.com/api/2.0/B2C/Records?tableName=CalendarEvent HTTP/1.1
X-Access-Key: xxxxxxxxxxxx^@@^###xxxxx000xxxxxxxx0000cccccccccccccccccc0000000
X-Username: username
X-Password: psssst123
Accept: application/json
Content-Type: application/json
{
  "x0x0xx00-0000-0000-0000-x000-000000xxxx00" : {
    "CalendarSetupId": "00000000-x00x-000x-x00x-x00x0x00xx00 ",
    "StartTime": "2017-02-25T08:30:00-08:00",
    "EndTime": "2017-02-25T10:30:00-08:00",
    "Subject": "Updated via API",
    "LocationId": "x000x000-x000-0000-x000-0x00x000xxxx"
  }
}
```

Deleting Existing Records

DELETE <https://{host}/api/2.0/B2C/Records?tableName={tableName}&ids={id}&ids={id2}...>
HTTP/1.1

{usual set of headers}

For Example:

Note: No specific headers are required and the body is also not required.

```
DELETE
https://live.perfectmind.com/api/2.0/B2C/Records?tableName=CalendarEvent&ids=x0x0xx00-
0000-0000-x000-000000xxxx00 HTTP/1.1
X-Access-Key: xxxxxxxxxxxx^*&^ (*&xxxxx000xxxxxxxx0000xxxxxLKUWDCFNUIAFY0000000
X-Username: OrgName
X-Password: psssst123
```