Echo360 Active Learning Platform API

Version 1.2
Reference Guide

Table of Contents

Introduction

This document covers the API supported by the Echo360 active learning cloud application. The API enables programmatic access to the resources within the institution via a RESTful API. The API is an extension to Echo360 that provides external systems direct access to query, create, update and delete the data used by the institution.

Example Use Cases:

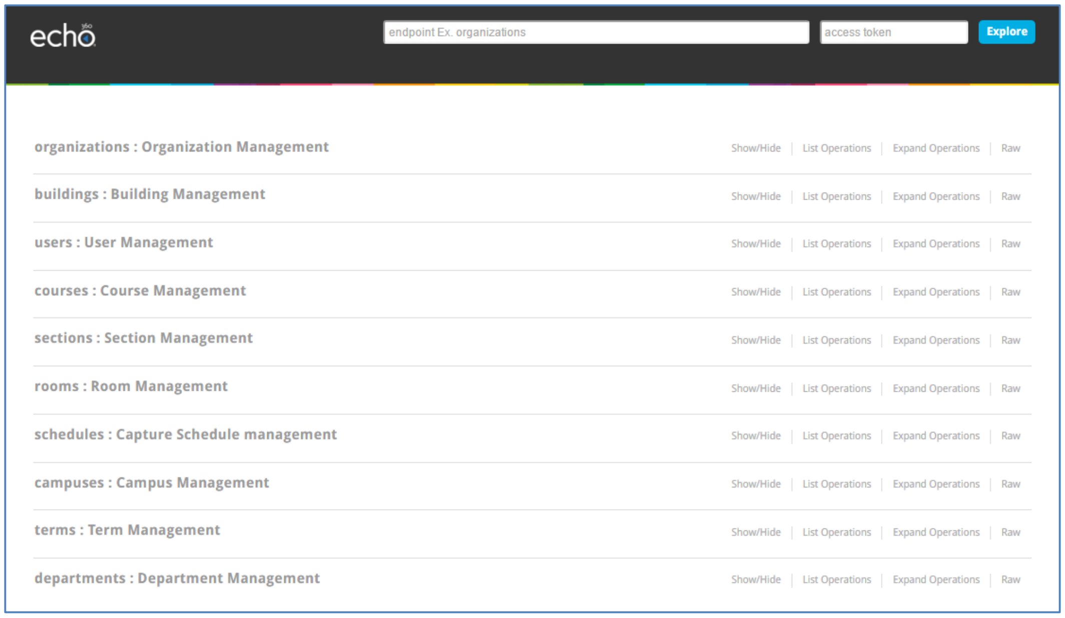
1) Integration with a third party timetabling system to automate the scheduling process within Echo360. Room booking is extracted from the third party system and would automatically create the required entities in Echo360 to populate schedules.
2) Creation of an online booking form that enables Faculty/Academics to request their lectures to be captured. Such a web form might have a backend that integrates with the Echo360 API to add the capture schedules dynamically.
3) Automating the process of creating users, courses, and sections between the Student Information System (SIS) and Echo360.

Online Documentation

The Echo360 API has developed Swagger Docs to accompany the API. If you are not familiar with Swagger docs for APIs, it simply provides a quick reference to the available API calls, their structure, and the model schema for each call.

The Echo360 Swagger docs are available at: **https://**{echo360-url}**/api-documentation**

The main swagger page, shown below, lists all of the endpoint available for the Echo360 API. Click each to browse the available calls, model objects, and model schema.



organizations : Organization Management — Show/Hide | List Operations | Expand Operations | Raw
buildings : Building Management — Show/Hide | List Operations | Expand Operations | Raw
users : User Management — Show/Hide | List Operations | Expand Operations | Raw
courses : Course Management — Show/Hide | List Operations | Expand Operations | Raw
sections : Section Management — Show/Hide | List Operations | Expand Operations | Raw
rooms : Room Management — Show/Hide | List Operations | Expand Operations | Raw
schedules : Capture Schedule management — Show/Hide | List Operations | Expand Operations | Raw
campuses : Campus Management — Show/Hide | List Operations | Expand Operations | Raw
terms : Term Management — Show/Hide | List Operations | Expand Operations | Raw
departments : Department Management — Show/Hide | List Operations | Expand Operations | Raw

If you want to RUN one of the calls from this page, you'll need to enter your access token in the text box on the right side of the top of the page.

See *Base URL* below for clarification on the base echo360 URL to use. See *Authorization* beginning on page 5 for instructions on obtaining an access token.

See *Summary Request URL Listing* beginning on page 14 for a list of the available endpoints for the Echo360 API.

The Basics

This section provides detailed documentation for all of the actions made available by the Echo360 active learning platform API.

Base URL

The API uses same regional domain URL used to access Echo360 via a browser, appended with the path to the API, as follows:

- **US** Region: https://echo360.org/public/api/v1/
- **UK** Region: https://echo360.org.uk/public/api/v1/
- **ANZ** Region: https://echo360.org.au/public/api/v1/

The URI constructs shown in this document typically identify the regional URL as **{echo-url}** with the corresponding API calls added. Where a specific URL is shown, the US region domain is used. Adjust yours according to your region.

Calling the APIs

The Echo360 API provides support for CRUD operations (or a subset thereof) on system objects. The Echo360 APIs accepts requests and returns responses in JSON formats only.

Best practice is to identify the format of the request by supplying a "content-type" HTTPS header, specifying **application/json** for the value. Since JSON is the default (and only) format currently supported, this is not required, however if other formats are introduced later, it is best to have the specification in place.

Most objects in the Echo360 system allow both a GET call for a collection of objects and a GET call for the properties of a particular object. For properties of a particular object, the GET call must include the ID number of the object you want.

For objects that allow PUT and POST calls, the API call must include the necessary objects or attribute values to be applied to the object you are either creating or updating.

For objects that allow DELETE calls, if you delete an object that has *dependent* objects in the system (for example a course that has sections), those dependent objects are deleted (the same as they are in the UI). If you delete an object that has *associated* items (for example, an organization with departments), the associated items are not deleted, but stand alone without the association.

API Objects and Attributes

Each API object has a set of attributes as reflected in the json model schema. In this document, the attributes for an object are the information either sent or returned for that object's API call, having a particular structure and containing particular information.

Each attribute for an object is typically a single value of a specified type (boolean, string, integer, etc.), though some allow for or an array of values, or contain nested values, with two "sub-values" such as startDate and endDate.

See *API Objects and Attributes* beginning on page 18.

Responses to API Calls

All API operations return a response and all responses have a response code (HTTP status code). The code number returned depends on the kind of operation you sent (Get, Put, Post, Delete) and the status of the operation (successful or error/failed).

In general codes in 2XX range indicate success, codes in 4XX range indicate an error, and codes in the 5XX range indicate a problem with ALP.

Successful response status codes are returned as follows:

- **Status code 200:** Typically returned for a request for or modification of information (GET or PUT), and indicates the call was successful. If the call was a GET, you should also receive an object containing the information you requested.
- **Status code 201**: Typically returned for a POST call and indicates that the object was created.
- **Status code 204:** Returned for DELETE calls, indicating the deletion was successful and there is no content to return. There is no body content for these responses.

If an error condition occurs, you will receive a 400 or 500 series HTTP status code along with an HTTP body containing a specific error code and a description of the error.

Error response status codes are returned as follows:

- **Status code 400 - Bad Request:** Indicates the request was not accepted, possibly because an attribute was missing was incorrectly formatted.
- **Status code 401 – Unauthorized:** The access token provided was invalid or the request did not contain an access token.
- **Status code 402 - Request Failed:** The attributes provided were valid but the request failed. Refer to the error code and message provided in the response for further information.
- **Status code 403 – Forbidden:** The request you attempted was forbidden by the system.
- **Status code 404 - Not Found:** The requested resource does not exist.
- **Status codes 500 - Server Error:** Something went wrong on the Echo360 end.

Authorization

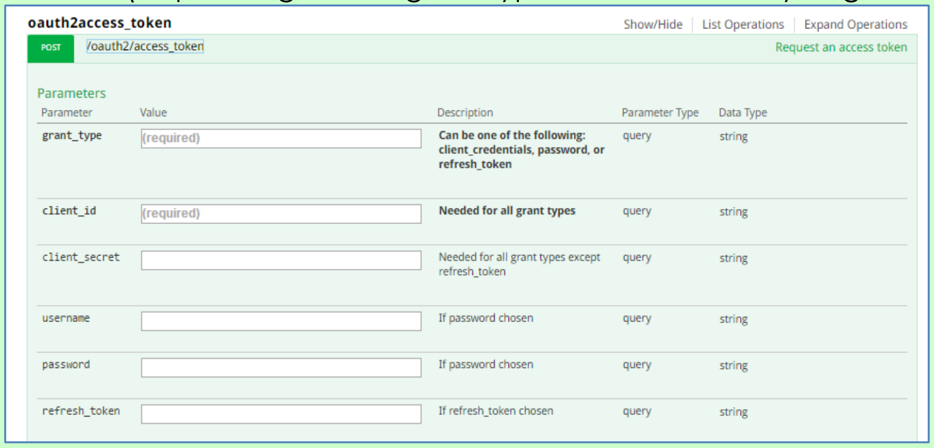
The Echo360 API utilizes OAuth2 for handling system authentication and authorization. OAuth2 is a two part authentication process. The first part requires you to submit a POST call to the authorization endpoint, which is:

https://{echo-url}**/oauth2/access_token**

The access token request must identify the grant_type as **client_credentials**, and include the client ID and client Secret, all in the body of the request.

For instructions on obtaining a client ID and secret from Echo360, see: http://help.echo360.org/API_Documentation.htm.

NOTE: You can now use the Swagger Docs UI to generate an access token and a refresh token. Open the **oauth2access_token** call as shown below, and provide the field values as needed (depending on the grant type selected when you generated the credentials).


oauth2access_token

POST /oauth2/access_token — Request an access token

Parameters

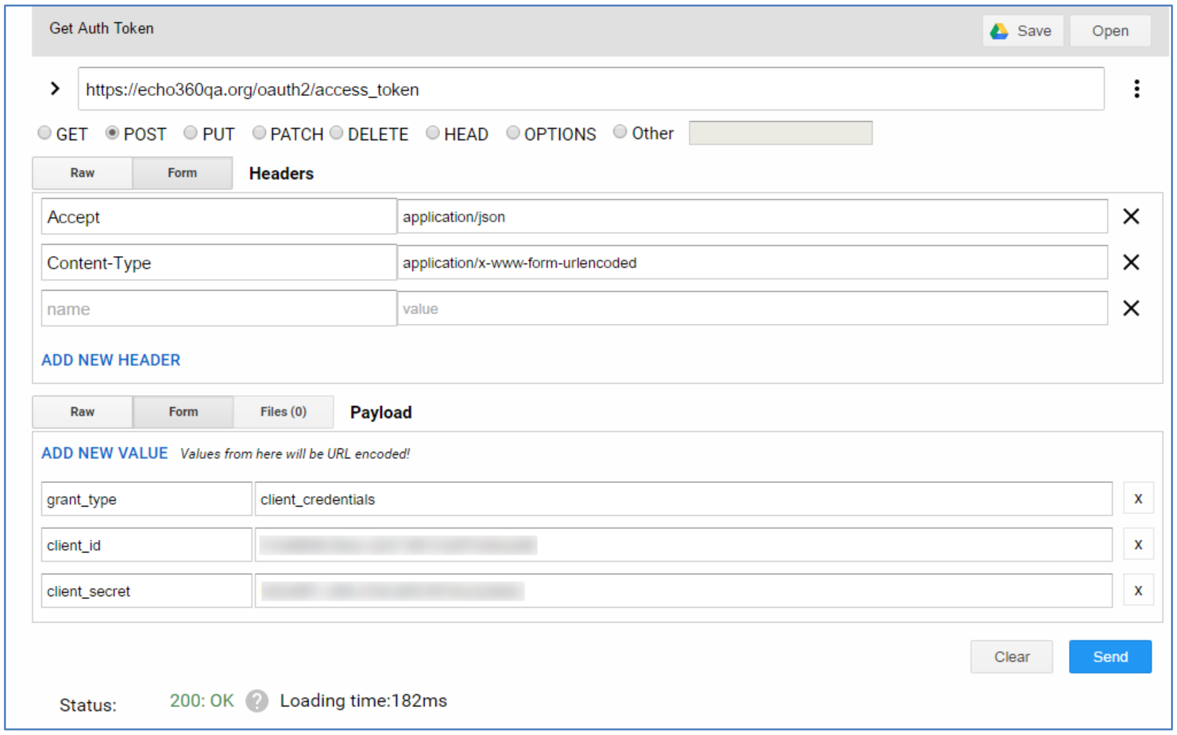
Parameter	Value	Description	Parameter Type	Data Type
grant_type	(required)	**Can be one of the following: client_credentials, password, or refresh_token**	query	string
client_id	(required)	**Needed for all grant types**	query	string
client_secret		Needed for all grant types except refresh_token	query	string
username		If password chosen	query	string
password		If password chosen	query	string
refresh_token		If refresh_token chosen	query	string

The headers must identify the Accept application type as **application/json**, and the Content-Type as **application/x-www-form-urlencoded**.

The request for an access token would include the following:

```
POST https://echo360qa.org/oauth2/access token
 --header ``Accept: application/json''
 --header ``Content-Type: application/x-www-form-urlencoded
 --body
    grant type=client credentials
    client id=<CLIENT ID>
    client_secret=<CLIENT_SECRET>
```

Using a desktop rest API client, the call would look like this:



https://echo360qa.org/oauth2/access_token

GET [✓ POST] PUT PATCH DELETE HEAD OPTIONS Other

Accept	application/json
Content-Type	application/x-www-form-urlencoded
name	value

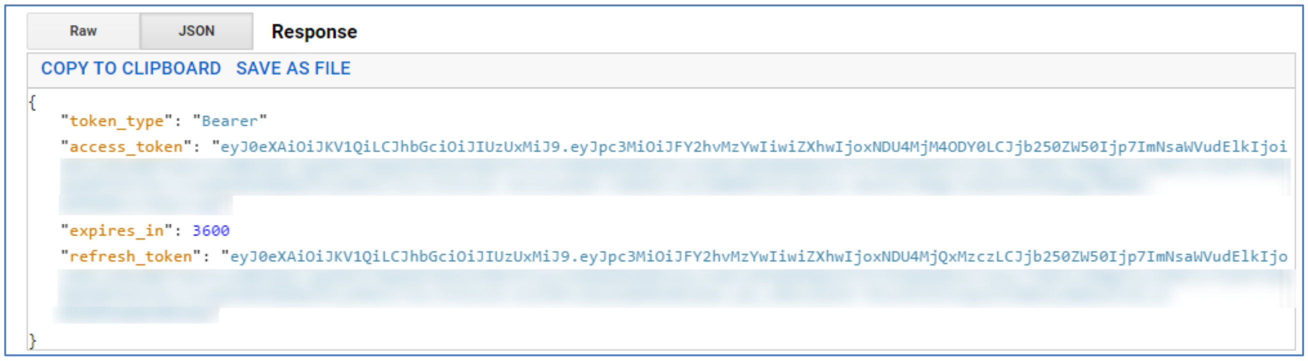
Values from here will be URL encoded!

grant_type	client_credentials
client_id	[redacted]
client_secret	[redacted]

Status: 200: OK Loading time:182ms

The response you receive will contain an access token, which must then be included in the request header for all subsequent API calls. It will also include an expiry time (in seconds) and a refresh token, which can be used after the access token expires to generate a new one. Or you can repeat the above steps to generate a new token.

Access tokens are good for one hour after they are generated.



```
{
  "token_type": "Bearer"
  "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJpc3MiOiJFY2hvMzYwIiwiZXhwIjoxNDU4MjM4ODY0LCJjb250ZW50Ijp7ImNsaWVudElkIjoi
  [redacted]
  "expires_in": 3600
  "refresh_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJpc3MiOiJFY2hvMzYwIiwiZXhwIjoxNDU4MjQxMzczLCJjb250ZW50Ijp7ImNsaWVudElkIjo
  [redacted]
}
```

Access tokens are good for one hour after they are generated.

Using a Refresh Token

If your original access token expires, you can use your Client ID and the refresh token to generate a new access token. This eliminates the need to have the client secret at all times, and the client ID can be obtained via the Echo360 UI, in the **Settings** > **Configurations** >**API client configurations** page.

Alternately, you can provide the client ID and refresh token to a 3rd party to generate their own access token as needed, eliminating the need to disseminate or use the client secret.

To use the refresh token, send a POST call to the authorization endpoint (same one used to get the token in the first place):

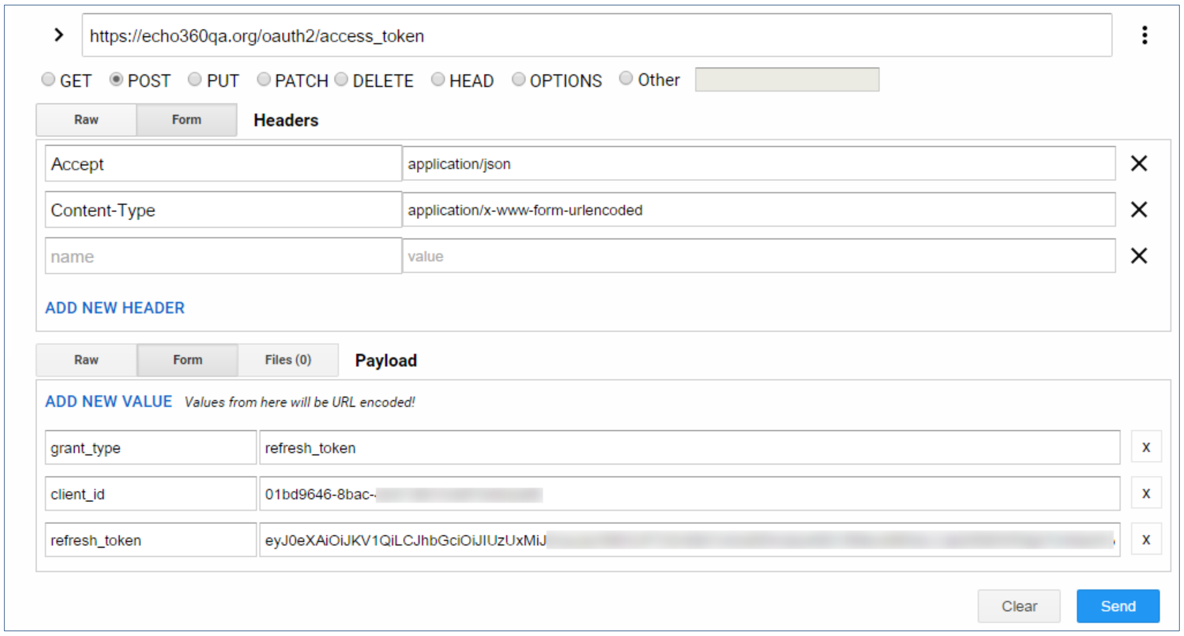
https://{echo-url}**/oauth2/access_token**

The call is almost identical to the original access token call, except that the body includes the client ID and the refresh token value and does NOT include the client secret.

A cURL formatted request for an access token using a refresh token would look as follows:

```
POST https://echo360qa.org/oauth2/access_token
  --header ''Accept: application/json''
  --header ''Content-Type: application/x-www-form-urlencoded
  --body
     grant_type=refresh_token
     client_id=<CLIENT_ID>
     refresh_token = biOncpRsT50jbzRn430zqMLgV3Ia
```

Using a desktop rest API client, the call would look like this:



https://echo360qa.org/oauth2/access_token

Method: ● POST

Accept	application/json
Content-Type	application/x-www-form-urlencoded
name	value

Values from here will be URL encoded!

grant_type	refresh_token
client_id	01bd9646-8bac-[redacted]
refresh_token	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ[redacted]

The response you receive is identical to the original access token call, including a new access token, expiration time, and a new refresh token. The token you just used is now invalid and will NOT return a new access token. Refresh tokens can be used once.

For a good explanation of how OAuth2 works, see https://aaronparecki.com/2012/07/29/2/oauth2-simplified as well as http://stackoverflow.com/questions/4727226/on-a-high-level-how-does-oauth-2-work.

API enabled objects and limitations

The following Echo360 system objects allow for CRUD operations on them via the API:

- Organization
- Department
- Campus
- Building
- Room
- Room configuration (GET only)
- User (DELETE not allowed)
- User enrollments
- Term
- Course
- Section
- Schedule

All of the above calls, except where noted, accept GET, GET (id), PUT, POST and DELETE calls.

Making API Calls

Once you have an access token, you can make requests to the API.

The client ID used to generate the token identifies the institution you are making calls to, and is encoded in the access token. This means that the institution ID does not need to be included in API requests. The access token is enough.

As noted earlier in this document, API calls are made to:

https://{echo-url}/public/api/v1/<endpoint>

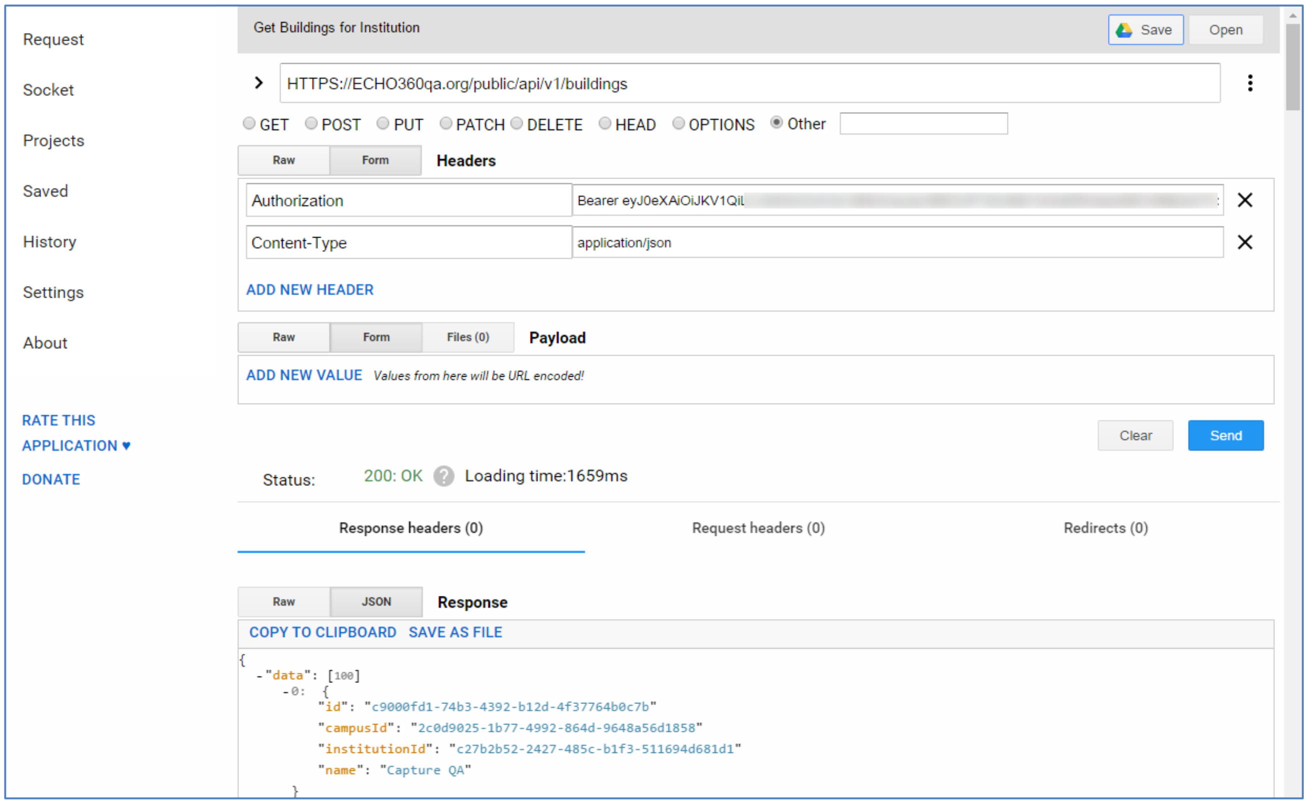
See *Summary Request URL Listing* on page 14 of this document for a full list of API endpoints.

Add the access token as an Authorization type header to each call, **being sure to include "Bearer" in front of the token value**. Headers for API calls should also include a Content-Type: **application/json**.

Using cURL, a GET call for a list of buildings in the institution would look as follows:

```
curl
--header "Authorization: Bearer RsT5OjbzRn430zqMLgV3Ia"
--header "Content-Type: application/json"
--request GET
https://{echo-url}/public/api/v1/buildings
```

Using a desktop API client, the same call would look as follows:


Get Buildings for Institution

HTTPS://ECHO360qa.org/public/api/v1/buildings

GET POST PUT PATCH DELETE HEAD OPTIONS [✓ Other]

Headers

Authorization	Bearer eyJ0eXAiOiJKV1QiL[redacted]
Content-Type	application/json

Payload

ADD NEW VALUE *Values from here will be URL encoded!*

Status: 200: OK Loading time:1659ms

Response headers (0) Request headers (0) Redirects (0)

Response

```
{
 -"data": [100]
   -0:  {
        "id": "c9000fd1-74b3-4392-b12d-4f37764b0c7b"
        "campusId": "2c0d9025-1b77-4992-864d-9648a56d1858"
        "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
        "name": "Capture QA"
      }
```

Making GET calls is a matter of including the proper endpoint and a valid authorization token.

Making PUT or POST calls involves including object information in the body of the request. See *API Objects and Attributes* on page 18 of this document for details on the data and format required for each call.

Besides that, make sure you always send requests over HTTPS. The Echo360 API only allows calls over HTTPS; calls made via HTTP will fail.

Using GET Calls

The Echo360 API uses GET calls to retrieve the following information:

- A list of all of the requested objects in the institution
- Properties for a particular object

GET all objects for an institution

The institution id is a function of the access token; it is already a known attribute when you use the access token to make the call. Any GET call that does not identify a particular object ID will return a list of all of that type of object in the institution.

For example, a GET call to retrieve a list of all buildings for your institution would look as follows (using cURL…you can construct your API calls using anything you like):

```
curl
  --header "Content-Type: application/json"
  --header "Authorization: Bearer 20dcd9fa-a6b1-4ed8-b368-9fe8f8c67027"
  --request GET
https://{echo-url}/public/api/v1/buildings
```

The response you receive will include Building object for each building in the system. Refer to the *API Objects and Attributes* section of this document, beginning on page 18.

```
  {
   "data": [100]
      -98:  {
         "id": "c9000fd1-74b3-4392-b12d-4f37764b0c7b"
         "campusId": "2c0d9025-1b77-4992-864d-9648a56d1858"
         "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
         "name": "Belknap Hall"
      }
      -99:  {
         "id": "8932961f-7276-4e13-836a-16fed48eaaa7"
         "campusId": "136bdaf2-e299-4b92-a350-9f233d052c61"
         "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
         "name": "Grant Science Center"
      }
      -100:  {
         "id": "68091f70-2f5d-46f9-8a9c-36400fb2fd96"
         "campusId": "692144f9-8874-41af-b07e-fb264739a9ed"
         "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
         "name": "Straughn Auditorium"
      }
   "has_more": true
   "next": "/public/api/v1/buildings?limit=100&offset=eyJpbnN0aXR1dGlvbklkIjp7
InMiOiJjMjdiMmI1Mi0yNDI3LTQ4NWMtYjFmMy01MTE2OTRkNjgxZDEifSwiaWQiOnsicyI6IjMzZG
M4OGYzLTY5ZTQtNGM2MC04ZjcyLTUxNDk2ZDBhOWQ4YiJ9fQ%3D%3D"
   }
```

Notice at the bottom of the results a `<next>` attribute. This identifies the last ID number in the currently retrieved list, and allows you to copy that URI construct into the call to retrieve the next set or page of objects.

Once on a subsequent page, you also receive a `<prev>` URI construct to move back to the previous page of results.

Some API clients provide this as a clickable link, as is shown in the example above.

By default, a GET collection call will return 100 items, with subsequent sets of 100 on subsequent pages. Adding a limit attribute to either the body or the URL of the call (for example, **?limit=50**) will change the number of results returned per page.

GET properties of a particular object

If you want to get details about a particular object, use a GET call but include the ID of the object whose properties you want. The following example shows a GET call for a specific building's properties:

```
curl
  --header "Content-Type: application/json"
  --header "Authorization: Bearer 20dcd9fa-a6b1-4ed8-b368-9fe8f8c67027"
  --request GET
https://{echo-url}/public/api/v1/buildings/63e2e36d-da0c-4c4b-a550-
75de97cd69d3
```

This call returns the specific properties of the term associated with the term Id shown, which in this case is pulled from the previous GET collection call.

The response for this GET properties call is the Building object for that building.

```
{
  "status": "ok",
  "message": "ok",
  "data": [
    {
      "id": "63e2e36d-da0c-4c4b-a550-75de97cd69d3"
      "campusId": "136bdaf2-e299-4b92-a350-9f233d052c61"
      "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
      "name": "TRENTON-BN-0000"
    },
  ],
```

As you can see, the Building object identifies the ID for the building and the campus to which it belongs, the institution, and the building name.

Using POST to create objects

The POST call is used to create objects via the API. The example below is the call used to create a course.

Creating a building uses the *Buildings* to provide the necessary data for the building.

The call might look something like this (again in cURL):

```
curl
  --header "Content-Type: application/json"
  --header "Authorization: Bearer 20dcd9fa-a6b1-4ed8-b368-9fe8f8c67027"
  --request POST
  https://{echo-url}/public/api/v1/buildings
  "data": [
    {
        "campusId": "2c0d9025-1b77-4992-864d-9648a56d1858"
        "name": "NEWARK-AN-0022"
    },
    {
        "campusId": "2c0d9025-1b77-4992-864d-9648a56d1858"
        "name": "TRENTON-BN-0000"
     },
```

The Institution ID is not required in the POST call because your institution is known through the access token.

Using PUT to update objects

The PUT call is used to update objects via the API. Logically, the item you are updating must already exist and you must include the object ID in the call.

The example call below is the call used to update a course. Updating a course uses the *Courses object* to provide the necessary data.

The call might look something like this (again in cURL):

```
curl
  --header "Content-Type: application/json"
  --header "Authorization: Bearer 20dcd9fa-a6b1-4ed8-b368-9fe8f8c67027"
  --request POST
  https://{echo-url}/public/api/v1/course
  "data": [
    {
      "courseName": "INTRO LINGUISTICS",
      "courseCode": "ENG-LING-100"
      "organizationId":
      "departmentId": ‘‘25a74b25-f000-49d0-9145-c33abca1d721’’,
    },
  ]
```

Notice in the example how the organizationId is left blank. This particular update changed the name of the course from “English Linguistics” to “Intro Linguistics” and removed the organization from the course hierarchy.

To use the same type of call but ADD this course to an organization (or change the organization to which it belongs), you must include the organizationId. That call would look as follows:

```
curl
  --header "Content-Type: application/json"
  --header "Authorization: Bearer 20dcd9fa-a6b1-4ed8-b368-9fe8f8c67027"
  --request POST
  https://{echo-url}/public/api/v1/course
  "data": [
    {
      "courseName": "INTRO LINGUISTICS",
      "courseCode": "ENG-LING-100"
      "organizationId": "67e2e42d-fa0c-7s4b-b743-39dv84fd69y2",
      "department": ‘‘25a74b25-f000-49d0-9145-c33abca1d721’’,
    },
  ]
```

Bottom line: if you do not include a field in the call, the value will be removed from the object, or the call will return an error indicating the field is required.

The exception is that in some cases the timeZone for the institution will be used if that parameter is missing from a call where it is needed.

Summary Request URL Listing

This section lists the valid API calls for Echo360, grouped by type. Each of the calls shown must be pre-pended with **https://{echo-url}/public/api/** as applies to your region.

Campus API
/v1/campuses GET retrieves a list of all campuses for the institution. POST creates a campus for the institution. Must include Campus object.
/v1/campuses/{campus-id} GET retrieves properties for the identified campus. PUT updates properties for the campus. Must include Campus object. DELETE deletes the campus and all associated buildings & rooms, and orphans any schedules for the deleted rooms.
Building API
/v1/buildings GET retrieves a list of all buildings for the institution. POST creates a building for the institution. Must include Building object.
/v1/buildings/{building-id} GET retrieves properties for the identified building. PUT updates properties for the building. Must include Building object. DELETE deletes the building and all associated rooms, and orphans any schedules for the deleted rooms.
Rooms API
/v1/rooms GET retrieves a list of all rooms for the institution. POST creates a room for the institution. Must include Room object.
/v1/rooms/{room-id} GET retrieves properties for the identified room. PUT updates properties for the campus. Must include Room object. DELETE deletes the room and orphans any schedules for the deleted room.

Room Configuration API – subset of Rooms API
/v1/rooms/{room-id}/{device or config info desired} Only GET calls are supported for this endpoint. See SwaggerDocs for full listing of GET calls available: - https://echo360qa.org/api-documentation#!/room-configuration/
Devices API
/v1/devices/assignable Single GET call available for this endpoint. Returns a list of devices (by MAC address) registered with Echo360 but which are not yet assigned to a room.
Organizations API
/v1/organizations GET retrieves a list of all organizations for the institution. POST creates an organization for the institution. Must include Organizations object.
/v1/organizations/{organization-id} GET retrieves properties for the identified organization. PUT updates properties for the organization. Must include Organizations object. DELETE deletes the organization; associated departments and courses now exist without an organization in their hierarchy.
Departments API
/v1/departments GET retrieves a list of all departments for the institution. POST creates an department for the institution. Must include Departments object.
/v1/departments/{department-id} GET retrieves properties for the identified department. PUT updates properties for the department. Must include Departments object. DELETE deletes the department; associated courses now exist without a department in their hierarchy.

User API
/v1/users GET retrieves a list of all users for the institution. POST creates a user for the institution. Must include User object.
/v1/users/{user-id or email} GET retrieves properties for the identified user. PUT updates properties for the user. Must include User object. **DELETE is NOT a valid call for users at this time**.
Terms API
/v1/terms GET retrieves a list of all terms configured for the institution. POST creates a term for the institution. Must include Term object.
/v1/terms/{term-id} GET retrieves properties for the identified term. PUT updates properties for the term. Must include Term object. DELETE deletes the term; all associated sections (and their schedules) are also deleted.
Course API
/v1/courses GET retrieves a list of all courses configured for the institution. POST creates a course for the institution. Must include Course object.
/v1/courses/{course-id} GET retrieves properties for the identified course. PUT updates properties for the course. Must include Course object. DELETE deletes the course; all associated sections (and their schedules) are also deleted.
Section API
/v1/sections/ GET retrieves a list of all sections configured for the institution. POST creates a section for the institution. Must include Section object. NOTE: Section Names must be unique for the Course/Term.

/v1/sections/{section-id} GET retrieves properties for the identified section. PUT updates properties for the section. Must include Section object. DELETE deletes the section; all associated schedules as well as section analytics are also deleted.
Enrollments API – subset of Section API
/v1/sections/{sectionId}/users GET retrieves a list of all users enrolled in the identified section, including role. POST adds a user to the section. Must include user id/email and role.
/v1/sections/{sectionId}/users/{userId} GET retrieves the role for the identified user in the identified section. PUT updates the role for the user in the section. DELETE removes the user from the identified section.
Schedule API
/v1/schedules/ GET retrieves a list of all schedules configured for the institution. POST creates a schedule for the institution. Must include Schedule object.
/v1/schedules/{schedule-id} GET retrieves properties for the identified schedule. PUT updates properties for the schedule. Must include Schedule object. DELETE deletes the schedule; all future classes created for the schedule are deleted (unless they already contain content).

API Objects and Attributes

The tables in this section list the objects necessary for utilizing the API, and provides details regarding the attributes for each object, whether returned with a GET or required for a PUT, POST, or DELETE call. Each table provides the attribute name, a description, the format, and whether or not the attribute is required.

API Objects simply provide a data structure for the information being used to create or update the resources in the system.

Note the following about the object information:

- The attribute name must be provided exactly as shown below, including camelCase capitalization.

- Valid values, where identified, are case sensitive.

- It does not matter what ORDER the attributes appear with the call.

- **institutionId** is returned in the response for each GET call. This attribute is not in the tables below because it is not required for PUT, POST, or DELETE calls, and it is the same for every object. The institution is tied to the access token and is therefore assumed.

Campuses object

Attribute	Description	Data type	Required?
id	System identifier for the campus	string	Yes for PUT or DELETE No for POST
name	The name of the campus	string	Yes
timeZone **NOTE:** value is case sensitive	The name of the time zone where the campus resides. Ex: US/Eastern	string (allowable Value)	Yes for PUT or POST No for DELETE
timeZoneOffsetMinutes	Offset of the local time zone from UTC, in minutes. Ex: -500	integer	No (returned with GET). Calculated based on timeZone

Sample GET response for campuses

```
65:   {
    "id": "19befdc8-f929-454f-94a0-f6c02f6bccde"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "BUNBURY"
    "timeZone": "Australia/Perth"
    "timeZoneOffsetMinutes": 480
}
-66{
    "id": "c38dc7dc-929f-4306-84a4-ac7bb7996fe1"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "BOSTON CAMPUS"
    "timeZone": "US/Eastern"
    "timeZoneOffsetMinutes": -300
```

Buildings object

Attribute	Description	Data type	Required?
id	System identifier for the building	string	Yes for PUT or DELETE No for POST
name	The name of the building	string	Yes
campusId	System identifier for the campus where the building exists	string	Yes

Sample GET response for buildings

```
0:  {
    "id": "c9000fd1-74b3-4392-b12d-4f37764b0c7b"
    "campusId": "2c0d9025-1b77-4992-864d-9648a56d1858"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "Belknap Hall"
}
-1: {
    "id": "8932961f-7276-4e13-836a-16fed48eaaa7"
    "campusId": "136bdaf2-e299-4b92-a350-9f233d052c61"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "Grant Science Center"
```

Rooms object

Attribute	Description	Data type	Required?
id	System identifier for the room	string	Yes for PUT or DELETE No for POST
name	The name of the room	string	Yes
buildingId	System identifier for the building where the room exists	string	Yes
roomConfigurationId	System identifier of the room configuration (device) assigned to the room	string	No (returned with GET if room has a device)
deviceSoftwareVersion	Version of the firmware running on the device	string	No (returned with GET if room has a device)
createdAt	DateTime value for room creation moment	instant	No (returned with GET)
updatedAt	DateTime value for last room update moment	instant	No (returned with GET)

Sample GET response for rooms:

```
0:   {
     "id": "24edb397-24ae-4a58-bb5e-b4a666634547"
     "buildingId": "4d5bfeb1-7bc4-41e1-87c8-b8438d14e735"
     "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
     "name": "Belknap 203"
     "roomConfigurationId": "0e3a813d-1934-43da-a4ef-5ca6dbcf6516"
     "deviceSoftwareVersion": "5.5.566602856"
     "createdAt": "2015-10-05T15:24:53.493Z"
     "updatedAt": "2015-10-05T15:24:53.493Z"
}
-1:   {
     "id": "dfc38578-3380-46b0-b400-c554a05a17a0"
     "buildingId": "bec9863f-33af-454e-a01d-910c0ff392d6"
     "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
     "name": "Belknap 201"
     "roomConfigurationId": "2eee4a83-a5ad-42f5-980d-e32888c8d899"
     "deviceSoftwareVersion": "5.5.566602856"
     "deviceId": "00-1C-08-00-40-26",
     "deviceType": "ProHardwareCapture2",
     "createdAt": "2015-10-05T15:24:53.493Z"
     "updatedAt": "2015-10-05T15:24:53.493Z"
}
```

Devices object – supports only GET calls

Retrieves a list of unassigned devices, visible to Echo360 but not yet assigned to a Room.

Attribute	Description	Data type
id	MAC address of the device.	string
deviceType	Type of capture applicance. SCHD, PRO, POD or SoftwareCapture (CCAP installation)	string

Sample GET response for devices

```
{
    "id": "00-1C-08-00-01-F2",
    "deviceType": "ProHardwareCapture",
  },
```

Organizations object

Attribute	Description	Data type	Required?
id	System identifier for the organization	string	Yes for PUT or DELETE No for POST
name	The name for the organization	string	Yes for PUT or POST No for DELETE
deptartmentCount	The number of departments in the organization	long	No (returned with GET)
courseCount	The number of courses in the organization	long	No (returned with GET)
sectionCount	The number of sections in the organization	long	No (returned with GET)

Sample GET response for organizations

```
-1:  {
    "id": "e01645ef-d542-41dd-9a5a-64d7e1f66410"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "ORG-0002"
    "departmentCount": 7
    "courseCount": 58
    "sectionCount": 123
}
```

Departments object

Attribute	Description	Data type	Required?
id	System identifier for the department	GUID	Yes for PUT or DELETE No for POST
name	The name of the department	string	Yes
organizationId	System identifier for the organization to which the department belongs	string	Yes for POST or PUT only if dept. must belong to an org. No if dept. exists without org. or for DELETE.
courseCount	The number of courses in the department	long	No (returned with GET)
sectionCount	The number of sections in the department	long	No (returned with GET)

Sample GET response for departments

```
-1:  {
    "id": "f138290d-3a1d-43a2-8c42-b2afe5d9b380"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "Computer Science"
    "courseCount": 7
    "sectionCount": 13
}
-2:  {
    "id": "4fc1ce22-07a7-47a1-94d2-2f3a1af3e222"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "Political Science"
    "courseCount": 9
    "sectionCount": 17
}
```

Courses object

Attribute	Description	Data type	Required?
id	System identifier for the course	string	Yes for PUT or DELETE No for POST

Attribute	Description	Data type	Required?
name	The name for the course	string	Yes for POST or PUT No for DELETE
courseIdentifier	The course code or short identifier for the course. Ex: MGMT-100	string	Yes
departmentId	System identifier for the department to which the course belongs	string	Yes for POST or PUT if course belongs to a dept. No for DELETE or if course exists without dept.
organizationId	System identifier for the organization to which the course belongs.	string	Yes for POST or PUT if course must belong to an org. No for DELETE or if course exists without org.
sectionCount	The number of sections in the course.	long	No (returned with GET)

Sample GET response for courses

```
-2:  {
    "id": "b0e4bfdc-a99a-4c64-b20c-19dd2b60f2c1"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "organizationId": "9b7b3bba-1b9c-44a7-9262-442a75682e5c"
    "departmentId": "25a74b25-f000-49d0-9145-c33abca1d721"
    "name": "Organic Chemistry"
    "courseIdentifier": "OrgChem300"
    "sectionCount": 3
}
-3:  {
    "id": "1d466b28-02b6-4d29-9b4f-dcd84a3ae762"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "organizationId": "9b7b3bba-1b9c-44a7-9262-442a75682e5c"
    "departmentId": "25a74b25-f000-49d0-9145-c33abca1d721"
    "name": "Organic Chemistry LAB"
    "courseIdentifier": "OrgChemLAB"
    "sectionCount": 3
}
```

Terms object

Attribute	Description	Data type	Required?
id	System identifier of the term	string	Yes for PUT or DELETE No for POST
name	The name for the term Ex: Spring 2016	string	Yes for POST or PUT No for DELETE
session	Identifies the start date and end date of the term. Is a nested attribute containing two date fields Format: YYYY-MM-DD	DateRange startDate endDate	Yes for POST or PUT No for DELETE
exceptions	Identifies any dates within the term session range during which classes will NOT occur. Is a nested attribute containing sets of start and end date fields Format: YYYY-MM-DD	Array [DateRange] startDate endDate	Yes for POST or PUT if applicable No for DELETE or if term has no exception dates
sectionCount	The number of sections in the term	long	No (returned with GET)

Example GET response for terms

```
{
    "id": "6b938ad1-a3bb-49cc-9082-24e8db8f2912"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "name": "FALL2015"
    "session":
    {
       "startDate": "2015-08-10"
       "endDate": "2015-12-15"
    }
    "exceptions":
    {
       "startDate": "2015-09-04"
       "endDate": "2015-09-07"
    }
    {
       "startDate": "2015-11-23"
       "endDate": "2015-11-30"
    }
"sectionCount": 237
}
```

Sections object

Attribute	Description	Data type	Required?
id	System identifier of the section	string	No for POST Yes for PUT or DELETE
courseId	System identifier for the course to which this section belongs **NOTE** can also be the course code instead of UUID	string	Yes for PUT or POST No for DELETE
termId	System identifier for the term during which the section occurs **NOTE** can also be the term name instead of UUID	string	Yes for PUT or POST No for DELETE
sectionNumber	Code or number identifier for the section. Ex: COMP-101	string	Yes for PUT or POST No for DELETE
description	Text description of the section	string	Yes for PUT or POST No for DELETE
instructorId	System identifier of the primary instructor for the section **NOTE** can also be the email address of the instructor	string	Yes for PUT or POST No for DELETE
scheduleIds	System identifier(s) for all scheduled captures configured for this section	string (set)	No (returned with GET)
lessonCount	Number of classes configured for this section	long	No (returned with GET)

Attribute	Description	Data type	Required?
userCount	Number of users enrolled in this section (student and instructor)	long	No (returned with GET)
secondaryInstructorIds	System identifiers of any additional instructors assigned to the section. **NOTE** can also be the email address of the instructor(s)	Plural Seq[String]	Optional
externalSystemIds	Values for the LMS course/section to which this section should be linked. Value to be used varies by LMS. See http://help.echo360.org/Linking_LMS_Courses_with_Echo360_Sections_via_the_API.htm	Plural Seq[String]	Optional

Sample GET response for sections

```
{
    "id": "060bf1d3-91a5-4c74-86b9-9e4f6533b93a"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "courseId": "10dfb872-2d22-47d7-8cb8-00725c146c7b"
    "termId": "96d9aaad-8a1c-4ac3-8a7e-e2983e3616d3"
    "scheduleIds": [2]
         0:  "e0584c53-d65a-4ca1-a611-7de0ae93455e"
         1:  "e6e6ab51-9623-4df3-a843-d5b2c8fdbc8c"

    "sectionNumber": "ODD-350"
    "instructorId": "1423d63a-4630-474a-b708-882fd7a84b25"
    "lessonCount": 13
    "userCount": 18
    "secondaryInstructorIds":
         0:  "e0584c53-d65a-4ca1-a611-7de0ae93455e"
         1:  "e6e6ab51-9623-4df3-a843-d5b2c8fdbc8c"
    "externalSystemId"
         0:  "1885468"
```

Schedules object

Attribute	Description	Data type	Required?
id	System identifier for the schedule	string	No for POST Yes for PUT or DELETE
name	Name to be given to the capture(s) generated by this schedule	string	Yes for PUT or POST No for DELETE
sectionId	System identifier for the section to which this schedule and the captures generated belong	string	Optional, but required if captures are to be auto-published to a section.
roomId	System identifier for the room (device) where this schedule's captures occur	string	Yes for PUT or POST No for DELETE
startTime	Start time for each capture generated by this schedule Format – 24-hour: HH:MM Ex: 09:30 for a 930am start time Ex: 18:00 for a 6pm start time Local timeZone for the room is used	string	Yes for PUT or POST No for DELETE
startDate	First date on which the scheduled captures occur Format: YYYY-MM-DD	string	Yes for PUT or POST No for DELETE
endDate	Last date on which the scheduled captures occur Format: YYYY-MM-DD	string	Optional for non-recurring captures

Attribute	Description	Data type	Required?
daysOfWeek	Days of the week on which recurring captures occur Valid values: SU, MO, TU, WE, TH, FR, SA	string separate multiple values by commas	Required if endDate is set
exclusionDates	Identifies any dates on which captures will NOT occur If a section is specified, these dates are in addition to any exclusions configured for the section's term If NO section is specified, these dates are the only exceptions to this series of scheduled captures Format: YYYY-MM-DD	Array [Date]	Optional **Only** used if endDate & daysOfWeek is set GET returns only exceptions set for schedule and does not include term exclusions for the section if one is specified.
instructorId	System identifier of the instructor for the schedule. **NOTE** can be the email address of the instructor. Captures generated by this schedule are "owned" by this user.	string	Optional If not included: - if no sectionId, captures generated have no owner. - with sectionId, primary instructor of section is capture owner
guestInstructor	Name of a guest instructor for the schedule. Used strictly to display the name with the capture.	string	Optional.
shouldCaption	Identifies whether the capture(s) generated for this schedule are sent for closed captioning	boolean (true or false)	Optional If left blank, defaults to false.

Attribute	Description	Data type	Required?
shouldAutoPublish	Identifies whether the capture(s) generated for this schedule are automatically published to the section when finished.	boolean (true or false)	Required If a sectionId is entered, can be true or false If sectionId is blank, value MUST be false
shouldStreamLive	Identifies whether the capture(s) is streamed live.	boolean (true or false)	Optional If left blank, defaults to false If sectionId is blank, value must be false
input1	Identifies the input being used as the primary visual input Valid values: Display, Video, AltVideo	string	Yes if capturing graphical feed (audio is always captured)
input2	Identifies the input being used as the secondary visual input Valid values: Display, Video, AltVideo	string	Yes if capturing dual graphical feed Optional if single graphical or audio only capture
captureQuality	Identifies the quality for the generated capture(s) Valid values: Medium or High	quality	Yes
streamQuality	Identifies the quality for the live stream Applies only to LIVE schedules for an Echo360 PRO appliance **Valid values**: Medium or High	string	Optional

Sample GET response for schedules

```
61: {
    "id": "79b5bf8d-c8de-4355-9b7b-4bf3c7c64c1f"
    "startDate": "2015-12-16"
    "startTime": "12:00:00"
    "endDate": "2015-12-21"
    "daysOfWeek": [4]
       0:  "MO"
       1:  "WE"
       2:  "FR"
       3:  "SA"
    "exclusionDates": [2]
       0:  "2015-12-18"
       1:  "2015-12-19"
    "durationMinutes": 35
    "sectionId": "dec0a763-c860-421d-8058-48f45380481e"
    "name": "Organic Chem 205"
    "roomId": "7ae4c7b1-6fef-468d-8995-07729750573f"
    "instructorId": "1423d63a-4630-474a-b708-882fd7a84b25"
    "shouldCaption": true
    "shouldAutoPublish": true
    "shouldStreamLive": true
    "input1": "Display"
    "input2": "Video"
    "captureQuality": "High"
}
-62:{
    "id": "0dbaed78-9229-4ec0-847a-08fe19acef86"
    "startDate": "2015-11-30"
    "startTime": "13:00:00"
    "durationMinutes": 15
    "sectionId": "b1033941-3145-4fea-9f02-dc4450740d45"
    "name": "Differential Equations II"
    "roomId": "d2e61fb7-361b-4795-8b4f-aacf1603e876"
    "instructorId": "1423d63a-4630-474a-b708-882fd7a84b25"
    "shouldCaption": true
    "shouldAutoPublish": true
    "shouldStreamLive": false
    "input1": "Video"
    "input2": "AltVideo"
    "captureQuality": "High"
}
```

Users object

Attribute	Description	Data type	Required?
id	System identifier of the user	string	No for POST Yes for PUT or DELETE
email	Email address of the user	string	Yes
firstName	First name of the user	string	Optional
lastName	Last name (family or surname) of the user	string	Optional
timeZone **NOTE:** value is case sensitive	The name of the time zone where the user is located Ex: US/Eastern If not included, will use the timeZone of the institution.	string	Optional If not included, will use the timeZone of the institution
timeZoneOffsetMinutes	Offset of the user's local time zone from UTC, in minutes. Ex: -500	integer	No (returned with GET)
phoneNumber	Region and phone number of the user. Is a nested object with two fields: **region** and **number** EX: US/212-555-1212	phoneNumber	Optional
profileImageUrl	URL or fully qualified network path to an image file for the user's avatar	string	Optional
roles **NOTE:** value is case sensitive	The system role for the user **Valid values**: Student Instructor Admin	string; separate multiple values with a comma	Yes for PUT or POST

Example GET response for users

```
0:  {
    "id": "f69c9e0e-5395-473e-be16-b4ecd6d49ab9"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "email": "TeachingAssistant@exampleuser.edu"
    "timeZone": "US/Eastern"
    "timeZoneOffsetMinutes": -300
    "firstName": "Teaching"
    "lastName": "Assistant"
    "roles":[2]
       0:   "Student"
       1:   "Instructor"
}
-1: {
    "id": "a4253ccd-8a8c-4e34-8794-461d6aca4b2e"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "email": "instructorName@exampleuser.edu"
    "timeZone": "US/Eastern"
    "timeZoneOffsetMinutes": -300
    "firstName": "Instructor"
    "lastName": "Name"
    "roles": [1]
       0:   "Instructor"
}
```

Section Enrollments object

Attribute	Description	Data type	Required?
userId	System identifier for the user enrolled in this section	string	Yes
role	Role of the user in this section (instructor or student)	string	Yes
sectionId	System identifier for the section whose enrollment the call is for.	string	Yes
termId	System identifier for the term during which the section occurs **NOTE** can also be the term name instead of UUID	string	Yes for PUT or POST No for DELETE
courseId	System identifier for the course to which this section belongs **NOTE** can also be the course code instead of UUID	string	Yes for PUT or POST No for DELETE

Sample GET response for enrollments

```
{
    "userId": "060bf1d3-91a5-4c74-86b9-9e4f6533b93a"
    "sectionId": "060bf1d3-91a5-4c74-86b9-9e4f6533b93a"
    "role": "Instructor"
    "institutionId": "c27b2b52-2427-485c-b1f3-511694d681d1"
    "courseId": "10dfb872-2d22-47d7-8cb8-00725c146c7b"
    "termId": "96d9aaad-8a1c-4ac3-8a7e-e2983e3616d3"
}
```

LMS-profiles object

Attribute	Description	Data type	Required?
id	System identifier for the LMS profile being returned	string	Yes
lmsName	Name of the LMS	string	Yes
token	Identifies the secure communication values used in the LTI tool configuration in the LMS.	Array [string] sharedSecret consumerKey	Yes
ltiEndpoint	Identifies the host URL value for Echo360 to be used in the LTI tool configuration in the LMS	string	Yes
externalSystemIdRef	Identifies the name of the field in the LTI payload whose value is to be used to directly link the LMS course/section to the Echo360 section. The corresponding value for each LMS course/section is entered as the externalReferenceId in the Section object	string	Optional
ltiCartridge	The LTI cartridge (typically XML) to be used for LMS>Echo360 LTI tool configuration	Array [string]	Optional

Sample GET response for lms-profiles

```
{
      "id": "1fd12eee-0cc2-416f-bc2a-e0985709b4f7",
      "lmsName": "Blackboard",
      "token": {
        "sharedSecret": "831aacde-555d-4d5c-a18a-2ee392d931db",
        "consumerKey": "198af60b-647c-44d0-8345-2f2123732210"
      },
      "externalSystemIdRef": "context label",
      "ltiEndpoint": "/lti/1fd12eee-0cc2-416f-bc2a-e0985709b4f7"
    },
    {
      "id": "70a17e88-5d9f-4000-af0d-f6d2cfd6b40e",
      "lmsName": "Canvas",
      "token": {
        "sharedSecret": "ac056d2e-f90e-423f-aa14-26d8663ba090",
        "consumerKey": "29c5dd4e-628f-46db-951d-c89e6d9971f5"
      },
      "externalSystemIdRef": "custom canvas course id",
      "ltiEndpoint": "/lti/70a17e88-5d9f-4000-af0d-f6d2cfd6b40e"
    },
```

Error Codes and Messages

This section lists error codes that may be returned from unsuccessful API calls to Echo360, along with the message you will receive for each.

Note: The list below is not an exhaustive list of errors that can be encountered. Furthermore we are working to provide likely explanations for each error (what might have gone wrong) where the returned message isn't obvious. When completed, an updated reference guide will be posted.

Error Code	Messages
BadRequest	400 Bad Request
NotAuthenticated	401 Not Authenticated
Forbidden	403 Forbidden
NotFound	404 (Resource) Not Found
MethodNotAllowed	405 Method Not Allowed
InternalError	500 Internal Server Error
BuildingNotFound	"msg": "Building Not Found"
CampusIdRequired	"msg": "Campus ID Required"
CampusNotFound	"msg": "Campus Not Found"
CourseIdentifierNotUnique	"message": "CourseIdentifier Not Unique"
CourseIdentifierRequired	""msg":"Course Identifier Required"
CourseNotFound	"error":"NotFound","message":"Course Not Found"
DepartmentNotFound	"message": "Department Not Found"
DepartmentNotInOrganization	"message": "Department not in Organization"

Error Code	Messages
DuplicateModelFound	Messages Vary
EmailAddressRequired	"obj.email":[{"msg":"Email Address Required",}]
FirstNameRequired	"obj.firstName":[{"msg":"First Name Required","args":[]}]}
InvalidEmailProvided	"msg": "Invalid Email Provided"

InvalidPhoneNumber	"msg": "Validation error -- InvalidPhoneNumber"
	or
	"obj.phoneNumber.number": [{
	"msg": "error.path.missing", // Internationalized to "Please fill this entry" }]
	or
	"obj.phoneNumber.region": [{
	"msg": "error.path.missing",// Internationalized to "Please fill this entry" }]
	or
	"obj.phoneNumber.region": [{
	"msg": "Enumeration expected of type: 'class models.CountryCode$', but it does not appear to contain the value.", }]

LastNameRequired	"obj.lastName":[{"msg":"Last Name Required",]}
NameRequired	"obj.name": [{ "msg": "Name Required", "args": [] }
NoBlankCourseIdentifier	"obj.courseIdentifier":[{"msg":"Invalid course identifier",]}
NoBlankDepartmentId	"obj.departmentId":[{"msg":"Invalid Department ID",]}
NoBlankOrganizationId	"obj.organizationId":[{"msg":"Invalid Organization ID",]}
NoBlankRoomName	"obj.name":[{"msg":"Invalid Name","args":[]}]}
NotUnique	Messages Vary - "obj.name":[{"msg":"<whatever attribute> Not Unique","]}

Error Code	Messages
OrganizationNotFound	"message": "Organization Not Found"
RoomNotFound	"error":"NotFound","message":"Room Not Found"
SectionNotFound	"error":"NotFound","message":"Section Not Found"
TermNameNotUnique	"obj.name":[{"msg":"Term Name not Unique",]}
TermNameRequired	"obj.name":[{"msg":"Name Required",]}
TermNotFound	"error":"NotFound","message":"Term not found"
UnknownTimeZone	"obj.timeZone":[{"msg":"Unknown Time Zone"]}