Table of Contents

Introduction .................................................................................................................................................. 2
Architectural design .................................................................................................................................. 2
API Overview .............................................................................................................................................. 2
  GET Methods list ..................................................................................................................................... 2
  POST Method List ................................................................................................................................... 5
Code examples ........................................................................................................................................... 5
  Perquisites ................................................................................................................................................ 5
  JQuery ..................................................................................................................................................... 5
  Base-64 ASCII encoding .......................................................................................................................... 5
  User Authorization .................................................................................................................................... 5
Example calls ............................................................................................................................................. 6
  Introduction: Simple GET call .................................................................................................................. 6
  Simple POST Call .................................................................................................................................... 7
  Passing dates as parameters .................................................................................................................... 7
  Passing lists as parameters ...................................................................................................................... 8
Method Specification ............................................................................................................................... 9
Introduction

The Accord LMS Web API provides access to all the LMS Administration Module services. Clients can programmatically pull their learner catalog, tracking and report information from the Administration Module into their own custom application, such as a specialized Learner module. The Administration Module will continue to directly track all Learner utilization of Learning Events including attempts, dates, scores, etc.

Architectural design

The Web API is based on a request/response architectural style. The series of REST Web methods exposed by the API are designed to be access via Http Requests, this requests must contain authorized user credentials along with the specific web method parameters. On the other hand the server response is returned as a JSON object.

API Overview

There are two different Http actions used for interacting with the API:
GET: Requests a representation of the specified resource. It has no side effects in the application.
POST: Requests result in the creation of a new resource and/or update of existing resources.

The web methods are listed below along with their description. The methods signature (without return types) can be found in the "Methods specification" section

GET Methods list

The following methods are designed to be called via GET:

GetLearnerReportScoreDetailsFile
Returns a custom formatted Learner Score Details report File

GetLearnerReportTranscriptFile
Returns a custom formatted Learner Transcript File

GetLearnerReportScoreDetailsFile
Returns a custom formatted Learner Attempts History report File

LearnerTranscriptGet
Returns the list of Folders for a given learner user, applying the filter criteria.

LearnerTranscriptByFolderGet
Returns the list of Folders for a given learner user, applying additional folder filter criteria.

GetLearningEvents
Returns the list of Learning Elements for a given learner user, applying the filter criteria.

GetFilteredLearningEvents
Returns the list of Learning Elements for a given learner user, applying additional filters

GetCatalog
Returns the Learning Elements and Folders tree structure applying filters

GetFilteredCatalog
Returns the Learning Elements and Folders tree structure applying additional filters

GetFilterCatalogSimplified
Returns the Learning Elements and Folders tree structure applying filters and simplifications if requested.

GetFilteredCatalogSimplifiedByFolder
Returns the Learning Elements and Folders tree structure applying regular filters, folder filters and simplifications if requested.

GetFilteredCatalogSimplifiedByFolderConditions
Returns the Learning Elements and Folders tree structure applying the regular filters, additional folder filters and simplifications if requested.

GetCatalogForResources
Returns the Learning Elements and Folders tree structure applying the filters.

GetLearningEventByLEId
Returns a fully loaded Learning Element object

GetLearningEventByLEIdByAttemptsOption
Returns a fully loaded Learning Element object filtered by attempt options

GetMinimalLEDataByLd
Returns a Learning Element with minimal information

GetConditionByFolderIdAndLdId
Gets the existent conditions for a given LE and a given folder

GetCondition
Returns the current condition status for the given User and LE, conditions met and not met

GetConditionByAttemptsOption
Returns the current condition status for the given User and LE, condition met not met filtered by attempt options

GetRoleAwardAssignedRolesByAttemptsOption
Returns the list of roles that the user should be (and probably has been) awarded filtered by attempt options

GetRoleAwardAssignedRoles
Returns the list of roles that the user should be (and probably has been) awarded
GetRoleAwardByLEId
Get all the Role award objects for a given LE

GetManagerByLauncher
Returns the Administration Module By providing the Launcher module

GetAttemptByld
Gets the Attempt By Id

GetInteractionsByAttemptId
Returns the LE Interactions By Attempt Id

GetAttempts
Returns a list of attempts for the given LE

GetAttemptsByDateFiltered
Returns the Attempts from the User with the roles specified to the LEs in the specified folders filtered by dates

GetAttemptsByDate
Returns the given user attempts to the given LE between the given dates

GetAttemptsHistoryHeader
Returns the learner, attempts count, passing score and average score data from the attempts of the given LE

GetLearningEventDetailsByLEId
Gets an LE’s detailed info

GetLearningEventDetailsByLEIdFiltered
Gets an LE’s detailed info filtered by attempt options

GetBaseFolderByld
Returns a BaseFolder object

HasLearnerFolderAccess
Returns true if the user has learner catalog access to the folder

GetHTTPRequestRedirect
Returns the URL associated with the Manager and the User roles
POST Method List

The following methods are designed to be called via POST:

ProcessAccessCode
Processes the given Access Code

EnrollLE
Enrolls the given User to the given LE

EnrollFolder
Enrolls the given User to the given Folder

DropEnrollFolder
Drops the enrollments for each LE of the given Folder for the given User

DropEnrollLE
Drops the enrollment to the given user for the given LE

Code examples

As stated before the API is designed to be called via Http requests. Here we will show how to perform calls using JavaScript due to its popularity as a client side language. In the following section, the different ways to call the web methods will be listed with focus on practical examples

Perquisites

JQuery
This guide assumes that the requests will be made using JQuery. JQuery minified can be found here: http://ajax.googleapis.com/ajax/libs/jquery/1.3/jquery.min.js

Base-64 ASCII encoding

To add an extra security measure we will be sending the user and password information encoded in base-64 using javascript’s “btoa” (https://developer.mozilla.org/en-US/docs/Web/API/window.btoa). This function comes embedded by default in most web browsers

User Authorization

In order for the web services to return a proper response, “Enable Web Services” checkbox must be checked and the role selected in the drop down list “Role” must match the requesting user Role. (Manager >> Configuration >> General Options >> API Web Services)
Example calls

Introduction: Simple GET call

In this first example we will be calling LearnerTranscriptGet:

LearnerTranscriptGet(int userId,
                        int managerModuleId,
                        string searchCriteria)

Before the ajax call we'll enable support.cors. This will allow us to include user credentials in the request. Then we will create a variable (params) containing the parameters for the call:

```javascript
jQuery.support.cors = true;
var params = {
    userId : 30,
    managerModuleId : 412,
    searchCriteria : "***"
};
```

The parameters name must match the exact (case sensitive) name provided in the "Method Specification" section below. Otherwise the server will return a resource not found response. In this particular example we use "***" (star character) as a search criteria. This character behaves the same way as in regular expressions.

Inside the Ajax call body we will include the web action type (GET or POST), url (the path where the service is listening), the params parameters variable mentioned above, the data type for formatting the request and the user credentials for authorizing the call. Please note that this user must belong to the role mentioned in "User authorization" section, otherwise the server will throw a non-authorized exception.

We will include the success and error handlers as well. These are executed once the response is obtained and is ready to be processed. The response will be returned as a JSON object.
$.ajax({
  type: "GET",
  data: dataQS,
  dataType: "json",
  beforeSend: function(xhr) {
    xhr.setRequestHeader('Authorization', 'Basic ' + btoa("user" + ";" + "password"); },
  success: function (resp) {
    //handle the response accordingly
  },
  error: function (x, y, z) {
    //handle the response accordingly
  }
});

For simplicity’s sake we won’t be listing the response types for every web method. We encourage you to read the descriptions in the “Method list” section above and make your own calls for analyzing and processing the different responses accordingly

**Simple POST Call**

For POST calls we should apply JSON.stringify to the DTO object that we send to the server. Note that **contentType** is a mandatory field when making POST calls. If this attribute is not present, the call won’t work

```javascript
jQuery.support.cors = true;
var myDTO = {
  managerId : 412,
  folderGuid : '09016489-e9c3-45a5-a151-ca24840d6e04',
  userId : 30
};
$.ajax({
  type: "POST",
  data: JSON.stringify(myDTO),
  contentType: "application/json; charset=utf-8",
  beforeSend: function(xhr) {
    xhr.setRequestHeader('Authorization', 'Basic ' + btoa("user" + ";" + "password"); },
  },
  dataType: "json"
});
```

If POST calls are successfully executed, they will return an OK string value

**Passing dates as parameters**

Some methods take Date parameters, these must be in 'yyyyMMdd' format as shown below:

```javascript
jQuery.support.cors = true;
```
Passing lists as parameters

GetAttemptsByDateFiltered has several parameters: (int  managerModuleId, string roleFiltersId, string folderFiltersGuids, String startDate, String endDate). Although they appear as strings they actually represent lists. In order to be able to match a single parameter with a list we will use ‘$’ (dollar sign) as a separator. For example, if we want to call GetAttemptsByDateFiltered passing the list containing role ids [0,1,2] we will use ‘0$1$2’ as roleFiltersId parameter. For sending a folder Guids list we use $ as well as is shown in the example call below

```javascript
jQuery.support.cors = true;
var myParams = {
    managerModuleId : 412,
    roleFiltersId : '0$1$2',
    folderFiltersGuids : '4a608796-438c-4846-88ba-1b8ebf8c12b4$09016489-e9c3-45a5-a151-ca24840d6e04',
    startDate: 20120413,
    endDate: 20130520
};
$.ajax({
    type: "GET",
    data: myParams,
    contentType: "application/json;charset=utf-8",
    beforeSend: function (xhr) {
        xhr.setRequestHeader('Authorization', 'Basic ' + btoa("user" + ":" + "password"));
    },
    dataType: "json"
});
```
Method Specification

In the section below we will provide the method signature (without return type) and a brief explanation for some of their parameters. To avoid repetition we will not comment the same parameters that were already explained in previous methods.

GetLearnerReportTranscriptFile(int format, 
int userId, 
int learnerModuleId, 
int portalId, 
int managerModuleId, 
string folders, 
bool getNotEnrolled, 
string scoreDefinitionUrl)

Format ReportViewer = 0, PDF = 1, XLSX = 2, XLS = 3, PPTX = 4, RTF = 5, IMAGE = 6, MHTML = 7, DOCX = 8, CSV = 10

scoreDefinitionUrl Parameter in development. Will be used in the future. As of now set it as an empty string

GetLearnerReportScoreDetailsFile(int format, 
int userId, 
int learnerModuleId, 
int portalId, 
int managerModuleId, 
int attemptId, 
String LEId)

GetLearnerReportAttemptHistoryFile(int format, 
int userId, 
int learnerModuleId, 
int portalId, 
int managerModuleId, 
int attemptId, 
String LEId)

ProcessAccessCode(string code, 
string accessCodeLabel, 
int portalId, 
int tabId, 
int managerId, 
int userId, 
int logType)
accessCodeLabel Internal use only parameter, should be set as an empty string. Will be deprecated in the following release

logType The Type of call for the method, can take the following values:
1 (Class Submit), 2 (URL Submit), 3 (Bulk Role Assignment), 4 (Bulk Learner Import)

LearnerTranscriptGet(int userId,
    int managerModuleId,
    string searchCriteria)

searchCriteria Text that the LE must contain in order to be included in the results. Follows the same behavior as when using learner’s module search filed

LearnerTranscriptByFolderGet(int userId,
    int managerModuleId,
    string searchCriteria,
    Guid baseFolderGuid,
    int depth)

baseFolderGuid The base folder from where the transcript will be generated. (LEs from upper levels will be ignored)

depth The folder depth that the transcript will have (LEs from lower levels will be ignored)

GetLearningEvents(int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions)

ignoreConditions if true links are generated no matter if the LE has conditions or not, so for instance it would be set to true if a manger wants to check if the LEs are fine without going through the perquisites. If set to false, when conditions are not met the links that depend on perquisites won’t be completely generated

GetFilteredLearningEvents(
    int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
    int enrolledStatusInt,
    bool? isExpired,
    bool? isPublished,
    bool? isVisible,
    string searchCriteria)

attemptsOptionInt Best = 0, Last = 1, All = 2, LastCompleted = 3
enrolledStatus  None = 2, All = 3, EnrolledOrAttempted = 4, AnyEnrollment = 5, AttemptedNotEnrolled = 6

GetCatalog(int userId,
            int managerModuleId,
            int learnerModuleId,
            int LMSTabId,
            bool ignoreConditions)

GetFilteredCatalog(
    int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
    int enrolledStatusInt,
    bool? isExpired,
    bool? isPublished,
    bool? isVisible,
    string searchCriteria)

GetFilterCatalogSimplified(
    int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
    int enrolledStatusInt,
    bool? isExpired,
    bool? isPublished,
    bool? isVisible,
    string searchCriteria,
    bool simplified)

Simplified  Sets the simplified version, it will not load all the properties in the LE and will ignore Enrolled by, has access and attempt options filters. Runs faster than when set to false

GetFilteredCatalogSimplifiedByFolder(
    int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
int enrolledStatusInt,
bool? isExpired,
bool? isPublished,
bool? isVisible,
string searchCriteria,
bool simplified,
Guid baseFolderGuid,
int depth,
bool includeLRResources)

GetFilteredCatalogSimplifiedByFolderConditions(
    int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
    int enrolledStatusInt,
    bool? isExpired,
    bool? isPublished,
    bool? isVisible,
    string searchCriteria,
    bool simplified,
    Guid baseFolderGuid,
    int depth,
    bool includeLRResources,
    bool forTreedGrid)

forTreedGrid Set it as false. To be deprecated in the next release

GetCatalogForResources(int userId,
    int managerModuleId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt,
    bool? hasFolderAccess,
    int enrolledStatusInt,
    bool? isExpired,
    bool? isPublished,
    bool? isVisible,
    string searchCriteria,
    bool simplified,
    Guid baseFolderGuid,
    int depth)

GetLearningEventByLEId(
    Guid LEId,
    Guid folderGuid,
    int userId,
int learnerModuleId,
int LMSTabId,
bool ignoreConditions)

GetLearningEventByLEIdByAttemptsOption(
    Guid LEId,
    Guid folderGuid,
    int userId,
    int learnerModuleId,
    int LMSTabId,
    bool ignoreConditions,
    int attemptsOptionInt)

GetMinimalLEDataById(Guid LEId)

GetConditionByFolderIdAndLEId(Guid folderGuid, Guid LEId)

GetCondition(int managerModuleId,
    Guid folderGuid,
    Guid LEId,
    int userId,
    int attemptsOptionInt)

GetConditionByAttemptsOption(int managerModuleId,
    Guid folderGuid,
    Guid LEId,
    int userId,
    int attemptsOptionInt)

GetRoleAwardAssignedRolesByAttemptsOption(int userId
    Guid folderGuid,
    Guid LEId,
    int attemptsOptionInt)

GetRoleAwardAssignedRoles(int userId,
    Guid folderGuid,
    Guid LEId)

GetRoleAwardByLEId(Guid folderGuid, Guid LEId)

EnrollLE(Guid LEId,
    int managerId,
    Guid parentFolderGuid,
    int userId,
    )
EnrollFolder(int managerId,
            Guid folderGuid,
            int userId)

DropEnrollFolder(int managerId,
                Guid folderId,
                int userId,
                bool isArchived)

DropEnrollLE(Guid LEId,
              int managerId,
              Guid parentFolderGuid,
              int userId)

GetManagerByLauncher(int learnerModuleId)

GetAttemptById(int attemptId)

GetInteractionsByAttemptId(int attemptId)

GetAttempts(int portalId,
              Guid LEId,
              int userId)

GetAttemptsByDateFiltered(int managerModuleId,
                           string roleFiltersId,
                           string folderFiltersGuids,
                           String startDate,
                           String endDate)

GetAttemptsByDate(Guid LEId,
                   int userId,
                   String startDate,
                   String endDate)

GetAttemptsHistoryHeader(Guid LEId,
                          int userId)

GetLearningEventDetailsByLEId(Guid LEId,
                                Guid folderGuid,
                                int userId,
                                int learnerModuleId,
                                int LMSTabId,
                                bool ignoreConditions)

GetLearningEventDetailsByLEIdFiltered(Guid LEId,
                                   Guid folderGuid,
                                   int userId,
int learnerModuleId,
int LMSTabId,
bool ignoreConditions,
int attemptsoptionInt)

GetBaseFolderById(Guid folderGuid,
                   int managerModuleId)

HasLearnerFolderAccess(Guid folderGuid, int userId)

GetHTTPRedirect(int managerModuleId, int userId)