PeopleFluent Learning

Learning Management System API

Contents

About this Document

Introduction

The PeopleFluent LMS exposes its data via an Application Programming Interface (API). The API is HT TP-based, and makes use of widely-supported Web data formats such as XML, JSON, Atom and RSS. The API is intended to conform to the design principles of Representation State Transfer (REST).

The API is intended to support at least the following scenarios:

- **Integration with back end systems** such as Human Resource Management Systems;
- **Integration with other websites and applications**, such as portals, that are built on server-side technologies such as Java, the Microsoft .NET Framework, or PHP;
- **Access via browser-based technologies** such as asynchronous JavaScript with XML/JSON;
- **Access via desktop applications** (e.g. Adobe AIR, browser extensions); and
- **Access via mobile applications** (e.g. Android, Cocoa for iPhone).

Document Information

Revision Information	
Revision Date:	April 25, 2024
Revised Document Version Number:	2.0
Details of Revision:	Initial publication.

API Authentication

Authentication Modes

Each API function utilizes one of the three authentication modes described below. The API function reference indicates the appropriate mode for each specific API function.

User Authentication

When calling a function that uses user authentication, the caller must authenticate using a **PeopleFluent LMS user account**. In general, the result of calling a function that requires user authentication will depend on the permissions of the LMS user account that the caller uses to authenticate. The caller can authenticate in either of the two ways listed below.

- The caller can supply the user ID and password of the user account using **HTTP basic access authentication**. This technique is generally appropriate for callers that are not browser-based—for example, for server-to-server calls, or for desktop or mobile clients.
- The caller can include a standard LMS session cookie in the request. This technique is appropriate for browser-based callers (for example, asynchronous JavaScript with XML/ JSON) assuming that the user has already logged into the LMS via the browser.

System Authentication

When calling a function that uses system authentication, the caller must supply a **global system password** using **HTTP basic access authentication**. The value of the password is configured in the `WEB-INF/conf/ekp.properties` configuration file using the `authentication.key` property. In this case, the LMS ignores the value of the supplied user name. However, some client and server software will not forward the credentials if the user name is empty; for this reason, we recommend supplying an arbitrary non-empty value for the user name.

Not Required

Some API functions provide access to public data that does not require authentication. Note that the function might still return additional information if the user is authenticated (using either of the two techniques described for user authentication above); however, the function is expected to return some useful result even if the user is not authenticated.

⚠ When using HTTP basic access authentication, credentials are passed in what is essentially clear text. Therefore, the caller should always use SSL/TLS when sending credentials using basic access authentication.

Delegated Authentication in API

Overview of Delegated Authentication

Delegated authentication is a technique whereby one application relies on a second application to authenticate the end user, thereby avoiding the need for the user to log in to each application separately. Delegated authentication is particularly useful when multiple websites or applications are combined to provide a common service.

An authentication service is a site that authenticates the end user on behalf of one or more other sites. A relying party is a site that relies on an authentication service to authenticate the end user.

PeopleFluent LMS can act as either an authentication service or a relying party in a delegated authentication relationship.

As an example, consider a portal site that displays a list of the end user's active enrollments (obtained using the PeopleFluent LMS API), but then redirects the user to the LMS to handle tracking when the course is actually launched. Delegated authentication could be used to avoid the need for the user to log into both the portal and the LMS. If the LMS is acting as the authentication service, the user will use a LMS login to access both sites. If the LMS is acting as the relying party, the user will log in through the portal site to access both sites.

Technical Approach

The basic technical approach used is the same whether the LMS is acting as an authentication service or a relying party. The steps are as follows.

1. The end user attempts to access a protected resource within the relying party. The relying party determines that the user is not yet authenticated.
2. The relying party generates a secure random one-time salt value, and stores it in the user's session.
3. The relying party redirects the user to a distinguished URL within the authentication service (the authentication service URL), including the salt as the value of a query-string parameter named salt. Optionally, the relying part may also provide a query string parameter namedcallback, whose (URL-encoded) value is a URL that the authentication service should redirect the user to after authenticating the user.
4. If the user is not currently authenticated with the authentication service, the authentication service prompts the user to log in. If the user is already authenticated with the authentication service, this step is skipped.

5. The authentication service redirects the user back to the relying party, using the value of the callback parameter if provided in step 3, or a default URL otherwise. (N.B. The authentication service should check the callback URL against a whitelist of approved sites, in order to prevent arbitrary websites from discovering the user's identity.) The authentication service appends two query string parameters to this URL:
    - `userId`, which is the identifier of the user as established by the authentication service
    - sig, a hex-encoded cryptographic hash of the string formed by concatenating the user's identifier, a shared secret, and the salt value. (Regardless of whether the LMS is acting as authentication service or relying party, the shared secret is configured using the `authentication.key` property in the `WEB-INF/conf/ekp.properties` configuration file. The hashing algorithm is configured using the `authentication.digestAlgorithm` property; the default is MD5.)
6. The relying party performs its own computation of the cryptographic hash, and compares the result with the value provided by the query string parameter. Authentication succeeds if and only if the values match.

LMS as an Authentication Service

In order to use the LMS as an authentication service in a delegated authentication relationship, the relying party should use an authentication service URL formed by appending servlet/ekp/authenticate to the base URL of the LMS site. For example, if the site is at http://www.example.com/ekp/, then the authentication service URL will be http://www.example.com/ekp/servlet/ekp/authenticate.

The default callback URL for the relying party can be configured using the authentication.callback.url property in the WEB-INF/conf/ekp.properties configuration file, as in the example below.

```
authentication.callback.url=http://www.example.com/authcallback.php
```

A specific relying party can override this value by passing a callback URL as the value of a query string parameter named callback, provided that the relying party has been designated as trusted as described here. Note that, as with all parameter values, the value of the callback parameter must be URL-encoded, as in the example below (which also includes the required nonce parameter).

```
servlet/ekp/authenticate?nonce=a6f8a4675a98a54d63d80c0c3b2a4e6c&callback=htt
p%3A%2F%2Fwww.example.com%2Fauthcallback.php
```

As an example, an application might be developed that displays a highly-customized view of a learner's training plan including his or her current progress. The application could use delegated authentication to establish the user's identity, and then call the LMS' trainingHistoryXml API function to determine the user's progress in the various courses in their plan so that the training plan can be presented appropriately.

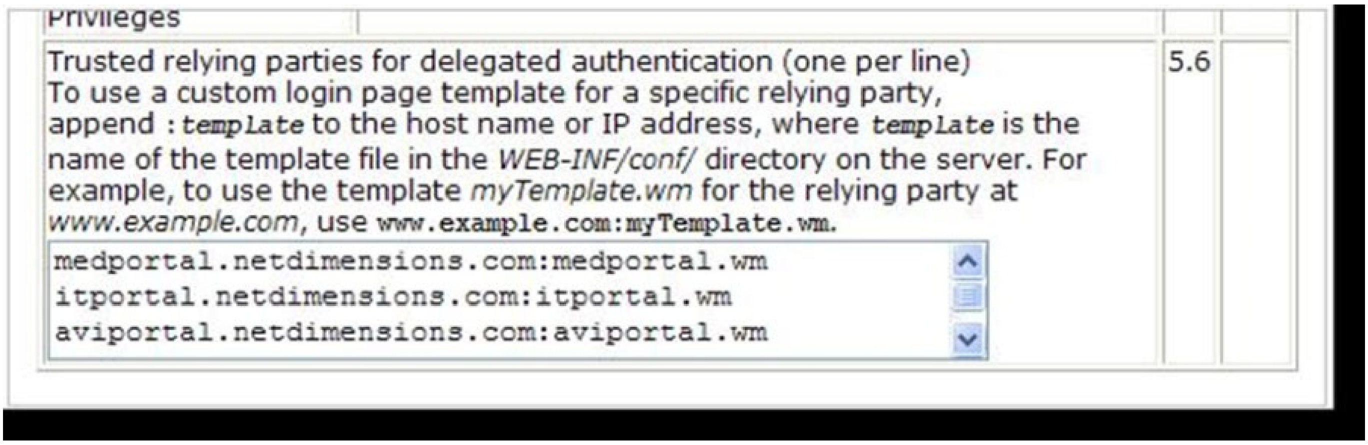
LMS as an Authentication Service for Multiple Relying Parties

Since each relying party may pass its own callback URL with an authentication request, multiple relying parties can be used with a single LMS instance. For example, multiple portals could be backed by one instance.

Configuring Relying Parties for Delegated Authentication

Using a technique known as delegated authentication, a third-party website such as a portal can rely on the LMS to authenticate the end user thereby avoiding the need for a separate login.

However, in order to protect the privacy of end users, the LMS does not allow just any website to discover the user's LMS identity. Instead, an administrator must explicitly configure each trusted site (known as a relying party) that is permitted to authenticate users using the LMS. You can do this using the text area labeled **Trusted relying parties for delegated authentication**, which is under the User category of the System Configuration page (**Manage > System Administration Manager> System Settings > System Configuration**).



Privileges

Trusted relying parties for delegated authentication (one per line)
To use a custom login page template for a specific relying party, append `:template` to the host name or IP address, where `template` is the name of the template file in the *WEB-INF/conf/* directory on the server. For example, to use the template *myTemplate.wm* for the relying party at *www.example.com*, use `www.example.com:myTemplate.wm`.

5.6

```
medportal.netdimensions.com:medportal.wm
itportal.netdimensions.com:itportal.wm
aviportal.netdimensions.com:aviportal.wm
```

You will need to enter the hostname or IP address of the relying party exactly as it appears in the callback URL. For example, if the callback URL is

`http://www.example.com/portal/auth_callback.php`

then you should enter the relying party as www.example.com. If you need to configure more than one relying party (multiple portals using a single instance for example), you should enter each one on a separate line. As always, click the Save button when you are done.

Custom Login Pages

⚠ The information in this section applies only if you are using the LMS as an authentication service in a delegated authentication relationship. If you are simply attempting to configure multiple login pages for a single instance, please refer to the documentation here.

When the relying party needs to authenticate the end user, the end user will be shown a login page with a form in which he or she can enter his or her user name and password. By default this will be the standard login page. However, it is possible to use a custom login page; moreover, the custom login page can be different for each relying party.

Follow the steps below to configure a custom login page for a specific relying party.

1. Create a login page template in the usual way, giving it a distinctive file name such as `loginportal.wm`.
2. Copy the login page template to the `WEB-INF/conf/` directory of your LMS installation. (Note that this is the only step that requires direct access to the file system of the LMS server.)
3. In the Trusted relying parties for delegated authentication text area, append a colon to the appropriate relying party, followed by the file name of the login page template, e.g. `www.example.com:loginportal.wm`. Then click the Save button.

Note that the custom login page will be used only when authentication is requested directly from the portal by redirecting the user to servlet/ekp/authenticate; it will not be used if the portal site redirects the user to another transaction that in turn requires authentication. This page discusses a couple of workarounds that can be used if this behavior is not desired.

Example:

I have configured a custom login page for my portal but users still see the standard login page when clicking an Enroll button. How can I fix this?

The information on this page assumes that you have configured a custom login page for a portal that uses delegated authentication.

A custom login page configured in this way will be used when authentication is requested directly from the portal, by redirecting the user to servlet/ekp/authenticate.

However, the sample portals redirect the user to the enrollment transaction when the user clicks an Enroll button; in turn, the enrollment transaction invokes the login process if it detects that the user is not already logged in. Because the login request came from a transaction rather than directly from the portal, the login process does not "know" that the request originated from the portal. Hence, the standard login page is displayed rather than the portal-specific one. A similar situation can arise whenever a portal redirects the user to a transaction.

There are a couple of ways to avoid this behavior, specifically:

- Ensure the user gets logged in before the redirect occurs; or
- Use the api/enroll-user function to perform the enrollment, thus completely avoiding the need to redirect the user to the LMS screens.

The first method simply involves ensuring that the user visits at least one portal page that requires authentication before the redirect occurs. (For example, with the standard PHP portal templates you can ensure that a page requires authentication by calling the `force_sign_in()` function at the top of the page.) One way to implement this would be to force authentication for the course description page on which the Enroll button appears. A slightly more sophisticated approach would be to create an intermediate redirect page within the portal that requires authentication and then simply redirects the user to the enrollment transaction. The Enroll button would then link to this intermediate page instead of linking directly to the enrollment transaction.

The second approach completely avoids the need for the user to see any LMS pages, which might be advantageous if you need complete control over the appearance and navigation of your portal. Using this approach, you would create a page in your portal that calls LMS' api/ enroll-user API function and then displays an appropriate response to the user. You would then modify the Enroll button so that it links to this new portal page instead of linking to the enrollment transaction.

LMS as a Relying Party

In order to have the LMS acting as a relying party in a delegated authentication relationship, configure the `authentication.service.url` property in the `WEB-INF/conf/ekp.properties` configuration file such that its value is the URL of the authentication service, as in the example below.

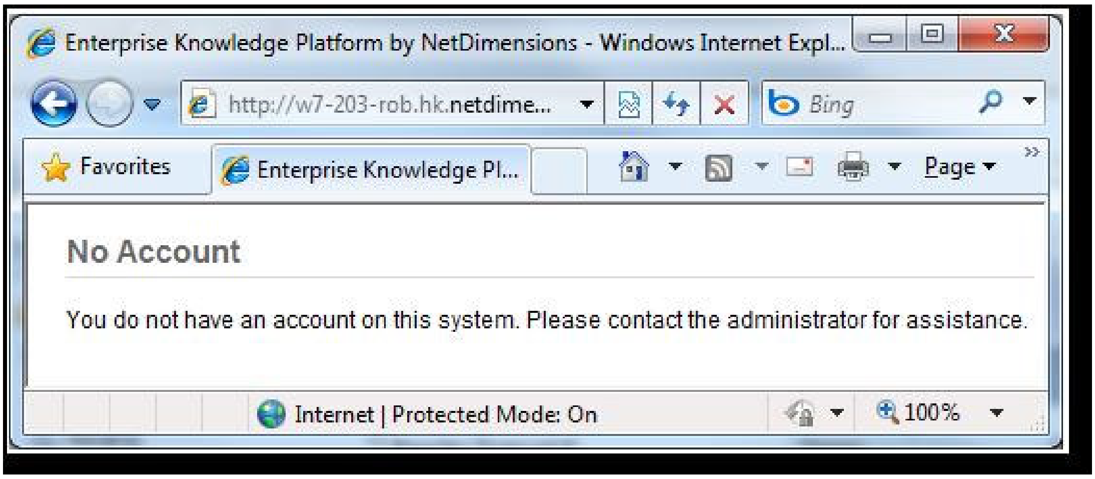
```
authentication.service.url=http://portal.example.com/login.asp
```

The callback URL to be used by the authentication service can be found by appending servlet/ ekp/ authenticationTokenVerifier to the base URL of the LMS site. For example, if the site is at

`http://www.example.com/ekp/` , then the callback URL will be `http://www.example.com/ekp/servlet/ekp/authenticationTokenVerifier`.

As an example, an organization might provide an intranet site for which each member of the organization has a username and password. The site could provide a link that would enable a user to navigate to the LMS and access a specified list of courses. When the user clicks the link, the site calls the LMS' users `Csv API` function to ensure that there is a user account for the user, followed by the `enrollmentHandler API` function to ensure that the user is enrolled in the specified courses. Finally, the site redirects the user to the LMS, which uses delegated authentication to establish the user's identity and display his or her active enrollments.

If the response from the authentication service is valid but there's no user account with the corresponding ID, the user will see an error page similar to the one shown below.

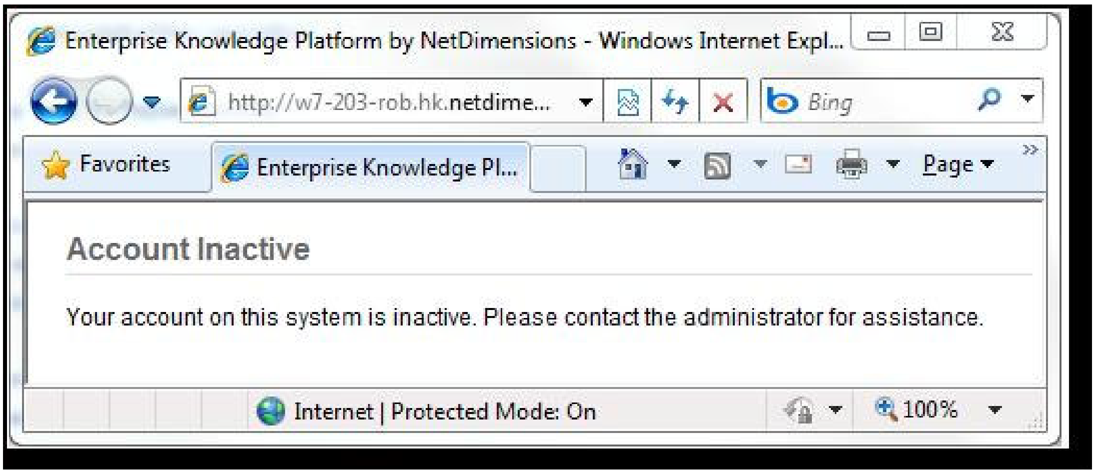


Enterprise Knowledge Platform by NetDimensions - Windows Internet Expl...
http://w7-203-rob.hk.netdime...
Bing
Favorites
Enterprise Knowledge Pl...
Page
Internet | Protected Mode: On
100%

No Account

You do not have an account on this system. Please contact the administrator for assistance.

If there's an user account with the corresponding ID but the account is not active, the user will see an error page similar to the one shown below.



Enterprise Knowledge Platform by NetDimensions - Windows Internet Expl...
http://w7-203-rob.hk.netdime...
Bing
Favorites
Enterprise Knowledge Pl...
Page
Internet | Protected Mode: On
100%

Account Inactive

Your account on this system is inactive. Please contact the administrator for assistance.

If you expect that the authentication service will regularly respond with user IDs for which there is no corresponding active user account, you might be able to improve the user experience by having the authentication service make an API call to check for an active account, and redirect to the LMS only if such an account exists.

This blog post describes how the LMS could be configured to communicate with an identity provider implemented using Java.

LMS as a Relying Party with Multiple Authentication Services

In some cases, different user groups might use different methods of authentication. However, the LMS needs to initiate the authentication process precisely when the user is not already authenticated, hence cannot use information about the user to select between multiple alternative authentication services. Therefore, only a single authentication service (that is, a single value for `authentication.service.url`) can be configured.

A workaround is to create an intermediate page (sometimes called a discovery service) that presents the user with a choice of possible authentication methods, and then sends the user to the appropriate authentication service based on their selection; the authentication service can then send its response to the LMS directly or via the intermediate page `authentication.service.url` is then configured with the address of the intermediate page.

Hostname-based Selection of Authentication Services

The LMS only permits you to configure a single external authentication service for use with delegated authentication. This is because an unauthenticated user can arrive at a protected page via an unexpected route, for example due to a session timeout or following an external link, and the LMS needs an unambiguous way to authenticate such users.

However, various workarounds are possible to simulate multiple authentication services. Each relies on there being some mechanism by which we can infer the appropriate authentication service for an unauthenticated user. This page describes one particular mechanism that assumes that a single instance can be accessed using any one of multiple hostnames, and uses the hostname to infer the appropriate authentication service for the unauthenticated user. (Other variations are possible, such as inferring the appropriate authentication service based on the user's IP address.)

⚠ In the approach described here, the same secret key is used by all authentication services. This means that the LMS cannot distinguish between responses from different services—any one of the services will be able to issue a response corresponding with any user account. Therefore this approach should only be used if all of the authentication services are completely trusted.

Apache Configuration

The basic idea is that, for each hostname, we create a similarly-named virtual resource (/signon in this example) that redirects to the appropriate authentication service. Since we are using multiple hostnames we will in any case need to use the Apache VirtualHost directive; we just need to add an appropriate Redirect directive inside each VirtualHost directive.

```
<VirtualHost *>
 ServerName ekp.organization1.example.com
 Redirect /signon
http://portal.organization1.example.com/cgi/authenticationservice.pl


 # Other VirtualHost configuration
</VirtualHost>


<VirtualHost *>
 ServerName ekp.organization2.example.org
 Redirect /signon
http://portal.organization2.example.org/authenticationservice.aspx


 # Other VirtualHost configuration
</VirtualHost>
```

With the above configuration, a user accessing http://ekp.organization1.example.com/signon?salt= {salt-value } will be redirected to

http://portal.organization1.example.com/cgi/authenticationservice.pl?salt= {salt-value} , while a user accessing

http://ekp.organization2.example.org/signon?salt= {salt-value }

will be redirected to

http://portal.organization2.example.org/authenticationservice.aspx?salt= {salt-value} .

Note that the `/signonURL` path need not correspond with a physical file or folder — it is simply a virtual resource that performs the redirect.

LMS Configuration

With Apache appropriately configured, we can now configure the `authentication.service.url` property in `WEB-INF/conf/ekp.properties`. The key point here is that we specify a relative, rather than absolute, URL.

```
authentication.service.url=/signon
```

Now, if an unauthenticated user attempts to access a protected resource using the hostname ekp.orga nization1.example.com, he or she will be redirected to

`http://ekp.organization1.example.com/signon?salt=\{salt-value}`, and hence to `http://portal.organization1.example.com/cgi/authenticationservice.pl?s alt={salt-value}`.

An unauthenticated user attempting to access the same protected resource using the hostname `ekp.organization2.example.org` would instead be redirected to

`http://ekp.organization2.example.org/signon?salt=\{salt-value}`, and hence to `http://portal.organization2.example.org/authenticationservice.aspx?salt={salt-value}`.

Thus, the authentication service to which the user is ultimately directed depends on which of several possible hostnames he or she used to access the LMS.

Note that this mechanism depends on the `salt` query string parameter being preserved during the second redirect. Fortunately, the `ApacheRedirect` directive matches a prefix of the URL path, and preserves the remainder of the path, including the query string.

Introducing API Function Reference

This document describes the most commonly used API functions. To see a complete, current list of functions, access the LMS' API Explorer tool by appending api/ to the base URL of your LMS site. For example, if your site were located at `http://www.example.com/ekp/`, you could access the API Explorer at `http://www.example.com/ekp/api/`. The API Explorer also enables you learn about the API by calling its functions interactively.

URLs are specified relative to the base URL of the LMS site. For example, if the base URL of the site is `http://www.example.com/ekp/`, then the absolute URL of the publicNews API function (for which the URL is specified asapi/publicNews) will be `http://www.example.com/ekp/api/pub licNews`.

Parameter values should be converted to UTF-8 and URL-encoded.

Actions by Category

News Actions

Learning Record and Enrollment Actions

Learning Module Actions

Career Development Actions

Community Actions

User Actions

Exam Actions

Report Actions

System Actions

News Actions

publicNews

Function	Returns a feed of public news articles. (That is, news articles that are configured to display on the log-in page.)
Authentication	Not supported
URL	`/api/publicNews`
Method	`GET`
Response Content Type	`application/rss+xml, application/atom+xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	atom, rssteasers, rssfull	`rssteasers` for RSS 2.0 with teasers, `rssfull` for RSS 2.0 with full text, or `atom` for Atom 1.0.
`categoryId`	text	Optional. If specified, only articles in the corresponding category are returned.

* indicates required parameter

userNews

Function	Returns a feed of news articles for a user.
Authentication	User Authentication (required)
URL	`/api/userNews`
Method	`GET`

Response Content Type	`application/rss+xml, application/atom+xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	atom, rssteasers, rssfull	`rssteasers` for RSS 2.0 with teasers, `rssfull` for RSS 2.0 with full text, or `atom` for Atom 1.0.
`categoryId`	text	Optional. If specified, only articles in the corresponding category are returned.
`ASSIGNMENTID`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

article

Function	Returns an article as an Atom 1.0 entry.
Authentication	Not supported
URL	`/api/article`
Method	`GET`
Response Content Type	`application/atom+xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	The ID of an article.

* indicates required parameter

articles

Function	Returns information about news articles.
Authentication	User Authentication (required)
URL	`/api/articles`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Search keywords.
`publishedAfter`	datetime	Only articles entered after this date are returned (HTML5/ISO format, e.g., 2009-11-13T20:00Z)
`publishedBefore`	datetime	Only articles entered before this date are returned (HTML5/ISO format, e.g., 2009-11-13T20:00Z)
`categoryId`	text	If specified, only articles in the corresponding category are returned
`includeExpired`	true or false	Expired articles are excluded from the results unless this is set

`max`	number	The maximum number of articles to return. If unspecified, all matching articles are returned.
`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

Learning Record and Enrollment Actions

enrollments

Function	Returns the active (incomplete) enrollments for a user.
Authentication	User Authentication (required)
URL	`/api/articles`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

programModuleEnroller

Function	Adds a module session enrollment to a program session enrollment.
Authentication	User Authentication (required)

URL	`/api/programModuleEnroller`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`programSessionEnrollmentId*`	text	the program session enrollment ID.
`moduleId*`	text	the module ID.
`moduleSessionId`	text	the module session ID.
`moduleSessionCode`	text	the module session code. At least one of {moduleSessionId, moduleSessionCode} must be specified.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

records

Function	Returns the training records for a user.
Authentication	User Authentication (required)
URL	`/api/records`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
assignmentId	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
onBehalfOf	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

record

Function	Returns a training record.
Authentication	User Authentication (required)
URL	`/api/record`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the identifier for a training record.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalTrainingRequests

Function	Return external training requests pending for approval.
Authentication	User Authentication (required)
URL	`/api/externalTrainingRequests`

Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalTrainingRecords

Function	Returns external training records.
Authentication	User Authentication (required)
URL	`/api/externalTrainingRecords`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalTrainingRecord

Function	Returns an external training record.
Authentication	User Authentication (required)
URL	`/api/externalTrainingRecord`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the external training record.
`format*`	json, xml	the response format, `json` or `xml`.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalRecordAdder

Function	Adds an external training record.
Authentication	User Authentication (required)
URL	`/api/externalRecordAdder`
Method	`POST`
Request Content Type	`multipart/form-data`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`title*`	text	the title (mandatory).

`type*`	learningProgram, classroom, onlineModule, workshopSeminar	the course type code.
`subject`	text	the subject ID.
`startDate*`	datetime	the start date as an HTML5 date (mandatory).
`endDate*`	datetime	the end date as an HTML5 date (mandatory).
`venue`	text	the venue.
`language*`	en, en_GB, en_CA, zh_HK, zh_CN, zh_TW, fr_FR, de_DE, ja_JP, es_ES, es_CO, nl_NL, el_GR, ar_SA, vi_VN, pt_BR, ms_MY, pl_PL, ko_KR, tl_PH, in_ID, iw_IL, cs_CZ, hu_HU, pt_PT, ru_RU, sk_SK, th_TH, it_IT, en_US, fi_FI, da_DK, sr_RS, et_EE, nl_NL_informal, bs_BA, sl_SI, ar_AE, bg_BG, fr_CA, nb_NO, ro_RO, es_MX, sv_SE, tr_TR	the language code.
`durationAmount`	text	the duration amount (numeric).
`durationUnit*`	hour, day, week, month, year	one of hour, day, week, month, year

`costAmount`	text	the cost amount (numeric).
`costCurrency*`	AED, AFA, ALL, AMD, ANG, AOA, ARS, AUD, AWG, AZM, BAM, BBD, BDT, BGN, BHD, BIF, BMD, BND, BOB, BRL, BSD, BTN, BWP, BYN, BYR, BZD, CAD, CDF, CHF, CLP, CNY, COP, CRC, CSD, CUP, CVE, CYP, CZK, DJF, DKK, DOP, DZD, EEK, EGP, ERN, ETB, EUR, FJD, FKP, GBP, GEL, GHC, GIP, GMD, GNF, GTQ, GYD, HKD, HNL, HRK, HTG, HUF, IDR, ILS, INR, IQD, IRR, ISK, JMD, JOD, JPY, KES, KGS, KHR, KMF, KPW, KRW, KWD, KYD, KZT, LAK, LBP, LKR, LRD, LSL, LTL, LVL, LYD, MAD, MDL, MGA, MKD, MMK, MNT, MOP, MRO, MTL, MUR, MVR, MWK, MXN, MYR, MZM, NAD, NGN, NIO, NOK, NPR, NZD, OMR, PAB, PEN, PGK,	the cost currency code (e.g. USD, HKD).

	PHP, PKR, PLN, PYG, QAR, ROL, RON, RSD, RUB, RWF, SAR, SBD, SCR, SDD, SDG, SEK, SGD, SHP, SIT, SKK, SLL, SOS, SRD, STD, SVC, SYP, SZL, THB, TJS, TMM, TND, TOP, TRY, TTD, TWD, TZS, UAH, UGX, USD, UYU, UZS, VEB, VEF, VND, VUV, WST, XAF, XAG, XAU, XCD, XDR, XOF, XPD, XPF, XPT, YER, ZAR, ZMK, ZWD	
`grade`	text	the grade.
`score`	text	the score.
`vendor`	text	vendor information.
`comments`	text	comments.
`attachment`	file	the attachment (file).
`status*`	openForEditing, pendingApproval, completed, cancelled	one of openForEditing, pendingApproval, completed, cancelled.
`format*`	json, xml	the response format, jso n or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalRecordUpdater

Function	Updates an external training record.
Authentication	User Authentication (required)
URL	`/api/externalRecordUpdater`
Method	`POST`
Request Content Type	`multipart/form-data`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the external training record.
`title*`	text	the title (mandatory).
`type*`	learningProgram, classroom, onlineModule, workshopSeminar	the course type code.
`subject`	text	the subject ID.

`startDate*`	datetime	the start date as an HTML5 date (mandatory).
`endDate*`	datetime	the end date as an HTML5 date (mandatory).
`venue`	text	the venue.
`language*`	en, en_GB, en_CA, zh_HK, zh_CN, zh_TW, fr_FR, de_DE, ja_JP, es_ES, es_CO, nl_NL, el_GR, ar_SA, vi_VN, pt_BR, ms_MY, pl_PL, ko_KR, tl_PH, in_ID, iw_IL, cs_CZ, hu_HU, pt_PT, ru_RU, sk_SK, th_TH, it_IT, en_US, fi_FI, da_DK, sr_RS, et_EE, nl_NL_informal, bs_BA, sl_SI, ar_AE, bg_BG, fr_CA, nb_NO, ro_RO, es_MX, sv_SE, tr_TR	the language code.
`durationAmount`	text	the duration amount (numeric).
`durationUnit*`	hour, day, week, month, year	one of hour, day, week, month, year
`costAmount`	text	the cost amount (numeric).
`costCurrency*`	AED, AFA, ALL, AMD, ANG, AOA, ARS, AUD, AWG, AZM, BAM, BBD,	the cost currency code (e.g. USD, HKD).

	BDT, BGN, BHD, BIF, BMD, BND, BOB, BRL, BSD, BTN, BWP, BYN, BYR, BZD, CAD, CDF, CHF, CLP, CNY, COP, CRC, CSD, CUP, CVE, CYP, CZK, DJF, DKK, DOP, DZD, EEK, EGP, ERN, ETB, EUR, FJD, FKP, GBP, GEL, GHC, GIP, GMD, GNF, GTQ, GYD, HKD, HNL, HRK, HTG, HUF, IDR, ILS, INR, IQD, IRR, ISK, JMD, JOD, JPY, KES, KGS, KHR, KMF, KPW, KRW, KWD, KYD, KZT, LAK, LBP, LKR, LRD, LSL, LTL, LVL, LYD, MAD, MDL, MGA, MKD, MMK, MNT, MOP, MRO, MTL, MUR, MVR, MWK, MXN, MYR, MZM, NAD, NGN, NIO, NOK, NPR, NZD, OMR, PAB, PEN, PGK, PHP, PKR, PLN, PYG, QAR, ROL, RON, RSD, RUB, RWF, SAR, SBD, SCR, SDD, SDG, SEK, SGD, SHP,	

	SIT, SKK, SLL, SOS, SRD, STD, SVC, SYP, SZL, THB, TJS, TMM, TND, TOP, TRY, TTD, TWD, TZS, UAH, UGX, USD, UYU, UZS, VEB, VEF, VND, VUV, WST, XAF, XAG, XAU, XCD, XDR, XOF, XPD, XPF, XPT, YER, ZAR, ZMK, ZWD	
`grade`	text	the grade.
`score`	text	the score.
`vendor`	text	vendor information.
`comments`	text	comments.
`attachment`	file	the attachment (file).
`status*`	openForEditing, pendingApproval, completed, cancelled	one of openForEditing, pendingApproval, completed, cancelled.
`format*`	json, xml	the response format, jso n or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

externalRecordDeleter

Function	Deletes an external training record.
Authentication	User Authentication (required)
URL	`/api/externalRecordDeleter`
Method	`POST`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the external training record.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

learningPath

Function	Returns the learning path for a user.

Authentication	User Authentication (required)
URL	`/api/learningpath`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

enrollUser

Function	Enroll a user into a learning module/program. If successful, it returns the corresponding enrollment. Otherwise, it returns error code/message.
Authentication	User Authentication (required)
URL	`/api/enrollUser`
Method	`POST`

Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`learning-id*`	text	The ID of the Learning Module/Program to Enroll Into
`session-code`	text	The Code (or ID) of the Specific Session within the Learning Module/Program to Enroll Into
`reason`	text	reason for enrolling.
`preferredLocation`	text	preferred location.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

enrollmentRequests

Function	Returns enrollment requests for a specific user.
Authentication	User Authentication (required)

URL	`/api/enrollmentRequests`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`inbox`	true or false	`true` to return only requests that are awaiting a response from the user; otherwise, returns all requests regardless of status.
`since`	datetime	If specified, only requests submitted after this date will be returned.
`filter`	text	Filter text.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

enrollmentRequestApprover

Function	Approves an enrollment request. Possible status values for the response are: `ok`, `waitlisted`, `not-successful`, `inactive-step`, `invalid-approver`, `no-such-request`, `no-such-step`, `payment-required`.
Authentication	User Authentication (required)
URL	`/api/enrollmentRequestApprover`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json`, `application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`request-id*`	text	Identifier of the enrollment request to be approved.
`step-id*`	text	Identifier of the step in the approval process for which the approval is to be issued.
`comments`	text	Additional Comments to be Passed to the Next Approver
`cost-center`	text	Charged Cost center
`email`	text	E-mail Address to which Notification Will Be Sent
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

enrollmentRequestDenier

Function	Denies an enrollment request. Possible status values for the response are: `ok`, `waitlisted`, `not-successful`, `inactive-step`, `invalid-approver`, `n o-such-request`, `no-such-step`, `payment-required`.
Authentication	User Authentication (required)
URL	`/api/enrollmentRequestDenier`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`request-id*`	text	Identifier of the enrollment request to be denied.
`step-id*`	text	Identifier of the step in the approval process for which the denial is to be issued.
`comments`	text	Additional Comments to be Returned to the Requester

`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

taskApproval

Function	Retrieve all task approvals
Authentication	User Authentication (required)
URL	`/api/taskApproval`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`status*`	all, pending, approved	"all" to return both pending and approved requests, "pending" to return pending requests only, "approved" to return approved requests only.
`format*`	json, xml	the response format, json or xml.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

taskApprover

Authentication	User Authentication (required)
URL	`/api/taskApprover`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`taskId*`	text	Identifier for the task (enrollment).
`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

contentPackage

Function	Returns a content package, typically for offline use. The user on whose behalf the content package is being returned must be enrolled in the corresponding course revision, and the revision must permit downloads.
Authentication	User Authentication (required)
URL	`/api/contentPackage`
Method	`GET`
Response Content Type	`application/zip`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`enrollmentId*`	text	The identifier for an enrollment, which must belong to the user on whose behalf the content package is being returned.
`pdfToHtml`	true or false	if true and the package contains a single PDF document, the document will be rendered as HTML
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

launchTracker

Function	Records a course launch in the context of an enrollment, which must belong to the user on whose behalf the launch is being recorded. The enrolled course revision must permit downloads.
Authentication	User Authentication (required)
URL	`/api/launchTracker`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`enrollmentId*`	text	the identifier for an enrollment, which must belong to the user on whose behalf the launch is being recorded.
`datetime*`	datetime	the date and time of the course launch.
`allowNewAttempt`	true or false	allowing the course to create new attempts if previous attempt has been completed
`format*`	json, xml	the response format, json or xml.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
onBehalfOf	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

hacpInfo

Function	Returns information that can be used to communicate on behalf of a content item using the HTTP CMI AICC Protocol (HACP). The user on whose behalf the HACP information is being returned must be enrolled in the corresponding course revision, and the revision must permit downloads. In this context, a content item refers to a SCO or asset in a SCORM content package, or an assignable unit in an AICC course package. Currently, the session IDs returned by this function do not expire. However, this behavior is subject to change and should not be relied upon. We recommend calling this function to obtain a current session ID immediately before initiating HACP communication.
Authentication	User Authentication (required)
URL	`/api/hacpInfo`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`enrollmentId*`	text	The the identifier for an enrollment, which must belong to the user on whose behalf the HACP information, is being returned.
`itemId*`	text	The identifier for an item in the enrolled course revision. For a SCORM course, this would correspond with the `identifier` attribute of an item element in `imsmanifest.xml`. For an AICC course, it would correspond with the ID of an assignable unit as specified in the .au file.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
onBehalfOf	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

overallStatusChanger

Function	Updates the overall status of an enrollment.
Authentication	User Authentication (required)
URL	`/api/overallStatusChanger`
Method	`POST`

Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`enrollmentId*`	text	the identifier of the enrollment for which the status is to be updated.
`status*`	unknown, waitlisted, enrolled, incomplete, completed, withdrawn, cancelled, pendingApproval, shared, administrativeEntry, resource, approvalDenied, attended, passed, failed, noShow, deadlineExpired, sessionSelectionNeeded, caCurrentLearnerSecondAttempt, caCurrentLearner, caCurrentLateEnrollment, waiverExempt, withdrawValidReason, withdrawInvalidReason, excused, droppedFromWaitlist, deactivated, withdrawAccountClosed, pendingSession	the new status.
`simpleResponse`	true or false	true to return only the affected enrollment; otherwise, returns all visible enrollments of the affected module.

`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

trainingHistoryXml

Function	Returns training records for the specified user (or for all users if user ID is not specified) in XML format
Authentication	System authentication
URL	`/api/trainingHistoryXml`
Method	`GET`
Response Content Type	`application/xml`
Response Schema	`xsd/trainingHistory200510.xsd`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`userId`	text	Specifies the user whose training records will be returned
`organizationId`	text	Specify the organization ID of the users whose training records will be returned. It includes users belonging to descendant organizations.
`lastUpdatedStart`	datetime	Start date filter to be applied on last updated date. If specified, it will return records added or updated on or after this date. [format: yyyy-MM-dd'T'HH:mm'Z']
`lastUpdatedEnd`	datetime	End date filter to be applied on last updated date. If specified, it will return records added or updated on or before this date. [format: yyyy-MM-dd'T'HH:mm'Z']
`moduleIds`	text	comma-separated list of IDs of modules for which to return records, or empty.
`includeOlder`	true or false	if true, all records will be returned for each module/assignment combination, rather than just the latest.
`fields`	text	field selector, e.g. field1(child1,child2),field2.

* indicates required parameter

completedTrainingRecordsXml

Function	Returns training records for completed courses for the specified user (or for all users if user ID is not specified) in XML format
Authentication	System authentication
URL	`/api/completedTrainingRecordsXml`

Method	`GET`
Response Content Type	`application/xml`
Response Schema	`xsd/trainingHistory200510.xsd`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`userId`	text	Specifies the user whose training records will be returned
`organizationId`	text	Specify the organization ID of the users whose training records will be returned. It includes users belonging to descendant organizations.
`lastUpdatedStart`	datetime	Start date filter to be applied on last updated date. If specified, it will return records added or updated on or after this date. [format: yyyy-MM-dd'T'HH:mm'Z']
`lastUpdatedEnd`	datetime	End date filter to be applied on last updated date. If specified, it will return records added or updated on or before this date. [format: yyyy-MM-dd'T'HH:mm'Z']
`moduleIds`	text	comma-separated list of IDs of modules for which to return records, or empty.
`includeOlder`	true or false	desc.api.completedTrainingRecordsXml.includeOlder
`fields`	text	field selector, e.g. field1(child1,child2),field2.

* indicates required parameter

supervisorAssessments

Function	List of supervisor assessments
Authentication	User Authentication (required)
URL	`/api/supervisorAssessments`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

supervisorAssessment

Function	API function to output the Supervisor Assessment assigned to an On-The-Job Training course
Authentication	User Authentication (required)
URL	`/api/supervisorAssessment`

Method	`GET`
Response Content Type	`application/zip`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the identifier for the enrollment for which the supervisor assessment is to be performed.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

withdrawalRequests

Function	Returns withdrawal requests for a specific user.
Authentication	User Authentication (required)
URL	`/api/withdrawalRequests`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`inbox`	true or false	true to return only requests that are awaiting a response from the user; otherwise, returns all requests regardless of status.
`since`	datetime	If specified, only requests submitted after this date will be returned.
`filter`	text	Filter text.
`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

withdrawalRequestApprover

Function	Approves a withdrawal request.
Authentication	User Authentication (required)
URL	`/api/withdrawalRequestApprover`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`

Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`request-id*`	text	ID of the transcript to which a pending withdrawal request is associated with.
`comments`	`text`	`Additional comments.`
`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

withdrawalRequestDenier

Function	Deny a withdrawal request.
Authentication	User Authentication (required)
URL	`/api/withdrawalRequestDenier`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`

Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`request-id*`	text	ID of the transcript to which a pending withdrawal request is associated with.
`comments`	text	Additional comments.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

participantSubstituter

Function	Substitutes a participant in a session with another user (assignment). It is equivalent to the Substitute Participant menu item on the Catalog Editor's Participants page. If targetAssignmentId is specified then targetUserId is not required. If targetUserId is specified then targetAssignmentId is required only if the user has more than one assignment. If both are specified they must be consistent (i.e. the specified assignment must belong to the specified user).
Authentication	User Authentication (required)
URL	`/api/participantSubstituter`

Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`enrollmentId*`	text	the identifier of the enrollment that is to be substituted
`targetUserId`	text	the display ID (login name) of the user to be substituted; may be empty if targetAssignmentId is provided
`targetAssignmentId`	text	the display ID of the assignment to be substituted; may be empty if targetUserId is provided and the user has exactly one assignment
`notifications`	true or false	email notifications will be sent if and only if this is true
format*	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User p ermission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

contentHandler/trainingHistoryCsv

This simple comma-delimited text (CSV) format allows training records to be added and modified. It can also be used to enroll users in existing courses. For details, refer to the Training History Import/Export document (EN145_EKP_TrainingHistoryImportExport_V1.0.pdf).

Authentication	System Authentication
URL	/contentHandler/trainingHistoryCsv
Method	POST
Response Content Type	text/plain

contentHandler/externalTrainingCsv

This simple comma-delimited text (CSV) format allows external training records to be added, modified, and deleted.

For details, refer to the online help of the external training CSV loader in Catalog Manager.

Authentication	System Authentication
URL	/contentHandler/externalTrainingCsv
Method	POST
Response Content Type	text/plain

enrollmentHandler

Function	Enables an external system to submit enrollment requests to the LMS.
Authentication	System Authentication
URL	/enrollmentHandler
Method	POST

Request Content Type	`text/plain`
Response Content Type	`text/plain`

Learning Module Actions

catalog

Function	Returns information about a catalog in XML format. Includes information about the catalog's parent and children and the modules it contains.
Authentication	User Authentication (supported)
URL	`/api/catalog`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	Specifies the ID of the catalog to return (*ROOT* for the top-level catalog)
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

module

Function	Returns information about a program or module in XML format. Includes information about the module or program's sessions, if any.
Authentication	User Authentication (supported)
URL	`/api/module`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	Specifies the ID of the module to return
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

modules

Authentication	User Authentication (supported)
URL	`/api/modules`
Method	`GET`

Response Content Type	`application/xml, application/rss+xml, application/atom+xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Keywords that must be present in the module's catalog entry. If unspecified, results are not filtered by keyword.
`sessionOnOrAfter`	datetime	the date on or after which matching sessions are scheduled, or empty.
`sessionBefore`	datetime	the date before which matching sessions are scheduled, or empty.
`sort*`	relevance, created	`Relevance` to return results according to how closely they match the search criteria; `created` to return results according to the date they were created, most recent first.
`max`	number	The maximum number of matching modules to return. If unspecified, all matching modules are returned.
`format*`	xml, rss, atom	`xml, rss` or `atom`
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** per mission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

preloadModules

Authentication	User Authentication (supported)
URL	`/api/preloadModules`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User per mission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

facility

Function	Returns information about a facility.
Authentication	User Authentication (required)
URL	`/api/facility`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the facility to return
`format*`	json, xml	the response format, `json` or `xml`.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

resources

Function	Returns a reference resource.
Authentication	User Authentication (supported)
URL	/api/resource
Method	GET
Response Content Type	*/*

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Search keywords.
`max`	number	The maximum number of reference resources to return. If unspecified, all matching reference resources are returned.
`format*`	json, xml	`xml`, or `json`.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

resource

Function	Returns a reference resource.
Authentication	User Authentication (supported)
URL	`/api/resource`
Method	`GET`
Response Content Type	`*/*`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`url*`	text	The URL of the reference resource, as returned by the `resources` function.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.

`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

participants

Function	Returns session participants.
Authentication	User Authentication (required)
URL	`/api/participants`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`moduleId*`	text	the module ID.
`sessionId*`	text	the session ID.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

contentPackageImporter

Function	Imports a SCORM or IMS content package.
Authentication	User Authentication (required)
URL	`/api/contentPackageImporter`
Method	`POST`
Request Content Type	`multipart/form-data`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`package*`	file	the content package

`versioning*`	/allowNewRevision/ publishToNotAttempted, /allowNewRevision/ publishToNotCompleted/ reEnrollNone, /allowNewRevision/ publishToNotCompleted/ reEnrollCompleted /ignorePrograms, /allowNewRevision/ publishToNotCompleted/ reEnrollCompleted /includePrograms, /allowNewRevision/ publishToAll	the versioning strategy to employ if there's already a package with the specified identifier
`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the Switch User permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

contentHandler/courses

This XML format allows for the addition and modification of courses and sessions.

Authentication	System authentication
URL	`/contentHandler/courses`
Method	`POST`
Request Content Type	`application/xml`
Request Schema	`xsd/courses.xsd`

contentHandler/courseCSV

This simple comma-delimited text (CSV) format allows learning modules to be added, modified, and deleted. The learning module can be at course level, session level or class level. For details, refer to the online help of the course CSV loader in Catalog Manager.

Authentication	System authentication
URL	/contentHandler/courseCsv
Method	`POST`
Response Content Type	`text/plain`

contentHandler/programCSV

This comma-delimited text format (CSV) allows learning programs, including session information and assigned modules to be added, updated, and deleted. For more details, please refer to the online help of the program CSV loader in Catalog Manager.

Authentication	System authentication

URL	`/contentHandler/programCsv`
Method	`POST`
Response Content Type	`text/plain`

Career Development Actions

competencies

Function	Returns information about competencies.
Authentication	User Authentication (required)
URL	`/api/competencies`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Search keywords.
`max`	number	The maximum number of competencies to return. If unspecified, all matching competencies are returned.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

competency

Function	Returns information about a competency.
Authentication	User Authentication (required)
URL	`/api/competency`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the competency to return
`format*`	json, xml	the response format, `json` or `xml`.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

jobProfiles

Function	Returns available job profiles, organized by catalog.
Authentication	User Authentication (required)
URL	`/api/jobProfiles`
Method	`GET`

Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Search keywords. If this is specified, results are returned in order of relevance.
`max`	number	The maximum number of job profiles to return. If unspecified, all matching job profiles are returned.
`groupBy*`	catalog, none	catalog or none. If unspecified, results are grouped by catalog.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

trainingPlan

Function	Return's a user's training plan.
Authentication	User Authentication (required)
URL	/api/trainingPlan
Method	`GET`

Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

assignedJobProfiles

Function	Returns the job profiles assigned to a user.
Authentication	User Authentication (required)
URL	`/api/assignedJobProfiles`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

competencyCompletionStatus

Function	Returns the completion status of competencies associated with a user.
Authentication	User Authentication (required)
URL	`/api/competencyCompletionStatus`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

certificationsAwarded

Function	Returns certifications awarded to a user.
Authentication	User Authentication (required)
URL	`/api/certificationsAwarded`
Method	`GET`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

certificate

Function	Returns a certificate in PDF format.
Authentication	User Authentication (required)
URL	`/api/certificate`

Method	`GET`
Response Content Type	`application/pdf`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`awardId*`	text	identifier for the certificate to be returned.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

competenciesAwarded

desc.api.competenciesAwarded	
Authentication	User Authentication (required)
URL	`/api/competenciesAwarded`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description

`format*`	json, xml	the response format, json or xml.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

jobProfileAssigner

Function	Assigns a job profile to a user.
Authentication	User Authentication (required)
URL	`/api/jobProfileAssigner`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`userId`	text	Identifier of the user to whom the job profile is to be assigned.
`targetAssignmentId`	text	Assignment of the user to whom the job profile is to be assigned.

`profileId*`	text	Identifier of the job profile to be assigned.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

jobProfileUnassigner

Function	Removes a job profile from a user assignment.
Authentication	User Authentication (required)
URL	`/api/jobProfileUnassigner`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description

`userId`	text	the ID of the user from whom the job profile is to be unassigned.
`profileId*`	text	the ID of the job profile to be unassigned.
`targetAssignmentId`	text	the ID of the user assignment from which the job profile is to be unassigned.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

contentHandler/jobProfilesCsv

Authentication	System authentication
URL	`/contentHandler/jobProfilesCsv`
Method	`POST`
Response Content Type	`text/plain`

contentHandler/sfiaImport

Authentication	System authentication
URL	`/contentHandler/sfiaImport`
Method	`POST`
Response Content Type	`application/vnd.ms-excel`

Community Actions

mailbox

Function	Returns a user's messages.
Authentication	User Authentication (required)
URL	`/api/mailbox`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

message

Function	Returns the content of a message.
Authentication	User Authentication (required)

URL	`/api/message`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the message ID.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

messageUpdater

Function	Marks a message as read.
Authentication	User Authentication (required)
URL	`/api/messageUpdater`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`

Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the message ID.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

latestForumPosts

Function	Returns the authenticating user's assignments.
Authentication	User Authentication (required)
URL	`/api/assignments`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`max`	number	The maximum number of posts to return. If unspecified, all posts are returned.
`format*`	rss, atom	rss for RSS 2.0, or atom for Atom 1.0.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

User Actions

assignments

Function	Returns the most recent forum posts (topics and replies).
Authentication	User Authentication (required)
URL	`/api/latestForumPosts`
Method	`GET`
Response Content Type	`application/rss+xml, application/atom+xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

people

Function	Returns information about people.
Authentication	User Authentication (required)
URL	`/api/people`
Method	`GET`

Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`keywords`	text	Search keywords.
`id`	text	User ID
`first`	text	First Name
`last`	text	Last Name
`max`	number	The maximum number of people to return. If unspecified, all matching people are returned
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

directReports

Function	Returns a user's direct reports.
Authentication	User Authentication (required)
URL	`/api/directReports`

Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

me

Function	Returns information about the authenticating user.
Authentication	User Authentication (required)
URL	`/api/me`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `JSON` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

recentActivity

Function	Returns information about the authenticating user's classmates' recent activity.
Authentication	User Authentication (required)
URL	`/api/recentActivity`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json, xml	the response format, `json` or `xml`.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

users

Function	Returns information about user accounts.
Authentication	User Authentication (required)
URL	`/api/users`
Method	`GET`
Response Content Type	`application/json, application/xml`

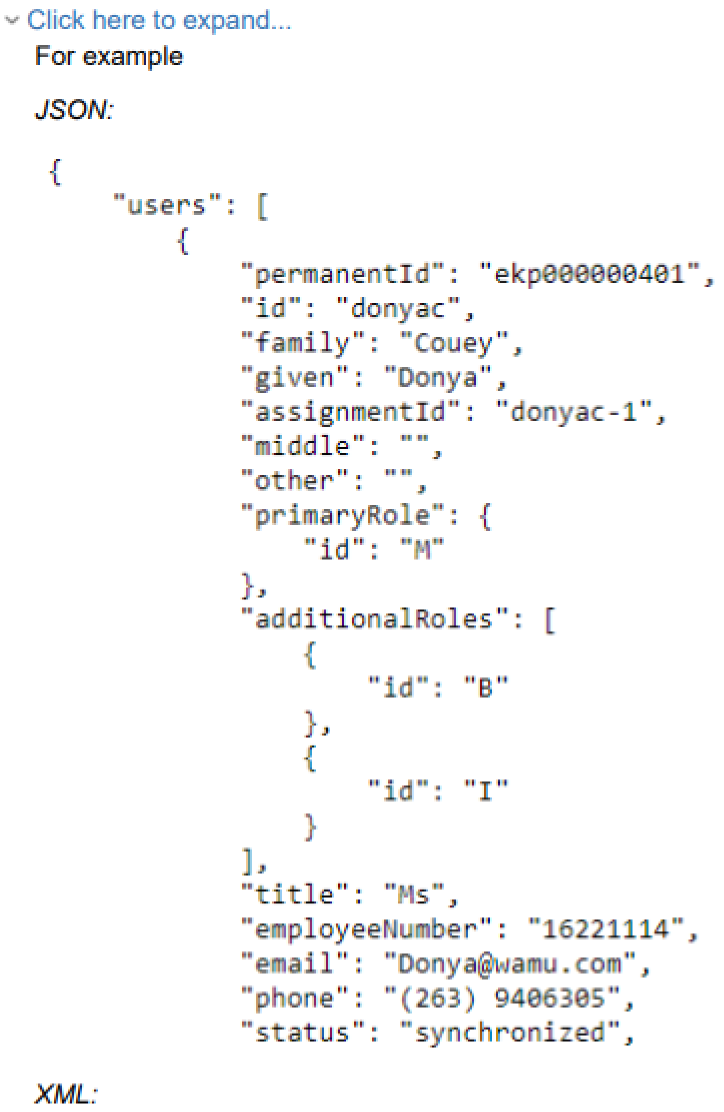
Parameters:

Parameter Name	Parameter Value	Parameter Description
`userId`	text	user ID, or empty.
`organizationId`	text	organization ID, or empty.
`email`	text	email address, or empty.
`format*`	json, xml	the response format, `json` or `xml`.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

Sample Response

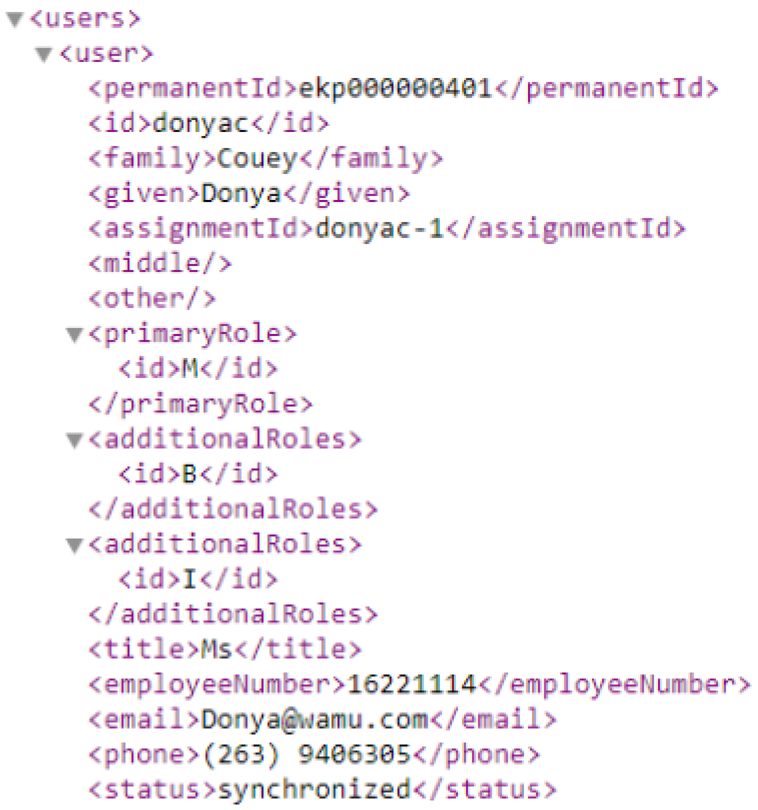


For example

JSON:

```
{
    "users": [
        {
            "permanentId": "ekp000000401",
            "id": "donyac",
            "family": "Couey",
            "given": "Donya",
            "assignmentId": "donyac-1",
            "middle": "",
            "other": "",
            "primaryRole": {
                "id": "M"
            },
            "additionalRoles": [
                {
                    "id": "B"
                },
                {
                    "id": "I"
                }
            ],
            "title": "Ms",
            "employeeNumber": "16221114",
            "email": "Donya@wamu.com",
            "phone": "(263) 9406305",
            "status": "synchronized",
```

XML:



```
▼<users>
  ▼<user>
     <permanentId>ekp000000401</permanentId>
     <id>donyac</id>
     <family>Couey</family>
     <given>Donya</given>
     <assignmentId>donyac-1</assignmentId>
     <middle/>
     <other/>
    ▼<primaryRole>
       <id>M</id>
     </primaryRole>
    ▼<additionalRoles>
       <id>B</id>
     </additionalRoles>
    ▼<additionalRoles>
       <id>I</id>
     </additionalRoles>
     <title>Ms</title>
     <employeeNumber>16221114</employeeNumber>
     <email>Donya@wamu.com</email>
     <phone>(263) 9406305</phone>
     <status>synchronized</status>
```

organization

Function	Returns information about an organization.
Authentication	User Authentication (required)
URL	`/api/organization`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	The ID of the organization to return.

`recursive`	true or false	tells whether to return all descendant organizations, rather than direct children only.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

userGroupCreator

Function	Creates a user group. An optional shared request parameter is supported to allow a group to be shared broadly.
Authentication	User Authentication (required)
URL	`/api/userGroupCreator`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`group-name*`	text	Name of the Group to be Created

`shared`	true or false	`true` to create a shared user group, which means that it will be visible to users within or below the highest-level organization visible to the creator; otherwise, the group will be visible only to the creator.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

userGroupDeleter

Function	Deletes the specified group.
Authentication	User Authentication (required)
URL	`/api/userGroupDeleter`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`group-name*`	text	Name of the Group to be Deleted
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

userGroupAdder

Function	Add the specified user to the specified group.
Authentication	User Authentication (required)
URL	`/api/userGroupAdder`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description

`group-name*`	text	Name of the Group to which the User is to be Added
`user-id`	text	Identifier of the user to be added to the group.
`targetAssignmentId`	text	Assignment of the user to be added to the group.
`targetAssignmentCode`	text	Assignment reference code of the user to be added to the group.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

userGroupRemover

Function	Removes the specified user from the specified group.
Authentication	User Authentication (required)
URL	`/api/userGroupRemover`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`

Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`group-name*`	text	Name of the Group from which the User is to be Removed
`user-id`	text	Identifier of the user to be removed from the group.
`targetAssignmentId`	text	Assignment of the user to be removed from the group.
`targetAssignmentCode`	text	Assignment reference code of the user to be removed from the group.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

supervisedGroupsAdder

Function	Removes the specified group from the specified user's supervised groups.

Authentication	User Authentication (required)
URL	`/api/supervisedGroupsRemover`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`user-id`	text	Identifier of the user to whose supervised groups the group is to be added.
`group-name*`	text	Name of the Group to be Added to the User's Supervised Groups
`targetAssignmentId`	text	Assignment of the user to whose supervised groups is to be added.
`targetAssignmentCode`	text	Assignment reference code of the user to whose supervised groups is to be added.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

supervisedGroupsRemover

Function	Add the specified group to the specified user's supervised groups.
Authentication	User Authentication (required)
URL	`/api/supervisedGroupsAdder`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`user-id`	text	Identifier of the user from whose supervised groups the group is to be removed.
`group-name*`	text	Name of the Group to be Removed from the User's Supervised Groups
`targetAssignmentId`	text	Assignment of the user from whose supervised groups is to be removed.
`targetAssignmentCode`	text	Assignment reference code of the user from whose supervised groups is to be removed.

`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

users200510Xml

Function	Returns information about all users in XML format
Authentication	System authentication
URL	`/api/users200510Xml`
Method	`GET`
Response Content Type	`application/xml`
Response Schema	`xsd/users200510.xsd`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`organizationId`	text	the ID of the organization to filter by.

`status*`	active, suspended, closed, pendingApproval, locked, migrated, licenseViolation, synchronized, inApproval	the status to filter by.
`fields`	text	field selector, e.g. `field1(child1,child2),field2`.

* indicates required parameter

contentHandler/imsEnterprise

Based on the IMS Enterprise Version 1.1 Final XML Binding specification, this format allows for addition, modification, and deletion of `person` (specific LMS user account), `group` (specific Talent Suite organisation) and `membership` (including organization memberships or assign / unassign direct appraisers) elements.

The outcome of processing of each `person / group / membership` element update will return a 200 OK response with an XML body with enterprise root element containing a `result` element.

For details, refer to the Specification for the LMS User Management APIs document (EN035_Specification_for_EKP_User_Management_API_4.0.pdf).

Authentication	System authentication
URL	`/contentHandler/imsEnterprise`
Method	`POST`
Response Content Type	`application/xml`
Request Schema	`dtd/ims_epv1p1.dtd`

contentHandler/usersCsv

This comma-delimited text (CSV) format allows for the addition, modification, and deletion of users. This is the same format used by the CSV user data loader.

For details, refer to the Data Uploading Guide (EN029_EKP_Data_Uploading_Guide_V1_5.pdf).

Authentication	System authentication
URL	`/contentHandler/usersCsv`
Method	`POST`
Response Content Type	`text/plain`

contentHandler/organizationsCsv

This comma-delimited text (CSV) format allows for the addition, modification, and deletion of organizations.This is the same format used by the CSV data loader.

For details, refer to the Data Uploading Guide (EN029_EKP_Data_Uploading_Guide_V5_8.pdf).

Authentication	System authentication
URL	/contentHandler/ekpXml
Method	POST
Request Content Type	application/xml
Response Content Type	application/json, application/xml

contentHandler/tokenAccountCsv

This comma-delimited text (CSV) format allows for the initialization, addition and deletion of token accounts. This is the same format used by the CSV data loader.

Authentication	System authentication
URL	`/contentHandler/tokenAccountCsv`
Method	`POST`
Response Content Type	`text/plain`

contentHandler/ekpXml

This XML format allows for the addition, modification, and deletion of users and learning objects. For details, refer to the Data Uploading Guide (EN029_EKP_Data_Uploading_Guide_V1_5.pdf).

Authentication	System authentication
URL	`/contentHandler/ekpXml`
Method	`POST`
Request Content Type	`application/xml`
Response Content Type	`application/json, application/xml`

Exam Actions

assessmentAttemptHandler

Function	Submission of assessment attempt.
Authentication	User Authentication (required)
URL	`/api/assessmentAttemptHandler`
Method	`POST`
Request Content Type	`application/x-www-form-urlencoded`
Response Content Type	`application/json`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`attempt*`	text	JSON text of the attempt descriptor.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

assessmentAttemptSummary

Function	Returns a summary of an attempt at an assessment.
Authentication	User Authentication (required)
URL	`/api/assessmentAttemptSummary`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`id*`	text	the ID of the attempt for which a summary is to be returned
`format*`	json, xml	the response format, `json` or `xml`.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

Report Actions

batchReports

Function	Returns a listing of batch reports.
Authentication	User Authentication (required)
URL	`/api/batchReports`
Method	`GET`
Response Content Type	`application/json, application/xml`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`name`	text	the report name, or empty.
`createdOnOrAfter`	datetime	the date on or after which returned reports were created, or empty.
`createdBefore`	datetime	the date before which returned reports were created, or empty.
`format*`	json, xml	the response format, `json` or `xml`.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

batchReport

Function	Returns the content of a batch report.
Authentication	User Authentication (required)
URL	`/api/batchReport`
Method	`GET`
Response Content Type	`*/*`

Parameters:

Parameter Name	Parameter Value	Parameter Description
`pathname*`	text	the report pathname.
`assignmentId`	text	the ID of the assignment in the context of which this call will be executed; may be empty if the authenticating user has only one assignment.
`onBehalfOf`	text	identifier of the user on whose behalf the call will be executed, if the authenticating user has the **Switch User** permission, or empty to execute the call on behalf of the authenticating user.

* indicates required parameter

System Actions

session

Function	Returns information about the session.
Authentication	Not supported
URL	/api/session
Method	GET
Response Content Type	application/json, application/xml

Parameters:

Parameter Name	Parameter Value	Parameter Description
format*	json,xml	the response format, json or xml.

* indicates required parameter

version

Function	Returns information about the LMS server version.
Authentication	Not supported
URL	/api/version
Method	GET
Response Content Type	application/json, application/xml

Parameters:

Parameter Name	Parameter Value	Parameter Description
`format*`	json,xml	the response format, `json` or `xml`.

* indicates required parameter

HTTP Response Codes

The PeopleFluent LMS API attempts to return appropriate HTTP status codes for every request. You might encounter the following response codes.

- **200 OK**: The request was processed successfully, and the response contains the requested data (if appropriate).
- **204 No Content**: The request was processed successfully, but the function does not return any response data.
- **400 Bad Request**: The request was invalid. For example, a required parameter was omitted, or a parameter value was not in the expected format.
- **401 Unauthorized**: The method requires authentication credentials and none were supplied, or the supplied credentials were not valid.
- **403 Forbidden**: Your license does not permit API access.
- **404 Not Found**: The URL path does not correspond with a valid API function, or parameters do not reference an existing record.
- **405 Method Not Allowed**: The caller sent a request using an HTTP method that is not supported by the API function, for example attempting to use HTTP GET to access an API function that requires HTTP POST.
- **410 Gone**: The caller attempted to access an API function that was available in an earlier version of the LMS, but is no longer supported.
- **500 Internal Server Error**: The request was valid, but the LMS was unable to process it due to a bug or misconfiguration. More details can usually be found in the log file.

Legal Notice

This document has been created for authorized licensees and subscribers ("Customers") of the software products and associated services of Learning Technologies Group, Inc. by its division PeopleFluent and all of its affiliates (individually and collectively, as applicable, "PeopleFluent"). It contains the confidential and proprietary information of PeopleFluent and may be used solely in accordance with the agreement governing the use of the applicable software products and services. This document or any part thereof may not be reproduced, translated or retransmitted in any form without the written permission of PeopleFluent. The information in this document is subject to change without notice.

PEOPLEFLUENT DISCLAIMS ALL LIABILITY FOR THE USE OF THE INFORMATION CONTAINED IN THIS DOCUMENT AND MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO ITS ACCURACY OR COMPLETENESS. PEOPLEFLUENT DISCLAIMS ALL IMPLIED WARRANTIES INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. PEOPLEFLUENT DOES NOT GUARANTEE THAT ITS PRODUCTS OR SERVICES OR ANY SAMPLE CONTENT CONTAINED IN ITS PRODUCTS AND SERVICES WILL CAUSE OR ENABLE CUSTOMER TO COMPLY WITH LAWS APPLICABLE TO CUSTOMER. USERS ARE RESPONSIBLE FOR COMPLIANCE WITH ALL LAWS, RULES, REGULATIONS, ORDINANCES AND CODES IN CONNECTION WITH THE USE OF THE APPLICABLE SOFTWARE PRODUCTS,INCLUDING, WITHOUT LIMITATION, LABOR AND EMPLOYMENT LAWS IN RELEVANT JURISDICTIONS. THE PEOPLEFLUENT PRODUCTS AND SAMPLE CONTENT SHOULD NOT BE CONSTRUED AS LEGAL ADVICE.

Without limiting the generality of the foregoing, PeopleFluent may from time to time link to third-party websites in its products and/or services. Such third-party links are for demonstration purposes only, and PeopleFluent makes no representations or warranties as to the functioning of such links or the accuracy or appropriateness of the content located on such third-party sites. You are responsible for reviewing all content, including links to third-party web sites and any content that you elect to use, for accuracy and appropriateness, and compliance with applicable law.

Any trademarks included in this documentation may comprise registered trademarks of PeopleFluent in the United States and in other countries.

Microsoft, Windows, and Internet Explorer are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries. Oracle and PeopleSoft are registered trademarks of Oracle International Corporation. Adobe and Acrobat are registered trademarks of Adobe Systems Incorporated. All other names are used for identification purposes only and are trademarks or registered trademarks of their respective owners. Portions of PeopleFluent Workforce Communication software may include technology licensed from Autonomy and are the copyright of Autonomy, Inc.Quartz Scheduler is licensed under the Apache License.

Website: peoplefluent.com