## Contents

### Before you begin
- Tips and training .............................................................. 1
- Development environment .................................................. 2
- Additional learning ............................................................. 2
- Conventions ........................................................................ 3

### Chapter 1: Architecture
- Data flow ............................................................................... 5
- Making your first API call ..................................................... 7

### Chapter 2: Login and requests
- Log in from an application ................................................... 10
- Send a request in an XML document ...................................... 15
- Parse a response with XPath .................................................. 15
- Parse an error response ......................................................... 16
- Log a user out ....................................................................... 17

### Chapter 3: Basics
- Find a principal-id ............................................................... 18
- List principals or guests ........................................................ 19
- Create users ......................................................................... 20
- Update users ........................................................................ 21
- Create custom fields ............................................................ 21
- Create groups ...................................................................... 22
- Find SCOs ............................................................................ 24
- Download files ...................................................................... 27
- Check permissions ................................................................ 27

### Chapter 4: Meetings
- Find meetings ...................................................................... 29
- Display meetings ................................................................... 30
- Create meeting room URLs .................................................. 31
- Create meetings .................................................................... 32
- Create customized meetings .................................................. 33
- Invite users to meetings ......................................................... 34
- Remove users from meetings .................................................. 36
- Calculate meeting usage ....................................................... 36
- Check meeting quotas ........................................................... 37
- Get meeting archives ............................................................ 38
- Get meeting poll results ......................................................... 38
- Launch meetings with external authentication ....................... 39
- Configure compliance settings .............................................. 40
Chapter 5: Training

Training library permissions .............................................. 44
Find courses and curriculums ............................................... 45
Create a course .................................................................. 47
View a user’s training .......................................................... 47
Enroll one user ................................................................. 48
Enroll a large number of users .............................................. 49
View curriculum information ................................................. 51
Report scores ................................................................... 53

Chapter 6: Action reference

New and removed actions .................................................... 54
account-expiry-info .......................................................... 55
acl-field-info ................................................................ 56
acl-field-list .................................................................. 57
acl-field-update ............................................................... 59
acl-preference-update ....................................................... 60
common-info .................................................................. 62
custom-fields .................................................................. 64
custom-fields-delete .......................................................... 66
custom-field-update ........................................................... 67
expiry-settings-info ............................................................ 70
expiry-settings-update .......................................................... 71
group-membership-update .................................................. 72
learning-path-info ............................................................. 73
learning-path-update ........................................................... 75
limited-administrator-permissions info ................................. 76
limited-administrator-permissions-update ............................... 79
login .............................................................................. 81
logout ............................................................................. 83
meeting-disclaimer-info ...................................................... 84
meeting-disclaimer-update .................................................... 85
meeting-feature-update ....................................................... 86
permissions-info ................................................................. 87
permissions-reset ............................................................... 90
permissions-update ............................................................. 91
principal-info ................................................................. 92
principal-list ................................................................. 95
principal-list-by-field .......................................................... 98
principals-delete .............................................................. 103
principal-update ............................................................. 104
quota-threshold-info .......................................................... 106
quota-threshold-exceeded .................................................. 108
quota-threshold-update ....................................................... 110
report-active-meetings ......................................................... 112
report-bulk-consolidated-transactions ................................. 114
Chapter 9: A sample application

Build an adapter class .................................................. 219
Log the user in ............................................................. 220
Send XML requests ....................................................... 221
Parse XML responses ................................................... 222
Display user information ............................................... 223
List a user's meetings .................................................. 224
Create and update meetings .......................................... 225
Display meeting detail ................................................. 227
Before you begin

Adobe® Acrobat® Connect Pro and Adobe Acrobat Connect Pro Server 7 expose web services that clients can call to exchange data with Acrobat Connect Pro accounts.

This guide explains how an application sends XML API calls to Acrobat Connect Pro and interprets the XML response. It is intended for developers who want to build custom applications for Acrobat Connect Pro or integrate it with another system such as a learning management system or LDAP directory service.

Before you use this guide, you should understand the basics of XML and of using HTTP to communicate with a server from a client application. This guide includes some Java™ code samples, but it does not presume that you are using one specific language or environment.

Tips and training

Adobe provides a wide range of resources to help you learn and use Adobe products.

- Adobe Developer Center: Tutorials, reference material, scripting guides, and other in-depth information.
- Other resources: Training, books, user forums, product certificates, and more.
- Extras and downloads: Downloadable content and software.

Adobe Developer Center

This community-based section on Adobe.com provides information for advanced users, including software and plug-in developers. Here you’ll find tutorials, SDKs, scripting guides, and sample code, in addition to forums, RSS feeds, online seminars, and other technical resources.

Other resources

Additional sources of information and help are available for Adobe products.

- Visit the Support Home area of the Adobe website for additional information about free and paid technical support options. Top issues are listed by product on the Adobe U.S. and Adobe Japan websites. Follow the Training link for access to Adobe Press books; online, video, and instructor-led training resources; Adobe software certification programs; and more.
- Visit the Communities area of the Adobe website to find forums, blogs, and other avenues through which users share tools and information, ask questions, and find out how others are getting the most out of their software. User forums are available in English, French, Spanish, German, and Japanese on the main Support page of your local Adobe website. Blogs are posted in a wide range of languages.

Extras and downloads

For free content and add-ons, visit Adobe Studio® Exchange, an online community where users download and share thousands of free actions, plug-ins, and other content for use with Adobe products. To visit Adobe Studio Exchange, go to Adobe Design Center from the home page of the Adobe website.

The downloads available from the Adobe website include free updates, tryouts, and other useful software. In addition, the Plug-ins section of the Adobe Store provides access to thousands of plug-ins from third-party developers, helping you automate tasks, customize workflows, create specialized professional effects, and more.
Development environment

Acrobat Connect Pro Web Services allows you to use any language or platform that can send and receive XML over HTTP to develop custom applications. For example, you can use Java and the J2EE platform, C#.NET, PHP, a portal server, or any web development platform. Most custom applications are web applications or portals.

In general, you may find these types of tools useful:

- An XML parser code library, if your programming language supports XML parsing.
- A cookie management code library, to help you manage the session cookies Acrobat Connect Pro returns.
- A tool for viewing HTTP request and response headers in a browser. Many such tools are available on the Internet.

Additional learning

You can find many useful resources on the Internet that provide information about Acrobat Connect Pro, web services and XML, and other technologies that Acrobat Connect Pro uses.

Acrobat Connect Pro

Acrobat Connect Pro Resource Center  The resource center, on the Adobe website, is updated regularly with tutorials, simulations, best practices information, and links to procedures.

Acrobat Connect Pro Developer Center  Developers will find the Developer Center useful. It is updated regularly with tutorials and articles about building applications for Acrobat Connect Pro.

Acrobat Connect Pro Documentation Center  The Documentation Center makes all Acrobat Connect Pro product documentation available for download or product viewing.

Acrobat Connect Pro Support Center  When you need technical support, the Support Center has current information, including technical notes, Acrobat Connect Pro presentations, and details of the support program. You can also check the Support Forums for peer-to-peer discussions on technical issues.

Acrobat Connect Pro Licensed Support Center  The Licensed Support Center has documentation and other resources for customers who license Adobe Acrobat Connect Pro Server 7.

XML and web services

The Web Services Primer  at the Xml.com website (xml.com) is a good introduction to web services.

The XML Tutorial  at the W3Schools website (w3schools.com) can help you get started with XML.

The XPath Tutorial  also at the W3Schools website (w3schools.com), describes XPath, which parses an XML document so that you can use it in an application.

The XSLT Tutorial,  a third tutorial at the W3Schools website (w3schools.com), teaches you XSL Transformations, which you use to convert XML data to other formats.

The XSL Transformations (XSLT) specification  at the W3C website (w3.org) is the official definition of XSLT, from the standards committee who created it.

Numeric Representation of Dates and Time,  at the International Organization for Standardization website (iso.org), provides information about how to use the ISO 8601 standard date and time format.

Date and Time Formats  at the W3C website (w3.org) is the official definition of the ISO 8601 date and time format.
Other technologies

Flash Player Developer Center and Flash Media Server Developer Center, both available from the Adobe Developer Center, offer articles, samples, and insights to developing applications that use Adobe Flash Player and Adobe Flash Media Server.

SCORM Concepts, at the Eduworks Corporation website (eduworks.com), is a tutorial about the Shareable Content Object Reference Model and describes Shareable Content Objects (SCOs) and Learning Management Systems (LMSs).

An LDAP Roadmap at the Kings Mountain Systems website (www.kingsmountain.com), provides a useful overview of the Lightweight Directory Access Protocol (LDAP). This site might provide good background material or links for developers integrating an LDAP directory with Acrobat Connect Pro.

Microsoft SQL Server Acrobat Connect Pro uses a Microsoft SQL Server database, which your custom applications retrieve data from and write data to. You may find useful resources at the Microsoft SQL Server Developer Center (msdn.microsoft.com) including references, community, support, and other information.

Conventions

This guide uses industry standard conventions for displaying code that you are already familiar with. However, API reference is a formal definition of the API contract between a calling application and the server. As such, the syntax definitions of request URLs should be described.

We have placed distinct sections of a request URL on separate lines for readability, like this:

http://server_name/api/xml
?action=custom-fields
&filter-definition=value
&session=BreezeSessionCookieValue

When you enter a request URL in the address bar of a browser or construct it in an application, enter it or construct it as a single line:

https://example.com/api/xml?action=custom-fields&filter-name=location

Syntax elements in blue code font represent definitions that you construct, with a hyperlink to the syntax of the definition.
Chapter 1: Architecture

Welcome to Adobe® Acrobat® Connect™ Pro Web Services, the web service layer over Adobe Acrobat Connect Pro Server 7 and the Acrobat Connect Pro suite of applications.

Web Services allows you to build portals or web applications that integrate Acrobat Connect Pro functionality and reporting information with third-party systems such as portals, customer relationship management systems, and enterprise resource planning systems.

Acrobat Connect Pro Web Services provides meeting, training, and events functionality to your applications through its XML API.

As an example, you might have a central user management system, such as an LDAP directory, Microsoft Active Directory, or another third-party system, that is an integral part of your business processes.

Using Web Services, you can write an application that synchronizes users between your system and Acrobat Connect Pro. The application can use the J2EE platform or another technology of your choice to pull a list of users from the directory, compare it against a list of Acrobat Connect Pro users, and then perform requested updates within the Acrobat Connect Pro user repository, such as adding or deleting users or groups. This is just one example of a custom application, and you can check the Acrobat Connect Pro Developer Center for more.
Data flow

The data flows between client applications and Acrobat Connect Pro are shown in the following diagram. Custom applications that you write use paths 1 to 2 and A to B. Acrobat Connect Pro applications (such as Adobe® Acrobat® Connect™ Pro, Acrobat Connect Pro Training, or Acrobat Connect Pro Events) can use any of the data flow paths.

The data flow can be encrypted with SSL or unencrypted.

**Unencrypted** If the data flow is unencrypted, connections are made over HTTP and Adobe Real Time Messaging Protocol (RTMP) and follow the paths described in the following table.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The client web browser requests an Adobe Acrobat Connect Pro meeting or content URL over port HTTP:80 (connection paths may vary).</td>
</tr>
<tr>
<td>2</td>
<td>The web server responds with content transfer or provides the client browser with information to enter Adobe Acrobat Connect Pro.</td>
</tr>
<tr>
<td>3</td>
<td>Adobe Flash® Player requests a connection to Adobe Flash Media Server over RTMP:1935 and HTTP:80.</td>
</tr>
<tr>
<td>4</td>
<td>Flash Media Server responds, and a persistent connection is opened to stream meeting traffic to the browser.</td>
</tr>
<tr>
<td>3a (alternate)</td>
<td>In some cases, Flash Player requests a connection to the Flash Media Server, but can only obtain a tunneled connection over RTMPT:80.</td>
</tr>
<tr>
<td>4a (alternate)</td>
<td>Flash Media Server responds, and a tunneled connection is opened to stream meeting traffic to the browser.</td>
</tr>
</tbody>
</table>
**Encrypted** If the data flow is encrypted, connections are made securely over HTTPS and RTMPS (Real Time Messaging Protocol over SSL), as follows.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The client web browser requests a secure meeting or content URL over an encrypted connection on HTTPS:443 (connection paths may vary).</td>
</tr>
<tr>
<td>B</td>
<td>The web/application server responds with an encrypted content transfer or provides the client with information to make an encrypted connection to Acrobat Connect Pro.</td>
</tr>
<tr>
<td>C</td>
<td>Flash Player requests an encrypted connection to Flash Media Server over RTMPS:443.</td>
</tr>
<tr>
<td>D</td>
<td>Flash Media Server responds, and a persistent connection is opened to stream meeting traffic to the browser.</td>
</tr>
</tbody>
</table>

**Custom applications**

Acrobat Connect Pro Web Services provides an XML API, so your application must be able to communicate with Acrobat Connect Pro Server using XML over HTTP or XML over HTTPS. Your application calls the API by building a request URL and passing it one or more parameters, either as name/value pairs or as an XML document. Web Services returns an XML response, from which you can extract values.

Custom applications retrieve metadata from the Acrobat Connect Pro database. Metadata includes meeting or course names and times, meeting room URLs, content URLs, and report information.

The data flow for a custom application retrieving metadata from the database is from a client web browser, to the client web application server, to the XML API, the Acrobat Connect Pro web application server, and the SQL database—and then back again.

The data flow between a custom application and Acrobat Connect Pro works like this:

1. A user accesses your custom application from a web browser.
2. The application calls the XML API over HTTP:80 or HTTPS:443.
3. The Acrobat Connect Pro web application server authorizes the application and its users, retrieves metadata from the SQL database, and returns the metadata.
4. On the client side, your web or application server, XML parser, and software libraries handle the response and return it to your application.
5. The user continues to work in your custom application, and clicks a meeting or content URL. At this point, the user accesses an Acrobat Connect Pro application to enter a meeting room, and the typical data flow between an Acrobat Connect Pro application and the server begins.

**Acrobat Connect Pro applications**

Acrobat Connect Pro applications call the server using the same Web Services XML API that you use from a custom application.

In general, content is transported over HTTP port 80 or HTTPS port 443. Content includes slides, HTTP pages, SWF files, and files transferred through the FileShare pod. These are default port numbers that you can configure (see Acrobat Connect Pro Installation and Configuration Guide for details).

Streamed, real-time communications from Flash Media Server are transported over RTMP port 1935. Streamed communications include audio, video (webcam and FLV), file share, and chat. Meeting state is also maintained over RTMP port 1935.

**Components of Acrobat Connect Pro**

Acrobat Connect Pro is architected with two server components, and each server uses a SQL database.
The web application server  The web application server is the brains of Acrobat Connect Pro. It contains and executes all of the business logic needed to deliver content to users. It handles access control, security, quotas, and licensing, as well as management functions such as clustering, failover, and replication.

The web application server also handles Acrobat Connect Pro Central, the application through which you view and manage your organization’s content and users—when you are not using a custom application or integrated third-party system. The metadata describing content and users can be stored in either single or multiple replicated SQL databases. The web application server is stateless, which means that scaling is near linear.

Flash Media Server  Flash Media Server is the muscle of Acrobat Connect Pro. Flash Media Server streams audio, video, and rich media content using RTMP. When a meeting is recorded and played back, audio and video are synchronized, or content is converted and packaged for real-time screen sharing. Flash Media Server does the job. Flash Media Server also plays a vital role in reducing server load by caching frequently accessed web pages, streams, and shared data.

The SQL database  Acrobat Connect Pro uses the Microsoft SQL Server database for persistent storage of transactional and application metadata, including users, groups, content, and reporting information. The XML API retrieves metadata stored in the database. The database can be implemented with either the Microsoft SQL Server Desktop Engine (MSDE) or the full version of Microsoft SQL Server 2005.

Making your first API call

Acrobat Connect Pro Web Services uses a servlet framework to handle XML API requests. In the data flow diagram, the servlet framework is represented by the API component. The API servlet receives XML requests from clients and returns XML responses from the web application server and the database.

A request to the XML API is formatted as an HTTP request URL that the API servlet handles. A request URL has an action name and parameters in name/value pairs, like this:

```
https://example.com/api/xml?action=sco-info&sco-id=200634909
```

If you have access to a Acrobat Connect Pro account in which you can test API calls, you can experiment. In fact, Adobe recommends testing API calls in the browser while you learn the API and write applications.

Before you begin, it's useful to install a tool that allows you to view HTTP request and response headers in your browser.

Call common-info in a browser

1  (Optional) Enable a tool for viewing HTTP headers in your browser.

2  Open a browser and navigate to your Acrobat Connect Pro login page.

3  Without logging in, delete the part of the URL after the domain name and add a call to common-info:

```
https://example.com/api/xml?action=common-info
```

The response from common-info gives you information about your session with the server, especially the cookie that identifies your session:

```
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <common locale="en" time-zone-id="85">
    <cookie>breezbryf9ur23mboks8</cookie>
    <date>2008-03-13T01:11:19+00:00</date>
    <host>https://example.com/</host>
    <local-host>abc123def789</local-host>
    <url>/api/xml?action=common-info</url>
</common>
</results>
```
When you log a user in from an application, you need to send the cookie value back to the server to identify the user's session (see “Log in from an application”).

**Call principal-list in a browser**

Once you have the BREEZESSESSION cookie value from common-info, the browser adds it to the request header on your next request.

1. In a web browser, log in to Acrobat Connect Pro. Change the browser URL to call `principal-list`:
   
   https://example.com/api/xml?action=principal-list

2. Check the request header. This time it sends the BREEZESSESSION cookie value back to the server:

   ```
   GET /api/xml?action=principal-list HTTP/1.1
   Accept: */*
   Accept-Language: en-us
   Accept-Encoding: gzip, deflate
   User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
   Host: example.com
   Connection: Keep-Alive
   Cookie: BREEZESSESSION=breezbryf9ur23mbokzs8
   ```

3. Check the response, which lists all principals on the server, each in its own `principal` element.

   ```
   <?xml version="1.0" encoding="utf-8" ?>
   <results>
     <principal principal-id="624526" account-id="624520" type="user"
       has-children="false" is-primary="false" is-hidden="false">
       <name>joe harrison</name>
       <login>jharrison@example.com</login>
       <email>jharrison@example.com</email>
     </principal>
     <principal principal-id="624550" account-id="624520" type="user"
       has-children="false" is-primary="false" is-hidden="false">
       <name>bob jones</name>
       <login>bjones@example.com</login>
       <email>bjones@example.com</email>
     </principal>
     ...
   </principal-list>
   </results>
   ```

**Add filters and sorts**

Many actions in the API allow you to add a filter to return only certain response elements or a sort to display response elements in a certain order.

A filter is a special parameter that starts with the keyword `filter`, followed by an optional modifier, then a field name and a value. These are all examples of filters:

- `filter-name=jazz doe` (which matches results with the exact name `jazz doe`)
- `filter-like-name=jazz` (which matches any results that contain `jazz` in the name)
• filter-out-type=user (which returns any results that do not have a type of user)

These are just a few filter types, and you can find more in filter-definition. Check an action in the reference (at “Action reference” on page 54) to see whether its response can be filtered. In general, if an action allows filters, you can use them on any response element or attribute.

A sort is another special parameter that starts with the keyword sort (or sort1 or sort2), followed by a field name and then one of the keywords asc or desc, for example:

• sort-name=asc (to sort in ascending order by name)
• sort-group-id=desc (to sort in descending order by group-id)

These are just a few sort examples. You can test sorts in the browser or see sort-definition for more.

Make a call with a filter and sort

1 Call principal-list again, displaying only groups and sorting them alphabetically by name:

   https://example.com/api/xml?action=principal-list&filter-type=group
   &sort-name=asc

2 To tighten the response, choose a group from the list and filter on its name:

   https://example.com/api/xml?action=principal-list&filter-name=developers

   This time, only one group is returned:

   ```xml
   <?xml version="1.0" encoding="utf-8" ?>
   <results>
     <status code="ok" />
     <principal-list>
       <principal principal-id="2007105030" account-id="624520"
                   type="group" has-children="true" is-primary="false"
                   is-hidden="false">
         <name>developers</name>
         <login>developers</login>
       </principal>
     </principal-list>
   </results>
   ```

Where to go from here

At this point, you can continue to test calls in the browser and observe how they work. It's the best and easiest way to learn the XML API. When you need more information, turn to any of these sources:

• The API reference in “Action reference” on page 54
• “Login and requests” on page 10 for information on how to log users in from applications
• “Basics” on page 18 to learn the three basic concepts underlying the API
• “Meetings” on page 29 if you want to create and manage meetings from an application
• “Training” on page 44 if you are building a training application
Chapter 2: Login and requests

This chapter explains how to log a user in from your application, make requests, handle responses, and log the user out.

There are several ways to accomplish most of these tasks, depending on your development environment, server configuration, and application design.

Log in from an application

Any custom application you write that uses Adobe® Acrobat® Connect™ Pro Web Services functionality or integrates with a third-party system needs to log in a user to Adobe Acrobat Connect Pro Server 7 or to an Adobe Acrobat Connect Pro hosted account. In its simplest form, the process of logging in calls the login action.

However, the technique for logging in varies according to whether you use cookie management, have a licensed server or a hosted account, and authenticate directly to Acrobat Connect Pro or use external authentication. Depending on your environment and server configuration, you might also use combinations of these options.

Cookie management  When a user logs in, Acrobat Connect Pro returns a cookie that identifies the user's session. You need to pass the cookie back to the server on all calls made to the server during the user's session. Then, when the user logs out, the server makes the cookie expire and you should invalidate it.

In your development environment, you can use a code library that manages cookies for you. The process of logging in and managing a user's session varies according to whether you use a cookie management library or manage the user's session yourself.

Licensed server or hosted account  Your organization might have a licensed Acrobat Connect Pro Server within your firewall, or you may have a Acrobat Connect Pro hosted account at Adobe. Either way, you send XML requests over HTTP or HTTPS, but security requirements and the login process vary. If you are a hosted customer, you can use certain parameters with the login action to avoid sending user IDs and passwords over the Internet.

Direct or external authentication  Whether you are a hosted or licensed customer, your application might authenticate directly to Acrobat Connect Pro, or you might authenticate users on your own network, set an identifier in an HTTP request header, and send it to Acrobat Connect Pro. The login process varies according to whether you use direct or external authentication.

Log in to Acrobat Connect Pro Server

The standard technique for logging a user in to Acrobat Connect Pro Server uses the login action, passing the user's login ID and password. This technique works with both HTTP GET and POST requests.

You also need to manage the BREEZESSESSION cookie the server returns for each user session. If you use a client-side cookie management library, it is much easier to allow it to manage cookies for you than to manage the cookies yourself. If you do not have such a library, call login with the session parameter, as it is easier and more reliable than setting HTTP header values.

Note: If you send user passwords to Acrobat Connect Pro Server, use SSL so passwords are encrypted in transit, even if you have a licensed Acrobat Connect Pro Server within your own firewall.

Log in with cookie management

1  Call the login action, passing it the user's login ID and password, but no session parameter:
http://example.com/api/xml?action=login&login=bobs@acme.com
&password=football

2 Parse the response for a status code of ok.
   If the login is successful, the server returns the BREEZESESSION cookie in the response header:
   Set-Cookie: BREEZESESSION=breezbryf9ur23mbokzs8;domain=.macromedia.com;path=/

3 Allow your cookie management library to manage the BREEZESESSION cookie.
   Your client-side library passes the cookie back to the server in a request header on subsequent calls for the
   remainder of the user's session. You do not need to set the cookie in the request header explicitly. When the user
   logs out, the cookie expires.

Log in using the session parameter
1 Before you log the user in, call common-info to get the value of the BREEZESESSION cookie:
   http://example.com/api/xml?action=common-info

2 Extract the cookie value from the response:
   <cookie>breezxq66rt43poa13if8</cookie>

3 Log the user in, specifying the cookie value:
   http://example.com/api/xml?action=login&login=bobs@acme.com
   &password=football&session=breezxq66rt43poa13if8

4 Parse the response for a status code of ok.
5 Use the session parameter with the same cookie value on subsequent calls for the user, until the user's session
   ends:
   https://example.com/api/xml?action=principal-list
   &session=breezxq66rt43poa13if8

6 When the user logs out or the user's session ends, do not reuse the cookie value.

Log in to a Acrobat Connect Pro hosted account
If you want to log in directly to a Acrobat Connect Pro hosted account or multiple hosted accounts, you still use the
login action, but you need to specify an account ID or domain name, in addition to the user's login ID and
password. You can specify a domain name if you want to avoid sending an account ID over the Internet.
With a Acrobat Connect Pro hosted account, you cannot use single sign-on or external authentication. You must pass
the user's authentication credentials on the Acrobat Connect Pro hosted account, not the credentials for an
external network.

Note: It is important to have SSL enabled on your Acrobat Connect Pro hosted account, because you are sending user
IDs, passwords, and account information over the Internet to your Acrobat Connect Pro account hosted at Adobe.

Log in to a Acrobat Connect Pro hosted account with an account ID
1 Before you log the user in, call common-info with the domain name of your Acrobat Connect Pro hosted
   account in either the request URL or the domain parameter:
   http://acme.adobe.com/api/xml?action=common-info
   http://adobe.com/api/xml?action=common-info&domain=acme.adobe.com

2 Parse the response for the values of cookie and account-id:
   <cookie>Sbreezzd2dfr2ua5gscogv</cookie>
   ...
   <account account-id="295153" />
3 Collect the user's login ID and password in your application.
4 Call the login action, adding the user's credentials and the account-id and session parameters:
   https://example.com/api/xml?action=login&login=joy@acme.com
   &password=happy&account-id=295153&session=Sbreeszd2dfr2ua5gscogv
5 Parse the response for a status code of ok.
6 (Optional) If you prefer, you can call login before common-info, extract the cookie value from the response header, and manage it yourself or using a cookie management library. For details, see "Log in with cookie management" or "Log in using the session parameter."

Log in to a Acrobat Connect Pro hosted account with a domain name
1 Before you log the user in, call common-info with the domain name of your Acrobat Connect Pro hosted account in either the request URL or the domain parameter:
   http://acme.adobe.com/api/xml?action=common-info
   http://adobe.com/api/xml?action=common-info&domain=acme.adobe.com
2 Parse the response for the values of cookie and host:
   <cookie>breezxq66rt43poai3if8</cookie>
   ...<host>https://acme.adobe.com</host>
3 Extract the domain name from the value of host:
   acme.adobe.com
4 In your application, collect the user's login ID and password.
   Be sure the login ID is the user's Acrobat Connect Pro hosted account login ID, not an external one.
5 Call login, adding the user's credentials and the domain and session parameters:
   https://example.com/api/xml?action=login&login=joe
   &password=smith99&domain=acme.adobe.com&session=breezxq66rt43poai3if8
   The domain is equivalent to the account-id, but by using it you can avoid sending an account ID over the Internet, especially if you use a non-encrypted connection.
6 Parse the response for a status code of ok.
7 (Optional) If you prefer, you can call login before common-info, extract the cookie value from the response header, and manage it yourself or using a cookie management code library. For details, see "Log in with cookie management" or "Log in using the session parameter."

Log in using HTTP header authentication
Note: The instructions in this section apply only to Acrobat Connect Pro Server.
Your application can use a trusted central server to authenticate users with single sign-on and pass your network's (here called external) authentication to Acrobat Connect Pro Server, without explicitly passing a Acrobat Connect Pro Server user ID and password. (For detailed instructions on how to set up and configure HTTP header authentication, see Adobe Acrobat Connect Pro Installation and Configuration Guide).
With HTTP header authentication, a user logs in to your authentication server. Once the user is authenticated, you add an HTTP request header that identifies the user, or configure a proxy server to add the header. The authentication filter on Acrobat Connect Pro (named HeaderAuthenticationFilter) converts your user identifier to a Acrobat Connect Pro login ID and authenticates the user.

Authentication filters convert external authentication credentials to Acrobat Connect Pro credentials.

External authentication works in addition to standard Acrobat Connect Pro authentication. Each user who needs to access Acrobat Connect Pro Server needs a valid Acrobat Connect Pro Server login and password.

When you send a login request to Acrobat Connect Pro Server with an external authentication credential:

- The authentication filter intercepts the request and checks for a user on Acrobat Connect Pro Server with an ext-login field that matches your external credential.
- If a match exists, the filter passes your external authentication to Acrobat Connect Pro Server, and the server logs the user in.
- If no match exists, the filter passes the login request to the server, which displays its login page. The user must then log in to Acrobat Connect Pro Server.
- If the user logs in successfully, Acrobat Connect Pro Server updates the ext-login field in the user’s profile with the external credential from your request. The next time you send a request with the user’s external credential, Acrobat Connect Pro Server finds a match in ext-login, and the user does not need to log in to Acrobat Connect Pro.
- If the user does not log in successfully, the user is not allowed access to Acrobat Connect Pro Server applications, content, or meetings.

The steps that follow describe how to call login when you use HTTP header authentication.

**Log in to Acrobat Connect Pro Server using HTTP header authentication**


2. In [your server directory]/appserv/conf/WEB-INF/web.xml, remove comment tags around the filter-mapping element for HeaderAuthenticationFilter and add comment tags around any other filter-mapping elements:

   ```
   <filter-mapping>
     <filter-name>HeaderAuthenticationFilter</filter-name>
     <url-pattern>/*</url-pattern>
   </filter-mapping>
   <!--
   !--
   <filter-mapping>
     <filter-name>NtlmAuthenticationFilter</filter-name>
     <url-pattern>/*</url-pattern>
   </filter-mapping>
   -->
   ```
3 In the filter element for `HeaderAuthenticationFilter`, enable the `/api/` pattern for request URLs. You have two choices for how to do this:

**If your application uses the XML API and any Acrobat Connect Pro applications**

In the filter element for `HeaderAuthenticationFilter`, use comment tags to disable the `init-param` element with a `param-value` of `/api/`:

```xml
<init-param>
  <param-name>ignore-pattern-0</param-name>
  <param-value>/api/</param-value>
</init-param>
```

**If your application uses only the XML API**

Change the `filter-mapping` element for your filter type to use the URL pattern `/api/*` instead of `/*`:

```xml
<filter-mapping>
  <filter-name>HeaderAuthenticationFilter</filter-name>
  <url-pattern>/api/*</url-pattern>
</filter-mapping>
```

Then, in the filter element for your filter type, add comment tags around all `init-param` elements with a `param-name` of `ignore-pattern-x`:

```xml
<filter>
  <filter-name>HeaderAuthenticationFilter</filter-name>
  <filter-class>com.macromedia.airspeed.servlet.filter.HeaderAuthenticationFilter</filter-class>

  <!--
  ...
  -->

  <init-param>
    <param-name>ignore-pattern-0</param-name>
    <param-value>/api/</param-value>
  </init-param>

  ... 

  <init-param>
    <param-name>ignore-pattern-4</param-name>
    <param-value>/servlet/testbuilder</param-value>
  </init-param>

  <!--
  ...
  -->

</filter>
```

4 Configure Acrobat Connect Pro Server so that users are created with the field `ext-login` set to the external user ID you send (see *Acrobat Connect Pro Installation and Configuration Guide* for details).

By default, `ext-login` has the same value as `login`, the Acrobat Connect Pro Server login ID.

5 Once your system authenticates the user, create a `login` request. Add the parameter `external-auth=use`, but no `login` or `password` parameters:

https://example.com/api/xml?action=login&external-auth=use

6 Add your authenticated user ID to the HTTP request header. By default, use the header name `x-user-id`:

```xml
x-user-id: joesmith
```

You can specify a different header name by setting a value for `HTTP_AUTH_HEADER` in the custom.ini file. You can also configure a proxy server to set the HTTP header value. See *Acrobat Connect Pro Installation and Configuration Guide* for details of either.

7 Parse the response for a status code of `ok`.

8 Handle the `BREEZESESSION` cookie value returned in the response header. You have two choices for how to do this:
If you use a client library that manages cookies  Allow your library to extract the cookie value, store it, and pass it back to the server on subsequent requests for the user.

If you manage cookies yourself  Extract the value of the BREEZESESSION cookie from the response header. Store it and pass it back to the server in the session parameter of all subsequent actions you call for the same user, as long as the user’s session is valid:

https://example.com/api/xml?action=principal:list&session=breezs7zuem4y9wh2tseu

Be sure not to reuse the cookie value when the user’s session ends.

Send a request in an XML document

At times, you may prefer to send an HTTP POST request to the server to make sure the data is secure and not visible in transit. In that case, specify the action name and parameters in an XML document.

Make an XML document request

1  Create an XML document with the root element params and param child elements for the action name and each parameter:

   <params>
      <param name="action">login</param>
      <param name="login">jon@doe.com</param>
      <param name="password">foobar</param>
   </params>

   • You can only send one action in the params root element. You cannot batch multiple actions to be executed sequentially.
   • The XML document you send must be valid and well-formed. Try validating the document in an XML editor before you send it.

2  Write code that sends an HTTP POST request to Acrobat Connect Pro and receives an XML response.

   The specific code will vary according to your programming language and development environment.

3  In your code, send the XML document to Acrobat Connect Pro in the body of the HTTP POST request.

   • Read the XML document into the request.
   • Be sure to set a content-type header of text/xml or application/xml.

Parse a response with XPath

When you receive an XML response from Acrobat Connect Pro, you need to be able to parse it to extract the XML elements you need.

If you are working in a language such as Java™, with an XML parser (such as Xerces or JDOM) installed, you can parse through an XML response, select values from nodes, and then use those values.

Use XPath to parse a response

1  Write a method that calls one or more actions. Create an instance of the XPath class so that you can use the XPath expressions. Call the actions, read the XML response, and use XPath syntax to select the values you need:

   public String scoUrl(String scoId) throws XMLApiException {
   try {
      Element e = request("sco-info", "sco-id=" + scoId);
if(!(codePath.valueOf(e).equalsIgnoreCase("ok")))
    return "";
XPath xpath = XPath.newInstance("//url-path/text()");
String path = ((Text) xpath.selectSingleNode(e)).getText();
e = request("sco-shortcuts", null);
xpath = XPath.newInstance("//domain-name/text()");
String url = ((Text) xpath.selectSingleNode(e)).getText();

    return url + "/" + path.substring(1) + "?session=" + breezesession;
} catch (JDOMException jde) {
    throw new XMLApiException(PARSE_ERROR, jde);
}

You can also use string pattern matching to check for a status code of ok. A successful action always returns this response:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="ok" />
</results>
```

You can check the response for the pattern ok or code="ok".

**Parse an error response**

When an API action completes successfully, it returns a status code of ok. If the call is not successful, it can also return any of the following status codes:

- **invalid** Indicates that the call is invalid in some way, usually invalid syntax.
- **no-access** Shows that the current user does not have permission to call the action, and includes a subcode attribute with more information.
- **no-data** Indicates that there is no data available for the action to return, when the action would ordinarily return data.
- **too-much-data** Means that the action should have returned a single result but is actually returning multiple results.

When the status code is invalid, the response also has an invalid element that shows which request parameter is incorrect or missing:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="invalid">
        <invalid field="has-children" type="long" subcode="missing" />
    </status>
</results>
```

When the status code is no-access, the subcode explains why:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="no-access" subcode="denied" />
</results>
```

All valid values for code, subcode, and invalid are described in status, in the API reference. Your application needs to read and handle status codes and subcodes.
**Handle status codes**

1. Write a method that parses an XML API response for the status code and subcode. This is an example in Java:

   ```java
   private String getStatus(Element el) throws JDOMException {
       String code = codePath.valueOf(el);
       String subcode = subcodePath.valueOf(el);
       StringBuffer status = new StringBuffer();
       if (null != code && code.length() > 0)
           status.append(code);
       if (null != subcode && subcode.length() > 0)
           status.append(" - " + subcode);
       return status.toString();
   }
   ```

2. When you call an action, parse the response for the status.

3. If the status is not ok, return a null value, display the error status code for debugging, or throw an application exception.

   The action to take depends on which call you are making and how your application is designed.

**Log a user out**

When a user logs out, the user's session ends, and Acrobat Connect Pro invalidates the BREEZESSESSION cookie by setting it to null and using an expiration date that has passed. For example, if you call logout on August 29, 2006, you see this `Set-Cookie` method in the response header, setting an empty cookie value and an expiration date a year earlier:

   ```
   Set-Cookie: BREEZESSESSION=;domain=.macromedia.com;expires=Mon, 29-Aug-2005 22:26:15 GMT;path=/
   ```

If you are managing the BREEZESSESSION cookie, invalidate the value so it is not reused after a user logs out.

**Log a user out and invalidate the session cookie**

1. Call `logout` to log the user out:

   ```
   https://example.com/api/xml?action=logout
   ```

2. Parse for a status code of ok to make sure the logout was successful.

3. Set the cookie value to null or otherwise invalidate it. For example, in this Java code snippet, the `breezesession` variable stores the cookie value and is set to null:

   ```java
   public void logout() throws XMLApiException {
       request("logout", null);
       this.breezesession = null;
   }
   ```
Chapter 3: Basics

Now that you understand the architecture of Adobe® Acrobat® Connect™ Pro and how to test an API call in a browser, you are ready to send XML API requests and analyze responses.

To get started with Acrobat Connect Pro Web Services, you need to understand three key concepts:

- Principals, who are users and groups
- SCOs, which are Shareable Content Objects and represent meetings, courses, and just about any content that can be created on Acrobat Connect Pro. SCOs (pronounced sko, which rhymes with snow) are compatible with the industry standard Shareable Content Object Reference Model (SCORM) specification and can be used with a Learning Management System (LMS).
- Permissions, which define how principals can act on objects

This chapter describes basic tasks you can do with Web Services, regardless of which Acrobat Connect Pro applications you have licensed. Many tasks are described as if you are running them in a browser. If you want to make the call from an application, translate the XML request to the language you are working in (for an example of how to do this in Java”, see “Send XML requests”).

Find a principal-id

A principal is a user or group that has a defined permission to interact with a SCO on the server. You can create users and groups for your organization and modify their permissions.

Acrobat Connect Pro also has built-in groups: Administrators, Limited Administrators, Authors, Training Managers, Event Managers, Learners, Meeting Hosts, and Seminar Hosts. You can add users and groups to built-in groups, but you can’t modify the permissions of built-in groups.

Note: The built-in groups that are available depend on your account.

Each Acrobat Connect Pro user and group has a principal-id. In some API calls, the principal-id is called a group-id or user-id to distinguish it from other values. The value of the ID that identifies a user or group is always the same, regardless of its name. You can check the syntax of any action in “Action reference” on page 54.

Get the principal-id of a user or group

1. Call principal-list with a filter:

   https://example.com/api/xml?action=principal-list&filter-name=jazz doe

   It is best to use filter-name, filter-login, or filter-email for an exact match. Be careful with filter-like-name, as it may affect server performance.

2. Parse the principal elements in the response for the principal-id:

   `<principal principal-id="2006282569" account-id="624520" type="user" has-children="false" is-primary="false" is-hidden="false">
   <name>jazz doe</name>
   <login>jazzdoe@example.com</login>
   <email>jazzdoe@newcompany.com</email>
   </principal>`

Get the principal-id of the current user

1. Call common-info after the user is logged in:
https://example.com/api/xml?action=common-info

2 Parse the user elements in the response for the user-id:

```xml
>User user-id="2007124930" type="user">
   <name>jazz doe</name>
   <login>jazz@doe.com</login>
</user>
```

Here, the principal-id is called user-id, because it always represents a user who is authenticated to Acrobat Connect Pro. A group cannot log in to the server. You can pass the user-id value as a principal-id in other actions.

## List principals or guests

A principal with a type of user is a registered Acrobat Connect Pro user, while a user with a type of guest has entered a meeting room as a guest. The server captures information about the guest and gives the guest a principal-id.

### List all principals on the server

1 Call principal-list with no parameters:

```xml
https://example.com/api/xml?action=principal-list
```

This call returns all Acrobat Connect Pro users, so be prepared for a large response.

2 Parse the principal elements in the response for the values you want:

```xml
<brincipal principal-id="2006282569" account-id="624520" type="user" has-children="false" is-primary="false" is-hidden="false">
   <name>jazz doe</name>
   <login>jazzdoe@example.com</login>
   <email>jazzdoe@newcompany.com</email>
</principal>
```

### List all guests on the server

1 Call report-bulk-users, filtering for a type of guest:

```xml
https://example.com/api/xml?action=report-bulk-users&filter-type=guest
```

2 Parse the row elements in the response:

```xml
<row principal-id="51157227">
   <login>joy@acme.com</login>
   <name>joy@acme.com</name>
   <email>joy@acme.com</email>
   <type>guest</type>
</row>
```

### List all users who report to a specific manager

When you call principal-info with a principal-id, the response shows the principal. If the principal is a user who has a manager assigned in Acrobat Connect Pro, the response also shows data about the principal's manager in a manager element:

```xml
<manager account-id="624520" disabled="" has-children="false" is-hidden="false" is-primary="false" principal-id="2006282569" type="user">
   <ext-login>jazzdoe@example.com</ext-login>
   <login>jazzdoe@example.com</login>
   <name>jazz doe</name>
</manager>
```
You can use the manager's principal-id with principal-list to list all users who are assigned to the manager.

1. Call principal-list, filtering on manager-id:
   ```
   https://example.com/api/xml?action=principal-list&filter-manager-id=2006282569
   ```

2. Parse the response for the principal elements:
   ```xml
   <principal principal-id="2006258745" account-id="624520" type="user" has-children="false" is-primary="false" is-hidden="false" manager-id="2006282569">
     <name>Pat Lee</name>
     <login>plee@mycompany.com</login>
     <email>plee@mycompany.com</email>
   </principal>
   ```

Create users

To create a new user, you need Administrator privilege. Adobe recommends that you create a user who belongs to the admins group for your application to use to make API calls that require Administrator privilege.

Create a new user and send a welcome e-mail

1. In your application, log in as an Administrator user.
   See "Log in from an application" for various ways to log in.

2. Call principal-update with at least these parameters:
   ```
   https://example.com/api/xml?action=principal-update
   &first-name=jazz&last-name=doe&login=jazz99@doe.com&password=hello
   &type=user&send-email=true&has-children=0&email=jazz99@doe.com
   ```
   The type must be user, has-children must be 0 or false, send-email must be true, and email must have a valid e-mail address.

   The server sends a welcome e-mail with login information to the user's e-mail address.

3. Parse the principal element in the response for the user's principal-id:
   ```xml
   <principal type="user" principal-id="2007184341" has-children="0"
   account-id="624520">
     <login>jammdoe@example.com</login>
     <ext-login>jammdoe@example.com</ext-login>
     <name>jammdoe</name>
   </principal>
   ```

Create a new user without using an e-mail address as a login ID

1. In Acrobat Connect Pro Central, navigate to Administration > Users and Groups > Edit Login and Password Policies. Make sure that Use E-mail Address as the Login is set to No.

2. In your application, log in as an Administrator user.

3. Call principal-update to create the new user, passing both login and email parameters:
   ```
   https://example.com/api/xml?action=principal-update&first-name=jazz
   &last-name=doe&login=jazz@email=jazzdoe@company.com
   &password=nothing&type=user&has-children=0
   ```
Parse the response for the `principal-id` of the new user:

```xml
<principal type="user" principal-id="2007184341" has-children="0"
   account-id="624520">
   <login>jazzdoe@example.com</login>
   <ext-login>jazzdoe@example.com</ext-login>
   <name>jazz doe</name>
</principal>
```

In the response, `ext-login` has the same value as `login` by default, until the user logs in successfully using external authentication (see “Log in using HTTP header authentication”).

## Update users

Once you create users, you often need to update their information. You can update standard fields that Acrobat Connect Pro defines for users by calling `principal-update` with the user's `principal-id`. The standard fields include `email`, `login`, `first-name`, and `last-name`.

If you have defined custom fields for the principal, use `acl-field-update` to update them.

You need Administrator privilege to update users, so your application must first log in as a user in the `admins` group. You cannot log in as the user and then have the user update his or her own profile.

### Update standard user information

1. Log in as an Administrator user.
2. Call `principal-list` with a filter to get the user's `principal-id` (see “Find a principal-id”).
3. Call `principal-update` to update the user:
   ```plaintext
   https://example.com/api/xml?action=principal-update
   &principal-id=2006258745&email=jazzdoe@newcompany.com
   ```
4. Parse the response for a status code of `ok`.

### Update custom field values for a user

1. Log in as an Administrator user.
2. Call `custom-fields` to get the `field-id` of the custom field:
   ```plaintext
   https://example.com/api/xml?action=custom-fields
   ```
3. Get the `principal-id`, `sco-id`, or `account-id` you want to update.
   This value is the `acl-id` you pass to `acl-field-update`.
4. Call `acl-field-update` to update the value of the custom field:
   ```plaintext
   https://example.com/api/xml?action=acl-field-update&field-id=x-2007396975&acl-id=2006258745&value=44444
   ```

## Create custom fields

Custom fields are additional data fields that you define. You can define up to eight custom fields on a principal or SCO using `custom-field-update`.

Once you define the custom field, by default you can set its value either by editing the value in Acrobat Connect Pro Central or by calling `custom-field-update`. 
To specify that the value can only be updated through the API, call `custom-field-update` with the parameter `object-type=object-type-read-only`.

**Define a custom field and set it on a user**

1. First, create the field with `custom-field-update`:
   ```
   https://example.com/api/xml?action=custom-field-update
   &object-type=object-type-principal
   &permission-id=manage
   &account-id=624520
   &name=Location
   &comments=adobe%20location
   &field-type=text
   &is-required=true
   &is-primary=false
   &display-seq=9
   ```

   The `name` field defines the field name as your application displays it, so use appropriate spelling and capitalization. The custom field in this example is defined for all Acrobat Connect Pro principals.

2. Parse the `field` element in the response for the `field-id`:
   ```
   <field field-id="2007184366" object-type="object-type-principal" display-seq="9" account-id="624520" is-primary="false" permission-id="manage" is-required="true" field-type="text">
     <comments>test</comments>
     <name>Country</name>
   </field>
   ```

3. Get the `principal-id` of the user (see “Find a principal-id”).

4. Call `acl-field-update` to set the value of the field, passing a `field-id`, the user's `principal-id` as `acl-id`, and a value:
   ```
   https://example.com/api/xml?action=acl-field-update
   &acl-id=2006258745
   &field-id=2007017474
   &value=San%20Francisco
   ```

5. Parse the response for a status code of `ok`.

**Create groups**

To add users to groups, you need to call `principal-update` as your application's Administrator user.

**Add a user to a group**

1. Log in as your application's Administrator user.

2. (Optional) If the user does not yet exist, create the user with `principal-update`:
   ```
   https://example.com/api/xml?action=principal-update
   &first-name=jazzwayjazz
   &last-name=doe
   &login=jazz@doe.com
   &password=nothing
   &type=user
   &has-children=0
   ```

3. (Optional) Parse the response for the new user's `principal-id`.

4. If the user already exists, call `principal-list` to get the user's `principal-id`:
   ```
   https://example.com/api/xml?action=principal-list&filter-type=user
   ```

5. Parse the response for the `principal-id`:
   ```
   <principal principal-id="5611980" account-id="624520" type="user">
     <name>Joy Black</name>
     <login>joy@acme.com</login>
     <email>joy@acme.com</email>
   </principal>
   ```

6. Call `principal-list` again to get the group's `principal-id`:
Call group-membership-update with is-member=true to add the user to the group:

https://example.com/api/xml?action=group-membership-update
&group-id=4930296&principal-id=2006258745&is-member=true

- The principal-id is the user's principal-id.
- The group-id is the group's principal-id.
- The parameter is-member must be true.

Check whether a specific user is in a group

1 Call principal-list with a group-id, filter-is-member, and a filter that identifies the principal:

https://example.com/api/xml?action=principal-list&group-id=624523
&filter-is-member=true&filter-like-name=bob

2 Parse for a principal element in the response. A successful response looks like this:

```xml
<principal-list>
  <principal principal-id="624660" account-id="624520" type="user"
    has-children="false" is-primary="false" is-hidden="false">
    <name>Bill Jones</name>
    <login>bjones@acme.com</login>
    <email>bjones@acme.com</email>
    <is-member>true</is-member>
  </principal>
</principal-list>
```

If the user is not a group member, the principal-list element is empty:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <principal-list />
</results>
```

Check which users are in a group

1 To get the group's principal-id, call principal-list with filters:

https://example.com/api/xml?action=principal-list&filter-type=group
&filter-name=developers

With filter-type and filter-name, principal-list should return a unique match.

2 Parse the response for the principal-id:

```xml
<principal principal-id="2007105030" account-id="624520" type="group"
  has-children="true" is-primary="false" is-hidden="false">
  <name>developers</name>
  <login>developers</login>
  <is-member>false</is-member>
</principal>
```

3 Call principal-list again, with the principal-id as a group-id and filter-is-member=true:

https://example.com/api/xml?action=principal-list&group-id=2007105030
&filter-is-member=true

4 Parse the response for the principal elements:

```xml
<principal principal-id="5698354" account-id="624520" type="group"
  has-children="true" is-primary="false" is-hidden="false">
```
List all groups a user belongs to

1. Call `principal-list` with the user's `principal-id` and `filter-is-member=true`:
   ```
   https://example.com/api/xml?action=principal-list
   &principal-id=2006258745&filter-is-member=true
   ```

2. Parse the response for the `principal` elements:
   ```
   <principal principal-id="5698354" account-id="624520" type="group"
   has-children="true" is-primary="false" is-hidden="false">
   <name>Bob Jones</name>
   <login>bobjones@acme.com</login>
   <is-member>true</is-member>
   </principal>
   ```

Find SCOs

All objects on Acrobat Connect Pro are Shareable Content Objects, or SCOs. The word `Shareable` comes from learning management systems in which content is combined into courses or curriculums and shared among them.

On the server, a SCO can be any content object that is combined with other content objects into a course or curriculum. Courses, curriculums, presentations, and other types of content are SCOs. Meetings, events, folders, trees, links, graphics files, or any other object are also SCOs.

Each SCO has a unique integer identifier called a `sco-id`. The `sco-id` is unique across the entire server. On a Acrobat Connect Pro hosted account, the `sco-id` is unique across all accounts.

Each SCO also has a type, such as `content`, `course`, `meeting`, and so on. You can see the `sco-id` and `type` values in the response from `sco-info` or other actions:

```
<sco account-id="624520" disabled="false" display-seq="0" folder-id="2006258747"
   icon="producer" lang="en" max-retries="" sco-id="2006334909"
   source-sco-id="" type="content" version="1">
```

Characteristics of SCOs

When you study the XML responses of various calls, you notice more characteristics of SCOs:

- A SCO's identifier is called a `sco-id` in some actions, but can also be called `folder-id`, `acl-id`, or another name in other actions. It's the same unique ID.
- Each SCO can be accessed by various principals, either users or groups. The specific principals who can access a SCO are defined in access control lists, or ACLs.
- Each SCO has a unique URL, with two parts: a domain name (like `http://example.com`) and an URL path (like `/f2006123456/`). You can concatenate these to form the full URL that accesses the SCO.
- Each SCO has a navigation path that describes where it resides in the folder hierarchy.
- Each SCO has a permission defined for each principal who can access it.
- Some SCOs have description fields, which are text strings that give you information about the SCO.

Often you need to find the ID of a SCO or some information about it. SCOs are arranged in a specific folder hierarchy where folders have names that indicate whether they are at the top level, contain shared content or templates, or hold user content and templates.
When you call `sco-shortcuts`, it returns a list of folders. Notice that folders have different types:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <shortcuts>
    <sco tree-id="624530" sco-id="2006258751" type="my-meeting-templates">
      <domain-name>http://example.com</domain-name>
    </sco>
    <sco tree-id="624530" sco-id="2006258750" type="my-meetings">
      <domain-name>http://example.com</domain-name>
    </sco>
    <sco tree-id="624529" sco-id="624529" type="meetings">
      <domain-name>http://example.com</domain-name>
    </sco>
    <sco tree-id="624530" sco-id="624530" type="user-meetings">
      <domain-name>http://example.com</domain-name>
    </sco>
    ...
  </shortcuts>
</results>
```

The folders shown in this example happen to be for meetings, but folders for other types of SCOs follow a similar pattern. Each folder type stores certain types of objects, with certain access privileges, as follows:

- **content, courses, meetings, events, seminars** These are shared folders, such as Shared Meetings, Shared Training, and so on. The Acrobat Connect Pro Administrator has access to this folder. The Administrator can assign Manage permission to any user, but only members of the built-in group associated with the folder can create new content or meetings within it.
- **user-content, user-meetings, user-courses, user-events** These folders each contain a folder for each user who can create content within it (for example, one folder for each meeting host or training developer).
- **my-courses, my-events, my-meetings, my-meeting-templates, my-content** Users create their own content in these folders and have Manage permission on the content. For example, meeting hosts create meetings in their `my-meetings` folder and have Manage permission on those meetings.
- **shared-meeting-templates** This folder is within the Shared Meetings folder, contains meeting templates, and inherits permissions from Shared Meetings.

You can list the contents of any folder to get information about a specific SCO. When you need to search for a SCO but do not have a `sco-id`, move through folders using `sco-shortcuts` and `sco-expanded-contents`. Do not use `sco-search`, as it returns only certain types of SCOs.

**Find a SCO when you do not know the sco-id**

1. Call `sco-shortcuts` to get a list of root folders on Acrobat Connect Pro:
   ```
   https://example.com/api/xml?action=sco-shortcuts
   ```
2. Parse the response for a type of the root folder that would logically contain the SCO, for example, `my-courses` for a course the user has created.
3. Parse the resulting `sco` element for a sco-id:
   ```
   <sco tree-id="4930295" sco-id="2006258748" type="my-courses">
     <domain-name>http://example.com</domain-name>
   </sco>
   ```
4. Create a call to `sco-expanded-contents` to list the contents of the folder, adding an exact match filter, if possible:
   ```
   https://example.com/api/xml?action=sco-expanded-contents
   &sco-id=2006258748&filter-name=All About Web Communities
   ```
You have several choices of filters:

- An exact match filter on name or url-path (like filter-name or filter-url-path), if you know the name or URL of the SCO.
- A greater-than or less-than date filter (filter-gt-date or filter-lt-date) on date-begin, date-created, or date-modified, if you know one of those dates.
- A partial name filter (like filter-like-name), if you do not know the exact SCO name. However, using this filter might affect system performance.

5 Parse the response for the sco-id:

```xml
<sco depth="1" sco-id="2006745671" folder-id="2006258748" type="folder"
    icon="folder" lang="en" source-sco-id="2006745669" display-seq="0"
    source-sco-type="14">
    <name>A Day in the Life Resources</name>
    <url-path>/f28435879/</url-path>
    <date-created>2006-06-12T14:47:59.903-07:00</date-created>
    <date-modified>2006-06-12T14:47:59.903-07:00</date-modified>
</sco>
```

Get information about a SCO

1 Call sco-info with the sco-id:

https://example.com/api/xml?action=sco-info&sco-id=2006745669

2 Parse the response for name, url-path, or any other value:

```xml
<sco account-id="624520" disabled="" display-seq="0"
    folder-id="2006258748" icon="curriculum" lang="en" max-retries=""
    sco-id="2006745669" source-sco-id="" type="curriculum" version="0">
    <date-begin>2006-06-12T14:45:00.000-07:00</date-begin>
    <date-created>2006-06-12T14:47:59.903-07:00</date-created>
    <date-modified>2006-06-12T14:47:59.903-07:00</date-modified>
    <name>A Day in the Life</name>
    <url-path>/day/</url-path>
</sco>
```

Construct the URL to a SCO

1 Call sco-shortcuts:

https://example.com/api/xml?action=sco-shortcuts

2 Parse the response for the domain-name value in any sco element:

```xml
<sco tree-id="624530" sco-id="2006258750" type="my-meetings">
    <domain-name>http://example.com</domain-name>
</sco>
```

3 Call sco-info with the sco-id:

https://example.com/api/xml?action=sco-info&sco-id=2006334909

4 Parse the response for the url-path:

```xml
<sco account-id="624520" disabled="" display-seq="0"
    folder-id="2006258747" icon="producer" lang="en"
    max-retries="" sco-id="2006334909" source-sco-id="" type="content" version="1">
    <date-created>2006-05-16T15:22:25.703-07:00</date-created>
    <date-modified>2006-05-16T15:22:25.703-07:00</date-modified>
    <name>Test Quiz</name>
    <url-path>/quiz/</url-path>
    <passing-score>10</passing-score>
    <duration>15100.0</duration>
```

```xml
</sco>
```
The `url-path` has both leading and trailing slashes. You can take the `url-path` from `report-my-meetings`, `report-my-training`, or any call that returns it.

5. Concatenate the `url-path` with the `domain-name`:
   
   ```
   http://example.com/f2006258748/
   ```

### Download files

You can download zip files from Acrobat Connect Pro to a user's local computer. A zip file is a SCO. To download it, you need to construct a download URL to the zip file, which looks like this:

```
http://server-domain/url-path/output/url-path.zip?download=zip
```

You probably already know the domain name of your server (such as `example.com`). If you do not, you can get it by calling `sco-shortcuts`.

#### Download a zip file from the server

1. Call `sco-shortcuts`:
   
   ```
   https://example.com/api/xml?action=sco-shortcuts
   ```

2. Extract any `domain-name` value from the response:
   
   ```
   http://example.com
   ```

3. Call `sco-info` with the `sco-id` of the zip file:
   
   ```
   https://example.com/api/xml?action=sco-info&sco-id=2006258747
   ```

   The SCO is the entire zip file.

4. Parse the response for the `url-path` element:
   
   ```
   <sco account-id="624520" disabled="" display-seq="0" folder-id="624522"
        icon="folder" lang="en" max-retries="" sco-id="2006258747"
        source-sco-id="" type="folder" version="1">
   <date-created>2006-04-18T10:21:47.020-07:00</date-created>
   <date-modified>2006-04-18T10:21:47.020-07:00</date-modified>
   <name>joy@acme.com</name>
   <url-path>/f124567890/</url-path>
   </sco>
   ```

5. Construct the download URL, for example:
   
   ```
   https://example.com/quiz/output/quiz.zip?download=zip
   ```

   Be sure to remove the trailing slash from the `url-path` value before adding `.zip` to it (so you have a value like `/quiz.zip`, not `/quiz/.zip`).

### Check permissions

Permissions define the ways in which a principal can interact with a SCO.

A permission mapping, indicating what permissions a principal has for a particular SCO, is called an access control list or ACL. An ACL consists of three pieces of information:

- The ID of a principal (`principal-id`).
• The ID of a SCO, account, or principal being acted on. In permission calls, it's called an acl-id. In other calls, the ID might be called a sco-id, account-id, or principal-id.
• A keyword that indicates the permission level the principal has, which is one of the valid values in permission-id.

**Check the permission a principal has on a SCO**

1. Call permissions-info with both an acl-id and principal-id:
   ```
   https://example.com/api/xml?action=permissions-info&acl-id=2006334909
   &principal-id=2006258745
   ```
   To check for permissions on a SCO, the acl-id is a sco-id. The acl-id can also be a principal-id or account-id.

2. Parse the response for a permission-id:
   ```
   <?xml version="1.0" encoding="utf-8" ?>
   <results>
     <status code="ok" />
     <permission acl-id="2007035246" permission-id="view",
     principal-id="2006258745" />
   </results>
   ```
   If a principal does not have an explicit permission to the SCO (in other words, if permission-id=""), the principal's permissions on the SCO's parent object apply.

**Check all principals' permissions on a SCO**

1. Call permissions-info with an acl-id, but no principal-id:
   ```
   https://example.com/api/xml?action=permissions-info&acl-id=2006293572
   ```

2. Iterate through the principal elements and parse them for permission-id values:
   ```
   <principal principal-id="2596608" is-primary="false" type="user"
   has-children="false" permission-id="view">
     <name>Jay Arnold</name>
     <login>jay@example.com</login>
   </principal>
   ```
   The valid permission values are listed in permission-id.
Chapter 4: Meetings

Custom applications can display, create, and delete Adobe® Acrobat® Connect™ Pro meetings in a web application, portal, or other environment. If you are just starting to design your meeting application, you may want to refer to the sample application on the product CD (and also check “A sample application” on page 219).

When users click a meeting room URL, they enter Acrobat Connect Pro, which hosts the meeting room. Acrobat Connect Pro Server then streams audio, video, and rich media content to the meeting room users.

Adobe recommends the following actions for meeting applications:

- **report-my-meetings** To display a user's meetings.
- **sco-update** To create a meeting room or update information about it.
- **permissions-update** To add a host, presenter, and participants to a meeting.
- **report-bulk-consolidated-transactions** To calculate meeting usage, especially the amount of time each user has spent in the meeting.
- **report-quiz-interactions** To get the results of a meeting poll.

Some actions that handle meetings require Administrator privilege, as noted in the task instructions. Create an Acrobat Connect Pro user who is a member of the **admins** group for your application to use to make these calls.

Find meetings

You often need to locate the **sco-id** of a meeting so that you can invite users, get report information about it, or update it in some other way.

You should understand the structure of folders in which meetings can be stored. By default, meetings are stored in the host's My Meetings folder (called **my-meetings** in the API). For more details on the folder structure, see “Characteristics of SCOs.”

**Find the sco-id of a meeting**

1. **Call sco-shortcuts:**
   
   https://example.com/api/xml?action=sco-shortcuts

2. **Parse the response for the sco-id of a meetings folder that is likely to contain the meeting:**
   
   `<sco tree-id="624530" sco-id="624530" type="user-meetings">
   <domain-name>http://example.com</domain-name>
   </sco>
   
   The folder name should be meetings, user-meetings, or my-meetings. Use a folder as far down the tree as you can.

3. **Call sco-contents on the folder, adding a filter or two to reduce the response:**

   https://example.com/api/xml?action=sco-contents&sco-id=2006258750
   &filter-type=meeting&filter-name=Intro to Film
   
   - The more specific you can make the filters, the better. Good filters to use are **filter-name**, **filter-url-path**, or a date filter. Be careful with using **filter-like-name**, as it might affect system performance.
   - You can also call **sco-expanded-contents** to list subfolders and their contents. However, **sco-contents** is better for server performance, if you know the **sco-id** of the folder that contains the meeting.
Parse the response for the sco-id of the meeting:

```xml
<sco sco-id="2006743452" source-sco-id="-1625529" folder-id="2006258750"
    type="meeting" icon="meeting" display-seq="0" is-folder="0">
  <name>Intro to Film</name>
  <url-path>/film/</url-path>
  <date-begin>2006-06-09T14:00:00.000-07:00</date-begin>
  <date-end>2006-06-09T20:00:00.000-07:00</date-end>
  <date-modified>2006-06-09T14:07:13.767-07:00</date-modified>
  <duration>06:00:00.000</duration>
</sco>
```

**List all meetings on the server**

1. Call `report-bulk-objects` with `type=meeting`:

   ```plaintext
   https://example.com/api/xml?action=report-bulk-objects&filter-type=meeting
   ```

   The response has a row element for each meeting, showing the meeting URL, name, and dates:

   ```xml
   <row sco-id="2007372149" type="meeting">
     <url>/monday/</url>
     <name>Monday Staff Meeting</name>
     <date-created>2006-12-18T14:15:00.000-08:00</date-created>
     <date-end>2006-12-19T02:15:00.000-08:00</date-end>
     <date-modified>2006-12-18T17:38:11.660-08:00</date-modified>
   </row>
   ```

**Display meetings**

In your application, you might want to lists of Acrobat Connect Pro meetings, such as a user’s present or future scheduled meetings.

An application workflow might log a user in and display the user's meetings, or it might add the user to a meeting and then display meetings. Displaying the user's meetings means listing the contents of the `my-meetings` folder.

**Display a user's meetings**

1. Log the user in (see “Log in from an application”).
2. Call `report-my-meetings` to list the user's meetings:

   ```plaintext
   https://example.com/api/xml?action=report-my-meetings
   ```

   You can add a filter to reduce the response. For example, you can exclude meetings that have ended:

   ```plaintext
   https://example.com/api/xml?action=report-my-meetings
   &filter-expired=false
   ```

3. Parse the response for values from the `meeting` elements:

   ```xml
   <meeting sco-id="2007063179" type="meeting" icon="meeting" permission-id="host" active-
     participants="0">
     <name>September All Hands Meeting</name>
     <domain-name>example.com</domain-name>
     <url-path>/sept15/</url-path>
     <date-begin>2006-09-15T14:00:00.000-07:00</date-begin>
     <date-end>2006-09-15T18:00:00.000-07:00</date-end>
     <expired>false</expired>
     <duration>09:00:00.000</duration>
   </meeting>
   ```

4. Create the URL to the meeting room by concatenating `http://` or `https://`, `domain-name`, and `url-path`. 

(continued)
Add a user to a meeting and display meetings
1. Log in as your application's Administrator user.
2. Get the user's principal-id (see "Find a principal-id").
3. Get the sco-id of the meeting (see "Find the sco-id of a meeting").
4. Call permissions-update to add the user to the meeting:
   https://example.com/api/xml?action=permissions-update
   &acl-id=2006258765&principal-id=2006258745&permission-id=view
   Use a permission-id of view for a participant, mini-host for presenter, or host for a meeting host.
5. Log out as the Administrator user, and log in as the user you just added to the meeting.
6. Display the user's current meetings:
   https://example.com/api/xml?action=report-my-meetings
   &filter-expired=false

Create meeting room URLs
You have several choices of how to construct the URL to a meeting room. The best action to call depends on how your application is logged in and where you are in your application workflow.

By default, the meeting room is created in the host's my-meetings folder.

Create the URL to a meeting room for which the user is host
1. If you are logged in as a user, and you want to create a URL to a meeting in the user's my-meetings folder, call report-my-meetings:
   https://example.com/api/xml?action=report-my-meetings
2. Parse the response for the values of domain-name and url-path:
   <meeting sco-id="2007063179" type="meeting" icon="meeting"
      permission-id="host" active-participants="0"><name>September All Hands Meeting</name>
      <domain-name>example.com</domain-name>
      <url-path>/sept15/</url-path>
      <date-begin>2006-09-15T09:00:00.000-07:00</date-begin>
      <date-end>2006-09-15T18:00:00.000-07:00</date-end>
      <expired>false</expired>
      <duration>09:00:00.000</duration>
   </meeting>
3. Concatenate the two values and add http:// or https:// at the beginning:
   https://example.com/online/
   If you are using HTTPS and you do not explicitly add https://, the URL defaults to http://, and the user might not be able to access the meeting room.

Create the URL to a meeting room for which the user is not host
1. Get the sco-id of the meeting (see "Find the sco-id of a meeting").
2. Call sco-info with the sco-id:
   https://example.com/api/xml?action=sco-info&sco-id=2006258750
3. Parse the response for the url-path:
   <sco account-id="624520" disabled="" display-seq="0" folder-id="624530"
Create meetings

A user must be an Administrator to create a Acrobat Connect Pro meeting, which means the user is a member of the Meeting Hosts group. In the response from principal-list, this group has type=live-admins.

A meeting can be public, protected, or private, and to create each, you need to set a specific combination of principal-id and permission-id:

- Public, equivalent to Anyone who has the URL for the meeting can enter the room
  principal-id=public-access&permission-id=view-hidden

- Protected, equivalent to Only registered users and accepted guests can enter the room
  principal-id=public-access&permission-id=remove

  If a meeting is protected, registered users invited as meeting participants can enter by clicking the meeting room URL and logging in. Users who are not invited can log in as guests. The meeting host receives a guest's request to enter (known as knocking) and can accept or decline.

- Private, which is equivalent to Only registered users and participants can enter. The login page does not allow guests to log in.
  principal-id=public-access&permission-id=denied

Create a public meeting and add host, presenter, and participants

1. Call principal-list to check that the user creating the Acrobat Connect Pro meeting is a member of the live-admins group:
   https://example.com/api/xml?action=principal-list&group-id=624523&filter-is-member=true&filter-like-name=bob

2. Call sco-shortcuts to obtain the sco-id of the user's my-meetings folder:
   https://example.com/api/xml?action=sco-shortcuts

3. Parse the response for the sco element with type=my-meetings:
   <sco tree-id="624530" sco-id="2006258750" type="my-meetings">
     <domain-name>http://example.com</domain-name>
   </sco>

4. Call sco-update to create the meeting room:
   https://example.com/api/xml?action=sco-update&type=meeting&name=October All Hands Meeting&folder-id=2006258750&date-begin=2006-10-01T09:00&date-end=2006-10-01T17:00&url-path=october
The folder-id is the sco-id of the user's my-meetings folder.

5 Parse the response for the sco-id of the new meeting:

```xml
<sco folder-id="2006258750" lang="en" account-id="624520"
  type="meeting" icon="meeting" sco-id="2007184134" version="0">
  <date-begin>2006-10-01T09:00</date-begin>
  <date-end>2006-10-01T17:00</date-end>
  <url-path>/october/</url-path>
  <name>October All Hands Meeting</name>
</sco>
```

You might want to store the url-path to the meeting, if you plan to create a URL to the meeting room later.

6 Call permissions-update to make the meeting public. Use the sco-id of the meeting as the acl-id:

```
https://example.com/api/xml?action=permissions-update&acl-id=2007018414
```

7 Call permissions-update to add a host, a presenter, and participants:

```
https://example.com/api/xml?action=permissions-update
  &principal-id=2006258745&acl-id=2007018414&permission-id=host
```

- Use a permission-id of host for the meeting host.
- Use mini-host for the presenter.
- Use view for meeting participants.
- You can specify multiple trios of principal-id, acl-id, and permission-id on one call to permissions-update.

8 Create the URL to the meeting room (see “Create meeting room URLs”).

Create a private meeting and add host, presenter, and participants

1 Log in as your application's Administrator user.

2 Follow the steps for creating a public meeting, but set the meeting permission to private:

```
https://example.com/api/xml?action=permissions-update&acl-id=2007018414
  &principal-id=public-access&permission-id=denied
```

3 Call permissions-update again to add a host, a presenter, and guests.

4 Create the URL to the meeting room (see “Create meeting room URLs”).

Create customized meetings

When you create a Acrobat Connect Pro meeting, you can assign it a meeting room template that creates a custom layout for the meeting room. If you don't assign a template, the meeting room is created with the default meeting template.

To edit a meeting room template, launch Acrobat Connect Pro Central and click the template's URL. You can edit the template while it is in a meeting templates folder (either My Templates or Shared Templates), if you have edit privileges on the folder.

Create a meeting room using a template

1 Log in as your application's Administrator user.

2 Call sco-shortcuts:

```
https://example.com/api/xml?action=sco-shortcuts
```
3 Parse the response for the sco-id of a folder that contains meeting templates:

```xml
<sco tree-id="624529" sco-id="-625529" type="shared-meeting-templates">

</sco>
```

```
<sco tree-id="624530" sco-id="2006258751" type="my-meeting-templates">

</sco>
```

4 Call `sco-contents`, passing it the sco-id of the meeting templates folder:

https://example.com/api/xml?action=sco-contents&sco-id=2006258751

5 Parse the response for the sco-id of the meeting template you want.

6 Create the meeting using `sco-update`. Pass it the sco-id of the meeting template as a `source-sco-id`:

https://example.com/api/xml?action=sco-update&type=meeting

&name=August All Hands Meeting

&folder-id=2006258750

&date-begin=2006-08-01T09:00

&date-end=2006-08-01T17:00

&url-path=august

&source-sco-id=2006349744

7 Continue to set permissions for the meeting and add participants, host, and presenter (see "Create meetings").

8 Create the URL to the meeting room (see "Create meeting room URLs").

### Invite users to meetings

Once you create a Acrobat Connect Pro meeting and add participants and presenters, you may want to send invitations by e-mail. To send a meeting invitation, you need information about the meeting, including the meeting name, the host's name and e-mail address, the meeting room URL, the date and time of the meeting, and the participant's (or presenter's) name and e-mail address.

You can construct an e-mail message using any technique that works with your user interface. Extract specific information about the meeting using the following steps.

### Send an e-mail to meeting participants

1 Call `sco-info` with the meeting sco-id:

https://example.com/api/xml?action=sco-info&sco-id=2006334033

2 Parse the response for the meeting name, date, or other values:

```xml
<sco account-id="624520" display-seq="0"

folder-id="2006258750" icon="meeting" lang="en" max-retries=""

sco-id="2007063163" source-sco-id="-1625529" type="meeting"

version="0">

<name>August All Hands Meeting</name>

<url-path>/august/</url-path>

</sco>
```

3 Construct the URL to the meeting room (see “Create meeting room URLs”).

4 Call `permissions-info` to get the principal-id values of the presenters or participants, filtering on permission-id:

https://example.com/api/xml?action=permissions-info

&acl-id=2007018414&filter-permission-id=mini-host
• For a list of presenters, use permission-id=mini-host.
• For participants, use permission-id=view.

5 Parse the response for the principal-id values you want:

```xml
<principal principal-id="2006282569" is-primary="false" type="user"
          has-children="false" permission-id="view">
  <name>jazz doe</name>
  <login>jazzdoe@example.com</login>
</principal>
```

6 Call principal-info with the principal-id:

```xml
https://example.com/api/xml?action=principal-info
&principal-id=2006282569
```

7 Extract the name and email values from the response:

```xml
<principal account-id="624520" disabled="" has-children="false"
          is-hidden="false" is-primary="false" principal-id="2006282569"
          type="user">
  <ext-login>jazzdoe@example.com</ext-login>
  <login>jazzdoe@example.com</login>
  <name>jazz doe</name>
  <email>jazzdoe@newcompany.com</email>
  <first-name>jazz</first-name>
  <last-name>doe</last-name>
  <x-2006293620>E3612</x-2006293620>
  <x-2007017651>San Francisco</x-2007017651>
</principal>
```

8 Call permissions-info again, filtering on a permission-id of host:

```xml
https://example.com/api/xml?action=permissions-info&acl-id=2007018414
&filter-permission-id=host
```

9 Parse the response for the principal-id:

```xml
<principal principal-id="2006282569" is-primary="false" type="user"
          has-children="false" permission-id="host">
  <name>jazz doe</name>
  <login>jazzdoe@example.com</login>
</principal>
```

10 Call principal-info, using the principal-id:

```xml
https://example.com/api/xml?action=principal-info
&principal-id=2006258745
```

11 Parse the principal element of the response for the name and login (or name and email):

```xml
<principal account-id="624520" disabled="" has-children="false"
          is-hidden="false" is-primary="false" principal-id="2006282569"
          type="user">
  <ext-login>jazzdoe@example.com</ext-login>
  <login>jazzdoe@example.com</login>
  <name>jazz doe</name>
  <email>jazzdoe@newcompany.com</email>
  <first-name>jazz</first-name>
  <last-name>doe</last-name>
  <x-2006293620>E3612</x-2006293620>
  <x-2007017651>San Francisco</x-2007017651>
</principal>
```

These are for the sender of the e-mail, who is the meeting host.
Remove users from meetings

Occasionally a user is invited to an Acrobat Connect Pro meeting as participant or presenter but later needs to be removed from the participant list. Removing the user has various results, depending on whether the meeting is public or private:

- **For a public meeting**  The user’s permission (participant, presenter, or host) is removed, but the user can still enter the meeting as a guest.
- **For a private meeting**  The user’s permission is removed, and the user can enter only as a guest and with approval from the meeting host.

To remove a user’s permission to enter, call `permissions-update` with a special permission value, `permission-id=remove`.

If the meeting is in progress and the user has already entered the room, the user is not removed from the meeting. However, when the user’s session times out, the user cannot reenter.

Remove a user’s permission to access a meeting

1. (Optional) Call `permissions-info` to check the principal’s permission to enter the meeting:  
   https://example.com/api/xml?action=permissions-info&acl-id=2007018414
   
   However, you do not need to know the specific permission the principal has before you remove the permission.

2. Get the meeting’s `sco-id` (see “Find the `sco-id` of a meeting”).

3. Get the user’s `principal-id` (see “Find a `principal-id`”).

4. Call `permissions-update`, using the meeting’s `sco-id` as the `acl-id` and `permission-id=remove`:  
   https://example.com/api/xml?action=permissions-update&acl-id=2007018414&principal-id=2006258745&permission-id=remove

Calculate meeting usage

Once you create users and Acrobat Connect Pro meetings, you may need to calculate meeting usage. Meeting usage is often calculated in one of these ways:

- The time each user spends in a specific meeting, in minutes per user
- The number of concurrent meeting participants

The time a user spends in a meeting is measured by a transaction, which is the interaction between a principal and a SCO (in this case, between a user and a meeting). The date and time a transaction begins and ends are returned by `report-bulk-consolidated-transactions`.

Calculate time spent in meetings per user

1. Call `report-bulk-consolidated-transactions`, filtering for meetings and another value to identify the meeting, such as a date:  
   
   - The second filter can be for the date the transaction began or ended, the `principal-id` of the user, the `sco-id` of a specific meeting, or another valid filter that meets your needs.
   - This call returns all transactions that meet the filter criteria. Be prepared for a large response.
   - The call also returns only users who logged in to the meeting as participants, not users who entered as guests.
2 Parse the row elements in the response for date-created and date-closed:

```xml
<row transaction-id="2007071217" sco-id="2007071193" type="meeting" principal-id="2007003123" score="0">
  <name>Thursday Meeting</name>
  <url>/thursday/</url>
  <login>jazz@doe.com</login>
  <user-name>jazzwayjazz doe</user-name>
  <status>completed</status>
  <date-created>2006-08-03T12:33:48.547-07:00</date-created>
  <date-closed>2006-08-03T12:34:04.093-07:00</date-closed>
</row>
```

3 In your application, calculate the time difference between the two dates.

One way to do this (in Java™) is to write a utility method that converts the ISO 8601 datetime values returned in the response to a `GregorianCalendar` object. Then, convert each `GregorianCalendar` date to milliseconds, calculate the difference between the creation and closing times, and convert the difference to minutes.

4 Repeat for all the meeting transactions that meet your criteria, and total the meeting usage times.

### Check meeting quotas

The number of concurrent meeting participants you can have is determined by your Acrobat Connect Pro license. To check your quota for the number of concurrent meeting participants, call `report-quotas` and look for the quota named `concurrent-user-per-meeting-quota` in the response:

```xml
<quota acl-id="624529" quota-id="concurrent-user-per-meeting-quota" used="0"
  limit="unlimited" soft-limit="1000000000">
  <date-begin>2004-03-09T09:45:02.297-08:00</date-begin>
  <date-end>2999-12-31T16:00:00.000-08:00</date-end>
</quota>
```

The quota has both a limit and a soft limit. The soft limit is the concurrency limit purchased for the account. It is the same as the limit, unless you purchase a Burst Pack for meetings, which allows additional participants to join past the limit, on an overage basis.

Without a Burst Pack, Acrobat Connect Pro enforces the concurrency limit and participants who try to enter after the quota is reached are rejected. If your limit is 20 attendees, attendee 21 receives a notice that the meeting room is full.

All accounts enforce the quotas that are set when the account is created. Accounts do not allow overages, unless you have a Burst Pack. Furthermore, Burst Packs are only for meetings, not for training or seminars.

#### Check your meeting concurrency quota and usage

1 Call `report-quotas` to check your quota for concurrent meeting users:

   https://example.com/api/xml?action=report-quotas

2 Parse the response for the `quota` element with a `quota-id` value of `concurrent-user-per-meeting-quota`.

3 Extract the value of `soft-limit`, the limit defined by your Acrobat Connect Pro license.

4 Call `report-meeting-concurrent-users` to check the peak number of concurrent meeting participants on your server or in your account:

   https://example.com/api/xml?action=report-meeting-concurrent-users

5 Parse the response for the `report-meeting-concurrent-users` element. Read the value of the `max-users` attribute and compare it to the value of `soft-limit`:  

```xml
<quota acl-id="624529" quota-id="concurrent-user-per-meeting-quota" used="0"
  limit="unlimited" soft-limit="1000000000">
  <date-begin>2004-03-09T09:45:02.297-08:00</date-begin>
  <date-end>2999-12-31T16:00:00.000-08:00</date-end>
</quota>
```

The quota has both a limit and a soft limit. The soft limit is the concurrency limit purchased for the account. It is the same as the limit, unless you purchase a Burst Pack for meetings, which allows additional participants to join past the limit, on an overage basis.

Without a Burst Pack, Acrobat Connect Pro enforces the concurrency limit and participants who try to enter after the quota is reached are rejected. If your limit is 20 attendees, attendee 21 receives a notice that the meeting room is full.

All accounts enforce the quotas that are set when the account is created. Accounts do not allow overages, unless you have a Burst Pack. Furthermore, Burst Packs are only for meetings, not for training or seminars.
Get meeting archives

A Acrobat Connect Pro meeting can have one or more recorded archives. If the meeting recurs weekly, for example, it might have an archive for each session.

A meeting archive is identified with type=content and icon=archive. The icon value works as a subcategory of type, to identify the type of content.

List archives for a meeting room

1. Get the sco-id of the meeting (see “Find the sco-id of a meeting”).
2. Call sco-expanded-contents with the sco-id and filter-icon=archive to list all archives associated with the meeting:
   
   https://example.com/api/xml?action=sco-contents&sco-id=2007018414
   &filter-icon=archive

3. Parse the response for the sco element and extract the information you want, such as name, date-created, or url-path:

   <sco sco-id="2598402" source-sco-id="" folder-id="2598379"
   type="content" icon="archive" display-seq="0" is-folder="0">
   <name>EN - Monday Night Football_0</name>
   <url-path>/p71144063/</url-path>
   <date-begin>2004-05-17T15:51:54.670-07:00</date-begin>
   <date-end>2004-05-17T15:54:52.920-07:00</date-end>
   <date-modified>2004-05-17T15:55:00.733-07:00</date-modified>
   <duration>00:02:58.250</duration>
</sco>

Get meeting poll results

To access the results of a poll used during a meeting, use report-quiz-interactions. This action returns all poll results, but you can use a filter to reduce the response.
Each multiple-choice response in the poll has an integer identifier, with the first response in the displayed list numbered 0, the second 1, and so on.

Get the results of a meeting poll
1. Be sure that the meeting host has closed the poll. The poll results are cached in the meeting until the poll is closed.
2. Get the sco-id of the meeting (see “Find the sco-id of a meeting”).
3. Call report-quiz-interactions, using the meeting's sco-id:
   https://example.com/api/xml?action=report-quiz-interactions
   &sco-id=2007071193
4. (Optional) Add a filter to reduce the response, for example:
   - filter-response=1 to check all users who made a specific response
   - filter-interaction-id=2007027923 to check all responses to a poll (a meeting might have several polls)
5. Parse the response for response, name, or any other values:

Launch meetings with external authentication

Once a user logs in to your network and you authenticate the user to the Acrobat Connect Pro Server using an external authentication credential, you may want to allow the user to enter a meeting as participant or guest without having to log in a second time to Acrobat Connect Pro.

Launch a meeting and let the user enter as participant
1. Once the user is authenticated on your network, log the user in to Acrobat Connect Pro (see Log in using HTTP header authentication for details).
2. Get the value of the BREEZESESSION cookie for the user's session, in one of two ways:
   - Call common-info and retrieve the value of cookie from the response:
Using Acrobat Connect Pro Web Services

1. Retrieve the value of the BREEZESESSION cookie from the response header:
   - Set-Cookie: BREEZESESSION=breezqw4vtfarqxfv9pk2;
     domain=.macromedia.com;path=/

2. Create a meeting room URL (see Create meeting room URLs for details).
3. Append a session parameter and the BREEZESESSION cookie value to the meeting room URL:
   - http://example.com/employeeMeeting/?session=breezbityp829r9ozv5rd

4. Open the meeting room URL that has session appended. One way to do this is with a JavaScript onClick command:
   - <a href="http://example.com/employeeMeeting/" onClick="javascript:window.open('http://example.com/employeeMeeting/?session=breezbityp829r9ozv5rd','Breeze', 'toolbar=no,menubar=no,width=800,height=600,resizable=yes'); return false">http://example.com/employeeMeeting/</a>

Launch a meeting and let the user enter as guest

1. Once the user is authenticated on your network, log the user in to Acrobat Connect Pro (see Log in using HTTP header authentication for details).
2. Get the value of the BREEZESESSION cookie for the user’s session, in one of two ways:
   - Call common-info and retrieve the value of cookie from the response:
     - <cookie>breezma6zor9rdfps8h6a</cookie>
   - Retrieve the value of the BREEZESESSION cookie from the response header after calling login:
     - Set-Cookie: BREEZESESSION=breezqw4vtfarqxfv9pk2;

3. In your application, create a meeting room URL (see Create meeting room URLs).
4. Append a guestname parameter and the user’s guest display name to the meeting room URL:
   - http://example.com/employeeMeeting/?guestName=joy

5. Open the meeting room URL that has the guestname parameter. One way to do this is with a JavaScript onClick command:
   - <a href="http://example.com/employeeMeeting/" onClick="javascript:window.open('http://example.com/employeeMeeting/?guestName=joy','Breeze', 'toolbar=no,menubar=no,width=800,height=600,resizable=yes'); return false">http://example.com/employeeMeeting/</a>

Configure compliance settings

Depending on your organization, you might need to configure your system to ensure compliance with governmental regulations and industry standards regarding communication. You can use Acrobat Connect Pro to monitor communication data in many ways. For example, you can disable the use of pods, set Acrobat Connect Pro to always or never record meetings, generate transcripts of chat sessions, create a notice that recording is taking place, and more. You can also control user access in several ways. For example, you can distinguish between authenticated and non-authenticated users, restrict access to meetings rooms based on roles, and block guest access to rooms. For more information, see Adobe Acrobat Acrobat Connect Pro User Guide.

When you change the settings for these features, the changes take effect when a new meeting is started or when the server is refreshed. The typical refresh interval is 10 minutes. The next meeting that starts after the server is refreshed reflects any new settings.
Changing certain settings through the XML API can affect the use of other features. For example, when the attendee list is disabled (`fid-meeting-people-list`), users cannot create breakout rooms. Therefore, to prevent user confusion, disable the breakout rooms feature at the same time.

**Disabling pods**

When you disable pods, the layout of a meeting room is affected and may have more empty white space than you want. Administrators can either resize remaining pods to occupy the empty space (the recommended approach), or create new meeting room templates. Otherwise, after a meeting starts, the host can manually resize pods as they see fit.

If a pod with persistent data, such a Chat pod, is disabled and then re-enabled between different sessions of the same meeting, the contents of the old pod are lost.

**Disable the Chat and Note pods**

1. Get the account ID for the account under which the meeting exists.
2. Log in using the administrative account.
3. Call `meeting-feature-update`, passing `fid-meeting-chat` and `fid-meeting-note` as arguments to the `feature-id` parameter, and setting the enable attribute for both parameters to `false`.
4. Refresh the server or start a new meeting to see the change.

The following code disables the Chat and Note pods:

```plaintext
http://localhost/api/xml?action=meeting-feature-update&account-id=7&feature-id=fid-meeting-chat&enable=false&feature-id=fid-meeting-note&enable=false
```

**Managing chat transcripts**

To configure Acrobat Connect Pro to generate chat transcripts, select Generate chat transcripts for all meetings in Acrobat Connect Pro Central or call `meeting-feature-update` with the `feature-id` `fid-chat-transcripts`.

To get a chat transcript, you need the sco-id of the chat session. Use a combination of XML APIs to get the sco-id of a specific transcript. You can then get the transcript from the following Acrobat Connect Pro directory: `[RootInstall]/content/account-id/transcript-sco-id/output/`.

**Get chat transcripts**

1. Get the sco-id of the chat transcripts tree by calling `sco-shortcuts`:
   
   [http://example.com/api/xml?action=sco-shortcuts&account-id=7]

2. Parse the response for the chat transcripts tree-id:
   
   `<shortcuts>
   <sco tree-id="10026" sco-id="2006258748" type="chat-transcripts">
   <domain-name>http://example.com</domain-name>
   </sco>
   ...
   </shortcuts>
   </results`

3. Get the list of chat transcripts for a particular meeting by calling `sco-contents` with the chat transcripts tree-id and the filter `source-sco-id`:


   In the example above, 10026 is the sco-id of the chat transcripts tree and 10458 is the sco-id of the meeting.  
   (You can get the sco-id of the meeting from the URL of the meeting information page.)
The list of SCOs that is returned represents the chat transcripts for the meeting.

4 Find the chat transcript in the Acrobat Connect Pro directory [RootInstall]/content/account-id/transcript-sco-id/output/.

**Forcing meetings to be recorded**

You can set up Acrobat Connect Pro to record all meetings. Adobe recommends that when meetings are recorded, you show a disclaimer to notify users that the meeting is being recorded.

**Force meetings to be recorded**

1 Disable the setting that lets hosts control recording (fid-archive) and enable automatic recording (fid-archive-force) by calling meeting-feature-update. Pass the two feature-id arguments:


2 See “Setting up disclaimer notices” on page 42.

3 Refresh the server or start a new meeting to see the change.

**Setting up disclaimer notices**

You can set up a disclaimer notice to appear when a user enters a meeting. A disclaimer notice typically displays boilerplate information for your organization. It advises users of the status of the meeting and the terms of use for the meeting. For example, a disclaimer notice could advise users that the meeting is being recorded, and that users cannot join the meeting unless they accept the notice. By default, this option is disabled.

**Set up a disclaimer notice**

1 Call meeting-disclaimer-update and set the text for the disclaimer notice:

   https://example.com/api/xml?action=meeting-disclaimer-update&account-id=7&disclaimer=Please note that this meeting is being recorded.

2 Call meeting-feature-update to activate the disclaimer:

   https://example.com/api/xml?action=meeting-feature-update&account-id=7&feature-id=fid-meeting-disclaimer&enable=true

3 Refresh the server or start a new meeting to see the change.

**Controlling share settings**

You can control settings related to the information that a user can share with other users during a meeting. Call meeting-feature-update and pass the appropriate feature ID or multiple feature IDs to enable or disable a share setting. For example, to disable screen sharing, call the following code:

https://example.com/api/xml?action=meeting-feature-update&account-id=7&feature-id=fid-meeting-desktop-sharing&enable=false

The following table lists the feature IDs for share settings. For a full list of feature IDs, see “feature-id” on page 204.

<table>
<thead>
<tr>
<th>Share setting</th>
<th>Feature ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Share a computer screen or control of the screen; share a document or white board</td>
<td>fid-meeting-desktop-sharing</td>
</tr>
<tr>
<td>Upload a document to the Share pod</td>
<td>fid-meeting-shared-upload</td>
</tr>
<tr>
<td>Upload and manage files using the File Share pod</td>
<td>fid-meeting-file-share</td>
</tr>
<tr>
<td>Share setting</td>
<td>Feature ID</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Share a white board</td>
<td>fid-meeting-white-board</td>
</tr>
<tr>
<td>Display web pages to attendees</td>
<td>fid-meeting-web-links</td>
</tr>
</tbody>
</table>
Chapter 5: Training

A custom training application or portal can access Adobe® Acrobat® Connect™ Pro Training to display training courses that are available, enroll users or allow them to self-enroll, list all courses and curriculums the user is enrolled in, and generate various reports. Acrobat Connect Pro Training has two types of training modules: courses and curriculums.

A course is content (for example, a presentation) that has a set of enrolled learners with usage tracking for each individual. The course can be delivered and administered independently or as part of a curriculum.

A curriculum is a group of courses and other learning content that moves students along a learning path. A curriculum contains primarily Acrobat Connect Pro Training courses, but may include other items such as content and meetings. As with courses, you can generate reports to track the progress of enrolled learners as they move through the curriculum. This way, you can ensure that enrollees meet the learning objectives.

Courses and content can both be modules within a curriculum, and a content object can be used in any number of courses and curriculums. In Acrobat Connect Pro Training, content objects, courses, and curriculums are all SCOs, and each has a unique sco-id. Content objects and courses are combinable and reusable, according to the SCORM standard.

As you develop training applications, Adobe recommends that you use the following XML API actions:

- **permissions-update** To enroll users in courses and make sure they have the appropriate permissions to access the course.
- **group-membership-update** To add users to groups if you want to enroll a group.
- **report-my-training** To list all courses and curriculums the current user is enrolled in, including the URL to access the course or curriculum.
- **report-curriculum-taker** To get details of a user's progress within a curriculum.
- **report-user-trainings-taken** To view the latest status of all of a user's courses and curriculums.
- **report-user-training-transcripts** To list all of a user's transcripts and scores.

These actions work on courses, curriculums, and training folders and use the permissions allowed for objects in the Training library.

Training library permissions

The Shared Training folder that you see in Acrobat Connect Pro Central is also called the Training library. Shared Training is called courses in the response from sco-shortcuts:

```xml
<sco tree-id="123456" sco-id="123456" type="courses">
  <domain-name>example.com</domain-name>
</sco>
```

Each folder, course, and curriculum in the library is a SCO. As you navigate the Training library, you see the sco-id of the current course or curriculum in the browser URL. You can also retrieve the sco-id by calling sco-contents or sco-expanded-contents on a folder in the Training library.

Each course, curriculum, or content object in the Training library has permissions that define which users can access it. As you design your application, be aware of these permission levels:

- **Enrollee permissions** Courses and curriculums have permissions that define which users are enrolled and can access them. The two permissions available are Enrolled and Denied.
**Training library permissions** Courses, curriculums, and folders in the Training library have either Manage or Denied permission. Manage permission means a user can create, delete, edit, or assign permissions. By default, users have Manage permission on their own training folders, and Administrators have Manage permission on any folder in the training library.

An Administrator can assign a user Manage permission on an individual course, curriculum, or folder with permissions-update or check the permissions a user has with permissions-info.

In XML API calls, you read, use, or set values of permission-id as you work with the Training library. These values of permission-id apply to courses and curriculums:

- **view** The user has access to the course or curriculum, and permission is Enrolled.
- **denied** The user is not allowed access, and permission is Denied.

You should also be aware of the permission a user has on a folder before executing an API call. Log in as a user with appropriate permission, or when needed, as your application’s Administrator user. These values of permission-id apply to training folders:

- **manage** The user can add, delete, change, or assign permissions to courses, curriculums, and content in a folder. The user can also list the contents of the folder with sco-contents or sco-expanded-contents.
- **denied** The user cannot add, delete, change, or assign permissions to anything in the folder, but can list the contents of the folder.

**Find courses and curriculums**

Most XML API actions that work with courses and curriculums require the sco-id of the course or curriculum. You often need to locate the sco-id dynamically, before you call another action, without knowing the exact name of the SCO.

Use these best practices to make searching for training SCOs efficient:

- Create specialized folders within the Shared Training folder for storing courses and curriculums. You can do this in Acrobat Connect Pro Central, or you can use the XML API, in which the Shared Training folder is named courses.
- Use these folders to store various categories of courses and curriculums, such as Marketing Training or Sales Training.
- Use a flat structure in the specialized folders, storing courses and curriculums one level deep.

This directory structure is also recommended when you want to display a list of all courses and curriculums (or all those in a subject area) and allow users to enroll themselves.

If you are working in Acrobat Connect Pro Central, you can find the sco-id of a course or curriculum by navigating to it, clicking its URL, and taking the value of sco-id from the browser URL. You can also locate the sco-id from an application, using the XML API.

**Find the sco-id of a course or curriculum**

1. Call sco-shortcuts:
   ```xml
call action=https://example.com/api/xml?action=sco-shortcuts
```

2. Parse the response for the sco-id of the courses folder:
   ```xml
<sco tree-id="624528" sco-id="624528" type="courses">
  <domain-name>http://example.com</domain-name>
</sco>```
You cannot use a filter with \texttt{SCO-Shortcuts}, but you can parse the response for the \texttt{SCO} element that has type=\texttt{courses}.

3 Call \texttt{SCO-Contents}, passing the \texttt{SCO-ID} of the \texttt{courses} folder and filtering for your specialized training folders:

\begin{verbatim}
https://example.com/api/xml?action=SCO-Contents&SCO-ID=624528
&filter-name=Sales Training
\end{verbatim}

- You can use \texttt{filter-name}, \texttt{filter-url-path}, another exact match filter, or a date filter. However, be careful when using \texttt{filter-like-name}, as it might affect server performance.
- You can also get the \texttt{SCO-ID} of your specialized training folder from the browser URL in Acrobat Connect Pro Central and pass it to \texttt{SCO-Contents}.

4 Parse the response for the \texttt{SCO-ID} of your specialized training folder:

\begin{verbatim}
<sco sco-id="2007122244" source-sco-id="" folder-id="624528"
   type="folder" icon="folder" display-seq="0" is-folder="1">
\end{verbatim}

5 Call \texttt{SCO-Contents}, passing it the \texttt{SCO-ID} of the specialized training folder and adding a filter that identifies the course or curriculum:

\begin{verbatim}
https://example.com/api/xml?action=SCO-Contents
&SCO-ID=2007122244&filter-name=Java 201
\end{verbatim}

- You can call \texttt{SCO-Contents}, rather than \texttt{SCO-Expanded-Contents}, if all courses and curriculums are stored at the top level of your specialized training folder. This improves performance.
- You can define custom fields for SCOs if it helps you identify them in searches (see “Create custom fields”).

6 Parse the \texttt{SCO} elements in the response for the \texttt{SCO-ID} of the course or curriculum:

\begin{verbatim}
<sco depth="2" sco-id="2006745673" folder-id="2006745671" type="content"
   icon="course" lang="en" source-sco-id="2006744233"
   display-seq="1" source-sco-type="0">
   <name>All About Web Communities</name>
   <url-path>/p33096345/</url-path>
   <description>Web 2.0 course</description>
   <date-created>2006-06-12T14:48:25.870-07:00</date-created>
   <date-modified>2006-06-12T14:48:25.870-07:00</date-modified>
</sco>
\end{verbatim}

**List all courses or curriculums available**

1 Get the \texttt{SCO-ID} of a specialized training folder you have created (see “Find the \texttt{SCO-ID} of a course or curriculum”). You can also get the \texttt{SCO-ID} by navigating to the folder in Acrobat Connect Pro Central, clicking its URL, and reading the \texttt{SCO-ID} in the browser URL.

2 Call \texttt{SCO-Contents}, passing the folder’s \texttt{SCO-ID}:

\begin{verbatim}
https://example.com/api/xml?action=SCO-Contents&SCO-ID=2006258748
\end{verbatim}

The best practice is to create the specialized training folders one level deep. By doing so, you can call \texttt{SCO-Contents} rather than \texttt{SCO-Expanded-Contents}. This gives better performance.

3 Parse the response for \texttt{name}, \texttt{url-path}, or any values you want to display:

\begin{verbatim}
<sco sco-id="2007035246" source-sco-id="2006334909"
   folder-id="2006258748" type="content" icon="course"
   display-seq="0" is-folder="0">
   <name>Java 101</name>
   <url-path>/java101/</url-path>
   <date-begin>2006-07-20T17:15:00.000-07:00</date-begin>
   <date-modified>2006-07-20T17:21:38.860-07:00</date-modified>
</sco>
\end{verbatim}
Create a course

You can use either Acrobat Connect Pro Central or Acrobat Connect Pro Web Services to create a course. If you use Web Services, first create an empty SCO and then add content to it.

1 Call `sco-update` to create a new SCO for the course:

   https://example.com/api/xml?action=sco-update&name=salescourse&folder-id=12345&icon=course&type=content

2 Parse the response for the `sco-id` value of the new course.

3 Add content to the new SCO, using the `sco-id` returned by `sco-update`:


4 Enroll users in the course (see “Enroll one user” and “Enroll a large number of users”).

View a user’s training

Once a user is logged in, you can list all courses the user is enrolled in with `report-my-courses`, or all of the user’s courses and curriculums with `report-my-training`. This lists only the courses (or courses and curriculums) the user is enrolled in, not all courses available.

View a user’s courses and curriculums

1 Log the user in (see “Log in from an application”).

2 Call `report-my-training` to list all courses and curriculums the user is enrolled in:

   https://example.com/api/xml?action=report-my-training

3 Parse the response for name, url, or any other values you want to display:

   <row sco-id="2007035246" type="content" icon="course" permission-id="view">
     <name>Java 101</name>
     <url>example.com/java101/</url>
     <date-created>2006-07-20T17:21:11.940-07:00</date-created>
     <date-modified>2006-07-20T17:21:38.860-07:00</date-modified>
     <date-begin>2006-07-20T17:15:00.000-07:00</date-begin>
     <url-path>/java101/</url-path>
     <expired>false</expired>
     <completed>false</completed>
   </row>

View the status of all of the user’s courses and curriculums

1 Get the `principal-id` of the user (see “Find a principal-id”).

2 Call `report-user-trainings-taken`:

   https://example.com/api/xml?action=report-user-trainings-taken
   &principal-id=2006258745

3 Parse the response for `status`:

   <row transcript-id="2006293632" max-retries="" sco-id="2564016" type="content" icon="course" status="completed">
     <name>Programming in Perl</name>
     <description>Info about Perl</description>
     <url-path>/p57283193/</url-path>
   </row>
A course can have many allowed values for status, but a curriculum can only have a status of completed or incomplete. The allowed values of status are described in status attribute in the reference.

Enroll one user

To give users access to training, Adobe recommends that you enroll them in courses. This gives the users appropriate permission to launch and complete the course, and it gives you usage tracking and access to various report actions. Courses differ from content. Courses are resumable and offer server-side review mode (for detailed information, see Adobe Acrobat Connect Pro User Guide).

Your application might allow users to self-enroll in courses, which involves calling permissions-update to enroll one user at a time. You may also want to write a workflow, which is a sequence of API calls, that creates a new user and enrolls the user in a course.

Enrolling users in training using the XML API (specifically, a call to permissions-update) does not send a notification. To send enrollment notifications, use Acrobat Connect Pro Central to enroll users.

Enroll one user in a course or curriculum

1. Get the sco-id of the course (see “Find the sco-id of a course or curriculum”).
2. Get the principal-id of the user (see “Get the principal-id of the current user”).
3. To enroll the user in the course, call permissions-update. Use the course sco-id as the acl-id, with a permission-id of view:
   
   https://example.com/api/xml?action=permissions-update
   &acl-id=2007035246&principal-id=2006258745&permission-id=view

4. Call report-my-training to list all courses and curriculums the user is enrolled in:
   
   https://example.com/api/xml?action=report-my-training

5. Parse the row elements in the response for values you want to display:
   
   <row sco-id="2007035246" type="content" icon="course"
   permission-id="view">
     <name>Java 101</name>
     <url>example.com/java101/</url>
     <date-created>2006-07-20T17:11.940-07:00</date-created>
     <date-modified>2006-07-20T17:11.860-07:00</date-modified>
     <date-begin>2006-07-20T17:15:00.000-07:00</date-begin>
     <url-path>/java101/</url-path>
     <expired>false</expired>
     <completed>false</completed>
   </row>

Enroll a new user by workflow

1. Call principal-update to create the new user and send a welcome e-mail:
   
   https://example.com/api/xml?action=principal-update&first-name=jazz
   &last-name=doe&login=jazz@doe.com&password=hello&type=user
   &send-email=true&has-children=0&email=jazz@doe.com

   To send the e-mail, make sure send-email=true.

2. Log the user in to the server:
Enroll a large number of users

When you enroll a large number of users in a course, first decide whether to enroll the users directly or create a group and enroll it. Adobe recommends these best practices for enrolling users in courses:

- Enroll users directly in courses using `permissions-update`, which allows you to enroll 1000, 10,000, or more users with a single API call.
- Add the users to a group and enroll it only if you plan to reuse the group (for example, to enroll it in multiple courses). In this case, you can add only 200 users at a time.

Enroll a large number of users (1000+) directly in a course

1. Get the `sco-id` of the course (see “Find the sco-id of a course or curriculum”).
2. Get the `principal-id` of each user you want to enroll.
   To do this, you can:
   - Call `principal-list` with filters to list the users you want to enroll:
     https://example.com/api/xml?action=principal-list&filter-type=user
     &filter-type=sales
   - Read the values from a file.
3. Write a method that calls `permissions-update` with multiple trios of `acl-id`, `principal-id`, and `permission-id`:
   https://example.com/api/xml?action=permissions-update

See “Log in from an application” for other ways to call `login`.

3. Call `group-membership-update` with `is-member=true` to add the user to the group:
   https://example.com/api/xml?action=group-membership-update
   &group-id=4930296&principal-id=2006258745&is-member=true

4. Call `permissions-update` to enroll the user in a curriculum:
   https://example.com/api/xml?action=permissions-update
   &acl-id=2006745669&principal-id=2007124930&permission-id=view
   Use a `permission-id` of `view`.

5. Call `report-my-training` to list courses and curriculums the user is enrolled in:
   https://example.com/api/xml?action=report-my-training

6. Parse the row elements in the response for values you want to display:
   `<row sco-id="2006745669" type="curriculum" icon="curriculum"
   permission-id="view">
   <name>A Day in the Life</name>
   <url>example.com/day/</url>
   <date-created>2006-06-12T14:47:59.903-07:00</date-created>
   <date-modified>2006-06-12T14:47:59.903-07:00</date-modified>
   <date-begin>2006-06-12T14:45:00.000-07:00</date-begin>
   <url-path>/day/</url-path>
   <expired>false</expired>
   <completed>false</completed>
   </row>`
• The acl-id is the sco-id of the course.
• The permission-id is view to enroll users.
• The principal-id is unique in each trio.

If any trios have incorrect information, permissions-update returns an ok status, executes the correct trios, and does not execute the invalid ones.

4 Call permissions-info to check that the users have been enrolled:
https://example.com/api/xml?action=permissions-info
 &acl-id=2007064258 &filter-permission-id=view

Without a principal-id, this call returns a list of all principals enrolled in the course.

**Unenroll a large number of users (1000+) from a course**

1 Get the sco-id of the course (see “Find the sco-id of a course or curriculum”).

2 Get the principal-id of each user you want to remove. You can:
   • Call principal-list with filters to list the users you want to unenroll:
     https://example.com/api/xml?action=principal-list&filter-type=user &filter-account-id=624520
   • Read the values from a file.

3 Write a method that calls permissions-update with multiple trios of acl-id, principal-id, and permission-id:
https://example.com/api/xml?action=permissions-update
 &acl-id=2007064258&principal-id=2007105030&permission-id=denied&acl-id=2007064258&principal-id=2006258745&permission-id=denied ...  

The permission-id is denied to unenroll users from the course.

4 Call permissions-info to check that the users have been removed:
https://example.com/api/xml?action=permissions-info
 &acl-id=2007064258 &filter-permission-id=denied

**Enroll a large group (1000+) in a course**

1 Create a group.

   With the XML API Call principal-update and parse the response for the principal-id:
   https://example.com/api/xml?action=principal-update&type=group &has-children=1&name=developersc5

   With Acrobat Connect Pro Central Create the group at Administration > Users and Groups > New Group. Take the principal-id of the new group from the browser URL.

2 Add the users you want to enroll to the group. You can use an API call or Acrobat Connect Pro Central, but you can add only 200 users at a time.

   With the XML API Call group-membership-update, using multiple trios of group-id, principal-id, and is-member=true:
https://example.com/api/xml?action=group-membership-update &group-id=4930296&principal-id=2006258745&is-member=true &group-id=4930296&principal-id=2007343711&is-member=true

If any trios have incorrect information, group-membership-update returns an ok status, but the user in the incorrect trio is not added to the group.
With Acrobat Connect Pro Central Navigate to Administration > Users and Groups > Import. You can import users from a CSV (comma-delimited) file with at least a login ID for each user.

3 Get the sco-id of the course (see Find the sco-id of a course or curriculum) using the sco-id of the specialized training folder that contains the course.

4 Call permissions-update to enroll the group in the course:

   https://example.com/api/xml?action=permissions-update
   &acl-id=2007064258&principal-id=2007105030&permission-id=view

Unenroll a large group (1000+) from a course

1 Call permissions-info on the course, filtering for a permission-id of view:

   https://example.com/api/xml?action=permissions-info
   &acl-id=2006298444&filter-permission-id=view&filter-type=group

2 Parse the response for the principal-id of the group:

   <principal principal-id="2006258745" is-primary="false" type="group"
   has-children="true" permission-id="view">
     <name>developers</name>
     <login>developers@acme.com</login>
   </principal>

3 Call permissions-update with a permission-id of denied to remove the group's access to the course:

   https://example.com/api/xml?action=permissions-update
   &acl-id=2007064258&principal-id=2007105030&permission-id=denied

View curriculum information

As training managers create curriculums and users take courses, you need to retrieve information about them to display in your application. Often you can make just a single call to get the information you need, once you have the sco-id of the curriculum or course and the user's principal-id.

You may, for example, want to display all users enrolled in a curriculum or all courses a curriculum has. Another common task is to display the courses in a curriculum the user has completed so far, and then display the remaining courses.

Display all users enrolled in a course or curriculum

1 Call permissions-info, filtering for a permission-id of view:

   https://example.com/api/xml?action=permissions-info
   &acl-id=2006298444&filter-permission-id=view

   • The acl-id is the sco-id of the course or curriculum.
   • The permission-id of view means the user is enrolled.

2 Parse the response for principal-id, name, and any other values you need:

   <principal principal-id="2006258745" is-primary="false" type="user"
   has-children="false" permission-id="view">
     <name>Joy Smith</name>
     <login>joy@acme.com</login>
   </principal>

Display a list of all training modules in a curriculum

A curriculum is a type of folder, and you can list its contents with sco-contents or sco-expanded-contents.
1 Get the sco-id of the curriculum (see “Find the sco-id of a course or curriculum”).

2 Call sco-expanded-contents, passing it the sco-id:
   https://example.com/api/xml?action=sco-expanded-contents&
   sco-id=2006745669

3 Parse the response for the sco-id, folder-id, and depth:
   <sco depth="1" sco-id="2006745674" folder-id="2006745669" type="link"
   display-seq="0" source-sco-id="2006745673"
   icon="course" lang="en" source-sco-type="0">
     <name>All About Web Communities</name>
     <url-path>/l80422078/</url-path>
     <description>test</description>
     <date-created>2006-06-12T14:48:25.980-07:00</date-created>
     <date-modified>2006-06-12T14:48:25.980-07:00</date-modified>
   </sco>

   The response returns a flat list of sco elements, including the curriculum and each SCO it contains. You can build a hierarchy using the sco-id, folder-id, and depth values. The SCO with type=curriculum is the curriculum that contains the courses.

View a user’s completed and remaining work in a curriculum

1 Get the sco-id of the curriculum (see “Find the sco-id of a course or curriculum”).

2 Get the principal-id of the user (“Find a principal-id”).

3 Call report-curriculum-taker, passing the principal-id as a user-id:
   https://example.com/api/xml?action=report-curriculum-taker
   &user-id=2006258745&sco-id=2006745669

4 Parse the response for the status attribute of each sco element and any other values you want to display in your application:
   <sco transcript-id="2006745722" path-type="prereq-none" asset-id=""
   sco-id="2006745674" depth="1" folder-id="2006745669" type="15"
   icon="course" lang="en" max-retries="" source-sco-id="2006745673"
   source-sco-type="0" status="user-passed" score="0" certificate="" max-score="0" attempts="0">
     <access>access-open</access>
     <credit-granted>true</credit-granted>
     <name>All About Web Communities</name>
     <url-path>/l80422078/</url-path>
     <description>test</description>
     <date-created>2006-06-12T15:06:02.947-07:00</date-created>
     <date-modified>2006-06-12T14:48:25.980-07:00</date-modified>
     <date-taken>2006-06-12T15:06:02.947-07:00</date-taken>
     <override>false</override>
   </sco>

   • A status of user-passed or completed indicates a module the user has completed.
   • A status of not-attempted or incomplete shows the user has not completed the module.
   • The curriculum itself can only have a status of completed or incomplete.
Report scores

Many courses offer learners a certain number of retries. If you use server-side review mode, a training manager can specify the maximum attempts the learner has to complete or pass the course successfully (see Acrobat Connect Pro User Guide for details of how course retry works in both server-side and client-side review mode).

This means that a learner can attempt a course multiple times and have multiple scores. In your application, you may want to display only the learner's highest score.

Report a user's highest score on a course or quiz

1. Get the user's principal-id (see “Get the principal-id of the current user”).
2. Get the sco-id of the course or quiz (see “Find the sco-id of a course or curriculum”).
3. Call report-user-training-transcripts, filtering on the sco-id and sorting on the score:
   https://example.com/api/xml?action=report-user-training-transcripts
   &principal-id=2006258745&filter-sco-id=2006334909&sort-score=desc
4. Parse the response for the highest score, which should be in the first row element in the list:
   <row transcript-id="2006335954" sco-id="2006334909"
       principal-id="2006258745" status="user-passed" score="20"
       max-score="20" certificate="2006335954" type="content"
       icon="producer">
   <name>Java Data Type Quiz</name>
   <url-path>/quiz/</url-path>
   <login>bob@acme.com</login>
   <date-taken>2006-05-12T11:55:24.940-07:00</date-taken>
   <principal-name>Bob Smith</principal-name>
   </row>
Chapter 6: Action reference

This section provides a reference for each action in the Adobe® Acrobat® Connect™ Pro Web Services XML API. You can call these actions on Adobe Acrobat Connect Pro Server 7.

All action, parameter, element, and attribute names are case sensitive. In other words, name is not the same as Name, and sco-id is not equivalent to sco-ID. You must enter them exactly as shown in this reference, unless a specific entry indicates an item is not case sensitive.

New and removed actions

This release adds some new actions.

**New actions for compliance** allow you to enable or disable functionality in meeting rooms:
- meeting-feature-update
- meeting-disclaimer-info
- meeting-disclaimer-update

**New actions for permissions** vary the scope of administrator abilities:
- limited-administrator-permissions info
- limited-administrator-permissions-update

**New search action** allows you to search by field:
- sco-search-by-field

**New actions for managing system capacity** allow administrators to manage users and licenses:
- account-expiry-info
- expiry-settings-update
- expiry-settings-info
- quota-threshold-update
- quota-threshold-info
- quota-threshold-exceeded

**Enhanced enrollment action** allows users to self-enroll in training:
- acl-field-update
account-expiry-info

Availability
Acrobat Connect Pro Server 7

Description
Returns the expiration date of an account.

Request URL
http://server_name/api/xml
?action=account-expiry-info
&account-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account for which you want expiration information. If you don’t provide an account ID, the expiration date for the current user is returned.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
  <Account account-id=integer>
    <name>String</name>
    <date-expired>Datetime</date-expired>
  </Account>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>Integer</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>Account</td>
<td>Container</td>
<td>Information about all fields describing the account.</td>
<td></td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>The ID of the account.</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the account.</td>
<td></td>
</tr>
<tr>
<td>date-expired</td>
<td>Datetime</td>
<td>The date the account expired.</td>
<td></td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=account-expiry-info&account-id=7
Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
  <Account account-id="7">
    <name>Enterprise Account</name>
    <date-expired>2009-09-11T18:15:00.000+05:30</date-expired>
  </Account>
</results>
```

See also

expiry-settings-info, expiry-settings-update

**acl-field-info**

**Availability**

Breeze 5, Connect Enterprise Web Services 6

**Description**

Returns information about a principal, account, or SCO, as defined in an access control list (ACL). The returned information includes fields and their values. Each field has an ID—a name that describes the field.

To call acl-field-info, you must have view permission for the principal, account, or object. You must also specify a value for acl-id, which is the object the principal has access to. The acl-id can be a sco-id, an account-id, or a principal-id. You can call principal-list to determine the account-id or principal-id, or sco-shortcuts or sco-contents to get a sco-id.

**Request URL**

```
http://server_name/api/xml
?action=acl-field-info
&acl-id=integer
&session=BreezeSessionCookieValue
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the SCO, account, or principal for which you want field information. Can be a valid sco-id, account-id, or principal-id.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

Results cannot be filtered or sorted.

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <acl-fields>
    <field acl-id=integer field-id=string>
      <value>string</value>
    </field>
  </acl-fields>
</results>
```
...</acl-fields>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>acl-fields</td>
<td></td>
<td>Container</td>
<td>Information about all fields describing the principal, account, or object.</td>
</tr>
<tr>
<td>field</td>
<td></td>
<td>Container</td>
<td>One field describing the principal, account, or object.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The acl-id specified in the request, which is a sco-id, principal-id, or account-id.</td>
</tr>
<tr>
<td>field-id</td>
<td></td>
<td>String</td>
<td>The name of the field.</td>
</tr>
<tr>
<td>value</td>
<td></td>
<td>String</td>
<td>The value of the field.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=acl-field-info&acl-id=2006258745

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <acl-fields>
    <field acl-id="2006258745" field-id="email">
      <value>joy@acme.com</value>
    </field>
    <field acl-id="2006258745" field-id="first-name">
      <value>Joy</value>
    </field>
    <field acl-id="2006258745" field-id="last-name">
      <value>Smith</value>
    </field>
  </acl-fields>
</results>

See also
acl-field-list, acl-field-update

acl-field-list

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Returns a list of values for all instances of a field name on your Acrobat Connect Pro Server account.
For example, to list the first names of all users in the account, call `acl-field-list` with `field-id=first-name`.

You can call `acl-field-info` first to get a list of field names.

**Request URL**

```
http://server_name/api/xml
?action=acl-field-list
&field-id=string
&session=BreezeSessionCookieValue
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field-id</td>
<td>String</td>
<td>Y</td>
<td>The name of a field in the access control list for which you want values and IDs. Only one field name is allowed.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

### Filters

Results cannot be filtered or sorted.

### Response structure

```
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <acl-field-list>
    <acl acl-id=integer>
      <value>string</value>
    </acl>
    ...
  </acl-field-list>
</results>
```

### Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>acl-field-list</td>
<td></td>
<td>Container</td>
<td>Information about all of the values in the account for the specified field.</td>
</tr>
<tr>
<td>acl</td>
<td></td>
<td>Container</td>
<td>Information about one value for the specified field.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the principal, SCO, or account the field belongs to.</td>
</tr>
<tr>
<td>value</td>
<td></td>
<td>String</td>
<td>The value of the field.</td>
</tr>
</tbody>
</table>

### Sample request

https://example.com/api/xml?action=acl-field-list&field-id=first-name
Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
  <acl-field-list>
    <acl acl-id="381223">
      <value>John</value>
    </acl>
    <acl acl-id="381302">
      <value>Daryl</value>
    </acl>
    <acl acl-id="381405">
      <value>Mary</value>
    </acl>
  </acl-field-list>
</results>
```

See also

acl-field-info, acl-field-update

**acl-field-update**

**Availability**
Breeze 5; Connect Enterprise Web Services 6

**Description**

Updates the value of a field that belongs to a SCO or an account.

*Note: To update a standard field for a principal (a user or a group), use the principal-update action. To update a custom field for a principal, use the acl-field-update action.*

The SCO or account belongs to at least one access control list (ACL), which lists the principals that have permission to access the SCO, or account.

Call acl-field-info to determine the fields the SCO or account has. In the response, you can see the field-id you need for the request to acl-field-update:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
  <acl-fields>
    <field acl-id="2006258745" field-id="email">
      <value>joy@acme.com</value>
    </field>
    ...
  </acl-fields>
</results>
```

You can specify multiple trios of acl-id, field-id, and value. If you do, use an HTTP POST method, rather than a GET, to make the request. The GET method has limitations that might cause the request to be truncated. With a POST, you can add about 50 trios to the request.

To call acl-field-update, you need modify permission on the SCO or account.

**Request URL**

http://server_name/api/xml
  ?action=acl-field-update
  &acl-id=integer
&field-id=string
&value=string
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the SCO or account. Can be a valid sco-id or account-id.</td>
</tr>
<tr>
<td>field-id</td>
<td>String</td>
<td>Y</td>
<td>The name of the field for which you want to update value. The field can be a server-defined field or a custom field. A custom field has a field-id starting with x-, such as x-12056.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Y</td>
<td>The value to set.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=acl-field-update&acl-id=2007035246
  &field-id=name&value=Java 101

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>

See also
acl-field-list

acl-preference-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6
Description
Updates a user profile with new language and time zone settings.

Request URL
http://server_name/api/xml
  ?action=acl-preference-update
  &acl-id=integer
  &lang=allowedValue
  &time-zone-id=allowedValue
  &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the user whose preferences will be updated. Can be a valid principal-id.</td>
</tr>
<tr>
<td>lang</td>
<td>Allowed value</td>
<td>N</td>
<td>An abbreviation for the new language (see <code>lang</code> for valid values).</td>
</tr>
<tr>
<td>time-zone-id</td>
<td>Allowed value</td>
<td>N</td>
<td>An integer setting for the new time zone (see <code>time-zone-id</code> for values).</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see <code>status</code>).</td>
</tr>
</tbody>
</table>

Sample request
  &lang=fr&time-zone-id=0

Sample response
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```
common-info

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Returns basic information about the current user and the Acrobat Connect Pro server or Acrobat Connect Pro hosted account, including the value of the BREEZESESSION cookie.

If you call common-info without logging in, the response does not contain user and account elements, because the server cannot identify a user. However, even without logging in, common-info returns a BREEZESESSION cookie value.

The response also contains host, local-host, and admin-host elements. If Acrobat Connect Pro is hosted on a cluster, host is the cluster name; local-host is the name of the server in the cluster that executes the call to common-info; and admin-host is the name of the secure host on a cluster that supports SSL. Your application can use the value of admin-host to convert HTTP URLs to more secure HTTPS URLs.

Request URL
http://server_name/api/xml?action=common-info&domain=string&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>String</td>
<td>N</td>
<td>A domain name identifying a Acrobat Connect Pro hosted account. Use to get information about your hosted account.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <common locale=allowedValue time-zone-id=integer>
    <cookie>string</cookie>
    <date>datetime</date>
    <host>url</host>
    <local-host>hostname</local-host>
    <admin-host>hostname</admin-host>
    <url>/api/xml?action=common-info</url>
    <version>string</version>
    <account account-id=integer />
    <user user-id=integer type="user">
      <name>string</name>
      <login>string</login>
    </user>
    <user-agent>string</user-agent>
  </common>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with</td>
<td>Empty, with</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>attributes</td>
<td>attributes</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>common</td>
<td>Container</td>
<td>Container</td>
<td>Common information about your connection to the server.</td>
</tr>
<tr>
<td>locale</td>
<td>Allowed value</td>
<td>Allowed value</td>
<td>A setting that defines how Acrobat Connect Pro Central or your application displays information to a user (see lang for values).</td>
</tr>
<tr>
<td>time-zone-id</td>
<td>Allowed value</td>
<td>Allowed value</td>
<td>A code that defines the user's time zone (see time-zone-id for values).</td>
</tr>
<tr>
<td>cookie</td>
<td>String</td>
<td>The value of the BREEZESESSION cookie (a string the server returns identifying this user for this login session).</td>
<td></td>
</tr>
<tr>
<td>date</td>
<td>Datetime</td>
<td>Date</td>
<td>The date and time the call to common-info was made, in ISO 8601 format.</td>
</tr>
<tr>
<td>host</td>
<td>String</td>
<td>If Acrobat Connect Pro runs on a server, the URL of the fully qualified host name of the server. If a cluster, the name that identifies the cluster.</td>
<td></td>
</tr>
<tr>
<td>local-host</td>
<td>String</td>
<td>The name of the computer that executed the action (on a single server, the same as host; on a cluster, the name of the server that executed the action).</td>
<td></td>
</tr>
<tr>
<td>admin-host</td>
<td>String</td>
<td>The name of the secure host on a cluster that supports SSL.</td>
<td></td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>The part of the URL making this call that identifies the action name.</td>
<td></td>
</tr>
<tr>
<td>version</td>
<td>String</td>
<td>The server version name and number.</td>
<td></td>
</tr>
<tr>
<td>account</td>
<td>Empty, with</td>
<td>Empty, with</td>
<td>Information about the account the user belongs to. Returned if you are logged in to Acrobat Connect Pro or are making the call on a Acrobat Connect Pro hosted account.</td>
</tr>
<tr>
<td></td>
<td>attribute</td>
<td>attributes</td>
<td></td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>Integer</td>
<td>The ID of the account the user belongs to.</td>
</tr>
<tr>
<td>user</td>
<td>Container</td>
<td>Container</td>
<td>Information about the user who established a session with the server. Returned only if the user making the call is logged in.</td>
</tr>
<tr>
<td>user-id</td>
<td>Integer</td>
<td>Integer</td>
<td>The ID of the user who established a session with the server.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>Allowed value</td>
<td>The type of principal who has a session (usually user; see allowed values for principals at type).</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The full name of the user who established a session with the server.</td>
<td></td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td>The login name of the user who is logged in to the server, often the user's e-mail address.</td>
<td></td>
</tr>
<tr>
<td>user-agent</td>
<td>String</td>
<td>The identifier of the web browser or client that established a session with the server.</td>
<td></td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=common-info

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <common locale="en" time-zone-id="4">
    <cookie>breezsi4undh5srfw2fg6</cookie>
    <date>2006-09-08T11:17:04.470-07:00</date>
    <host>https://example.com</host>
    <local-host>localserver17</local-host>
    <admin-host>securehost.com</admin-host>
    <url>/api/xml?action=common-info</url>
    <version>connect_6000</version>
    <account account-id="624520" />
    <user user-id="2006258745" type="user">
      <name>Joy Smith</name>
      <login>joy@acme.com</login>
    </user>
    <user-agent>Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)</user-agent>
  </common>
</results>

**custom-fields**

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Lists all custom fields defined in an account and details about the fields.
Custom fields provide information about objects (SCOs) or principals that is not already defined in Acrobat Connect Pro Central. You can create custom fields, or update their value, using `custom-field-update`.

**Request URL**
http://server_name/api/xml
?action=custom-fields
&filter-definition=value
&session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the <code>BREEZESESSION</code> cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**
You can filter the response on any element or attribute it contains.

**Response structure**
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <custom-fields>
    <field permission-id=allowedValue object-type=allowedValue
            field-id=string account-id=integer display-seq=integer
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td></td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td></td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>custom-fields</td>
<td></td>
<td>Container</td>
<td>The list of custom fields that match the query.</td>
</tr>
<tr>
<td>field</td>
<td></td>
<td></td>
<td>Details about one custom field.</td>
</tr>
<tr>
<td>permission-id</td>
<td></td>
<td>Allowed value</td>
<td>The permission the current user has to access the custom field (see permission-id for values).</td>
</tr>
<tr>
<td>object-type</td>
<td></td>
<td>Allowed value</td>
<td>The type of object the custom field describes (see permission-id for values).</td>
</tr>
<tr>
<td>field-id</td>
<td></td>
<td>String</td>
<td>The name of the field, as identified on the server.</td>
</tr>
<tr>
<td>account-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the account in which the custom field is defined.</td>
</tr>
<tr>
<td>display-seq</td>
<td></td>
<td>Integer</td>
<td>The sequence in which Acrobat Connect Pro Central or your application displays the custom field, relative to other custom fields.</td>
</tr>
<tr>
<td>field-type</td>
<td></td>
<td>Allowed value</td>
<td>The type of data the custom field accepts. Allowed values are text, textarea, and password.</td>
</tr>
<tr>
<td>is-primary</td>
<td></td>
<td>Boolean</td>
<td>Whether the custom field can be deleted [true if no, and false if yes].</td>
</tr>
<tr>
<td>is-required</td>
<td></td>
<td>Boolean</td>
<td>Whether this custom field is required. true if a value must be specified for this field in each object that uses it. Otherwise, false.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the custom field as Acrobat Connect Pro Central or your application displays it.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=custom-fields&filter-like-name=name

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <custom-fields>
    <field permission-id="manage" object-type="object-type-principal"
      field-id="first-name" account-id="624520" display-seq="1"
      field-type="text" is-primary="true" is-required="true">
      <name>First Name</name>
    </field>
    <field permission-id="manage" object-type="object-type-principal"
      field-id="last-name" account-id="624520" display-seq="2"
      field-type="text" is-primary="true" is-required="true">
      <name>Last Name</name>
    </field>
  </custom-fields>
</results>
custom-fields-delete

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Deletes a custom field.
The value of is-primary for a custom field must be false before the field can be deleted. If is-primary is true and you want to change its value, call custom-field-update.

Request URL
http://server_name/api/xml
?action=custom-fields-delete
&field-id=string
&object-type=allowedValue
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field-id</td>
<td>String</td>
<td>Y</td>
<td>The ID of the field to be deleted. Call custom-fields-delete to obtain the ID, which is returned in the field-id attribute of the field element.</td>
</tr>
<tr>
<td>object-type</td>
<td>String</td>
<td>Y</td>
<td>The type of SCO for which the field is defined (for values, see type).</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=custom-fields-delete&field-id=2006338719&object-type=object-type-principal

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code="ok" />
</results>

See also
custom-field-update

custom-field-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Creates a new custom field or updates the value of an existing one.

You can define up to eight custom fields on a principal or SCO. To create a custom field, call custom-field-update with at least the following fields: object-type, permission-id, name, field-type, is-required, and is-primary. If custom-field-update is successful, it returns a field-id.

To update a custom field, specify the field-id, an object-type, and a name for each field that has a value you want to change.

Be careful when defining custom fields, as retrieving those fields in a report (for example, by calling report-bulk-users) can affect the performance of the server and the database.

Request URL
http://server_name/api/xml
?action=custom-field-update
&account-id=integer
&object-type=object-type-allowedValue
&permission-id=allowedValue
&name=string
&comments=string
&field-type=allowedValue
&is-required=boolean
&is-primary=boolean
&display-seq=integer
&field-id=integer
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The account ID in which the field is created.</td>
</tr>
<tr>
<td>object-type</td>
<td>String</td>
<td>Y</td>
<td>The type of SCO this field applies to. Required to create and update fields.</td>
</tr>
<tr>
<td>permission-id</td>
<td>String</td>
<td>Y</td>
<td>The permission a principal needs on the object to set or view the field's</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>value. The only allowed value is manage. Required to create a field.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Y</td>
<td>The label for the field in the user interface. Required to create a field.</td>
</tr>
<tr>
<td>comments</td>
<td>String</td>
<td>N</td>
<td>Any comments you define for the custom field, displayed as hint text in your</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>user interface. Can be up to 60 characters long.</td>
</tr>
<tr>
<td>field-type</td>
<td>String</td>
<td>Y</td>
<td>The type of field. Allowed values are text, textarea, and password. Required</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>to create a field.</td>
</tr>
<tr>
<td>is-required</td>
<td>Boolean</td>
<td>Y</td>
<td>Whether this custom field is required. Use true if a value must be specified</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>for this field in each object that uses it. Otherwise, use false. Required</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>to create a field.</td>
</tr>
<tr>
<td>is-primary</td>
<td>Boolean</td>
<td>Y</td>
<td>Whether this custom field can be deleted through the user interface (true if</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>it cannot be deleted, and false if it can).</td>
</tr>
<tr>
<td>display-seq</td>
<td>Integer</td>
<td>N</td>
<td>The sequence in which Acrobat Connect Pro Central or your application displays</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the custom field, relative to other custom fields.</td>
</tr>
<tr>
<td>field-id</td>
<td>Integer</td>
<td>Y</td>
<td>The name of a field that has a value you want to update. Required to update</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a field.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <field field-id=integer display-seq=integer object-type=allowedValue account-id=integer is-primary=boolean permission-id=allowedValue is-required=boolean field-type=string>
    <comments>string</comments>
    <name>string</name>
</field>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>field</td>
<td></td>
<td>Empty, with attributes</td>
<td>Information about the custom field.</td>
</tr>
<tr>
<td></td>
<td>field-id</td>
<td>Integer</td>
<td>A numeric identifier for the field.</td>
</tr>
<tr>
<td></td>
<td>display-seq</td>
<td>Integer</td>
<td>The sequence in which Acrobat Connect Pro Central or your application displays the field.</td>
</tr>
<tr>
<td></td>
<td>object-type</td>
<td>Allowed value</td>
<td>The type of object the field describes (see type for allowed values).</td>
</tr>
<tr>
<td></td>
<td>account-id</td>
<td>Integer</td>
<td>For customers on Acrobat Connect Pro hosted accounts, the ID of the account in which the field is defined.</td>
</tr>
<tr>
<td></td>
<td>is-primary</td>
<td>Boolean</td>
<td>Whether this custom field can be deleted (true if no, false if yes).</td>
</tr>
<tr>
<td></td>
<td>permission-id</td>
<td>Allowed value</td>
<td>The permission needed to access the custom field (see permission-id for allowed values).</td>
</tr>
<tr>
<td></td>
<td>is-required</td>
<td>Boolean</td>
<td>Whether a value for this custom field is required (true if yes and false if no).</td>
</tr>
<tr>
<td></td>
<td>field-type</td>
<td>Allowed value</td>
<td>The type of data the field accepts. Allowed values are text, textarea, and password.</td>
</tr>
<tr>
<td></td>
<td>comments</td>
<td>String</td>
<td>The comment entered in comments in the request.</td>
</tr>
<tr>
<td></td>
<td>name</td>
<td>String</td>
<td>The name of the field entered in name in the request.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=custom-field-update
&object-type=object-type-principal&permission-id=manage
&account-id=624520&name=jobtitle&comments=test&field-type=text
&is-required=true&is-primary=false&display-seq=1

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <field field-id="2006472106" object-type="object-type-principal"
       display-seq="1" account-id="624520" is-primary="false"
       permission-id="manage" is-required="true" field-type="text">
    <comments>test</comments>
    <name>jobtitle</name>
  </field>
</results>

See also

report-bulk-users
expiry-settings-info

Availability
Acrobat Connect Pro Server 7

Description
Returns information about the current settings for account-expiration notifications (the warnings given to users before an account expires). A user is notified \( x \) number of days before their account expires. This action simply returns the value of \( x \).

Request URL
https://example.com/api/xml
 ?action=expiry-settings-info
 &account-id=\text{Integer}
 &session=\text{String}

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account. If you don’t provide an account ID, the information for the current account is returned.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the \text{BREEZESESSION} cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=\text{allowedValue} />
  <expiry-num-of-days>
    <value>30</value>
  </expiry-num-of-days>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>expiry-num-of-days</td>
<td>Container</td>
<td>Information about the current settings for account-expiration notifications.</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td></td>
<td>Integer</td>
<td>The user is notified this many days before their account expiration. The default value is 30. For example, if a user’s account expires on December 31, the user is notified on December 1.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=expiry-settings-info&account-id=7

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
```
<results>
  <status code="ok"/>
  <expiry-num-of-days>
    <value>30</value>
  </expiry-num-of-days>
</results>

See also
expiry-settings-update

### expiry-settings-update

**Availability**

Acrobat Connect Pro Server 7

**Description**

Updates information about the settings for account-expiration notification (the notification given to users before an account expires). A user is notified \( x \) number of days before their account expires. This action simply updates the value of \( x \).

**Request URL**

https://example.com/api/xml
  ?action=expiry-settings-update
  &account-id=Integer
  &session=String

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account. If you don't provide an account ID, the information for the current account is updated.</td>
</tr>
<tr>
<td>expiry-num-of-days</td>
<td>Integer</td>
<td>Y</td>
<td>A user is notified this many days before their account expires. The default value is 30; possible values are 30, 60, and 90. For example, if the value of this parameter is 30, a user is notified 30 days before their account is due to expire.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=expiry-settings-update&account-id=7&expiry-num-of-days=30

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
</results>
```

See also
account-expiry-info, expiry-settings-info

group-membership-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Adds one or more principals to a group, or removes one or more principals from a group.
To update multiple principals and groups, specify multiple trios of group-id, principal-id, and is-member parameters.
You can obtain a group-id by calling principal-list and filtering the response with filter-type=group or another filter value such as filter-type=admins. The built-in groups have distinctive types other than group (see type for a list of values).

Request URL

http://server_name/api/xml
  ?action=group-membership-update
  &group-id=integer
  &principal-id=integer
  &is-member=boolean
  &session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the group in which you want to add or change members.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the principal whose membership status you want to update. Returns by principal-info.</td>
</tr>
<tr>
<td>is-member</td>
<td>Boolean</td>
<td>Y</td>
<td>Whether the principal is added to (true) or deleted from (false) the group.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="allowedValue" />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>Top-level element for the response.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=group-membership-update&group-id=632398&principal-id=2006258745&is-member=true

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

learning-path-info

Availability

Breeze 5; Connect Enterprise Web Services 6

Description

Returns a list of learning paths for a learning object that belongs to a curriculum.

A learning object is any SCO that has been added to a curriculum. A learning path is determined by rules that establish whether a learner can proceed to the next learning object.
You can create a learning path by establishing prerequisite requirements, completion requirements, or preassessment requirements. For example, a learning path might be the rule that the class Welcome to AcmeCo must be completed before Managing Projects at AcmeCo.

A call to learning-path-info lists modules within a curriculum and their paths to each other. To see the complete contents of a curriculum, including content, meetings, and so on, call sco-expanded-contents.

**Request URL**

http://server_name/api/xml

?action=learning-path-info
&curriculum-id=integer
&sco-id=integer
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curriculum-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the curriculum the learning object belongs to.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the curriculum module (course, presentation, or similar) for which you want a learning path.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

You can filter or sort the response on any element or attribute it contains.

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <learning-paths>
    <learning-path curriculum-id=integer current-sco-id=integer target-sco-id=integer path-type=allowedValue>
      <name>string</name>
    </learning-path>
  </learning-paths>
</results>
```

**Response values**

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>learning-paths</td>
<td></td>
<td>Container</td>
<td>Information about learning paths in a curriculum.</td>
</tr>
<tr>
<td></td>
<td>curriculum-id</td>
<td>Integer</td>
<td>The numeric ID of the curriculum.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=learning-path-info&sco-id=2006334909
&curriculum-id=2006298444

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code="ok" />
   <learning-paths>
      <learning-path curriculum-id="2006298444" current-sco-id="2006298444"
                     target-sco-id="2006298445" path-type="completion-required">
         <name>Security at AcmeCo</name>
      </learning-path>
   </learning-paths>
</results>

See also
learning-path-update

learning-path-update

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Updates the learning path for a single learning object in a curriculum. A learning object is any SCO that is added to a curriculum.

Request URL
http://server_name/api/xml?
   action=learning-path-update
   &curriculum-id=integer
   &current-sco-id=integer
   &target-sco-id=integer
   &path-type=allowedValue
   &session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curriculum-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the curriculum to which this learning object belongs.</td>
</tr>
<tr>
<td>current-sco-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the learning object that has the access you want to update.</td>
</tr>
<tr>
<td>target-sco-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the learning object that restricts access to the current learning object (for example, a prerequisite course).</td>
</tr>
<tr>
<td>path-type</td>
<td>Allowed value</td>
<td>Y</td>
<td>The type of path between the target learning object and the current learning object (see path-type for allowed values).</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>Top-level element for the response.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

Sample response
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

See also
learning-path-info

limited-administrator-permissions info

Availability
Acrobat Connect Pro 7
Description
Returns a list of permissions that can be enabled or disabled for the Limited Administrators group and whether or not that permission is currently enabled. For more information on Limited Administrators, see limited-administrator-permissions-update.

Request URL
http://server_name/api/xml
?action=limited-administrator-permissions-info
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8"?>
<results>
  <status code="ok"/>
  <permissions>
    <permission>
      <enabled>Boolean</enabled>
      <name>string</name>
    </permission>
  </permissions>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>permissions</td>
<td></td>
<td>Container</td>
<td>A list of permissions.</td>
</tr>
<tr>
<td>permission</td>
<td></td>
<td>Container</td>
<td>A list of information about the permission.</td>
</tr>
<tr>
<td>enabled</td>
<td></td>
<td>Boolean</td>
<td>A value indicating whether the permission is enabled (true) or not (false).</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the permission.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=limited-administrator-permissions-info
&session=breeze6qdeheiso93efb5

Sample response
```xml
<?xml version="1.0" encoding="utf-8"?>
<results>
```
<status code="ok"/>
<permissions>
  <permission>
    <enabled>true</enabled>
    <name>edit-account-info</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>view-disk-usage-and-reports</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>reset-password</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>view-user-data</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>add-users-groups-webui</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>add-users-groups-csv</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>set-content-meeting-permissions</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>user-profile-fields</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>change-login-pw-policy</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>delete-users-groups</name>
  </permission>
  <permission>
    <enabled>true</enabled>
    <name>modify-current-users-groups</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>customization</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>compliance</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>chargebacks</name>
  </permission>
  <permission>
    <enabled>false</enabled>
    <name>view-system-usage-reports</name>
</permissions>
limited-administrator-permissions-update

Availability
Acrobat Connect Pro 7

Description
Updates the permissions that can be enabled for Limited Administrators.

With Limited Administrators, your organization can have finer control over administrators and what types of things they can access. Your organization can separate system administrators who control all aspects of the system from Limited Administrators, who can access and control a subset of the system.

Each Acrobat Connect Pro installation has one Limited Administrators group. Users in the Administrators group can edit the permissions of Limited Administrators.

Request URL
http://server_name/api/xml
?action=limited-administrator-permissions-update
&view-disk-usage-and-reports=boolean
&reset-password=boolean
&view-user-data=boolean
&add-users-groups-webui=boolean
&add-users-groups-csv=boolean
&user-profile-fields=boolean
&change-login-pw-policy=boolean
&delete-users-groups=boolean
&modify-current-users-groups=boolean
&customization=boolean
&edit-account-info=boolean
&set-content-meeting-permissions=boolean
&compliance=boolean
&chargebacks=boolean
&view-training-reports=boolean
&reset-to-default=value
Parameters
When you use this command, pass at least one parameter. The descriptions that follow indicate if the permission is set to true by default.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>view-disk-usage-and-reports</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to view disk usage and reports. The default value is true.</td>
</tr>
<tr>
<td>reset-password</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to reset the password of a user. Part of the view-user-data set. The default value is true.</td>
</tr>
<tr>
<td>view-user-data</td>
<td>Boolean</td>
<td>N</td>
<td>Superset; a value of true allows limited administrators to view user data. By setting this parameter to enable, you enable all parameters in this set. (See all parameters that are part of the view-user-data set.) The default value is true.</td>
</tr>
<tr>
<td>add-users-groups-webui</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to add users and groups by using the management console. Part of the view-user-data set. The default value is true.</td>
</tr>
<tr>
<td>add-users-groups-csv</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to add users or groups by importing a CSV file. Part of the view-user-data set.</td>
</tr>
<tr>
<td>user-profile-fields</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to modify user profile fields.</td>
</tr>
<tr>
<td>change-login-pw-policy</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to change the login and password policies.</td>
</tr>
<tr>
<td>delete-users-groups</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to delete users or groups. Part of the view-user-data set.</td>
</tr>
<tr>
<td>modify-current-users-groups</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to modify currents users and groups. Part of the view-user-data set. The default value is true.</td>
</tr>
<tr>
<td>customization</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to customize the colors of the account web pages, meetings, and the login page.</td>
</tr>
<tr>
<td>edit-account-info</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to edit account information.</td>
</tr>
<tr>
<td>set-content-meeting-permissions</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to set the permissions for content or meetings. The default value is true.</td>
</tr>
<tr>
<td>compliance</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to change compliance settings (settings for enabling pods, sharing, and recording, and for training settings).</td>
</tr>
<tr>
<td>chargebacks</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to access the cost-center settings for this account.</td>
</tr>
<tr>
<td>view-training-reports</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true allows limited administrators to view training reports.</td>
</tr>
<tr>
<td>reset-to-default</td>
<td>Boolean</td>
<td>N</td>
<td>A value of true resets all permissions to the default permissions set by Adobe.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

The example shows

https://admin.ibreeze.macromedia.com/api/xml?action=limited-administrator-permissions-update
&session=breezgh9nxddh768vpob
&view-user-data=true
&view-user-data=false
&reset-password=true
&reset-password=false
&modify-current-users-groups=true
&modify-current-users-groups=false
&add-users-groups-webui=true
&add-users-groups-webui=false
&add-users-groups-csv=false
&delete-users-groups=false
&user-profile-fields=true
&user-profile-fields=false
&change-login-pw-policy=true
&change-login-pw-policy=false
&chargebacks=false
&edit-account-info=true
&edit-account-info=false
&quota-threshold-notifications=false
&customization=false
&view-disk-usage-and-reports=false
&view-system-usage-reports=false
&compliance=false
&set-content-meeting-permissions=true
&set-content-meeting-permissions=false

Sample response

```xml
<?xml version="1.0" encoding="utf-8"?>
<results><status code="ok"/>
</results>
```

login

Availability

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description

Logs a user in to Acrobat Connect Pro Server.
In a client application, after logging in a user, you must read and store the cookie called BREEZESESSION, which can be found in the HTTP headers of the response from login. You must then include the value of that cookie in every subsequent request that you make for that user.

If you cannot retrieve cookie values from HTTP response headers, you can call common-info to get the cookie value before the user logs in. Then, pass the value to login using the session request parameter:

https://example.com/api/xml?action=login&login=loginId&password=password&session=value

You can also use the session parameter on any API call you make after login. For example, to call principal-list after logging in, you can enter:

https://example.com/api/xml?action=principal-list&session=value

The BREEZESESSION value is valid for only one login session. Your application must store a new cookie value each time the user logs in.

When you call the login action, you are sending a login ID and password across a network, unless you use external authentication. Use SSL or another appropriate security method to protect passwords in transit.

**Request URL**

http://server_name/api/xml

?action=login

&login=string

&password=string

&account-id=integer

&external-auth=use

&domain=string

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of your Acrobat Connect Pro hosted account. If your organization is running a licensed Acrobat Connect Pro Server, do not use account-id.</td>
</tr>
<tr>
<td>external-auth</td>
<td>Allowed value</td>
<td>N</td>
<td>A value indicating whether you send an external network login ID to represent the user to Acrobat Connect Pro. If so, use external-auth=use.</td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td>Y/N</td>
<td>The user's login name. Do not use if you use external or HTTP header authentication.</td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td>Y/N</td>
<td>The user's password. Do not use if you use external or HTTP header authentication.</td>
</tr>
<tr>
<td>domain</td>
<td>String</td>
<td>N</td>
<td>The domain name of your Acrobat Connect Pro hosted account. If your organization is running a licensed Acrobat Connect Pro Server, do not use domain.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

Results cannot be filtered or sorted.

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>All results the action returns.</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status [see status].</td>
<td></td>
</tr>
</tbody>
</table>

Sample request

http://example.com/api/xml?action=login&login=joy@acme.com&password=happy&session=breeztg8m25r93vebwur

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

See also

logout

logout

Availability

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description

Ends a user's login session, invalidating the cookie value associated with the user's session.

After calling logout, set the BREEZESESSION cookie value to null. Do not reuse the cookie value after your user logs out.

Request URL

http://server_name/api/xml?action=logout&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
http://example.com/api/xml?action=logout

Sample response
<?xml version='1.0' encoding='utf-8' ?>
<results>
  <status code='ok'/>
</results>

See also
login

meeting-disclaimer-info

Availability
Acrobat Connect Pro 7

Description
Provides information about the disclaimer text that is shown when a user enters a meeting. For more information about the disclaimer, see meeting-disclaimer-update.

Request URL
https://servername/api/xml?action=meeting-disclaimer-info
  &account-id=integer
  &session=string

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account for which the disclaimer text is retrieved. If not used, the account that you are currently logged in to is updated.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>A string: the value of the BREEZESESSION cookie.</td>
</tr>
</tbody>
</table>

Filters
Filters cannot be used with this action.

Response structure
<?xml version='1.0' encoding='utf-8' ?>
<results>
Using Acrobat Connect Pro Web Services

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>disclaimer</td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>disclaimer</td>
<td>string</td>
<td>String</td>
<td>The text of the disclaimer notice.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=meeting-disclaimer-info&account-id=7

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <disclaimer>
    This meeting may be recorded for compliance purpose. By clicking OK you agree to the terms of meeting.
  </disclaimer>
</results>
```

meeting-disclaimer-update

Availability

Acrobat Connect Pro 7

Description

Updates the disclaimer text that is shown when a user enters a meeting.

To comply with communications regulations or standards, you can set up a disclaimer notice to appear when a user enters a meeting. The disclaimer notice typically displays boilerplate information for your organization. It advises users of the status of the meeting and the terms of use for the meeting. For example, a disclaimer notice could advise users that the meeting is being recorded, and that users cannot join the meeting unless they accept the notice.

If the disclaimer is activated, the notice is shown in all meetings. Activate the disclaimer either through the management console or by using the meeting-feature-update action with the fid-meeting-disclaimer parameter set to enabled.

Request URL

https://servername/api/xml?action=meeting-disclaimer-update
&account-id=integer
&disclaimer=string
&session=string
Parameters

<table>
<thead>
<tr>
<th>Value</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account for which the disclaimer text is updated. If not used,</td>
</tr>
<tr>
<td>disclaimer</td>
<td>String</td>
<td>Y</td>
<td>The disclaimer text that is shown when a user starts a meeting. The disclaimer can,</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie.</td>
</tr>
</tbody>
</table>

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="code" />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=meeting-disclaimer-update&disclaimer=Please note that this meeting is being recorded.

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

meeting-feature-update

Availability

Acrobat Connect Pro 7 Web Services

Description

Enables or disables features in a meeting. This action is used to manage features such as recording of meetings and control of pods. For more information on usage, see Configure compliance settings. You can append multiple feature-id and enable pairs to the end of the request URL.

Request URL

```
http://server name/api/xml
 ?action=meeting-feature-update
  &account-id=integer
  &feature-id=value
```
&enable=value

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of your Acrobat Connect Pro hosted account. For enterprise installations, the ID is 7. For licensed installations, use common-info to get the ID.</td>
</tr>
<tr>
<td>feature-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the feature to enable or disable. For available IDs, see feature-id.</td>
</tr>
<tr>
<td>enable</td>
<td>Boolean</td>
<td>Y</td>
<td>Whether to enable the specified feature (true) or not (false).</td>
</tr>
</tbody>
</table>

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="code" />  
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

The following sample disables the Chat pod.

https://example.com/api/xml?action=meeting-feature-update&account-id=7&feature-id=fid-meeting-chat&enable=false

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />  
</results>
```

permissions-info

Availability

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description

Returns the list of principals (users or groups) who have permissions to act on a SCO, principal, or account.

To call permissions-info, you must specify an acl-id, which is the ID of a SCO, principal, or account that can be acted on. ACL stands for access control list, and means the list of entities who have permission.

With just an acl-id, permissions-info returns a list of all principals in the account, showing each principal’s permission on the principal or SCO specified in the acl-id:

https://example.com/api/xml?action=permissions-info&acl-id=2006258745
To check the permissions a specific principal has on a principal or SCO within an account, call permissions-info with an acl-id and a filter on principal-id:

http://example.com/api/xml?action=permissions-info&acl-id=7&filter-principal-id=10022

To check the permissions a principal has on an account, call permissions-info with both an acl-id (specifying an account-id) and a principal-id:

https://example.com/api/xml?action=permissions-info&acl-id=624520&principal-id=624523

**Request URL**

http://server_name/api/xml

?action=permissions-info

&acl-id=integer

&principal-id=integer

&filter-definition=value

&sort-definition=value

&session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a SCO, account, or principal that a principal has permission to act on. The acl-id is a sco-id, principal-id, or account-id in other calls.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of a principal who has a permission (even if denied) to act on an object.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

You can filter or sort the response on any element or attribute it contains.

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <permissions>
    <principal principal-id=integer is-primary=boolean type=allowedValue has-children=boolean permission-id=integer training-group-id=integer>
      <name>string</name>
      <login>string</login>
    </principal>
    ...
  <permission acl-id=integer permission-id=allowedValue principal-id=integer />
  </permissions>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>permissions</td>
<td></td>
<td>Container</td>
<td>A list of principals showing their permission to access the SCO, account, or principal.</td>
</tr>
<tr>
<td>principal</td>
<td></td>
<td>Container</td>
<td>Information about one principal showing the principal’s permission level on the SCO, account, or principal.</td>
</tr>
<tr>
<td>principal-id</td>
<td></td>
<td>Integer</td>
<td>The ID of a principal who has permission on a SCO, account, or principal.</td>
</tr>
<tr>
<td>is-primary</td>
<td></td>
<td>Boolean</td>
<td>A value indicating whether the principal is a primary group (same as a built-in group).</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The type of principal (see type for allowed values).</td>
</tr>
<tr>
<td>has-children</td>
<td></td>
<td>Boolean</td>
<td>A value indicating whether the principal has children. Groups have children and users don’t, so if true, the principal is a group.</td>
</tr>
<tr>
<td>permission-id</td>
<td></td>
<td>Allowed value</td>
<td>The permission the principal has on the SCO, account, or principal (see permission-id for values).</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the SCO on which the permission is defined.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the principal who has permission to access the SCO.</td>
</tr>
<tr>
<td>login</td>
<td></td>
<td>String</td>
<td>The login name of the principal who has permission to access the SCO.</td>
</tr>
<tr>
<td>permission</td>
<td></td>
<td>Empty, with attributes</td>
<td>Information about the permission one principal has on a SCO, account, or principal. If empty, no permission is defined.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the object on which the principal has permission.</td>
</tr>
<tr>
<td>permission-id</td>
<td></td>
<td>Allowed value</td>
<td>The permission the principal has to act on the object (see permission-id for values).</td>
</tr>
<tr>
<td>principal-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the principal who has permission to act on the object.</td>
</tr>
<tr>
<td>training-group-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the training group.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=permissions-info&acl-id=2006334033

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <permissions>
    <principal principal-id="2006258745" is-primary="false" type="user" has-children="false" permission-id="host" training-group-id="2007842424">
      <name>Joy Smith</name>
      <login>joy@acme.com</login>
    </principal>
    ...
  </permissions>
</results>
```
permissions-reset

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Resets all permissions any principals have on a SCO to the permissions of its parent SCO. If the parent has no permissions set, the child SCO will also have no permissions.

Request URL
http://server_name/api/xml
?action=permissions-reset
&acl-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a SCO that has permissions you want to reset.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code=code />
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=permissions-reset&acl-id=200634033

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code="ok" />
</results>

See also
permissions-info, permissions-update

permissions-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Updates the permissions a principal has to access a SCO, using a trio of principal-id, acl-id, and permission-id. To update permissions for multiple principals or objects, specify multiple trios. You can update more than 200 permissions in a single call to permissions-update.

Call permissions-update to give a user access to a Acrobat Connect Pro meeting, course, curriculum, or other SCO. For example, you can use permissions-update to:

- Invite a user to a meeting as participant, presenter, or host (with a permission-id of view, mini-host, or host, respectively)
- Remove a user's participant, presenter, or host access to a meeting (with a permission-id of remove)
- Enroll users in courses (with a permission-id of view)

If you use multiple trios and any of them have invalid information (for example, an incorrect acl-id or principal-id), permissions-update returns an ok status, the correct trios execute, and the invalid ones do not.

Request URL
http://server_name/api/xml
?action=permissions-update
&acl-id=integer
&principal-id=integer
&permission-id=allowedValue
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a SCO (a sco-id) for which you want to update permissions.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a principal, either a user or group.</td>
</tr>
<tr>
<td>permission-id</td>
<td>String</td>
<td>Y</td>
<td>The permission to assign (see permission-id for values).</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=permissions-update&acl-id=2006334033
&principal-id=2006258745&permission-id=host

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

See also

permissions-info, permissions-reset

principal-info

Availability

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description

Provides information about one principal, either a user or a group.

You must specify a principal-id. To find the principal-id, call principal-list, using a filter if necessary to limit the response.

Request URL

http://server_name/api/xml
?action=principal-info
&principal-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a user or group you want information about. You can get the ID by calling principal-list.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.
Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <contact>
    <email>string</email>
    <first-name>string</first-name>
    <last-name>string</last-name>
  </contact>
  <manager account-id=integer disabled=boolean has-children=boolean
            is-hidden=boolean is-primary=boolean principal-id=integer
            type=allowedValue>
    <ext-login>string</ext-login>
    <login>string</login>
    <name>string</name>
    <email>string</email>
    <first-name>string</first-name>
    <last-name>string</last-name>
    <x-customfield1>string</x-customfield1>
    <x-customfield2>string</x-customfield2>
    ...
  </manager>
  <preferences acl-id=integer lang=allowedValue
              time-zone-id=allowedValue />
  <principal account-id=integer disabled=boolean has-children=boolean
             is-hidden=boolean is-primary=boolean principal-id=integer
             type=allowedValue>
    <description>string</description>
    <ext-login>string</ext-login>
    <login>string</login>
    <name>string</name>
    <email>string</email>
    <first-name>string</first-name>
    <last-name>string</last-name>
    <x-customfield1>string</x-customfield1>
    <x-customfield2>string</x-customfield2>
    ...
  </principal>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>Top-level element for the response.</td>
</tr>
<tr>
<td>status</td>
<td>code</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>contact</td>
<td></td>
<td>Container</td>
<td>Information about the contact person for a principal. If the principal is a user, usually the same as information in principal.</td>
</tr>
<tr>
<td>email</td>
<td></td>
<td>String</td>
<td>The e-mail address of the contact person.</td>
</tr>
<tr>
<td>first-name</td>
<td></td>
<td>String</td>
<td>The first name of the contact person.</td>
</tr>
<tr>
<td>last-name</td>
<td></td>
<td>String</td>
<td>The last name of the contact person.</td>
</tr>
<tr>
<td>manager</td>
<td></td>
<td>Container</td>
<td>Information describing a user's manager, who is also a principal.</td>
</tr>
<tr>
<td>principal</td>
<td></td>
<td>Container</td>
<td>Information describing the principal.</td>
</tr>
</tbody>
</table>
### Sample request

https://example.com/api/xml?action=principal-info&principal-id=2006258745

### Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <contact>
    <email>bob@acme.com</email>
    <first-name>Bob</first-name>
    <last-name>Jones</last-name>
  </contact>
  <manager account-id="624520" disabled="false" has-children="false"
           is-hidden="false" is-primary="false" principal-id="2006282569"
           type="user">
    <ext-login>jazzdoe@example.com</ext-login>
  </manager>
</results>
```
<login>jazzdoe@example.com</login>
<name>jazz doe</name>
<email>jazzdoe@example.com</email>
<first-name>Jazz</first-name>
<last-name>Doe</last-name>
<x-2007017651>San Francisco</x-2007017651>
</manager>
<preferences acl-id="2006258745" lang="en" time-zone-id="4" />
<principal account-id="624520" disabled="" has-children="false"
is-hidden="false" is-primary="false" principal-id="2006258745"
type="user">
<ext-login>joy@acme.com</ext-login>
<login>joy@acme.com</login>
<name>Joy Smith</name>
<email>joy@acme.com</email>
<first-name>Joy</first-name>
<last-name>Smith</last-name>
<x-2007017651>San Francisco</x-2007017651>
</principal>
</results>

See also
principal-list, principal-list-by-field, principal-update

principal-list

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides a complete list of users and groups, including primary groups.
This call is useful for getting a principal-id when you don't have one. However, be aware that it returns a list of all
principals on your Acrobat Connect Pro Server or Acrobat Connect Pro hosted account, unless you use a filter to
limit the response.
You can also use principal-list to get a list of groups in an account by filtering on the type and is-member fields:
https://example.com/api/xml?action=principal-list&filter-type=group
&filter-is-member=true

However, filter-type=group returns groups you have created, not built-in groups predefined on the server. Built-in
groups have type values other than group, such as admins and authors (see type for a list of the values).
You can filter the response with a filter-type parameter set to the type of group you want, then parse the response
for a principal-id, then pass the principal-id as a group-id on another request to principal-list.

Request URL
http://server_name/api/xml
?action=principal-list
&group-id=integer
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of a group. Same as the principal-id of a principal that has a type value of group.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

*Note: Filtering on the login element is useful but slow, and reduced performance is unavoidable.*

You can also filter on a special field name, manager-id, to return a list of principals who report to a given manager, for example:

https://example.com/api/xml?action=principal-list
&filter-manager-id=2006282569

When you use filter-manager-id, each principal element in the response has a manager-id attribute:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <principal-list>
    <principal principal-id="2006258745" account-id="624520" type="user"
     has-children="false" is-primary="false" is-hidden="false"
     manager-id="2006282569">
      <name>Pat Lee</name>
      <login>plee@mycompany.com</login>
      <email>plee@mycompany.com</email>
    </principal>
  </principal-list>
</results>
```

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="allowedValue" />
  <principal-list>
    <principal principal-id="integer" account-id="integer" type="allowedValue"
     has-children="boolean" is-primary="boolean" is-hidden="boolean"
     manager-id="integer" training-group-id="integer">
      <name>string</name>
      <login>string</login>
      <email>string</email>
    </principal>
  </principal-list>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>All results the action returns.</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
<td></td>
</tr>
<tr>
<td>principal-list</td>
<td>Container</td>
<td>The entire list of principals.</td>
<td></td>
</tr>
<tr>
<td>principal</td>
<td>Container</td>
<td>Details about one principal.</td>
<td></td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the principal.</td>
<td></td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>The ID of the account the principal belongs to.</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>The type of principal (see type for values).</td>
<td></td>
</tr>
<tr>
<td>has-children</td>
<td>Boolean</td>
<td>Indicates whether the principal has children. Groups have children and users do not, so when has-children is true, the principal is a group.</td>
<td></td>
</tr>
<tr>
<td>is-primary</td>
<td>Boolean</td>
<td>Whether the principal is a built-in group (true) or not (false).</td>
<td></td>
</tr>
<tr>
<td>is-hidden</td>
<td>Boolean</td>
<td>Whether Acrobat Connect Pro Central or your application displays the principal (true for not displayed and false for displayed).</td>
<td></td>
</tr>
<tr>
<td>manager-id</td>
<td>Integer</td>
<td>The principal-id of the manager the principal reports to. Returned only if you use filter-manager-id in the request.</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The principal’s full name.</td>
<td></td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td>The principal’s login ID, often an e-mail address.</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>String</td>
<td>The principal’s e-mail address.</td>
<td></td>
</tr>
<tr>
<td>principal-custom-field-values</td>
<td>Container</td>
<td>The entire list of custom field values defined for the principal.</td>
<td></td>
</tr>
<tr>
<td>field</td>
<td>Container</td>
<td>Details about one custom field defined for the principal (see field for contents).</td>
<td></td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=principal-list

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <principal-list>
    <principal principal-id="624526" account-id="624520" type="user"
      has-children="false" is-primary="false" is-hidden="false" training-group-id="">
      <name>ned mack</name>
      <login>nmack@acme.com</login>
      <email>nmack@acme.com</email>
    </principal>
    <principal principal-id="624550" account-id="624520" type="user"
      has-children="false" is-primary="false" is-hidden="false" training-group-id="">
      <name>ned mack</name>
      <login>nmack@acme.com</login>
      <email>nmack@acme.com</email>
    </principal>
  </principal-list>
</results>
```
principal-list-by-field

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Lists principals that have a specified value in a custom field. Use this action to query custom fields for principals. Use principal-list to get a list of custom fields that are defined for the principal.

In the value parameter, enter the value of a custom database field. The name element returned by principal-list, for example, is a full name concatenated from the first-name (bob) and last-name (jones) database fields. If you search on bob jones, principal-list-by-field does not return a value, unless the full name is defined as a database field (in this case, a custom field defined on principals).

The search is case insensitive, and the query string can contain spaces.

Wildcards are not allowed in the query string. For example, if you enter t*, principal-list-by-field searches for the exact string t*.

The principal-list-by-field action searches in all custom database fields defined for the principal; it does not search principal fields.

Request URL
http://server_name/api/xml
   ?action=principal-list-by-field
   &value=string
   &filter-definition=value
   &sort-definition=value
   &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Y</td>
<td>The value for which you want to search all fields. You do not need to enter a field name.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>
Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <principal-list>
    <principal account-id=integer principal-id=integer type=allowedValue
      has-children=boolean is-primary=boolean is-hidden=boolean>
      <name>string</name>
      <login>string</login>
    </principal>
  </principal-list>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>principal-list</td>
<td></td>
<td>Container</td>
<td>The entire list of principals that match the value in one or more custom fields.</td>
</tr>
<tr>
<td>principal</td>
<td></td>
<td>Container</td>
<td>One principal that matches the value.</td>
</tr>
<tr>
<td></td>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the principal.</td>
</tr>
<tr>
<td></td>
<td>account-id</td>
<td>Integer</td>
<td>The ID of the account the principal belongs to.</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>Allowed value</td>
<td>The type of principal (see type for values).</td>
</tr>
<tr>
<td></td>
<td>has-children</td>
<td>Boolean</td>
<td>Indicates whether the principal has children. Groups have children and users don’t, so this attribute indicates whether the principal is a group.</td>
</tr>
<tr>
<td></td>
<td>is-primary</td>
<td>Boolean</td>
<td>Whether the principal is a built-in group (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td>is-hidden</td>
<td>Boolean</td>
<td>Whether the principal is hidden in the user interface (true) or not (false).</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The principal’s full name, concatenated from the first-name and last-name fields.</td>
</tr>
<tr>
<td>login</td>
<td></td>
<td>String</td>
<td>The principal’s login ID, often an e-mail address.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=principal-list-by-field&value=inactive

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <principal-list>
    <principal account-id="624520" principal-id="2616099" type="user"
      has-children="false" is-primary="false" is-hidden="false">
      <name>Bob Jones</name>
    </principal>
  </principal-list>
</results>
<login>bjones@acme.com</login>
</principal>
</principal-list>
</results>

See also
principal-info, principal-list, principal-update
principals-delete

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Removes one or more principals, either users or groups. To delete principals, you must have Administrator privilege.

To delete multiple principals, specify multiple principal-id parameters. All of the principals you specify will be deleted.

The principal-id can identify either a user or group. If you specify a user, the user is removed from any groups the user belongs to. If you specify a group, the group is deleted, but the users who belong to it are not.

Request URL
http://server_name/api/xml
  ?action=principals-delete
  &principal-id=integer
  &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a user or group you want to delete.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=principals-delete &principal-id=2006339311&principal-id=2006339323

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
See also
principal-info, principal-list, principal-list-by-field, principal-update

principal-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Creates a principal (a user or group) or updates a standard field for a principal. The principal is created or updated in the same account as the user making the call.

To create a new principal, call principal-update without specifying a principal-id. To update, add the principal-id. Before you update metadata about a principal, call principal-info to get the existing version.

If a principal has custom fields, use acl-field-update to update them, rather than principal-update.

You need Administrator privileges to create or update a principal.

Request URL
http://server_name/api/xml
?action=principal-update
&description=string
&email=string
&first-name=string
&has-children=boolean
&last-name=string
&login=string
&name=string
&password=string
&principal-id=integer
&send-email=boolean
&type=allowedValue
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>N</td>
<td>The new group's description. Use only when creating a new group.</td>
</tr>
<tr>
<td>email</td>
<td>String</td>
<td>N</td>
<td>The user's e-mail address. Can be different from the login. Be sure to specify a value if you use send-email=true.</td>
</tr>
<tr>
<td>first-name</td>
<td>String</td>
<td>Y/N</td>
<td>The user's new first name. Use only with users, not with groups. Required to create a user.</td>
</tr>
<tr>
<td>has-children</td>
<td>Boolean</td>
<td>Y</td>
<td>Whether the principal has children. If the principal is a group, use 1 or true. If the principal is a user, use 0 or false.</td>
</tr>
<tr>
<td>last-name</td>
<td>String</td>
<td>Y/N</td>
<td>The new last name to assign to the user. Required to create a user. Do not use with groups.</td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td>Y/N</td>
<td>The principal's new login name, usually the principal's e-mail address. Must be unique on the server. Required to create or update a user. Do not use with groups.</td>
</tr>
</tbody>
</table>
Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <principal principal-id=integer account-id=integer
    has-children=integer type=integer>
    <login>string</login>
    <ext-login>string</ext-login>
    <name>string</name>
  </principal>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty,</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>principal</td>
<td></td>
<td>Container</td>
<td>Information about the newly created principal.</td>
</tr>
<tr>
<td>principal-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the newly created user.</td>
</tr>
<tr>
<td>account-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the account the new user belongs to. Same as the account of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>current user.</td>
</tr>
<tr>
<td>has-children</td>
<td></td>
<td>Boolean</td>
<td>Whether the principal has children, which indicates whether the principal is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a user or group (1 if a group, or 0 if a user).</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed</td>
<td>The type of principal (see type for values).</td>
</tr>
</tbody>
</table>
Sample request

```
https://example.com/api/xml?action=principal-update&first-name=jake
&last-name=doe&has-children=0&login=jakedoe@example.com&type=user
```

Sample response

```
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <principal principal-id="2006403978" account-id="624520" type="user"
    has-children="0">
    <login>jakedoe@example.com</login>
    <ext-login>jakedoe@example.com</ext-login>
    <name>jake doe</name>
  </principal>
</results>
```

See also

```
principal-info, principal-list, principal-list-by-field, acl-field-update
```

**quota-threshold-info**

**Availability**

Acrobat Connect Pro Server 7

**Description**

Provides the list of quotas for which capacity notifications are provided, along with their current threshold settings. Each Acrobat Connect Pro account has system quotas that determine, for example, how many seats are available for Meeting Hosts, Learners, and so on. Each quota has a threshold; when the threshold is crossed, the system notifies administrators that the quota is in danger of being reached. The settings for the threshold and the notifications vary depending on the quota.

**Request URL**

```
https://example.com/api/xml
?action=quota-threshold-info
&account-id=integer
&session=integer
```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account for which you want quota threshold information. If you don't specify an ID, the current account to which the user is logged in is used.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="allowedValue" />  
  <Principals>
    <Principal principal-id="integer" type="string"/>
  </Principals>
  <Quotas>
    <Quota acl-id="integer" quota-id="string" threshold-pct="integer" login-notif="boolean" email-notif="boolean" monthly-emails="boolean" limit="integer" used="integer"/>
    <Quota acl-id="integer" quota-id="string" threshold-pct="integer" login-notif="boolean" email-notif="boolean" monthly-emails="boolean" limit="integer" used="integer"/>
  </Quotas>
  <Trees>
    <Tree tree-id="integer" type="string"/>
  </Trees>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>Principals</td>
<td></td>
<td>Container</td>
<td>Lists the principals specifying the groups for which system capacity notifications are provided.</td>
</tr>
<tr>
<td>Principal</td>
<td></td>
<td>Container</td>
<td>Information about the principal for which system capacity notification is provided.</td>
</tr>
<tr>
<td>principal-id</td>
<td></td>
<td>Integer</td>
<td>The principal ID specifying the group.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>String</td>
<td>The group type. Depending on the license, this value can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• authors</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• live-admins, which specifies that the group is Meeting Hosts</td>
</tr>
<tr>
<td>Quotas</td>
<td></td>
<td>Container</td>
<td>Lists the quotas.</td>
</tr>
<tr>
<td>Quota</td>
<td></td>
<td>Container</td>
<td>Information about the quota and its settings.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>ACL ID of the quota.</td>
</tr>
<tr>
<td>quota-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the quota. For possible values, see &quot;quota-ID&quot; on page 210.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=quota-threshold-info&account-id=7

Sample response
<?xml version="1.0" encoding="utf-8" >
<results>
  <status code="ok"/>
  <Principals>
    <Principal principal-id="20013" type="authors"/>
    <Principal principal-id="10051" type="live-admins"/>
  </Principals>
  <Quotas>
    <Quota acl-id="7" quota-id="training-user" threshold-pct="90" login-notif="true" email-notif="true" monthly-emails="true" limit="10" used="0"/>
    <Quota acl-id="20013" quota-id="num-of-members-quota" threshold-pct="10" login-notif="true" email-notif="true" monthly-emails="true" limit="10" used="3"/>
  </Quotas>
  <Trees/>
</results>

quota-threshold-exceeded

Availability
Acrobat Connect Pro Server 7

Description
Returns information about system quota thresholds that have been exceeded.
Each Acrobat Connect Pro account has system quotas that determine, for example, how many seats are available for Meeting Hosts, Learners, and so on. Each quota has a threshold; when the threshold is crossed, the system notifies administrators that the quota is in danger of being reached. The threshold varies depending on the quota. For more information about automatic notification, see *Acrobat Connect Pro User Guide*.

**Request URL**

https://example.com/api/xml

?action=quota-threshold-exceeded

&account-id=Integer

&acl-id=Integer

&quota-id=String

&num-of-days=Integer

&session=String

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account. Specify either account-id or acl-id (not both). If you do not specify either account-id or acl-id, results are returned for the account which the current user is logged into. If you specify account-ID, results are returned for all quota thresholds that have been reached in the account.</td>
</tr>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the SCO, account, or principal for which you want threshold information. Can be a valid sco-id, account-id, or principal-id. If you do not specify a value for acl-id, the value for account-id is used. The value to use for acl-id depends on the quota ID used; for more information, see &quot;quota-ID&quot; on page 210</td>
</tr>
<tr>
<td>quota-id</td>
<td>String</td>
<td>N</td>
<td>The ID of the system quota for which you want information. For available values, see &quot;quota-ID&quot; on page 210.</td>
</tr>
<tr>
<td>num-of-days</td>
<td>Integer</td>
<td>N</td>
<td>Number of days from the current day for which records are retrieved. If you do not specify a value, all the previous records for the specified quotas are retrieved.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <Records>
    <Record acl-id="Integer" quota-id="String" peak-used="Integer" count="Integer" threshold-pct="Integer" sco-id="Integer">
      <record-date>Date</record-date>
    </Record>
  </Records>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>Records</td>
<td></td>
<td>Container</td>
<td>Lists the records returned.</td>
</tr>
<tr>
<td>Record</td>
<td></td>
<td>Container</td>
<td>Lists information about the record returned.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the ACL returned.</td>
</tr>
<tr>
<td>quota-id</td>
<td></td>
<td>String</td>
<td>The ID of the quota returned.</td>
</tr>
<tr>
<td>peak-used</td>
<td></td>
<td>Integer</td>
<td>The peak value of quota used on the specified date. This attribute is null for training and group quotas.</td>
</tr>
<tr>
<td>count</td>
<td></td>
<td>Integer</td>
<td>Number of times the threshold was crossed for the quota on the specified date. This attribute is null for training and group quotas.</td>
</tr>
<tr>
<td>threshold-pct</td>
<td></td>
<td>Integer</td>
<td>Percentage threshold when the threshold was crossed.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the meeting for which the threshold was crossed. This attribute is applicable only for the quota ID concurrent-users-per-meeting; for other quotas, it is null.</td>
</tr>
<tr>
<td>record-date</td>
<td></td>
<td>Date</td>
<td>Date when the threshold was crossed (UTC), in MM/DD/YYYY format.</td>
</tr>
</tbody>
</table>

Sample request


Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok"/>
  <Records>
    <Record acl-id="20013" quota-id="num-of-members-quota" peak-used="" count=""
      threshold-pct="10" sco-id="">
      <record-date>11/20/2007</record-date>
    </Record>
  </Records>
</results>
```

quota-threshold-update

Availability

Acrobat Connect Pro Server 7

Description

Updates the threshold settings of the specified quotas.
Each Acrobat Connect Pro account has system quotas that determine, for example, how many seats are available for meeting hosts, trainers, training managers, and so on. Each quota has a threshold; when the threshold is crossed, the system notifies administrators that the quota is in danger of being reached. The settings for the thresholds and notifications vary depending on the quota, and you can configure the settings using this action.

**Request URL**

https://example.com/api/xml

?action=quota-threshold-update

&account-id=integer

&acl-id=integer

&quota-id=string

&threshold-pct=integer

&login-notif=Boolean

&email-notif=Boolean

&monthly-emails=Boolean

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the account for which quota settings are updated.</td>
</tr>
</tbody>
</table>
| acl-id       | Integer | Y        | The ID of the SCO, account, or principal for which you want threshold informa-
|              |         |          | tion. Can be a valid sco-id, account-id, or principal-id. If you do not specify a
|              |         |          | value for acl-id, the value for account-id is used.                         |
|              |         |          | The value to use for acl-id depends on the quota ID used; for more information,
|              |         |          | see “quota-ID” on page 210.                                                 |
| quota-id     | String  | Y        | The ID of the quota whose settings are updated. For available values, see
|              |         |          | “quota-ID” on page 210.                                                     |
| threshold-pct| Integer | Y        | The percent threshold for the quota. The lower the value, the more frequently
|              |         |          | administrators are notified when the threshold is exceeded (if notifications are enabled). |
| login-notif  | Boolean | Y        | Specifies whether to notify administrators upon logging in that a threshold is
|              |         |          | exceeded (true) or not (false).                                             |
| email-notif  | Boolean | Y        | Specifies whether to notify administrators through e-mail that a threshold is
|              |         |          | exceeded (true) or not (false).                                             |
| monthly-emails| Boolean | Y        | Specifies whether to send administrators monthly threshold reports through e-mail
|              |         |          | (true) or not (false).                                                      |
| session      | String  | N        | The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library. |

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code=allowedValue />
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request


Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

See also
quota-threshold-info, quota-threshold-exceeded

### report-active-meetings

**Availability**

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**

Returns a list of Adobe® Acrobat® Connect™ Pro meetings that are currently in progress, including the number of minutes the meeting has been active.

For report-active-meetings to return results, at least one user must be present in at least one meeting room. If meetings are scheduled at present, but no users are attending those meetings, report-active-meetings returns an empty response.

**Request URL**

http://server_name/api/xml

?action=report-active-meetings
&session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

Results cannot be filtered or sorted.

**Response structure**

```xml
<results>
  <status code="allowedValue" />
</results>
```
<report-active-meetings>
  <sco sco-id="integer" active-participants="integer"
    length-minutes="integer">
    <name>string</name>
    <url-path>string</url-path>
    <date-begin>datetime</date-begin>
  </sco>
</report-active-meetings>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-active-meetings</td>
<td></td>
<td>Container</td>
<td>The list of all meetings currently in progress.</td>
</tr>
<tr>
<td>sco</td>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of a meeting in progress.</td>
</tr>
<tr>
<td></td>
<td>active-participants</td>
<td>Integer</td>
<td>The number of users attending the meeting in progress, including hosts and presenters.</td>
</tr>
<tr>
<td></td>
<td>length-minutes</td>
<td>Integer</td>
<td>The number of minutes the meeting has been active.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the meeting, defined when the meeting was created.</td>
</tr>
<tr>
<td>url-path</td>
<td></td>
<td>String</td>
<td>The part of the meeting URL that comes after the domain and is unique to this meeting.</td>
</tr>
<tr>
<td>date-begin</td>
<td></td>
<td>Datetime</td>
<td>The date and time the meeting began.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-active-meetings

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-active-meetings>
    <sco sco-id="2006778715" active-participants="" length-minutes="1">
      <name>Designing Online Courses</name>
      <url-path>/online/</url-path>
      <date-begin>2006-06-28T14:35:21.307-07:00</date-begin>
    </sco>
  </report-active-meetings>
</results>
report-bulk-consolidated-transactions

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about principal-to-SCO transactions on your Acrobat Connect Pro server or in your Acrobat Connect Pro hosted account.

A transaction is an instance of one principal visiting one SCO. The SCO can be a Acrobat Connect Pro meeting, course, document, or any content on the server.

These are all examples of transactions:
- If a principal attends a meeting twice, two transactions exist: one for each time the principal attended the meeting.
- If five people attend a meeting, five transactions exist: one for each user who attended the meeting.
- If a principal takes two courses three times each and passes each only on the third try, six transactions exist: one for each attempt on each course.

This call returns all transactions, so consider using a filter to reduce the volume of the response. For example, if you use filter-type=meeting, the call returns all meeting transactions:

https://example.com/api/xml?action=report-bulk-consolidated-transactions
&filter-type=meeting

From the response, you can calculate Acrobat Connect Pro meeting usage by comparing times in date-created and date-closed (see “Calculate meeting usage”). However, this call to `report-bulk-consolidated-transactions`, with filter-type=meeting, returns only users who logged in to the meeting as participants, not users who entered the meeting as guests.

Request URL
http://server_name/api/xml
?action=report-bulk-consolidated-transactions
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<results>
  <status code=allowedValue />
  <report-bulk-consolidated-transactions>
```
<row transaction-id=integer sco-id=integer type=allowedValue
        principal-id=integer score=integer>
    <name>string</name>
    <url>relativeUrl</url>
    <login>string</login>
    <user-name>string</user-name>
    <status>allowedValue</status>
    <date-created>datetime</date-created>
    <date-closed>datetime</date-closed>
</row>

...<report-bulk-consolidated-transactions>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-bulk-consolidated-transactions</td>
<td>Container</td>
<td>The entire list of transactions that matches the request.</td>
<td></td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Details of one transaction that matches the request.</td>
</tr>
<tr>
<td></td>
<td>transaction-id</td>
<td>Integer</td>
<td>The ID of the transaction.</td>
</tr>
<tr>
<td></td>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the object (SCO) the user interacted with.</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>Allowed value</td>
<td>The type of the SCO (see type for allowed values).</td>
</tr>
<tr>
<td></td>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the principal involved in the transaction.</td>
</tr>
<tr>
<td></td>
<td>score</td>
<td>Integer</td>
<td>If the transaction (such as a quiz) assigned a score, the actual score.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name assigned to the SCO involved in the transaction.</td>
</tr>
<tr>
<td>url</td>
<td></td>
<td>String</td>
<td>The file name portion of the URL to the SCO involved in the transaction.</td>
</tr>
<tr>
<td>login</td>
<td></td>
<td>String</td>
<td>The principal’s login ID.</td>
</tr>
<tr>
<td>user-name</td>
<td></td>
<td>String</td>
<td>The full name of the user involved in the transaction (concatenated from first-name and last-name).</td>
</tr>
</tbody>
</table>
Sample request

https://example.com/api/xml?action=report-bulk-consolidated-transactions
&filter-type=meeting

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-bulk-consolidated-transactions>
    <row transaction-id="2006905086" sco-id="2006905049" type="meeting"
     principal-id="2006258745" score="0">
      <name>Celebrate End of June Meeting</name>
      <url>/endjune/</url>
      <login>joy@acme.com</login>
      <user-name>Joy Smith</user-name>
      <status>completed</status>
      <date-created>2006-06-30T11:10:37.003-07:00</date-created>
      <date-closed>2006-06-30T11:45:21.397-07:00</date-closed>
    </row>
    <row transaction-id="2006905795" sco-id="2006905049" type="meeting"
     principal-id="2006258745" score="0">
      <name>Celebrate End of June Meeting</name>
      <url>/endjune/</url>
      <login>joy@acme.com</login>
      <user-name>Joy Smith</user-name>
      <status>completed</status>
      <date-created>2006-06-30T11:10:37.003-07:00</date-created>
      <date-closed>2006-06-30T11:45:21.397-07:00</date-closed>
    </row>
    ...
  </report-bulk-consolidated-transactions>
</results>
```

See also

report-bulk-objects, report-bulk-questions, report-bulk-slide-views, report-bulk-users

### report-bulk-objects

**Availability**

Breeze 5; Connect Enterprise Web Services 6
Description

Returns information about all objects (SCOs) on a licensed Acrobat Connect Pro Server or in a Acrobat Connect Pro hosted account. The object types returned include archive, attachment, authorware, captivate, course, curriculum, external-event, flv, image, meeting, presentation, and swf.

Because the response is likely to be large, use filters to limit it. For example, to return a list of all meetings on the server, filter on the type field:

http://example.com/api/xml?action=report-bulk-objects&filter-type=meeting

Request URL

http://server_name/api/xml
?action=report-bulk-objects
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
 <status code=allowedValue />
 <report-bulk-objects>
  <row sco-id=integer type=allowedValue>
   <url>string</url>
   <name>string</name>
   <date-created>datetime</date-created>
   <date-end>datetime</date-end>
   <date-modified>datetime</date-modified>
   <description>datetime</description>
  </row>
  ...
 </report-bulk-objects>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-bulk-objects</td>
<td></td>
<td>Container</td>
<td>The entire list of SCOs on the server; or, if a filter is used, the entire list of SCOs that matches the filter.</td>
</tr>
</tbody>
</table>
Sample request
http://example.com/api/xml?action=report-bulk-objects&filter-type=meeting &filter-gt-date-created=2006-06-01

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-bulk-objects>
    <row sco-id="2006778715" type="meeting">
      <url>/online/</url>
      <name>Designing Online Courses</name>
      <date-created>2006-06-28T14:15:00.000-07:00</date-created>
      <date-end>2006-06-28T14:30:00.000-07:00</date-end>
      <date-modified>2006-07-13T14:57:54.150-07:00</date-modified>
    </row>
    ...
  </report-bulk-objects>
</results>

See also

report-bulk-questions

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about every quiz question in the account you are logged in to.
The response includes a combination of the quiz question, the answer, the ID of the user who answered, and the ID of the transaction.
This action returns all question-and-answer combinations in the account, unless you use a filter to limit the size of the response.

Request URL

http://server_name/api/xml

?action=report-bulk-questions
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-bulk-questions>
    <row transaction-id=integer score=integer principal-id=integer>
      <question>string</question>
      <response>string</response>
      <date-created>datetime</date-created>
    </row>
    ...
  </report-bulk-questions>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-bulk-questions</td>
<td>Container</td>
<td></td>
<td>The entire list of question-and-answer combinations that match the request.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Details about one question-and-answer combination.</td>
</tr>
<tr>
<td></td>
<td>transaction-id</td>
<td>Integer</td>
<td>The ID of the interaction between a user and a quiz.</td>
</tr>
<tr>
<td></td>
<td>score</td>
<td>Integer</td>
<td>The score assigned to the question.</td>
</tr>
<tr>
<td></td>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the user who answered or viewed the question.</td>
</tr>
</tbody>
</table>
Using Acrobat Connect Pro Web Services

Sample request
https://example.com/api/xml?action=report-bulk-questions

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-bulk-questions>
    <row transaction-id="2006335803" score="10" principal-id="2006258745">
      <question>The capital of California is</question>
      <response>Sacramento</response>
      <date-created>2006-05-11T15:50:23.643-07:00</date-created>
    </row>
    <row transaction-id="2006335827" score="0" principal-id="2006258745">
      <question>The capital of California is</question>
      <response>san francisco</response>
      <date-created>2006-05-11T17:32:53.970-07:00</date-created>
    </row>
  </report-bulk-questions>
</results>

See also

report-bulk-slide-views

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about each occasion on which a principal views a slide. The slide can be in any presentation in the account the current user belongs to.

Each slide view is a transaction. A transaction is an interaction between a user and any SCO on Acrobat Connect Pro. In this case, the transaction is between a user and a slide.

This action returns all occurrences of principals viewing slides in the account, unless you filter the response.

Request URL
http://server_name/api/xml
?action=report-bulk-slide-views
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="allowedValue" />
  <report-bulk-slide-views>
    <row transaction-id="integer" principal-id="integer">
      <page>integer</page>
      <date-created>datetime</date-created>
    </row>
    ...
  </report-bulk-slide-views>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>All results the action returns.</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
<td></td>
</tr>
<tr>
<td>report-bulk-slide-views</td>
<td>Container</td>
<td>The entire list of slide views that match the request.</td>
<td></td>
</tr>
<tr>
<td>row</td>
<td>Container</td>
<td>Details about one slide view.</td>
<td></td>
</tr>
<tr>
<td>transaction-id</td>
<td>Integer</td>
<td>The ID of the interaction between the user and the slide.</td>
<td></td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the user who viewed the slide.</td>
<td></td>
</tr>
<tr>
<td>page</td>
<td>Integer</td>
<td>The page number of the slide in the presentation.</td>
<td></td>
</tr>
<tr>
<td>date-created</td>
<td>Datetime</td>
<td>The date and time the user viewed the slide.</td>
<td></td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-bulk-slide-views
&filter-principal-id=123456

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=ok />
  <report-bulk-slide-views>
    <row transaction-id="2006334916" principal-id="123456">
      <page>0</page>
      <date-created>2006-05-11T12:02:01.470-07:00</date-created>
    </row>
    ...
  </report-bulk-slide-views>
</results>
```
See also
report-bulk-objects, report-bulk-questions, report-bulk-consolidated-transactions, report-bulk-users

report-bulk-users

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about all users in an account. The difference between this call and principal-list is that principal-list returns both users and groups, while report-bulk-users returns only users.

The response from report-bulk-users can be quite large, especially if you use custom fields, so remember that you can filter and sort it. For example, the following call returns a list of all users who have the letters Jo in their name, in ascending order by name:

http://myserver.com/api/xml?action=report-bulk-users&sort-name=asc
&filter-like-name=Jo

If you pass custom-fields=true, by default report-bulk-users returns up to eight custom fields defined for users. If you have defined more than eight custom fields for users, report-bulk-users returns the first eight in the list in the Customize User Profile screen in Acrobat Connect Pro Central (at Administration > Users and Groups > Customize User Profile).

If you use Acrobat Connect Pro Server, you can set a value for REPORT_MAX_CUSTOM_FIELDS in the custom.ini file to have report-bulk-users return more than eight custom fields (for more information, see Adobe Acrobat Connect Pro Installation and Configuration Guide). You can use any value, but higher values risk a greater impact to database performance. You cannot change this setting on a Acrobat Connect Pro hosted account.

Request URL
http://server_name/api/xml
?action=report-bulk-users
&custom-fields=boolean
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom-fields</td>
<td>Boolean</td>
<td>N</td>
<td>Whether to return custom fields in the response. Returns up to eight custom fields. If true, the manager field is not returned in the response.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

You can use filter-type with report-bulk-users to filter the type of users returned (user or guest).

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code=allowedValue />
   <report-bulk-users>
      <row principal-id=integer type="string">  
         <login>string</login>
         <name>string</name>
         <email>string</email>
         <manager>string</manager>
         .. any custom fields ..
      </row>
      ...
   </report-bulk-users>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>Top-level element for the response.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-bulk-users</td>
<td></td>
<td>Container</td>
<td>The entire list of users in the account.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Details about one user in the account.</td>
</tr>
<tr>
<td></td>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the user.</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>String</td>
<td>The type of user, either user or guest.</td>
</tr>
<tr>
<td></td>
<td>login</td>
<td>String</td>
<td>The user’s login ID, often an e-mail address.</td>
</tr>
</tbody>
</table>
Sample request

https://example.com/api/xml?action=report-bulk-users&filter-like-name=john

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-bulk-users>
    <row principal-id="5417288" type="guest">
      <login>john@example.com</login>
      <name>John Owens</name>
      <email>john@example.com</email>
    </row>
    <row principal-id="5417255" type="user">
      <login>jsmith@example.com</login>
      <name>John Smith</name>
      <email>jsmith@example.com</email>
    </row>
    ...
  </report-bulk-users>
</results>
```

See also

report-bulk-objects, report-bulk-questions, report-bulk-slide-views, report-bulk-consolidated-transactions

### report-course-status

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Returns summary information about a course, including the number of users who have passed, failed, and completed the course, as well as the current number of enrollees. The request requires the *sco-id* of a course. Acrobat Connect Pro Central uses this call to display Course Status in the Summary report. This report is available at Training > Shared Training > [course name] > Reports > Summary.

**Request URL**

http://server_name/api/xml?
?action=report-course-status
&sco-id=integer
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of the course for which you want summary information.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<results>
  <status code="allowedValue" />
  <report-course-status total-course-completions="integer"
    total-unique-course-completions="integer",
    num-passed="integer",
    num-failed="integer",
    num-enrollees="integer" />  
  <date-last-taken>datetime</date-last-taken>
</report-course-status>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-course-status</td>
<td></td>
<td>Container</td>
<td>Summary information about a course.</td>
</tr>
<tr>
<td>total-course-completions</td>
<td></td>
<td>Integer</td>
<td>The total number of times users have completed the course, including passing scores, failing scores, and multiple attempts by the same user.</td>
</tr>
<tr>
<td>total-unique-course-completions</td>
<td></td>
<td>Integer</td>
<td>The number of distinct users who have completed the course, including passing and failing scores but not multiple attempts by the same user.</td>
</tr>
<tr>
<td>num-completed</td>
<td></td>
<td>Integer</td>
<td>The number of users who have completed the course, for courses that do not have a passing score.</td>
</tr>
<tr>
<td>num-passed</td>
<td></td>
<td>Integer</td>
<td>The number of users who have passed the course, for courses that have a passing score.</td>
</tr>
</tbody>
</table>
Using Acrobat Connect Pro Web Services

Sample request
https://example.com/api/xml?action=report-course-status&sco-id=123456

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-course-status total-course-completions="1" total-unique-course-completions="1" num-completed="0" num-passed="1" num-failed="0" num-enrollees="4">
    <date-last-taken>2006-10-10T13:55:24.480-07:00</date-last-taken>
  </report-course-status>
</results>

report-curriculum-taker

Availability
Connect Enterprise Web Services 6

Description
Returns information about a user's progress in a curriculum.
The response includes a row element for each course in the curriculum, which has information such as access to the course, whether credit was granted, the user's score, the unique url-path to the course, and so on.

Request URL
http://server_name/api/xml?action=report-curriculum-taker
&user-id=integer
&sco-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the user whose scores you want to check.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of the curriculum for which you want a summary.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>
Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?
<results>
  <status code=allowedValue />
  <report-curriculum-taker>
    <sco transcript-id=integer path-type=allowedValue asset-id=integer sco-id=integer depth=integer folder-id=integer type=integer icon=allowedValue lang=allowedValue max-retries=integer source-sco-id=integer source-sco-type=allowedValue status=allowedValue score=integer certificate=integer max-score=integer attempts=integer>
      <access>allowedValue</access>
      <credit-granted>boolean</credit-granted>
      <name>string</name>
      <url-path>string</url-path>
      <date-modified>datetime</date-modified>
      <override>boolean</override>
    </sco>
  </report-curriculum-taker>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-curriculum-taker</td>
<td>Container</td>
<td>Information about the user's performance in the entire curriculum.</td>
<td></td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Information about the user's work with one course or curriculum.</td>
</tr>
<tr>
<td>transcript-id</td>
<td>Integer</td>
<td>The ID of the user's transcript for the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>path-type</td>
<td></td>
<td>Allowed value</td>
<td>The learning path a user must take before attempting this course or curriculum (see path-type for allowed values).</td>
</tr>
<tr>
<td>asset-id</td>
<td>Integer</td>
<td>The version of the course or curriculum the user attempted to complete. The asset-id is incremented each time the course or curriculum has new content uploaded.</td>
<td></td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>depth</td>
<td>Integer</td>
<td>A course's level below the curriculum in the navigation hierarchy. For a curriculum, 0; for a course one level below the curriculum, 1.</td>
<td></td>
</tr>
<tr>
<td>folder-id</td>
<td>Integer</td>
<td>The ID of the folder that contains the course or curriculum. For a course, the ID of a curriculum; for a curriculum, the ID of a user.</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>Integer</td>
<td>The type of the course or curriculum (see type for allowed values).</td>
<td></td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td>The type of icon that identifies the course or curriculum in Acrobat Connect Pro Central (see icon for values).</td>
<td></td>
</tr>
<tr>
<td>lang</td>
<td>Allowed value</td>
<td>The language associated with the course or curriculum (see lang for values).</td>
<td></td>
</tr>
<tr>
<td>max-retries</td>
<td>Integer</td>
<td>The maximum number of times a user can retake the course or curriculum. If a user can take the course 3 times, max-retries is 2.</td>
<td></td>
</tr>
<tr>
<td>source-sco-id</td>
<td>Integer</td>
<td>The unique ID of the SCO used as a template for the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>source-sco-type</td>
<td>Integer</td>
<td>The type of SCO used as a template for the course or curriculum (see type for values).</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Allowed value</td>
<td>The status of the user's attempt to use the course or curriculum. For courses, allowed values are completed, incomplete, user-passed, user-failed, and not-attempted. For curriculums and folders, allowed values are completed and incomplete.</td>
<td></td>
</tr>
<tr>
<td>score</td>
<td>Integer</td>
<td>The score the user earned on the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>certificate</td>
<td>Integer</td>
<td>The ID of the user's certificate.</td>
<td></td>
</tr>
<tr>
<td>max-score</td>
<td>Integer</td>
<td>The maximum score possible for the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>attempts</td>
<td>Integer</td>
<td>The number of times the user has attempted the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>access</td>
<td>Allowed value</td>
<td>The level of access the user has to the course or curriculum (see access for allowed values).</td>
<td></td>
</tr>
<tr>
<td>credit-granted</td>
<td>Boolean</td>
<td>A value indicating whether credit was granted for the course or curriculum.</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the learning object or curriculum.</td>
<td></td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td>The part of the URL after the domain name that uniquely identifies the object on the server.</td>
<td></td>
</tr>
<tr>
<td>date-modified</td>
<td>Datetime</td>
<td>The date and time the SCO was last modified, in ISO 8601 format.</td>
<td></td>
</tr>
<tr>
<td>override</td>
<td>Boolean</td>
<td>A value indicating whether the transcript for the SCO has been adjusted.</td>
<td></td>
</tr>
</tbody>
</table>

**Sample request**

https://example.com/api/xml?action=report-curriculum-taker
&user-id=2006258748&sco-id=2006298444
Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-curriculum-taker>
    <sco transcript-id="2006905613" path-type="prereq-none"
      asset-id="2006334911" sco-id="2006334909" depth="0"
      folder-id="2006258747" type="content" icon="producer"
      lang="en" max-retries="" source-sco-id="" source-sco-type=""
      status="user-failed" score="0" certificate="" max-score="0"
      attempts="5">
      <access>access-open</access>
      <credit-granted>false</credit-granted>
      <name>Test Quiz</name>
      <url-path>/quiz/</url-path>
      <date-created>2006-06-30T15:24:34.897-07:00</date-created>
      <date-modified>2006-05-16T15:22:25.703-07:00</date-modified>
      <date-taken>2006-06-30T15:24:34.897-07:00</date-taken>
      <override>false</override>
    </sco>
  </report-curriculum-taker>
</results>
```

`report-meeting-attendance`

**Availability**

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**

Returns a list of users who attended a Acrobat Connect Pro meeting. The data is returned in row elements, one for each person who attended. If the meeting hasn’t started or had no attendees, the response contains no rows. The response does not include meeting hosts or users who were invited but did not attend.

To call `report-meeting-attendance`, you must have publish, mini-host, or host permission on the meeting (see `permission-id` for details).

**Request URL**

http://server_name/api/xml

?action=report-meeting-attendance
&sco-id=integer
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a meeting.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>
Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="allowedValue" />
  <report-meeting-attendance>
    <row transcript-id="integer" sco-id="integer" principal-id="integer"
        answered-survey="boolean">
      <login>string</login>
      <session-name>string</session-name>
      <sco-name>string</sco-name>
      <date-created>datetime</date-created>
      <date-end>datetime</date-end>
      <participant-name>string</participant-name>
    </row>
  ...
</report-meeting-attendance>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Container</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-meeting-attendance</td>
<td>Container</td>
<td>Container</td>
<td>The entire list of attendees for the meeting.</td>
</tr>
<tr>
<td>row</td>
<td>Container</td>
<td>Data about one meeting attendee.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transcript-id</td>
<td>Integer</td>
<td>The ID of the meeting transcript.</td>
</tr>
<tr>
<td></td>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the meeting.</td>
</tr>
<tr>
<td></td>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the principal who attended the meeting.</td>
</tr>
<tr>
<td></td>
<td>answered-survey</td>
<td>Boolean</td>
<td>Whether the meeting participant responded to a meeting poll. If 0 or false, the meeting did not have a poll or the participant did not respond (if 1 or true, the opposite). This value is updated when the poll is closed.</td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td>The meeting attendee's login name.</td>
<td></td>
</tr>
<tr>
<td>session-name</td>
<td>String</td>
<td>The name of the user who entered the meeting room, creating a session.</td>
<td></td>
</tr>
<tr>
<td>sco-name</td>
<td>String</td>
<td>The name of the meeting.</td>
<td></td>
</tr>
<tr>
<td>date-created</td>
<td>Datetime</td>
<td>The date the meeting was created.</td>
<td></td>
</tr>
<tr>
<td>date-end</td>
<td>Datetime</td>
<td>The date the meeting ended.</td>
<td></td>
</tr>
<tr>
<td>participant-name</td>
<td>String</td>
<td>The name of the meeting attendee as registered with the server.</td>
<td></td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=report-meeting-attendance
&SCO-ID=2006778715

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-meeting-attendance>
    <row transcript-id="2006778723" SCO-ID="2006778715"
        principal-id="2006258745" answered-survey="0">
      <login>joy@acme.com</login>
      <session-name>Joy Smith</session-name>
      <sco-name>Designing Online Courses</sco-name>
      <date-created>2006-06-28T14:35:21.307-07:00</date-created>
      <date-end>2006-06-28T15:09:05.447-07:00</date-end>
      <participant-name>Joy Smith</participant-name>
    </row>
  </report-meeting-attendance>
</results>

report-meeting-concurrent-users

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Returns the maximum number of users in Acrobat Connect Pro meetings concurrently in the last 30 days, and the number of times the maximum has been reached. The maximum is the peak number of users in any meetings at a single moment, whether one meeting, multiple concurrent meetings, or multiple overlapping meetings.

You can change the time period to a period greater than 30 days by adding a length parameter, for example, length=120.

The maximum number of users (max-users) is determined by the account license and applies to the server overall, not to a specific meeting. This action also returns the number of times in the current month the maximum has been reached (max-participants-freq).

Request URL
http://server_name/api/xml
?action=report-meeting-concurrent-users
&length=integer
&session=BREEZESSESSIONCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td>Integer</td>
<td>N</td>
<td>The number of days in the time period to check for concurrent meeting usage. Use a value greater than 30. The default value is 30.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>
Filters
Results cannot be filtered or sorted.

Response structure
```xml
<results>
  <status code=allowedValue />
  <report-meeting-concurrent-users max-users=integer
      max-participants-freq=integer />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Attribute</td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>attributes</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-meeting-concurrent-users</td>
<td></td>
<td>Empty, with</td>
<td>Information about the peak number of users in meetings at the same moment.</td>
</tr>
<tr>
<td></td>
<td>max-users</td>
<td>Integer</td>
<td>The peak number of users in meetings at the same moment (either a single meeting or concurrent meetings) during the time period.</td>
</tr>
<tr>
<td></td>
<td>max-participants-freq</td>
<td>Integer</td>
<td>The number of times the maximum has been reached in the time period.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=report-meeting-concurrent-users

Sample response
```xml
<results>
  <status code="ok" />
  <report-meeting-concurrent-users max-users="400"
      max-participants-freq="1" />
</results>
```

report-meeting-sessions

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about all the sessions of a Acrobat Connect Pro meeting. A session is created when a participant enters an empty meeting. As more participants join the meeting, they join the session. The session ends when all attendees leave the meeting. When a new participant enters the now-empty meeting, a new session starts. For example, a recurring weekly meeting has a session each week when the meeting is held.

You can call `report-meeting-sessions` on past meetings, active meetings, or future meetings, but future meetings are not likely to have sessions.
Request URL

http://server_name/api/xml
  ?action=report-meeting-sessions
  &sco-id=integer
  &filter-definition=value
  &sort-definition=value
  &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a meeting for which you want session information.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-meeting-sessions>
    <row sco-id=integer asset-id=integer version=integer
         num-participants=integer>
      <date-created>datetime</date-created>
      <date-end>datetime</date-end>
    </row>
    ...
  </report-meeting-sessions>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td></td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td></td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td></td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-meeting-sessions</td>
<td>Container</td>
<td></td>
<td>The entire list of sessions for the meeting.</td>
</tr>
<tr>
<td>row</td>
<td>Container</td>
<td></td>
<td>Information about one session.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the meeting.</td>
</tr>
<tr>
<td>asset-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the session.</td>
</tr>
<tr>
<td>version</td>
<td>Integer</td>
<td></td>
<td>A sequential ID for the session, starting at 1.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=report-meeting-sessions
   &sco-id=2006811328

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-meeting-sessions>
    <row sco-id="2006811328" asset-id="2006811333" version="1"
         num-participants="1">
      <date-created>2006-06-29T11:46:52.210-07:00</date-created>
      <date-end>2006-06-29T13:34:43.410-07:00</date-end>
    </row>
  </report-meeting-sessions>
</results>

report-meeting-summary

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Returns summary information about a specific Acrobat Connect Pro meeting. The results indicate how many users
were invited, how many invited participants and guests attended, and other information about the meeting.
To use report-meeting-summary, you need publish, host, or mini-host permission on the meeting. With one
of these permissions, you can run report-meeting-summary on a current, completed, or future meeting. The
results are most useful for a completed meeting.
A meeting might be recurring (for example, a weekly staff meeting) and have an occurrence each time the meeting
is held. If the meeting is recurring, the statistics returned by report-meeting-summary are cumulative, applying to
all occurrences of the meeting, not just the latest one.

Request URL
http://server_name/api/xml
?action=report-meeting-summary
&sco-id=integer
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a meeting for which you have publish or host permission.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-meeting-summary num-unique-meetings=integer peak-users=integer
                           num-invitees=integer num-invitees-attended=integer ispublic=boolean
                           num-guests-attended=integer />
</results>
```

Returned XML elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-meeting-summary</td>
<td></td>
<td>Empty, with attributes</td>
<td>Details about the meeting or meeting series.</td>
</tr>
<tr>
<td>num-unique-meetings</td>
<td></td>
<td>Integer</td>
<td>The number of occurrences of a recurring meeting.</td>
</tr>
<tr>
<td>peak-users</td>
<td></td>
<td>Integer</td>
<td>The highest number of participants in the meeting room at one time, during any one meeting occurrence.</td>
</tr>
<tr>
<td>num-invitees</td>
<td></td>
<td>Integer</td>
<td>The number of users who were invited.</td>
</tr>
<tr>
<td>num-invitees-attended</td>
<td></td>
<td>Integer</td>
<td>The number of invited users who attended.</td>
</tr>
<tr>
<td>ispublic</td>
<td></td>
<td>Boolean</td>
<td>Whether the meeting is public and guests can enter automatically (if 1 or true), or private and must wait for permission (if 0 or false).</td>
</tr>
<tr>
<td>num-guests-attended</td>
<td></td>
<td>Integer</td>
<td>The number of participants who entered the meeting room as guests rather than as registered attendees.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-meeting-summary&sco-id=200634033

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
```
report-my-courses

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about each course the current user is or was enrolled in.

The returned courses include future courses, past courses, and courses the user is presently taking. The list of courses can be quite large, so remember to use a filter to reduce the response.

Each course has a permission-id that shows the level of access the user has to the course. For example, the access might be view, publish, or manage.

Request URL
http://server_name/api/xml?action=report-my-courses &filter-definition=value &sort-definition=value &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <my-courses>
    <course sco-id=integer type="content" icon="course"
       permission-id=allowedValue>
      <name>string</name>
      <description>string</description>
      <url>string</url>
      <date-created>datetime</date-created>
    </course>
  </my-courses>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td></td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>Code</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>my-courses</td>
<td>Container</td>
<td></td>
<td>Information about all courses the user is enrolled in.</td>
</tr>
<tr>
<td>course</td>
<td>Container</td>
<td></td>
<td>Information about one course the user is enrolled in.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the course.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td></td>
<td>The type of the course (for allowed values, see type).</td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td></td>
<td>The type of icon that identifies the course in the user interface. For a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>course, always course.</td>
</tr>
<tr>
<td>permission-id</td>
<td>Allowed value</td>
<td></td>
<td>The level of permission the user has on the course (see permission-id for values).</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
<td>The name of the course.</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td></td>
<td>The URL at which a user can reach the course on the server. Includes the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>domain name and the course identifier.</td>
</tr>
<tr>
<td>date-created</td>
<td>Datetime</td>
<td></td>
<td>The date and time the course was created.</td>
</tr>
<tr>
<td>date-modified</td>
<td>Datetime</td>
<td></td>
<td>The date and time the course was last modified.</td>
</tr>
<tr>
<td>date-begin</td>
<td>Datetime</td>
<td></td>
<td>The date and time the course is available for users to start.</td>
</tr>
<tr>
<td>date-end</td>
<td>Datetime</td>
<td></td>
<td>The date and time the course closes.</td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td></td>
<td>The part of the course URL that is the course identifier, after the domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>name.</td>
</tr>
<tr>
<td>expired</td>
<td>Boolean</td>
<td></td>
<td>Whether the course has expired (true if it has, false if it has not).</td>
</tr>
<tr>
<td>completed</td>
<td>Boolean</td>
<td></td>
<td>Whether the user has completed the course (true if yes, false if no).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-my-courses

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <my-courses>
    <course sco-id="2006298431" type="content" icon="course"
             permission-id="view">
      <name>Test Course</name>
    </course>
  </my-courses>
</results>
report-my-events

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about each event the current user has attended or is scheduled to attend. The user can be either a host or a participant in the event. The events returned are those in the user’s my-events folder.

To obtain information about all events on your Acrobat Connect Pro Server or in your Acrobat Connect Pro hosted account, call sco-shortcuts to get the sco-id of the events folder. Then, call sco-contents with the sco-id to list all events.

Request URL
http://server_name/api/xml
?action=report-my-events
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <my-events>
    <event sco-id=integer type="event" icon="event"
      permission-id=allowedValue>
      <name>string</name>
      <domain-name>string</domain-name>
      <url-path>string</url-path>
      <date-begin>datetime</date-begin>
    </event>
  </my-events>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>my-events</td>
<td></td>
<td>Container</td>
<td>The entire list of events the user is or has been registered for.</td>
</tr>
<tr>
<td>event</td>
<td></td>
<td>Container</td>
<td>Information about one event.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of the event.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The type of the object. For an event, always event.</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Allowed value</td>
<td>An icon identifying the object. For an event, always event.</td>
</tr>
<tr>
<td>permission-id</td>
<td></td>
<td>Allowed value</td>
<td>The permission the user has for the event (see permission-id for values).</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the event.</td>
</tr>
<tr>
<td>domain-name</td>
<td></td>
<td>String</td>
<td>The domain name of the Acrobat Connect Pro server, which comes after http://(or https://) and before the unique event name in the event URL.</td>
</tr>
<tr>
<td>url-path</td>
<td></td>
<td>String</td>
<td>The unique event name, which comes after the domain name in the event URL.</td>
</tr>
<tr>
<td>date-begin</td>
<td></td>
<td>Date</td>
<td>The date the event begins, in ISO 8601 format.</td>
</tr>
<tr>
<td>date-end</td>
<td></td>
<td>Date</td>
<td>The date the event ends, in ISO 8601 format.</td>
</tr>
<tr>
<td>expired</td>
<td></td>
<td>Boolean</td>
<td>A value indicating whether the event has ended. If the event is currently underway, the value is false.</td>
</tr>
<tr>
<td>duration</td>
<td></td>
<td>Time</td>
<td>The amount of time the event is scheduled to last. Uses the time portion of an ISO 8601 date format.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=report-my-events

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <my-events>
    <event sco-id="2006334107" type="event" icon="event"
      permission-id="host">
      <name>Meet the Famous Author</name>
      <domain-name>example.com</domain-name>
      <url-path>/author/</url-path>
      <date-begin>2006-05-12T18:00:00.000-07:00</date-begin>
  </event>
  ...
</my-events>
</results>
report-my-meetings

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about all Acrobat Connect Pro meetings for which the user is a host, invited participant, or registered guest. The meeting can be scheduled in the past, present, or future.

Request URL
http://server_name/api/xml
?action=report-my-meetings
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <my-meetings>
    <meeting sco-id=integer type="meeting" icon="meeting"
      permission-id=allowedValue active-participants=integer>
      <name>string</name>
      <description>string</description>
      <domain-name>domain</domain-name>
      <url-path>url</url-path>
      <date-begin>date</date-begin>
      <date-end>date</date-end>
      <expired>boolean</expired>
      <duration>time</duration>
    </meeting>...
  </my-meetings>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
<td></td>
</tr>
<tr>
<td>my-meetings</td>
<td>Container</td>
<td>Information about all meetings the user is, or has been, invited to.</td>
<td></td>
</tr>
<tr>
<td>meeting</td>
<td>Container</td>
<td>Details about one of the user's meetings.</td>
<td></td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the meeting.</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>The type of the object returned (for this call, always meeting).</td>
<td></td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td>The icon that visually identifies the meeting in Acrobat Connect Pro Central (for this call, always meeting).</td>
<td></td>
</tr>
<tr>
<td>permission-id</td>
<td>Allowed value</td>
<td>The level of permission the user has to the meeting (see permission-id for values).</td>
<td></td>
</tr>
<tr>
<td>active-participants</td>
<td>Integer</td>
<td>The number of participants the meeting currently has, including hosts and presenters.</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the meeting.</td>
<td></td>
</tr>
<tr>
<td>domain-name</td>
<td>String</td>
<td>The domain name portion of the URL to the meeting room.</td>
<td></td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td>The part of the meeting room URL that identifies the meeting and comes after the domain name.</td>
<td></td>
</tr>
<tr>
<td>date-begin</td>
<td>Datetime</td>
<td>The date and time the meeting begins (or has begun).</td>
<td></td>
</tr>
<tr>
<td>date-end</td>
<td>Datetime</td>
<td>The date and time the meeting ends (or has ended).</td>
<td></td>
</tr>
<tr>
<td>expired</td>
<td>Boolean</td>
<td>Whether the meeting has ended (true if it has, false if it has not).</td>
<td></td>
</tr>
<tr>
<td>duration</td>
<td>Time</td>
<td>The actual length of time of the meeting. This may be longer or shorter than the time the meeting was scheduled for.</td>
<td></td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=report-my-meetings

Sample response
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <my-meetings>
    <meeting sco-id="2006334033" type="meeting" icon="meeting"
      permission-id="host" active-participants="0">
      <name>How to Write a Novel</name>
      <domain-name>example.com</domain-name>
      <url-path>/novel/</url-path>
      <date-begin>2006-05-11T11:30:00.000-07:00</date-begin>
      <date-end>2006-05-11T12:30:00.000-07:00</date-end>
      <expired>true</expired>
      <duration>01:00:00.000</duration>
    </meeting>
  </my-meetings>
</results>
```
<meeting sco-id="2006743452" type="meeting" icon="meeting"
  permission-id="host" active-participants="0">
  <name>Intro to Film</name>
  <domain-name>example.com</domain-name>
  <url-path>/film/</url-path>
  <date-begin>2006-06-09T14:00:00.000-07:00</date-begin>
  <date-end>2006-06-09T20:00:00.000-07:00</date-end>
  <expired>true</expired>
  <duration>06:00:00.000</duration>
</meeting>
</my-meetings>
</results>

report-my-training

Availability
Connect Enterprise Web Services 6

Description
Returns a list of all courses and curriculums a user or group is enrolled in. If you do not use a principal-id, the list is for the current user. If you add a principal-id, the list is for the principal you specify.

The response contains a list of row elements. In the list, courses have the attributes type=content and icon=course, while curriculums have type=curriculum and icon=curriculum.

Request URL
http://server_name/api/xml
?action=report-my-training
&principal-id=integer
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>N</td>
<td>The unique ID of a user or group whose courses and curriculums you want to list. If you do not specify a value, the response is for the current user.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-my-training>
<row sco-id=integer type=allowedValue icon=allowedValue max-retries=integer permission-id=allowedValue transcript-id=integer attempts=integer>
    <name>string</name>
    <url>string</url>
    <date-created>datetime</date-created>
    <date-modified>datetime</date-modified>
    <date-begin>datetime</date-begin>
    <url-path>string</url-path>
    <expired>boolean</expired>
    <completed>boolean</completed>
</row>

</report-my-training>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-my-training</td>
<td></td>
<td>Container</td>
<td>The entire list of courses and curriculums the user is enrolled in.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one course or curriculum the user is enrolled in.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of the course or curriculum.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The type of the object (see type for allowed values).</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Allowed value</td>
<td>The icon that identifies the object in Acrobat Connect Pro Central (see icon for allowed values). If type is content, the icon value describes the content.</td>
</tr>
<tr>
<td>max-retries</td>
<td></td>
<td>Integer</td>
<td>The allowed number of attempts that the course can be retaken.</td>
</tr>
<tr>
<td>permission-id</td>
<td></td>
<td>Allowed value</td>
<td>The permission the principal has on the object (see permission-id for allowed values).</td>
</tr>
<tr>
<td>transcript-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the course transcript.</td>
</tr>
<tr>
<td>attempts</td>
<td></td>
<td>Integer</td>
<td>The number of times the user has tried to complete the course.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the course or curriculum.</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>String</td>
<td>The course or curriculum description.</td>
</tr>
<tr>
<td>url</td>
<td></td>
<td>String</td>
<td>The part of the URL to the course or curriculum that includes the domain name and unique name, without http:// or https://.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date and time the course or curriculum was created.</td>
</tr>
<tr>
<td>date-modified</td>
<td></td>
<td>Datetime</td>
<td>The date and time the course or curriculum was last modified.</td>
</tr>
<tr>
<td>date-begin</td>
<td></td>
<td>Datetime</td>
<td>The start date and time of the course or curriculum, either past or future.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=report-my-training
&principal-id=2006258745

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-my-training>
    <row sco-id="2006298431" type="content" icon="course" max-retries="" permission-id="view" transcript-id="" attempts="0">
      <name>Intro to Psychology</name>
      <url>example.com/psychology/</url>
      <date-created>2006-05-03T10:21:46.810-07:00</date-created>
      <date-modified>2006-05-03T10:22:30.803-07:00</date-modified>
      <date-begin>2006-05-03T10:15:00.000-07:00</date-begin>
      <url-path>/psychology/</url-path>
      <expired>false</expired>
      <completed>true</completed>
    </row>
    <row sco-id="2006745669" type="curriculum" icon="curriculum" permission-id="view">
      <name>A Day in the Life</name>
      <url>example.com/day/</url>
      <date-created>2006-05-03T10:21:46.810-07:00</date-created>
      <date-modified>2006-05-03T10:22:30.803-07:00</date-modified>
      <date-begin>2006-05-03T10:15:00.000-07:00</date-begin>
      <url-path>/day/</url-path>
      <expired>false</expired>
      <tr-status>not-attempted</tr-status>
    </row>
  </report-my-training>
</results>

report-quiz-interactions

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6
Description
Provides information about all the interactions users have had with a certain quiz. An interaction identifies all answers one user makes to one quiz question. If a user answers the same question more than once, all answers are part of the same interaction and have the same interaction-id.
This report provides information about every answer that any user has ever given to questions on a quiz. You can filter the response to make it more meaningful, using any allowed filters. For example, you can request all answers a certain user has given:
https://example.com/api/xml?action=report-quiz-interactions
   &sco-id=2006334909&filter-like-name=Joy%20Smith
Or, you can request only a certain user’s answers to a specific question:
https://example.com/api/xml?action=report-quiz-interactions
   &sco-id=2006334909&filter-name=Joy%20Smith
   &filter-like-description=What%20is%20the%20capital%20of%20California

Request URL
http://server_name/api/xml
   ?action=report-quiz-interactions
   &sco-id=integer
   &filter-definition=value
   &sort-definition=value
   &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation or course that contains a quiz.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<results>
   <status code=allowedValue />
   <report-quiz-interactions>
      <row display-seq=integer transcript-id=integer interaction-id=integer
         sco-id=integer score=integer>
         <name>string</name>
         <sco-name>string</sco-name>
         <date-created>datetime</date-created>
         <description>string</description>
         <response>integer</response>
      </row>
   ...
</report-quiz-interactions>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-interactions</td>
<td></td>
<td>Container</td>
<td>Information about all interactions all users have had with the quiz.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one user, one quiz question, and one answer. Multiple row elements can be part of the same interaction.</td>
</tr>
<tr>
<td>display-seq</td>
<td></td>
<td>Integer</td>
<td>The sequence number of this question in the quiz.</td>
</tr>
<tr>
<td>transcript-id</td>
<td></td>
<td>Integer</td>
<td>The ID of one user's attempt to take a quiz, with one user, one attempt at a quiz, and multiple questions and answers. Each time the user takes the quiz, the transcript-id changes.</td>
</tr>
<tr>
<td>interaction-id</td>
<td></td>
<td>Integer</td>
<td>The ID of all answers one user makes to one quiz question.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of a presentation or course that contains the quiz.</td>
</tr>
<tr>
<td>score</td>
<td></td>
<td>Integer</td>
<td>The user's score for this question.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the user.</td>
</tr>
<tr>
<td>sco-name</td>
<td></td>
<td>Integer</td>
<td>The name of the presentation or course that contains the quiz.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date the presentation or course was created.</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>String</td>
<td>The quiz question the user answered.</td>
</tr>
<tr>
<td>response</td>
<td></td>
<td>String</td>
<td>The response the user gave.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-quiz-interactions
&<sco-id>2006334909</sco-id>&<filter-name>Joy Smith</filter-name>&<filter-like-description>governor</filter-like-description>

Sample request

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-interactions>
    <row display-seq="2" transcript-id="2006335803"
      interaction-id="2006334914" sco-id="2006334909" score="10">
      <name>Joy Smith</name>
      <sco-name>California Quiz</sco-name>
      <date-created>2006-05-11T15:50:23.643-07:00</date-created>
      <description>
        The governor of California is a former actor.
      </description>
      <response>true</response>
    </row>
  </report-quiz-interactions>
</results>
**report-quiz-question-answer-distribution**

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Returns information about the number of users who chose a specific answer to a quiz question. The combination of one quiz question and all of one user’s answers to it is called an interaction. If the user answers the question more than once, all answers are part of the same interaction and have the same interaction-id.

Use report-quiz-interactions to determine an interaction-id to specify in the request. The interaction-id does not correspond to the question number in the quiz (for example, question 1, question 2, and so on).

**Request URL**
http://server_name/api/xml

`?action=report-quiz-question-answer-distribution`  
`&interaction-id=integer`  
`&sco-id=integer`  
`&filter-definition=value`  
`&sort-definition=value`  
`&session=BreezeSessionCookieValue`

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interaction-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID that describes all of one user’s responses to one quiz question.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation or course that contains a quiz.</td>
</tr>
</tbody>
</table>
Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-quiz-question-answer-distribution>
    <row display-seq=integer interaction-id=integer score=integer asset-id=integer num-selected=integer>
      <response>string</response>
    </row>
    ...
  </report-quiz-question-answer-distribution>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-question-answer-distribution</td>
<td></td>
<td>Container</td>
<td>The list of questions and answers.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one user, one question, and one answer.</td>
</tr>
<tr>
<td></td>
<td>display-seq</td>
<td>Integer</td>
<td>The sequence number of the question in the quiz.</td>
</tr>
<tr>
<td></td>
<td>interaction-id</td>
<td>Integer</td>
<td>The ID of all of one user’s responses to one quiz question. If the user answers the question multiple times, all answers have the same interaction-id.</td>
</tr>
<tr>
<td></td>
<td>score</td>
<td>Integer</td>
<td>The score the user earned on the question.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml
?action=report-quiz-question-answer-distribution&sco-id=2006334909

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-question-answer-distribution>
    <row display-seq="1" interaction-id="2006334913" score="0"
      asset-id="2006334911" num-selected="1">
      <response>san francisco</response>
    </row>
    <row display-seq="1" interaction-id="2006334913" score="10"
      asset-id="2006334911" num-selected="2">
      <response>Sacramento</response>
    </row>
    <row display-seq="2" interaction-id="2006334914" score="0"
      asset-id="2006334911" num-selected="1">
      <response>false</response>
    </row>
    ...
  </report-quiz-question-answer-distribution>
</results>

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asset-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the version of the quiz in which the user answered the question.</td>
</tr>
<tr>
<td>num-selected</td>
<td>Integer</td>
<td></td>
<td>In multiple choice or true/false quiz questions, the sequence number of the answer selected.</td>
</tr>
<tr>
<td>response</td>
<td>String</td>
<td></td>
<td>The response the user gave to the question.</td>
</tr>
</tbody>
</table>
report-quiz-question-distribution

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about the number of correct and incorrect answers to the questions on a quiz. This call can help you determine how a group responded to a quiz question overall.

Because this call returns information about all the questions on a quiz, you may want to filter the results for a specific question or group of questions.

Request URL
http://server_name/api/xml
  ?action=report-quiz-question-distribution
  &sco-id=integer
  &filter-definition=value
  &sort-definition=value
  &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation that contains a quiz.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-quiz-question-distribution>
    <row display-seq=integer interaction-id=integer num-correct=integer num-incorrect=integer total-responses=integer percentage-correct=integer score=integer>
      <name>string</name>
      <description>string</description>
    </row>
    ...
  </report-quiz-question-distribution>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-question-distribution</td>
<td></td>
<td>Container</td>
<td>Information about all the questions on a quiz.</td>
</tr>
<tr>
<td>row</td>
<td>display-seq</td>
<td>Integer</td>
<td>The sequence in the quiz in which this question falls.</td>
</tr>
<tr>
<td></td>
<td>interaction-id</td>
<td>Integer</td>
<td>The ID of the quiz question.</td>
</tr>
<tr>
<td></td>
<td>num-correct</td>
<td>Integer</td>
<td>The number of correct answers to this question.</td>
</tr>
<tr>
<td></td>
<td>num-incorrect</td>
<td>Integer</td>
<td>The number of incorrect answers to this question.</td>
</tr>
<tr>
<td></td>
<td>total-responses</td>
<td>Integer</td>
<td>The total number of responses to this question.</td>
</tr>
<tr>
<td></td>
<td>percentage-correct</td>
<td>Integer</td>
<td>The percentage of the total responses that were correct.</td>
</tr>
<tr>
<td></td>
<td>score</td>
<td>Integer</td>
<td>The score assigned to the quiz question.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the quiz question, defined when the question was created in Quiz Manager.</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>String</td>
<td>The definition of the quiz question, defined when the question was created in Quiz Manager.</td>
</tr>
</tbody>
</table>

Sample request


Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-question-distribution>
    <row display-seq="1" interaction-id="2006334913" num-correct="2"
      num-incorrect="1" total-responses="3" percentage-correct="66"
      score="10">
      <name>The capital of California is</name>
      <description>The capital of California is</description>
    </row>
  </report-quiz-question-distribution>
</results>
```
report-quiz-question-response

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides a list of answers that users have given to questions on a quiz.
Without filtering, this action returns all answers from any user to any question on the quiz. However, you can filter the response for a specific user, interaction, or answer (see the filter syntax at filter-definition).
An interaction is a combination of one user and one question. If the user answers the same question more than once, all answers are part of the same interaction and have the same interaction-id.

Request URL
http://server_name/api/xml
?action=report-quiz-question-response
&sco-id=integer
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation that contains a quiz.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-quiz-question-response>
    <row principal-id=integer interaction-id=string>
      <user-name>string</user-name>
      <response>string</response>
      <date-created>datetime</date-created>
    </row>
  </report-quiz-question-response>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-question-response</td>
<td>Container</td>
<td></td>
<td>Information about all responses to all questions on the quiz.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one response.</td>
</tr>
<tr>
<td>principal-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the user who answered the quiz question.</td>
</tr>
<tr>
<td>interaction-id</td>
<td></td>
<td>Integer</td>
<td>The ID of one response to one question.</td>
</tr>
<tr>
<td>user-name</td>
<td></td>
<td>String</td>
<td>The name of the user as registered on the server.</td>
</tr>
<tr>
<td>response</td>
<td></td>
<td>String</td>
<td>The user's response to the question, including a word or phrase, true, false, or a letter choice.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date and time the user responded.</td>
</tr>
</tbody>
</table>

Sample request


Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-question-response>
    <row principal-id="2006258745" interaction-id="2006334913">
      <user-name>Joy Smith</user-name>
      <response>Sacramento</response>
      <date-created>2006-05-11T15:50:23.643-07:00</date-created>
    </row>
    <row principal-id="2006258745" interaction-id="2006334913">
      <user-name>Joy Smith</user-name>
      <response>san francisco</response>
      <date-created>2006-05-11T17:32:53.970-07:00</date-created>
    </row>
    <row principal-id="2006258745" interaction-id="2006334913">
      <response>Sacramento</response>
      <date-created>2006-05-12T11:55:24.940-07:00</date-created>
    </row>
  </report-quiz-question-response>
</results>
```

report-quiz-summary

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6
Description
Provides a summary of data about a quiz, including the number of times the quiz has been taken; average, high, and low scores; and other information.

Request URL
http://server_name/api/xml
   ?action=report-quiz-summary
   &sco-id=integer
   &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation that contains a quiz.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code=allowedValue />
   <report-quiz-summary>
      <row num-questions=integer average-score=integer low-score=integer
           high-score=integer numtaken=integer numdistincttaken=integer
           maxpossiblescore=integer asset-id=integer />
   </report-quiz-summary>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-summary</td>
<td></td>
<td>Container</td>
<td>Contains information about the quiz.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Empty, with attributes</td>
<td>Summary information about the quiz. Can return more than one row element if the maxpossiblescore is different for different transcripts.</td>
</tr>
<tr>
<td>num-questions</td>
<td>Integer</td>
<td></td>
<td>The number of questions on the quiz.</td>
</tr>
<tr>
<td>average-score</td>
<td>Integer</td>
<td></td>
<td>The average score, across all users who have taken the quiz.</td>
</tr>
<tr>
<td>low-score</td>
<td>Integer</td>
<td></td>
<td>The lowest score a user has received on the quiz.</td>
</tr>
<tr>
<td>high-score</td>
<td>Integer</td>
<td></td>
<td>The highest score a user has received on the quiz.</td>
</tr>
<tr>
<td>numtaken</td>
<td>Integer</td>
<td></td>
<td>The total number of times the quiz has been taken.</td>
</tr>
</tbody>
</table>
Sample request

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-summary>
    <row num-questions="2" average-score="0" low-score="0" high-score="0"
      numtaken="1" numdistincttaken="1" maxpossiblescore="0"
      asset-id="2006334911" />
    <row num-questions="2" average-score="13" low-score="0" high-score="20"
      numtaken="3" numdistincttaken="3" maxpossiblescore="20"
      asset-id="2006334911" />
  </report-quiz-summary>
</results>

report-quiz-takers

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about all users who have taken a quiz in a training. Use a sco-id to identify the quiz.
To reduce the volume of the response, use any allowed filter or pass a type parameter to return information about just one type of SCO (courses, presentations, or meetings).

Request URL
http://server_name/api/xml?
?action=report-quiz-takers
&sco-id=integer
&principal-id=integer
&type=allowedValue
&filter-definition=value
&sort-definition=value
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation or course that contains a quiz.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of a principal for whom you want quiz results.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed</td>
<td>N</td>
<td>The type of content for which you want results. Allowed values are course, presentation, and meeting.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-quiz-takers>
    <row transcript-id=integer sco-id=integer principal-id=integer status=allowedValue score=integer asset-id=integer permission-id=allowedValue attempts=integer time-taken=integer certificate=integer answered-survey=boolean version=integer>
      <name>string</name>
      <login>string</login>
      <date-created>datetime</date-created>
      <principal-name>string</principal-name>
      <override>boolean</override>
    </row>
  </report-quiz-takers>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quiz-takers</td>
<td></td>
<td>Container</td>
<td>Information about all users who have taken the quiz.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one user who has taken the quiz.</td>
</tr>
<tr>
<td>transcript-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the transcript on which the user's quiz score is recorded.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the presentation, course, or meeting that has the quiz.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the user who took the quiz.</td>
</tr>
<tr>
<td>status</td>
<td>Allowed</td>
<td></td>
<td>Whether the user passed or failed the most recent attempt at the quiz. Allowed values are user-passed and user-failed.</td>
</tr>
</tbody>
</table>
### Sample request

```
https://example.com/api/xml?action=report-quiz-takers&sco-id=2006334909
```

### Sample response

```
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-quiz-takers>
    <row transcript-id="2006337854" sco-id="2006334909"
      principal-id="2006258745" status="incomplete" score="0"
      max-score="20" asset-id="2006334911" permission-id=""
      attempts="4" time-taken="12593" certificate="" answered-survey="1"
      version="1">
      <name>California State Quiz</name>
      <login>joy@acme.com</login>
      <date-created>2006-05-16T11:14:47.000-07:00</date-created>
      <principal-name>Joy Smith</principal-name>
      <override>false</override>
    </row>
  </report-quiz-takers>
</results>
```

### report-quotas

**Availability**

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6
Description

Returns information about the quotas that apply to your Acrobat Connect Pro license or Acrobat Connect Pro hosted account. Acrobat Connect Pro enforces various quotas, for example, the number of concurrent users in training, the number of downloads, the number of authors, and so on.

Although your server license determines certain quotas, you can scale your license beyond your limit. In the response from report-quotas, the soft-limit is the number defined by your license. The soft-limit is the same as the limit, unless you purchase a Burst Pack for meetings, which allows additional participants to join past the limit, on an overage basis.

Request URL

http://server_name/api/xml
?action=report-quotas
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-quotas>
    <quota acl-id=integer quota-id=string used=integer
      limit=allowedValue soft-limit=integer>
      <date-begin>datetime</date-begin>
      <date-end>datetime</date-end>
    </quota>
  </report-quotas>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-quotas</td>
<td></td>
<td>Container</td>
<td>Information about all of the quotas set for the account.</td>
</tr>
<tr>
<td>quota</td>
<td></td>
<td>Container</td>
<td>Information about one quota.</td>
</tr>
<tr>
<td>acl-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the account on which the quota is defined.</td>
</tr>
<tr>
<td>quota-id</td>
<td></td>
<td>Allowed value</td>
<td>The name of the quota defined by the server, ending in -quota.</td>
</tr>
<tr>
<td>used</td>
<td></td>
<td>Integer</td>
<td>The number of uses that count toward this quota.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=report-quotas

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="ok" />
    <report-quotas>
        <quota acl-id="624520" quota-id="download-quota" used="1" limit="unlimited" soft-limit="1000000000">
            <date-begin>2004-03-09T09:45:41.047-08:00</date-begin>
            <date-end>3000-01-01T00:00:00.000-08:00</date-end>
        </quota>
        <quota acl-id="624520" quota-id="bandwidth-quota" used="12802" limit="unlimited" soft-limit="1000000000">
            <date-begin>2006-05-31T17:00:00.943-07:00</date-begin>
            <date-end>2006-06-30T17:00:00.943-07:00</date-end>
        </quota>
    </report-quotas>
</results>

report-sco-slides

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Returns information about the slides in a presentation. The information includes how many times, and how recently, each slide has been viewed.

Request URL
http://server_name/api/xml?action=report-sco-slides
&sco-id=integer
&asset-id=integer
&sort-definition=value
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a presentation.</td>
</tr>
<tr>
<td>asset-id</td>
<td>Integer</td>
<td>N</td>
<td>The version number of a presentation, incremented each time a presentation is uploaded.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-sco-slides>
    <row slide=integer name=integer asset-id=integer views=integer>
      <date-created>datetime</date-created>
    </row>
    ...
  </report-sco-slides>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-sco-slides</td>
<td></td>
<td>Container</td>
<td>Information about all of the slides in a presentation, indicating how many times and how recently a slide has been viewed.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one slide in the presentation.</td>
</tr>
<tr>
<td>slide</td>
<td></td>
<td>Integer</td>
<td>The number of the slide within the presentation.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>Integer</td>
<td>The name of the slide in the presentation.</td>
</tr>
<tr>
<td>asset-id</td>
<td></td>
<td>Integer</td>
<td>The version number of the presentation. Each time a presentation is published, it has a new asset-id.</td>
</tr>
<tr>
<td>views</td>
<td></td>
<td>Integer</td>
<td>The number of times the slide has been viewed.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date and time the slide was last viewed.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-sco-slides&sco-id=2006334909

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
```
<status code="ok" />
<report-sco-slides>
  <row slide="1" name="1" views="4">
    <date-created>2006-05-16T11:14:54.453-07:00</date-created>
  </row>
  <row slide="2" name="2" views="4">
    <date-created>2006-05-16T11:14:59.593-07:00</date-created>
  </row>
  <row slide="3" name="3" views="3">
    <date-created>2006-05-12T11:55:52.330-07:00</date-created>
  </row>
  <row slide="4" name="4" views="3">
    <date-created>2006-05-12T11:55:55.487-07:00</date-created>
  </row>
  <row slide="5" name="5" views="3">
    <date-created>2006-05-12T11:56:00.233-07:00</date-created>
  </row>
</report-sco-slides>
</results>

See also
report-sco-views

report-sco-views

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Indicates how many times, and how recently, a SCO was viewed.

Request URL
http://server_name/api/xml
?action=report-sco-views
&sco-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a SCO to check for views.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-sco-views sco-id=integer type=allowedValue is-folder=boolean
                   views=integer>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td>Allowed</td>
<td>value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-sco-views</td>
<td>Container</td>
<td></td>
<td>Information about how many times, and how recently, the presentation was viewed.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the presentation.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed</td>
<td>value</td>
<td>The type of content object (SCO). Allowed values are content, curriculum, event, folder, link, meeting, and tree.</td>
</tr>
<tr>
<td>is-folder</td>
<td>Boolean</td>
<td></td>
<td>A value indicating whether the SCO is a folder (if 1) or another type of object (if 0).</td>
</tr>
<tr>
<td>views</td>
<td>Integer</td>
<td></td>
<td>The number of times users have viewed the SCO.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
<td>The name of the SCO.</td>
</tr>
<tr>
<td>last-viewed-date</td>
<td>Datetime</td>
<td></td>
<td>The date and time the SCO was last viewed.</td>
</tr>
</tbody>
</table>

Sample request


Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-sco-views sco-id="2006334909" type="content" is-folder="0"
    views="3">
    <name>Quiz on California</name>
    <last-viewed-date>2006-05-12T11:55:24.940-07:00</last-viewed-date>
  </report-sco-views>
</results>
```

report-user-trainings-taken

Availability

Connect Enterprise Web Services 6

Description

Returns a list of all courses and curriculums a user has taken, whether or not the user has completed the training. Each course or curriculum is returned in a separate row element and has the most recent transcript of the user's scores.
Request URL

http://server_name/api/xml
?action=report-user-trainings-taken
&principal-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a user for whom you want a list of trainings.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter or sort the response on any element or attribute it contains.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-user-trainings-taken>
    <row transcript-id=integer max-retries=integer sco-id=integer
    type=allowedValue icon=allowedValue status=allowedValue
    certificate=integer score=integer permission-id=allowedValue
    attempts=allowedValue>
      <name>string</name>
      <description>string</description>
      <url-path>string</url-path>
      <date-taken>datetime</date-taken>
      <from-curriculum>boolean</from-curriculum>
    </row>
    ...
  </report-user-trainings-taken>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td>All results the action returns.</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>The type of the SCO. Allowed values are user-content, content, and my-content.</td>
<td></td>
</tr>
<tr>
<td>report-user-trainings-taken</td>
<td>Container</td>
<td>A list of all trainings the user has attempted.</td>
<td></td>
</tr>
<tr>
<td>row</td>
<td>Container</td>
<td>Information about one course or curriculum the user has taken, whether or not passed.</td>
<td></td>
</tr>
<tr>
<td>transcript-id</td>
<td>Integer</td>
<td>The ID of the record of the user's most recent score on this training.</td>
<td></td>
</tr>
<tr>
<td>max-retries</td>
<td>Integer</td>
<td>The maximum number of times the user can repeat the training.</td>
<td></td>
</tr>
</tbody>
</table>
Sample request

https://example.com/api/xml?action=report-user-trainings-taken
&principal-id=2006258745&principal-id=4797406

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-user-trainings-taken>
    <row transcript-id="2006745722" max-retries="" sco-id="2006745673"
      type="content" icon="course" status="user-passed"
      certificate="2006745722" score="0" permission-id=""
      attempts="1">
      <name>All About Web Communities</name>
      <description>test</description>
      <url-path>/p33096345/</url-path>
      <date-taken>2006-06-12T15:06:02.947-07:00</date-taken>
    </row>
  </report-user-trainings-taken>
</results>
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the training SCO.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed</td>
<td></td>
<td>The type of the training SCO (see allowed values at type).</td>
</tr>
<tr>
<td>icon</td>
<td>Allowed</td>
<td></td>
<td>The type of icon that identifies the course or curriculum in Connect Pro Central. Provides information about the course or curriculum in addition to its type (see allowed values at icon).</td>
</tr>
<tr>
<td>status</td>
<td>Allowed</td>
<td></td>
<td>The status of the user's work with the SCO. Allowed values for a course or presentation are user-passed, user-failed, completed, incomplete, not-attempted, and review. A curriculum or folder can only be completed or incomplete.</td>
</tr>
<tr>
<td>certificate</td>
<td>Integer</td>
<td></td>
<td>The ID of the record that shows the user passed or completed the training.</td>
</tr>
<tr>
<td>score</td>
<td>Integer</td>
<td></td>
<td>The score the user earned on the most recent attempt at the training.</td>
</tr>
<tr>
<td>permission-id</td>
<td>Allowed</td>
<td></td>
<td>The permission the user has been assigned to access the course or curriculum (see permission-id for allowed values).</td>
</tr>
<tr>
<td>attempts</td>
<td>Integer</td>
<td></td>
<td>The number of attempts the user has made at this training.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
<td>The name of the training SCO.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td></td>
<td>The description of the training SCO.</td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td></td>
<td>The unique identifier of the SCO that appears in its URL after the domain name.</td>
</tr>
<tr>
<td>date-taken</td>
<td>Datetime</td>
<td></td>
<td>The date the user interacted with the training SCO (viewed a presentation, took a quiz, and so on).</td>
</tr>
<tr>
<td>from-curriculum</td>
<td>Boolean</td>
<td></td>
<td>A value indicating whether this course was taken as part of a curriculum.</td>
</tr>
</tbody>
</table>
report-user-training-transcripts

Availability
Connect Enterprise Web Services 6

Description
Returns a list of transcripts for trainings a user has taken. A transcript is the record of one score a user obtained from one attempt at taking one training. A training can be a course, curriculum, meeting, or event.

The response can include more than one transcript for a training SCO, if the user has attempted the training more than once. A user can fail a training the first time and then pass on the second attempt. Each attempt has its own transcript, and both transcripts are included in the report.

Request URL
http://server_name/api/xml
?action=report-user-training-transcripts
&amp;principal-id=integer
&amp;session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of a user whose transcripts you want.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
You can filter or sort the response on any element or attribute it contains.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <report-user-training-transcripts>
    <row transcript-id=integer sco-id=integer
           principal-id=integer status=allowedValue score=integer
           max-score=integer certificate=integer type=allowedValue
           icon=allowedValue>
      <name>string</name>
      <url-path>string</url-path>
      <login>string</login>
      <date-taken>datetime</date-taken>
      <principal-name>string</principal-name>
      <sco-tag>string</sco-tag>
    </row>
    ...
  </report-user-training-transcripts>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td></td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>report-user-training-transcripts</td>
<td>Container</td>
<td></td>
<td>Information about all transcripts for the specified user.</td>
</tr>
<tr>
<td>row</td>
<td></td>
<td>Container</td>
<td>Information about one attempt the user made on the training.</td>
</tr>
<tr>
<td>transcript-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the transcript on which the user's score is recorded. A distinct transcript exists for each of the user's attempts at taking the SCO.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the SCO, which can be a meeting, content, course, curriculum, event, or seminar.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the user.</td>
</tr>
<tr>
<td>status</td>
<td>Allowed value</td>
<td></td>
<td>The status of the user's work with the SCO (see status attribute for allowed values).</td>
</tr>
<tr>
<td>score</td>
<td>Integer</td>
<td></td>
<td>The score the user earned on the SCO. If the SCO does not have a score, as with a meeting, the value of score is 0.</td>
</tr>
<tr>
<td>max-score</td>
<td>Integer</td>
<td></td>
<td>The maximum score possible on the course or curriculum.</td>
</tr>
<tr>
<td>certificate</td>
<td>Integer</td>
<td></td>
<td>The ID of the record of the courses and curriculums the user has passed or completed.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td></td>
<td>The type of the SCO. Allowed values are user-content, content, and my-content.</td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td></td>
<td>The name of the icon that identifies the course or curriculum in Acrobat Connect Pro Central.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
<td>The name of the course or curriculum.</td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td></td>
<td>The unique identifier of the course or curriculum that appears in its URL after the domain name.</td>
</tr>
<tr>
<td>login</td>
<td>String</td>
<td></td>
<td>The user's login ID on Acrobat Connect Pro Server.</td>
</tr>
<tr>
<td>date-taken</td>
<td>Datetime</td>
<td></td>
<td>The date the user interacted with the course or curriculum (viewed a presentation, took a quiz, and so on).</td>
</tr>
<tr>
<td>principal-name</td>
<td>String</td>
<td></td>
<td>The name of the user interacting with the SCO.</td>
</tr>
<tr>
<td>sco-tag</td>
<td>String</td>
<td></td>
<td>A descriptive label for the SCO, in addition to the name (for example, a short course name).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=report-user-training-transcripts
&principal-id=2006258745
Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-user-training-transcripts>
    <row transcript-id="2006905612" sco-id="2006298431"
      principal-id="2006258745" status="review" score="0" max-score=""
      certificate="" type="content" icon="course">
      <name>Test Course</name>
      <url-path>/test/</url-path>
      <login>joy@acme.com</login>
      <date-taken>2006-06-30T15:23:55.070-07:00</date-taken>
      <principal-name>Joy Smith</principal-name>
    </row>
    <row transcript-id="2007016805" sco-id="2006298431"
      principal-id="2006258745" status="review" score="0" max-score=""
      certificate="" type="content" icon="course">
      <name>Test Course</name>
      <url-path>/test/</url-path>
      <login>joy@acme.com</login>
      <date-taken>2006-07-14T16:55:28.440-07:00</date-taken>
      <principal-name>Joy Smith</principal-name>
    </row>
  </report-user-training-transcripts>
</results>
```

**sco-contents**

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Returns a list of SCOs within another SCO. The enclosing SCO can be a folder, meeting, or curriculum.

In general, the contained SCOs can be of any type—meetings, courses, curriculums, content, events, folders, trees, or links (see the list in `type`). However, the type of the contained SCO needs to be valid for the enclosing SCO. For example, courses are contained within curriculums, and meeting content is contained within meetings.

Because folders are SCOs, the returned list includes SCOs and subfolders at the next hierarchical level, but not the contents of the subfolders. To include the subfolder contents, call `sco-expanded-contents`.

**Request URL**
http://server_name/api/xml
?action=sco-contents
&sco-id=integer
&filter-definition=value
&sort-definition=value
&session=value
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a folder for which you want to list contents. You can get the sco-id by calling sco.shortcuts.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

You can filter the response on any element or attribute, with these exceptions:

- You cannot filter on duration.
- If you use filter-date-begin, filter-date-end, or filter-date-modified, specify a time without a time zone, for example:
  ```
  filter-date-modified=2005-01-05T10:44:03
  ```

You can use filter-gt or filter-lt with a date field and a full date, including the time zone.

You can sort the response on any element or attribute.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <scos>
    <sco sco-id=integer source-sco-id=integer folder-id=integer
          type=allowedValue icon=allowedValue display-seq=integer
          is-folder=boolean byte-count=integer ref-count=integer>
      <name>string</name>
      <url-path>string</url-path>
      <description>string</description>
      <date-begin>string</date-begin>
      <date-modified>datetime</date-modified>
      <date-end>string</date-end>
      <sco-tag>string</sco-tag>
    </sco>
  </scos>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>scos</td>
<td></td>
<td>Container</td>
<td>The list of objects within the folder.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>One object within the folder.</td>
</tr>
<tr>
<td></td>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of one object within the folder.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=sco-contents&sco-id=2006258748

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <scos>
    <sco sco-id="2007035246" source-sco-id="2006334909"
      folder-id="2006258748" type="content" icon="course"
      display-seq="0" is-folder="0">
      <name>Java 101</name>
      <url-path>/java101/</url-path>
      <date-begin>2006-07-20T17:15:00.000-07:00</date-begin>
      <date-modified>2006-07-20T17:21:38.860-07:00</date-modified>
    </sco>
  </scos>
</results>
sco-delete

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Deletes one or more objects (SCOs).

If the sco-id you specify is for a folder, all the contents of the specified folder are deleted. To delete multiple SCOs, specify multiple sco-id parameters.

You can use a call such as sco-contents to check the ref-count of the SCO, which is the number of other SCOs that reference this SCO. If the SCO has no references, you can safely remove it, and the server reclaims the space.

If the SCO has references, removing it can cause the SCOs that reference it to stop working, or the server not to reclaim the space, or both. For example, if a course references a quiz presentation, removing the presentation might make the course stop working.

As another example, if a meeting has used a content SCO (such as a presentation or video), there is a reference from the meeting to the SCO. Deleting the content SCO does not free disk space, because the meeting still references it.

To delete a SCO, you need at least manage permission (see permission-id for details). Users who belong to the built-in authors group have manage permission on their own content folder, so they can delete content within it.

Request URL
http://server_name/api/xml
?action=sco-delete
&sco-id=integer
&session=value

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a SCO.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=sco-delete&sco-id=2007171127

Sample response
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
</results>
```

See also
sco-info, sco-move, sco-nav, sco-expanded-contents

sco-expanded-contents

Availability
Breeze 5; Connect Enterprise Web Services 6

Description
Lists all of the SCOs in a folder, including the contents of subfolders, curriculums, and any type of enclosing SCO.

Note: If you call this command on a large folder—such as the root meeting folder for a large account—the amount of data returned is very large.

If you do not use a filter, the list of SCOs is returned in the same order as it appears in Acrobat Connect Pro Central. If you use a filter or a sort, the list is returned according to the filter or sort you use.

Request URL
http://server_name/api/xml?
  ?action=sco-expanded-contents
  &sco-id=integer
  &filter-definition=value
  &sort-definition=value
  &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a folder.</td>
</tr>
</tbody>
</table>
Filters

You can filter the response on any element or attribute, with these exceptions:

- You cannot filter on `duration`.
- If you use `filter-date-begin`, `filter-date-end`, or `filter-date-modified`, specify a date in ISO 8601 format but without a time zone, for example:
  ```
  filter-date-modified=2005-01-05T10:44:03
  ```
  However, you can use `filter-gt-datefield` or `filter-lt-datefield` with a full date that includes a time zone.
- Do not use partial match filters constructed with `filter-like` (such as `filter-like-name`), as they might affect server performance.

You can sort the response on any element or attribute except `date-begin`, `date-created`, `date-modified`, and `url-path`.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <expanded-scos>
    <sco depth=integer sco-id=integer folder-id=integer type=allowedValue
         icon=allowedValue lang=allowedValue source-sco-id=integer
         display-seq=integer source-sco-type=integer>
      <name>string</name>
      <url-path>string</url-path>
      <date-created>datetime</date-created>
      <date-modified>datetime</date-modified>
    </sco>
    ... more sco elements ...
  </expanded-scos>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see <code>status</code>).</td>
</tr>
<tr>
<td>expanded-scos</td>
<td></td>
<td>Container</td>
<td>The list of all SCOs the folder contains.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Details about one SCO in the folder. This SCO can be a folder or any other type of object.</td>
</tr>
<tr>
<td>depth</td>
<td></td>
<td>Integer</td>
<td>The depth in the content tree at which this object appears, with top-level objects at 1.</td>
</tr>
</tbody>
</table>
Using Acrobat Connect Pro Web Services

Sample request
https://example.com/api/xml?action=sco-expanded-contents&sco-id=624529

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <expanded-scos>
    <sco depth="0" sco-id="624529" folder-id="624520" type="folder"
      icon="folder" lang="en" source-sco-id="" display-seq="0"
      source-sco-type="">
      <name>Shared Meetings</name>
      <url-path>/f624529/</url-path>
      <date-created>2004-03-09T09:45:41.060-08:00</date-created>
      <date-modified>2005-03-18T10:19:38.950-08:00</date-modified>
    </sco>
    <sco depth="1" sco-id="2598379" folder-id="624529" type="meeting"
      icon="meeting" lang="en" source-sco-id="-8888" display-seq="0"
      source-sco-type="3">
      <name>Monday Night Football</name>
      <url-path>/r68075204/</url-path>
      <description>Monday Night Football</description>
      <date-begin>2004-05-17T15:30:00.000-07:00</date-begin>
      <date-end>2004-05-17T15:30:00.000-07:00</date-end>
      <date-created>2004-05-17T15:30:39.733-07:00</date-created>
      <date-modified>2006-08-16T00:34:52.930-07:00</date-modified>
    </sco>
  </expanded-scos>
</results>

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the SCO. If the SCO is a folder, same as folder-id.</td>
<td></td>
</tr>
<tr>
<td>folder-id</td>
<td>Integer</td>
<td>The ID of the folder the SCO belongs to.</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>The type of this content object (see type).</td>
<td></td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td>The name of the icon that visually identifies this object.</td>
<td></td>
</tr>
<tr>
<td>lang</td>
<td>Allowed value</td>
<td>The language in which information about the SCO is displayed (see lang for values).</td>
<td></td>
</tr>
<tr>
<td>source-sco-id</td>
<td>Integer</td>
<td>The ID of a SCO from which this SCO was created, such as a meeting template or course content.</td>
<td></td>
</tr>
<tr>
<td>display-seq</td>
<td>Integer</td>
<td>The sequence in which Acrobat Connect Pro Central (or your application, if you use this value) displays a list of SCOs. Values are not necessarily unique, so multiple SCOs can have the same display-seq value. In that case, the application must define the display sequence. The default is 0.</td>
<td></td>
</tr>
<tr>
<td>source-sco-type</td>
<td>Integer</td>
<td>An integer indicating the type of the SCO from which this SCO was created.</td>
<td></td>
</tr>
</tbody>
</table>
sco-info

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides information about a SCO on Acrobat Connect Pro. The object can have any valid SCO type. See type for a list of the allowed SCO types.

The response includes the account the SCO belongs to, the dates it was created and last modified, the owner, the URL that reaches it, and other data. For some types of SCOs, the response also includes information about a template from which this SCO was created.

Request URL
http://server_name/api/xml
   ?action=sco-info
   &sco-id=integer
   &session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a SCO on the server.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code=allowedValue />
   <sco account-id=integer disabled=datetime display-seq=integer
        folder-id=integer icon=allowedValue lang=allowedValue
        max-retries=integer sco-id=integer source-sco-id=integer
        type=allowedValue version=integer>
      <date-begin>datetime</date-begin>
      <date-created>datetime</date-created>
      <date-end>datetime</date-end>
      <date-modified>datetime</date-modified>
      <description>string</description>
      <name>string</name>
      <url-path>string</url-path>
      <passing-score>integer</passing-score>
      <duration>datetime</duration>
      <section-count>integer</section-count>
   </sco>
   <source-sco>
      <source-sco account-id=integer display-seq=integer folder-id=integer
                   icon=allowedValue lang=allowedValue max-retries=integer
                   sco-id=integer source-sco-id=integer type=allowedValue
                   version=integer>
      <date-created>datetime</date-created>
      <date-modified>datetime</date-modified>
   </source-sco>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Information about the SCO.</td>
</tr>
<tr>
<td>account-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the account the SCO belongs to.</td>
</tr>
<tr>
<td>disabled</td>
<td></td>
<td>Datetime</td>
<td>An empty value if the SCO has not been disabled. If it has, the date and time it was disabled.</td>
</tr>
<tr>
<td>display-seq</td>
<td></td>
<td>Integer</td>
<td>The sequence in which Acrobat Connect Pro Central (or your application, if you use this value) displays a list of SCOs. Values are not necessarily unique, so multiple SCOs can have the same display-seq value. In that case, the application must define the display sequence. The default is 0.</td>
</tr>
<tr>
<td>folder-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the folder the SCO belongs to.</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Allowed value</td>
<td>The type of icon used as a visual identifier for the SCO (see icon).</td>
</tr>
<tr>
<td>lang</td>
<td></td>
<td>Allowed value</td>
<td>An abbreviation for the new language (see type for values).</td>
</tr>
<tr>
<td>max-retries</td>
<td></td>
<td>Integer</td>
<td>The number of times the user is allowed to attempt to take the SCO.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of the SCO.</td>
</tr>
<tr>
<td>source-sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of a template from which the SCO is derived.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The content type of the SCO (see type for values). type is a high-level category. icon provides more detail on the type of content.</td>
</tr>
<tr>
<td>version</td>
<td></td>
<td>Integer</td>
<td>The version number of the SCO, incremented when the object is modified or uploaded to the server.</td>
</tr>
<tr>
<td>date-begin</td>
<td></td>
<td>Datetime</td>
<td>If the SCO is a meeting, the date and time the meeting starts.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date and time the SCO was created (or, for content, uploaded).</td>
</tr>
<tr>
<td>date-end</td>
<td></td>
<td>Datetime</td>
<td>If the SCO is a meeting, the date and time the meeting ends.</td>
</tr>
<tr>
<td>date-modified</td>
<td></td>
<td>Datetime</td>
<td>The date and time the SCO was last modified.</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>String</td>
<td>The description of the SCO entered when the SCO was created.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the SCO.</td>
</tr>
<tr>
<td>url-path</td>
<td></td>
<td>String</td>
<td>The path to the SCO on the server.</td>
</tr>
</tbody>
</table>
Sample request

https://example.com/api/xml?action=sco-info&sco-id=2006320683

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <sco account-id="624520" disabled="" display-seq="0"
      folder-id="2006258750" icon="meeting" lang="en" max-retries="" sco-id="2006320683" source-sco-id="-1625529" type="meeting" version="0">
    <date-begin>2006-05-04T11:15:00.000-07:00</date-begin>
    <date-created>2006-05-04T11:27:47.087-07:00</date-created>
    <date-end>2006-05-04T12:15:00.000-07:00</date-end>
    <date-modified>2006-05-04T11:27:47.087-07:00</date-modified>
    <name>Technology and Law Review Meeting</name>
    <url-path>/tlawreview/</url-path>
  </sco>
  <source-sco account-id="624520" display-seq="0" folder-id="-625529" icon="meeting" lang="en" max-retries="" sco-id="-1625529" source-sco-id="-8888" type="meeting" version="0">
    <date-created>2004-10-05T00:49:30.217-07:00</date-created>
    <date-modified>2005-01-04T15:03:25.937-08:00</date-modified>
    <name>Default Meeting Template</name>
    <url-path>/defaultMeetingTemplate/</url-path>
  </source-sco>
  <source-sco account-id="624520" display-seq="0" folder-id="-625529" icon="meeting" lang="en" max-retries="" sco-id="-1625529" source-sco-id="-8888" type="meeting" version="0">
    <date-created>2004-10-05T00:49:30.217-07:00</date-created>
    <date-modified>2005-01-04T15:03:25.937-08:00</date-modified>
    <name>Default Meeting Template</name>
    <url-path>/defaultMeetingTemplate/</url-path>
  </source-sco>
</results>
```

**sco-move**

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Moves a SCO from one folder to another.
To move a SCO to a folder, the current user must have permission to create content in the target folder. In general, users have permission on their own folders (such as my-meetings, my-courses, my-events, my-content, and my-meeting-templates) by default. To move SCOs to a shared folder such as content, courses, and meetings, a user must have Manage permission or be an Administrator.

Request URL
http://server_name/api/xml?action=sco-move&folder-id=integer&sco-id=integer&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folder-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the destination folder.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of the SCO to move.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<<?xml version="1.0" encoding="utf-8" ?>
<results>
   <status code=allowedValue />
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=sco-move&sco-id=2006744233&folder-id=2006258748

Sample response
<<?xml version="1.0" encoding="utf-8" ?>
<results>
<status code="ok" />
</results>

See also
sco-nav
sco-nav

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Describes the folder hierarchy that contains a SCO.

The sco-nav call is useful for creating a navigation tree, breadcrumb trail, or any other type of user interface hierarchy. The response contains a list of sco elements, one for the SCO you are querying and one for each of its enclosing folders up to the top-level folder. The top-level folder is one of the list of folders returned by sco-shortcuts.

In each sco element, the depth attribute indicates how many hierarchical levels the SCO is from the SCO you specify in the request. A depth of 0 indicates the SCO you are querying, a depth of 1 indicates the folder that contains the SCO, and so on.

Request URL
http://server_name/api/xml
?action=sco-nav
&sco-id=integer
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a SCO for which you want a folder hierarchy up to the root level.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
<?xml version="1.0" encoding="utf-8" ?>

<results>
  <status code=allowedValue />
  <sco-nav>
    <sco sco-id=integer type=allowedValue icon=allowedValue depth=integer>
      <name>string</name>
    </sco>
  ...
  </sco-nav>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>sco-nav</td>
<td></td>
<td>Container</td>
<td>The entire navigation tree from the top-level folder to the SCO.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Information about one SCO in the hierarchy.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of the SCO.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The type of SCO (see type for values).</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Allowed value</td>
<td>The icon that visually represents the SCO (see icon for values).</td>
</tr>
<tr>
<td>depth</td>
<td></td>
<td>Integer</td>
<td>A number representing the level of a SCO in the folder hierarchy relative to the SCO passed in the request (0 for the passed SCO, 1 for one level above, and so on). Values increase as you move up the hierarchy toward the top-level folder.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the SCO.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=sco-nav&sco-id=2006334909

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <sco-nav>
    <sco sco-id="624522" type="folder" icon="folder" depth="2">
      <name>User Content</name>
    </sco>
    <sco sco-id="2006258747" type="folder" icon="folder" depth="1">
      <name>joy@acme.com</name>
    </sco>
    <sco sco-id="2006334909" type="content" icon="producer" depth="0">
      <name>Test Quiz</name>
    </sco>
  </sco-nav>
</results>
```

See also

sco-move

**sco-search**

**Availability**
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

**Description**
Provides a list of all SCOs that have content matching the search text.
The sco-search action searches the content of some types of SCOs for the query string. The types of SCOs searched include presentation archives, meeting archives, and the presentation components of a course or curriculum. A presentation that is included in a course returns two sets of results, one for the actual presentation and one for the course. The search does not include the SCO name or any metadata about the SCO stored in the database.

The query is not case-sensitive and allows wildcards at the end of a query string. The allowed wildcards are:

- An asterisk (*) to match any character or characters
- A question mark (?) to match any one character

For example, you can use the query strings quiz, qu*, or qui?. However, you cannot use a wildcard at the beginning or within a query string.

You can also use the operators and and or to return multiple matches, with spaces separating the operator and the search terms, like this:

`https://example.com/api/xml?action=sco-search&query=quiz or test`

If you search on quizortest, for example, the server interprets it as a literal string and returns only exact matches.

**Request URL**

http://server_name/api/xml
  ?action=sco-search
  &query=querystring
  &filter-definition=value
  &sort-definition=value
  &session=BreezeSessionCookieValue

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>Query string</td>
<td>Y</td>
<td>A string to search for. To use any of these special characters in the query string, escape them with a backslash before the character:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&amp; - +</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The query string is not case-sensitive and allows wildcard characters * and ? at the end of the query string.</td>
</tr>
<tr>
<td>filter-definition</td>
<td>Filter definition</td>
<td>N</td>
<td>A filter to reduce the volume of the response.</td>
</tr>
<tr>
<td>sort-definition</td>
<td>Sort definition</td>
<td>N</td>
<td>A sort to return results in a certain sequence.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

**Filters**

You can filter the response on any element or attribute it contains.

**Response structure**

```xml
<?xml version="1.0" encoding="utf-8" />
<results>
  <status code=allowedValue />
  <sco-search-info>
    <sco sco-id=integer folder-id=integer type=allowedValue
      icon=allowedValue byte-count=integer tree-type=integer>
      <name>string</name>
    </sco>
  </sco-search-info>
</results>
```
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>sco-search-info</td>
<td></td>
<td>Container</td>
<td>The list of objects (SCOs) that match the search query.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Details about one object that matches the search.</td>
</tr>
<tr>
<td></td>
<td>sco-id</td>
<td>Integer</td>
<td>The unique ID of the SCO.</td>
</tr>
<tr>
<td></td>
<td>folder-id</td>
<td>Integer</td>
<td>The ID of the folder in which the SCO is stored.</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>Allowed value</td>
<td>The content type assigned to the SCO (see type for values).</td>
</tr>
<tr>
<td></td>
<td>icon</td>
<td>Allowed value</td>
<td>The icon that visually identifies the SCO in a user interface.</td>
</tr>
<tr>
<td></td>
<td>byte-count</td>
<td>Integer</td>
<td>The size of the SCO, in bytes.</td>
</tr>
<tr>
<td></td>
<td>tree-type</td>
<td>Integer</td>
<td>The tree type.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The file name of the SCO.</td>
</tr>
<tr>
<td>url-path</td>
<td></td>
<td>String</td>
<td>The unique identifier that comes after the domain name in the SCO URL.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date the SCO was created.</td>
</tr>
<tr>
<td>date-modified</td>
<td></td>
<td>Datetime</td>
<td>The date the SCO was modified.</td>
</tr>
<tr>
<td>hit</td>
<td></td>
<td>Integer</td>
<td>The sequence number of this occurrence of the query string in the SCO.</td>
</tr>
<tr>
<td>hit-type</td>
<td></td>
<td>Allowed value</td>
<td>The type of content in which the search term was found. Allowed values are metadata and slide.</td>
</tr>
<tr>
<td>hit-url</td>
<td></td>
<td>String</td>
<td>A relative URL to the position where the search term was found in the content, for example, to a specific slide. Must be appended to the url-path.</td>
</tr>
<tr>
<td>thumbnail-path</td>
<td></td>
<td>String</td>
<td>A relative URL to an image of the SCO that contains the search term.</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=sco-search&query=quiz

Sample response

```xml
<results>
  <status code="ok" />
  <sco-search-info>
    ...
  </sco-search-info>
</results>
```
sco-search-by-field

Availability
Acrobat Connect Pro 7

Description
Provides a list of all SCOs matching the search text within the specified field. This action allows you to search for objects in the database based on the SCO’s name, description, or author, or all three of those fields.

The sco-search-by-field action searches the content of some types of SCOs for the query string. The search includes folders, training courses, curriculums, meetings, content, and archives.

To search for multi-word terms with spaces between the words, search only on the first word in the term and use a wildcard at the end. For example, to search for Sales Presentation, use the following string:

query=sales*

Note: The sco-search-by-field command does not support the and/or operators.

Request URL
http://server_name/api/xml
?action=sco-search-by-field
&query=SearchTerm
&field=allowedValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>Y</td>
<td>The term to search for within the specified field. The query is case-insensitive.</td>
</tr>
</tbody>
</table>
| field      | String | N        | The field to search. Accepts four possible values: name, description, author, or allfields:  
|            |        |          | • name searches only the name field for SCOs.  
|            |        |          | • description searches only the description field for SCOs.  
|            |        |          | • author matches the full name field (not the first-name or last-name fields) of the principal that created the SCOs.  
|            |        |          | • allfields searches the name, description, and author fields.  
|            |        |          | If this parameter is omitted, the name field is searched.                  |

Filters

Filters are supported on any field that can be returned. For example, you can use

&filter-gt-date-created=2007-09-12T08:00:00.000

if you want to show only results created after 8:00 AM on September 12, 2007.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />  
  <sco-search-info>
    <sco sco-id=integer tree-id=integer folder-id=integer type=allowedValue
      status=allowedValue sco-data-id=integer source-sco-id=integer host-id=integer author-contact-id=integer learning-time=allowedValue lang=allowedValue seq-id=integer
      icon=allowedValue display-seq=integer max-retries=integer version=integer account-id=integer tree-type=integer>
      <name>string</name>
      <url-path>string</url-path>
      <date-created>datetime</date-created>
      <date-modified>datetime</date-modified>
      <principal-name>string</principal-name>
      <folder-name>string</folder-name>
    </sco>
  </sco-search-info>
</results>
```

Response value

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td></td>
<td>code</td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>sco-search-by-field-info</td>
<td></td>
<td>Container</td>
<td>The list of objects (SCOs) that match the search query.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Details about one object that matches the search.</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Integer</td>
<td>The unique ID of the SCO.</td>
</tr>
</tbody>
</table>
Sample request
https://example.com/api/xml?action=sco-search-by-field&query=Marketing*&field=description

Sample response
<results>
  <$sco-search-by-field-info>
  <sco sco-id="2007775205" tree-id="" folder-id="2007470298" type="meeting" status="" sco-data-id="" source-sco-id="2007470292" host-id="" author-contact-id="" learning-time="" lang="en" seq-id="" icon="virtual-classroom" display-seq="0" max-retries="" version="0" account-id="2007470268" tree-type="4">
    <name>virt1</name>
    <url-path>/r72655596/</url-path>
    <date-created>2007-10-10T16:41:31.643-07:00</date-created>
    <date-modified>2007-10-10T16:41:31.643-07:00</date-modified>
    <principal-name>Piet Pompies</principal-name>
    <folder-name>ppompies@adobe.com</folder-name>
  </sco>
  <sco sco-id="2007775257" tree-id="" folder-id="2007775254" type="folder" status="" sco-data-id="" source-sco-id="" host-id="" author-contact-id="" learning-time="" lang="en" seq-id="" icon="folder" display-seq="0" max-retries="" version="0" account-id="2007470268">
    <name>test1</name>
    <url-path>/f13818712/</url-path>
    <date-created>2007-10-10T18:00:31.083-07:00</date-created>
    <date-modified>2007-10-10T18:00:31.083-07:00</date-modified>
    <principal-name>trainer two</principal-name>
    <folder-name>trainer@two.com</folder-name>
  </sco>
  </sco-search-by-field-info>
</results>

sco-shortcuts

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folder-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the folder in which the SCO is stored.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The content type assigned to the SCO (see type for values).</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Allowed value</td>
<td>The icon that visually identifies the SCO in a user interface.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The file name of the SCO.</td>
</tr>
<tr>
<td>url-path</td>
<td></td>
<td>String</td>
<td>The unique identifier that comes after the domain name in the SCO URL.</td>
</tr>
<tr>
<td>date-created</td>
<td></td>
<td>Datetime</td>
<td>The date the SCO was created.</td>
</tr>
<tr>
<td>date-modified</td>
<td></td>
<td>Datetime</td>
<td>The date the SCO was last modified.</td>
</tr>
<tr>
<td>principal-name</td>
<td></td>
<td>String</td>
<td>The author of the SCO.</td>
</tr>
<tr>
<td>folder-name</td>
<td></td>
<td>String</td>
<td>The name of the folder in which the SCO is stored.</td>
</tr>
</tbody>
</table>
Description
Provides information about the folders relevant to the current user. These include a folder for the user's current meetings, a folder for the user's content, as well as folders above them in the navigation hierarchy.

To determine the URL of a SCO, concatenate the url-path returned by sco-info, sco-contents, or sco-expanded-contents with the domain-name returned by sco-shortcuts. For example, you can concatenate these two strings:
- http://test.server.com (the domain-name returned by sco-shortcuts)
- /f2006123456/ (the url-path returned by sco-info, sco-contents, or sco-expanded-contents)

The result is this URL:
http://test.server.com/f2006123456/

You can also call sco-contents with the sco-id of a folder returned by sco-shortcuts to see the contents of the folder.

Request URL
http://server_name/api/xml
?action=sco-shortcuts
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="code"/>
  <shortcuts>
    <sco tree-id=integer sco-id=integer type=allowedValue>
      <domain-name>string</domain-name>
    </sco>
    ...
  </shortcuts>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>shortcuts</td>
<td></td>
<td>Container</td>
<td>Information about all of the folders that relate to the current user.</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Information about one of the current user's folders.</td>
</tr>
<tr>
<td>tree-id</td>
<td></td>
<td>Integer</td>
<td>The ID of the navigation tree that contains the folder. Several folders might have the same tree-id.</td>
</tr>
</tbody>
</table>
The values that can be returned in the `type` attribute of the `sco` element (for this call only, `sco-shortcuts`) identify Acrobat Connect Pro folders. Each folder type maps to a folder in Acrobat Connect Pro Central and requires certain permission levels to access, described in the following table.

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td></td>
<td>The unique ID of the folder.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td></td>
<td>The type of the folder. Allowed values are shown in the following table.</td>
</tr>
<tr>
<td>domain-name</td>
<td>String</td>
<td></td>
<td>The domain name of the folder.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value of type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-custom</td>
<td>Customized content for an account, such as a customized login page, banner, and so on.</td>
</tr>
<tr>
<td>content</td>
<td>The Shared Content folder. Requires Administrator privilege or Manage permission.</td>
</tr>
<tr>
<td>courses</td>
<td>The Shared Training folder. Requires Administrator privilege or Manage permission.</td>
</tr>
<tr>
<td>events</td>
<td>The Shared Events folder. Requires Administrator privilege or Manage permission.</td>
</tr>
<tr>
<td>meetings</td>
<td>The Shared Meetings folder. Requires Administrator privilege or Manage permission.</td>
</tr>
<tr>
<td>my-courses</td>
<td>The My Training folder. By default, the individual user has Manage permission.</td>
</tr>
<tr>
<td>my-content</td>
<td>The My Content folder. By default, the individual user has Manage permission.</td>
</tr>
<tr>
<td>my-events</td>
<td>The My Events folder. By default, the individual user has Manage permission.</td>
</tr>
<tr>
<td>my-meetings</td>
<td>The My Meetings folder. By default, the individual user has Manage permission.</td>
</tr>
<tr>
<td>my-meeting-templates</td>
<td>The My Templates folder. By default, the individual user has Manage permission.</td>
</tr>
<tr>
<td>seminars</td>
<td>The Shared Seminars folder. Requires Administrator privilege or Manage permission.</td>
</tr>
<tr>
<td>shared-meeting-templates</td>
<td>The Shared Templates folder. Inherits permissions from Shared Meetings.</td>
</tr>
<tr>
<td>user-content</td>
<td>Contain the user content folders.</td>
</tr>
<tr>
<td>user-courses</td>
<td>Contain the user courses folders.</td>
</tr>
<tr>
<td>user-events</td>
<td>Contain the user events folders.</td>
</tr>
<tr>
<td>user-meetings</td>
<td>Contain the user meeting folders.</td>
</tr>
</tbody>
</table>

Sample request
http://example.com/api/xml?action=sco-shortcuts

Sample response
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="ok" />
    <shortcuts>
        <sco tree-id="4930295" sco-id="2006258748" type="my-courses">
            <domain-name>http://example.com</domain-name>
        </sco>
        <sco tree-id="4930293" sco-id="2006258749" type="my-events">
            <domain-name>http://example.com</domain-name>
        </sco>
        ...
    </shortcuts>
</results>
```
See also
sco-info, sco-expanded-contents

sco-update

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Creates metadata for a SCO, or updates existing metadata describing a SCO.

Call sco-update to create metadata only for SCOs that represent content, including meetings. You also need to upload content files with either sco-upload or Acrobat Connect Pro Central.

You must provide a folder-id or a sco-id, but not both. If you pass a folder-id, sco-update creates a new SCO and returns a sco-id. If the SCO already exists and you pass a sco-id, sco-update updates the metadata describing the SCO.

After you create a new SCO with sco-update, call permissions-update to specify which users and groups can access it.

Request URL
http://server_name/api/xml
?action=sco-update
&author-info-1=string
&author-info-2=string
&author-info-3=string
&date-begin=datetime
&date-end=datetime
&description=string
$email=string
&first-name=string
&folder-id=integer
&icon=allowedValue
&lang=allowedValue
&last-name=string
&name=string
&sco-id=integer
&sco-tag=string
&source-sco-id=integer
&type=allowedValue
$url-path=string
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>author-info-1</td>
<td>String</td>
<td>N</td>
<td>Information about the author. Used only with presentations. Can be used for the author's name or any other information.</td>
</tr>
<tr>
<td>author-info-2</td>
<td>String</td>
<td>N</td>
<td>Additional information about the author. Used only with presentations. Can be used for the author's professional title or any other information.</td>
</tr>
<tr>
<td>author-info-3</td>
<td>String</td>
<td>N</td>
<td>Additional information about the author. Used only with presentations. Can be used for the author's company name or any other information.</td>
</tr>
<tr>
<td>date-begin</td>
<td>Datetime</td>
<td>N</td>
<td>The scheduled beginning date and time, in ISO 8601 format. Used only for meetings and courses.</td>
</tr>
<tr>
<td>date-end</td>
<td>Datetime</td>
<td>N</td>
<td>The scheduled ending date and time, in ISO 8601 format. Used only for meetings and courses.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>N</td>
<td>A description of the SCO to be displayed in the user interface.</td>
</tr>
<tr>
<td>email</td>
<td>String</td>
<td>N</td>
<td>The e-mail address of the contact person for a presentation (used only with presentation SCOs).</td>
</tr>
<tr>
<td>first-name</td>
<td>String</td>
<td>N</td>
<td>The first name of the contact person for a presentation (used only with presentation SCOs).</td>
</tr>
<tr>
<td>folder-id</td>
<td>Integer</td>
<td>Y/N</td>
<td>The ID of the folder in which a new SCO will be stored. Required for a new SCO, but do not use for an existing SCO.</td>
</tr>
<tr>
<td>lang</td>
<td>Allowed value</td>
<td>N</td>
<td>An abbreviation for the language associated with the SCO (see lang for values). If not specified, the default value for the folder in which the SCO is created is used.</td>
</tr>
<tr>
<td>icon</td>
<td>Allowed value</td>
<td>N</td>
<td>The visual symbol used to identify a SCO in Acrobat Connect Pro Central; also provides information about the SCO in addition to its type.</td>
</tr>
<tr>
<td>last-name</td>
<td>String</td>
<td>N</td>
<td>The last name of the contact person for a presentation (used only with presentations).</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Y/N</td>
<td>The name of the SCO, with or without spaces. Required to create a SCO.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y/N</td>
<td>The unique ID of a SCO to update. Use sco-id or folder-id, but not both. Required to update an existing SCO.</td>
</tr>
<tr>
<td>sco-tag</td>
<td>String</td>
<td>N</td>
<td>A label for any information you want to record about a course. Use only with courses.</td>
</tr>
<tr>
<td>source-sco-id</td>
<td>Integer</td>
<td>N</td>
<td>The unique ID of a template you can use to create a meeting or a piece of content from which you can build a course.</td>
</tr>
<tr>
<td>type</td>
<td>Allowed value</td>
<td>N</td>
<td>The type of the new SCO (for allowed values, see type). The default value is content.</td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td>N</td>
<td>The custom part of the URL to the meeting room that comes after the domain name. The url-path must be unique within the folder. If not specified, the server assigns a value.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.
Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <sco folder-id=integer lang=allowedValue type=allowedValue
    sco-id=integer version=integer account-id=integer icon=integer>
    <url-path>string</url-path>
    <description>string</description>
    <name>string</name>
  </sco>
</results>

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>sco</td>
<td></td>
<td>Container</td>
<td>Information about a new SCO just created, including the sco-id. Returned only if you create a SCO.</td>
</tr>
<tr>
<td>folder-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the folder in which the new SCO is stored.</td>
</tr>
<tr>
<td>lang</td>
<td></td>
<td>Allowed value</td>
<td>A code for the language associated with the SCO (see lang for values).</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Allowed value</td>
<td>The type of the new SCO (see type for values).</td>
</tr>
<tr>
<td>sco-id</td>
<td></td>
<td>Allowed value</td>
<td>The unique ID of the new SCO.</td>
</tr>
<tr>
<td>version</td>
<td></td>
<td>Integer</td>
<td>The version number of the new SCO. When the SCO is first created, the version is 0.</td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the account in which the new SCO is created.</td>
</tr>
<tr>
<td>icon</td>
<td></td>
<td>Integer</td>
<td>The type of icon that identifies a new SCO in Acrobat Connect Pro Central (see icon for values).</td>
</tr>
<tr>
<td>url-path</td>
<td>String</td>
<td></td>
<td>The part of the SCO URL that comes after the domain name and uniquely identifies the SCO.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td></td>
<td>A text description of the SCO.</td>
</tr>
<tr>
<td>name</td>
<td></td>
<td>String</td>
<td>The name of the SCO.</td>
</tr>
</tbody>
</table>

Sample request
https://example.com/api/xml?action=sco-update&folder-id=2006258747
&description=test&name=More About Web Communities&type=content
&lang=en

Sample response
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <sco folder-id="2006258747" account-id="624520" type="content" lang="en"
    icon="content" sco-id="2006752036" version="0">
    <url-path>/p53884157/</url-path>
    <description>test</description>
    <name>More About Web Communities</name>
  </sco>
</results>
sco-upload

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Uploads a file to the server and then builds the file, if necessary.
If you are adding a new file, call sco-update first and pass the sco-id returned to sco-upload. If you are updating the content of a file that already exists on the server, you can call sco-upload directly.
You must call sco-upload within an HTML form element. The form element must have an encoding type of multipart/form-data. The HTML form must also have an input element with name=file, as this example shows:

```xml
  <P>
    What files are you sending?
    <INPUT type="file" name="file">
    <BR>
    <INPUT type="submit" value="Send"> <INPUT type="reset">
  </FORM>
```

This form uploads a single file. To upload multiple files (for example, a PPT and a PPC file), you must use additional input elements with name=file, for example:

```xml
  <P>
    PPT files you are sending <INPUT type="file" name="file"></P>
    PPC files you are sending <INPUT type="file" name="file">
    <INPUT type="submit" value="Send"> <INPUT type="reset">
  </FORM>
```

After the upload, call sco-info to get the status of the SCO. The status is initially in-progress, which means that the content is being built. When the status becomes active, the content build is finished, and users can access the content.

Request URL
http://server_name/api/xml
?action=sco-upload
&file=formElementName
&sco-id=integer
&summary=string
$title=string
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file</td>
<td>Form element name</td>
<td>Y</td>
<td>The file to upload, sent from an input element with name=file in an HTML form. The HTML form must also have an encoding type of multipart/form-data defined in the form element.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the SCO you want to upload, returned by sco-update.</td>
</tr>
<tr>
<td>summary</td>
<td>String</td>
<td>N</td>
<td>A brief description of the SCO that Acrobat Connect Pro Central or your application displays.</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>N</td>
<td>The title of the SCO.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters

Results cannot be filtered or sorted.

Response structure

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <files>
    <file>
      <path>string</path>
    </file>
  </files>
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Container</td>
<td></td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td>Empty, with attributes</td>
<td></td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td>Allowed value</td>
<td></td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>files</td>
<td>Container</td>
<td></td>
<td>Information about all of the uploaded files. Deprecated and may be removed in a future release.</td>
</tr>
<tr>
<td>file</td>
<td>Container</td>
<td></td>
<td>Information about one file. Deprecated and may be removed in a future release.</td>
</tr>
<tr>
<td>path</td>
<td>String</td>
<td></td>
<td>The path to the newly uploaded file. For Adobe internal use only. Deprecated and may be removed in a future release.</td>
</tr>
</tbody>
</table>

Sample request

This request is created by uploading a file through an HTML form:

http://example.com/api/xml?action=sco-upload&sco-id=2006768386

Sample response

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <files>
    <file>
      <path>string</path>
    </file>
  </files>
</results>
```
user-accounts

Availability
Breeze 4 and Breeze 5; Connect Enterprise Web Services 6

Description
Provides a list of the accounts a user belongs to.
The user-accounts action is only used when a user belongs to more than one account on the server and uses the same login ID and password for each. In that case, a user's login is likely to fail with a status message of too-much-data. This action is useful when you want to retrieve a list of a user's accounts and give the user a choice of which account to log in to.

Request URL
http://server_name/api/xml
?action=user-accounts
&login=string
&password=string
&session=BreezeSessionCookieValue

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>login</td>
<td>String</td>
<td>Y</td>
<td>The user's login name, which may be the user's e-mail address.</td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td>Y</td>
<td>The user's password.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREESESESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted. The default sort is by account-name.

Response structure
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
  <users>
    <user user-id=integer account-id=integer>
      <name>string</name>
      <date-expired>datetime</date-expired>
    </user>
    ...
  </users>
</results>
Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
<tr>
<td>users</td>
<td></td>
<td>Container</td>
<td>Information about the accounts the user belongs to.</td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td></td>
<td>The ID of the account the user belongs to.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
<td>The name of the account the user belongs to.</td>
</tr>
<tr>
<td>date-expired</td>
<td>Datetime</td>
<td></td>
<td>The date and time the user's login expires.</td>
</tr>
</tbody>
</table>

Sample request

https://sample.com/api/xml?action=user-accounts&login=joy@acme.com
&password=bigdog

Sample response

<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <users>
    <user user-id="2006258745" account-id="624520">
      <name>Test Account</name>
      <date-expired>2099-12-31T16:00:00.000-08:00</date-expired>
    </user>
  </users>
</results>

**user-transcript-update**

**Availability**

Breeze 5; Connect Enterprise Web Services 6

**Description**

Overrides the score on an item within a curriculum.

For example, you can use `user-transcript-update` to give a user a score for an external training. This action works only for items within a curriculum, and you need manage permission for the curriculum.

**Request URL**

http://server_name/api/xml
?action=user-transcript-update
&curriculum-id=integer
&sco-id=integer
&status=allowedValue
&score=integer
&principal-id=integer
&session=BreezeSessionCookieValue
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curriculum-id</td>
<td>Integer</td>
<td>N</td>
<td>The ID of the curriculum.</td>
</tr>
<tr>
<td>sco-id</td>
<td>Integer</td>
<td>Y</td>
<td>The unique ID of a SCO with a score you want to override.</td>
</tr>
<tr>
<td>status</td>
<td>Allowed value</td>
<td>Y</td>
<td>A value showing the status of the user's attempt to use this SCO. Allowed values are completed, incomplete, user-passed, user-failed, and not-attempted.</td>
</tr>
<tr>
<td>score</td>
<td>Integer</td>
<td>Y</td>
<td>An integer value that represents the score the user has attained on this SCO.</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the user whose transcript will be overridden.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure

```xml
<results>
  <status code="allowedValue" />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see status).</td>
</tr>
</tbody>
</table>

Sample request

https://example.com/api/xml?action=user-transcript-update
&curriculum-id=2006298444&sco-id=2006298445&status=user-passed
&principal-id=2006258745&score=100

Sample response

```xml
<results>
  <status code="ok" />
</results>
```

See also

learning-path-info, learning-path-update

user-update-pwd

Availability

Breeze 4 and Breeze 5; Connect Enterprise Web Services 6
Description
Changes a user's password. A password can be changed in either of these cases:

- By an Administrator logged in to the account, with or without the user's old password
- By any Acrobat Connect Pro Server user, with the user's principal-id number, login name, and old password

An Administrator can create rules for valid passwords on the server. These rules might include, for example, the number and types of characters a password must contain. If a user submits a new password that does not adhere to the rules, Acrobat Connect Pro would throw an error showing that the new password is invalid.

When you call `user-update-pwd`, the password is sent over HTTP or HTTPS in hashed form.

Request URL
```
http://server_name/api/xml
?action=user-update-pwd
&user-id=integer
&password-old=string
&password=string
&password-verify=string
&session=BreezeSessionCookieValue
```

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-id</td>
<td>Integer</td>
<td>Y</td>
<td>The ID of the user.</td>
</tr>
<tr>
<td>password-old</td>
<td>String</td>
<td>Y/N</td>
<td>The user’s current password. Required for regular users, but not for Adminis-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>trator users.</td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td>Y</td>
<td>The new password.</td>
</tr>
<tr>
<td>password-verify</td>
<td>String</td>
<td>Y</td>
<td>A second copy of the new password, for verification.</td>
</tr>
<tr>
<td>session</td>
<td>String</td>
<td>N</td>
<td>The value of the BREEZESSESSION cookie. Use this parameter if you do not use a client-side cookie management library.</td>
</tr>
</tbody>
</table>

Filters
Results cannot be filtered or sorted.

Response structure
```
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```

Response values

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td></td>
<td>Container</td>
<td>All results the action returns.</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td>Empty, with attributes</td>
<td>The status of the response.</td>
</tr>
<tr>
<td>code</td>
<td></td>
<td>Allowed value</td>
<td>A code indicating the response status (see <code>status</code>).</td>
</tr>
</tbody>
</table>

Sample request
This request can be used by an Administrator to change a user's password without knowing the old password:
```
https://example.com/api/xml?action=user-update-pwd&user-id=12345&password=newone&password-verify=newone
```
Sample response
This response shows that the change was successful:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="ok" />
</results>
```
Chapter 7: Filter and sort reference

This chapter is a reference for filters and sort values you use to reduce the volume of the response from XML actions in Adobe® Acrobat® Connect™ Pro Web Services.

**filter-definition**

**Description**

A filter is a special type of parameter that reduces the volume of the response. When you see `filter-definition` in a request URL syntax, substitute a filter definition.

To create a filter definition, start with the keyword `filter`, add an modifier (if desired), then a field name (if allowed), and then a value, using this syntax:

`filter-modifier-field=value`

The modifiers you can add are listed in the following table.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>filter-field=value</code></td>
<td>Returns all items for which the data in field exactly matches value.</td>
</tr>
<tr>
<td><code>filter-like-field=value</code></td>
<td>Returns all items with the string value within field, even if field is not an exact match.</td>
</tr>
<tr>
<td><code>filter-out-field=value</code></td>
<td>Filters out or excludes any items with value in field.</td>
</tr>
<tr>
<td><code>filter-rows=value</code></td>
<td>Limits the results to the number of rows specified in value.</td>
</tr>
<tr>
<td><code>filter-start=value</code></td>
<td>Starts the results at the index number specified in value.</td>
</tr>
<tr>
<td><code>filter-gt-datefield=value</code></td>
<td>Selects all items with a data after value. Works only with date fields. The value must be a date in ISO 8601 format.</td>
</tr>
<tr>
<td><code>filter-lt-datefield=value</code></td>
<td>Selects all items with a data earlier than value. Works only with date fields. The value must be a date in ISO 8601 format.</td>
</tr>
<tr>
<td><code>filter-gte-datefield=value</code></td>
<td>Selects all items with a value in field greater than or equal to value. Works only with date fields. The value must be a date in ISO 8601 format.</td>
</tr>
<tr>
<td><code>filter-lte-datefield=value</code></td>
<td>Selects all items with a value in field less than or equal to value. Works only with dates. The date uses ISO 8601 format.</td>
</tr>
<tr>
<td><code>filter-ismember=value</code></td>
<td>Selects all principals that are members of a group, specified in a separate parameter. Takes a Boolean value of true or false.</td>
</tr>
</tbody>
</table>

The value is case insensitive. For example, either of these filters matches a meeting with the name *August All Hands Meeting*:

`&filter-name=August All Hands Meeting`

`&filter-name=august all hands meeting`

Some modifiers require a field name on which to filter results, for example, `name`. Other filters do not take a field name. For those filters that accept field names, the allowed fields vary for different actions. Check a specific action in “Action reference” on page 54 to learn which field names you can use in filters.
**Exact match filter**

filter-name=Goals Review

Matches items with *Goals Review* (or any mixed case pattern of the same string) in the name.

**Similar match filter**

filter-like-name=Goals

Matches any item that includes *Goals* (or any mixed case pattern of the same string) in the name, including *Goals Review* and *Quarterly Goals*.

**Exclude items filter**

filter-out-name=Status

Excludes all items with *Status* (or any mixed case pattern of the same string) in the name.

**Match and exclude items**

filter-like-name=Goals&filter-out-status=active

Matches any item with *Goals* (or any mixed case pattern of the same string) in the name that is no longer active.

**Match a start date**

filter-gt-date-begin=2005-05-01&sort-name=asc

Matches any item with a start date of May 1, 2005, sorting the items in ascending order by name.

**Match a date range**


Returns all items with a start date after May 1, 2005 and before May 31, 2005.

**See also**

sort-definition

---

**sort-definition**

When you see `sort-definition` in a request URL in this reference, create a sort filter with a field name and a value describing how you want the results sorted, in this syntax:

`sort-field=value`

Replace `sort` with any of these exact values: `sort` (for a single sort), `sort1` (for the primary sort of two), or `sort2` (for a secondary sort on the results returned by `sort1`).

The `field` variable defines the field you are sorting on. The fields you can use vary by call, so check the API reference for the call you are making.
The value is always **asc** (for *ascending*) or **desc** (for *descending*), defining the sequence of the results. Putting this all together, the parts of a sort filter are shown in the following table:

<table>
<thead>
<tr>
<th>Sort Type</th>
<th>Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort</td>
<td>Vary by call.</td>
<td>asc or desc</td>
<td>Sort results by the specified field, in either ascending or descending order.</td>
</tr>
<tr>
<td>sort1</td>
<td>Vary by call.</td>
<td>asc or desc</td>
<td>Sort results by a field, either ascending or descending, and then pass the results to the next sort.</td>
</tr>
<tr>
<td>sort2</td>
<td>Vary by call.</td>
<td>asc or desc</td>
<td>When results returned by the primary sort are equal, such as same name or group, do a secondary sort by the specified field in either ascending or descending order.</td>
</tr>
</tbody>
</table>

Simple examples of **sort-field=value**, with one level of sort, look like this:

- `sort-name=asc`
- `sort-date=desc`

The following table gives you more detail on how the sort values **asc** and **desc** work:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asc</td>
<td>Ascending order. For alphabetical lists, begin with A and end with Z. For lists ordered by number or date, start with lowest number or earliest date.</td>
</tr>
<tr>
<td>desc</td>
<td>Descending order. For alphabetical lists, begin with Z and end with A. For lists ordered by number or date, start with highest number or most recent date.</td>
</tr>
</tbody>
</table>

Your results may call for using both primary and secondary sorts with **sort1** and **sort2**. For example, when calling `principal-list` to list principals, you can do a primary sort on the `type` field, and then a secondary sort on the `name` field (this way, all principals of a specific type are grouped together and then sorted by name in each group).

You would specify two levels of sort like this:

- `sort1-type=asc&sort2-name=desc`

**See also**

- `filter-definition`
Chapter 8: Common reference

This reference section describes XML elements and attributes that are used by more than one action in Adobe® Acrobat® Connect™ Pro Web Services. The elements described here are referenced from the request and response tables describing actions in the Web Services XML API.

All parameter, element, and attribute names and values are case sensitive. For example, name is not the same as Name, and sco-id is not equivalent to sco-ID. You must enter them exactly as shown in this reference.

access

Description
An attribute describing the level of access a user has to a course or curriculum.

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>access-blocked</td>
<td>The course or curriculum is restricted and users cannot take it.</td>
</tr>
<tr>
<td>access-hidden</td>
<td>The course or curriculum is restricted, users cannot take it, and it is hidden in Acrobat Connect Pro Central (or the user interface of a custom application, if you use this value).</td>
</tr>
<tr>
<td>access-open</td>
<td>The course or curriculum is open and users can take it.</td>
</tr>
<tr>
<td>access-optional</td>
<td>The course or curriculum is optional.</td>
</tr>
<tr>
<td>access-pass</td>
<td>The user has already taken the course or curriculum and passed it.</td>
</tr>
</tbody>
</table>

feature-id

Description
An attribute describing a feature that either users can use or things that can occur during a meeting. Use feature-id with the meeting-feature-update action.

For more information on the pods that can be enabled or disabled, see the Acrobat Connect Pro User Guide.

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description of functionality when value is enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>fid-archive</td>
<td>Lets a host start and stop the recording of a meeting. Disabling this setting means that recording settings are not controllable by the host. To set Connect to automatically record all meetings, you must both disable fid-archive and enable fid-archive-force.</td>
</tr>
<tr>
<td>fid-archive-force</td>
<td>Sets all meetings to be recorded upon the start of the meeting. Recorded meetings appear in Acrobat Connect Pro Central.</td>
</tr>
<tr>
<td>fid-archive-publish-link</td>
<td>When meetings are set to be automatically recorded (by enabling fid-archive-force), lets host create a link to the recording in the meeting folder.</td>
</tr>
<tr>
<td>Value</td>
<td>Description of functionality when value is enabled</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fid-chat-transcripts</td>
<td>Creates a transcript file (one per meeting session) of all text messages exchanged in Chat, Q&amp;A, and IM Pods.</td>
</tr>
<tr>
<td>fid-meeting-app-sharing</td>
<td>Lets host request control of attendee’s input device (mouse or keyboard) when attendee is sharing their screen, desktop, or application.</td>
</tr>
<tr>
<td>fid-meeting-auto-promote</td>
<td>Lets hosts enable the option to automatically promote participants to presenters.</td>
</tr>
<tr>
<td>fid-meeting-breakout</td>
<td>Allows users to create breakout meetings.</td>
</tr>
<tr>
<td>fid-meeting-chat</td>
<td>Enables the Chat pod.</td>
</tr>
<tr>
<td>fid-meeting-chat-clear</td>
<td>Automatically clears Chat pod history when a new session of an existing meeting is started.</td>
</tr>
<tr>
<td>fid-meeting-chat-presenter</td>
<td>Lets users send chat messages only to the presenter.</td>
</tr>
<tr>
<td>fid-meeting-chat-private</td>
<td>Lets users send chat messages to specific attendees.</td>
</tr>
<tr>
<td>fid-meeting-chat-public</td>
<td>Lets users send chat messages to all attendees.</td>
</tr>
<tr>
<td>fid-meeting-desktop-sharing</td>
<td>Lets users share their screen (both the complete desktop and individual applications).</td>
</tr>
<tr>
<td>fid-meeting-dialout</td>
<td>Lets users use Call Out and Call Me features.</td>
</tr>
<tr>
<td>fid-meeting-disclaimer</td>
<td>Shows a disclaimer (for example, explaining that the meeting is being recorded) when a user starts or attends any meeting in this account. To proceed with the meeting, the host or attendees must first accept the disclaimer. If a user does not accept, the disclaimer user is returned to the Connect home page. To set the text of the disclaimer, use the meeting-disclaimer-update action.</td>
</tr>
<tr>
<td>fid-meeting-enhanced-rights</td>
<td>Lets host change the access of attendees to specific subfeatures.</td>
</tr>
<tr>
<td>fid-meeting-file-share</td>
<td>Enables the File Share pod.</td>
</tr>
<tr>
<td>fid-meeting-flv</td>
<td>Lets users use FLV files and mp3 files in the meeting.</td>
</tr>
<tr>
<td>fid-meeting-full-screen-affects-all</td>
<td>Enables &quot;Presenter Changes affect everyone&quot; for full-screen mode. Note: this feature does not enable or disable users' ability to enter full-screen mode.</td>
</tr>
<tr>
<td>fid-meeting-hold</td>
<td>Lets hosts place participants on hold.</td>
</tr>
<tr>
<td>fid-meeting-host-cursors</td>
<td>Lets a host change the display of the host's cursor.</td>
</tr>
<tr>
<td>fid-meeting-im</td>
<td>Enables the Instant Messages pod. This feature is part of the Acrobat Connect Pro integration with supported Microsoft real-time communication servers. Disable this feature when you want to show the Invitee List pod (fid-meeting-invitee-presence-true) but hide the associated Instant Messages pod.</td>
</tr>
<tr>
<td>fid-meeting-invitee-presence</td>
<td>Enables the Invitee List pod and the associated Instant Messages pod. This feature is part of the Acrobat Connect Pro integration with supported Microsoft real-time communication servers.</td>
</tr>
<tr>
<td>fid-meeting-manage-link</td>
<td>Enables the &quot;Manage Room with Web Manager&quot; link in the meeting menu.</td>
</tr>
<tr>
<td>fid-meeting-note</td>
<td>Enables the Notes pod.</td>
</tr>
<tr>
<td>fid-meeting-notes-clear</td>
<td>Automatically clears Notes pod history when a new session of an existing meeting is started.</td>
</tr>
<tr>
<td>fid-meeting-pause-annotate</td>
<td>Lets users pause during screen sharing and annotate on an overlay white board.</td>
</tr>
<tr>
<td>fid-meeting-people-list</td>
<td>Enables the Attendee List pod.</td>
</tr>
<tr>
<td>fid-meeting-poll</td>
<td>Enables the Poll pod.</td>
</tr>
<tr>
<td>fid-meeting-pres-only</td>
<td>Enables the use of the Presenter-Only area.</td>
</tr>
</tbody>
</table>
### field

**Description**

An element containing information about a custom field defined for an object or principal.

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>permission-id</td>
<td>Allowed value</td>
<td>The permission needed to access the custom field. See <code>permission-id</code> for valid values.</td>
</tr>
<tr>
<td>object-type</td>
<td>Allowed value</td>
<td>A definition for a valid object on the server (see <code>type</code> for values).</td>
</tr>
<tr>
<td>field-id</td>
<td>String</td>
<td>A piece of text that identifies the custom field. Acrobat Connect Pro Central does not display the <code>field-id</code>, but various actions return it.</td>
</tr>
<tr>
<td>account-id</td>
<td>Integer</td>
<td>An ID for the user who is presently logged in, assigned by the server.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>display-seq</td>
<td>Integer</td>
<td>The order in which the custom field is displayed in the user interface or returned by the action.</td>
</tr>
<tr>
<td>field-type</td>
<td>String</td>
<td>The type of data the field accepts. Allowed values are text, textarea, and password.</td>
</tr>
<tr>
<td>is-primary</td>
<td>Boolean</td>
<td>Whether the custom field can be deleted. true means the field cannot be deleted. false means it can.</td>
</tr>
<tr>
<td>is-required</td>
<td>Boolean</td>
<td>Whether the custom field is required. true means a value must be specified for this field in each principal or SCO that uses it. false means values for this field are not required.</td>
</tr>
<tr>
<td>acl-id</td>
<td>Integer</td>
<td>The custom field's ID in an access control list (ACL).</td>
</tr>
<tr>
<td>custom-seq</td>
<td>Integer</td>
<td>The sequence number assigned to the custom field in UI display.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>The type of custom field (see type for values).</td>
</tr>
<tr>
<td>principal-id</td>
<td>Integer</td>
<td>The ID of the principal for whom the custom field is defined.</td>
</tr>
</tbody>
</table>

**icon**

**Description**
The symbol used to identify a SCO in Acrobat Connect Pro Central.

**Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive</td>
<td>An archive of an Adobe Acrobat Acrobat Connect Pro meeting.</td>
</tr>
<tr>
<td>attachment</td>
<td>A piece of content uploaded as an attachment.</td>
</tr>
<tr>
<td>authorware</td>
<td>A piece of multimedia content created with Macromedia® Authorware® from Adobe.</td>
</tr>
<tr>
<td>captivate</td>
<td>A demo or movie created with Adobe Captivate™.</td>
</tr>
<tr>
<td>course</td>
<td>A training course.</td>
</tr>
<tr>
<td>curriculum</td>
<td>A curriculum.</td>
</tr>
<tr>
<td>external-event</td>
<td>An external training that can be added to a curriculum.</td>
</tr>
<tr>
<td>flv</td>
<td>A media file in the FLV file format.</td>
</tr>
<tr>
<td>html</td>
<td>An HTML file.</td>
</tr>
<tr>
<td>image</td>
<td>An image.</td>
</tr>
<tr>
<td>lms-plugin</td>
<td>A piece of content from an external learning management system.</td>
</tr>
<tr>
<td>logos</td>
<td>A custom logo used in a meeting room or Acrobat Connect Pro Central.</td>
</tr>
<tr>
<td>meeting-template</td>
<td>A custom look and feel for a meeting.</td>
</tr>
<tr>
<td>mp3</td>
<td>An MP3 file.</td>
</tr>
</tbody>
</table>
**lang**

Description
A two-letter or three-letter code describing a language according to the ISO 639 specifications. ISO 639-1 describes two-letter codes, and ISO 639-2 describes three-letter codes. The language code affects an Acrobat Connect Pro application display, for example, a meeting room or a Acrobat Connect Pro Central display.

Values

<table>
<thead>
<tr>
<th>Two-Letter Value</th>
<th>Three-Letter Value</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>en</td>
<td>eng</td>
<td>English</td>
</tr>
<tr>
<td>fr</td>
<td>fre</td>
<td>French (do not use fra)</td>
</tr>
<tr>
<td>de</td>
<td>ger</td>
<td>German (do not use deu)</td>
</tr>
<tr>
<td>ja</td>
<td>jpn</td>
<td>Japanese</td>
</tr>
<tr>
<td>ko</td>
<td>kor</td>
<td>Korean</td>
</tr>
</tbody>
</table>

**object-type**

Description
An attribute describing the type of a Acrobat Connect Pro object.

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>object-type-account</td>
<td>An account that contains principals and SCOs.</td>
</tr>
<tr>
<td>object-type-action</td>
<td>An action in the Web Services XML API.</td>
</tr>
<tr>
<td>object-type-event</td>
<td>An Acrobat Connect Pro event.</td>
</tr>
<tr>
<td>object-type-hidden</td>
<td>A SCO that is not visible in Acrobat Connect Pro Central (or in your application, if you use this value).</td>
</tr>
<tr>
<td>object-type-meeting</td>
<td>An Acrobat Connect Pro meeting.</td>
</tr>
</tbody>
</table>
path-type

Description
The path-type attribute describes a type of learning path between two objects in a curriculum, for example, whether one must be completed as a prerequisite to the next.

Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>completion-none</td>
<td>The current SCO is not a completion requirement for the curriculum.</td>
</tr>
<tr>
<td>completion-required</td>
<td>The current SCO is a completion requirement.</td>
</tr>
<tr>
<td>prereq-none</td>
<td>The current learning object has no prerequisites.</td>
</tr>
<tr>
<td>prereq-required</td>
<td>The current SCO has a prerequisite that must be completed.</td>
</tr>
<tr>
<td>prereq-hidden</td>
<td>The target learning object is required as a prerequisite. The current learning object is hidden until the target learning object is completed.</td>
</tr>
<tr>
<td>prereq-suggested</td>
<td>The current SCO has a prerequisite that is recommended, not required.</td>
</tr>
<tr>
<td>preass-blocked</td>
<td>The current SCO has a test-out. If the enrollee passes, this item is locked. If the enrollee fails, this item is available.</td>
</tr>
<tr>
<td>preass-none</td>
<td>The current SCO has no test-outs.</td>
</tr>
<tr>
<td>preass-hidden</td>
<td>The current SCO has a test-out. If the user passes, the current SCO can be hidden from the user. If the user fails, the current SCO is visible and the user can enroll.</td>
</tr>
<tr>
<td>preass-optional</td>
<td>The current SCO has a test-out. If the user passes, the current SCO is no longer required to complete the curriculum.</td>
</tr>
</tbody>
</table>

permission-id

Description
The permission-id parameter (or attribute) defines a permission. Depending on the context of the action or response, the permission might be one a principal has on a SCO, or a permission that is needed in order to execute an action.
Values

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>view</td>
<td>The principal can view, but cannot modify, the SCO. The principal can take a course, attend a meeting as participant, or view a folder's content.</td>
</tr>
<tr>
<td>host</td>
<td>Available for meetings only. The principal is host of a meeting and can create the meeting or act as presenter, even without view permission on the meeting's parent folder.</td>
</tr>
<tr>
<td>mini-host</td>
<td>Available for meetings only. The principal is presenter of a meeting and can present content, share a screen, send text messages, moderate questions, create text notes, broadcast audio and video, and push content from web links.</td>
</tr>
<tr>
<td>remove</td>
<td>Available for meetings only. The principal does not have participant, presenter or host permission to attend the meeting. If a user is already attending a live meeting, the user is not removed from the meeting until the session times out.</td>
</tr>
<tr>
<td>publish</td>
<td>Available for SCOs other than meetings. The principal can publish or update the SCO. The publish permission includes view and allows the principal to view reports related to the SCO. On a folder, publish does not allow the principal to create new subfolders or set permissions.</td>
</tr>
<tr>
<td>manage</td>
<td>Available for SCOs other than meetings or courses. The principal can view, delete, move, edit, or set permissions on the SCO. On a folder, the principal can create subfolders or view reports on folder content.</td>
</tr>
<tr>
<td>denied</td>
<td>Available for SCOs other than meetings. The principal cannot view, access, or manage the SCO.</td>
</tr>
</tbody>
</table>

Special permissions

The server defines a special principal, public-access, which combines with values of permission-id to create special access permissions to meetings:

- principal-id=public-access and permission-id=view-hidden means the Acrobat Connect meeting is public, and anyone who has the URL for the meeting can enter the room.
- principal-id=public-access and permission-id=remove means the meeting is protected, and only registered users and accepted guests can enter the room.
- principal-id=public-access and permission-id=dened means the meeting is private, and only registered users and participants can enter the room.

quota-ID

Description

The quota-ID parameter defines a quota in the system. The quota type you specify determines the value of acl-id to use.

Values

<table>
<thead>
<tr>
<th>Quota type</th>
<th>Quota description</th>
<th>Corresponding acl-id to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>live-user-quota</td>
<td>The number of account-wide Meeting Attendees.</td>
<td>The account-id of the account.</td>
</tr>
</tbody>
</table>
**status**

**Description**
A status code returned by Acrobat Connect Pro in the response to all actions in the Web Services XML API.

**Response structure**
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue />
</results>
```

**or**
```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code=allowedValue>
    <invalid field=string type=allowedValue subcode=allowedValue />
  </status>
</results>
```

**Attributes**

code  The status of the response.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invalid</td>
<td>Indicates that a call is invalid in some way. The invalid element provides more detail.</td>
</tr>
<tr>
<td>no-access</td>
<td>Indicates that you don’t have permission to call the action. The subcode attribute provides more details.</td>
</tr>
<tr>
<td>no-data</td>
<td>Indicates that there is no data available (in response to an action that would normally result in returning data). Usually indicates that there is no item with the ID you specified. To resolve the error, change the specified ID to that of an item that exists.</td>
</tr>
<tr>
<td>ok</td>
<td>Indicates that the action has completed successfully.</td>
</tr>
<tr>
<td>too-much-data</td>
<td>Indicates that the action should have returned a single result but is actually returning multiple results. For example, if there are multiple users with the same user name and password, and you call the login action using that user name and password as parameters, the system cannot determine which user to log you in as, so it returns a too-much-data error.</td>
</tr>
</tbody>
</table>
subcode If present, provides more detail describing the status of the response. For example, subcode values are used to differentiate between different situations in which code is set to no-access.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-expired</td>
<td>The customer account has expired.</td>
</tr>
<tr>
<td>denied</td>
<td>Based on the supplied credentials, you don't have permission to call the action.</td>
</tr>
<tr>
<td>no-login</td>
<td>The user is not logged in. To resolve the error, log in (using the login action) before you make the call. For more information, see login.</td>
</tr>
<tr>
<td>no-quota</td>
<td>The account limits have been reached or exceeded.</td>
</tr>
<tr>
<td>not-available</td>
<td>The required resource is unavailable.</td>
</tr>
<tr>
<td>not-secure</td>
<td>You must use SSL to call this action.</td>
</tr>
<tr>
<td>pending-activation</td>
<td>The account is not yet activated.</td>
</tr>
<tr>
<td>pending-license</td>
<td>The account's license agreement has not been settled.</td>
</tr>
<tr>
<td>sco-expired</td>
<td>The course or tracking content has expired.</td>
</tr>
<tr>
<td>sco-not-started</td>
<td>The meeting or course has not started.</td>
</tr>
</tbody>
</table>

The invalid element
An element that gives information describing a status code of invalid.

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invalid</td>
<td></td>
<td>Empty, with attributes</td>
<td>Information about why the call was invalid.</td>
</tr>
<tr>
<td></td>
<td>field</td>
<td>String</td>
<td>The name of the request parameter that was incorrect or missing.</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>Allowed value</td>
<td>The type of the incorrect or missing field.</td>
</tr>
<tr>
<td></td>
<td>subcode</td>
<td>Allowed value</td>
<td>A code explaining why the request was invalid (see the following table for values).</td>
</tr>
</tbody>
</table>

The following table shows the allowed values for subcode when code is invalid.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duplicate</td>
<td>The call attempted to add a duplicate item in a context where uniqueness is required.</td>
</tr>
<tr>
<td>format</td>
<td>A passed parameter had the wrong format.</td>
</tr>
<tr>
<td>illegal-operation</td>
<td>The requested operation violates integrity rules (for example, moving a folder into itself).</td>
</tr>
<tr>
<td>missing</td>
<td>A required parameter is missing.</td>
</tr>
<tr>
<td>no-such-item</td>
<td>The requested information does not exist.</td>
</tr>
<tr>
<td>range</td>
<td>The value is outside the permitted range of values.</td>
</tr>
</tbody>
</table>

Returned by
All actions in the Acrobat Connect Pro Web Services XML API.

Samples
This is a successful response:
<status code="ok" />

This is an invalid response:
<status code="invalid">
<invalid field="principal-id" type="id" subcode="missing" />
</status>

status attribute

Description
An attribute that describes a user's progress with a course or curriculum SCO. It is returned by actions that provide training reports.

A curriculum or folder can only have a status of completed or incomplete.

The following table shows the allowed values for status when returned in a row element describing a course.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-passed</td>
<td>The SCO has scored interactions the user has passed.</td>
</tr>
<tr>
<td>user-failed</td>
<td>The SCO has scored interactions. The user has answered them, but failed.</td>
</tr>
<tr>
<td>completed</td>
<td>The user has viewed all of the SCO's content, but the content has no scored interactions.</td>
</tr>
<tr>
<td>incomplete</td>
<td>The user has not viewed all of the SCO's content.</td>
</tr>
<tr>
<td>not-attempted</td>
<td>The user has not started viewing all of the SCO's content.</td>
</tr>
<tr>
<td>review</td>
<td>The user has passed or completed the course or used all available retries.</td>
</tr>
</tbody>
</table>

Sample

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <report-user-training-transcripts>
    <row transcript-id="2006905612" sco-id="2006298431" principal-id="2006258745" status="review" score="0" max-score="" certificate="" type="content" icon="course">
      <name>Test Course</name>
      <login>joy@acme.com</login>
      <date-taken>2006-06-30T15:23:55.070-07:00</date-taken>
      <principal-name>Joy Smith</principal-name>
    </row>
  </report-user-training-transcripts>
</results>
```

time-zone-id

Description
Settings that describe time zones that you can use with time-zone-id.
Values

<table>
<thead>
<tr>
<th>Time zone setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(GMT-12:00) International Date Line West</td>
<td>0</td>
</tr>
<tr>
<td>(GMT-11:00) Midway Island, Samoa</td>
<td>1</td>
</tr>
<tr>
<td>(GMT-10:00) Hawaii</td>
<td>2</td>
</tr>
<tr>
<td>(GMT-09:00) Alaska</td>
<td>3</td>
</tr>
<tr>
<td>(GMT-08:00) Pacific Time (US and Canada); Tijuana</td>
<td>4</td>
</tr>
<tr>
<td>(GMT-07:00) Mountain Time (US and Canada)</td>
<td>10</td>
</tr>
<tr>
<td>(GMT-07:00) Chihuahua, La Paz, Mazatlan</td>
<td>13</td>
</tr>
<tr>
<td>(GMT-07:00) Arizona</td>
<td>15</td>
</tr>
<tr>
<td>(GMT-06:00) Central Time (US and Canada)</td>
<td>20</td>
</tr>
<tr>
<td>(GMT-06:00) Saskatchewan</td>
<td>25</td>
</tr>
<tr>
<td>(GMT-06:00) Guadalajara, Mexico City, Monterrey</td>
<td>30</td>
</tr>
<tr>
<td>(GMT-06:00) Central America</td>
<td>33</td>
</tr>
<tr>
<td>(GMT-05:00) Eastern Time (US and Canada)</td>
<td>35</td>
</tr>
<tr>
<td>(GMT-05:00) Indiana (East)</td>
<td>40</td>
</tr>
<tr>
<td>(GMT-05:00) Bogota, Lima, Quito</td>
<td>45</td>
</tr>
<tr>
<td>(GMT-04:00) Atlantic Time (Canada)</td>
<td>50</td>
</tr>
<tr>
<td>(GMT-04:00) Caracas, La Paz</td>
<td>55</td>
</tr>
<tr>
<td>(GMT-04:00) Santiago</td>
<td>56</td>
</tr>
<tr>
<td>(GMT-03:30) Newfoundland</td>
<td>60</td>
</tr>
<tr>
<td>(GMT-03:00) Brasilia</td>
<td>65</td>
</tr>
<tr>
<td>(GMT-03:00) Buenos Aires, Georgetown</td>
<td>70</td>
</tr>
<tr>
<td>(GMT-03:00) Greenland</td>
<td>73</td>
</tr>
<tr>
<td>(GMT-02:00) Mid-Atlantic</td>
<td>75</td>
</tr>
<tr>
<td>(GMT-01:00) Azores</td>
<td>80</td>
</tr>
<tr>
<td>(GMT-01:00) Cape Verde Islands</td>
<td>83</td>
</tr>
<tr>
<td>(GMT) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London</td>
<td>85</td>
</tr>
<tr>
<td>(GMT) Casablanca, Monrovia</td>
<td>90</td>
</tr>
<tr>
<td>(GMT+01:00) Belgrade, Bratislava, Budapest, Ljubljana, Prague</td>
<td>95</td>
</tr>
<tr>
<td>(GMT+01:00) Sarajevo, Skopje, Warsaw, Zagreb</td>
<td>100</td>
</tr>
<tr>
<td>(GMT+01:00) Brussels, Copenhagen, Madrid, Paris</td>
<td>105</td>
</tr>
<tr>
<td>(GMT+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna</td>
<td>110</td>
</tr>
<tr>
<td>(GMT+01:00) West Central Africa</td>
<td>113</td>
</tr>
<tr>
<td>(GMT+02:00) Bucharest</td>
<td>115</td>
</tr>
<tr>
<td>Time zone setting</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>(GMT+02:00) Cairo</td>
<td>120</td>
</tr>
<tr>
<td>(GMT+02:00) Helsinki, Kiev, Riga, Sofia, Tallinn, Vilnius</td>
<td>125</td>
</tr>
<tr>
<td>(GMT+02:00) Athens, Istanbul, Minsk</td>
<td>130</td>
</tr>
<tr>
<td>(GMT+02:00) Jerusalem</td>
<td>135</td>
</tr>
<tr>
<td>(GMT+02:00) Harare, Pretoria</td>
<td>140</td>
</tr>
<tr>
<td>(GMT+03:00) Moscow, St. Petersburg, Volgograd</td>
<td>145</td>
</tr>
<tr>
<td>(GMT+03:00) Kuwait, Riyadh</td>
<td>150</td>
</tr>
<tr>
<td>(GMT+03:00) Nairobi</td>
<td>155</td>
</tr>
<tr>
<td>(GMT+03:00) Baghdad</td>
<td>158</td>
</tr>
<tr>
<td>(GMT+03:30) Tehran</td>
<td>160</td>
</tr>
<tr>
<td>(GMT+04:00) Abu Dhabi, Muscat</td>
<td>165</td>
</tr>
<tr>
<td>(GMT+04:00) Baku, Tbilisi, Yerevan</td>
<td>170</td>
</tr>
<tr>
<td>(GMT+04:30) Kabul</td>
<td>175</td>
</tr>
<tr>
<td>(GMT+05:00) Ekaterinburg</td>
<td>180</td>
</tr>
<tr>
<td>(GMT+05:00) Islamabad, Karachi, Tashkent</td>
<td>185</td>
</tr>
<tr>
<td>(GMT+05:30) Chennai, Kolata, Mumbai, New Delhi</td>
<td>190</td>
</tr>
<tr>
<td>(GMT+05:45) Kathmandu</td>
<td>193</td>
</tr>
<tr>
<td>(GMT+06:00) Astana, Dhaka</td>
<td>195</td>
</tr>
<tr>
<td>(GMT+06:00) Sri Jayawardenepura</td>
<td>200</td>
</tr>
<tr>
<td>(GMT+06:00) Almaty, Novosibirsk</td>
<td>201</td>
</tr>
<tr>
<td>(GMT+06:30) Rangoon</td>
<td>203</td>
</tr>
<tr>
<td>(GMT+07:00) Bangkok, Hanoi, Jakarta</td>
<td>205</td>
</tr>
<tr>
<td>(GMT+07:00) Krasnoyarsk</td>
<td>207</td>
</tr>
<tr>
<td>(GMT+08:00) Beijing, Chongqing, Hong Kong SAR, Urumqi</td>
<td>210</td>
</tr>
<tr>
<td>(GMT+08:00) Kuala Lumpur, Singapore</td>
<td>215</td>
</tr>
<tr>
<td>(GMT+08:00) Taipei</td>
<td>220</td>
</tr>
<tr>
<td>(GMT+08:00) Perth</td>
<td>225</td>
</tr>
<tr>
<td>(GMT+08:00) Irkutsk, Ulaan Bataar</td>
<td>227</td>
</tr>
<tr>
<td>(GMT+09:00) Seoul</td>
<td>230</td>
</tr>
<tr>
<td>(GMT+09:00) Osaka, Sapporo, Tokyo</td>
<td>235</td>
</tr>
<tr>
<td>(GMT+09:00) Yakutsk</td>
<td>240</td>
</tr>
<tr>
<td>(GMT+09:30) Darwin</td>
<td>245</td>
</tr>
<tr>
<td>(GMT+09:30) Adelaide</td>
<td>250</td>
</tr>
</tbody>
</table>
**type**

**Description**
A return element or attribute defining the type of a SCO or principal on the server. The allowed values for `type` are different for SCOs and principals.

**SCO types**
A SCO can be content, an Adobe Acrobat Connect Pro meeting, an event, a course, a curriculum, a folder or tree, or any other object on Acrobat Connect Pro. All Acrobat Connect Pro objects are SCOs. In general, a SCO can have any of the following values for `type`:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>A viewable file uploaded to the server, for example, an FLV file, an HTML file, an image, a pod, and so on.</td>
</tr>
<tr>
<td>course</td>
<td>A course, part of a curriculum.</td>
</tr>
<tr>
<td>curriculum</td>
<td>A curriculum.</td>
</tr>
<tr>
<td>event</td>
<td>A event.</td>
</tr>
<tr>
<td>folder</td>
<td>A folder on the server's hard disk that contains content.</td>
</tr>
<tr>
<td>link</td>
<td>A reference to another SCO. These links are used by curriculums to link to other SCOs. When content is added to a curriculum, a link is created from the curriculum to the content.</td>
</tr>
<tr>
<td>meeting</td>
<td>An Acrobat Connect Pro meeting.</td>
</tr>
<tr>
<td>session</td>
<td>One occurrence of a recurring Acrobat Connect Pro meeting.</td>
</tr>
<tr>
<td>tree</td>
<td>The root of a folder hierarchy. A tree’s root is treated as an independent hierarchy; you can’t determine the parent folder of a tree from inside the tree.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time zone setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(GMT+10:00) Canberra, Melbourne, Sydney</td>
<td>255</td>
</tr>
<tr>
<td>(GMT+10:00) Brisbane</td>
<td>260</td>
</tr>
<tr>
<td>(GMT+10:00) Hobart</td>
<td>265</td>
</tr>
<tr>
<td>(GMT+10:00) Vladivostok</td>
<td>270</td>
</tr>
<tr>
<td>(GMT+10:00) Guam, Port Moresby</td>
<td>275</td>
</tr>
<tr>
<td>(GMT+11:00) Magadan, Solomon Islands, New Caledonia</td>
<td>280</td>
</tr>
<tr>
<td>(GMT+12:00) Fiji Islands, Kamchatka, Marshall Islands</td>
<td>285</td>
</tr>
<tr>
<td>(GMT+12:00) Auckland, Wellington</td>
<td>290</td>
</tr>
<tr>
<td>(GMT+13:00) Nuku’alofa</td>
<td>300</td>
</tr>
</tbody>
</table>
However, content objects returned by some actions (for example, `report-bulk-objects`) have the `type` values shown in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive</td>
<td>An archived copy of a live Acrobat Connect Pro meeting or presentation.</td>
</tr>
<tr>
<td>attachment</td>
<td>A piece of content uploaded as an attachment.</td>
</tr>
<tr>
<td>authorware</td>
<td>A piece of multimedia content created with Macromedia Authorware from Adobe.</td>
</tr>
<tr>
<td>captivate</td>
<td>A demo or movie authored in Adobe Captivate.</td>
</tr>
<tr>
<td>course</td>
<td>A course.</td>
</tr>
<tr>
<td>curriculum</td>
<td>A curriculum, including courses, presentations, and other content.</td>
</tr>
<tr>
<td>external-event</td>
<td>An external training that can be added to a curriculum.</td>
</tr>
<tr>
<td>flv</td>
<td>A media file in the FLV file format.</td>
</tr>
<tr>
<td>image</td>
<td>An image, for example, in GIF or JPEG format.</td>
</tr>
<tr>
<td>meeting</td>
<td>An Acrobat Connect Pro meeting.</td>
</tr>
<tr>
<td>presentation</td>
<td>A presentation.</td>
</tr>
<tr>
<td>swf</td>
<td>A SWF file.</td>
</tr>
</tbody>
</table>

**Principal types**

For principals, the allowed values for `type` are shown in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>admins</td>
<td>The built-in group Administrators, for Acrobat Connect Pro server Administrators.</td>
</tr>
<tr>
<td>authors</td>
<td>The built-in group Authors, for authors.</td>
</tr>
<tr>
<td>course-admins</td>
<td>The built-in group Training Managers, for training managers.</td>
</tr>
<tr>
<td>event-admins</td>
<td>The built-in group Event Managers, for anyone who can create an Acrobat Connect Pro meeting.</td>
</tr>
<tr>
<td>event-group</td>
<td>The group of users invited to an event.</td>
</tr>
<tr>
<td>everyone</td>
<td>All Acrobat Connect Pro users.</td>
</tr>
<tr>
<td>external-group</td>
<td>A group authenticated from an external network.</td>
</tr>
<tr>
<td>external-user</td>
<td>A user authenticated from an external network.</td>
</tr>
<tr>
<td>group</td>
<td>A group that a user or Administrator creates.</td>
</tr>
<tr>
<td>guest</td>
<td>A non-registered user who enters an Acrobat Connect Pro meeting room.</td>
</tr>
<tr>
<td>learners</td>
<td>The built-in group learners, for users who take courses.</td>
</tr>
<tr>
<td>live-admins</td>
<td>The built-in group Meeting Hosts, for Acrobat Connect Pro meeting hosts.</td>
</tr>
<tr>
<td>seminar-admins</td>
<td>The built-in group Seminar Hosts, for seminar hosts.</td>
</tr>
<tr>
<td>user</td>
<td>A registered user on the server.</td>
</tr>
</tbody>
</table>
Custom field types

When used with a custom field, `type` can have any of the following values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>required</td>
<td>A required custom field for the account.</td>
</tr>
<tr>
<td>optional</td>
<td>An optional field that is displayed during self-registration.</td>
</tr>
<tr>
<td>optional-no-self-reg</td>
<td>An optional field that is hidden during self-registration.</td>
</tr>
</tbody>
</table>
Chapter 9: A sample application

This guide includes a sample application, firstapp, which is available in the documentation/samples folder on the product CD. The sample application is written in Java and JSP using a model-view-controller architecture and runs on any web application server with a J2EE servlet container. The sample demonstrates how to implement Acrobat Connect Pro meeting functionality in a Java custom application or portal, showing how to log in a user, list a user's meetings, and create, update, and delete meetings.

The sample application code is available for you to reuse, modify, or extend. All sample files referenced in this appendix are within the firstapp.zip sample file. You will find instructions for installing and running the sample application in its ReadMe file.

As you build and design your application, there are several points about the Adobe Acrobat Connect Pro Web Services XML API you should keep in mind:

**Sequence of API calls** Calls to the XML API often need to be made in a specific sequence. For example, you need to get the `principal-id` of a user and the `sco-id` of a meeting before you call `permissions-update` to make the user a meeting presenter. Call sequences for various tasks are described in the first chapters of this guide.

**Different parameter names for the same value** A value might have one parameter name in one call and a different parameter name in another call. For example, the unique ID of a SCO might be a `sco-id` when used with `sco-info`, but an `acl-id` in `permissions-update`. It’s the same value in both calls. The best way to learn this is to use the API reference in this guide.

**SCOs are not object-oriented** A SCO is a shareable content object on the server (for a complete definition, see Find SCOs). A SCO can be a meeting, presentation, course, image, folder, or any object on the server. SCOs are stored within folders in a navigation hierarchy. However, there is no object-oriented structure for SCOs, and one type of SCO is not a subclass of another type. Keep this in mind as you design your application.

Build an adapter class

*Note: See the sample files XMLApiAdapter.java, login.jsp, and mymeetings.jsp*

When you are building a custom application to call Acrobat Connect Pro, it’s very handy to have an adapter class. You create an instance of the class for each user login session, and the adapter handles connecting to the server, logging the user in, making requests to the XML API, and parsing XML responses.

Write constructors for the adapter class

The following constructor (from the sample application file XMLApiAdapter.java) creates an instance of the adapter class to represent a user accessing Acrobat Connect Pro. This is the constructor to use when you already have the `BREEZESESSION` cookie value (see Log in from an application). The constructor also calls the `createXPaths` method to create valid XPath instances to use in other methods:

```java
public XMLApiAdapter(String baseUrl, String breezesession) throws XMLApiException {
    this.setBaseUrl(baseUrl);
    this.breezesession = breezesession;
    createXPaths();
}
```

The second constructor takes a user’s login ID and password, as well as a `BREEZESESSION` cookie value:
public XMLApiAdapter(String baseUrl, String login, String password,
        String breezesession) throws XMLApiException {
    this(baseUrl, breezesession);
    this.setLogin(login);
    this.setPassword(password);
}

You can get the BREEZESESSION cookie value before the user logs in by calling common-info.

Create an instance of the adapter

The following code (from mymeetings.jsp) creates an instance of the XMLApiAdapter class to represent a user who is logged in to Acrobat Connect Pro. The current value of breezesession, which holds the BREEZESESSION cookie value, is then stored in the JSP session attribute for other files to access.

<%! XMLApiAdapter breeze = null; %>
<%...
breeze = new XMLApiAdapter(breezeBase, login, password, breezesession);
breeze.getBreezesession();
session.setAttribute("breezesession", breeze);
...%>

Log the user in

Note: See the sample files XMLApiAdapter.java and login.jsp.

Your application needs a method that logs users in to Acrobat Connect Pro. A login method needs to open a connection to the server, call the login action, and get the XML response. The method also needs to read the value of the BREEZESESSION cookie from the response header and store the value.

The simplest form of the login action is:

https://example.com/api/xml?action=login&login=joy@example.com
&password=jazz

You might also need to add session, account-id, or domain parameters to the login action (see “Log in from an application” for more ways to call login).

A successful login returns this response, with a status code of ok:

<?xml version="1.0" encoding="utf-8" ?>
<results>
<status code="ok" />
</results>

Build the base request URL

The login method should first build the base request URL to send to the server. In the sample, the breezeUrl method builds a URL like this:

http://example.com/api/xml?action=

The method also adds an action name and query string that you pass to it. This is the full method:

protected URL breezeUrl(String action, String queryString)
    throws MalformedURLException {
    return new URL(getBaseUrl() + "\api/xml?" + "action=" + action
Send the user's login information

The login method calls the login action, opens the connection to the server, reads the BREEZESESSION cookie from the response header, and then checks for a status code of ok in the response:

```java
protected void login() throws XMLApiException {
    try {
        if (breezesession != null)
            logout();

        URL loginUrl = breezeUrl("login", "login=" + getLogin()
                        + "&password=" + getPassword());

        URLConnection conn = loginUrl.openConnection();
        conn.connect();

        InputStream resultStream = conn.getInputStream();
        Document doc = new SAXBuilder(false).build(resultStream);

        String breezesessionString = (String) (conn
                        .getHeaderField("Set-Cookie"));

        StringTokenizer st = new StringTokenizer(breezesessionString, "=");
        String sessionName = null;
        if (st.countTokens() > 1)
            sessionName = st.nextToken();

        String breezesessionNext = st.nextToken();
        int semiIndex = breezesessionNext.indexOf(';');
        breezesession = breezesessionNext.substring(0, semiIndex);

        Element root = doc.getRootElement();
        String status = getStatus(root);
        if (breezesession == null || !"ok".equalsIgnoreCase(status))
            throw new XMLApiException("Could not log into Acrobat Connect Pro.");
    } catch (IOException ioe) {
        throw new XMLApiException(IO_ERROR, ioe);
    } catch (JDOMException jde) {
        throw new XMLApiException(PARSE_ERROR, jde);
    }
}
```

Send XML requests

**Note:** See the sample files `XMLApiAdapter.java` and `createmeting.jsp`.

Once a user is logged in, it’s useful to have a generic request method that sends a request to the server when you provide an action name and query string.

The `request` method in the sample takes an action and a query string and sends the BREEZESESSION cookie value back to the server in the request header:
protected Element request(String action, String queryString) 
    throws XMLApiException {
    try {
        if (breezesession == null)
            login();
        URL url = breezeUrl(action, queryString);
        URLConnection conn = url.openConnection();
        conn.setRequestProperty("Cookie", "BREEZESESSION=" + breezesession);
        conn.connect();
        InputStream resultStream = conn.getInputStream();
        Document doc = new SAXBuilder(false).build(resultStream);
        return doc.getRootElement();
    } catch (IOException ioe) {
        throw new XMLApiException("A communication error occurred", ioe);
    } catch (JDOMException jde) {
        throw new XMLApiException("A parsing error occurred", jde);
    }
}

Parse XML responses

Note: See the sample file XMLApiAdapter.java.

When you send an XML request to Acrobat Connect Pro, the server returns an XML response. You need to parse the response and extract values, including status codes. One way to parse the response is to use XPath to traverse XML elements (see the XPath tutorial at w3schools.com for more information).

As an example, this is the response from sco-shortcuts:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <shortcuts>
    <sco tree-id="4930295" sco-id="2006258748" type="my-courses">
      <domain-name>http://example.com</domain-name>
    </sco>
    <sco tree-id="4930293" sco-id="2006258749" type="my-events">
      <domain-name>http://example.com</domain-name>
    </sco>
    ...
  </shortcuts>
</results>
```

Extract values

The getShortcuts method calls sco-shortcuts and parses the response using XPath. This is an example of how to extract a list of sco elements and the sco-id of each:

```java
public List getShortcuts() throws XMLApiException {
    try {
        Element e = request("sco-shortcuts", null);
        List scosXml = XPath.selectNodes(e, "/sco");
        List scos = new ArrayList();
        for (Iterator i = scosXml.iterator(); i.hasNext()) {
            sco = (Sco) i.next();
            scos.add(sco);
        }
    } catch (IOException ioe) {
        throw new XMLApiException("A communication error occurred", ioe);
    } catch (JDOMException jde) {
        throw new XMLApiException("A parsing error occurred", jde);
    }
    return scos;
}
```
Element s = (Element) i.next();
SCO sco = getSco(id.valueOf(s));
    if(null != sco)
        scos.add(sco);
    }
return scos;
} catch (JDOMException jde) {
    throw new XMLApiException(PARSE_ERROR, jde);
}

Extract a status code
Your application also needs to parse both successful and unsuccessful responses for status codes. For example, when a user calls an action without sufficient permission, the error response has a status element with both code and subcode attributes:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="no-access" subcode="denied" />
</results>
```

These lines in the createXPaths method parse for the values of code and subcode:

codePath = XPath.newInstance("//status/@code");
subcodePath = XPath.newInstance("//status/@subcode");

In the sample, the createXPaths method is called when you create an instance of XMLApiAdapter. The getStatus method then uses codePath and subcodePath to return the code and subcode values, separated by a hyphen:

```java
private String getStatus(Element el) throws JDOMException {
    String code = codePath.valueOf(el);
    String subcode = subcodePath.valueOf(el);
    StringBuffer status = new StringBuffer();
    if(null != code && code.length() > 0)
        status.append(code);
    if(null != subcode && subcode.length() > 0)
        status.append(" - " + subcode);
    return status.toString();
}
```

Display user information

**Note:** See the sample files XMLApiAdapter.java, UserInfo.java, and header.jsp.

In your user interface, you might want to display information about a user, such as a name, during the user's login session.

You can retrieve simple information about the user by calling common-info after the user logs in, like this:

https://example.com/api/xml?action=common-info

The response has a user element with information you can display or store in variables to use later:

```xml
<user user-id="2006258745" type="user">
    <name>Joy Smith</name>
    <login>joy@acme.com</login>
</user>
```

If you call common-info before the user logs in, the response does not have a user element.
Get information about the user

In the sample, the getUserInfo method calls common-info and parses the response for the values of name, login, and user-id. The method then stores information about the user in an instance of the UserInfo class, which is a standard bean class with getter and setter methods.

```java
public UserInfo getUserInfo(String login, String password) throws XMLApiException {
    try {
        Element e = request("common-info", "login=" + login + "+password=" + password);
        XPath name = XPath.newInstance("//user/name");
        XPath log = XPath.newInstance("//user/login");
        XPath id = XPath.newInstance("//user/user-id");

        UserInfo user = new UserInfo();
        user.setLogin(log.valueOf(e));
        user.setName(name.valueOf(e));
        user.setUserId(id.valueOf(e));

        return user;
    } catch (JDOMException jde) {
        throw new XMLApiException(PARSE_ERROR, jde);
    }
}
```

List a user’s meetings

Note:

You may want to list a user’s meetings in your application. You can choose the meetings to list and the information to display based on your application design. This illustration shows one example of a meeting list:

<table>
<thead>
<tr>
<th>Scheduled Meetings</th>
<th>Start Time</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>October Company Meeting</td>
<td>09:00 AM</td>
<td>Host</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expired meetings</th>
<th>Start Time</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN - Monday Night Football</td>
<td>03:30 PM</td>
<td>Participant</td>
</tr>
<tr>
<td>seminar invite</td>
<td>12:00 PM</td>
<td>Host</td>
</tr>
<tr>
<td>How to Write a Novel</td>
<td>10:00 PM</td>
<td>Presenter</td>
</tr>
<tr>
<td>September All Hands Meeting</td>
<td>09:00 AM</td>
<td>Host</td>
</tr>
<tr>
<td>Platinum Support Team Meeting</td>
<td>11:45 AM</td>
<td>Participant</td>
</tr>
</tbody>
</table>

To list a user’s meetings using the XML API, call report-my-meetings with or without a filter. Without a filter, report-my-meetings returns all of a user’s meetings:

https://example.com/api/xml?action=report-my-meetings
You can add `filter-expired=false` to return only meetings that are currently in progress and scheduled in the future:

https://example.com/api/xml?action=report-my-meetings&filter-expired=false

Even with a filter, the response is likely to have multiple meeting elements that you need to iterate through and extract data from. A meeting element looks like this:

```html
<meeting sco-id="2007063179" type="meeting" icon="meeting"
        permission-id="host" active-participants="0">
    <name>September All Hands Meeting</name>
    <description>For all company employees</description>
    <domain-name>example.com</domain-name>
    <url-path>/sept15/</url-path>
    <date-begin>2006-09-15T09:00:00.000-07:00</date-begin>
    <date-end>2006-09-15T18:00:00.000-07:00</date-end>
    <expired>false</expired>
    <duration>09:00:00.000</duration>
</meeting>
```

Get the meeting list

To get the meeting list in Java, write a method like `getMyMeetings`, which lists a user's meetings with a filter as an argument. If you don't want to use a filter, you can pass null as the filter argument. A meeting is a SCO, so `getMyMeetings` calls the `getSco` method to extract values from the response and store them in an instance of `SCO.java`.

```java
public List getMyMeetings(String filter) throws XMLApiException {
try {
    Element meetingDoc = request("report-my-meetings", filter);
    List list = XPath.selectNodes(meetingDoc, "//meeting");
    Iterator meetings = list.iterator();
    List result = new ArrayList();
    while (meetings.hasNext()) {
        Element m = (Element) meetings.next();
        SCO meeting = getSco(m);
        result.add(meeting);
    }
    return result;
} catch (JDOMException jde) {
    throw new XMLApiException(PARSE_ERROR, jde);
}
}
```

The `SCO` object encapsulates data about the SCO so you can easily retrieve it from a web page (for example, HTML or JSP) in your application.

Create and update meetings

Note: See the sample files `XMLApiAdapter.java` and `SCO.java`.

You might also want to allow users to create meetings in your application. To create a meeting, call `sco-update` with the `folder-id` of a meetings folder and `type=meeting`:

https://example.com/api/xml?action=sco-update
&folder-id=2006258750&description=For all company employees
&name=Company All Hands Meeting&type=meeting&lang=en
&date-begin=2006-06-16T23:00&date-end=2006-06-16T23:30
The response returns the `sco-id` of the meeting, which you can extract and store:

```xml
<sco account-id="624520" disabled="" display-seq="0" folder-id="2006258750" icon="meeting" lang="en" max-retries="" sco-id="2006743452"
source-sco-id="-1625529" type="meeting" version="1">
</sco>
```

The difference between calling `sco-update` to create or update a meeting is:

- Pass a `folder-id` to create a meeting.
- Pass a `sco-id` to update an existing meeting. A meeting only has a `sco-id` if it already exists.

**Create a meeting**

In an application, you first need to collect from the user the information needed to create or update the meeting, such as the meeting name, date, time, and so on. With that information, use a method such as `updateSco` that calls the `sco-update` action.

In `sco-update`, be sure to set the type of the SCO to `meeting`. As an option, you can also set a language code for the meeting room, such as `lang=en`, for example:

```
https://example.com/api/xml?action=sco-update&folder-id=2006258750
&description=nov&name=Nov%20All%20Hands%20Meeting&type=meeting&lang=en
&date-begin=2006-11-11T09:00&date-end=2006-11-11T17:00
```

The `updateSco` method shows how to implement the `sco-update` call in Java, once you collect information about the meeting from the user:

```java
public String updateSco(String action, SCO sco) throws XMLApiException {
    try {
        StringBuffer sb = new StringBuffer();
        Map data = sco.getUpdateFields();

        if (CREATE.equals(action))
            sb.append("folder-id=" + sco.getFolderId());
        else
            sb.append("sco-id=" + sco.getId());

        Iterator iter = data.keySet().iterator();
        while (iter.hasNext()) {
            String key = (String) iter.next();
            if (key.indexOf("sco-id") != -1)
                continue;
            if (key.indexOf("folder-id") != -1)
                continue;
            String value = (String) data.get(key);
            sb.append("&" + key + "+" + value);
        }

        if (null == data.get("type"))
            throw new XMLApiException("SCO type not defined");

        Element e = request("sco-update", sb.toString());
        XPath scoId = XPath.newInstance("//results/sco/@sco-id");
        if (scoId.valueOf(e) == null)
            return null;
        else
            return scoId.valueOf(e);
    }
    catch (JDOMException jde) {
        throw new XMLApiException(PARSE_ERROR, jde);
    }
    catch (ParseException pe) {
        throw new XMLApiException(PARSE_ERROR, pe);
    }
}
```
Set meeting access

Once you have a sco-id, a meeting needs access. The user who creates the meeting is the host by default and chooses whether the meeting is public or private, set by a combination of permission-id and principal-id in permissions-update. For example, this call makes a meeting public:

https://example.com/api/xml?action=permissions-update&acl-id=2006334033&principal-id=public-access&permission-id=view-hidden

Once a user chooses these values, the setPermissions method calls permissions-update to set the meeting access:

```java
public void setPermissions(String aclId, String principalId, String permissionId) throws XMLApiException {
    request("permissions-update", "acl-id=" + aclId + "&principal-id=" + principalId + "&permission-id=" + permissionId);
}
```

Display meeting detail

**Note:** See the sample files XMLApiAdapter.java, SCO.java, mymeetings.jsp, and showmeeting.jsp.

Most of the information you want to display about a meeting comes from sco-info:

https://example.com/api/xml?action=sco-info&sco-id=2006334909

The response has many values that you can display, for example:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
    <status code="ok" />
    <sco account-id="624520" disabled="" display-seq="0"
        folder-id="2006258747" icon="producer" lang="en" max-retries=""
        sco-id="2006334909" source-sco-id="" type="content" version="1">
        <date-created>2006-05-11T12:00:02.000-07:00</date-created>
        <date-modified>2006-05-16T15:22:25.703-07:00</date-modified>
        <name>Test Quiz</name>
        <url-path>/quiz/</url-path>
        <passing-score>10</passing-score>
        <duration>15100.0</duration>
        <section-count>6</section-count>
    </sco>
</results>
```

Get information about a SCO

The getSco Java method makes the call to sco-info and parses the result, storing values in variables so that you can display them in a user interface:

```java
public SCO getSco(String scoId) throws XMLApiException {
    try {
        Element e = scoInfo(scoId);
        if(!"ok".equalsIgnoreCase(codePath.valueOf(e)))
            return null;
        Element sco = (Element) XPath.selectSingleNode(e, "/sco");
        ...
```
Construct the URL to the meeting room

You also need to create the URL to the meeting room. You can do this with a call to `sco-info` and another to `sco.shortcuts`:

https://example.com/api/xml?action=sco-info&sco-id=2006258750
https://example.com/api/xml?action=sco-shortcuts

Extract the `url-path` from the `sco-info` response. Then, extract the `domain-name` from the `sco-shortcuts` response and concatenate the two values:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<results>
  <status code="ok" />
  <shortcuts>
    <sco tree-id="4930295" sco-id="2006258748" type="my-courses">
      <domain-name>http://example.com</domain-name>
    </sco>
  ..
</results>
```

You can also use a single call to `report-my-meetings` if the user is logged in and the meeting is in the user's `my-meetings` folder:

https://example.com/api/xml?action=report-my-meetings

In this case, extract both the `domain-name` and `url-path` from the `report-my-meetings` response.

The `scoUrl` Java method constructs the URL by calling `sco-info` to retrieve the `url-path` and then `sco-shortcuts` to retrieve the `domain-name`. In this case, two calls are used so that you do not need to make the assumption that the meeting is in the current user's `my-meetings` folder:

```java
public String scoUrl(String scoId) throws XMLApiException {
    try {
        Element e = request("sco-info", "sco-id=" + scoId);
        if(!codePath.valueOf(e).equalsIgnoreCase("ok"))
            return "";
        XPath xpath = XPath.newInstance("//url-path/text()");
        String path = ((Text) xpath.selectSingleNode(e)).getText();
        e = request("sco-shortcuts", null);
        xpath = XPath.newInstance("//domain-name/text()");
        String url = ((Text) xpath.selectSingleNode(e)).getText();

        return url + "+" + path.substring(1) + "?session=" + breezesession;
    } catch (JDOMException jde) {
        throw new XMLApiException(PARSE_ERROR, jde);
    }
}
```
Index

A
access attribute 204
access control list (ACL) 27
account ID 11
account-expiry-info 55
acl-field-info 56
acl-field-list 57
acl-field-update 59
acl-preference-update 60
Acrobat Connect Pro resources 2
Acrobat Connect Pro Web Services 4
actions
acl-field-info 56
acl-field-list 57
acl-field-update 59
acl-preference-update 60
common-info 7, 62, 223, 224
custom-field-update 67
custom-fields 64
custom-fields-delete 66
group-membership-update 44, 72
learning-path-info 73
learning-path-update 75
login 81, 221
logout 17, 83
new or removed 54
permissions-info 87
permissions-reset 90
permissions-update 91
principal-info 92
principal-list 95
principal-list-by-field 98
principal-update 20, 104
principals-delete 103
report-active-meetings 112
report-bulk-consolidated-transactions 29, 114
report-bulk-objects 116
report-bulk-questions 118
report-bulk-slide-views 120
report-bulk-users 122
report-course-status 124
report-curriculum-taker 44, 126
report-meeting-attendance 129
report-meeting-concurrent-users 131
report-meeting-sessions 132
report-meeting-summary 134
report-my-courses 136
report-my-events 138
report-my-meetings 29, 140, 225
report-my-training 44, 142
report-quiz-interactions 29, 144
report-quiz-question-answer-distribution 147
report-quiz-question-distribution 152
report-quiz-question-response 154
report-quiz-summary 155
report-quiz-takers 157
report-quotas 159
report-sco-slides 161
report-sco-views 163
report-user-training-transcripts 44, 167
report-user-trainings-taken 44, 164
sco-contents 169
sco-delete 172
sco-expanded-contents 173
sco-info 176, 227
sco-move 178
sco-nav 180
sco-search 181
sco-shortcuts 25, 186, 222
sco-update 29, 189, 226
sco-upload 192
user-accounts 194
user-transcript-update 195
user-update-pwd 196
adapter class
building 219
constructors 219
creating an instance 220
Adobe Acrobat Connect Pro resources 2
Adobe Acrobat Connect Pro Web Services 4
Adobe Developer Center 1
Adobe resources, downloading 1
API calls
making your first 7
sequence 219
application server, web 7
applications
Acrobat Connect Pro 6
custom 2, 6
meeting, recommended calls 29
sample 219
training, recommended calls 44
archives, meeting 38
attributes
access 204
icon 207
lang 208
object-type 208
permission-id 31, 45, 209
status 213
time-zone-id 213
type 216
authentication, external 13, 39
B
BREEZESESSION
in adapter class 219
from common-info 11
with cookie management 10
from common-info 40
invalidating 17
storing in JSP session 220
built-in groups 18
C
calculating meeting usage 36
chat transcripts 41
Adobe Acrobat Connect Pro resources 2
common-info 62
before login 82
browser test 7
Java example 224
compliance settings 40
components 6
Flash Media Server 7
SQL Server database 7
web application server 7
content folder 25
conventions, typographical 3
cookie management 10
courses
defined 44
display all users 51
enrolling group 50
enrolling multiple users 49
enrolling one user 48
find sco-id 45
listing all available 46
listing in curriculum 51
reporting scores 53
unenrolling group 51
unenrolling users 50
view for user 47
courses folder 25, 44
creating
custom fields 21
customized meetings 33
groups 22
meeting room URL 31
meetings 32
new user without e-mail address 20
new users 20
private meetings 33
public meetings 32
users 20
current user, principal-id of 18
curriculums
defined 44
display all users 51
enrolling group 50
enrolling multiple users 49
enrolling one user 48
find sco-id 45
listing all available 46
listing all courses in 51
reporting scores 53
unenrolling group 51
unenrolling users 50
view completed work 52
view for user 47
view remaining work 52
viewing information 51
viewing status 47
custom applications 6
data flow 6
languages 2
custom fields
create 21
field element 206
types 218
for users 22
custom-field-update 67
custom-fields 64
custom-fields-delete 66
data flow
Acrobat Connect Pro applications 6
custom applications 6
diagram 5
encrypted or unencrypted 5
date/time formats 2
deleting principals 103
denied permission 45
developer resources 1
development environment 2
disabling pods 41
disclaimers, setting up 42
displaying
meetings 30
user information 223
downloading
Adobe resources 1
files 27
downloading 27
uploading 192
filters
authentication 13
defined 8
elements 8
sample call 9
syntax 201
finding
meeting sco-id 29
principal-id 18
SCO without sco-id 25
firstapp sample 219
Flash Media Server 7
group-membership-update
defined 72
in training applications 44
groups
adding users 22
built-in 18
checking users in 23
creating 22
enrolling in courses or curriculums 50
finding principal-id 18
list of, for user 24
unenrolling from courses or curriculums 51
guests, listing 19

H
HeaderAuthenticationFilter 13
host, adding 32
HTTP header authentication
login 12

I
icon attribute 207
interactions 145
invitations 34

J
Javascript, opening meeting room
URL 40

L
lang attribute 208
languages and custom applications 2
INDEX

learning-path-info 73
learning-path-update 75
limited-administrator-permissions info 76
limited-administrator-permissions-update 79
links to Acrobat Connect Pro Server 7
resources 2
listing
  all courses or curriculums 46
courses in curriculum 51
guests 19
meeting archives 38
meetings (Java example) 224
principals 19
logging in
  with account ID 11
to Acrobat Connect Pro Server 10
  from an application 10, 220
  basic technique 10
  building request URL 220
  cookie management 10
  with domain name 12
  to hosted account 11
  with session parameter 11
  and updating users 21
login
  action 81
  external authentication 12, 13
  HTTP header authentication 12, 13
  Java example 221
logout
  action 83
  Java example 17
M
manage permission 45
meeting applications, recommended calls 29
meeting archives 38
meeting poll results 38
meeting room URL 31, 228
meeting usage, calculating 36
meeting-disclaimer-info 84
meeting-disclaimer-update 85
meeting-feature-update 85
meetings
  adding host 32
  adding participants 32
adding presenter 32
adding user 31
calculating usage 36
creating 32, 225
creating in Java 226
creating meeting room URL 31, 228
creating with template 33
customized 33
disclaimer notices for 42
displaying 227
displaying for user 30
finding sco-id 29
get poll results 38
invite users 34
launch for guest 40
launch for participant 39
launch with external authentication 39
listing archives 38
listing for user 225
listing in Java 224
prevent double login 40
private 32
public 32
recording 42
remove participants 36
send invitations 34
sessions 132
set permissions 227
time per user 36
update 225
meetings folder 25
Microsoft SQL Server 3, 7
my-content folder 25
my-courses folder 25
my-events folder 25
my-meeting-templates folder 25
my-meetings folder 25
N
new actions 54
O
object-type attribute 208
P
parameters with different names 27, 219
  parsing responses
for errors 16
Java example 222
with XPath 15
participants
  adding 32
  permission-id 209
denied 45
manage 45
for meetings 31
view 45
permissions 27
  check for principal 28
  check for SCO 28
  defined 209
  setting for meetings 227
Training library 44
permissions-info 87
permissions-reset 90
permissions-update 91
in meeting applications 29
in training applications 44
pods, disabling 41
poll results 38
presenters, adding 32
principal-id
  of current user 18
  finding 18
principal-info 92
principal-list
  browser test 8
  defined 95
principal-list-by-field 98
principal-update
  defined 104
  example 20
principals
  check permissions 28
  defined 18
  deleting 103
  finding principal-id 18
  listing all 19
types 217
principals-delete 103
private meetings
  creating 32
  remove users from 36
public meetings
  creating 32
  remove users from 36
Q
quiz scores, reporting 53
quota-threshold-exceeded 108
quota-threshold-info 106
quota-threshold-update 110
R
recording meetings 42
report-active-meetings 112
report-bulk-consolidated-transactions
defined 114
in meeting applications 29
report-bulk-objects 116
report-bulk-questions 118
report-bulk-slide-views 120
report-bulk-users 122
report-course-status 124
report-curriculum-taker
defined 126
in training applications 44
report-meeting-attendance 129
report-meeting-concurrent-users 131
report-meeting-sessions 132
report-meeting-summary 134
report-my-courses 136
report-my-events 138
report-my-meetings
defined 140
Java example 225
in meeting applications 29
report-my-training
defined 142
in training applications 44
report-quiz-interactions
defined 144
in meeting applications 29
report-quiz-question-answer-distribution 147
report-quiz-question-distribution 152
report-quiz-question-response 154
report-quiz-summary 155
report-quiz-takers 157
report-quotas
defined 159
report-sco-slides 161
report-sco-views 163
report-user-training-transcripts
defined 167
in training applications 44
report-user-trainings-taken
defined 164
in training applications 44
reporting quiz scores 53
request URLs
Java example 220
requests in XML document 15
resources
Acrobat Connect Pro 2
Adobe 1
date/time formats 2
Microsoft SQL Server 3
online 2
SCORM 3
web services 2
XML 2
XPath 2
XSLT 2
responses
extract status code 223
extract values 222
parse 222
parse for error 16
parse with XPath 15
status code 17
S
sample application 219
sco-contents 169
sco-delete 172
sco-expanded-contents 173
sco-id
defined 24
finding for course or curriculum 45
finding for meeting 29
sco-info
defined 176
Java example 227
sco-move 178
sco-nav 180
sco-search 181
sco-search-by-field 184
sco-shortcuts 186
example 25
Java example 222
sco-update
defined 189
Java example 226
in meeting applications 29
sco-upload 192
scores, reporting 53
SCORM 3
SCOs
characteristics 24
check permissions 28
create URL 26
defined 18, 24
finding 24
finding without sco-id 25
get information about 26, 227
not object-oriented 219
types 24, 216
seminars folder 25
Server, Flash Media 7
session parameter 11
session, defined 132
share settings, controlling 42
shareable content objects. See SCOs
shared folders 25
Shared Training folder 44
shared-meeting-templates folder 25
sort
defined 8
sample call 9
syntax 202
SQL Server 3, 7
status attribute 47, 213
status code, Java example 17, 223
status element 211
T
technical support 2
time-zone-id attribute 213
training
courses 44
curriculums 44
find courses 45
find curriculums 45
interactions 145
listing all courses or curriculums 46
permissions 44
view for user 47
training applications, recommended calls 44
Training library permissions 44
transactions 36
transcripts, of chat sessions 41
type
  attribute 216
custom fields 218
element 216
principals 217
SCOs 24, 216
typographical conventions 3

U
unencrypted data flow 5
unenrolling users from courses or curriculums 50
updating users 21
uploading files 192
URLs
  create for SCO 26
  meeting room 31, 228
  request 220
usage, meeting 36
user information
  displaying 223
  retrieving 224
user-accounts 194
user-content folder 25
user-courses folder 25
user-events folder 25
user-meetings folder 25
user-transcript-update 195
user-update-pwd 196
users
  adding to groups 22
  adding to meetings 31
  checking group membership 23
  creating 20
  creating without e-mail address 20
  display enrollments 51
  display meetings 30, 31
  enrolling 48
  finding principal-id 18
  invite to meetings 34
  listing all groups 24
  remove from meetings 36
  report highest score 53
  send meeting invitations 34
  unenrolling from courses or curriculums 50
  updating 21
  view curriculum progress 52
  view training 47

V
  view permission 45
  viewing
    course or curriculum status 47
    curriculum information 51

W
  web application server 7
  Web Services 4
  web.xml file, setting authentication filter 13

X
  XML document request 15
  XML requests, Java example 221
  XML resources 2
  XPath
    Java example 15
    resources 2
    to parse responses 15
  XSLT 2

Z
  zip files, downloading 27