



# Integration Best Practices

**Developer Technical Note**

**WebEx Communications Inc.**  
3979 Freedom Circle, Santa Clara, CA 95054, U.S.A.

**Corp.:** +1.408.435.7000 **Sales:** 1.877.509.3239

[www.webex.com](http://www.webex.com)

## Introduction

This developer technical note discusses a variety of best practices and tips for effectively utilizing MediaTone Integration Platform APIs. The guidelines presented will help you avoid common integration problems, improve the flexibility and robustness of your integration, and help make sure your integration proceeds successfully.



# URL API Best Practices

## Security

The security of host account information is extremely important, and there are a number of options to make sure this information does not fall into the wrong hands. Beyond the standard use of HTTPS and POST, WebEx recommends the use of the IP Referrer and Domain Referrer options. Additionally, your Partner ID is a crucial piece of information that you should take measures to protect.

### IP Referrer

Creating a host account via the URL API command (AT=SU) uses a server-to-server connection. To make sure the data passing over the wire is secure (in addition to using HTTPS/POST), you can request the use of IP Referrer on your WebEx site. The IP Referrer option checks the IP address from which any AT=SU command originates, and validates the source. If an AT=SU command does not come from a valid IP address, the request will be denied.

To use the IP Referrer option, you will need to provide WebEx with the IP address of your server, or the IP of the last exit point before the Internet if you use a firewall or proxy configuration. You have the option to provide one IP address or a range of IP addresses. Forward this information and a requests for enabling the IP Referrer option on your site to your WebEx Client Services Manager.

### Domain Referrer

Although the login command (AT=LI) is not a server-to-server command, it does have an additional security option in the Domain Referrer. The Domain Referrer makes sure that no one, even a host with a valid login and password, can use the API to access your site unless they are on your domain. In other words, when you implement the Domain Referrer option, you will deny anyone outside your network the ability to login using the WebEx API. If they know their WebEx ID (WID) and password (PW), they will be able to visit the WebEx site to login through the normal web interface, but will not be able to use the API. Requests for site changes should be directed to the WebEx Client Services Manager.

NOTE: The URL API is supported for use in browser-based applications. WebEx uses a number of methods in its process of operation that do not work well when the browser is not allowed to “do the work.” If Java or similar technologies are used, the WebEx XML API is a better choice depending on the goal of the implementation.

### Partner ID

The Partner ID (PID) is a configurable value assigned to your organization at the time the API is enabled on your site. This parameter is required by the AT=SU command as an identifier that the API command has the authority to create user accounts without logging in as a site administrator. This is a very valuable piece of information that should be protected just as would any other crucial password. Because this is security sensitive information, it is not released to just anyone who requests the information. Requests for this information should be routed through the WebEx Client Services Manager who will make sure the contact receiving the information is approved.



# XML API Best Practices

## Posting XML Requests

All WebEx XML API function requests should be posted using HTTPS. Although HTTP will technically work, this is a significant security risk. Each XML API request requires the WebEx Username and Password be included in the <securityContext> of the request. HTTPS will prevent this sensitive information from being sent over the public Internet in clear text. Note that if you're using Java, JDK 1.4+ is required to post the XML request using HTTPS.

Developers should not hard code the URL of the XML API to which they post requests. Integrations should be configurable to run on different WebEx customer sites which have different domains. In addition, the service directory of the XML API URL is subject to change. For example, WebEx deploys new versions of the XML API to a “/preview” directory before replacing the current version at the regular directory to allow testing of integrations before migrating the production site.

Thus, developers should use two configurable parameters to build the XML API URL to post their request:

```
https://[site_domain]/[xml_directory]
where
```

site\_domain = WebEx customer site, (i.e.:mwapi.webex.com)

xml\_directory = Regular or preview directory: (i.e. WBXService/preview/XMLService or WBXService/XMLService)

## Handling TimeZones

WebEx users can be simultaneously scattered across the globe. Therefore, all scheduled times need to specify a time zone. In the XML API, all WebEx time elements require an accompanying <timeZoneID> code. Many XML functions return a <timeZone> string as well. As with exception handling, developer code should only process the <timeZoneID> codes and never parse the <timeZone> strings.

The strings returned in the <timeZone> elements (such as “GMT-08:00, Pacific (San Jose)”), list the city and hours relative to GMT during Standard Time. These <timeZone> string values are constant. Thus, during Daylight Savings Time, the <timeZone> string hours relative to GMT appear incorrect. For example, for Pacific (San Jose), the string should read “GMT-7:00”, but will still appear as “GMT-8:00”. The WebEx servers automatically adjust user interface displays according to Daylight Savings Time. If you plan to use these strings in your user interfaces, you must handle this conversion logic on your own. A more robust approach is to generate user interface strings from the <timeZoneID>.

## Meeting Types

The meeting.CreateMeeting function requires a <meetingType> value. Developers should not hardcode this value. This is especially true if the integration is meant to run on multiple WebEx customer sites. Meeting Types can vary based on the different types of meetings that each site and user has access to. In addition, WebEx customers with the newer Named-Host pricing model will have different meeting types than other pricing models.



There are two approaches to handling this issue in partner integrations.

1. The <meetingType> value can be a user selectable option. The user.GetUser function returns all the meeting types available for a user. An integration can use this function to dynamically present the user with a choice of available meeting types for the site. Then, the meetingType.GetMeetingType function can be called if required to get detailed information for each meeting type. The meeting type selected by the user is then specified in the meeting.CreateMeeting function.

2. The <meetingType> value can be set to a configurable default value. This approach hides the meeting type issue from the user, simplifying the scheduling operation and eliminating the need to train users on the meanings of each available meeting type. The configuration of the default meeting type should be in a site administration area of the integrated solution.

Lastly, both the URL and XML APIs have optional elements to specify a meeting type when creating or updating a user. Again, partner applications should not hard-code meeting type values in these calls. If left unspecified, the user is created with access to meeting types available on their site as determined by their service and site administrator.

## Processing a Large Volume of Response Records

Many XML API query requests can potentially return hundreds or thousands of response records. For performance reasons, the WebEx XML API caps the maximum number of records that can be returned in a single query. Developer applications should make multiple queries to “step through” the result set until all matching records are retrieved.

All query requests have <listControl>, <startFrom>, and <maximumNum> elements. These elements allow each query to return a fixed subset of the total number response records. Each subsequent query can step through the next subset until all response records are returned. Developer applications should check the <totalRecords> response value and increase the <startFrom> value in each subsequent query until all records are returned.

Here is Java pseudocode that makes multiple requests to eventually retrieve the entire list of WebEx Users for a site:

```
int segment=20;
// get first record subset
call LstSummaryUser with
  <listControl>
    <startFrom>1</startFrom>
    <maximumNum>segment</maximumNum>
  </listControl>;
process LstsummaryUserReponse;

int totalRecords = value for <matchingRecords><total>;

// loop to get additional subsets
for (int n = 1 + segment; n < totalRecords; ++n) {
  call LstSummaryUser with
    <listControl>
      <startFrom>n</startFrom>
      <maximumNum>segment</maximumNum>
    </listControl>;
  process LstsummaryUserReponse;
}
```



## Exception Handling

The WebEx XML API returns `<result>FAULURE</result>` when a request cannot execute successfully. XML API 3.6 and later versions return an `<exception ID>` code along with the exception `<reason>` string.

When coding XML API exception handling, developer integrations should only process the `<exceptionID>` codes and NOT parse the exception reason strings. Exception reason strings are subject to change in future releases. Developers who are currently parsing the XML API exception `<reason>` strings should change their integration code to process the `<exceptionID>` instead.

For a list of Exception ID codes and reasons, refer to The XML API Developer Reference Guide, Appendix C.

## Meeting Passwords

Meeting passwords are a security feature that many WebEx sites require. Developers should allow users to enter a meeting password in the schedule or start meeting page of their integration. For ease of use, once a meeting password is entered, the integration could store that password and have it filled in by default when the user is scheduling future meetings.

Normally attendees have to enter the meeting password to join the session from the WebEx microsite. However, the integration could include the meeting password in the join meeting URL API command that is displayed or sent in a custom email to attendees. This will allow attendees to more easily join the meeting without having to type in the meeting password.

## Meeting Invitations

XML API version 3.6+ offers the option for the WebEx server to email attendees a meeting invitation. For Meeting Center and Training Center, this is controlled by the `create/setMeeting` and `create/setTrainingSession` `<attendeeOptions><emailInvitations>` boolean element. `createMeetingAttendees` `<emailInvitations>` also determines whether to send WebEx email invitations.

For Event Center, `sendInvitationEmail` will send session invitation emails on the spot to attendees and/or presenters.

Many WebEx developers choose to create and email their own meeting invitations to attendees. Developers should refer to the WebEx URL API documentation to create their own join meeting command (`m.php?AT=JM`) to include in their attendee meeting invitation. The WebEx URL API join meeting command includes parameters for custom behavior. This join meeting command can include the `PW=meeting password` parameter so the attendee doesn't have to enter the password to join the meeting. Like most URL API commands, the join meeting command can specify a `BU=back url` parameter to be redirected to after the attendee leaves the meeting.

## Joining a Meeting Before the Host has Started It

The `create/setMeeting`, `create/setTrainingSession`, `create/setEventSession`, and `create/setSalesSession` functions have a `<schedule><openTime>` element that allows attendees to join a session before the host has officially started it. This feature is supported in Meeting Center 6.0+, Training Center 3.0+, Event Center 4.0+, and all versions of Sales Center.



## **Worldwide Sales Offices:**

Americas & Canada

Tel: +1.877.509.3239

[AmericasInfo@webex.com](mailto:AmericasInfo@webex.com)

Europe, Middle East & Africa

Tel: + 31 (0)20.4108.700

[europa@webex.com](mailto:europa@webex.com)

United Kingdom

Tel: 0800.389.9772

[europa@webex.com](mailto:europa@webex.com)

Australia & New Zealand

Tel: + 61 (0)3.9653.9581

[AsiaPacInfo@webex.com](mailto:AsiaPacInfo@webex.com)

China (HK)

Tel: + 852.8201.0228

[AsiaPacInfo@webex.com](mailto:AsiaPacInfo@webex.com)

India

Tel: 080.2228.6377/17030 9330

[sales@cyberbazaarindia.com](mailto:sales@cyberbazaarindia.com)

Japan

Tel: + 81 3 5501 3272

[JapanInfo@webex.com](mailto:JapanInfo@webex.com)

