

# WebEx API 快速指南

1	文档目的 .....	2
2	前提 .....	2
3	API 种类 .....	3
4	XML API .....	3
5	XML API 测试页 .....	3
6	第一次调用 XML API .....	4
7	XML API 结构 .....	5
7.1	Request .....	5
7.2	Response .....	6
7.3	Error .....	7
7.4	数据格式 .....	7
8	XML API 分类 .....	7
9	用户 API .....	8
9.1	创建用户 .....	8
9.2	取得用户信息 .....	9
10	会议 API .....	9
10.1	创建会议 .....	9
10.2	取得会议信息 .....	10
10.3	修改会议信息 .....	11
10.4	列出多个会议信息 .....	12
10.5	获得主持人开会地址 .....	13
10.6	获得加会地址 .....	14
10.7	删除会议 .....	15
11	培训 API .....	15
11.1	创建培训 .....	15
11.2	取得培训信息 .....	16
11.3	修改培训信息 .....	17
11.4	列出多个培训信息 .....	18

12	编程中调用 .....	18
13	SSO Site .....	19
14	SSO Site API 调用 .....	20
14.1	获得 tokenId .....	20
14.2	获得 samlResponse .....	21
14.3	获得 sessionTicket .....	22
14.4	根据 sessionTicket 去调用 API .....	24
15	URL API .....	24
16	URL API 语法 .....	25
17	URL API 常用功能 .....	25
17.1	开会 .....	26
17.2	登陆 .....	26
17.3	一步执行 .....	26
17.4	加会 .....	27
17.5	关闭会议 .....	27
17.6	详细文档 .....	27
18	Mobile API .....	28
19	附录 .....	28

## 1 文档目的

本文档目的是让用户快速了解 Webex API 的使用，它是对 Webex 官方英文文档的简化。

## 2 前提

1. 您大致了解了 WebEx 的会议系统，以及关于站点，用户，会议，参与者的概念。
2. 在和 WebEx XML API 做集成开发前，您应该已经知道你帐号的 WebEx 的站点名字（下面以 SiteName 表示）和您的用户名（WebExID）和密码(Password)。下面的所有的 demo 中，需要填写你自己的信息。

### 3 API 种类

Webex 提供了 XMLAPI, URL API, WebEx Mobile API, TSPAPI, WebACD API, NBR Web Service API 。 本文档主要介绍 XML API , URL API 和 WebEx Mobile API。

### 4 XML API

XML API 是用 web 的方式 访问 , 用 post 方式发送一个 XML 文本请求, 服务器会返回一个 xml 文本 。

### 5 XML API 测试页

你可以用下面的 test page 在浏览器中测试 XML API.

**注意, WebEx 因为安全原因, 不久将来将只支持 https , 不支持 http 访问。**

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/xml; charset=utf-8"
/>
</head>
<body>
<font size="5" >WebEx XML API Test Page</font>
<br>
<br>
<form action=
https://SiteName.webex.com.cn/WBXService/xml8.0.0/XMLService
method=get>
<textarea rows =22 columns = 50 cols="88" name="XML">
</textarea>
<br>
<input name="submit" type="submit" value="submit">
</form>
</body>
</html>
```

**注意 :** 需要用你的站点名称替换红色的 SiteName 。

然后将上面的文本存为 html 文件, 在浏览器中打开, 就可以进行简单的 Webex XML API 测试。

## 6 第一次调用 XML API

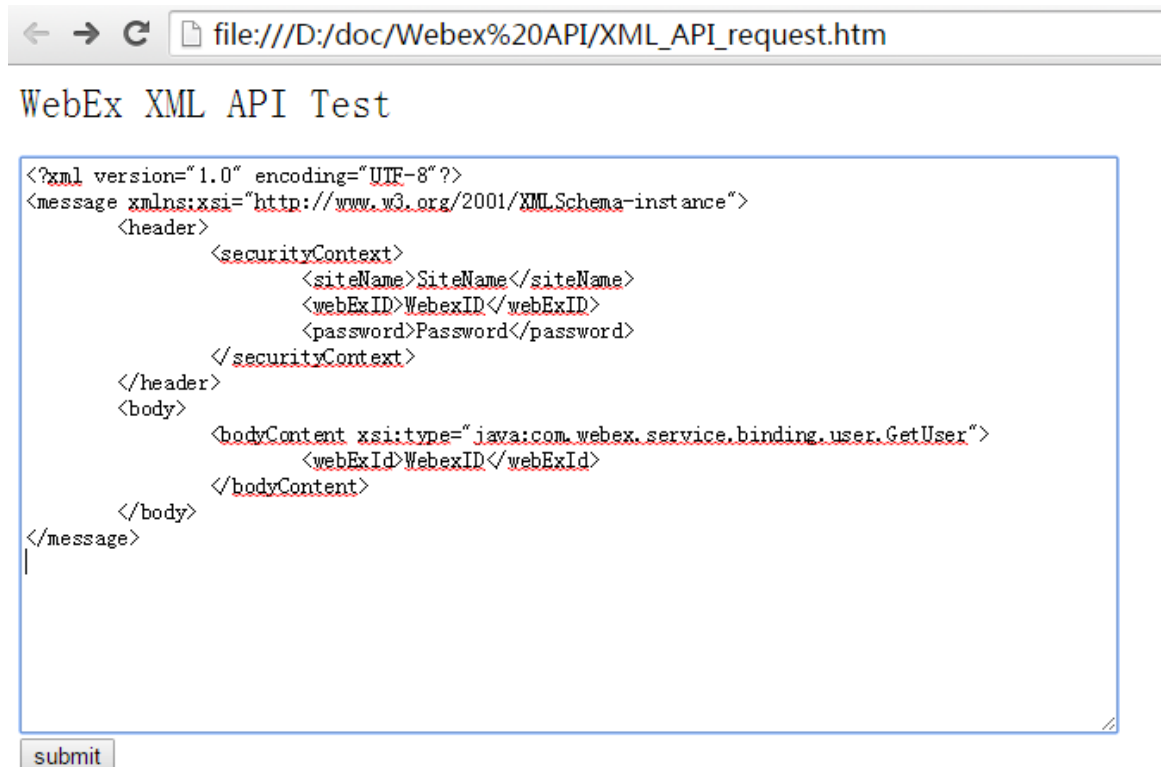
配置完 Test Page 后就可以在浏览器中访问 test page 进行测试。

我们调用的第一个 API 是 GetUser API。

这个 API 可以取得你自己的帐号信息，下面是 request text。

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent xsi:type="java:com.webex.service.binding.user.GetUser">
      <webExId>WebexID</webExId>
    </bodyContent>
  </body>
</message>
```

在浏览器中打开刚才生成的 test page，输上面的内容。



← → ↻ file:///D:/doc/Webex%20API/XML\_API\_request.htm

### WebEx XML API Test

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent xsi:type="java:com.webex.service.binding.user.GetUser">
      <webExId>WebexID</webExId>
    </bodyContent>
  </body>
</message>
```

submit

再将上面的 xml 文本，红色的文字部分换成你的信息，  
点击 submit 按钮就可以执行。

你将得到类似下面的结果。



The screenshot shows a web browser window with the address bar displaying `https://bigpad.webex.com.cn/WBXService/xml8.0.0/XMLService`. The page content shows an XML document. The first line is a message: `This XML file does not appear to have any style information associated with it`. Below this, the XML structure is displayed with a tree view on the left. The root element is `<serv:message>` with namespaces `xmlns:serv="http://www.webex.com/schemas/2002/06/service"` and `xmlns:com="http://www.webex.com"`. It contains a `<serv:header>` element with a `<serv:response>` element. Inside the response, there is a `<serv:result>SUCCESS</serv:result>` and a `<serv:gsbStatus>PRIMARY</serv:gsbStatus>`. The `<serv:body>` element contains a `<serv:bodyContent>` element with namespace `xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"` and `xsi:type="use"`. Inside this, there is a `<use:firstName>bigpade</use:firstName>`.

这样你就的第一个 XML API 就访问成功了。

注意，如果测试的时候，没有返回，可以把测试页中的 `method=post` 改成 `method=get`，或者换用 IE 浏览器试一试。

## 7 XML API 结构

你的第一个 API 调用成功，现在就来介绍 XML API 的结构。

每个 XML API 都分成 Request（请求），Response（返回）两部分。  
通过上面的 Test page 可以看到。通过 Http Post 提交 Request，就会返回 response。

### 7.1 Request

XML API 的请求分为两部分一部分是请求头（header）另一部分是请求的主体（body）。

header 部分，里面主要包含 `securityContext` 安全验证信息，每个 API 都一样。如上面最简单的 GetUser 请求为例：

```
<header>
  <securityContext>
    <siteName>SiteName</siteName>
    <webExID>WebexID</webExID>
    <password>Password</password>
  </securityContext>
</header>
```

主要包含站点，用户名，和密码三个字段。

另一部分是 Body，每个 API 都不相同。包含请求的具体信息。你需要在 `xsi:type` 中指定到底访问的是哪个 API。同样以 `GetUser` 为例。

```
<body>
  <bodyContent xsi:type="java:com.webex.service.binding.user.GetUser">
    <webExId>WebexID</webExId>
  </bodyContent>
</body>
```

`com.webex.service.binding.user.GetUser` 指定的是具体需要执行的 API。至于具体内容，其实只有一个 `WebexID`，就是你想要要取的用户帐号的帐号名。

## 7.2 Response

返回的 `response` 部分也同 `request` 一样，包含 `header` 和 `body` 两部分，

`header` 内容包含命令执行成功还是失败以及当前服务器是 `PRIMARY` 还是 `BACKUP`。所有的 API 返回都是一样。`GetUser` 例子如下。

```
<serv:header>
  <serv:response>
    <serv:result>SUCCESS</serv:result>
    <serv:gsbStatus>PRIMARY</serv:gsbStatus>
  </serv:response>
</serv:header>
```

`body` 里包含返回的主体。每个 API 都不一样。`GetUser` 例子如下。它返回了所有用户相关信息。

```
<serv:body>
  <serv:bodyContent
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="use:getUserResponse">
    <use:firstName>bigpade</use:firstName>
    .....
  </serv:bodyContent>
</serv:body>
```

## 7.3 Error

如果 API request 错误。比如上面的 request 中的用户名使用某个不存在的 ID 。就会返回下面的错误 response 。

```
<serv:message
.....>
  <serv:header>
    <serv:response>
      <serv:result>FAILURE</serv:result>
      <serv:reason>Corresponding User not found</serv:reason>
      <serv:gsbStatus>PRIMARY</serv:gsbStatus>
      <serv:exceptionID>030001</serv:exceptionID>
    </serv:response>
  </serv:header>
  <serv:body>
    <serv:bodyContent />
  </serv:body>
</serv:message>
```

你可以根据 result 是否为 FAILURE ，判断是否出错。  
还可以取具体错误原因 reason 和 错误 ID exceptionID 。

## 7.4 数据格式

API 中的 Date 类型的格式为 MM/dd/yyyy HH:mm:ss

# 8 XML API 分类

Webex 提供了许多服务。根据服务不同，XML API 可以分为下面几类：

User service : 提供用户相关 API  
General Session service : 通用的会议相关 API  
Meeting service: 会议中心 (MC) 相关 API  
Training Session Service: 培训中心 (TC) 相关 API  
Event Session Service: 网络研讨会，事件中心 (EC) 相关 API  
Support Session Service: 支持中心 (SC) 相关 API  
History Service: 历史记录相关 API  
Site Service: 站点管理相关 API  
Meeting Attendee Service: 会议参加者相关 API

Meeting Type Service: 会议类型相关 API。

具体可以参考附录中的官方 xml API 参考手册。

本文档只简单介绍几种常用的 API 。

其他 API ，以及具体 API 的细节，都可以在 API 参考手册，以及 API 的 xsd 文件中查找到具体使用规范。

注意，下面的所有的 xml 文档 ，测试时，都需要把用户信息和上面 GetUser API 一样替换成自己的相关信息，文档中不会再做说明。

## 9 用户 API

User service 里面常用的是 CreateUser ， GetUser 。

### 9.1 创建用户

建立个拥有开会权限的新用户 Create User

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent
xsi:type="java:com.webex.service.binding.user.CreateUser">
      <firstName>jason</firstName>
      <lastName>chen</lastName>
      <webExId>jasonc</webExId>
      <email>jasonc@sina.com</email>
      <password>111111</password>
      <active>ACTIVATED</active>
      <privilege>
        <host>true</host>
      </privilege>
    </bodyContent>
  </body>
</serv:message>
```



```
        </bodyContent>
    </body>
</serv:message>
```

## 9.2 取得用户信息

文档开始第一次调用的 API ，就是 GetUser API 。

## 10 会议 API

会议 API ，Meeting service 里面最常用的是 create meeting ，get Meeting ，Set Meeting ，List meeting ，Del meeting ，已经取到开会和加会信息。

### 10.1 创建会议

你可以用 Create Meeting API 创建会议 。

比如用户安排了一个 2015 年 11 月 30 号 10 点开的一个会议，会议的密码是 111111，时间 20 分钟 ，会议的名称是 sample meeting，议程为 test。

XML request

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <header>
        <securityContext>
            <siteName>SiteName</siteName>
            <webExID>WebexID</webExID>
            <password>Password</password>
        </securityContext>
    </header>
    <body>
        <bodyContent
xsi:type="java:com.webex.service.binding.meeting.CreateMeeting">
            <accessControl>
                <meetingPassword>111111</meetingPassword>
            </accessControl>
            <metaData>
```

```

        <confName>Sample Meeting</confName>
        <agenda>Test</agenda>
    </metaData>
    <schedule>
        <startDate>11/30/2015 10:00:00</startDate>
        <duration>20</duration>
    </schedule>
</bodyContent>
</body>
</serv:message>

```

执行成功后返回如下

```

▼<serv:message xmlns:serv="http://www.webex.com/schemas/2002/06/service" xmlns:com="http://www.
xmlns:att="http://www.webex.com/schemas/2002/06/service/attendee">
  ▼<serv:header>
    ▼<serv:response>
      <serv:result>SUCCESS</serv:result>
      <serv:gsbStatus>PRIMARY</serv:gsbStatus>
    </serv:response>
  </serv:header>
  ▼<serv:body>
    ▼<serv:bodyContent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="meet:cre:
      <meet:meetingkey>448248575</meet:meetingkey>
      ▼<meet:iCalendarURL>

```

注意返回的 meetingKey 需要记下来，这是会议的 ID，取这个会议信息的时候需要用。

## 10.2 取得会议信息

你可以用 GetMeeting API 取到某个会议的详细信息。  
XML request 如下

```

<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>

```

```

        <body>
            <bodyContent
xsi:type="java:com.webex.service.binding.meeting.GetMeeting"
>
                <meetingKey>448248575</meetingKey>
            </bodyContent>
        </body>
    </serv:message>

```

注意 meeting key 用的是上面 Create Meeting response 的 meeting key 。

Reponse 此处省略。

### 10.3 修改会议信息

创建会议后，你还可以用 SetMeeting API 来修改 meeting 。  
 比如你想修改上面创建会议的议程。你可以用下面的 request 。  
 注意，你会发现 所有 Meeting service 的 xsd 是一样的。也就是 Create Meeting ,Set meeting 的输入格式 和 Get meeting 的返回格式是一致的。  
 你可以根据 xsd ，也可以参考 Get meeting 的 response 来写 request 。  
 其他类别的 API 也一样。

```

<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <header>
        <securityContext>
            <siteName>SiteName</siteName>
            <webExID>WebexID</webExID>
            <password>Password</password>
        </securityContext>
    </header>
    <body>
        <bodyContent
xsi:type="java:com.webex.service.binding.meeting.SetMeeting"
>
            <metaData>
                <agenda>chnanged</agenda>
            </metaData>
        </bodyContent>
    </body>
</serv:message>

```

```
        </metaData>
        <meetingkey>448248575</meetingkey>
        </bodyContent>
    </body>
</serv:message>
```

## 10.4 列出多个会议信息

如果你希望 list 多个 Meeting 信息, 你可以调用 LstsummaryMeeting API , 这个 API 可以安装你的条件, 进行 内容列表。比如下面的 request 就是列出这个 site 所有的 meeting (public , 公开的 会议) 。按照会议的开始时间 排序。

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <header>
        <securityContext>
            <siteName>SiteName</siteName>
            <webExID>WebexID</webExID>
            <password>Password</password>
        </securityContext>
    </header>
    <body>
        <bodyContent

xsi:type="java:com.webex.service.binding.meeting.LstsummaryM
eeting">
            <listControl>
                <startFrom>1</startFrom>
                <maximumNum>10</maximumNum>
                <listMethod>OR</listMethod>
            </listControl>
            <order>
                <orderBy>STARTTIME</orderBy>
                <orderAD>ASC</orderAD>
            </order>
        </bodyContent>
    </body>
```

```
</serv:message>
```

这里面有两个在 API 比较通用的 element 。

*Order*: 按照多个条件进行排序。

*ListControl*: 分页显示 。

## 10.5 获得主持人开会地址

当你开会的时候, 你希望获得开会地址, 你可以通过 GethosturlMeeting API 来获得开会地址。

。

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent
xsi:type="java:com.webex.service.binding.meeting.GethosturlMeeting">
      <meetingKey>448248575</meetingKey>
    </bodyContent>
  </body>
</serv:message>
```

返回信息为

```

<?xml version="1.0" encoding="UTF-8"?>
<serv:message xmlns:att="http://www.webex.com/schemas/2002/06/service/attendee" xmlns:meet="http://www.webex.com/schemas/2002/06/service/attendee" xmlns:com="http://www.webex.com/schemas/2002/06/common" xmlns:serv="http://www.webex.com/schemas/2002/06/service">
  <serv:header>
    <serv:response>
      <serv:result>SUCCESS</serv:result>
      <serv:gsbStatus>PRIMARY</serv:gsbStatus>
    </serv:response>
  </serv:header>
  <serv:body>
    <serv:bodyContent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="meet:gethosturlMeetingResponse">
      <meet:hostMeetingURL>https://cussupport.webex.com.cn/cussupport/p.php?AT=LI&WID=jason&TK=8310484a7e36e5a883c35f8a3a609711c8bc2941c56ecdfcf79fc8292425b97f&MU=https%3A%2F%2Fcussupport%2Fm.php%3FAT%3D183859958%26Rnd%3D0.250853363960454</meet:hostMeetingURL>
    </serv:bodyContent>
  </serv:body>
</serv:message>

```

在浏览器上输入 返回的 hostMeetingURL 地址，就可以直接打开 webex meeting client 开会

## 10.6 获得加会地址

同样道理，调用 GetjoinurlMeeting

```

<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent
xsi:type="java:com.webex.service.binding.meeting.GetjoinurlMeeting">
      <meetingKey>448248575</meetingKey>
    </bodyContent>
  </body>
</serv:message>

```

会返回 joinMeetingURL ，该地址可以用来加会。

## 10.7 删除会议

如果你改变计划，希望删除掉某个会议，可以用 DelMeeting 来删除会议。

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent
      xsi:type="java:com.webex.service.binding.meeting.
DelMeeting">
      <meetingKey>11111</meetingKey>
    </bodyContent>
  </body>
</serv:message>
```

## 11 培训 API

Train Session Service （在线培训） 和 Meeting service 类似 ， 不过 一个管理的是 培训课程（Train session ） ， 一个管理的是会议。 里面最常用的是 create Train Session， get Train Session， Set Train Session 和 List Train Session 。

### 11.1 创建培训

使用 Create Train Session API

用户安排了一个 2015 年 11 月 30 号 10 点开的一个培训课程，会议的密码是 111111，时间 60 分钟 ，会议的名称是 sample training session，议程为 test。

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent xsi:type=
"java:com.webex.service.binding.training.CreateTrainingSessi
on">
      <accessControl>
        <sessionPassword>111111</sessionPassword>
      </accessControl>
      <schedule>
        <startDate>11/30/2015 10:00:00</startDate>
        <duration>60</duration>
      </schedule>
      <metaData>
        <confName>sample training session
</confName>
        <agenda>test</agenda>
        <description>description</description>
      </metaData>
    </bodyContent>
  </body>
</serv:message>

```

## 11.2 取得培训信息

取某个培训课程的信息

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>

```



```

        <webExID>WebexID</webExID>
        <password>Password</password>
    </securityContext>
</header>
<body>
    <bodyContent
xsi:type="java:com.webex.service.binding.training.GetTrainin
gSession">
        <sessionKey>76736484</sessionKey>
    </bodyContent>
</body>
</serv:message>

```

### 11.3 修改培训信息

修改某个培训课程的信息

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <header>
        <securityContext>
            <siteName>SiteName</siteName>
            <webExID>WebexID</webExID>
            <password>Password</password>
        </securityContext>
    </header>
    <body>
        <bodyContent
xsi:type="java:com.webex.service.binding.training.SetTrainin
gSession">
            <metaData>
                <agenda>changed</agenda>
            </metaData>
            <sessionKey>18975177</sessionKey>
        </bodyContent>
    </body>
</serv:message>

```

## 11.4 列出多个培训信息

LstsummaryTrainingSession API 列出从 2015 年 3 月 10 日起的,这个 site 所有的培训课程 (public , 公开的 会议) 。按照开始时间 排序。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>SiteName</siteName>
      <webExID>WebexID</webExID>
      <password>Password</password>
    </securityContext>
  </header>
  <body>
    <bodyContent xsi:type=
"java:com.webex.service.binding.training.LstsummaryTrainingS
ession">
      <ListControl>
        <startFrom>1</startFrom>
        <maximumNum>10</maximumNum>
        <listMethod>OR</listMethod>
      </ListControl>
      <order>
        <orderBy>STARTTIME</orderBy>
        <orderAD>ASC</orderAD>
      </order>
      <dateScope>
        <startDateStart>03/10/2015
00:00:00</startDateStart>
        <timeZoneID>45</timeZoneID>
      </dateScope>
    </bodyContent>
  </body>
</serv:message>
```

## 12 编程中调用

测试的时候, 你可以用 test page 去测试  
实际在代码中调用 XML API 可以参考下面 sample

Java code 调用例子



javaSample.txt

PHP , ASP , C# code 调用的例子可以  
参考

<https://developer.cisco.com/site/webex-developer/develop-test/xml-api/sample-code/>

## 13 SSO Site

上面的 XML API 都是对于普通 Site 的调用，用户在 API 中直接输入 webex sit 密码 来提供验证。但是对于 SSO site （Single Sign On，单点登录），因为安全性考虑，用户不能泄露 密码 给 webex server 。要验证用户，用户只能提供 session ticket 给 Webex API server 。Webex API server 再到相关的 SSO server 上去验证用户。

科天云 支持 SSO 。科天 SSO ，可以分为 Non-SSO，SSO-1 和 SSO-2 三种方式。



Non-SSO:



SSO-1：到科天 CI 认证



SSO-2: 到企业 CI 认证

## 14 SSO Site API 调用

对于 NON-SSO 的站点，调用比较简单，上面所有的 XML 的请求就是 NON-SSO site 的调用，需要提供密码。

但是对于 SSO 的站点，您需要获得 WebEx 站点的授权凭证及 sessionTicket，这个 sessionTicket 被用来当作用户对 WebEx 的授权访问，在 XML 请求的 header 里面您需要用 sessionTicket 来替换掉 password。

SSO-2 和 SSO-1 类似，都需要通过先通过 samlResponse 取的 Sesion Tikcet。然后通过 Sesion Tikcet 调用 API。  
所以本文以 SSO-1 为主介绍调用。

SSO 调用 API 需要分为下面几步。

### 14.1 获得 tokenId

访问科天 CI 去获得 tokenId:

ServerURL:

<https://ci.ketianyun.com:8280/sso/json/authenticate>

Set header: X-OpenAM-Username=%您的用户名%  
X-OpenAM-Password=%您的密码%  
Content-Type=application/json

Java code 片段如下:

```

CloseableHttpClient httpClient =
HttpClientBuilder.createDefault();
HttpPost httpPost = new
HttpPost("https://ci.ketianyun.com:8280/sso/json/authenticat

```

```

e");
    httpPost.setHeader("X-OpenAM-Username",
"wanjun.zhu@tcl.com");
    httpPost.setHeader("X-OpenAM-Password", "P@ss1234");
    httpPost.setHeader("Content-Type", "application/json");

    CloseableHttpResponse response =
httpClient.execute(httpPost);
    HttpEntity entity = response.getEntity();
    String tokenIdPattern = "\"tokenId\": \"(.+?)\"";
    String tokenId = RegexUtil.find(tokenIdPattern,
EntityUtils.toString(entity));
    response.close();

```

下面是 find 的 code

```

public static String find(String pattern, String content)
{
    Pattern regexPattern = Pattern.compile(pattern);
    Matcher matcher = regexPattern.matcher(content);
    if (matcher.find()) {
        return matcher.group(1);
    }
    return null;
}

```

## 14.2 获得 samlResponse

访问科天 CI 去获得 samlResponse

ServerURL: <https://ci.ketianyun.com:8280/sso/idpssoinit?>

spEntityID=<https://ketianyundev.webex.com.cn&metaAlias=/idp>

Set Header: Cookie=iPlanetDirectoryPro=%tokenId from step 1%

Java code 片段:

```

CloseableHttpClient httpClient =
HttpClient.createDefault();
StringBuilder sb = new StringBuilder();
sb.append("https://ci.ketianyun.com:8280/sso/idpssoinit"
);

```

```

        sb.append("?spEntityID=https://ketianyundev.webex.com.cn
&metaAlias=idp");
        HttpGet httpGet = new HttpGet(sb.toString());
        httpGet.setHeader("Cookie", "iPlanetDirectoryPro=" +
tokenId);
        CloseableHttpResponse response =
httpClient.execute(httpGet);
        HttpEntity entity = response.getEntity();
        String entity = EntityUtils.toString(entity);
        String pattern =
"name=\"SAMLResponse\"\\s+?value=\"(.+?)\"";
        String samlResponse = RegexUtil.find(pattern, entity);
        response.close();

```

### 14.3 获得 sessionTicket

Call XML API authenticateUser 去获得 sessionTicket

ServiceURL: HTTPS://%siteName%.webex.com.cn/WBXService/XMLService

Parameter:XML

Value:

```

<?xml version="1.0" encoding="UTF-8"?>
<serv:message
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:serv="http://www.webex.com/schemas/2002/06/service">
    <header>
        <securityContext>
            <siteName>%siteName%</siteName>
            <webExID>%webExID%</webExID>
        </securityContext>
    </header>
    <body>

        <bodyContent
xsi:type="java:com.webex.service.binding.user.AuthenticateUs
er">
            <samlResponse>%samlResponse%</samlResponse>
            <protocol>SAML2.0</protocol>
        </bodyContent>
    </body>
</serv:message>

```

Java code:

```

        CloseableHttpClient httpClient =
HttpClients.createDefault();
        HttpPost xmlPost = new
HttpPost("https://ketianyundev.webex.com.cn/WBXService/xml8.
0.0/XMLService");
        List <NameValuePair> parameters = new ArrayList
<NameValuePair>();
        parameters.add(new BasicNameValuePair("XML",
getAuthenticateUserRequest("wanjun.zhu@tcl.com", "ketianyunde
v", response)));
        xmlPost.setEntity(new UrlEncodedFormEntity(parameters));
        CloseableHttpResponse response =
httpClient.execute(xmlPost);
        HttpEntity entity3 = response.getEntity();
        String sessionTicketPattern =
"<use:sessionTicket>(.*?)</use:sessionTicket>";
        String sessionTicket =
RegexUtil.find(sessionTicketPattern,
EntityUtils.toString(entity3));
        response.close();

        private static String getAuthenticateUserRequest(String
userName,String siteName,String samlResponse) {
            StringBuilder sb = new StringBuilder();
            sb.append("<?xml version=\"1.0\"
encoding=\"UTF-8\"?>");
            sb.append("<serv:message
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xmlns:serv=\"http://www.webex.com/schemas/2002/06/service\">
");
            sb.append("<header><securityContext><siteName>").append(site
Name).append("</siteName>");
            sb.append("<webExID>").append(userName).append("</webExID></
securityContext></header>");
            sb.append("<body><bodyContent
xsi:type=\"java:com.webex.service.binding.user.AuthenticateU
ser\">");
            sb.append("<samlResponse>").append(samlResponse).append("</s
amlResponse>");

```

```

sb.append("<protocol>SAML2.0</protocol></bodyContent></body>
</serv:message>");
    return sb.toString();
}

```

## 14.4 根据 sessionTicket 去调用 API

以 Create Meeting 为例。

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<serv:message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <webExID>wayne@tcl.com</webExID>
      <sessionTicket>AAABT4bE7mYAABUYA0gAKEgyU0sAAAACLKPeGvfCHSCvPWBmhtgwKko2uifs/6
        SFrpqdqLtV2/ZBSAA8U0sAAAAccYjNe11Fs0g3ruojNJhmcMfSX/vPrkdexHeAvsAg1ed52AmyiD
        Mn6StcIC42HKvznM88kiNlc9La+M8+Dsh+7u8A2hjgySsv/NILE7L+EEyD935UVuK3X1MZ7L6Kq9L
        zkvGj5atlVOV9gcEU06PrkWi0dK/6SnfhBj4p5dMI8MoQSVxDVm+DlZDSVvPHgW+mssaq5XRQ81Kc
        4EFf1PhJAODKHyIUnCr1LZRrziDpmxfRkxBR19BRERJTkdF00hBMjU2X0FMR09SSVRITV8=</sessionTicket>
      <siteName>51learning</siteName>
    </securityContext>
  </header>
  <body>
    <bodyContent xsi:type="java:com.webex.service.binding.meeting.CreateMeeting">
      <accessControl>
        <meetingPassword>pass123</meetingPassword>
      </accessControl>
      <metaData>
        <confName>Sample Meeting</confName>
        <agenda>Test</agenda>
      </metaData>
      <schedule>
        <startDate>08/31/2015 10:30:00</startDate>
        <duration>60</duration>
      </schedule>
    </bodyContent>
  </body>
</serv:message>

```

## 15 URL API

XML API 只能处理会前会后的逻辑。

如果开会，加会，可以按照上面介绍，先调用 XML 取得开会加会地址，然后再加会。也可以通过另外的方法，使用 WebEx URL API 直接加会。

下面对 URL API 使用也做个简单介绍

WebEx URL API 提供一种 基于浏览器（HTTPS） 对 WebEx 的操作，你可以在浏览器地址栏中直接执行。



## 16 URL API 语法

URL AP 的语法为

`https://siteName.webex.com/siteName/function.php?AT=command&commandArgument`

*siteName* : 你帐号的 WebEx 的站点名

*function* : 这个 API 的 funtion page , URL api 把按照功能分为几个 php page 功能。比如 meeting 相关的功能都集中在 m.php。

*command* : 是 这个 funciton 下面的子功能 。比如 start meeting 就是 HM。

*commandArgument* : URL 格式的多个 API 参数。

下面以 start meeting 为例, 在 test site 上召开 meeting key 为 189644419 的会议 API 为

`https://test.webex.com.cn/test/m.php?AT=HM&MK=189644419` 。

**注意 get 格式 只是为了方便说明格式 , WebEx 因为安全原因, 要求密码的API只能使用POST方式, 请参照如下代码提交 (请用相应的值替换%%变量):**

```
<html>
<body>
<form
action="https://%siteName%.webex.com.cn/%siteName%/p.php"
method="post">
  <input name="AT" Type="text" value="LI" size=30 />
  <br>
  <input name="WID" Type="text" value="%UserName%" size=30
/>
  <br>
  <input name="PW" Type="text" value="%Password%" size=30 />
  <br>
  <input name="MU" Type="text"
value=https://%siteName%.webex.com.cn/%siteName%/m.php?AT=HM
&MK=%MeetingNumber% size=75 />
  <br>
  <input type="submit" text="submit" />
</form>
</body>
</html>
```

## 17 URL API 常用功能

## 17.1 开会

你可以通过 HM (Host Meeting ) API 来开会

语法

m.php?AT=HM

&MK=*MeetingKey*

MK , 你相要开会的会议号 meeting key , 这个会议好你安排会议后应该会得到的。

例子见上面

## 17.2 登陆

上面的命令是 Start meeting , 但是在 Start meeting 前, 你必须要登陆, 验证了你身份后, 才有权限开会。

下面是登录的语法

p.php?AT=LI

&WID=*WebExID*

[ &PW=*Password*]

WID: 你的 webex 帐号

PW: 你的帐号密码

例子

<https://test.webex.com.cn/test/p.php?AT=LI&WID=johnson&PW=111111>

你可以在浏览器中执行这个 API, 就会发现用户会处于登录后状态。

再在这个浏览器地址栏执行上面的开会命令, 就可以开会了。

## 17.3 一步执行

上面开会分为两部, URL API 也支持通过一个 API 来执行。

你可以在登录后再执行下一个 API 。

把 start meeting API url encode 后作为 MU 的参数加在 login API 后面就可以了。系统会在执行 Login API 后在执行 MU 的参数 。

比如上面两个 API 一步执行的例子如下。

<https://test.webex.com.cn/test/p.php?AT=LI&WID=johnson&PW=111111&MU=https%3a%2f%2ftest.webex.com.cn%2ftest%2fm.php%3fAT%3dHM%26MK%3d189644419>

这样就可以在浏览器上执行一次就可以登陆开会了。

## 17.4 加会

如果你不是 meeting 主持人，你可以通过调用下面 JM(Join Meeting)API 加入会议。

```
m.php?AT=JM
&MK=MeetingKey
[ &AN=AttendeeName ]
[ &AE=AttendeeEmail ]
```

MK ， 会议号 meeting key 。

AN 和 AE ， 加会时候需要输入的参加者姓名和邮件， 可选， 如果在API 里面填写了， 加会时就不需要再输入了。

例子

<https://test.webex.com.cn/test/m.php?AT=JM&MK=189644419&AN=test&AE=test@163.com>

## 17.5 关闭会议

如何你是 host ， 你还可以通过 KM (kill meeting) API 来关闭会议。

语法

```
w.php?AT=KM
&MK=MeetingKey
[ &WID=WebExID ]
[ &PW=Password ]
```

MK ， 会议号 meeting key

WID: 你的 webex 帐号

PW: 你的帐号密码

## 17.6 详细文档

上面只是 URLAPI 简单的介绍， 详细文档， 可以参考附录中的 webex 官方文档。

## 18 Mobile API

可以通过 wbx schema 在移动端实现 一键启会，和一键加会。

比如在 Android 加会可以使用下面的 url

wbx://meeting?MK=123456789&MPW=1111111

在 Android 端用下面的代码调用

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse(url));
activity.startActivityForResult(intent, WebexInvokeRequest);
```

将会调用 WebEx app 的加入会议功能。

具体可以参考相关文档。

## 19 附录

WebEx XML API 最新版本 官方 手册

<https://developer.cisco.com/site/webex-developer/develop-test/xml-api/xml-api-reference/>

WebEx XML API 手册

<https://developer.cisco.com/media/webex-xml-api/Chapter2XML-ExpressedRequestandResponseDocuments.html>

虽然这个文档 API 不是最新，但是这个文档里面比最新手册多了相关概念的介绍和附录。更全面。

最新的 WebEx XML API xsd

<https://developer.cisco.com/site/webex-developer/develop-test/xml-api/schema/>

WebEx URL API Overview

<https://developer.cisco.com/site/webex-developer/develop-test/url-api/overview/>

WebEx URL API release and guide

<https://developer.cisco.com/site/webex-developer/develop-test/url-api/reference/>