webexteamssdk Documentation

Release 1.6.1.post28.dev0+g22dcb08

Chris Lunsford

Jan 11, 2023

Contents

Simple, lightweight, scalable Python API wrapper for the Webex Teams APIs

Welcome to the docs! webexteamssdk is a *community developed* Pythonic wrapping of the Webex Teams APIs. The package represents all of the Cisco Webex Teams API interactions via native Python tools. Making working with the Cisco Webex Teams APIs in Python a *native* and *natural* experience.

webexteamssdk helps you get things done faster. We take care of the API semantics, and you can focus on writing your code.

With webexteamssdk, you can easily:

- Interact with the Webex Teams APIs in an interactive Python session

- Quickly create code that enables you get something done in Webex Teams

- Leverage the API wrapper to cleanly add Webex Teams functionality to your project without having to write the boilerplate code for working with the Webex Teams APIs

To *dive in* and see how webexteamssdk makes your life better, check out the *Quickstart*!

The User Guide

1.1 Installation

1.1.1 PIP Install

webexteamssdk is available via PIP and the Python Package Index (PyPI). To install webexteamssdk, simply run this command from your terminal of choice:

```
$ pip install webexteamssdk
```

The webexteamssdk package is distributed as a *source distribution* (no binaries).

1.1.2 PIP Upgrade

To ensure that you have the latest version, check-for and install upgrades via PIP:

```
$ pip install webexteamssdk --upgrade
```

1.1.3 Get the Source Code

webexteamssdk is developed on GitHub. If you like and use this package, please take a few seconds to Star the package on the CiscoDevNet/webexteamssdk GitHub page. Your feedback and contributions are always welcome.

Use the following command to download the source code (GIT repository):

```
$ git clone https://github.com/CiscoDevNet/webexteamssdk.git
```

You can then install the package to your environment, with the following command:

```
$ python setup.py install
```

Copyright (c) 2016-2020 Cisco and/or its affiliates.

1.2 Introduction

1.2.1 Work with the Webex Teams APIs in Native Python!

Sure, working with the Webex Teams APIs is easy (see developer.webex.com). They are RESTful, naturally structured, require only a simple Access Token for authentication, and the data is elegantly represented in intuitive JSON. What could be easier?

```python
import requests

URL = 'https://api.ciscospark.com/v1/messages'
ACCESS_TOKEN = '<your_access_token>'
ROOM_ID = '<room_id>'
MESSAGE_TEXT = '<message_text>'

headers = {'Authorization': 'Bearer ' + ACCESS_TOKEN,
           'Content-type': 'application/json;charset=utf-8'}
post_data = {'roomId': ROOM_ID,
             'text': MESSAGE_TEXT}
response = requests.post(URL, json=post_data, headers=headers)
if response.status_code == 200:
    # Great your message was posted!
    message_id = response.json['id']
    message_text = response.json['text']
    print("New message created, with ID:", message_id)
    print(message_text)
else:
    # Oops something went wrong...  Better do something about it.
    print(response.status_code, response.text)
```

Like I said, EASY. However, in use, the code can become rather repetitive. . .

- You have to setup the environment every time

- You have to remember URLs, request parameters and JSON formats (or reference the docs)

- You have to parse the returned JSON and work with multiple layers of list and dictionary indexes

- When requesting lists of items, you have to deal with pagination

Enter **webexteamssdk**, a simple API wrapper that wraps all of the Webex Teams API calls and returned JSON objects within native Python objects and methods.

With webexteamssdk, the above Python code can be consolidated to the following:

```python
from webexteamssdk import WebexTeamsAPI

api = WebexTeamsAPI()
try:
    message = api.messages.create('<room_id>', text='<message_text>')
    print("New message created, with ID:", message.id)
    print(message.text)
except ApiError as e:
    print(e)
```

webexteamssdk handles all of this for you:

- Reads your Webex Teams access token from a `WEBEX_TEAMS_ACCESS_TOKEN` environment variable

- Wraps and represents all Webex Teams API calls as a simple hierarchical tree of native-Python methods (with default arguments provided everywhere possible!)

- If your Python IDE supports **auto-completion** (like PyCharm), you can navigate the available methods and object attributes right within your IDE

- Represents all returned JSON objects as native Python objects - you can access all of the object's attributes using native *dot.syntax*

- **Automatic and Transparent Pagination!** When requesting 'lists of objects' from Webex Teams, requests for additional pages of responses are efficiently and automatically requested as needed

- **Automatic Rate-Limit Handling** Sending a lot of requests to Webex Teams? Don't worry; we have you covered. Webex Teams will respond with a rate-limit response, which will automatically be caught and "handled" for you. Your requests and script will automatically be "paused" for the amount of time specified by Webex Teams, while we wait for the Webex Teams rate-limit timer to cool down. After the cool-down, your request will automatically be retried, and your script will continue to run as normal. Handling all of this requires zero (0) changes to your code - you're welcome.

  Just know that if you are are sending a lot of requests, your script might take longer to run if your requests are getting rate limited.

- Multipart encoding and uploading of local files, when creating messages with local file attachments

All of this, combined, lets you do powerful things simply:

```python
from webexteamssdk import WebexTeamsAPI

api = WebexTeamsAPI()

# Find all rooms that have 'webexteamssdk Demo' in their title
all_rooms = api.rooms.list()
demo_rooms = [room for room in all_rooms if 'webexteamssdk Demo' in room.title]

# Delete all of the demo rooms
for room in demo_rooms:
    api.rooms.delete(room.id)

# Create a new demo room
demo_room = api.rooms.create('webexteamssdk Demo')

# Add people to the new demo room
email_addresses = ["test01@cmlccie.com", "test02@cmlccie.com"]
for email in email_addresses:
    api.memberships.create(demo_room.id, personEmail=email)

# Post a message to the new room, and upload a file
api.messages.create(demo_room.id, text="Welcome to the room!",
                    files=["https://www.webex.com/content/dam/wbx/us/images/dg-integ/
↪teams_icon.png"])
```

That's more than 6 Webex Teams API calls in less than 23 lines of code (with comments and whitespace), and likely more than that since webexteamssdk handles pagination for you automatically!

Head over to the *Quickstart* page to begin working with the **Webex Teams APIs in native Python**!

1.2.2 MIT License

webexteamssdk is currently licensed under the MIT Open Source License, and distributed as a source distribution (no binaries) via *PyPI*, and the complete *source code* is available on GitHub.

1.2.3 webexteamssdk License

The MIT License (MIT)

Copyright (c) 2016-2020 Cisco and/or its affiliates.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 2016-2020 Cisco and/or its affiliates.

1.3 Quickstart

Dive in! . . . to get started using the webexteamssdk package:

Make sure that you have:

- A Webex Teams Account *(a free account works fine,* sign-up for one here *)*

- webexteamssdk *installed*

- webexteamssdk *upgraded to the latest version*

1.3.1 Get your Webex Teams Access Token

To interact with the Webex Teams APIs, you must have a **Webex Teams Access Token**. A Webex Teams Access Token is how the Webex Teams APIs validate access and identify the requesting user.

To get your personal access token:

1. Login to developer.webex.com

2. Click on Docs or browse to the Getting Started page

3. You will find your *personal access token* in the Authentication section

Note:

> "Your personal access token is great for testing the API with your account but it should never be used in applications."

1.3.2 Use your Webex Teams Access Token

As a best practice, you can store your Webex Teams access token 'credential' as an environment variable in your development or production environment. By default, webexteamssdk will look for a `WEBEX_TEAMS_ACCESS_TOKEN` environment variable when creating new connection objects.

There are many places and diverse ways that you can set an environment variable, which can include:

- A setting within your development IDE
- A setting in your container / PaaS service
- A statement in a shell script that configures and launches your app

It can be as simple as setting it in your CLI before running your script. . .

```
$ export WEBEX_TEAMS_ACCESS_TOKEN=your_access_token_here
$ python myscript.py
```

. . . or putting your credentials in a shell script that you `source` when your shell starts up or before your run a script:

```
$ cat mycredentials.sh
export WEBEX_TEAMS_ACCESS_TOKEN=your_access_token_here
$ source mycredentials.sh
$ python myscript.py
```

However you choose to set it, if you have your access token stored in a `WEBEX_TEAMS_ACCESS_TOKEN` environment variable when using webexteamssdk, you are good to go. webexteamssdk will pull and use this access token, by default, when creating new *WebexTeamsAPI* objects.

If you don't want to set your access token as an environment variable, or perhaps your application will acquire access tokens via some other means, you can manually provide your access token when creating a WebexTeamsAPI object.

1.3.3 Create a WebexTeamsAPI "Connection Object"

To make interacting with the Webex Teams APIs as simple and intuitive as possible, all of the APIs have 'wrapped' underneath a single interface. To get started, import the *WebexTeamsAPI* class and create an API "connection object".

```
>>> from webexteamssdk import WebexTeamsAPI
>>> api = WebexTeamsAPI()
```

As discussed above (*Use your Webex Teams Access Token*), webexteamssdk defaults to pulling your Webex Teams access token from a WEBEX_TEAMS_ACCESS_TOKEN environment variable. If you do not have this environment variable set and you try to create a new *WebexTeamsAPI* object without providing a Webex Teams access token, a *webexteamssdkException* will be raised.

```
>>> from webexteamssdk import WebexTeamsAPI
>>> api = WebexTeamsAPI()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "webexteamssdk/__init__.py", line 114, in __init__
    raise webexteamssdkException(error_message)
webexteamssdkException: You must provide an Webex Teams access token to interact
with the Webex Teams APIs, either via a WEBEX_TEAMS_ACCESS_TOKEN environment
variable or via the access_token argument.
```

Use the access_token argument to manually provide your access token, when creating a new *WebexTeamsAPI* connection object.

```
>>> from webexteamssdk import WebexTeamsAPI
>>> api = WebexTeamsAPI(access_token='lkj345w...')
```

Note that this can be very useful if you are reading in access token(s) from a file or database and/or when you want to create more than one connection object.

```
>>> from webexteamssdk import WebexTeamsAPI
>>> chris_at = 'lkj345w...'
>>> veronica_at = 'kl45kln...'
>>> chris_api = WebexTeamsAPI(access_token=chris_at)
>>> veronica_api = WebexTeamsAPI(access_token=veronica_at)
```

If you authenticate a user via a OAuth flow you can also initialize a api object directly by providing the OAuth information.

```
>>> from webexteamssdk import WebexTeamsAPI
>>> client_id = "<from oauth>"
>>> client_secret = "<from oauth>"
>>> oauth_code = "<from oauth>"
>>> redirect_uri = "<from oauth>"
>>> api = WebexTeamsAPI(client_id=client_id,
                        client_secret=client_secret,
                        oauth_code=oauth_code,
                        redirect_uri=redirect_uri
                        )
```

You can also pass a proxy configuration upon initialization in case you are behind a firewall. See the requests documentation on Proxies for details.

```
>>> from webexteamssdk import WebexTeamsAPI
>>> proxy = {'https': 'http://<proxy_ip>:<proxy_port>'}
>>> api = WebexTeamsAPI(access_token=<your_access_token>, proxies=proxy)
```

1.3.4 Making API Calls

Now that you have created a *WebexTeamsAPI* "connection object," you are ready to start making API calls.

```
>>> api.people.me()
Person({"displayName": "Chris Lunsford", "firstName": "Chris", "created": "2012-06-
↪15T20:36:48.914Z", "lastName": "Lunsford", "emails": ["chrlunsf@cisco.com"], "avatar
↪": "https://1efa7a94ed216783e352-c62266528714497a17239ececf39e9e2.ssl.cf1.rackcdn.
↪com/V1~ba1ecf557a7e0b7cc3081998df965aad~7-HrvYOJSQ6eJgWJuFVbzg==~1600", "id":
↪"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk"})
```

It really is that easy.

All of the calls have been wrapped and represented as native Python method calls, like *WebexTeamsAPI.people. me()* which gets the person details for the authenticated user (the user who's access token you are using) - see the https://api.ciscospark.com/v1/people/me API endpoint documentation.

As you can see, we have represented the API endpoints using simple terms that are aligned with the API docs; for example, representing the `people/me` API endpoint as a `people.me()` method available underneath the *WebexTeamsAPI* connection object.

A full list of the available API methods, with their descriptions and parameters, is available in the *User API Doc*, and a brief summary of the structure is provided here.

WebexTeamsAPI	access_tokens	get() refresh()
	ad-min_audit_events	list()
	attach-ment_actions	create() get()
	events	list() get()
	guest_issuer	create()
	licenses	list() create()
	memberships	list() create() get() update() delete()
	messages	list() list_direct() create() get() delete()
	organizations	list() create()
	people	list() create() get() update() me()
	roles	list() create()
	rooms	list() create() get() get_meeting_info() update() delete()
	team_memberships	list() create() get() update() delete()
	teams	list() create() get() update() delete()
	webhooks	list() create() get() update() delete()

You can easily access and call any of these methods directly from your *WebexTeamsAPI* connection object:

```
>>> chris_id =
↪"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk"
>>> api.people.get(personId=chris_id)
Person({"displayName": "Chris Lunsford", "firstName": "Chris", "created": "2012-06-
↪15T20:36:48.914Z", "lastName": "Lunsford", "emails": ["chrlunsf@cisco.com"], "avatar
↪": "https://1efa7a94ed216783e352-c62266528714497a17239ececf39e9e2.ssl.cf1.rackcdn.
↪com/V1~ba1ecf557a7e0b7cc3081998df965aad~7-HrvYOJSQ6eJgWJuFVbzg==~1600", "id":
↪"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk"})
```

1.3.5 Catching Exceptions

If something should go wrong with the API call, an exception will be raised. *ApiError* exceptions are raised when an error condition is returned from the Webex Teams cloud. Details will be provided in the error message.

```python
>>> from webexteamssdk import WebexTeamsAPI, ApiError
>>> api = WebexTeamsAPI()
>>> room = api.rooms.create("webexteamssdk Test Room")
>>> me = api.people.me()
>>> api.memberships.create(roomId=room.id, personId=me.id)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "webexteamssdk/api/memberships.py", line 212, in create
    json_obj = self._session.post('memberships', json=post_data)
  File "webexteamssdk/restsession.py", line 187, in post
    check_response_code(response, erc)
  File "webexteamssdk/utils.py", line 104, in check_response_code
    response=response)
webexteamssdk.exceptions.ApiError: Response Code [409] - The request
could not be processed because it conflicts with some established rule of
the system. For example, a person may not be added to a room more than
once.
```

You can catch any errors returned by the Webex Teams cloud by catching *ApiError* exceptions in a try-except block.

```python
>>> try:
...     api.memberships.create(roomId=room.id, personId=me.id)
... except ApiError as e:
...     memberships = api.memberships.list(roomId=room.id)
...     for membership in memberships:
...         if membership.personId == me.id:
...             print("Doh!  I forgot that I am automatically added to a"
...                   "room when I create it.")
...             break
...     else:
...         print(e)
...
Doh!  I forgot that I am automatically added to a room when I create it.
>>>
```

webexteamssdk will also raise a number of other standard errors (`TypeError`, `ValueError`, etc.); however, these errors are usually caused by incorrect use of the package or methods and should be sorted while debugging your app.

1.3.6 Working with Returned Objects

The Webex Teams cloud returns data objects in JSON format, like so:

```json
{
  "displayName": "Chris Lunsford",
  "firstName": "Chris",
  "created": "2012-06-15T20:36:48.914Z",
  "lastName": "Lunsford",
```

```
  "emails": [
    "chrlunsf@cisco.com"
  ],
  "avatar": "https://1efa7a94ed216783e352-c62266528714497a17239ececf39e9e2.ssl.cf1.
→rackcdn.com/V1~ba1ecf557a7e0b7cc3081998df965aad~7-HrvYOJSQ6eJgWJuFVbzg==~1600",
  "id":
→"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk"
}
```

Sure, JSON data objects can easily be parsed and represented in Python using dictionaries, but when working with an 'object' wouldn't it be nice to be able to work with it like an object - using native object syntax (like accessing attributes using '.' notation)? webexteamssdk enables you to do just that:

```
>>> me = api.people.me()
>>> me.id
u'Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk'
>>> me.displayName
u'Chris Lunsford'
```

Representing and treating Webex Teams data objects as Python data objects, can really help clean up your code and make coding easier:

1. You don't need to create variables to hold the data attributes, just use the attributes available underneath the data object.

```
>>> # Do this
>>> api.people.get(personId=me.id)
Person({"displayName": "Chris Lunsford", "firstName": "Chris", "created": "2012-
→06-15T20:36:48.914Z", "lastName": "Lunsford", "emails": ["chrlunsf@cisco.com"],
→"avatar": "https://1efa7a94ed216783e352-c62266528714497a17239ececf39e9e2.ssl.
→cf1.rackcdn.com/V1~ba1ecf557a7e0b7cc3081998df965aad~7-HrvYOJSQ6eJgWJuFVbzg==~
→1600", "id":
→"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk
→"})
>>> # Instead of this
>>> my_id = me.id
>>> api.people.get(personId=my_id)
Person({"displayName": "Chris Lunsford", "firstName": "Chris", "created": "2012-
→06-15T20:36:48.914Z", "lastName": "Lunsford", "emails": ["chrlunsf@cisco.com"],
→"avatar": "https://1efa7a94ed216783e352-c62266528714497a17239ececf39e9e2.ssl.
→cf1.rackcdn.com/V1~ba1ecf557a7e0b7cc3081998df965aad~7-HrvYOJSQ6eJgWJuFVbzg==~
→1600", "id":
→"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk
→"})
```

2. If your IDE supports auto-completion (like PyCharm for example), then you can easily see and 'tab-out' available attributes while coding.

   For Example: When working with a `Person` object, your can type the object name followed by a dot 'me.' and see a list of available attributes. Typing a few more letters 'me.dis' narrows down the attribute list to 'displayName', and you can now simply hit '<tab>' to complete your attribute 'me.displayName'.

   This speeds up coding and reduces typo coding errors.

3. When accessing 'optional' attributes, like the `teamId` attribute of a Webex Teams Room object (only present when the room is part of a Webex Teams Team), the `webexteamssdk.Room` object will return `None` when the attribute is not present and will return the attribute's value when it is present. This avoids some boiler plate code and/or needless exception handling, when working with optional attributes.

```
>>> # Instead of doing this
>>> if hasattr(room, 'teamId'):
...         # Do something with the teamId attribute
...         pass
>>> # Or this
>>> try:
...         # Do something with the teamId attribute
...         room.teamId
... except AttributeError as e:
...         pass
>>> # You can do this, which is cleaner
>>> if room.teamId:
...         # Do something with the teamId attribute
...         pass
```

4. It just feels more *natural*. :-) When iterating through sequences, and working with objects in those sequences (see the next section), working with objects as objects is definitely more Pythonic.

   The Zen of Python (PEP 20): "Beautiful is better than ugly." "Simple is better than complex."

A full list of the currently modeled *Webex Teams Data Objects*, with their attributes, is available *here* in the *User API Doc*.

What if Webex Teams adds new data attributes?

Attribute access WILL WORK for the newly added attributes (yes, without a package update!), but tab-completion WILL NOT. webexteamssdk is written to automatically take advantage of new attributes and data as they are returned; however, tab-completion (which relies on source code and introspection) will not work until we update the webexteamssdk package (which is easy to do; raise the issue on the issues page and bug us to add it).

1.3.7 Working with Returned 'Lists' of Objects

Challenge

When you ask Webex Teams for a list of items (like all of the rooms that you are a member of or all of the messages in a room), Webex Teams needs to return these items to you in an efficient way. Sending all of the messages in a room in one transaction or request isn't really feasible (imaging if the room had existed for years!). Additionally, what if you found what you were looking for in the first few (most recent) messages? Sending all of the items would have been a waste of time and resources.

To facilitate efficient transactions when requesting lists of items, the Webex Teams APIs implement RFC5988 (Web Linking) to efficiently send 'pages' of responses (see Pagination on the Webex Teams for Developers site). When you make a request to an Webex Teams API that leverages pagination, Webex Teams returns the first 'page' of results and a link to the 'next page' of results. If information you need isn't contained the first page, you can request the next and so forth.

Solution

Python has a similar construct as well - iterable objects. Iterable objects return their members one at a time, until they have all been returned.

webexteamssdk marries these two concepts (pagination and iterables) to create a simple interface for working with sequences of returned objects.

```
>>> # Returns a iterable object yielding all of the rooms you are a member of
>>> rooms = api.rooms.list()

>>> # Which can easily be iterated to find what you are looking for
```

(continues on next page)

```
>>> for room in rooms:
...     if 'webexteamssdk' in room.title:
...         demo_room = room
...         break

>>> demo_room
Room({"title": "webexteamssdk Test Room", "created": "2016-11-12T03:24:39.278Z",
→"isLocked": false, "lastActivity": "2016-11-12T03:24:39.308Z", "creatorId":
→"Y2lzY29zcGFyazovL3VzL1BFT1BMRS9mZjhlZTZmYi1hZmVmLTRhNGQtOTJiMS1kNmIyMTZiNTg5NDk",
→"type": "group", "id":
→"Y2lzY29zcGFyazovL3VzL1JPT00vOGI1MTIwZTAtYTg4Ny0xMWU2LWFhZjUtZTlmYWEzMWQ1ZmRm"})
```

webexteamssdk provides this functionality by returning `GeneratorContainer` objects for API calls that return lists of items.

In short, `GeneratorContainer` s are iterable objects that incrementally yield 'the next object' returned from your Webex Teams API query request until all items have been returned, and they are reusable. If you create an `rooms` `GeneratorContainer`, like we did above with `rooms = api.rooms.list()`, you can use that object to iterate through the rooms not just once but many times.

Note: Every time you iterate a `GeneratorContainer` object, fresh API calls are made so you are always working with 'live data' from the Cisco Webex Teams Cloud.

webexteamssdk *automatically handles the pagination for you* so that you don't have to think about it or write the boiler plate code to handle requesting pages of responses. webexteamssdk automatically and efficiently requests additional pages from Webex Teams as needed to yield the items you have requested.

A `GeneratorContainer` records all of the parameters of your API call, and uses them to request data from Webex Teams each time you iterate the container.

```
>>> # Returns a iterable object representing all of group rooms you are a member of
>>> group_rooms = api.rooms.list(type='group')

>>> # Returns a iterable object representing all of direct rooms you are a member of
>>> direct_rooms = api.rooms.list(type='direct')

>>> # Iterate through your group rooms
>>> for room in group_rooms:
...     pass

>>> # Iterate through your direct rooms
>>> for room in direct_rooms:
...     pass

>>> # You can iterate through your group rooms again;
>>> # if a new room has been created since the last time, it will show up.
>>> for room in group_rooms:
...     pass
```

These iterable objects are great, but what if I really DO want a list?

Sometimes you really DO want a `list` of items. Perhaps you want to work with the same static `list` of items to ensure you are looking at 'all of the items' and to make sure that your list doesn't change while you are working with it. . .

Whatever your reason for doing so, you can easily 'convert' an iterable object to a standard Python `list` with the `list()` function. This may take a little time for all of the API calls to be made, but the list will contain all of the returned objects.

```
>>> rooms_iterable = api.rooms.list()
>>> rooms_list = list(rooms_iterable)
```

1.3.8 Extending the API with bound methods

As the Webex Teams API is developed and features are added, it may be necessary to extend the webexteamssdk to access those features. Extending the API is simple by binding your own methods to the WebexTeamsAPI connection object. By binding a method, you can extend functionality and leverage all of the objects and quality of life features of webexteamssdk.

```
>>> new_method(self, param):
    ...     json_obj = self._session.get('/example/action/' + param)
    ...     return json_obj

>>> api = WebexTeamsAPI()
>>> api.new_method = new_method
>>> output = WebexTeamsAPI.new_method(param)
```

Copyright (c) 2016-2020 Cisco and/or its affiliates.

1.4 User API Doc

1.4.1 WebexTeamsAPI

The *WebexTeamsAPI* class creates "connection objects" for working with the Webex Teams APIs and hierarchically organizes the Webex Teams APIs and their endpoints underneath these connection objects.

WebexTeamsAPI	*access_tokens*	`get() refresh()`
	ad-min_audit_events	`list()`
	attach-ment_actions	`create() get()`
	events	`list() get()`
	guest_issuer	`create()`
	licenses	`list() create()`
	memberships	`list() create() get() update() delete()`
	messages	`list() list_direct() create() get() delete()`
	organizations	`list() create()`
	people	*list() create() get() update() me()*
	roles	`list() create()`
	rooms	`list() create() get() get_meeting_info() update() delete()`
	team_memberships	`list() create() get() update() delete()`
	teams	`list() create() get() update() delete()`
	webhooks	`list() create() get() update() delete()`

class WebexTeamsAPI
Webex Teams API wrapper.

Creates a 'session' for all API calls through a created WebexTeamsAPI object. The 'session' handles authentication, provides the needed headers, and checks all responses for error conditions.

WebexTeamsAPI wraps all of the individual Webex Teams APIs and represents them in a simple hierarchical structure.

__**init**__ (*access_token=None*, *base_url='https://webexapis.com/v1/'*, *single_request_timeout=60*, *wait_on_rate_limit=True*, *object_factory=<function immutable_data_factory>*, *client_id=None*, *client_secret=None*, *oauth_code=None*, *redirect_uri=None*, *proxies=None*, *be_geo_id=None*, *caller=None*, *disable_ssl_verify=False*)
Create a new WebexTeamsAPI object.

An access token must be used when interacting with the Webex Teams API. This package supports three methods for you to provide that access token:

1. You may manually specify the access token via the *access_token* argument, when creating a new WebexTeamsAPI object.

2. If an access_token argument is not supplied, the package checks for a WE-BEX_TEAMS_ACCESS_TOKEN environment variable.

3. Provide the parameters (client_id, client_secret, oauth_code and oauth_redirect_uri) from your oauth flow.

An AccessTokenError is raised if an access token is not provided via one of these two methods.

Parameters

- **access_token** (*basestring*) – The access token to be used for API calls to the Webex Teams service. Defaults to checking for a WEBEX_TEAMS_ACCESS_TOKEN environment variable.

- **base_url** (*basestring*) – The base URL to be prefixed to the individual API endpoint suffixes. Defaults to webexteamssdk.DEFAULT_BASE_URL.

- **single_request_timeout** (*int*) – Timeout (in seconds) for RESTful HTTP requests. Defaults to webexteamssdk.config.DEFAULT_SINGLE_REQUEST_TIMEOUT.

- **wait_on_rate_limit** (*bool*) – Enables or disables automatic rate-limit handling. Defaults to webexteamssdk.config.DEFAULT_WAIT_ON_RATE_LIMIT.

- **object_factory** (*callable*) – The factory function to use to create Python objects from the returned Webex Teams JSON data objects.

- **client_id** (*basestring*) – The client id of your integration. Provided upon creation in the portal.

- **client_secret** (*basestring*) – The client secret of your integration. Provided upon creation in the portal.

- **oauth_code** (*basestring*) – The oauth authorization code provided by the user oauth process.

- **oauth_redirect_uri** (*basestring*) – The redirect URI used in the user OAuth process.

- **proxies** (*dict*) – Dictionary of proxies passed on to the requests session.

- **be_geo_id** (*basestring*) – Optional partner identifier for API usage tracking. Defaults to checking for a BE_GEO_ID environment variable.

- **caller** (*basestring*) – Optional identifier for API usage tracking. Defaults to checking for a WEBEX_PYTHON_SDK_CALLER environment variable.

- **disable_ssl_verify** (*bool*) – Optional boolean flag to disable ssl verification. Defaults to False. If set to True, the requests session won't verify ssl certs anymore.

Returns A new WebexTeamsAPI object.

Return type *WebexTeamsAPI*

Raises

- `TypeError` – If the parameter types are incorrect.

- `AccessTokenError` – If an access token is not provided via the access_token argument or an environment variable.

single_request_timeout
Timeout (in seconds) for an single HTTP request.

wait_on_rate_limit
Automatic rate-limit handling enabled / disabled.

access_tokens = <webexteamssdk.api.access_tokens.AccessTokensAPI object>

classmethod from_oauth_code(*client_id*, *client_secret*, *code*, *redirect_uri*)
Create a new WebexTeamsAPI connection object using an OAuth code.

Exchange an Authorization Code for an Access Token, then use the access token to create a new WebexTeamsAPI connection object.

Parameters

- **client_id** (*basestring*) – Provided when you created your integration.

- **client_secret** (*basestring*) – Provided when you created your integration.

- **code** (*basestring*) – The Authorization Code provided by the user OAuth process.

- **redirect_uri** (*basestring*) – The redirect URI used in the user OAuth process.

Returns A new WebexTeamsAPI object initialized with the access token from the OAuth Authentication Code exchange.

Return type *WebexTeamsAPI*

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

classmethod from_oauth_refresh(*client_id*, *client_secret*, *refresh_token*)
Create a new WebexTeamsAPI connection object using an OAuth refresh.

Exchange a refresh token for an Access Token, then use the access token to create a new WebexTeamsAPI connection object.

Parameters

- **client_id** (*basestring*) – Provided when you created your integration.

- **client_secret** (*basestring*) – Provided when you created your integration.

- **refresh_token** (*basestring*) – Provided when you requested the Access Token.

Returns A new WebexTeamsAPI object initialized with the access token from the OAuth Refresh Token exchange.

Return type *WebexTeamsAPI*

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

access_tokens

class AccessTokensAPI
>   Webex Teams Access-Tokens API.

>   Wraps the Webex Teams Access-Tokens API and exposes the API as native Python methods that return native Python objects.

>   **base_url**
>   >   The base URL the API endpoints.

>   **single_request_timeout**
>   >   Timeout in seconds for the API requests.

>   **get** (*client_id*, *client_secret*, *code*, *redirect_uri*)
>   >   Exchange an Authorization Code for an Access Token.

>   >   Exchange an Authorization Code for an Access Token that can be used to invoke the APIs.

>   >   **Parameters**

>   >   - **client_id** (*basestring*) – Provided when you created your integration.

>   >   - **client_secret** (*basestring*) – Provided when you created your integration.

>   >   - **code** (*basestring*) – The Authorization Code provided by the user OAuth process.

>   >   - **redirect_uri** (*basestring*) – The redirect URI used in the user OAuth process.

>   >   **Returns** An AccessToken object with the access token provided by the Webex Teams cloud.

>   >   **Return type** *AccessToken*

>   >   **Raises**

>   >   - `TypeError` – If the parameter types are incorrect.

>   >   - `ApiError` – If the Webex Teams cloud returns an error.

>   **refresh** (*client_id*, *client_secret*, *refresh_token*)
>   >   Return a refreshed Access Token from the provided refresh_token.

>   >   **Parameters**

>   >   - **client_id** (*basestring*) – Provided when you created your integration.

>   >   - **client_secret** (*basestring*) – Provided when you created your integration.

>   >   - **refresh_token** (*basestring*) – Provided when you requested the Access Token.

>   >   **Returns** With the access token provided by the Webex Teams cloud.

>   >   **Return type** *AccessToken*

>   >   **Raises**

>   >   - `TypeError` – If the parameter types are incorrect.

>   >   - `ApiError` – If the Webex Teams cloud returns an error.

admin_audit_events

class AdminAuditEventsAPI
>   Admin Audit Events API.

Wraps the Webex Teams Admin Audit Events API and exposes the API as native Python methods that return native Python objects.

list (*orgId*, *_from*, *to*, *actorId=None*, *max=100*, *offset=0*, ***request_parameters*)
    List Organizations.

    This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all audit events returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

    Parameters

- **orgId** (*basestring*) – List events in this organization, by ID.
- **_from** (*basestring*) – List events which occurred after a specific date and time.
- **to** (*basestring*) – List events which occurred before a specific date and time.
- **actorId** (*basestring*) – List events performed by this person, by ID.
- **max** (*int*) – Limit the maximum number of events in the response. The maximum value is 200.
- **offset** (*int*) – Offset from the first result that you want to fetch.
- ***request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

    Returns A GeneratorContainer which, when iterated, yields the organizations returned by the Webex Teams query.

    Return type GeneratorContainer

    Raises

- *TypeError* – If the parameter types are incorrect.
- *ApiError* – If the Webex Teams cloud returns an error.

attachment_actions

class AttachmentActionsAPI
    Webex Teams Attachment Actions API.

    Wraps the Webex Teams Attachment Actions API and exposes the API as native Python methods that return native Python objects.

    create (*type*, *messageId*, *inputs*, ***request_parameters*)
        Create a new attachment action.

        Parameters

- **type** (*basestring*) – The type of action to perform.
- **messageId** (*basestring*) – The ID of the message which contains the attachment.
- **inputs** (*dict*) – The attachment action's inputs.
- ***request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

        Returns A attachment action object with the details of the created attachment action.

> **Return type** *AttachmentAction*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.
> - `ValueError` – If the files parameter is a list of length > 1, or if the string in the list (the only element in the list) does not contain a valid URL or path to a local file.

get(*id*)

Get the details for a attachment action, by ID.

> **Parameters** **id** (*basestring*) – A unique identifier for the attachment action.
>
> **Returns** A Attachment Action object with the details of the requested attachment action.
>
> **Return type** *AttachmentAction*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

events

class EventsAPI

Webex Teams Events API.

Wraps the Webex Teams Events API and exposes the API as native Python methods that return native Python objects.

list(*resource=None, type=None, actorId=None, _from=None, to=None, max=None, **request_parameters*)

List events.

List events in your organization. Several query parameters are available to filter the response.

Note: *from* is a keyword in Python and may not be used as a variable name, so we had to use *_from* instead.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all events returned by the query. The generator will automatically request additional 'pages' of responses from Wevex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **resource** (*basestring*) – Limit results to a specific resource type. Possible values: "messages", "memberships".
> - **type** (*basestring*) – Limit results to a specific event type. Possible values: "created", "updated", "deleted".
> - **actorId** (*basestring*) – Limit results to events performed by this person, by ID.
> - **_from** (*basestring*) – Limit results to events which occurred after a date and time, in ISO8601 format (yyyy-MM-dd'T'HH:mm:ss.SSSZ).
> - **to** (*basestring*) – Limit results to events which occurred before a date and time, in ISO8601 format (yyyy-MM-dd'T'HH:mm:ss.SSSZ).

- **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.

- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns A GeneratorContainer which, when iterated, yields the events returned by the Webex Teams query.

Return type GeneratorContainer

Raises

- *TypeError* – If the parameter types are incorrect.

- *ApiError* – If the Webex Teams cloud returns an error.

get (*eventId*)
Get the details for an event, by event ID.

Parameters eventId (*basestring*) – The ID of the event to be retrieved.

Returns A event object with the details of the requested room.

Return type *Event*

Raises

- *TypeError* – If the parameter types are incorrect.

- *ApiError* – If the Webex Teams cloud returns an error.

guest_issuer

class GuestIssuerAPI
Webex Teams Guest Issuer API.

Wraps the Webex Teams Guest Issuer API and exposes the API as native methods that return native Python objects.

create (*sub*, *name*, *iss*, *exp*, *secret*)
Create a new guest issuer using the provided issuer token.

This function returns a guest issuer with an api access token.

Parameters

- **sub** (*basestring*) – The subject of the token. This is your unique and public identifier for the guest user. This claim may contain only letters, numbers, and hyphens.

- **name** (*basestring*) – The display name of the guest user. This will be the name shown in Webex Teams clients.

- **iss** (*basestring*) – The issuer of the token. Use the Guest Issuer ID provided in My Webex Teams Apps.

- **exp** (*basestring*) – The exp time of the token, as a UNIX timestamp in seconds. Use the lowest practical value for the use of the token. This is not the exp time for the guest user's session.

- **secret** (*basestring*) – Use the secret Webex provided you when you created your Guest Issuer App. The secret will be used to sign the token request.

Returns A Guest Issuer token with a valid access token.

> **Return type** *GuestIssuerToken*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect
>
> - `ApiError` – If the webex teams cloud returns an error.

licenses

class LicensesAPI

Webex Teams Licenses API.

Wraps the Webex Teams Licenses API and exposes the API as native Python methods that return native Python objects.

list(*orgId=None*, ***request_parameters*)

List all licenses for a given organization.

If no orgId is specified, the default is the organization of the authenticated user.

> **Parameters**
>
> - **orgId** (*basestring*) – Specify the organization, by ID.
>
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the licenses returned by the Webex Teams query.
>
> **Return type** GeneratorContainer
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

get(*licenseId*)

Get the details of a License, by ID.

> **Parameters** **licenseId** (*basestring*) – The ID of the License to be retrieved.
>
> **Returns** A License object with the details of the requested License.
>
> **Return type** *License*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

memberships

class MembershipsAPI

Webex Teams Memberships API.

Wraps the Webex Teams Memberships API and exposes the API as native Python methods that return native Python objects.

list(*roomId=None*, *personId=None*, *personEmail=None*, *max=None*, ***request_parameters*)

List room memberships.

By default, lists memberships for rooms to which the authenticated user belongs.

Use query parameters to filter the response.

Use *roomId* to list memberships for a room, by ID.

Use either *personId* or *personEmail* to filter the results.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all memberships returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **roomId** (`basestring`) – Limit results to a specific room, by ID.
> - **personId** (`basestring`) – Limit results to a specific person, by ID.
> - **personEmail** (`basestring`) – Limit results to a specific person, by email address.
> - **max** (`int`) – Limit the maximum number of items returned from the Webex Teams service per request.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the memberships returned by the Webex Teams query.
>
> **Return type** GeneratorContainer
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

create(*roomId*, *personId=None*, *personEmail=None*, *isModerator=False*, ***request_parameters*)

Add someone to a room by Person ID or email address.

Add someone to a room by Person ID or email address; optionally making them a moderator.

> **Parameters**
>
> - **roomId** (`basestring`) – The room ID.
> - **personId** (`basestring`) – The ID of the person.
> - **personEmail** (`basestring`) – The email address of the person.
> - **isModerator** (`bool`) – Set to True to make the person a room moderator.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A Membership object with the details of the created membership.
>
> **Return type** *Membership*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.

> • `ApiError` – If the Webex Teams cloud returns an error.

get(*membershipId*)

> Get details for a membership, by ID.

> > **Parameters** **membershipId**(`basestring`) – The membership ID.

> > **Returns** A Membership object with the details of the requested membership.

> > **Return type** *Membership*

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

update(*membershipId*, *isModerator=None*, ***request_parameters*)

> Update properties for a membership, by ID.

> > **Parameters**

> > > • **membershipId**(`basestring`) – The membership ID.

> > > • **isModerator**(`bool`) – Set to True to make the person a room moderator.

> > > • ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

> > **Returns** A Membership object with the updated Webex Teams membership details.

> > **Return type** *Membership*

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

delete(*membershipId*)

> Delete a membership, by ID.

> > **Parameters** **membershipId**(`basestring`) – The membership ID.

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

messages

class MessagesAPI

> Webex Teams Messages API.

> Wraps the Webex Teams Messages API and exposes the API as native Python methods that return native Python objects.

> **list**(*roomId*, *parentId=None*, *mentionedPeople=None*, *before=None*, *beforeMessage=None*, *max=50*, ***request_parameters*)

> > Lists messages in a room.

> > Each message will include content attachments if present.

> > The list API sorts the messages in descending order by creation date.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all messages returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **roomId** (*basestring*) – List messages for a room, by ID.
> - **parentId** (*basestring*) – List messages with a parent, by ID.
> - **mentionedPeople** (*basestring*) – List messages where the caller is mentioned by specifying "me" or the caller *personId*.
> - **before** (*basestring*) – List messages sent before a date and time, in ISO8601 format.
> - **beforeMessage** (*basestring*) – List messages sent before a message, by ID.
> - **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the messages returned by the Webex Teams query.
>
> **Return type** GeneratorContainer
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

list_direct (*personId=None*, *personEmail=None*, *parentId=None*, ***request_parameters*)
List all messages in a 1:1 (direct) room.

Use the *personId* or *personEmail* query parameter to specify the room.

The list API sorts the messages in descending order by creation date.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all messages returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **personId** (*basestring*) – List messages in a 1:1 room, by person ID.
> - **personEmail** (*basestring*) – List messages in a 1:1 room, by person email.
> - **parentId** (*basestring*) – List messages with a parent, by ID.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the messages returned by the Webex Teams query.
>
> **Return type** GeneratorContainer

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

create(*roomId=None*, *parentId=None*, *toPersonId=None*, *toPersonEmail=None*, *text=None*, *markdown=None*, *files=None*, *attachments=None*, ***request_parameters*)
Post a message to a room.

The files parameter is a list, which accepts multiple values to allow for future expansion, but currently only one file may be included with the message.

Parameters

- **roomId** (*basestring*) – The room ID.

- **toPersonId** (*basestring*) – The ID of the recipient when sending a private 1:1 message.

- **toPersonEmail** (*basestring*) – The email address of the recipient when sending a private 1:1 message.

- **text** (*basestring*) – The message, in plain text. If *markdown* is specified this parameter may be optionally used to provide alternate text for UI clients that do not support rich text.

- **markdown** (*basestring*) – The message, in markdown format.

- **files** (*list*) – A list of public URL(s) or local path(s) to files to be posted into the room. Only one file is allowed per message.

- **attachments** (*list*) – Content attachments to attach to the message. See the Cards Guide for more information.

- **parentId** (*basestring*) – The parent message to reply to. This will start or reply to a thread.

- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns A Message object with the details of the created message.

Return type *Message*

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

- `ValueError` – If the files parameter is a list of length > 1, or if the string in the list (the only element in the list) does not contain a valid URL or path to a local file.

get(*messageId*)
Get the details of a message, by ID.

Parameters **messageId** (*basestring*) – The ID of the message to be retrieved.

Returns A Message object with the details of the requested message.

Return type *Message*

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

delete(*messageId*)
> Delete a message.

> > Parameters **messageId** (*basestring*) – The ID of the message to be deleted.

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

organizations

class OrganizationsAPI
> Webex Teams Organizations API.

> Wraps the Webex Teams Organizations API and exposes the API as native Python methods that return native Python objects.

> **list**(***request_parameters*)
> > List Organizations.

> > Parameters ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

> > **Returns** A GeneratorContainer which, when iterated, yields the organizations returned by the Webex Teams query.

> > **Return type** GeneratorContainer

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

> **get**(*orgId*)
> > Get the details of an Organization, by ID.

> > Parameters **orgId** (*basestring*) – The ID of the Organization to be retrieved.

> > **Returns** An Organization object with the details of the requested organization.

> > **Return type** *Organization*

> > **Raises**

> > > • `TypeError` – If the parameter types are incorrect.

> > > • `ApiError` – If the Webex Teams cloud returns an error.

people

class PeopleAPI
> Webex Teams People API.

> Wraps the Webex Teams People API and exposes the API as native Python methods that return native Python objects.

> **list**(*email=None*, *displayName=None*, *id=None*, *orgId=None*, *max=None*, ***request_parameters*)
> > List people in your organization.

> > For most users, either the *email* or *displayName* parameter is required. Admin users can omit these fields and list all users in their organization.

Response properties associated with a user's presence status, such as *status* or *lastActivity*, will only be displayed for people within your organization or an organization you manage. Presence information will not be shown if the authenticated user has disabled status sharing.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all people returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **email** (*basestring*) – The e-mail address of the person to be found.
>
> - **displayName** (*basestring*) – The complete or beginning portion of the display-Name to be searched.
>
> - **id** (*basestring*) – List people by ID. Accepts up to 85 person IDs separated by commas.
>
> - **orgId** (*basestring*) – The organization ID.
>
> - **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.
>
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the people returned by the Webex Teams query.
>
> **Return type** GeneratorContainer
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

create(*emails*, *displayName=None*, *firstName=None*, *lastName=None*, *avatar=None*, *orgId=None*, *roles=None*, *licenses=None*, ***request_parameters*)
Create a new user account for a given organization

Only an admin can create a new user account.

> **Parameters**
>
> - **emails** (*list*) – Email address(es) of the person (list of strings).
>
> - **displayName** (*basestring*) – Full name of the person.
>
> - **firstName** (*basestring*) – First name of the person.
>
> - **lastName** (*basestring*) – Last name of the person.
>
> - **avatar** (*basestring*) – URL to the person's avatar in PNG format.
>
> - **orgId** (*basestring*) – ID of the organization to which this person belongs.
>
> - **roles** (*list*) – Roles of the person (list of strings containing the role IDs to be assigned to the person).
>
> - **licenses** (*list*) – Licenses allocated to the person (list of strings - containing the license IDs to be allocated to the person).

- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

> **Returns** A Person object with the details of the created person.
>
> **Return type** *Person*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

get (*personId*)

Get a person's details, by ID.

> **Parameters personId** (*basestring*) – The ID of the person to be retrieved.
>
> **Returns** A Person object with the details of the requested person.
>
> **Return type** *Person*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

update (*personId*, *emails=None*, *displayName=None*, *firstName=None*, *lastName=None*, *avatar=None*, *orgId=None*, *roles=None*, *licenses=None*, ***request_parameters*)

Update details for a person, by ID.

Only an admin can update a person's details.

Email addresses for a person cannot be changed via the Webex Teams API.

Include all details for the person. This action expects all user details to be present in the request. A common approach is to first GET the person's details, make changes, then PUT both the changed and unchanged values.

> **Parameters**
>
> - **personId** (*basestring*) – The person ID.
>
> - **emails** (*list*) – Email address(es) of the person (list of strings).
>
> - **displayName** (*basestring*) – Full name of the person.
>
> - **firstName** (*basestring*) – First name of the person.
>
> - **lastName** (*basestring*) – Last name of the person.
>
> - **avatar** (*basestring*) – URL to the person's avatar in PNG format.
>
> - **orgId** (*basestring*) – ID of the organization to which this person belongs.
>
> - **roles** (*list*) – Roles of the person (list of strings containing the role IDs to be assigned to the person).
>
> - **licenses** (*list*) – Licenses allocated to the person (list of strings - containing the license IDs to be allocated to the person).
>
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A Person object with the updated details.
>
> **Return type** *Person*

> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

delete(*personId*)
:   Remove a person from the system.

    Only an admin can remove a person.

    > **Parameters personId** (`basestring`) – The ID of the person to be deleted.

    > **Raises**
    >
    > - `TypeError` – If the parameter types are incorrect.
    >
    > - `ApiError` – If the Webex Teams cloud returns an error.

me()
:   Get the details of the person accessing the API.

    > **Raises** `ApiError` – If the Webex Teams cloud returns an error.

roles

class RolesAPI
:   Webex Teams Roles API.

    Wraps the Webex Teams Roles API and exposes the API as native Python methods that return native Python objects.

    list(***request_parameters*)
    :   List all roles.

        > **Parameters **request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

        > **Returns** A GeneratorContainer which, when iterated, yields the roles returned by the Webex Teams query.

        > **Return type** GeneratorContainer

        > **Raises**
        >
        > - `TypeError` – If the parameter types are incorrect.
        >
        > - `ApiError` – If the Webex Teams cloud returns an error.

    get(*roleId*)
    :   Get the details of a Role, by ID.

        > **Parameters roleId** (`basestring`) – The ID of the Role to be retrieved.

        > **Returns** A Role object with the details of the requested Role.

        > **Return type** *Role*

        > **Raises**
        >
        > - `TypeError` – If the parameter types are incorrect.
        >
        > - `ApiError` – If the Webex Teams cloud returns an error.

class RoomsAPI

Webex Teams Rooms API.

Wraps the Webex Teams Rooms API and exposes the API as native Python methods that return native Python objects.

list (*teamId=None*, *type=None*, *sortBy=None*, *max=100*, ***request_parameters*)

List rooms.

By default, lists rooms to which the authenticated user belongs.

This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all rooms returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

> **Parameters**
>
> - **teamId** (*basestring*) – Limit the rooms to those associated with a team, by ID.
> - **type** (*basestring*) – 'direct' returns all 1-to-1 rooms. *group* returns all group rooms. If not specified or values not matched, will return all room types.
> - **sortBy** (*basestring*) – Sort results by room ID (*id*), most recent activity (*lastactivity*), or most recently created (*created*).
> - **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A GeneratorContainer which, when iterated, yields the rooms returned by the Webex Teams query.
>
> **Return type** GeneratorContainer
>
> **Raises**
>
> - *TypeError* – If the parameter types are incorrect.
> - *ApiError* – If the Webex Teams cloud returns an error.

create (*title*, *teamId=None*, ***request_parameters*)

Create a room.

The authenticated user is automatically added as a member of the room.

> **Parameters**
>
> - **title** (*basestring*) – A user-friendly name for the room.
> - **teamId** (*basestring*) – The team ID with which this room is associated.
> - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A Room with the details of the created room.
>
> **Return type** *Room*
>
> **Raises**

- `TypeError` – If the parameter types are incorrect.
- `ApiError` – If the Webex Teams cloud returns an error.

get(*roomId*)

Get the details of a room, by ID.

> **Parameters** `roomId` (*basestring*) – The ID of the room to be retrieved.
>
> **Returns** A Room object with the details of the requested room.
>
> **Return type** *Room*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

get_meeting_info(*roomId*)

Get the meeting details for a room.

> **Parameters** `roomId` (*basestring*) – The unique identifier for the room.
>
> **Returns** A Room Meeting Info object with the meeting details for the room such as the SIP address, meeting URL, toll-free and toll dial-in numbers.
>
> **Return type** *RoomMeetingInfo*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

update(*roomId*, *title*, ***request_parameters*)

Update details for a room, by ID.

> **Parameters**
>
> - `roomId` (*basestring*) – The room ID.
> - `title` (*basestring*) – A user-friendly name for the room.
> - `**request_parameters` – Additional request parameters (provides support for parameters that may be added in the future).
>
> **Returns** A Room object with the updated Webex Teams room details.
>
> **Return type** *Room*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

delete(*roomId*)

Delete a room.

> **Parameters** `roomId` (*basestring*) – The ID of the room to be deleted.
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
> - `ApiError` – If the Webex Teams cloud returns an error.

class TeamsAPI

    Webex Teams Teams API.

    Wraps the Webex Teams Teams API and exposes the API as native Python methods that return native Python objects.

    list(*max=100*, ***request_parameters*)

        List teams to which the authenticated user belongs.

        This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all teams returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.

        Parameters

            • **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.

            • ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

        Returns A GeneratorContainer which, when iterated, yields the teams returned by the Webex Teams query.

        Return type GeneratorContainer

        Raises

            • TypeError – If the parameter types are incorrect.

            • ApiError – If the Webex Teams cloud returns an error.

    create(*name*, ***request_parameters*)

        Create a team.

        The authenticated user is automatically added as a member of the team.

        Parameters

            • **name** (*basestring*) – A user-friendly name for the team.

            • ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

        Returns A Team object with the details of the created team.

        Return type *Team*

        Raises

            • TypeError – If the parameter types are incorrect.

            • ApiError – If the Webex Teams cloud returns an error.

    get(*teamId*)

        Get the details of a team, by ID.

        Parameters **teamId** (*basestring*) – The ID of the team to be retrieved.

        Returns A Team object with the details of the requested team.

> **Return type** *Team*
>
> > **Raises**
> >
> > - `TypeError` – If the parameter types are incorrect.
> >
> > - `ApiError` – If the Webex Teams cloud returns an error.

update(*teamId*, *name*, ***request_parameters*)
> Update details for a team, by ID.
>
> > **Parameters**
> >
> > - **teamId** (*basestring*) – The team ID.
> >
> > - **name** (*basestring*) – A user-friendly name for the team.
> >
> > - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
>
> > **Returns** A Team object with the updated Webex Teams team details.
>
> > **Return type** *Team*
>
> > **Raises**
> >
> > - `TypeError` – If the parameter types are incorrect.
> >
> > - `ApiError` – If the Webex Teams cloud returns an error.

delete(*teamId*)
> Delete a team.
>
> > **Parameters** **teamId** (*basestring*) – The ID of the team to be deleted.
>
> > **Raises**
> >
> > - `TypeError` – If the parameter types are incorrect.
> >
> > - `ApiError` – If the Webex Teams cloud returns an error.

team_memberships

class TeamMembershipsAPI
> Webex Teams Team-Memberships API.
>
> Wraps the Webex Teams Memberships API and exposes the API as native Python methods that return native Python objects.
>
> **list**(*teamId*, *max=100*, ***request_parameters*)
> > List team memberships for a team, by ID.
> >
> > This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all team memberships returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.
> >
> > > **Parameters**
> > >
> > > - **teamId** (*basestring*) – List team memberships for a team, by ID.
> > >
> > > - **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.

- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns A GeneratorContainer which, when iterated, yields the team memberships returned by the Webex Teams query.

Return type GeneratorContainer

Raises

- *TypeError* – If the parameter types are incorrect.
- *ApiError* – If the Webex Teams cloud returns an error.

create(*teamId*, *personId=None*, *personEmail=None*, *isModerator=False*, ***request_parameters*)
    Add someone to a team by Person ID or email address.

Add someone to a team by Person ID or email address; optionally making them a moderator.

Parameters

- **teamId** (*basestring*) – The team ID.
- **personId** (*basestring*) – The person ID.
- **personEmail** (*basestring*) – The email address of the person.
- **isModerator** (*bool*) – Set to True to make the person a team moderator.
- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns A TeamMembership object with the details of the created team membership.

Return type *TeamMembership*

Raises

- *TypeError* – If the parameter types are incorrect.
- *ApiError* – If the Webex Teams cloud returns an error.

get(*membershipId*)
    Get details for a team membership, by ID.

Parameters **membershipId** (*basestring*) – The team membership ID.

Returns A TeamMembership object with the details of the requested team membership.

Return type *TeamMembership*

Raises

- *TypeError* – If the parameter types are incorrect.
- *ApiError* – If the Webex Teams cloud returns an error.

update(*membershipId*, *isModerator=None*, ***request_parameters*)
    Update a team membership, by ID.

Parameters

- **membershipId** (*basestring*) – The team membership ID.
- **isModerator** (*bool*) – Set to True to make the person a team moderator.
- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

> **Returns** A TeamMembership object with the updated Webex Teams team-membership details.
>
> **Return type** *TeamMembership*
>
> **Raises**
>
> - `TypeError` – If the parameter types are incorrect.
>
> - `ApiError` – If the Webex Teams cloud returns an error.

delete(*membershipId*)

> Delete a team membership, by ID.
>
> > **Parameters membershipId** (`basestring`) – The team membership ID.
> >
> > **Raises**
> >
> > - `TypeError` – If the parameter types are incorrect.
> >
> > - `ApiError` – If the Webex Teams cloud returns an error.

webhooks

class WebhooksAPI

> Webex Teams Webhooks API.
>
> Wraps the Webex Teams Webhooks API and exposes the API as native Python methods that return native Python objects.
>
> **list**(*max=100*, ***request_parameters*)
>
> > List all of the authenticated user's webhooks.
> >
> > This method supports Webex Teams's implementation of RFC5988 Web Linking to provide pagination support. It returns a generator container that incrementally yields all webhooks returned by the query. The generator will automatically request additional 'pages' of responses from Webex as needed until all responses have been returned. The container makes the generator safe for reuse. A new API call will be made, using the same parameters that were specified when the generator was created, every time a new iterator is requested from the container.
> >
> > > **Parameters**
> > >
> > > - **max** (*int*) – Limit the maximum number of items returned from the Webex Teams service per request.
> > >
> > > - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
> > >
> > > **Returns** A GeneratorContainer which, when iterated, yields the webhooks returned by the Webex Teams query.
> > >
> > > **Return type** GeneratorContainer
> > >
> > > **Raises**
> > >
> > > - `TypeError` – If the parameter types are incorrect.
> > >
> > > - `ApiError` – If the Webex Teams cloud returns an error.
>
> **create**(*name*, *targetUrl*, *resource*, *event*, *filter=None*, *secret=None*, ***request_parameters*)
>
> > Create a webhook.
> >
> > > **Parameters**
> > >
> > > - **name** (*basestring*) – A user-friendly name for this webhook.

- **targetUrl** (*basestring*) – The URL that receives POST requests for each event.
- **resource** (*basestring*) – The resource type for the webhook.
- **event** (*basestring*) – The event type for the webhook.
- **filter** (*basestring*) – The filter that defines the webhook scope.
- **secret** (*basestring*) – The secret used to generate payload signature.
- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns A Webhook object with the details of the created webhook.

Return type *Webhook*

Raises

- `TypeError` – If the parameter types are incorrect.
- `ApiError` – If the Webex Teams cloud returns an error.

get (*webhookId*)

Get the details of a webhook, by ID.

Parameters **webhookId** (*basestring*) – The ID of the webhook to be retrieved.

Returns A Webhook object with the details of the requested webhook.

Return type *Webhook*

Raises

- `TypeError` – If the parameter types are incorrect.
- `ApiError` – If the Webex Teams cloud returns an error.

update (*webhookId*, *name=None*, *targetUrl=None*, ***request_parameters*)

Update a webhook, by ID.

Parameters

- **webhookId** (*basestring*) – The webhook ID.
- **name** (*basestring*) – A user-friendly name for this webhook.
- **targetUrl** (*basestring*) – The URL that receives POST requests for each event.
- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns

A Webhook object with the updated Webex Teams webhook details.

Return type *Webhook*

Raises

- `TypeError` – If the parameter types are incorrect.
- `ApiError` – If the Webex Teams cloud returns an error.

delete (*webhookId*)

Delete a webhook, by ID.

Parameters **webhookId** (*basestring*) – The ID of the webhook to be deleted.

Raises

- `TypeError` – If the parameter types are incorrect.

- `ApiError` – If the Webex Teams cloud returns an error.

1.4.2 Webex Teams Data Objects

Access Token

class AccessToken
> Webex Teams Access-Token data model.

> **access_token**
> > Webex Teams access token.

> **expires_in**
> > Access token expiry time (in seconds).

> **json_data**
> > A copy of the data object's JSON data (OrderedDict).

> **refresh_token**
> > Refresh token used to request a new/refreshed access token.

> **refresh_token_expires_in**
> > Refresh token expiry time (in seconds).

> **to_dict**()
> > Convert the Webex Teams object data to a dictionary.

> **to_json**(***kwargs*)
> > Convert the Webex Teams object data to JSON.

> > Any keyword arguments provided are passed through the Python JSON encoder.

Admin Audit Event

class AdminAuditEvent
> Webex Teams Admin Audit Event data model.

> **data**
> > The event resource data.

> **actorId**
> > The personId of the person who made the change.

> **created**
> > The date and time the event took place.

> **id**
> > A unique identifier for the event.

> **json_data**
> > A copy of the data object's JSON data (OrderedDict).

> **orgId**
> > The orgId of the person who made the change.

> **to_dict**()
> > Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.

> Any keyword arguments provided are passed through the Python JSON encoder.

Attachment Action

class AttachmentAction
> Webex Attachment Actions data model

> **created**
>> The date and time the action was created.

> **id**
>> A unique identifier for the action.

> **inputs**
>> The attachment action's inputs

> **json_data**
>> A copy of the data object's JSON data (OrderedDict).

> **messageId**
>> The parent message the attachment action was performed on.

> **personId**
>> The ID of the person who performed the action.

> **roomId**
>> The ID of the room the action was performed within.

> **to_dict**()
>> Convert the Webex Teams object data to a dictionary.

> **to_json**(***kwargs*)
>> Convert the Webex Teams object data to JSON.

>> Any keyword arguments provided are passed through the Python JSON encoder.

> **type**
>> The type of action performed.

>> **Attachment action enum:** 'submit': submit filled in inputs

Event

class Event
> Webex Teams Event data model.

> **data**
>> The event's data representation.

>> This object will contain the event's resource, such as memberships or messages, at the time the event took place.

> **actorId**
>> The ID of the person who performed the action.

> **appId**
>> The ID of the application for the event.

created
> The date and time of the event.

id
> The unique identifier for the event.

json_data
> A copy of the data object's JSON data (OrderedDict).

orgId
> The ID of the organization for the event.

resource
> The type of resource in the event.
>
> **Event Resource Enum:** *messages memberships*

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

type
> The action which took place in the event.
>
> **Event Type Enum:** *created updated deleted*

Guest Issuer Token

class GuestIssuerToken
> Webex Teams Guest Issuer Token data model

> **expiresIn**

> **json_data**
> > A copy of the data object's JSON data (OrderedDict).

> **to_dict**()
> > Convert the Webex Teams object data to a dictionary.

> **to_json**(***kwargs*)
> > Convert the Webex Teams object data to JSON.
> >
> > Any keyword arguments provided are passed through the Python JSON encoder.

> **token**

License

class License
> Webex Teams License data model.

> **consumedUnits**
> > Total number of license units consumed.

> **id**
> > A unique identifier for the license.

json_data
> A copy of the data object's JSON data (OrderedDict).

name
> Name of the licensed feature.

siteType
> The type of site associated with this license.
>
> *Control Hub managed site* the site is managed by Webex Control Hub.
>
> *Linked site* the site is a linked site
>
> *Site Admin managed site* the site is managed by Site Administration

siteUrl
> The Webex Meetings site associated with this license.

subscriptionId
> The subscription ID associated with this license.
>
> This ID is used in other systems, such as Webex Control Hub.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

totalUnits
> Total number of license units allocated.

Membership

class Membership
> Webex Teams Membership data model.

created
> The date and time when the membership was created.

id
> A unique identifier for the membership.

isModerator
> Whether or not the participant is a room moderator.

isMonitor
> Whether or not the participant is a monitoring bot (deprecated).

json_data
> A copy of the data object's JSON data (OrderedDict).

personDisplayName
> The display name of the person.

personEmail
> The email address of the person.

personId
> The person ID.

personOrgId
  The organization ID of the person.

roomId
  The room ID.

to_dict()
  Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
  Convert the Webex Teams object data to JSON.

  Any keyword arguments provided are passed through the Python JSON encoder.

Message

class Message
  Webex Teams Message data model.

  attachments
    Message content attachments attached to the message.

  created
    The date and time the message was created.

  files
    Public URLs for files attached to the message.

  html
    The text content of the message, in HTML format.

    This read-only property is used by the Webex Teams clients.

  id
    The unique identifier for the message.

  json_data
    A copy of the data object's JSON data (OrderedDict).

  markdown
    The message, in Markdown format.

  mentionedGroups
    Group names for the groups mentioned in the message.

  mentionedPeople
    People IDs for anyone mentioned in the message.

  parentId
    The unique identifier for the parent message.

  personEmail
    The email address of the message author.

  personId
    The person ID of the message author.

  roomId
    The room ID of the message.

  roomType
    The type of room.

> **Room Type Enum:** *direct*: 1:1 room *group*: Group room

text
> The message, in plain text.

toPersonEmail
> 1 message.
>
> > **Type** The email address of the recipient when sending a 1

toPersonId
> 1 message.
>
> > **Type** The person ID of the recipient when sending a 1

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

updated
> The date and time the message was updated.

Organization

class Organization
> Webex Teams Organization data model.

created
> The date and time the organization was created.

displayName
> Full name of the organization.

id
> A unique identifier for the organization.

json_data
> A copy of the data object's JSON data (OrderedDict).

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

Person

class Person
> Webex Teams Person data model.

avatar
> The URL to the person''s avatar in PNG format.

created
> The date and time the person was created.

displayName
> The full name of the person.

emails
> The email addresses of the person.

firstName
> The first name of the person.

id
> A unique identifier for the person.

invitePending
> Whether or not an invite is pending for the user to complete account activation.
>
> **Person Invite Pending Enum:**
>
> > *true*: **The person has been invited to Webex Teams but has not** created an account
> >
> > *false*: An invite is not pending for this person

json_data
> A copy of the data object's JSON data (OrderedDict).

lastActivity
> The date and time of the person"s last activity within Webex Teams.

lastModified
> The date and time the person was last changed.

lastName
> The last name of the person.

licenses
> An array of license strings allocated to this person.

loginEnabled
> Whether or not the user is allowed to use Webex Teams.
>
> **Person Login Enabled Enum:** *true*: The person can log into Webex Teams
>
> > "false": The person cannot log into Webex Teams

nickName
> The nickname of the person if configured.
>
> If no nickname is configured for the person, this field will not be present.

orgId
> The ID of the organization to which this person belongs.

phoneNumbers()
> Phone numbers for the person.

roles
> An array of role strings representing the roles to which this person belongs.

status
> The current presence status of the person.
>
> **Person Status Enum:** *active*: Active within the last 10 minutes
>
> > *call*: The user is in a call
> >
> > *DoNotDisturb*: **The user has manually set their status to** "Do Not Disturb"

---

inactive: Last activity occurred more than 10 minutes ago

meeting: The user is in a meeting

***OutOfOffice*: The user or a Hybrid Calendar service has indicated** that they are "Out of Office"

***pending*: The user has never logged in; a status cannot be** determined

presenting: The user is sharing content

unknown: The user's status could not be determined

timezone
> The time zone of the person if configured.
>
> If no timezone is configured on the account, this field will not be present.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

type
> The type of person account, such as person or bot.
>
> **Person Type Enum:** *person*: Account belongs to a person *bot*: Account is a bot user *appuser*: Account is a guest user

Role

class Role
> Webex Teams Role data model.

id
> A unique identifier for the role.

json_data
> A copy of the data object's JSON data (OrderedDict).

name
> The name of the role.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

Room

class Room
> Webex Teams Room data model.

created
> The date and time the room was created.

creatorId
> The ID of the person who created this room.

id
> A unique identifier for the room.

isLocked
> Whether the room is moderated (locked) or not.

json_data
> A copy of the data object's JSON data (OrderedDict).

lastActivity
> The date and time of the room"s last activity.

ownerId
> The ID of the organization which owns this room.

teamId
> The ID for the team with which this room is associated.

title
> A user-friendly name for the room.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

type
> The room type.
>
> > **Room Type Enum:** *direct*: 1:1 room
> >
> > > *group*: Group room

Room Meeting Info

class RoomMeetingInfo
> Webex Teams Room Meeting Info data model.

callInTollFreeNumber
> The toll-free PSTN number for the room.

callInTollNumber
> The toll (local) PSTN number for the room.

json_data
> A copy of the data object's JSON data (OrderedDict).

meetingLink
> The Webex meeting URL for the room.

meetingNumber
> The Webex meeting number for the room.

roomId
> A unique identifier for the room.

sipAddress
> The SIP address for the room.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.

> Any keyword arguments provided are passed through the Python JSON encoder.

Team

class Team
> Webex Teams Team data model.

> **created**
> > The date and time the team was created.

> **creatorId**
> > The ID of the person who created the team.

> **id**
> > A unique identifier for the team.

> **json_data**
> > A copy of the data object's JSON data (OrderedDict).

> **name**
> > A user-friendly name for the team.

> **to_dict**()
> > Convert the Webex Teams object data to a dictionary.

> **to_json**(***kwargs*)
> > Convert the Webex Teams object data to JSON.

> > Any keyword arguments provided are passed through the Python JSON encoder.

Team Membership

class TeamMembership
> Webex Teams Team-Membership data model.

> **created**
> > The date and time when the team membership was created.

> **id**
> > A unique identifier for the team membership.

> **isModerator**
> > Whether or not the participant is a team moderator.

> **json_data**
> > A copy of the data object's JSON data (OrderedDict).

> **personDisplayName**
> > The display name of the person.

> **personEmail**
> > The email address of the person.

> **personId**
> > The person ID.

personOrgId
>   The organization ID of the person.

teamId
>   The team ID.

to_dict()
>   Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
>   Convert the Webex Teams object data to JSON.

>   Any keyword arguments provided are passed through the Python JSON encoder.

Webhook

class Webhook
>   Webex Teams Webhook data model.

>   **appId**
>   >   The ID of the application that created the webhook.

>   **created**
>   >   The date and time the webhook was created.

>   **createdBy**
>   >   The ID of the person that created the webhook.

>   **event**
>   >   The event type for the webhook.

>   >   **Webhook Event Type Enum:** *all*: Subscribe to all events *created*: An object was created *updated*: An object was updated *deleted*: An object was deleted

>   **filter**
>   >   The filter that defines the webhook scope.

>   **id**
>   >   A unique identifier for the webhook.

>   **json_data**
>   >   A copy of the data object's JSON data (OrderedDict).

>   **name**
>   >   A user-friendly name for the webhook.

>   **orgId**
>   >   The ID of the organization that owns the webhook.

>   **ownedBy**
>   >   Indicates if the webhook is owned by the *org* or the *creator*.

>   >   Webhooks owned by the creator can only receive events that are accessible to the creator of the webhook. Those owned by the organization will receive events that are visible to anyone in the organization.

>   >   **Webhook Owned By Enum:** *org creator*

>   **resource**
>   >   The resource type for the webhook.

>   >   Creating a webhook requires 'read' scope on the resource the webhook is for.

> **Webhook Resource Enum:** *all*: Subscribe to all resources *memberships*: The Memberships resource *messages*: The Messages resource *rooms*: The Rooms resource

secret
> The secret used to generate payload signature.

status
> The status of the webhook.

> **Webhook Status Enum:** *active*: The webhook is active *inactive*: The webhook is inactive

targetUrl
> The URL that receives POST requests for each event.

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.

> Any keyword arguments provided are passed through the Python JSON encoder.

Webhook Event

class WebhookEvent
> Webex Teams Webhook-Events data model.

data
> The event resource data.

actorId
> The ID of the person that caused the webhook to be sent.

appId
> The ID of the application that created the webhook.

createdBy
> The ID of the person that created the webhook.

event
> The event type for the webhook.

> **Webhook Event Type Enum:** *created*: An object was created *updated*: An object was updated *deleted*: An object was deleted

filter
> The filter that defines the webhook scope.

id
> A unique identifier for the webhook.

json_data
> A copy of the data object's JSON data (OrderedDict).

name
> A user-friendly name for the webhook.

orgId
> The ID of the organization that owns the webhook.

ownedBy
> Indicates if the webhook is owned by the *org* or the *creator*.

Webhooks owned by the creator can only receive events that are accessible to the creator of the webhook. Those owned by the organization will receive events that are visible to anyone in the organization.

> **Webhook Owned By Enum:** *org creator*

resource
> The resource type for the webhook.
>
> Creating a webhook requires 'read' scope on the resource the webhook is for.
>
> **Webhook Resource Enum:** *memberships*: The Memberships resource *messages*: The Messages resource *rooms*: The Rooms resource

status
> The status of the webhook.
>
> **Webhook Status Enum:** *active*: The webhook is active *inactive*: The webhook is inactive

to_dict()
> Convert the Webex Teams object data to a dictionary.

to_json(***kwargs*)
> Convert the Webex Teams object data to JSON.
>
> Any keyword arguments provided are passed through the Python JSON encoder.

1.4.3 Exceptions

exception webexteamssdkException
> Bases: `Exception`
>
> Base class for all webexteamssdk package exceptions.

exception AccessTokenError
> Bases: `webexteamssdk.exceptions.webexteamssdkException`
>
> Raised when an incorrect Webex Teams Access Token has been provided.

exception ApiError
> Bases: `webexteamssdk.exceptions.webexteamssdkException`
>
> Errors returned in response to requests sent to the Webex Teams APIs.
>
> Several data attributes are available for inspection.
>
> **response = None**
> > The `requests.Response` object returned from the API call.
>
> **request = None**
> > The `requests.PreparedRequest` of the API call.
>
> **status_code = None**
> > The HTTP status code from the API response.
>
> **status = None**
> > The HTTP status from the API response.
>
> **description = None**
> > A description of the HTTP Response Code from the API docs.
>
> **details = None**
> > The parsed JSON details from the API response.

message = None
> The error message from the parsed API response.

tracking_id = None
> The Webex Tracking ID from the response.

exception MalformedResponse
> Bases: `webexteamssdk.exceptions.webexteamssdkException`

> Raised when a malformed response is received from Webex Teams.

exception RateLimitError
> Bases: `webexteamssdk.exceptions.ApiError`

> Webex Teams Rate-Limit exceeded Error.

> Raised when a rate-limit exceeded message is received and the request **will not** be retried.

> **retry_after = None**
> > The *Retry-After* time period (in seconds) provided by Webex Teams.

> > Defaults to 15 seconds if the response *Retry-After* header isn't present in the response headers, and defaults to a minimum wait time of 1 second if Webex Teams returns a *Retry-After* header of 0 seconds.

1.4.4 Warnings

exception webexteamssdkWarning
> Bases: `webexteamssdk.exceptions.webexteamssdkException,` `Warning`

> Base class for all webexteamssdk warnings.

exception ApiWarning
> Bases: `webexteamssdk.exceptions.webexteamssdkWarning,` `webexteamssdk.exceptions.ApiError`

> Warnings raised from API responses received from the Webex APIs.

> Several data attributes are available for inspection.

exception RateLimitWarning
> Bases: `webexteamssdk.exceptions.ApiWarning,` `webexteamssdk.exceptions.RateLimitError`

> Webex Teams rate-limit exceeded warning.

> Raised when a rate-limit exceeded message is received and the request will be retried.

1.4.5 Cards and Buttons

class AdaptiveCard
> Adaptive Card data model.

> Note: Webex Teams currently supports version 1.1 of adaptive cards and thus only features from that release are supported in this abstraction.

> **type = 'AdaptiveCard'**

> **schema = 'http://adaptivecards.io/schemas/adaptive-card.json'**

> **version = '1.1'**

to_dict()
Serialize the component into a Python dictionary.

The to_dict() method recursively serializes the object's data into a Python dictionary.

> **Returns** Dictionary representation of this component.
>
> **Return type** dict

Components

class Image
Adaptive Card Image component.

__init__(*url*, *altText=None*, *backgroundColor=None*, *height=None*, *horizontalAlignment=None*, *selectAction=None*, *size=None*, *style=None*, *width=None*, *separator=None*, *spacing=None*, *id=None*)
Initialize a new image component.

> **Parameters**
>
> - **url** (`str`) – The URL to the image
> - **altText** (`str`) – Alternative text describing the image
> - **backgroundColor** (`str`) – Background color for transparent images.
> - **height** (`str,` `BlockElementHeight`) – Height of the image either as a pixel value(i.e. '50px') or as an instance of BlockElementHeight
> - **horizontalAlignmnet** (`HorizontalAlignment`) – Controls how the component is positioned within its parent.
> - **selectAction** (`OpenUrl,` `Submit`) – Option that is carried out when the card is selected.
> - **size** (`ImageSize`) – Controls the approximate size of the image.
> - **style** (`ImageStyle`) – The display style of this image.
> - **width** (`str`) – Width of the image as a pixel value (i.e. '50px')
> - **separator** (`bool`) – Draw a separating line when set to true
> - **spacing** (`Spacing`) – Specify the spacing of this component
> - **id** (`str`) – The id of this component

type = 'Image'

class TextBlock
Adaptive Card Text Block component.

__init__(*text*, *color=None*, *horizontalAlignment=None*, *isSubtle=None*, *maxLines=None*, *size=None*, *weight=None*, *wrap=None*, *separator=None*, *spacing=None*, *id=None*)
Initialize a new TextBlock component.

type = 'TextBlock'

class Column
Adaptive Card Column component.

type = 'Column'

class Fact
Adaptive Card Fact component.

class Choice

Options

class VerticalContentAlignment
 An enumeration.

 TOP = 'top'

 CENTER = 'center'

 BOTTOM = 'bottom'

class Colors
 An enumeration.

 DEFAULT = 'default'

 DARK = 'dark'

 LIGHT = 'light'

 ACCENT = 'accent'

 GOOD = 'good'

 WARNING = 'warning'

 ATTENTION = 'attention'

class HorizontalAlignment
 An enumeration.

 LEFT = 'left'

 CENTER = 'center'

 RIGHT = 'right'

class FontSize
 An enumeration.

 DEFAULT = 'default'

 SMALL = 'small'

 MEDIUM = 'medium'

 LARGE = 'large'

 EXTRA_LARGE = 'extraLarge'

class FontWeight
 An enumeration.

 DEFAULT = 'default'

 LIGHTER = 'lighter'

 BOLDER = 'bolder'

class BlockElementHeight
 An enumeration.

 AUTO = 'auto'

 STRETCH = 'auto'

class Spacing
   An enumeration.

   DEFAULT = 'default'

   NONE = 'none'

   SMALL = 'small'

   MEDIUM = 'medium'

   LARGE = 'large'

   EXTRA_LARGE = 'extraLarge'

   PADDING = 'padding'

class ImageSize
   An enumeration.

   AUTO = 'auto'

   STRETCH = 'stretch'

   SMALL = 'small'

   MEDIUM = 'medium'

   LARGE = 'large'

class ImageStyle
   An enumeration.

   DEFAULT = 'default'

   PERSON = 'person'

class ContainerStyle
   An enumeration.

   DEFAULT = 'default'

   EMPHASIS = 'emphasis'

class TextInputStyle
   An enumeration.

   TEXT = 'text'

   TEL = 'tel'

   URL = 'url'

   EMAIL = 'email'

class ChoiceInputStyle
   An enumeration.

   COMPACT = 'compact'

   EXPANDED = 'expanded'

Container

class Container
   Adaptive Card Container component.

---

    type = 'Container'

class ColumnSet

    Adaptive Card Column Set component.

    type = 'ColumnSet'

class FactSet

    Adaptive Card Fact Set component.

    type = 'FactSet'

class ImageSet

    Adaptive Card Image Set component.

    type = 'ImageSet'

Inputs

class Text

    Adaptive Card Text component.

    type = 'Input.Text'

class Number

    Adaptive Card Number component.

    type = 'Input.Number'

class Date

    Adaptive Card Date component.

    type = 'Input.Date'

class Time

    Adaptive Card Time component.

    type = 'Input.Time'

class Toggle

    Adaptive Card Toggle component.

    type = 'Input.Toggle'

class Choices

    Adaptive Card Choice Set component.

    type = 'Input.ChoiceSet'

Actions

class OpenUrl(*url*, *title=None*, *iconURL=None*)

    Open URL Action.

    type = 'Action.OpenUrl'

class Submit(*data=None*, *title=None*, *iconURL=None*)

    Submit Action.

    type = 'Action.Submit'

class ShowCard(*card=None*, *title=None*, *iconURL=None*)

    Show Card Action.

```
        type = 'Action.ShowCard'
```

Copyright (c) 2016-2020 Cisco and/or its affiliates.

1.5 Cards and Buttons

Webex supports AdaptiveCards to allow new levels of interactivity for bots and integrations. You can read more about how cards and buttons work in the official guide.

In this guide I want to cover the abstraction built into the webexteamssdk that lets you author adaptive cards in pure python without having to touch the underlying JSON of an adaptive card.

1.5.1 Sending a card

Lets dive into a simple example that sends a card to a room

```python
from webexteamssdk import WebexTeamsAPI
from webexteamssdk.models.cards.card import AdaptiveCard
from webexteamssdk.models.cards.inputs import Text, Number
from webexteamssdk.models.cards.components import TextBlock
from webexteamssdk.models.cards.actions import Submit

greeting = TextBlock("Hey hello there! I am a adaptive card")
first_name = Text('first_name', placeholder="First Name")
age = Number('age', placeholder="Age")

submit = Submit(title="Send me!")

card = AdaptiveCard(body=[greeting, first_name, age], actions=[submit])

api = WebexTeamsAPI()
api.messages.create(text="fallback", roomId="...", attachments=[card])
```

The message we send with this code then looks like this in our Webex space client:
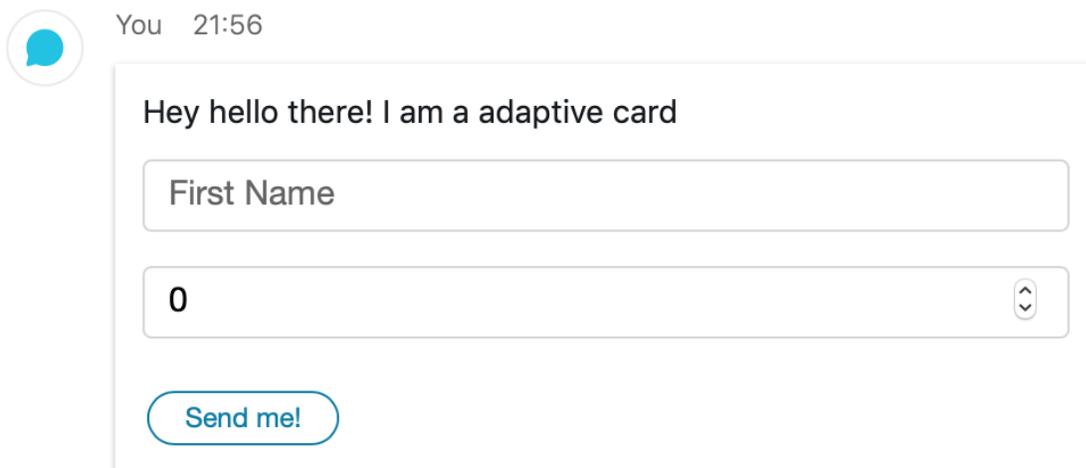
1.5.2 Processing a card action

Adaptive card interactions are treated as "attachment actions". Once user interacts with your card and submits an action, your app will receive a webhook from Webex. You must setup a webhook in advance with `resource = "attachmentActions"` and `event = "created"`.

Webhook payload will contain a JSON:

```
{
    "resource": "attachmentActions",
    "event": "created",
    "data": {
        "id": "XYXYXY",
        "type": "submit"
    }
}
```

Extract attachment action ID from `['data']['id']` and use attachment_actions.get() to get full information about user action and any submitted data.

```
action = api.attachment_actions.get(webhookJson['data']['id'])

first_name = action.inputs['first_name']
age = action.inputs['age']
```

The Development Community

Interested in contributing to the project? Please review our community's Code of Conduct and then check out the Contributing page for info to help you get started.

2.1 Contributor Covenant Code of Conduct

2.1.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

2.1.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment

- Publishing others' private information, such as a physical or electronic address, without explicit permission

- Other conduct which could reasonably be considered inappropriate in a professional setting

2.1.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

2.1.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

2.1.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at chrlunsf@cisco.com. The project team will review and investigate all complaints, and will respond in a way that it deems appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

2.1.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4.

2.2 Contributing

webexteamssdk is a community development project. Feedback, thoughts, ideas, and code contributions are most welcome!

2.2.1 How to contribute Feedback, Issues, Thoughts and Ideas

Please use the issues page to report issues or post ideas for enhancement.

Join our webexteamssdk - Webex Teams SDK - Python Community Contributors Webex Teams space to join the conversation with other contributors to this project.

2.2.2 Interested in Contributing Code?

Developer Scripts

We have created some scripts to automate everyday actions needed when working on the project. Please see the script directory, and it's README for more information.

Notes on the Test Suite

To test all the API endpoints, the account that you use for testing must be an *admin* user for your Webex Teams Organization. Additionally, you should know that that the testing process creates some test people, rooms, messages, teams, and etc. as part of executing the test suite. We strongly recommend *NOT* running the test suite using your personal Webex Teams account (not that you can't; it's just that you probably don't want it cluttering your account with all these test artifacts).

If you cannot create a test account with *admin* privileges or configure your environment to run the test suite locally, you may always submit your code via a pull request. We will test your code before merging and releasing the changes.

Contributing Code

1. Check for open issues or create a new *issue* for the item you want to work on and make sure to comment and let us know that you are working on it.

2. Fork a copy of the repository and clone your forked repository to your development environment.

3. Run `script/setup` to install the development dependencies and setup your environment.

4. Configure the following environment variables in your development environment:

    - `WEBEX_TEAMS_ACCESS_TOKEN` - Your test account's Webex Teams access token.

5. Add your code to your forked repository.

    If you are creating some new feature or functionality (excellent!), please also write a test to verify that your code works as expected.

6. We follow PEP8 reasonably strictly for this project. Please make sure your code passes the linter.

    Run `script/test lint` or simply run `flake8` from the project root.

7. Commit your changes.

8. Submit a pull request.

Running the Test Suite Locally

Configure the following environment variables in your development environment:

- `WEBEX_TEAMS_ACCESS_TOKEN` - Your test account's Webex Teams access token.

- `WEBEX_TEAMS_TEST_DOMAIN` - The test suite creates some users as part of the testing process. The test suite uses this domain name as the e-mail suffix of for the user's e-mail addresses.

- `WEBEX_TEAMS_TEST_ID_START` - The test suite uses this integer as the starting number for creating test user accounts (example: "test42@domain.com").

- `WEBEX_TEAMS_TEST_FILE_URL` - Configure this environment variable with a URL referencing a file that can be downloaded and posted to Webex Teams as part of the testing process.

Example:

```bash
#!/usr/bin/env bash
export WEBEX_TEAMS_ACCESS_TOKEN="<test account's access token>"
export WEBEX_TEAMS_TEST_DOMAIN="domain.com"
export WEBEX_TEAMS_TEST_ID_START=42
export WEBEX_TEAMS_TEST_FILE_URL="https://www.webex.com/content/dam/wbx/us/images/dg-
↪integ/teams_icon.png"
```

Ensure your code passes all of the default tests. Run `script/test` and ensure all tests execute successfully.

General Information about the Webex Teams Service

3.1 What is Webex Teams?

"Webex Teams is where all your work lives. Bring your teams together in a place that makes it easy to keep people and work connected."

Visit the official Webex Teams website for more information and to create a free account!

3.2 Webex Teams for Developers

Leveraging the Webex Teams APIs and developing on top of the Webex Teams cloud is easy. Signup for a free account and then head over to the Webex Teams for Developers website to learn more.

Copyright (c) 2016-2020 Cisco and/or its affiliates.

Index

Symbols