cisco Live!

Let’s go

The bridge to possible

Agile Management of Webex Calling

There's an API for that

Johannes Krohn, Principal Technical Marketing Engineer

Agenda

- Why APIs?
- Coverage/Capabilities
- Getting started
- Use cases/Examples
- Closing

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Why APIs?

API, What, Where, and Why?

- Definition: .. is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components. .. Documentation for the API is usually provided to facilitate usage.”[1]
- APIs
  - Enabler for open systems integration
  - Universally available
  - Unleash developer innovation

[1]Wikipedia

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

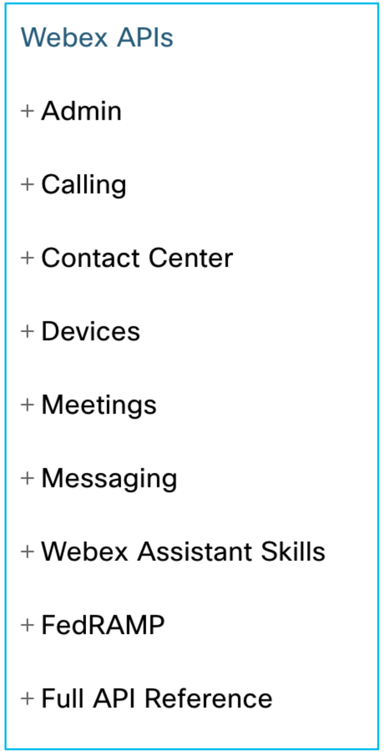
Webex Calling Provisioning Methods

	Control Hub	CSV	API
Ease of Use	+ + +	+ +	+
Speed	+	+ + +	+ + +
Customization		+	+ + +

Coverage / Capabilities

Webex APIs

- Documentation: http://developer.webex.com
- Various APIs available:
- Admin (licenses, locations, memberships, people, ..)
- Calling (call control, locations, people, org/location settings, ...)
- Devices (configuration, places, workspace locations, xAPI, ...)
- Meetings (invitees, participants, preferences, ...)
- ...
- OAuth access token used for authorization



© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Webex Calling API capabilities

- Provisioning
  - Users (incl. calling entitlements), locations (r/o), call pickups, call queues, hunt groups, auto attendant, call parks, schedules, voice messaging settings, …
  - person settings: barge, call forwarding, call intercept, call recording, caller ID, voicemail settings, …
  - Coverage continuously growing*

*Check https://help.webex.com/en-us/article/rdmb0/What's-new-in-Webex-Calling and https://developer.webex.com/ for updates

- Call Control
  - Dial, answer, reject, hangup, hold/resume, divert, transfer, park/retrieve, start/stop/pause/resume recording, DTMF, push, pickup, barge
- Webhook Notifications/Events
  - Voice messages
  - Call events

References:
https://developer.webex.com/docs/webex-calling
https://developer.webex.com/blog/calling-apis-overview

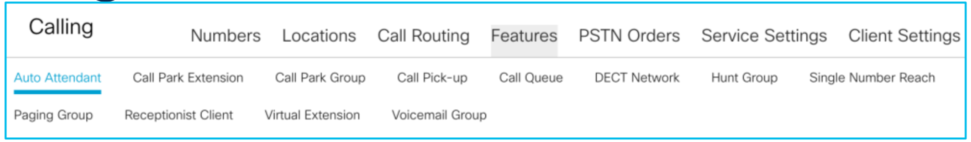
© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Webex Calling Provisioning APIs

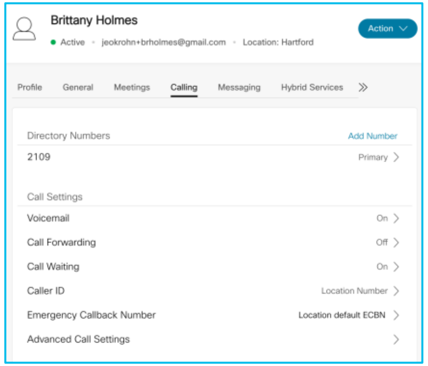
- Locations, https://developer.webex.com/docs/api/v1/locations
  - List locations
  - Get location details
  - Create/Update locations
  - Delete: not possible
- People, https://developer.webex.com/docs/api/v1/people
  - List
  - CRUD
  - `callingData` parameter to access calling data*

*Note: US TNs use 10D format instead of +E.164

Webex Calling Provisioning APIs



- Organization Settings, https://developer.webex.com/docs/api/v1/webex-calling-organization-settings
  - Calling features found in Feature tab in Control Hub
- Person Settings, https://developer.webex.com/docs/api/v1/webex-calling-person-settings
  - Settings found in person's Calling tab in Control Hub



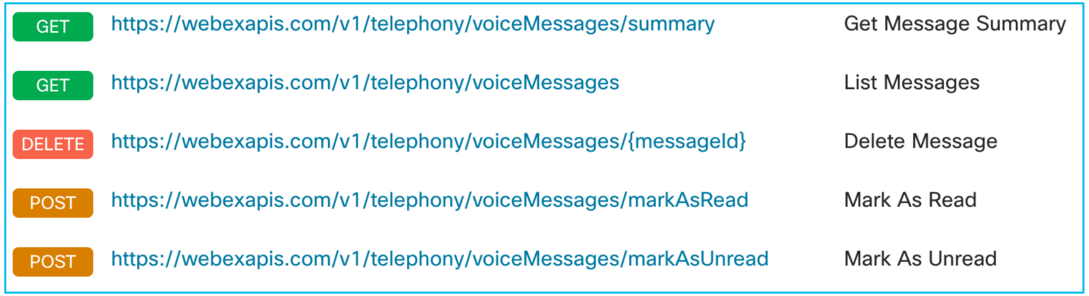
Brittany Holmes
● Active • jeokrohn+brholmes@gmail.com • Location: Hartford

Directory Numbers
2109 Primary

Call Settings
Voicemail On
Call Forwarding Off
Call Waiting On
Caller ID Location Number
Emergency Callback Number Location default ECBN
Advanced Call Settings

Webex Calling Voice Messaging APIs

- Voice Messaging, https://developer.webex.com/docs/api/v1/webex-calling-voice-messaging
  - Handle voicemail and MWI
  - User access only; no admin access
  - Message summary, list messages, delete message, mark read/unread

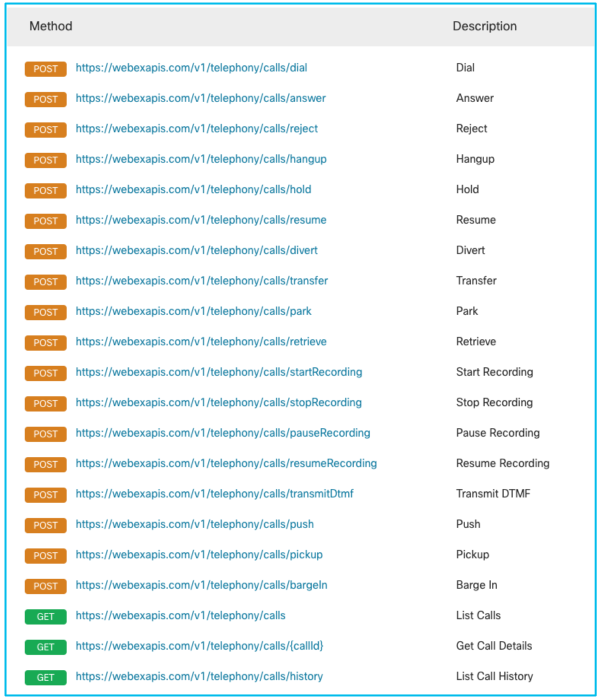


GET	https://webexapis.com/v1/telephony/voiceMessages/summary	Get Message Summary
GET	https://webexapis.com/v1/telephony/voiceMessages	List Messages
DELETE	https://webexapis.com/v1/telephony/voiceMessages/{messageId}	Delete Message
POST	https://webexapis.com/v1/telephony/voiceMessages/markAsRead	Mark As Read
POST	https://webexapis.com/v1/telephony/voiceMessages/markAsUnread	Mark As Unread

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Webex Calling Call Controls

- Actions
  - Dial, answer, reject, hangup, hold/resume, divert, transfer, park/retrieve, start/stop/pause/resume recording, DTMF, push, pickup, barge
- Management
  - List, get details, call history
  - List/Details use common call object
- Requires user access token
  - No org level (admin) operations



Method		Description
POST	https://webexapis.com/v1/telephony/calls/dial	Dial
POST	https://webexapis.com/v1/telephony/calls/answer	Answer
POST	https://webexapis.com/v1/telephony/calls/reject	Reject
POST	https://webexapis.com/v1/telephony/calls/hangup	Hangup
POST	https://webexapis.com/v1/telephony/calls/hold	Hold
POST	https://webexapis.com/v1/telephony/calls/resume	Resume
POST	https://webexapis.com/v1/telephony/calls/divert	Divert
POST	https://webexapis.com/v1/telephony/calls/transfer	Transfer
POST	https://webexapis.com/v1/telephony/calls/park	Park
POST	https://webexapis.com/v1/telephony/calls/retrieve	Retrieve
POST	https://webexapis.com/v1/telephony/calls/startRecording	Start Recording
POST	https://webexapis.com/v1/telephony/calls/stopRecording	Stop Recording
POST	https://webexapis.com/v1/telephony/calls/pauseRecording	Pause Recording
POST	https://webexapis.com/v1/telephony/calls/resumeRecording	Resume Recording
POST	https://webexapis.com/v1/telephony/calls/transmitDtmf	Transmit DTMF
POST	https://webexapis.com/v1/telephony/calls/push	Push
POST	https://webexapis.com/v1/telephony/calls/pickup	Pickup
POST	https://webexapis.com/v1/telephony/calls/bargeIn	Barge In
GET	https://webexapis.com/v1/telephony/calls	List Calls
GET	https://webexapis.com/v1/telephony/calls/{callId}	Get Call Details
GET	https://webexapis.com/v1/telephony/calls/history	List Call History

Webhook Notifications/Events

- Webhook API to manage webhooks: https://developer.webex.com/docs/api/v1/webhooks
- Resource: `telephony_calls`
- Events: `created`, `updated`, `deleted`

https://developer.webex.com/docs/webhooks

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Telephony_call event example

```
{
  "id": "Y2lzY2…wMTc5",
  "name": "d9c193c3-4787-4726-b9fa-6acff173e15a",
  "targetUrl": "https://c780-149-249-133-109.ngrok.io/callevent/Y2lzY29...ZWIzZGE",
  "resource": "telephony_calls",
  "event": "created",
  "orgId": "Y2lz…mUzZTc",
  "createdBy": "Y2lz…ZGE",
  "appId": "Y2lzY29zc…5NzZlZWQ0ODM1",
  "ownedBy": "creator",
  "status": "active",
  "created": "2022-03-18T14:53:51.669Z",
  "actorId": "Y2lzY2…2FjZWIzZGE",
  "data": {
    "eventType": "received",
    "eventTimestamp": "2022-03-18T14:54:02.442Z",
    "callId": "Y2lzY2…AxNDYxOTow",
    "callSessionId": "Zjg1OWExYTYtNDI5NS00OTU0LWEwYzktMDY0MjFjOTY5Mzk3",
    "personality": "terminator",
    "state": "alerting",
    "remoteParty": {
      "name": "Henry Green",
      "number": "7101",
      "personId": "Y2lzY29z…zNTg",
      "privacyEnabled": false,
      "callType": "location"
    },
    "appearance": 1,
    "created": "2022-03-18T14:54:02.440Z"
  }
}
```

Webhook ID

Webhook name

Target URL

Resource "telephony_calls" → call event

Created → new call

Id of app used to create the webhook

Information about the actual call

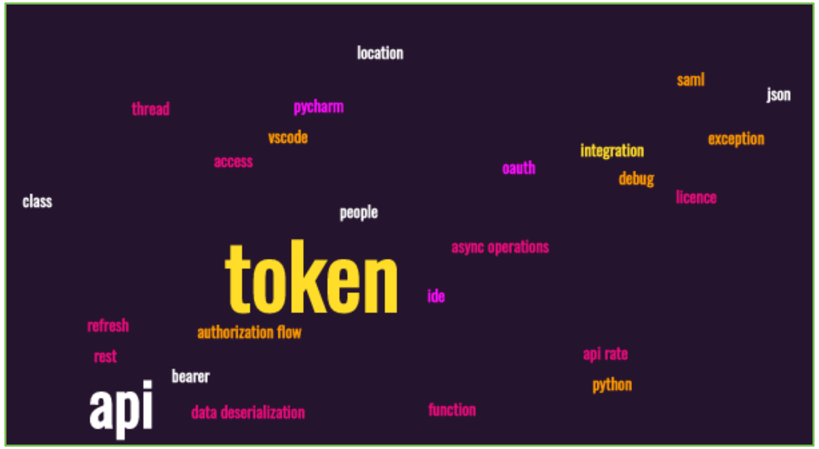
Webex Calling APIs Overview

	PROVISIONING	CALL CONTROL	ANALYTICS & REPORTING
Customer Journey	Setup, Onboard, Manage	Call, Meet, Collaborate	Achieve Customer Success
Representative Tasks	• Manage users, phone #s, locations, & services • Assign licenses • Create and manage location features	• Place, answer, hang up calls • Stop / start / pause recording • Transmit DTMF digits • List active calls / get history	• Detailed call records • Onboarding, usage, & quality reporting • Automated reporting setup
Sample Solutions	• Installation, activation, & onboarding • Ongoing services management & care • Self-service via partner portal	• Custom enterprise calling integrations • Cloud business platform integration • Custom app development	• User training & adoption services • Business process design & optimization • Vertical solutions design & oversight

Getting Started

Using Webex APIs

- Documentation at: https://developer.webex.com/
- But: Steep learning curve
- A lot of concepts to master
- SDK helps to abstract from the "dirty details"



Developer Sandbox

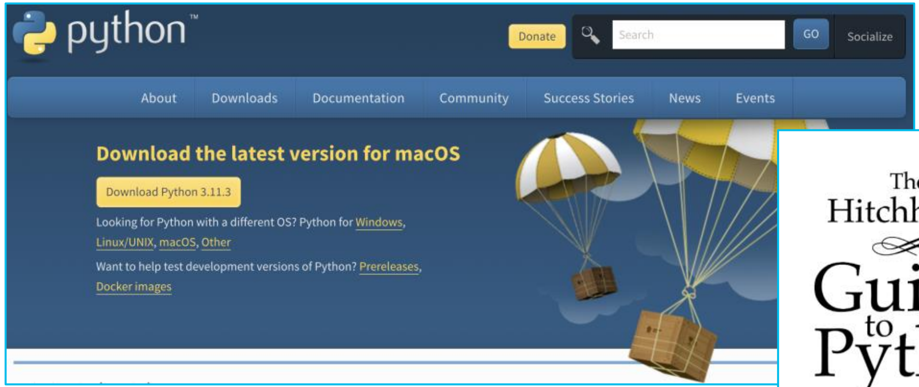
- Sandbox
  - playground to test API calls
  - Avoid impact on production org
- Limited to 10 users
- Allows to test capabilities not available w/ Webex free plans
- No Cisco PSTN
  - Can add Local Gateway for PSTN access
  - Working on a solution to get PSTN added to sandbox

https://developer.webex.com/docs/developer-sandbox-guide

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Installing Python

- Installers are are available at https://www.python.org/downloads/
- Mac tip: install Python via Homebrew: https://docs.python-guide.org/starting/install3/osx/
  - Avoids issues with GNU readline (for example when using https://pypi.org/project/cmd2/)



Download the latest version for macOS

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

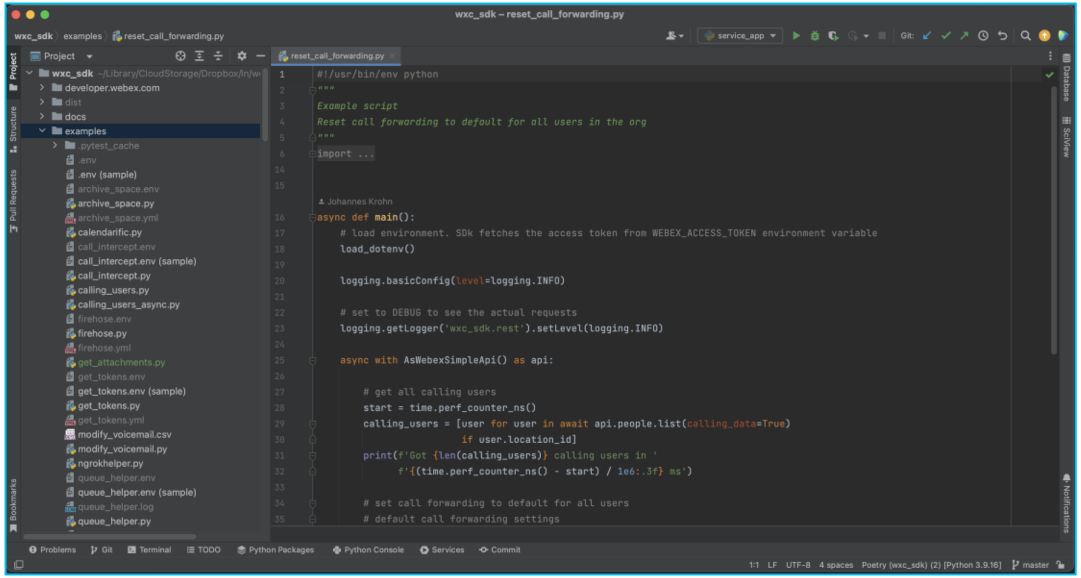
The Hitchhiker's Guide to Python
Star
24,679
Search the doc
Master Real-World Python Skills
With a Community of Experts
Level Up With Unlimited Access to Our Vast Library of Python Tutorials and Video Lessons
Watch Now »

Installing Python 3 on Mac OS X

Tools

IDE – Integrated Development Environment

- Helps to develop and test your application
- Features
  - GUI
  - Editor
  - Build automation
  - Syntax highlighting
  - Debugger
  - Integration w/ revision control system (e.g. Git)
  - …

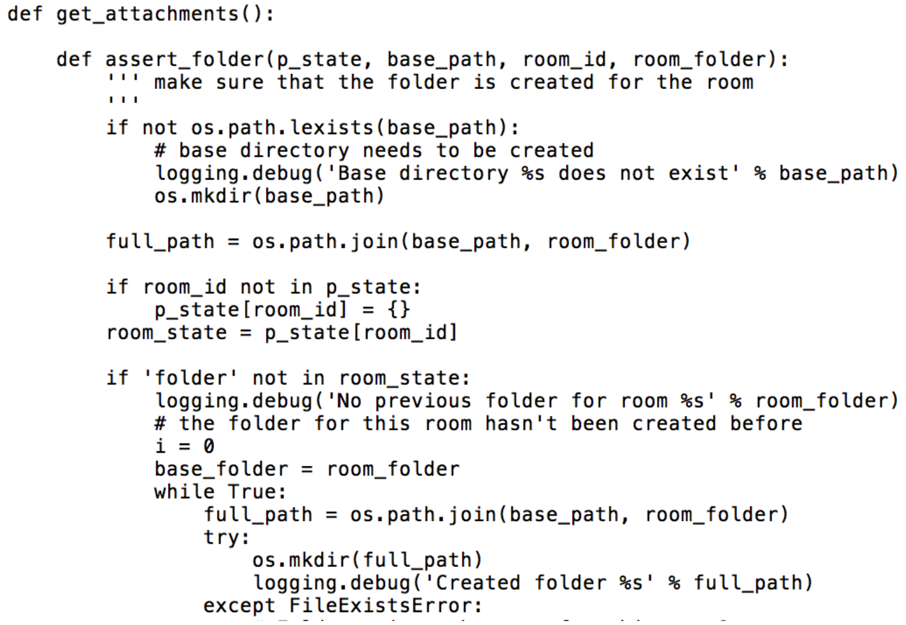


```
wxc_sdk ~/Library/CloudStorage/Dropbox/ln/w
  developer.webex.com
  dist
  docs
  examples
    .pytest_cache
    .env
    .env (sample)
    archive_space.env
    archive_space.py
    archive_space.yml
    calendarific.py
    call_intercept.env
    call_intercept.env (sample)
    call_intercept.py
    calling_users.py
    calling_users_async.py
    firehose.env
    firehose.py
    firehose.yml
    get_attachments.py
    get_tokens.env
    get_tokens.env (sample)
    get_tokens.py
    get_tokens.yml
    modify_voicemail.csv
    modify_voicemail.py
    ngrokhelper.py
    queue_helper.env
    queue_helper.env (sample)
    queue_helper.log
    queue_helper.py
```

```python
#!/usr/bin/env python
"""
Example script
Reset call forwarding to default for all users in the org
"""
import ...


Johannes Krohn
async def main():
    # load environment. SDk fetches the access token from WEBEX_ACCESS_TOKEN environment variable
    load_dotenv()

    logging.basicConfig(level=logging.INFO)

    # set to DEBUG to see the actual requests
    logging.getLogger('wxc_sdk.rest').setLevel(logging.INFO)

    async with AsWebexSimpleApi() as api:

        # get all calling users
        start = time.perf_counter_ns()
        calling_users = [user for user in await api.people.list(calling_data=True)
                         if user.location_id]
        print(f'Got {len(calling_users)} calling users in '
              f'{(time.perf_counter_ns() - start) / 1e6:.3f} ms')

        # set call forwarding to default for all users
        # default call forwarding settings
```

Syntax Highlighting

- What Do you prefer?
- This?

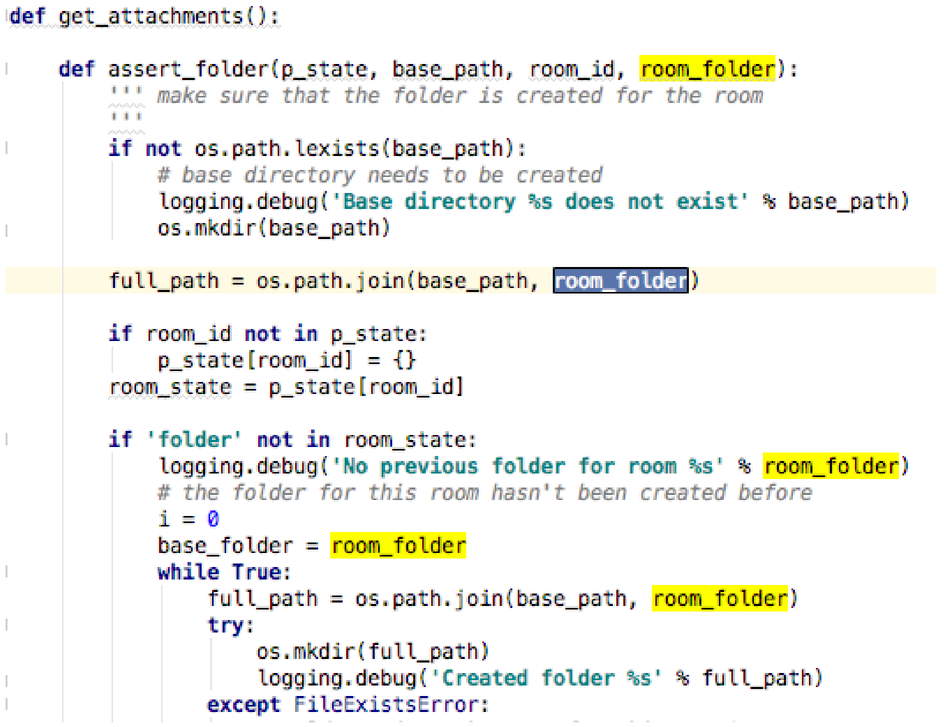


```
def get_attachments():

    def assert_folder(p_state, base_path, room_id, room_folder):
        ''' make sure that the folder is created for the room
        '''
        if not os.path.lexists(base_path):
            # base directory needs to be created
            logging.debug('Base directory %s does not exist' % base_path)
            os.mkdir(base_path)

        full_path = os.path.join(base_path, room_folder)

        if room_id not in p_state:
            p_state[room_id] = {}
        room_state = p_state[room_id]

        if 'folder' not in room_state:
            logging.debug('No previous folder for room %s' % room_folder)
            # the folder for this room hasn't been created before
            i = 0
            base_folder = room_folder
            while True:
                full_path = os.path.join(base_path, room_folder)
                try:
                    os.mkdir(full_path)
                    logging.debug('Created folder %s' % full_path)
                except FileExistsError:
```

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Syntax Highlighting

- What Do you prefer?
- Or this?

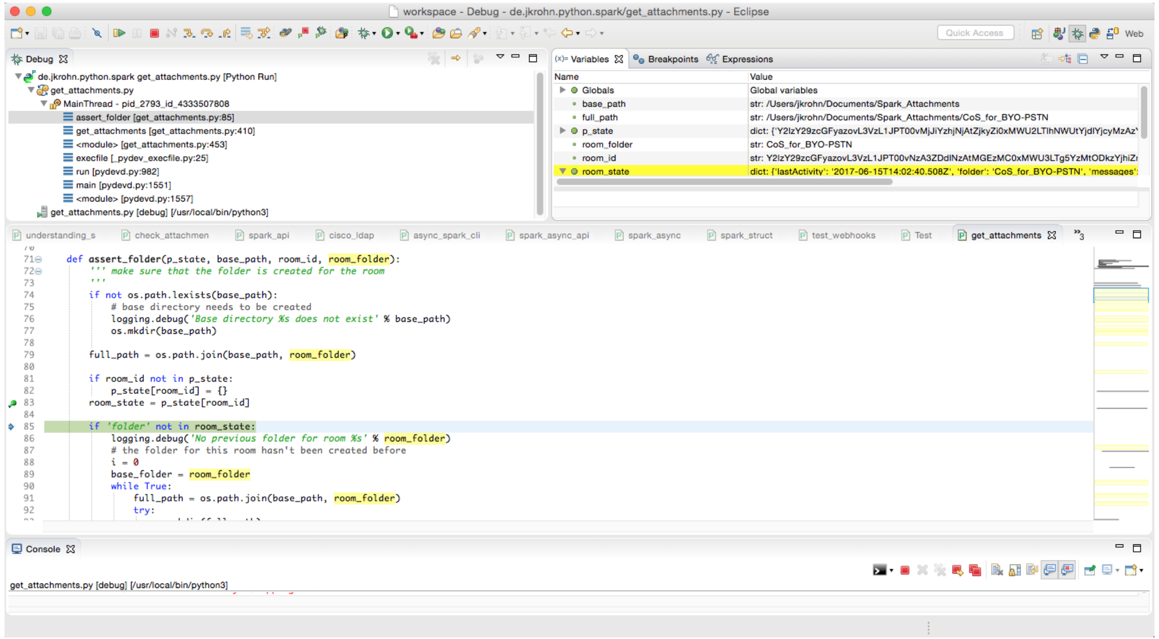


```
def get_attachments():

    def assert_folder(p_state, base_path, room_id, room_folder):
        ''' make sure that the folder is created for the room
        '''
        if not os.path.lexists(base_path):
            # base directory needs to be created
            logging.debug('Base directory %s does not exist' % base_path)
            os.mkdir(base_path)

        full_path = os.path.join(base_path, room_folder)

        if room_id not in p_state:
            p_state[room_id] = {}
        room_state = p_state[room_id]

        if 'folder' not in room_state:
            logging.debug('No previous folder for room %s' % room_folder)
            # the folder for this room hasn't been created before
            i = 0
            base_folder = room_folder
            while True:
                full_path = os.path.join(base_path, room_folder)
                try:
                    os.mkdir(full_path)
                    logging.debug('Created folder %s' % full_path)
                except FileExistsError:
```

Live Debugger

- Live Debugger allows to
  - Set breakpoints
  - Check variables
  - Evaluate expressions

→ Essential for effective SW development


workspace - Debug - de.jkrohn.python.spark/get_attachments.py - Eclipse

Debug

```
de.jkrohn.python.spark get_attachments.py [Python Run]
  get_attachments.py
    MainThread - pid_2793_id_4333507808
      assert_folder [get_attachments.py:85]
      get_attachments [get_attachments.py:410]
      <module> [get_attachments.py:453]
      execfile [_pydev_execfile.py:25]
      run [pydevd.py:982]
      main [pydevd.py:1551]
      <module> [pydevd.py:1557]
  get_attachments.py [debug] [/usr/local/bin/python3]
```

Variables | Breakpoints | Expressions

Name	Value
Globals	Global variables
base_path	str: /Users/jkrohn/Documents/Spark_Attachments
full_path	str: /Users/jkrohn/Documents/Spark_Attachments/CoS_for_BYO-PSTN
p_state	dict: {'Y2lzY29zcGFyazovL3VzL1JPT00vMjJiYzhjNjAtZjkyZi0xMWU2LTlhNWUtYjdlYjcyMzAz
room_folder	str: CoS_for_BYO-PSTN
room_id	str: Y2lzY29zcGFyazovL3VzL1JPT00vNzA3ZDdlNzAtMGEzMC0xMWU3LTg5YzMtODkzYjhiZı
room_state	dict: {'lastActivity': '2017-06-15T14:02:40.508Z', 'folder': 'CoS_for_BYO-PSTN', 'messages':

```
71  def assert_folder(p_state, base_path, room_id, room_folder):
72      ''' make sure that the folder is created for the room
73      '''
74      if not os.path.lexists(base_path):
75          # base directory needs to be created
76          logging.debug('Base directory %s does not exist' % base_path)
77          os.mkdir(base_path)
78
79      full_path = os.path.join(base_path, room_folder)
80
81      if room_id not in p_state:
82          p_state[room_id] = {}
83      room_state = p_state[room_id]
84
85      if 'folder' not in room_state:
86          logging.debug('No previous folder for room %s' % room_folder)
87          # the folder for this room hasn't been created before
88          i = 0
89          base_folder = room_folder
90          while True:
91              full_path = os.path.join(base_path, room_folder)
92              try:
```

Console

```
get_attachments.py [debug] [/usr/local/bin/python3]
```

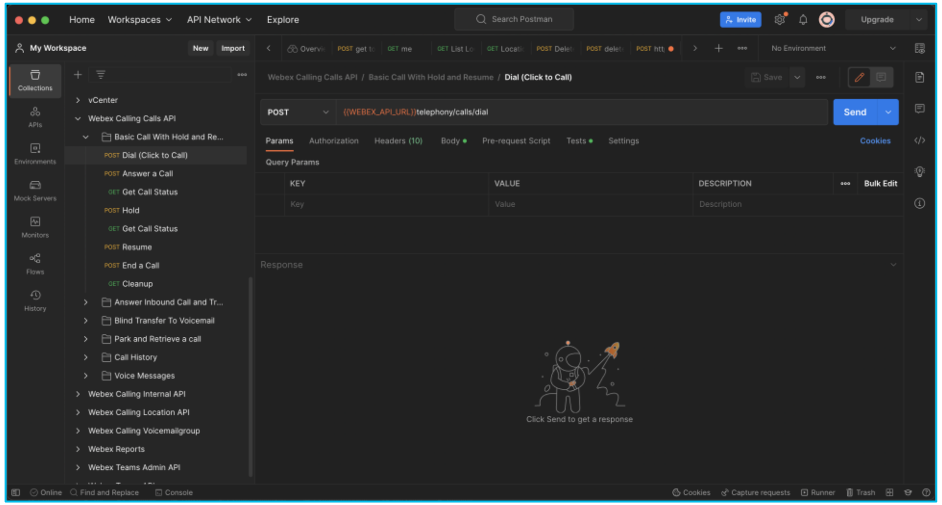
IDEs for Python

- IDLE (Standard IDE)
- PyCharm
- VS Code
- PythonAnywhere
- Cloud9 (AWS)

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

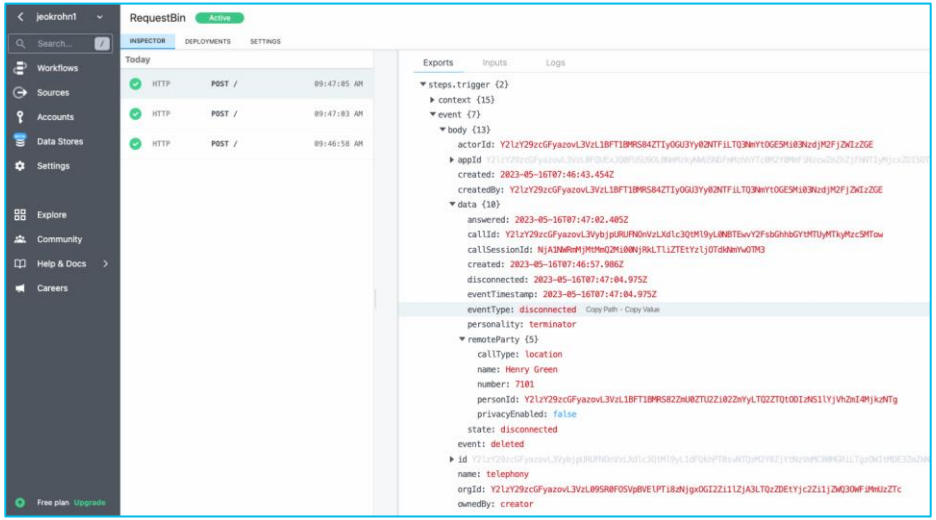
Postman: Test APIs

- Share, test, document & monitor APIs
- Easily test API calls
- Generate code (Python, curl, ..)
- Create collections
- Available for Mac, Windows, Linux, and Chrome apps
- https://www.getpostman.com/
- Postman collection for Webex Teams: https://github.com/CiscoDevNet/postman-webex



RequestBin: See Webhooks in Action

- Free service: https://pipedream.com/requestbin
- Creates unique URL
- Use case: Webex webhook pointing to Requestbin to test webhook operation
- Provides real-time view on requests hitting the URL



```
Today
HTTP    POST /    09:47:05 AM
HTTP    POST /    09:47:03 AM
HTTP    POST /    09:46:58 AM
```

```
▼ steps.trigger {2}
  ▶ context {15}
  ▼ event {7}
    ▼ body {13}
        actorId: Y2lzY29zcGFyazovL3VzL1BFT1BMRS84ZTIyOGU3Yy02NTFiLTQ3NmYtOGE5Mi03NzdjM2FjZWIzZGE
      ▶ appId [illegible]
        created: 2023-05-16T07:46:43.454Z
        createdBy: Y2lzY29zcGFyazovL3VzL1BFT1BMRS84ZTIyOGU3Yy02NTFiLTQ3NmYtOGE5Mi03NzdjM2FjZWIzZGE
      ▼ data {10}
          answered: 2023-05-16T07:47:02.405Z
          callId: Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL0NBTEwvY2FsbGhhbGYtMTUyMTkyMzc5MTow
          callSessionId: NjA1NWRmMjMtMmQ2Mi00NjRkLTliZTEtYzljOTdkNmYwOTM3
          created: 2023-05-16T07:46:57.986Z
          disconnected: 2023-05-16T07:47:04.975Z
          eventTimestamp: 2023-05-16T07:47:04.975Z
          eventType: disconnected
          personality: terminator
        ▼ remoteParty {5}
            callType: location
            name: Henry Green
            number: 7101
            personId: Y2lzY29zcGFyazovL3VzL1BFT1BMRS82ZmU0ZTU2Zi02ZmYyLTQ2ZTQtODIzNS1lYjVhZmI4MjkzNTg
            privacyEnabled: false
          state: disconnected
        event: deleted
      ▶ id [illegible]
        name: telephony
        orgId: Y2lzY29zcGFyazovL3VzL09SR0FOSVpBVElPTi8zNjgxOGI2Zi1lZjA3LTQzZDEtYjc2Zi1jZWQ3OWFiMmUzZTc
        ownedBy: creator
```

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

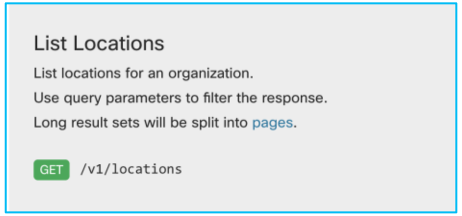
GitHub

- Git repository hosting service
- Offers
  - Revision control
  - Source code management
- THE place to share your code

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Consuming APIs

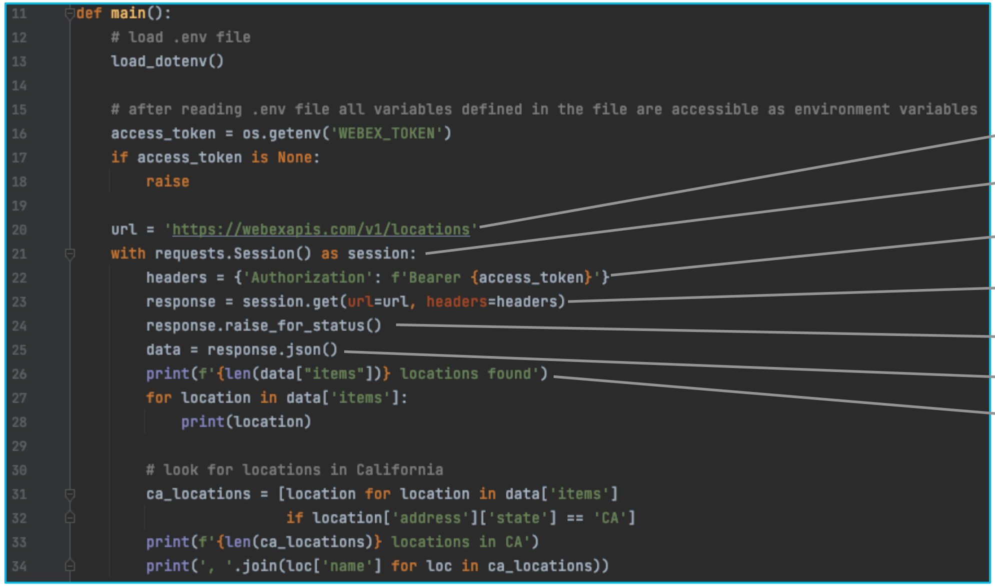
Calling a Webex API Endpoint
Listing Webex Calling Locations



List Locations

List locations for an organization.
Use query parameters to filter the response.
Long result sets will be split into pages.

GET /v1/locations



```python
def main():
    # load .env file
    load_dotenv()

    # after reading .env file all variables defined in the file are accessible as environment variables
    access_token = os.getenv('WEBEX_TOKEN')
    if access_token is None:
        raise

    url = 'https://webexapis.com/v1/locations'
    with requests.Session() as session:
        headers = {'Authorization': f'Bearer {access_token}'}
        response = session.get(url=url, headers=headers)
        response.raise_for_status()
        data = response.json()
        print(f'{len(data["items"])} locations found')
        for location in data['items']:
            print(location)

        # look for locations in California
        ca_locations = [location for location in data['items']
                        if location['address']['state'] == 'CA']
        print(f'{len(ca_locations)} locations in CA')
        print(', '.join(loc['name'] for loc in ca_locations))
```

URL of the endpoint

Session() from requests module is used

Fabricate the Authorization header

Call the endpoint

Check for errors

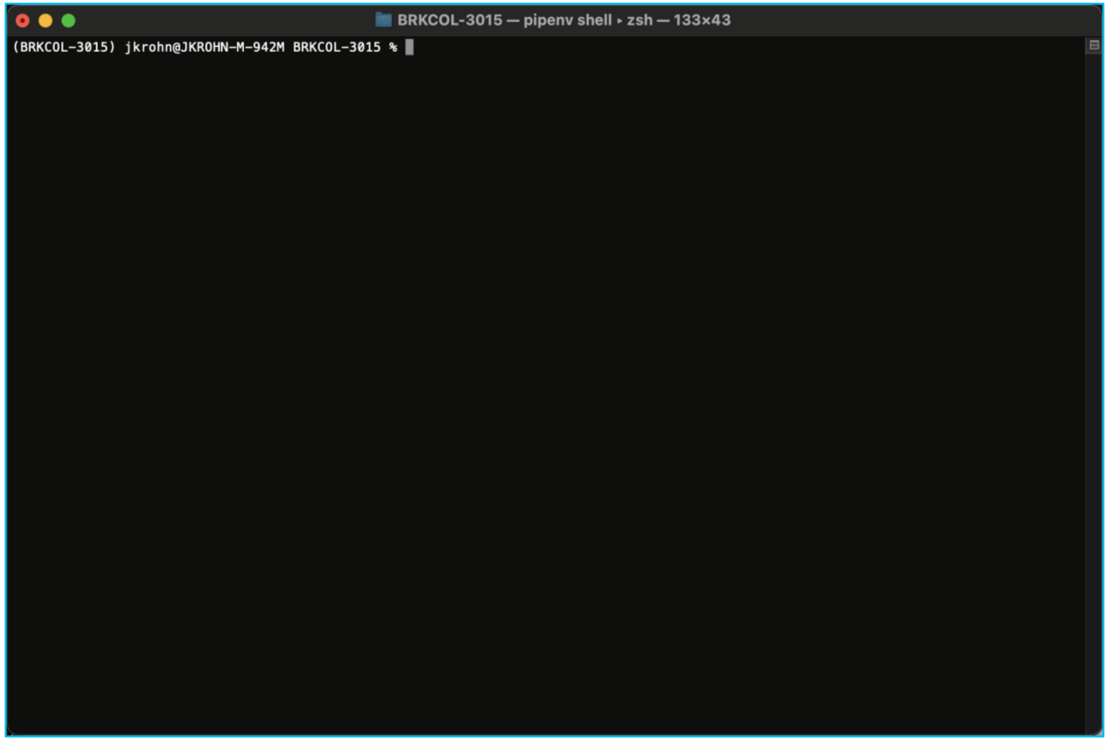
Parse the JSON response into a dict

Accessing the response values as dict keys

https://github.com/jeokrohn/BRKCOL-3015/blob/main/list_locations_direct.py

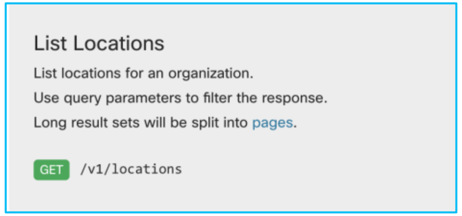
Calling a Webex API Endpoint

Listing Webex Calling Locations



```
(BRKCOL-3015) jkrohn@JKROHN-M-942M BRKCOL-3015 %
```

Calling a Webex API Endpoint
Listing Webex Calling Locations



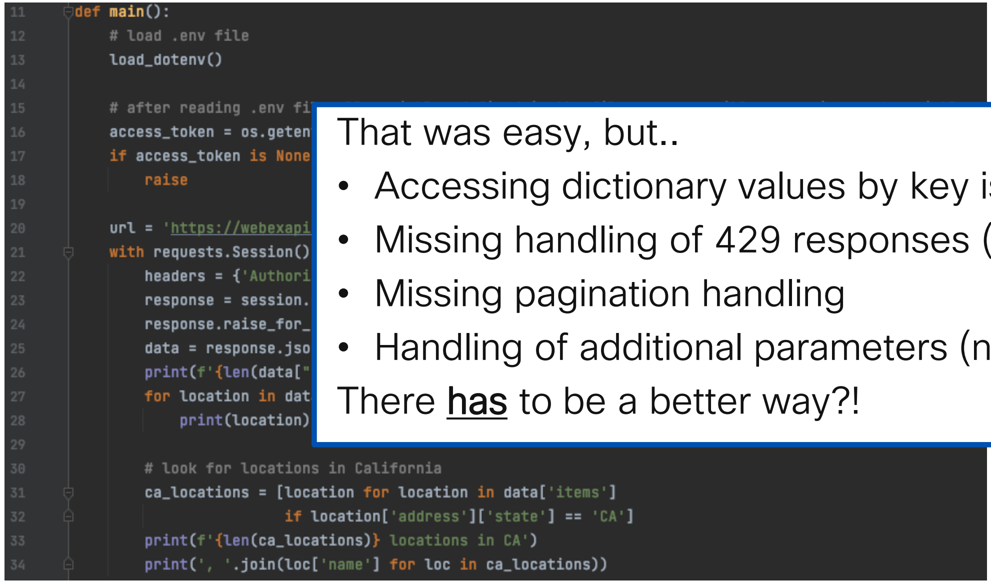
List Locations

List locations for an organization.

Use query parameters to filter the response.

Long result sets will be split into pages.

GET /v1/locations



```
def main():
    # load .env file
    load_dotenv()

    # after reading .env fi
    access_token = os.geten
    if access_token is None
        raise

    url = 'https://webexapi
    with requests.Session()
        headers = {'Authori
        response = session.
        response.raise_for_
        data = response.jso
        print(f'{len(data["
        for location in dat
            print(location)

        # look for locations in California
        ca_locations = [location for location in data['items']
                        if location['address']['state'] == 'CA']
        print(f'{len(ca_locations)} locations in CA')
        print(', '.join(loc['name'] for loc in ca_locations))
```

URL of the endpoint

ests module is used

isation header

ponse into a dict

nse values as dict keys

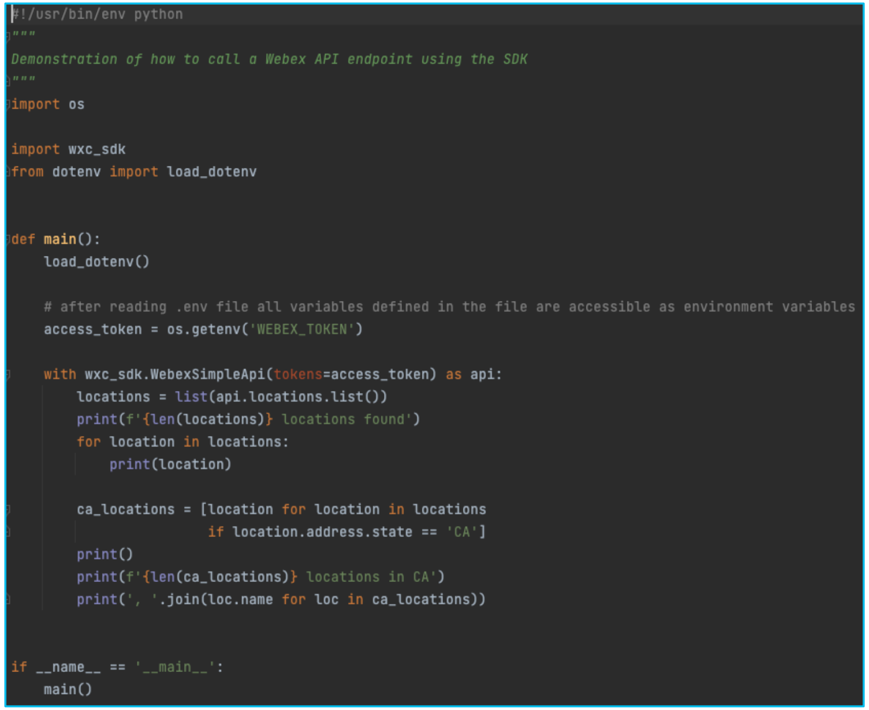
That was easy, but..

- Accessing dictionary values by key is hard and error prone
- Missing handling of 429 responses (throttling)
- Missing pagination handling
- Handling of additional parameters (name, id)

There has to be a better way?!

wxc_sdk: SDK for Webex Calling APIs

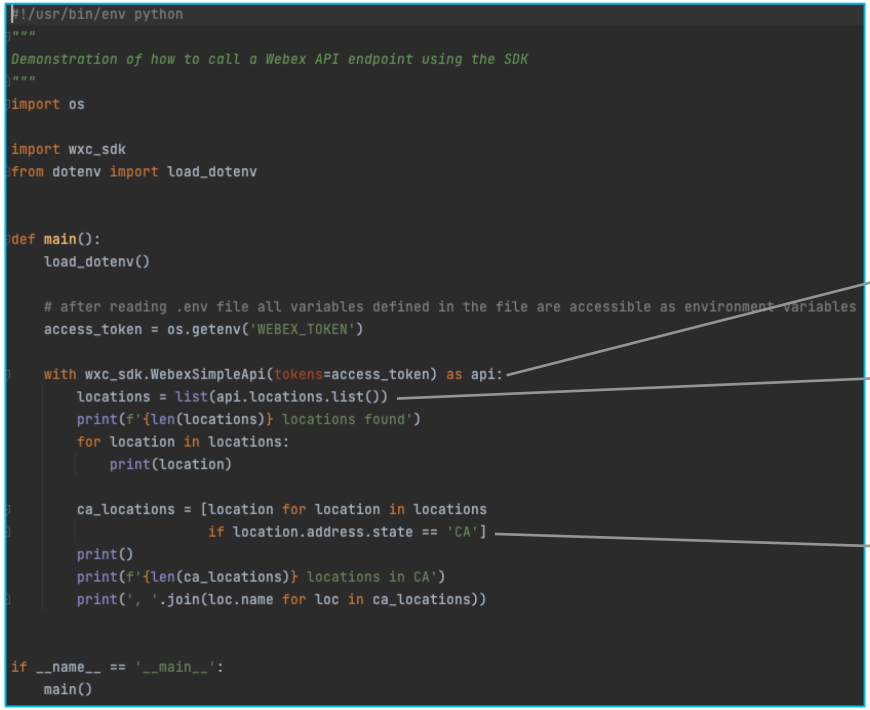
- PyPi: https://pypi.org/project/wxc-sdk/
- Homepage: https://github.com/jeokrohn/wxc_sdk
- Documentation: https://wxc-sdk.readthedocs.io/en/latest/
- Simple SDK to work with Webex APIs
  - Focus on Webex Calling specific endpoints … and more
- Takes care of all the “ugly” stuff
  - JSON (de-)serialisation, authentication, 429 retries,
  - Pagination, …
  - Logging
- Python objects for all API objects
  - Tab completion → efficient coding
- Actively maintained
  - New API endpoints will be added continously
- Foundation for your provisioning automation and other projects around Webex Calling



```
#!/usr/bin/env python
"""
Demonstration of how to call a Webex API endpoint using the SDK
"""
import os

import wxc_sdk
from dotenv import load_dotenv


def main():
    load_dotenv()

    # after reading .env file all variables defined in the file are accessible as environment variables
    access_token = os.getenv('WEBEX_TOKEN')

    with wxc_sdk.WebexSimpleApi(tokens=access_token) as api:
        locations = list(api.locations.list())
        print(f'{len(locations)} locations found')
        for location in locations:
            print(location)

        ca_locations = [location for location in locations
                        if location.address.state == 'CA']
        print()
        print(f'{len(ca_locations)} locations in CA')
        print(', '.join(loc.name for loc in ca_locations))


if __name__ == '__main__':
    main()
```

Calling a Webex API Endpoint

Listing Webex Calling Locations using the SDK



```python
#!/usr/bin/env python
"""
Demonstration of how to call a Webex API endpoint using the SDK
"""
import os

import wxc_sdk
from dotenv import load_dotenv


def main():
    load_dotenv()

    # after reading .env file all variables defined in the file are accessible as environment variables
    access_token = os.getenv('WEBEX_TOKEN')

    with wxc_sdk.WebexSimpleApi(tokens=access_token) as api:
        locations = list(api.locations.list())
        print(f'{len(locations)} locations found')
        for location in locations:
            print(location)

        ca_locations = [location for location in locations
                        if location.address.state == 'CA']
        print()
        print(f'{len(ca_locations)} locations in CA')
        print(', '.join(loc.name for loc in ca_locations))


if __name__ == '__main__':
    main()
```

The API object

Get list of locations

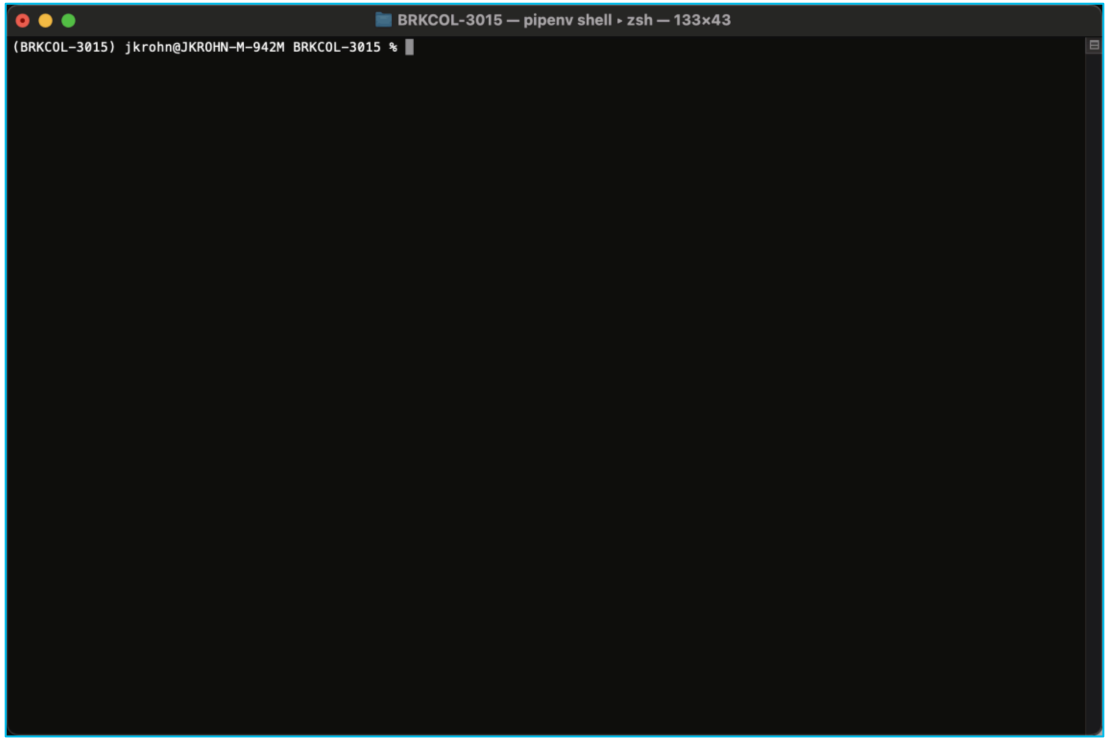
Access data using attributes of Python classes

https://github.com/jeokrohn/BRKCOL-3015/blob/main/list_locations_sdk.py

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Calling a Webex API Endpoint

Listing Webex Calling Locations using the SDK

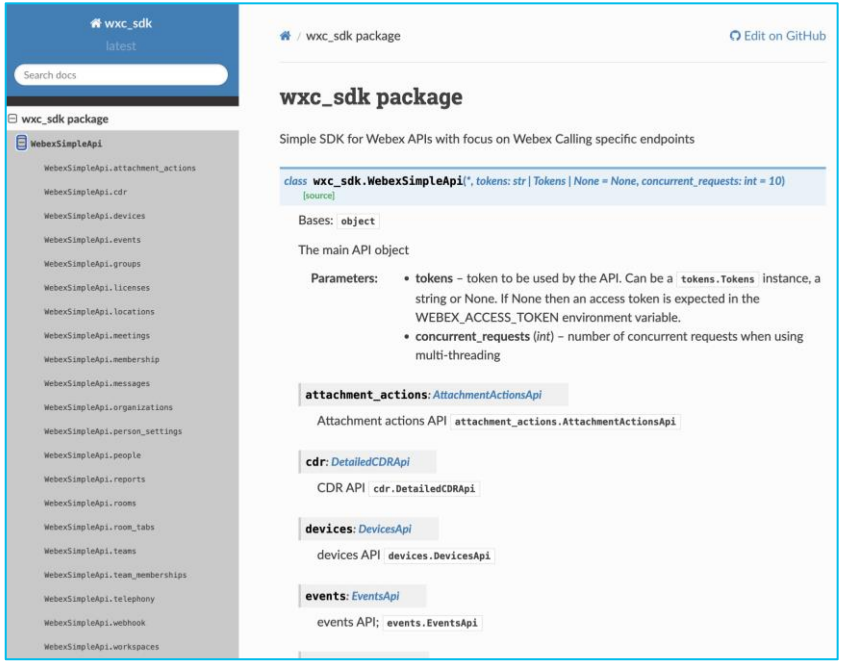


```
(BRKCOL-3015) jkrohn@JKROHN-M-942M BRKCOL-3015 %
```

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

wxc_sdk: Comprehensive Coverage

- SDK covers all Webex Calling specific API endpoints
- Additionally:
  - Licenses, memberships, messages, people, teams, team memberships, webhooks, ...
- Easy token management



wxc_sdk
latest
Search docs

wxc_sdk package
WebexSimpleApi
WebexSimpleApi.attachment_actions
WebexSimpleApi.cdr
WebexSimpleApi.devices
WebexSimpleApi.events
WebexSimpleApi.groups
WebexSimpleApi.licenses
WebexSimpleApi.locations
WebexSimpleApi.meetings
WebexSimpleApi.membership
WebexSimpleApi.messages
WebexSimpleApi.organizations
WebexSimpleApi.person_settings
WebexSimpleApi.people
WebexSimpleApi.reports
WebexSimpleApi.rooms
WebexSimpleApi.room_tabs
WebexSimpleApi.teams
WebexSimpleApi.team_memberships
WebexSimpleApi.telephony
WebexSimpleApi.webhook
WebexSimpleApi.workspaces

/ wxc_sdk package — Edit on GitHub

wxc_sdk package

Simple SDK for Webex APIs with focus on Webex Calling specific endpoints

class `wxc_sdk.WebexSimpleApi`(**, tokens: str | Tokens | None = None, concurrent_requests: int = 10*) [source]

Bases: `object`

The main API object

Parameters:
- **tokens** – token to be used by the API. Can be a `tokens.Tokens` instance, a string or None. If None then an access token is expected in the WEBEX_ACCESS_TOKEN environment variable.
- **concurrent_requests** (*int*) – number of concurrent requests when using multi-threading

`attachment_actions`: *AttachmentActionsApi*

Attachment actions API `attachment_actions.AttachmentActionsApi`

`cdr`: *DetailedCDRApi*

CDR API `cdr.DetailedCDRApi`

`devices`: *DevicesApi*

devices API `devices.DevicesApi`

`events`: *EventsApi*

events API; `events.EventsApi`

https://wxc-sdk.readthedocs.io/en/latest/

Tokens

Tokens
Why and How?

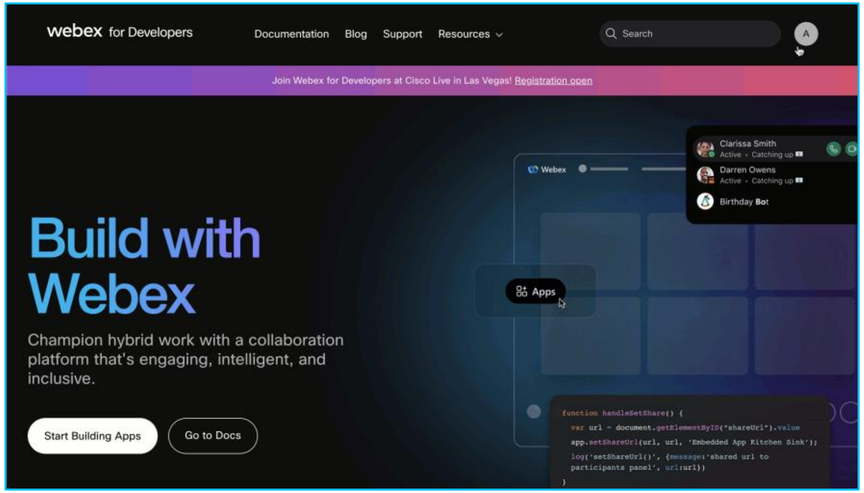
- Access tokens are required to authorize API accces
- .. Can be obtained in different ways:
  - Personal access token (developer token)
  - Integration (OAuth2 authorization flow)
  - Service App
- NEVER(!!!) store tokens in your source files
- NEVER(!!!) push tokens to GitHub repositories
- NEVER(!!!) share tokens in any shape or form
- Best practice:
  - In your code read access token from environment variable
  - Use `dotenv.loadenv()` to load `.env` file with environment variables
  - Exclude `.env` from version control (Git) by adding exclusion in `.gitignore`
  - Integration tokens can be cached in local files .. but make sure to restrict access and not push to GitHub

```
GET https://webexapis.com/v1/people
User-Agent: python-requests/2.30.0
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
Authorization: Bearer MGY4MzNjNzgt***
content-type: application/json;charset=utf-8
```

https://developer.webex.com/docs/getting-started

Personal Access Token

- From developer.webex.com
- Limited lifetime (12 h)
- Should NEVER be used in production
- Testing only



Join Webex for Developers at Cisco Live in Las Vegas! Registration open

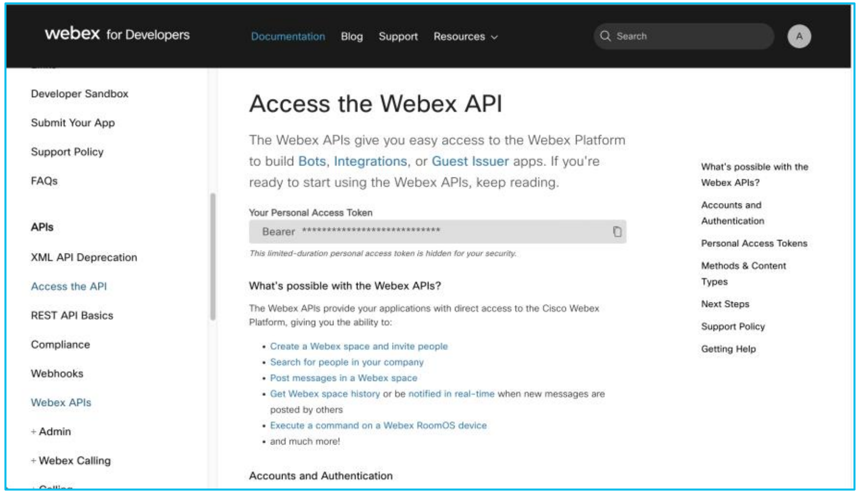
Build with Webex

Champion hybrid work with a collaboration platform that's engaging, intelligent, and inclusive.

```
function handleSetShare() {
  var url = document.getElementByID("shareUrl").value
  app.setShareUrl(url, url, 'Embedded App Kitchen Sink');
  log('setShareUrl()', {message:'shared url to
  participants panel', url:url})
}
```

Personal Access Token

- From developer.webex.com
- Limited lifetime (12 h)
- Should NEVER be used in production
- Testing only



Access the Webex API

The Webex APIs give you easy access to the Webex Platform to build Bots, Integrations, or Guest Issuer apps. If you're ready to start using the Webex APIs, keep reading.

Your Personal Access Token

Bearer ****************************

This limited-duration personal access token is hidden for your security.

What's possible with the Webex APIs?

The Webex APIs provide your applications with direct access to the Cisco Webex Platform, giving you the ability to:

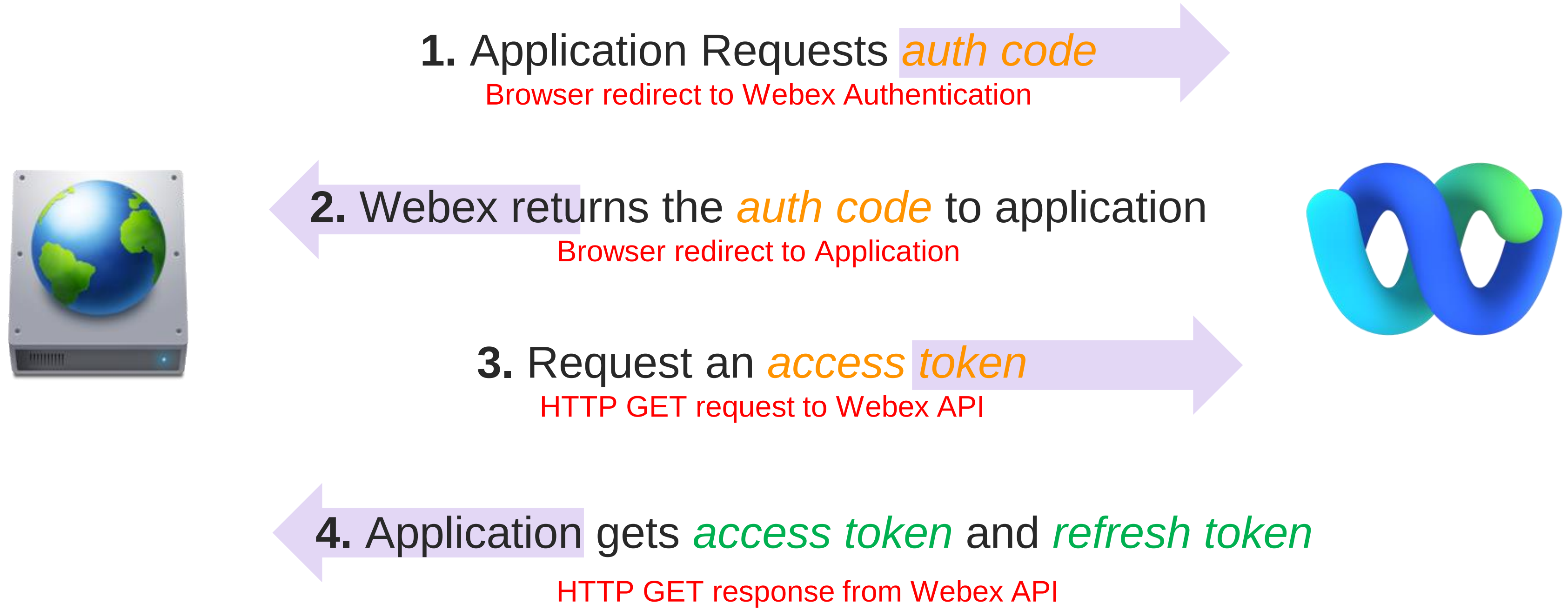
- Create a Webex space and invite people
- Search for people in your company
- Post messages in a Webex space
- Get Webex space history or be notified in real-time when new messages are posted by others
- Execute a command on a Webex RoomOS device
- and much more!

Accounts and Authentication

Integration – The Better Way to Obtain Tokens

- Act on behalf of a Webex user
  - Access equivalent to a real Webex User (limited by authorized scopes)
- Invoke Webex APIs on behalf of user
- Requires authorization of integration by user
  - OAuth Grant Flow to authenticate user and ask for authorization
  - User approves authorization levels (scopes) requested by the integration
- Each Integration has a client ID, client secret and redirect URI
- Documentation: https://developer.webex.com/docs/integrations

OAuth Authorization Code Flow



1. Application Requests *auth code*

Browser redirect to Webex Authentication

2. Webex returns the *auth code* to application

Browser redirect to Application

3. Request an *access token*

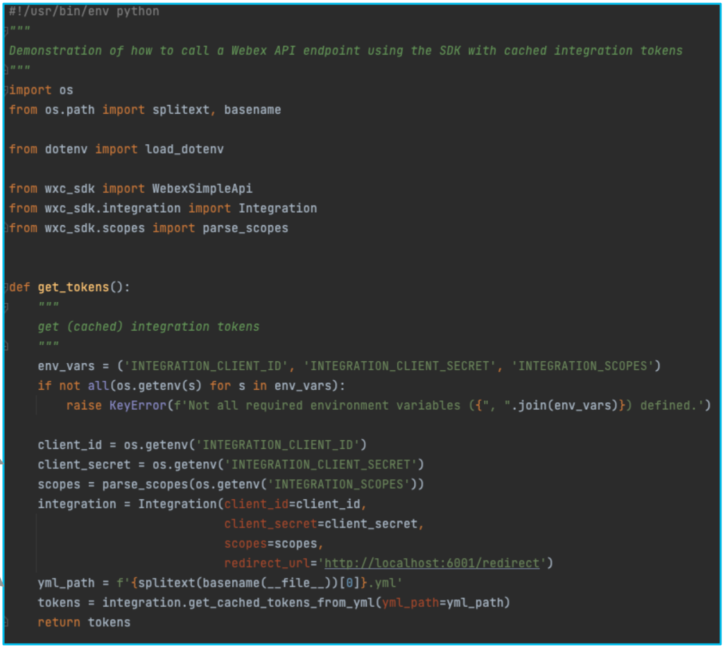
HTTP GET request to Webex API

4. Application gets *access token* and *refresh token*

HTTP GET response from Webex API

Integration Tokens in Scripts Using wxc_sdk

- `wxc_sdk` offers an easy way to work with cached tokens in scripts
- Cache tokens in YML file
- Get tokens from OAuth flow redirecting to http://localhost:6001/redirect
- Spin up temporary (primitive) server to handle final step of OAuth flow



```python
#!/usr/bin/env python
"""
Demonstration of how to call a Webex API endpoint using the SDK with cached integration tokens
"""
import os
from os.path import splitext, basename

from dotenv import load_dotenv

from wxc_sdk import WebexSimpleApi
from wxc_sdk.integration import Integration
from wxc_sdk.scopes import parse_scopes


def get_tokens():
    """
    get (cached) integration tokens
    """
    env_vars = ('INTEGRATION_CLIENT_ID', 'INTEGRATION_CLIENT_SECRET', 'INTEGRATION_SCOPES')
    if not all(os.getenv(s) for s in env_vars):
        raise KeyError(f'Not all required environment variables ({", ".join(env_vars)}) defined.')

    client_id = os.getenv('INTEGRATION_CLIENT_ID')
    client_secret = os.getenv('INTEGRATION_CLIENT_SECRET')
    scopes = parse_scopes(os.getenv('INTEGRATION_SCOPES'))
    integration = Integration(client_id=client_id,
                              client_secret=client_secret,
                              scopes=scopes,
                              redirect_url='http://localhost:6001/redirect')
    yml_path = f'{splitext(basename(__file__))[0]}.yml'
    tokens = integration.get_cached_tokens_from_yml(yml_path=yml_path)
    return tokens
```

Prepare integration based on values read from environment

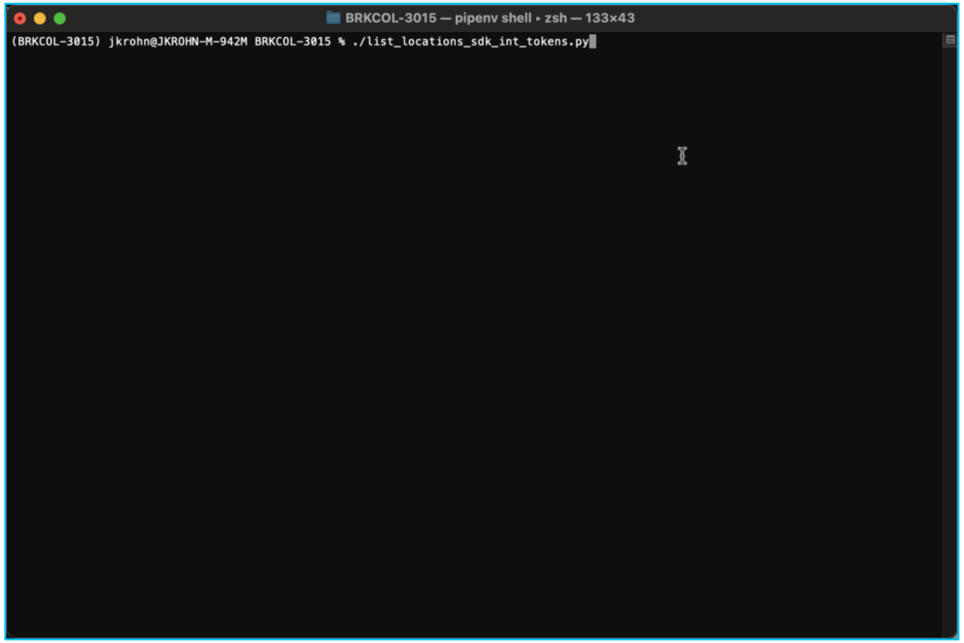
Read, create, refresh, cache tokens

```
access_token: ZGVlY2***
expires_at: '2023-05-30T14:35:37.246110+00:00'
refresh_token: NTlk***
refresh_token_expires_at: '2023-08-14T14:08:57.246110+00:00'
token_type: Bearer
```

https://github.com/jeokrohn/BRKCOL-3015/blob/main/list_locations_sdk_int_tokens.py

Integration Tokens in Scripts Using wxc_sdk

- Without cached tokens OAuth flow gets initiated
- Auth code exchanged for tokens
- Tokens are cached
- Next execution uses cached tokens



BRKCOL-3015 — pipenv shell ▸ zsh — 133×43

```
(BRKCOL-3015) jkrohn@JKROHN-M-942M BRKCOL-3015 % ./list_locations_sdk_int_tokens.py
```

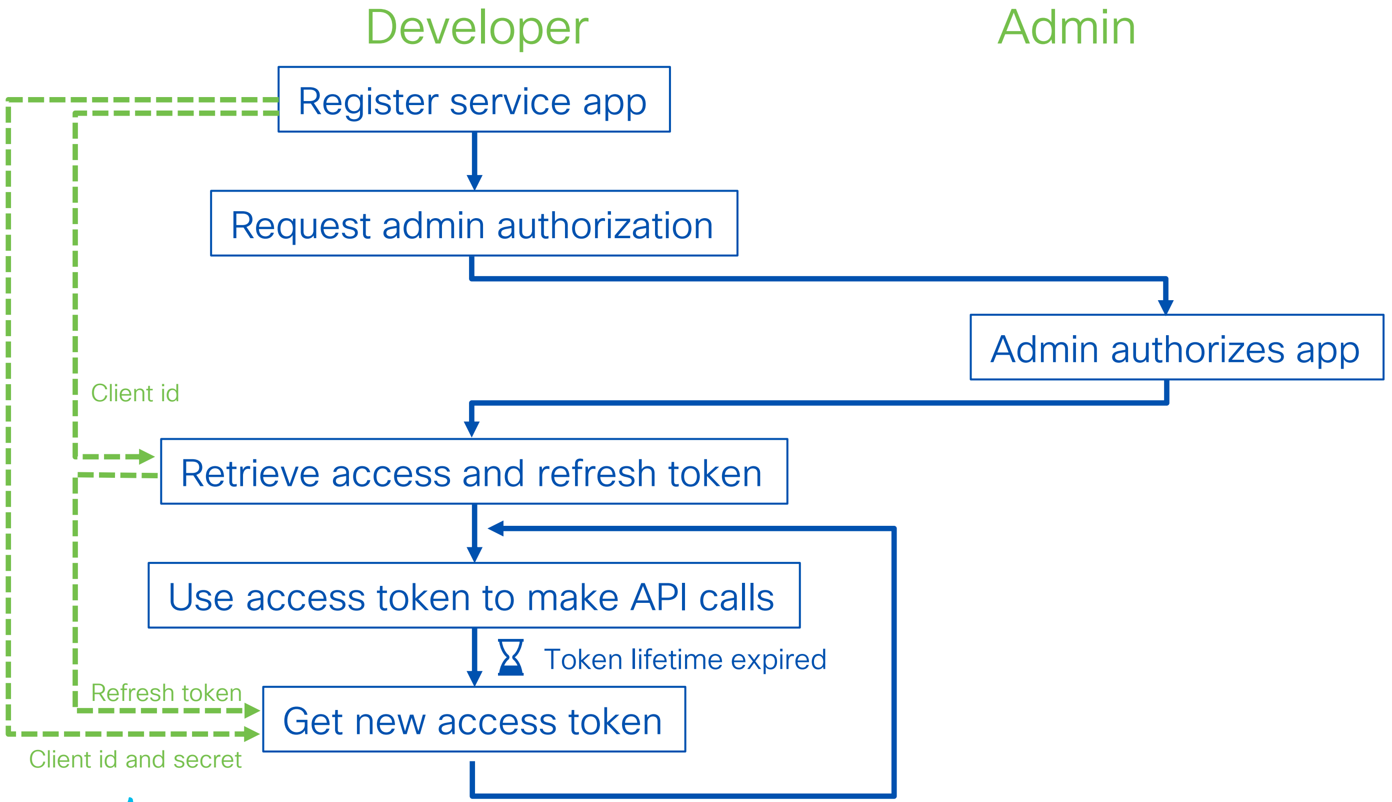
https://github.com/jeokrohn/BRKCOL-3015/blob/main/list_locations_sdk_int_tokens.py

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

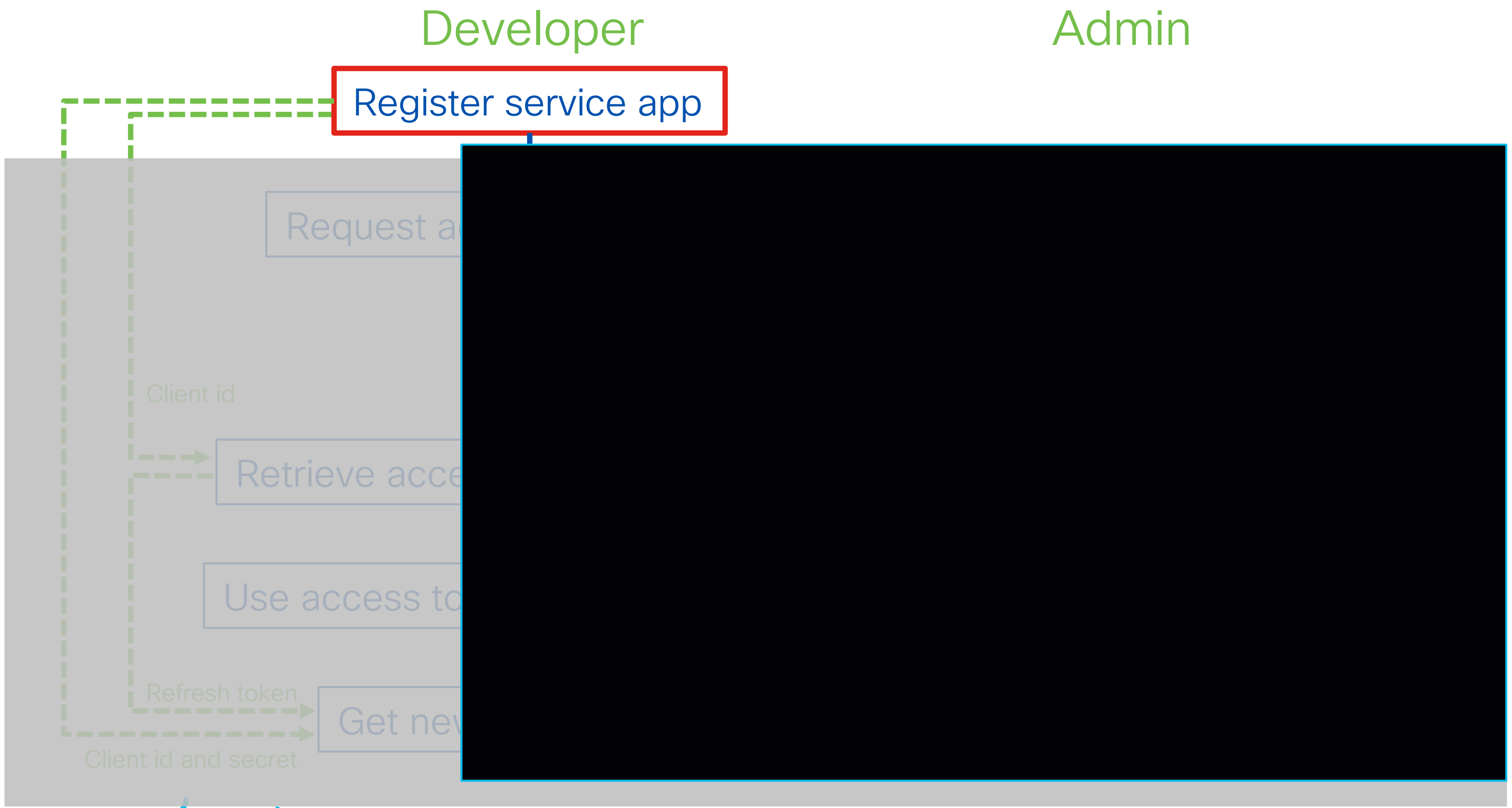
Service Apps

- Machine account
- Request admin permission independent of user account
- Use cases
  - Provisioning
  - Reporting
  - Scheduling systems
  - …
- Similar to integrations .. but no user specific authorization flow

Using Service App Tokens

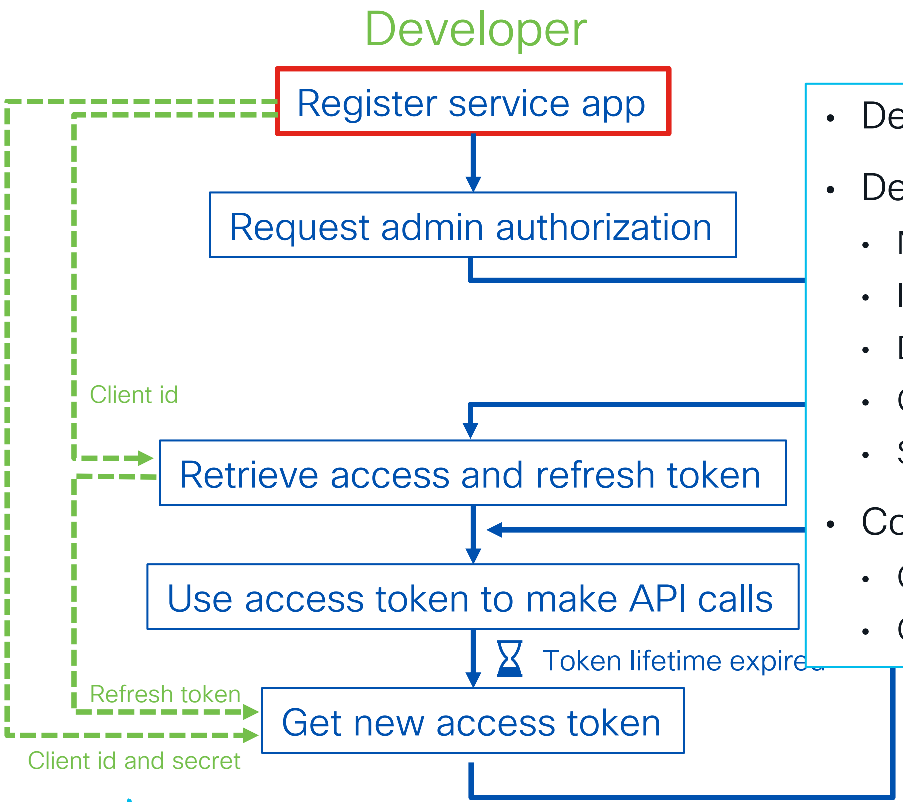


Using Service App Tokens



© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Using Service App Tokens



Admin

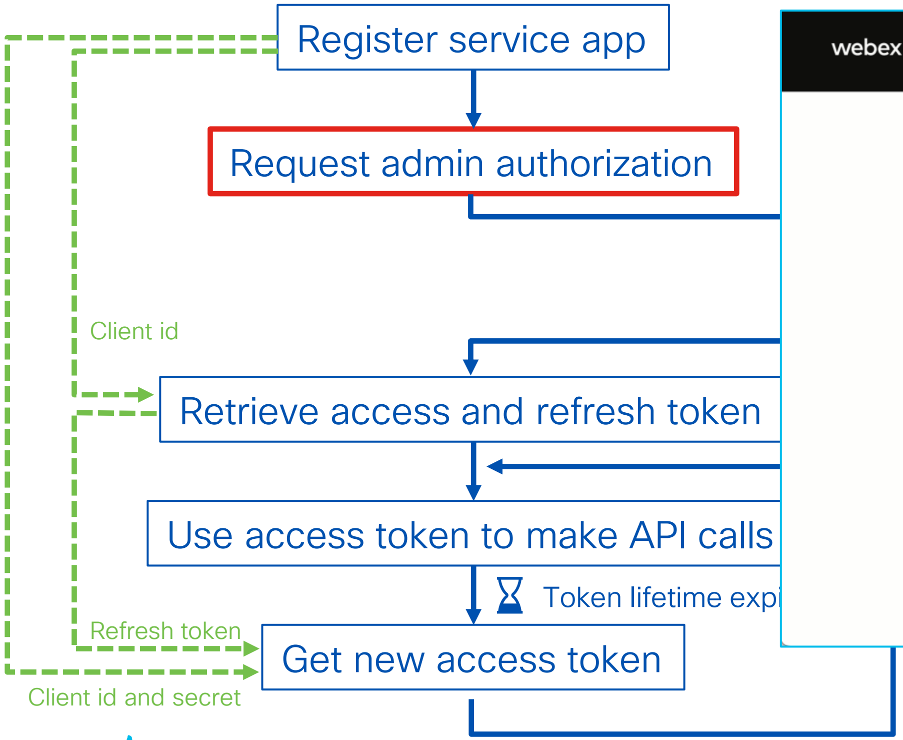
- Developer.cisco.com, “My Webex apps”
- Define:
  - Name
  - Icon
  - Description
  - Contact email address
  - Scopes
- Collect
  - Client id
  - Client secret

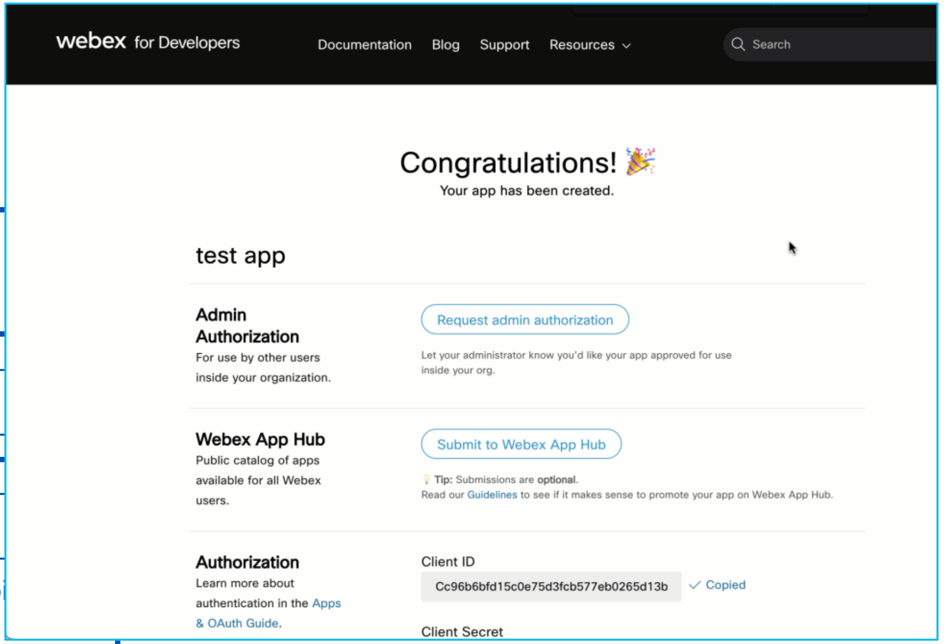
© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Using Service App Tokens

Developer

Admin





Congratulations! 🎉

Your app has been created.

test app

Admin Authorization
For use by other users inside your organization.

Request admin authorization

Let your administrator know you'd like your app approved for use inside your org.

Webex App Hub
Public catalog of apps available for all Webex users.

Submit to Webex App Hub

Tip: Submissions are optional.
Read our Guidelines to see if it makes sense to promote your app on Webex App Hub.

Authorization
Learn more about authentication in the Apps & OAuth Guide.

Client ID
Cc96b6bfd15c0e75d3fcb577eb0265d13b ✓ Copied

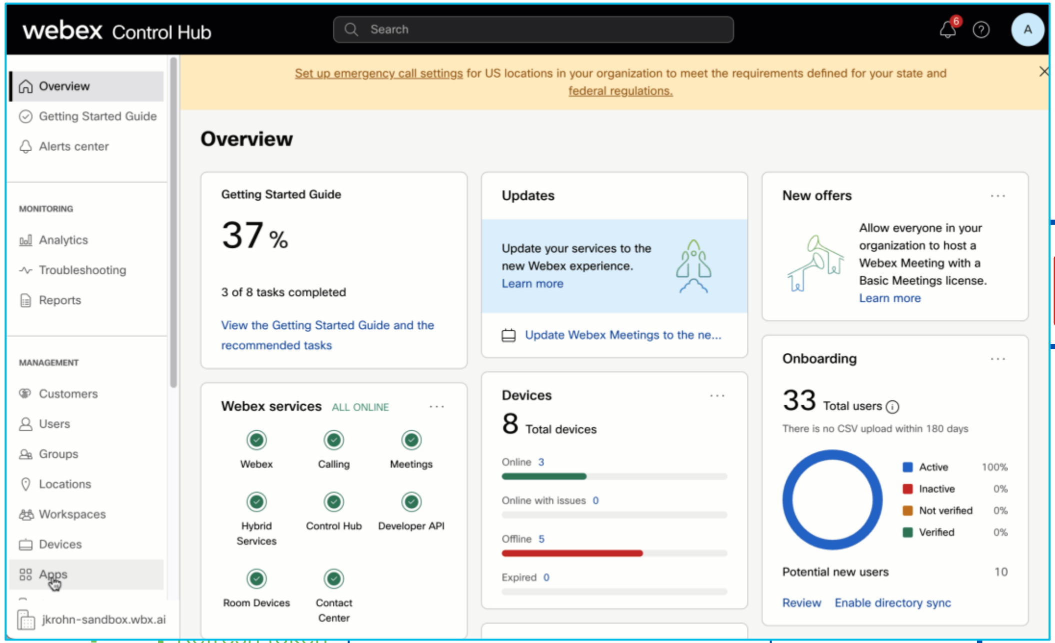
Client Secret

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Using Service App Tokens

Developer

Admin



webex Control Hub

Search

Set up emergency call settings for US locations in your organization to meet the requirements defined for your state and federal regulations.

- Overview
- Getting Started Guide
- Alerts center

MONITORING

- Analytics
- Troubleshooting
- Reports

MANAGEMENT

- Customers
- Users
- Groups
- Locations
- Workspaces
- Devices
- Apps

jkrohn-sandbox.wbx.ai

Overview

Getting Started Guide

37 %

3 of 8 tasks completed

View the Getting Started Guide and the recommended tasks

Updates

Update your services to the new Webex experience.
Learn more

Update Webex Meetings to the ne...

New offers

Allow everyone in your organization to host a Webex Meeting with a Basic Meetings license.
Learn more

Webex services ALL ONLINE

Webex, Calling, Meetings, Hybrid Services, Control Hub, Developer API, Room Devices, Contact Center

Devices

8 Total devices

Online 3

Online with issues 0

Offline 5

Expired 0

Onboarding

33 Total users

There is no CSV upload within 180 days

Active	100%
Inactive	0%
Not verified	0%
Verified	0%

Potential new users 10

Review Enable directory sync

Refresh token

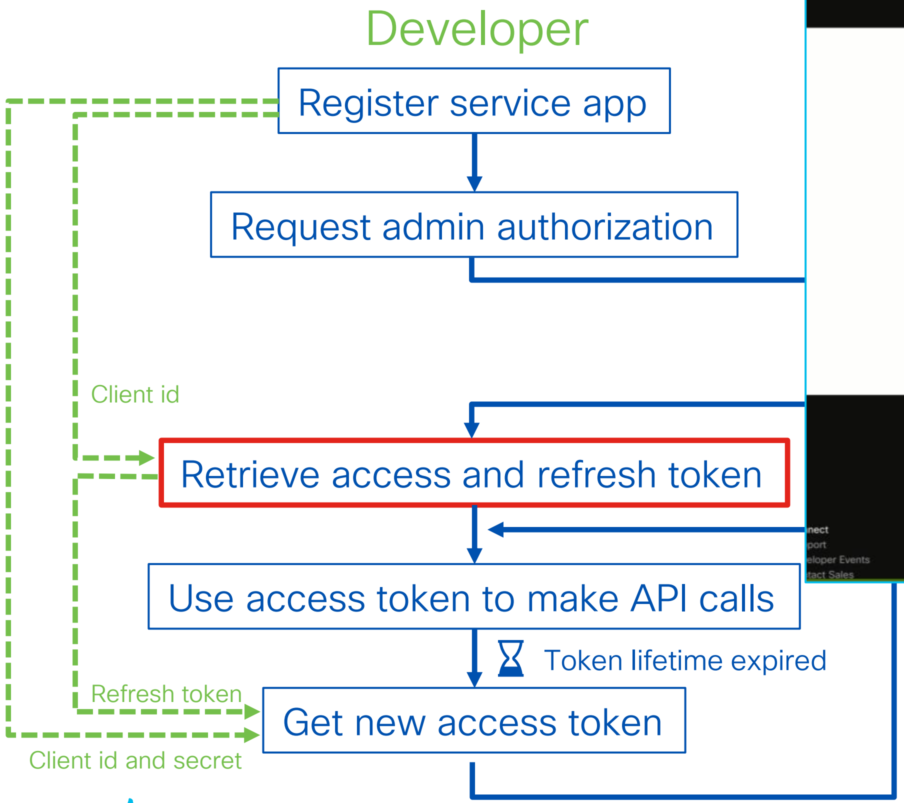
Client id and secret

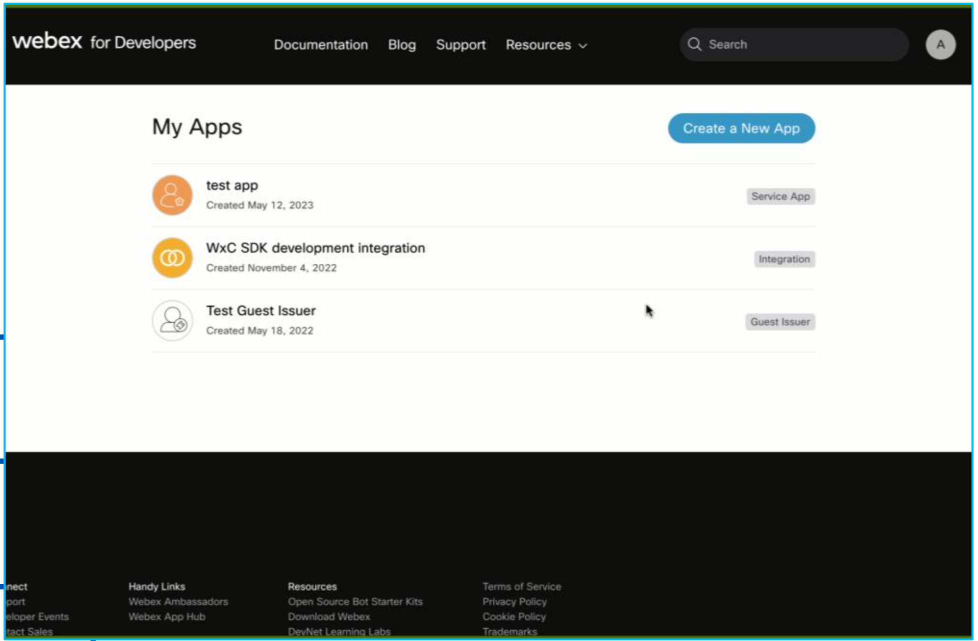
Get new access token

Admin authorizes app

- “Apps” section in Control Hub
- Review scopes
- Authorizing user is documented

Using Service App Tokens



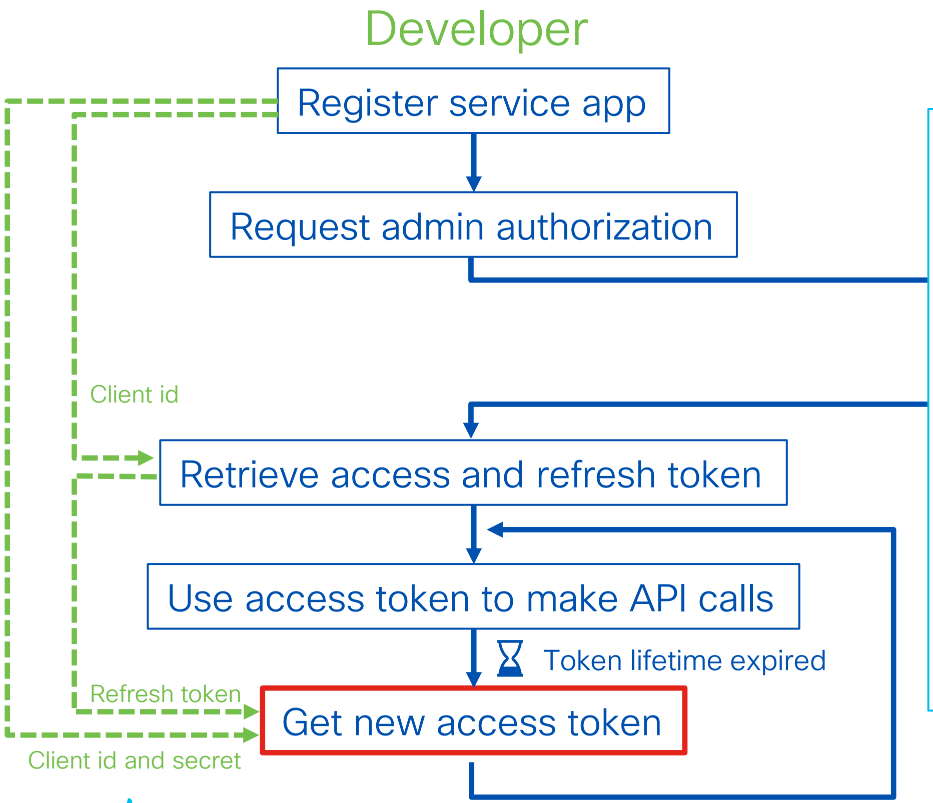


My Apps

test app Created May 12, 2023	Service App
WxC SDK development integration Created November 4, 2022	Integration
Test Guest Issuer Created May 18, 2022	Guest Issuer

- Developer requests tokens for authorized org
- Client secret needed to authorize the request

Using Service App Tokens



Admin

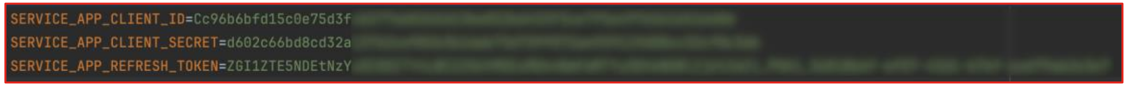
- When getting close to token lifetime
- New access token can be obtained using the refresh token

```
POST https://webexapis.com/v1/access_token
Content-Type: application/x-www-form-urlencoded
  --- body ---
  grant_type: refresh_token
  client_id: ***
  client_secret: ***
  refresh_token: ***
Response
Content-Type: application/json
--- response body ---
  {
    "access_token": "***",
    "expires_in": 1209599,
    "refresh_token": ″***",
    "refresh_token_expires_in": 7775954,
    "token_type": "Bearer",
    "scope": "spark:kms …"
  }
```

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

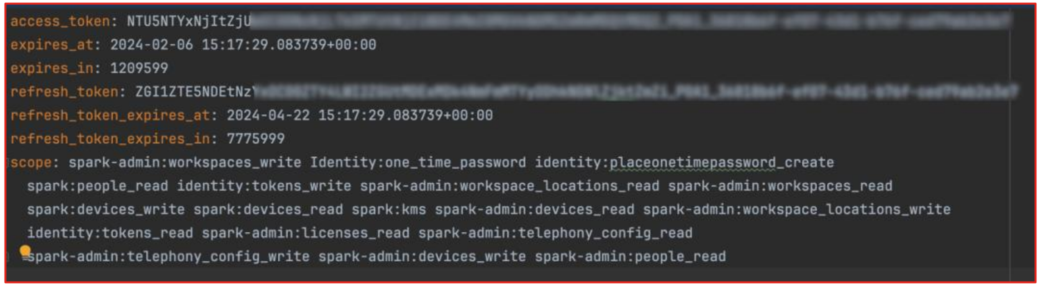
Using Service App with wxc_sdk

Read secrets from environment



```
SERVICE_APP_CLIENT_ID=Cc96b6bfd15c0e75d3f[redacted]
SERVICE_APP_CLIENT_SECRET=d602c66bd8cd32a[redacted]
SERVICE_APP_REFRESH_TOKEN=ZGI1ZTE5NDEtNzY[redacted]
```

Create access token and persist tokens in YML file



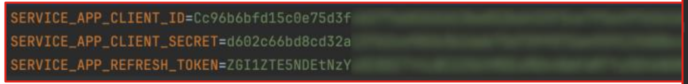
```
access_token: NTU5NTYxNjItZjU[redacted]
expires_at: 2024-02-06 15:17:29.083739+00:00
expires_in: 1209599
refresh_token: ZGI1ZTE5NDEtNzY[redacted]
refresh_token_expires_at: 2024-04-22 15:17:29.083739+00:00
refresh_token_expires_in: 7775999
scope: spark-admin:workspaces_write Identity:one_time_password identity:placeonetimepassword_create
  spark:people_read identity:tokens_write spark-admin:workspace_locations_read spark-admin:workspaces_read
  spark:devices_write spark:devices_read spark:kms spark-admin:devices_read spark-admin:workspace_locations_write
  identity:tokens_read spark-admin:licenses_read spark-admin:telephony_config_read
  spark-admin:telephony_config_write spark-admin:devices_write spark-admin:people_read
```

Use tokens to call endpoints

https://github.com/jeokrohn/wxc_sdk/blob/master/examples/service_app.py

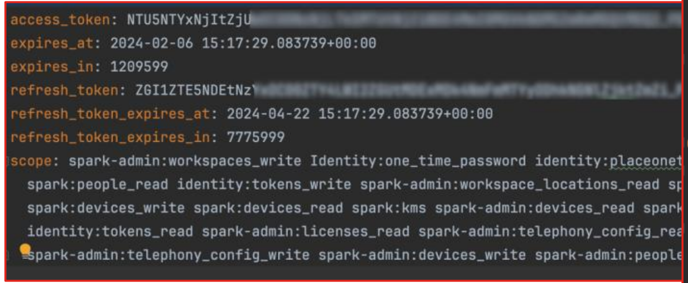
Using Service App with wxc_sdk

Read secrets from environment



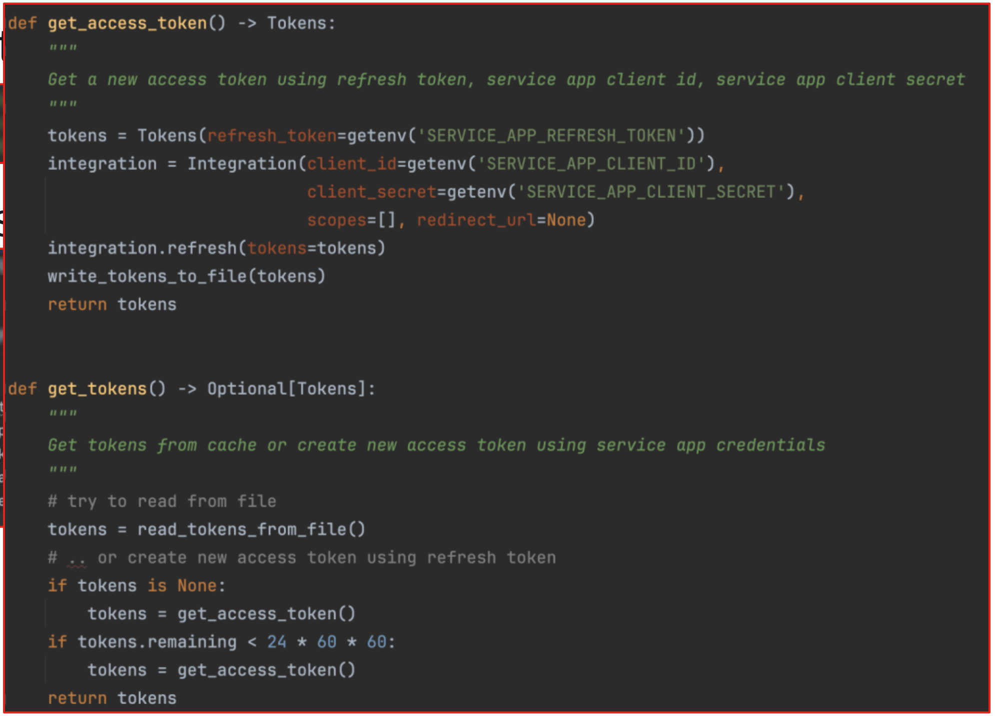
```
SERVICE_APP_CLIENT_ID=Cc96b6bfd15c0e75d3f[redacted]
SERVICE_APP_CLIENT_SECRET=d602c66bd8cd32a[redacted]
SERVICE_APP_REFRESH_TOKEN=ZGI1ZTE5NDEtNzY[redacted]
```

Create access token and persis



```
access_token: NTU5NTYxNjItZjU[redacted]
expires_at: 2024-02-06 15:17:29.083739+00:00
expires_in: 1209599
refresh_token: ZGI1ZTE5NDEtNzY[redacted]
refresh_token_expires_at: 2024-04-22 15:17:29.083739+00:00
refresh_token_expires_in: 7775999
scope: spark-admin:workspaces_write Identity:one_time_password identity:placeonet
  spark:people_read identity:tokens_write spark-admin:workspace_locations_read sp
  spark:devices_write spark:devices_read spark:kms spark-admin:devices_read spark
  identity:tokens_read spark-admin:licenses_read spark-admin:telephony_config_rea
  spark-admin:telephony_config_write spark-admin:devices_write spark-admin:people
```

Use tokens to call endpoints

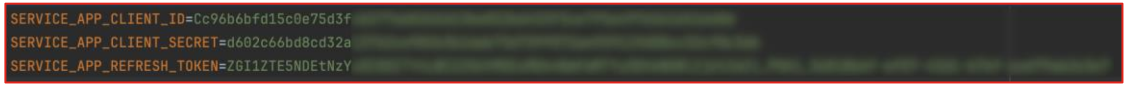


```python
def get_access_token() -> Tokens:
    """
    Get a new access token using refresh token, service app client id, service app client secret
    """
    tokens = Tokens(refresh_token=getenv('SERVICE_APP_REFRESH_TOKEN'))
    integration = Integration(client_id=getenv('SERVICE_APP_CLIENT_ID'),
                              client_secret=getenv('SERVICE_APP_CLIENT_SECRET'),
                              scopes=[], redirect_url=None)
    integration.refresh(tokens=tokens)
    write_tokens_to_file(tokens)
    return tokens


def get_tokens() -> Optional[Tokens]:
    """
    Get tokens from cache or create new access token using service app credentials
    """
    # try to read from file
    tokens = read_tokens_from_file()
    # .. or create new access token using refresh token
    if tokens is None:
        tokens = get_access_token()
    if tokens.remaining < 24 * 60 * 60:
        tokens = get_access_token()
    return tokens
```

https://github.com/jeokrohn/wxc_sdk/blob/master/examples/service_app.py

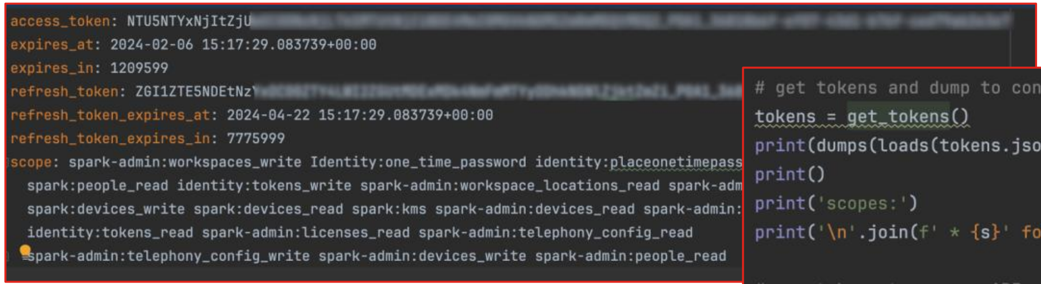
Using Service App with wxc_sdk

Read secrets from environment

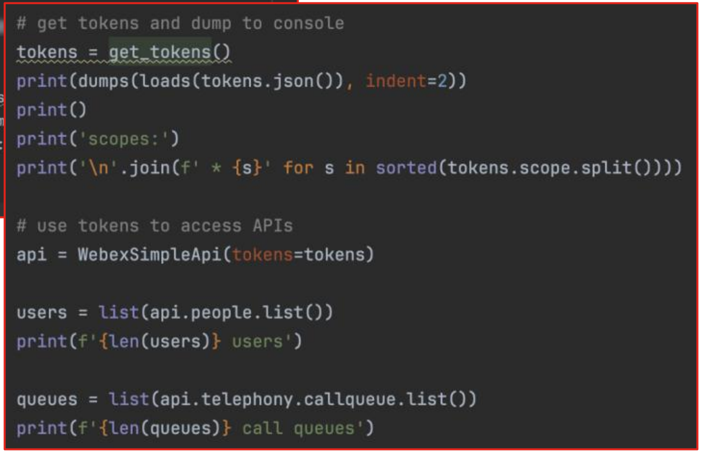


```
SERVICE_APP_CLIENT_ID=Cc96b6bfd15c0e75d3f[redacted]
SERVICE_APP_CLIENT_SECRET=d602c66bd8cd32a[redacted]
SERVICE_APP_REFRESH_TOKEN=ZGI1ZTE5NDEtNzY[redacted]
```

Create access token and persist tokens in YML file



```
access_token: NTU5NTYxNjItZjU[redacted]
expires_at: 2024-02-06 15:17:29.083739+00:00
expires_in: 1209599
refresh_token: ZGI1ZTE5NDEtNzY[redacted]
refresh_token_expires_at: 2024-04-22 15:17:29.083739+00:00
refresh_token_expires_in: 7775999
scope: spark-admin:workspaces_write Identity:one_time_password identity:placeonetimepass
  spark:people_read identity:tokens_write spark-admin:workspace_locations_read spark-adm
  spark:devices_write spark:devices_read spark:kms spark-admin:devices_read spark-admin:
  identity:tokens_read spark-admin:licenses_read spark-admin:telephony_config_read
  spark-admin:telephony_config_write spark-admin:devices_write spark-admin:people_read
```



```python
# get tokens and dump to console
tokens = get_tokens()
print(dumps(loads(tokens.json()), indent=2))
print()
print('scopes:')
print('\n'.join(f' * {s}' for s in sorted(tokens.scope.split())))

# use tokens to access APIs
api = WebexSimpleApi(tokens=tokens)

users = list(api.people.list())
print(f'{len(users)} users')

queues = list(api.telephony.callqueue.list())
print(f'{len(queues)} call queues')
```

Use tokens to call endpoints

https://github.com/jeokrohn/wxc_sdk/blob/master/examples/service_app.py

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Token Overview

	Developer Token	Integration Token	Service App Token
How to get	From developer.webex.com	Requires OAuth auth code authorization flow, web server required	Service app access to org granted by org admin. Service app owner creates token on developer.webex.com
Granular Access Control	No, always has all scopes	Set of scopes assigned to integration	Set of scopes assigned to service app
Actor	Owner of developer token	User granting authorization	Service app
Lifetime	12 hrs	Access token: 14 d Refresh token: 90 d (extended when getting new access token)	
Use cases	Development, Tests	(Web) applications acting on behalf of user	(Web) services explicitly authorized by org admin

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Use Cases / Examples

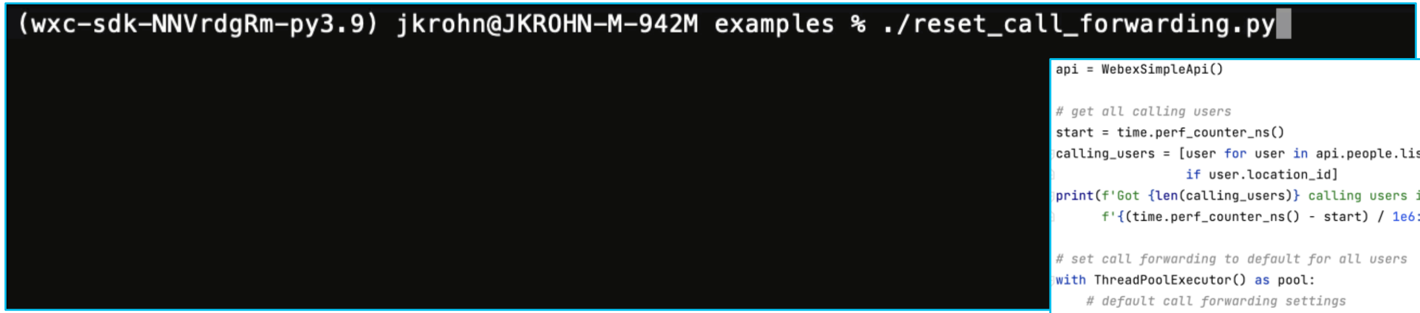
Use Cases

- Scripts
  - Bulk Provisioning: user, calling features, ...
  - Validation: check/verify settings; scripting saves you from navigating through individual menus in ControlHub
  - Automation reduces the risk of errors in repetitive tasks
  - CLI tools
- Integration with existing enterprise mangement systems/tools
- Backend for web services (portal applications)

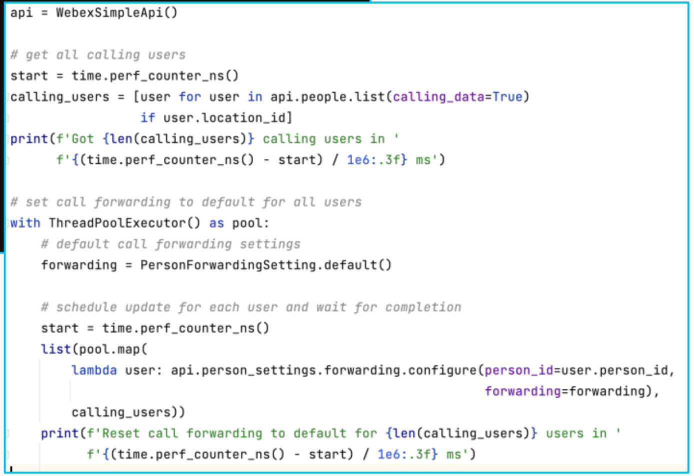
© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

SDK Examples

- Examples available at:
  https://wxc-sdk.readthedocs.io/en/latest/examples.html
  https://github.com/jeokrohn/wxc_sdk/tree/master/examples



```
(wxc-sdk-NNVrdgRm-py3.9) jkrohn@JKROHN-M-942M examples % ./reset_call_forwarding.py
```



```python
api = WebexSimpleApi()

# get all calling users
start = time.perf_counter_ns()
calling_users = [user for user in api.people.list(calling_data=True)
                 if user.location_id]
print(f'Got {len(calling_users)} calling users in '
      f'{(time.perf_counter_ns() - start) / 1e6:.3f} ms')

# set call forwarding to default for all users
with ThreadPoolExecutor() as pool:
    # default call forwarding settings
    forwarding = PersonForwardingSetting.default()

    # schedule update for each user and wait for completion
    start = time.perf_counter_ns()
    list(pool.map(
        lambda user: api.person_settings.forwarding.configure(person_id=user.person_id,
                                                              forwarding=forwarding),
        calling_users))
    print(f'Reset call forwarding to default for {len(calling_users)} users in '
          f'{(time.perf_counter_ns() - start) / 1e6:.3f} ms')
```

Examples

- queue_helper.py: read/update call queue join state of users
- call_intercept.py: read/update call intercept settings of a user
- us_holidays_async.py: create schedule with US holidays for all US location
- reset_call_forwarding.py: reset call forwarding settings for all users
- users_wo_devices.py: identify users without devices
- catch_tns.py: pool unassigned TNs on hunt groups to catch calls to unassigned TNs
- Room_devices.py: remove calling entitlements from selected Workspaces

Example: User Web Portal

User Web Portal

- Only a limited set of configuration options is available for users on settings.webex.com
- User portal can expose additional options (which usually require admin privileges)
- Perfect example for using service app tokens
    - User does not have the required privileges → integration token granted by user not sufficient

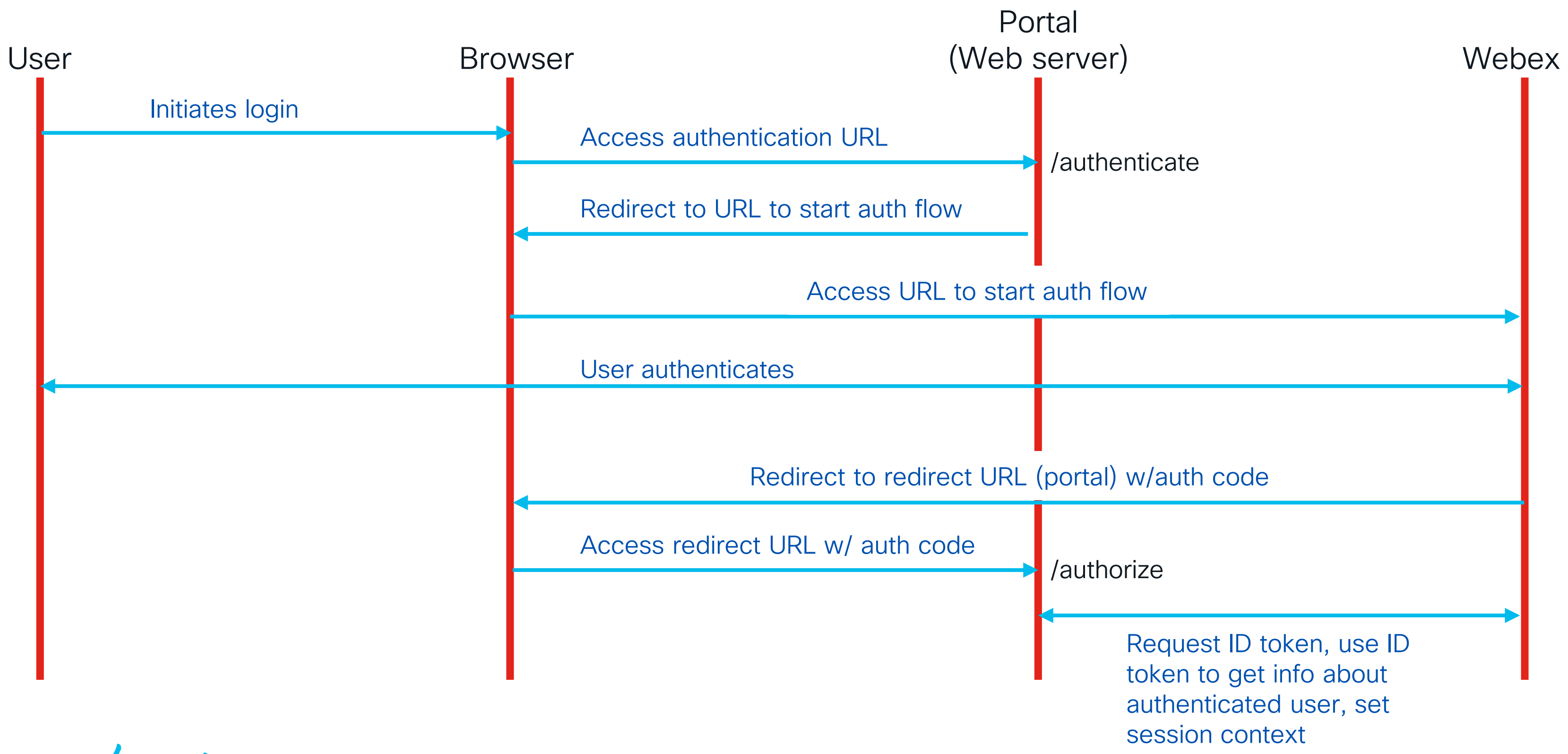
Concepts

- Web based portal
- Authenticate users using “Login with Webex”
- For provisioning operations portal uses service app tokens
- Browser only interacts with portal endpoints
  - Includes Javascript Ajax access
  - Portal acts as proxy between browser and Webex APIs

https://developer.webex.com/docs/login-with-webex
https://github.com/jeokrohn/BRKCOL-3015

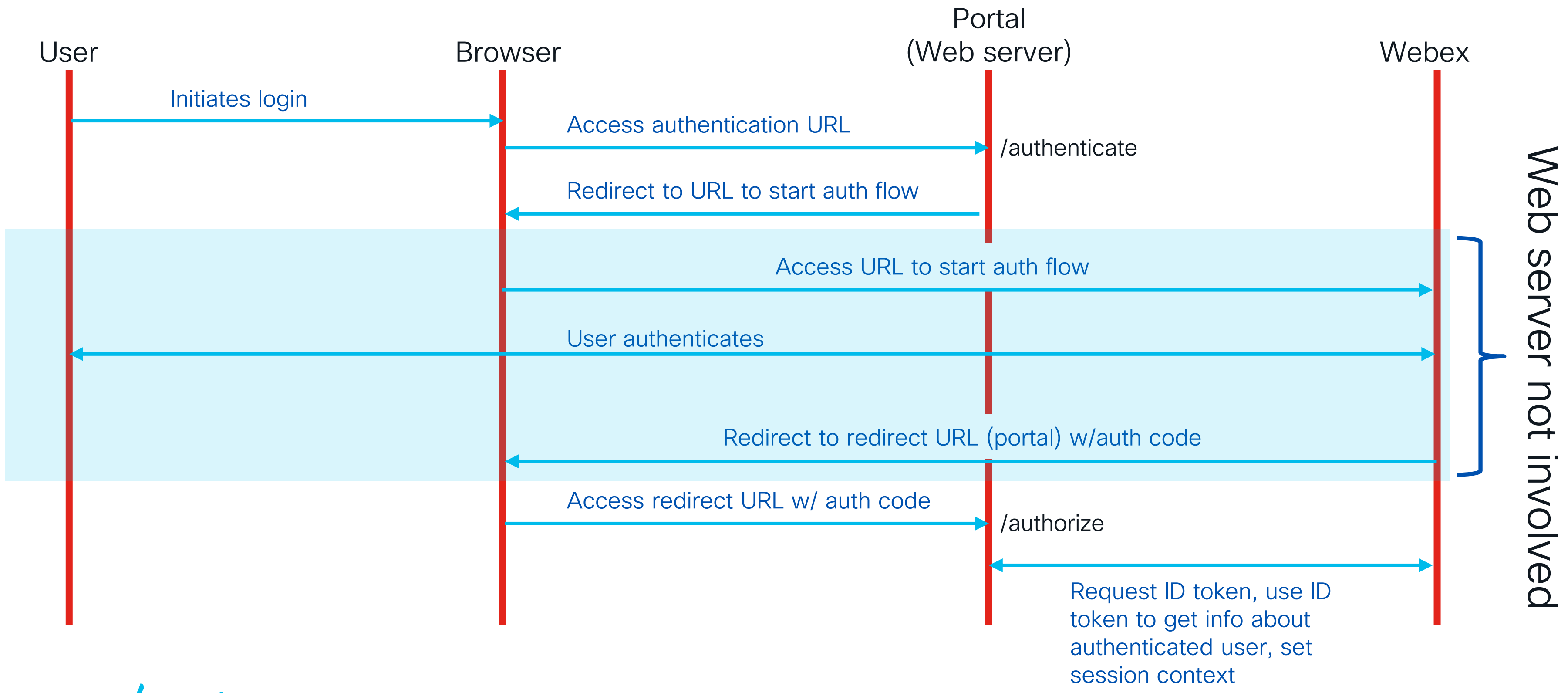
© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Login with Webex

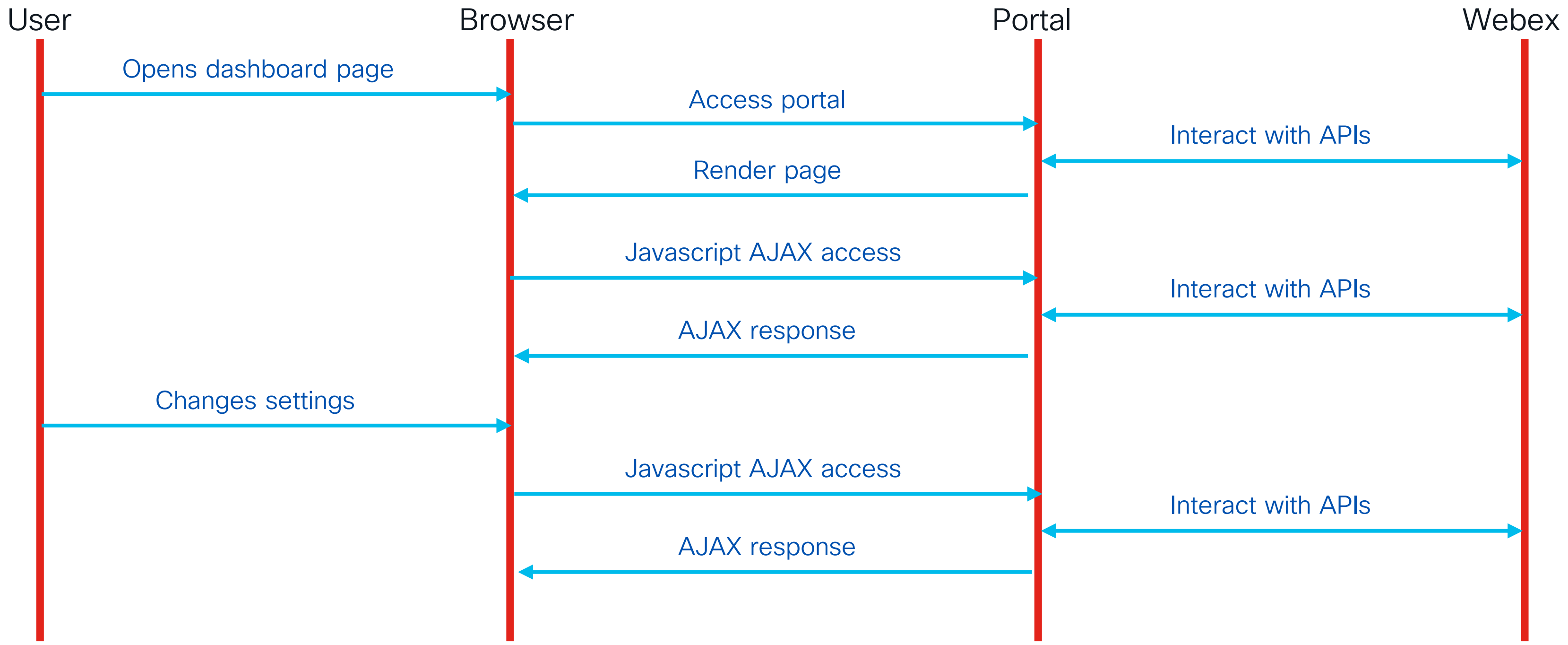

User
Browser
Portal (Web server)
Webex
Initiates login
Access authentication URL
/authenticate
Redirect to URL to start auth flow
Access URL to start auth flow
User authenticates
Redirect to redirect URL (portal) w/auth code
Access redirect URL w/ auth code
/authorize
Request ID token, use ID token to get info about authenticated user, set session context

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Login with Webex


User
Browser
Portal
(Web server)
Webex
Initiates login
Access authentication URL
/authenticate
Redirect to URL to start auth flow
Access URL to start auth flow
User authenticates
Redirect to redirect URL (portal) w/auth code
Access redirect URL w/ auth code
/authorize
Request ID token, use ID token to get info about authenticated user, set session context
Web server not involved

Portal Transactions



© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

Demo Web App

Disclaimer, Where to Go Next

- Demo code is not “production ready”
  - Missing token lifetime monitoring
  - Filesysten based session backend
  - …
- Where to go next
  - Add roles: can be based on groups in Webex
  - Additional functions: manage devices, bulk update, …

Closing

References

- Webex for Developers: https://developer.webex.com/
- Examples for the session on GitHub
  https://github.com/jeokrohn/BRKCOL-3015
- Python SDK: https://pypi.org/project/wxc-sdk/
- SDK Examples available at:
  https://wxc-sdk.readthedocs.io/en/latest/examples.html
  https://github.com/jeokrohn/wxc_sdk/tree/master/examples
- Call control bot: https://wxc-callcontrol.readthedocs.io/en/latest/

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

The bridge to possible

Thank you

Let’s go