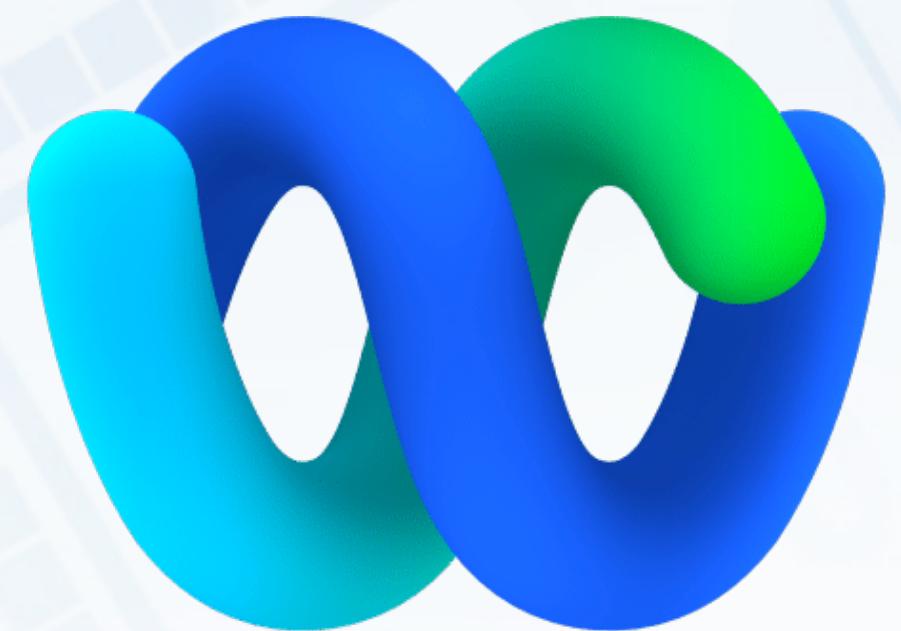




Doing Cool Things with the Cisco Webex API

Part 2 - Programmatically Retrieving Data



About Me



Josh Kittle

Solutions Architect for Collaboration

20+ years of experience supporting Cisco Collaboration technologies

jkittle@conres.com

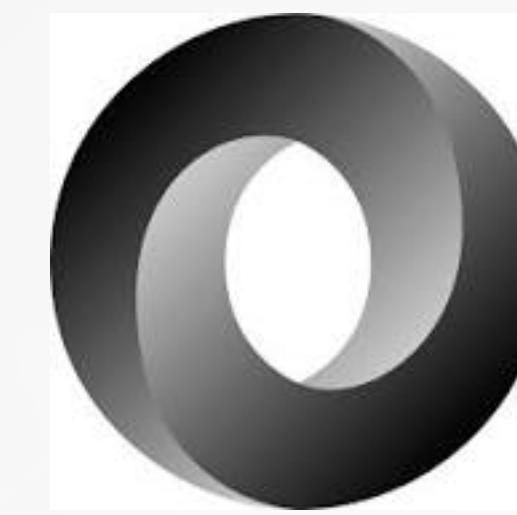
Who is this for?

- Anybody supporting a Webex environment
- Desire to use the API to ‘Do Cool Things’
- Interest in automation / bulk provisioning changes

What will we cover?

- JSON Basics
- Creating an Integration
- OAuth Grant Process
- Using Postman to HELP YOU Write Working Code
- Retrieving Basic Data using the Requests library using Python

What is JSON?



- JSON (JavaScript Object Notation) is a lightweight data-interchange format.
- It is easy for humans to read and write.
- It is easy for machines to parse and generate

```
{  
    hey: "guy",  
    anumber: 243,  
    - anobject: {  
        whoa: "nuts",  
        - anarray: [  
            1,  
            2,  
            "thr<h1>ee"  
        ],  
        more: "stuff"  
    },  
    awesome: true,  
    bogus: false,  
    meaning: null,  
    japanese: "明日がある。",  
    link: http://jsonview.com,  
    notLink: "http://jsonview.com is great"  
}
```

JSON Object Structure

- A JSON object contains zero, one, or more key-value pairs, also called properties.
- The object is surrounded by curly braces { }.
- Every key-value pair is separated by a comma. The order of the key-value pair is irrelevant.
- A key-value pair consists of a key and a value, separated by a colon (:)

```
{'grant_type': 'authorization_code',
 'client_id' : client_id,
 'client_secret' : client_secret,
 'code' : code,
 'redirect_uri' : redirect_uri}
```

REST API Process Simplified....

- 1. Register your Integration with Webex - you'll be issued 'Client Credentials'**
 - 1. Client ID**
 - 2. Client Secret**
- 2. Request permission using an OAuth Grant Flow**
- 3. Exchange the resulting authorization 'code' for an access 'token'**
- 4. Use your 'token' to make API calls.**

Creating an Integration

<https://developer.webex.com/my-apps/new>

New Integration

Integration

Request OAuth to invoke Webex APIs on behalf of another user.

[Create an Integration](#)

[Learn More](#)

Will this integration use a mobile SDK?
(Webex iOS or Android SDK 3.0+)

No
 Yes

Integration name*
Name of your integration as it will appear in Webex.
e.g. My App

Icon*
Upload your own or select from our defaults. Must be exactly 512x512px in JPEG or PNG format.
Upload Default 1 Default 2 Default 3

App Hub Description*
What does your app do, how does it benefit users, how do users get started? Does your app require a non-Webex account? If your app is not free or has additional features for paid users, please note that and link to pricing information. 1024

Redirect URI(s)*
One or more URLs that a user will be redirected to when completing an OAuth grant flow.
[Learn more](#)

e.g. https://www.example.com

[+ Add URI](#)

Scopes

Scopes define the level of access that your integration requires.
[Learn more](#)

meeting:schedules_read
Retrieve your Webex meeting lists

meeting:schedules_write
Create, manage, or cancel your sc

meeting:recordings_read
Retrieve your Webex meeting recc

Your Completed Integration

Take note of the following values

- Client ID
- Client Secret
- OAuth Authorization URL

(Includes the callback URL)

Demo Integration

OAuth settings

Learn more about authentication in the [Apps & OAuth Guide](#).

Client ID

C1934caa8bcc778899e006424f280aeb193c0d8991b454e

[Copy](#)

Client Secret

[Regenerate the client secret](#)

OAuth Authorization URL

You can use the URL below to initiate an OAuth permission request for this app. It is configured with your redirect URI and app scopes. Be sure to update the state parameter.

```
https://webexapis.com/v1/authorize?  
client_id=C1934caa8bcc778899e006424f280aeb193c0d8991b45  
4e2c4c5aa2054144d6ac&response_type=code&redirect_uri=http  
%3A%2F%2Fwww.cisco.com&scope=spark-  
admin%3Abroadworks_subscribers_write%20meeting%3Aadmin_  
preferences_write%20spark%3Aall%20meeting%3Aadmin_prefere  
nces_read%20analytics%3Aread_all%20meeting%3Aadmin_partici  
pants_read%20spark-  
admin%3Apeople_write%20spark%3Apeople_write%20spark%3Ao  
_
```

Integration ID

Unique system generated ID for your integration.

Y2lzY29zcGFyazovL3VzL0FQUExJQ0FUSU9OL0MxOTM0Y2

[Copy](#)

Integration name*

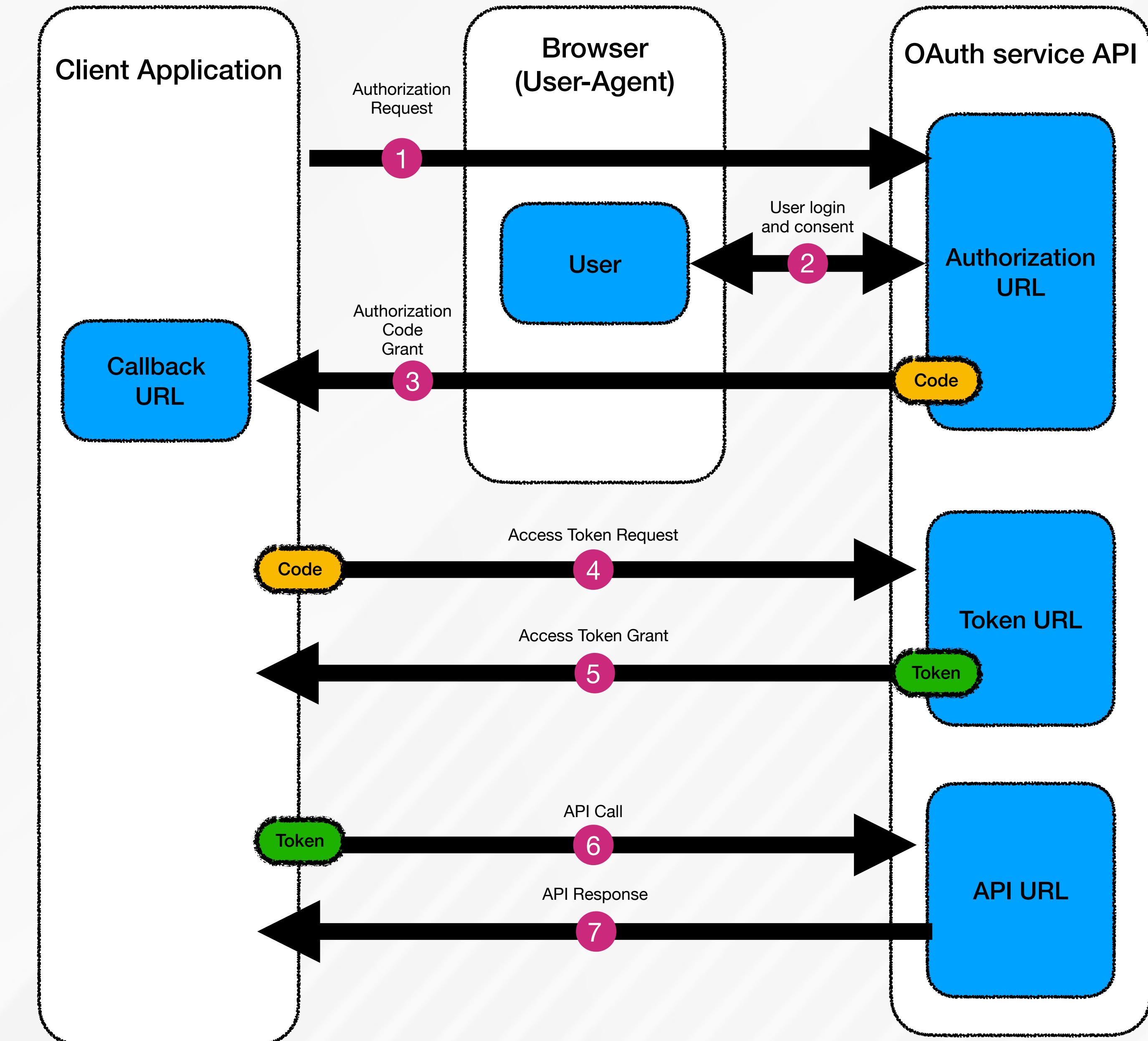
Name of your integration as it will appear in Webex and Webex App Hub.

Demo Integration

[Edit](#)

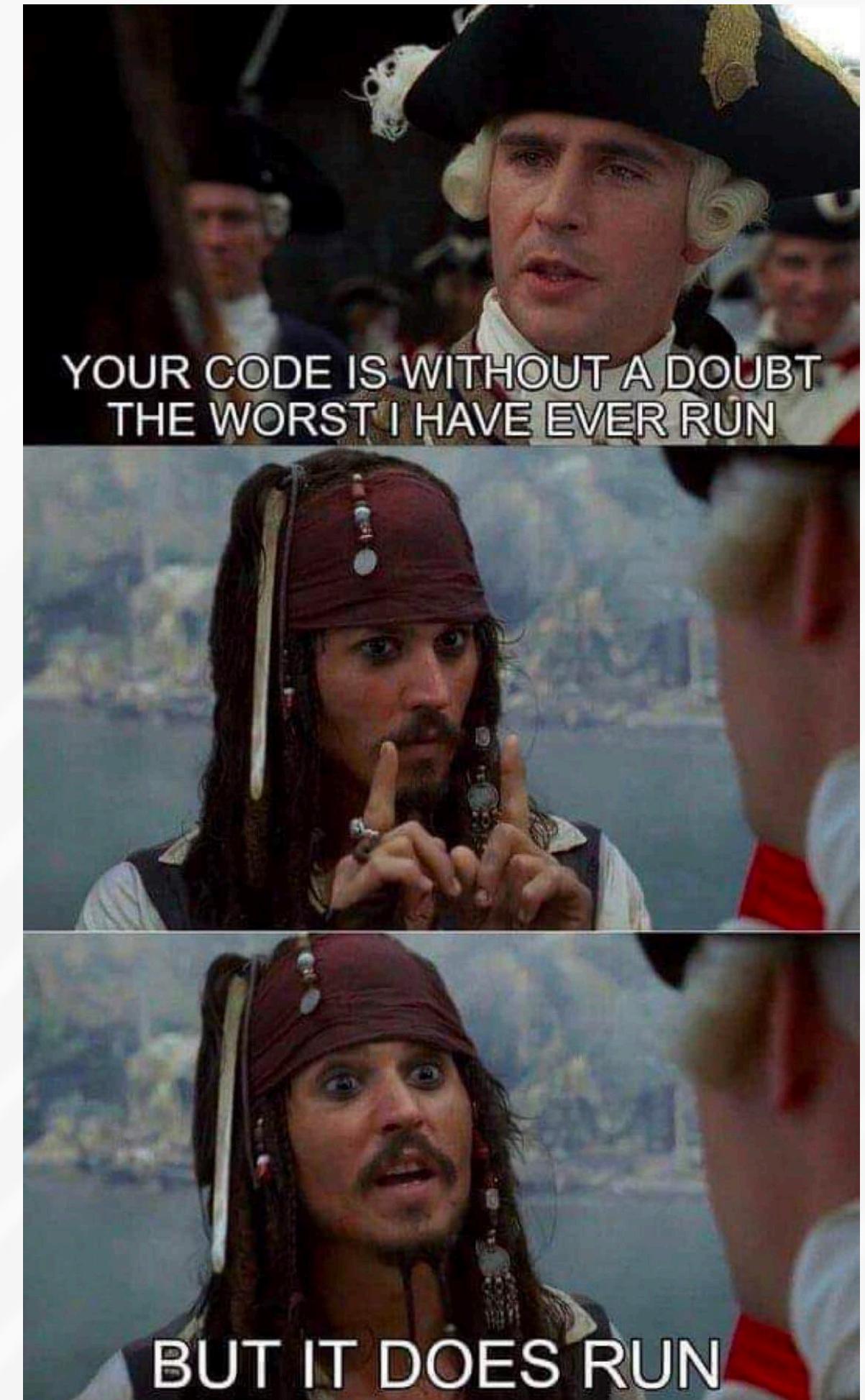
What is OAuth?

- An open protocol to allow secure authorization in a simple and standard method for web, mobile, and desktop applications



On the topic of writing code...

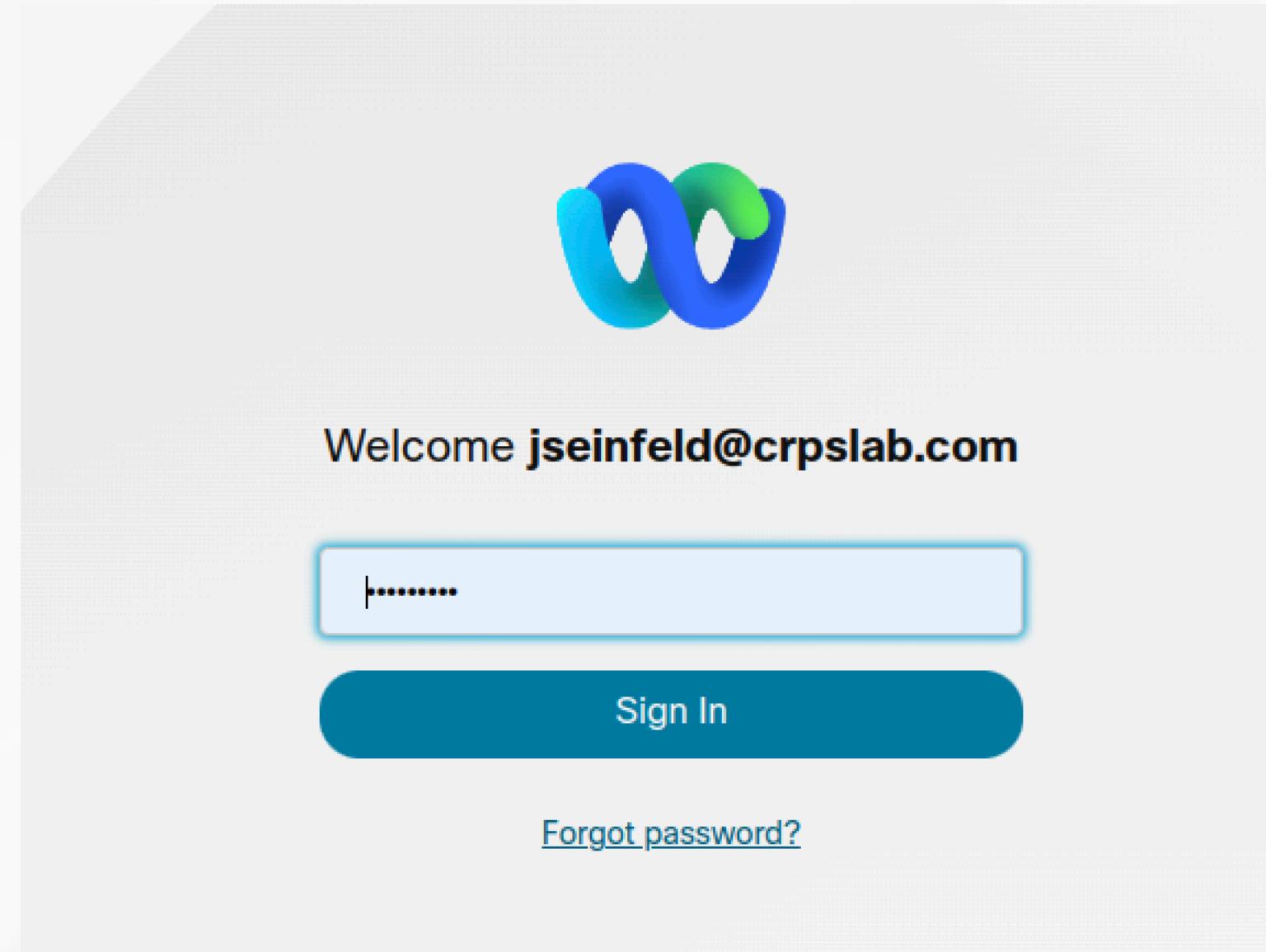
- We don't need to be expert programmers to 'get things done'
- We will share some examples that you can use to hit the ground running
- <https://github.com/conresinc/cool-stuff-webex-api>



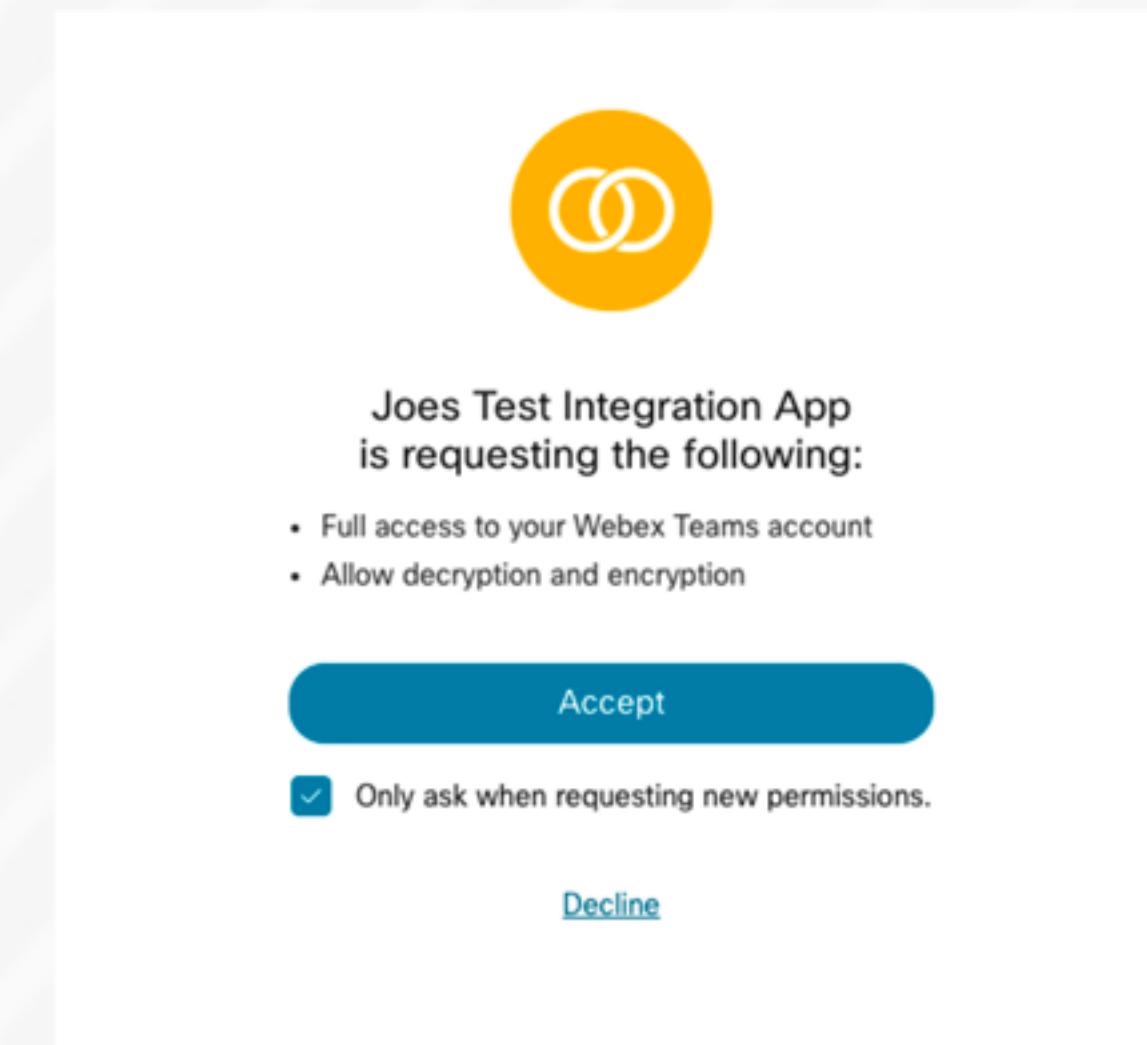
grant.php (Available on our GitHub!)

```
1  <?php
2  $oauthAuthorizationURL ="Paste Value from OAuth Authorization URL Here in Double Quotes";
3  ?>
4
5  <!DOCTYPE html>
6  <html>
7  |  <head>
8  |  |  <title>Get a Webex OAuth authorization code</title>
9  |  |  <meta charset='utf-8'>
10 |  </head>
11 |  <body>
12 |  |  <h1>Get a Webex OAuth authorization code</h1>
13 |  |  </div>
14 |  |  <div class='spacer-small'></div>
15 |  |  <div class='center'>
16
17 <?php
18 echo "<a href=\"\" . $oauthAuthorizationURL . "\">";
19 ?>
20
21 |  |  |  <div class='button' style='width:512px;'>GRANT</div>
22 |  |  |  </a>
23 |  |  |  </div>
24 |  |  </div>
25 |  </section>
26 |  </body>
27 </html>
```

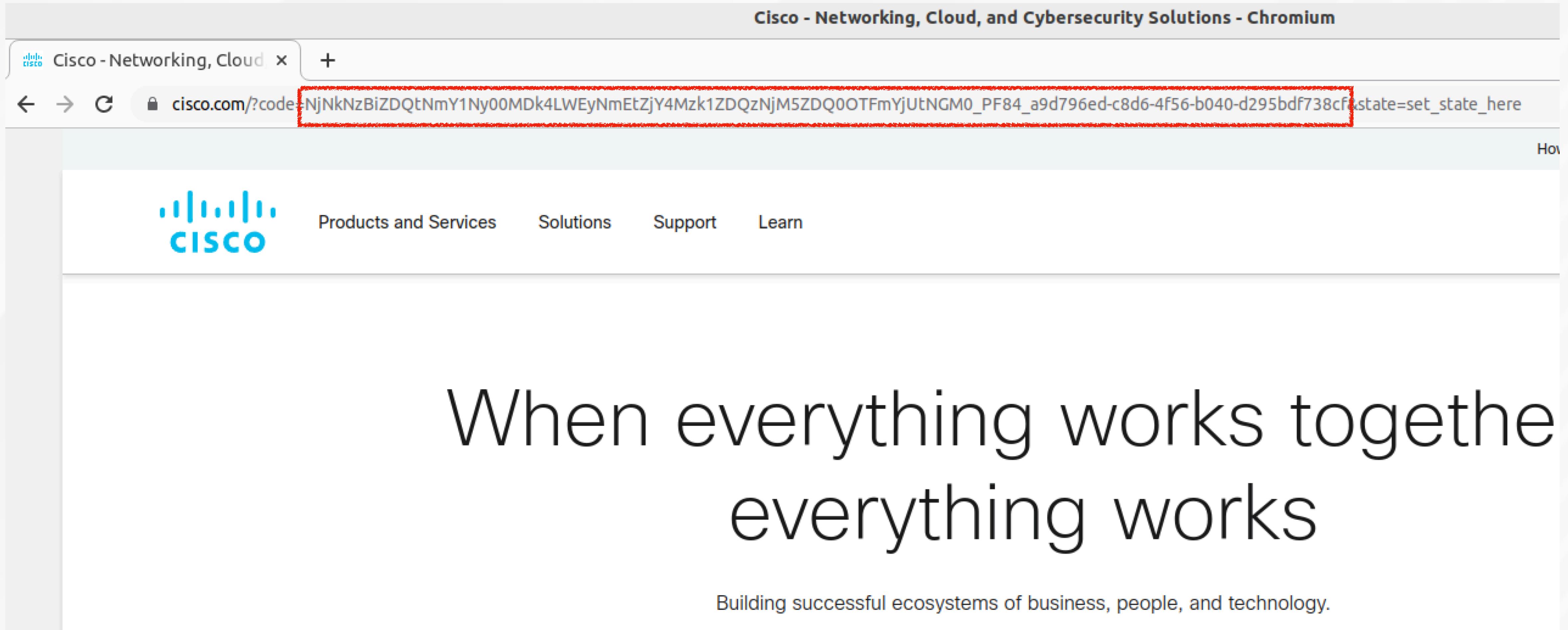
What it looks like



- The ‘request for access’ page can be quite lengthy, depending on the roles and permissions being granted.



Getting your authorization code



The screenshot shows a web browser window with the title "Cisco - Networking, Cloud, and Cybersecurity Solutions - Chromium". The address bar contains the URL "cisco.com/?code=NjNkNzBiZDQtNmY1Ny00MDk4LWEyNmEtZjY4Mzk1ZDQzNjM5ZDQ0OTFmYjUtNGM0_PF84_a9d796ed-c8d6-4f56-b040-d295bdf738cf&state=set_state_here", which is highlighted with a red box. The page itself displays the Cisco logo and navigation links for "Products and Services", "Solutions", "Support", and "Learn". Below the navigation, a large slogan reads "When everything works together, everything works". At the bottom, a tagline states "Building successful ecosystems of business, people, and technology."

Cisco - Networking, Cloud, and Cybersecurity Solutions - Chromium

Cisco - Networking, Cloud +

cisco.com/?code=NjNkNzBiZDQtNmY1Ny00MDk4LWEyNmEtZjY4Mzk1ZDQzNjM5ZDQ0OTFmYjUtNGM0_PF84_a9d796ed-c8d6-4f56-b040-d295bdf738cf&state=set_state_here

How

CISCO

Products and Services Solutions Support Learn

When everything works together, everything works

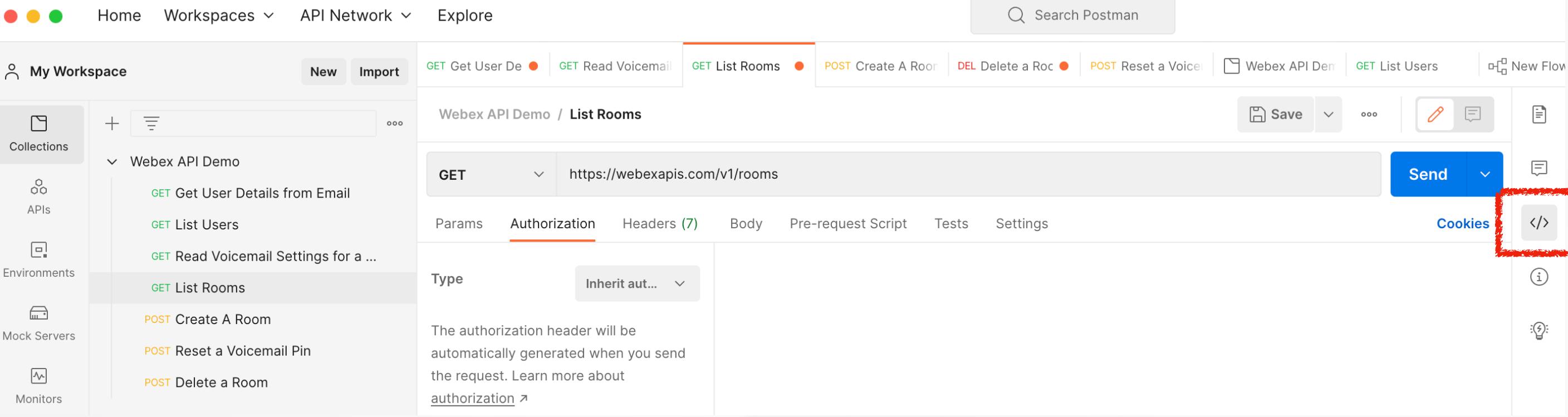
Building successful ecosystems of business, people, and technology.

getTokenFromCode.py (Available on our GitHub!)

```
1 import requests
2 import json
3
4 client_id = "Your Client ID in double quotes"
5 client_secret = "Your Client Secret in double quotes"
6 code = "Your OAuth Authorization Code in double quotes"
7 redirect_uri = 'http://www.cisco.com'
8
9 API_BASE_URL = "https://webexapis.com/v1/access_token"
10
11
12 def getTokenFromCode(client_id,client_secret,code,redirect_uri):
13     data = {'grant_type': 'authorization_code',
14             'client_id' : client_id,
15             'client_secret' : client_secret,
16             'code' : code,
17             'redirect_uri' : redirect_uri}
18
19     bodyPayloadText = json.dumps(data)
20     API_HEADERS = { 'Content-Type' : 'application/json'}
21     API_RESPONSE = requests.post(API_BASE_URL, headers=API_HEADERS, data = bodyPayloadText)
22     access_token = API_RESPONSE.json()
23     return access_token
24
25 results = getTokenFromCode(client_id,client_secret,code,redirect_uri)
26 print("Access Token is: ")
27 print(results['access_token'])
28 print("\n")
29 print("Complete JSON Response: \n")
30 print(results)
31
```

Coding Tip - Don't work too hard!

This is a Standard API Call in Postman

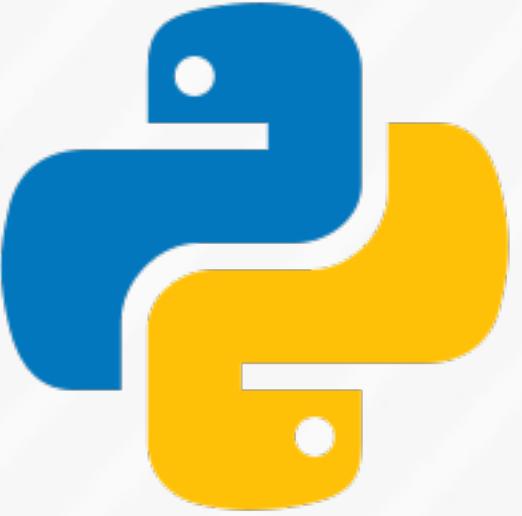


The screenshot shows the Postman interface. On the left, the sidebar includes 'My Workspace' with 'Webex API Demo' selected, containing various API endpoints like 'Get User Details from Email', 'List Users', 'Read Voicemail Settings', 'List Rooms', 'Create A Room', 'Reset a Voicemail Pin', and 'Delete a Room'. The main workspace shows a 'List Rooms' request: Method 'GET', URL 'https://webexapis.com/v1/rooms', Headers tab selected, and a note: 'The authorization header will be automatically generated when you send the request. Learn more about authorization'. The 'Cookies' button in the top right of the request panel is highlighted with a red box and labeled with the text '<- Click This Button!'. To the right, there's a 'Code snippet' panel titled 'Python - Requests' containing the following code:

```
1 import requests
2 url = "https://webexapis.com/v1/rooms"
3 payload={}
4 headers = {
5     'Authorization': 'Bearer
6         Nzg0ZTU0MmEtZmE1ZC00MzI4LWE2ZDIzZDMyOT
7         M5Y2Jm0WYzzWYWmNiZDctNzMw_PF84_a9d796
8         ed-c8d6-4f56-b040-d295bdf738cf'
9 }
10 response = requests.request("GET", url,
11                             headers=headers, data=payload)
12 print(response.text)
13 
```

- Code examples for the language of your choice will be revealed!
- In this example we are looking at a python script that's using the requests library to make an API call

Python3 Demo Code



- Obtain a user id key from an email address
- Get a list of phone numbers for a user, from the user id key
- Read the voicemail configuration for a user, from the user id key

getUserIdFromEmail.py (Available on our GitHub!)

```
1 import requests
2
3 email_address = "jseinfeld@crpslab.com"
4 bearer_token = "Enter Your Token Here in Double Quotes"
5
6 API_HEADER = {'Authorization': 'Bearer ' + bearer_token,
7               'Content-Type' : 'application/json',
8               'Accept' : '*/*'}
9
10 API_RESPONSE = requests.get('https://webexapis.com/v1/people/?email=' + email_address, headers=API_HEADER, verify=True)
11
12 if API_RESPONSE.json()["items"]:
13     userId = API_RESPONSE.json()["items"][0]["id"]
14     print(userId)
15 else:
16     print("No User Found")
17
```

getPhoneNumbersFromEmail.py (Available on our GitHub!)

```
1 import requests
2 import json
3
4 email_address = "jseinfeld@crpslab.com"
5 bearer_token = "Enter Your Token Here in Double Quotes"
6
7 API_HEADER = {'Authorization': 'Bearer ' + bearer_token,
8               'Content-Type' : 'application/json',
9               'Accept' : '*/*'}
10
11 API_RESPONSE = requests.get('https://webexapis.com/v1/people/?email=' + email_address, headers=API_HEADER, verify=True)
12
13 if API_RESPONSE.json()["items"]:
14     userId = API_RESPONSE.json()["items"][0]["id"]
15     print ("The user ID for the email address " + email_address + " is :" + userId + "\n")
16 else:
17     print("No User Found")
18
19
20 API_RESPONSE = requests.get('https://webexapis.com/v1/people/' + userId, headers=API_HEADER, verify=True)
21
22 results = json.loads(API_RESPONSE.text)
23
24 print("The phone number associated with the email address " + email_address + " is "+ results['phoneNumbers'][0]['value'])
```