

300-835^{Q&As}

Automating Cisco Collaboration Solutions (CLAUTO)

Pass Cisco 300-835 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.leads4pass.com/300-835.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Cisco
Official Exam Center

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers



QUESTION 1

DRAG DROP



Refer to the exhibit. A developer has implemented ChatOps to a Webex Teams space as described in the exhibit. The Python script that pushes the notifications to the Teams space is shown. Drag and drop the code to complete the script. Not all options are used.

Select and Place:

Answer Area

```
import requests, json, os

header = { [ ] : [ ] % os.environ.get("BOT_TOKEN"),
           "Content-Type": "application/json"}

message = "_The bot says:\n> I am a developer too!"

payload = {"roomId": os.environ.get("SPACE_ID"),
           "markdown": message}

res = requests.request([ ], url="https://api.ciscospark.com/v1/messages",
                       headers=header, data=[ ], verify=True)

if res.[ ] == 200:
    print("your message was successfully posted to Webex Teams")
else:
    print('failed with status code: %d' %res.status_code)

    if res.status_code == 404:
        print("please check the bot is in the space you're attempting to post to...")
    elif res.status_code == 400:
        print("please check the identifier of the space you're attempting to post to...")
    elif res.status_code == 401:
        print("please check if the access token is correct...")
```

json.dumps(payload)	"PUT"	status_code	payload
"Authentication"	"Authorization"	"Bearer"	status
"Bearer %s"	json(payload)	"POST"	code

Correct Answer:

Answer Area

```
import requests, json, os

header = { "Authentication" : "Bearer" % os.environ.get("BOT_TOKEN"),
           "Content-Type": "application/json"}

message = "_The bot says: \n> I am a **developer** too!"

payload = {"roomId": os.environ.get("SPACE_ID"),
           "markdown": message}

res = requests.request("POST", url="https://api.ciscospark.com/v1/messages",
                       headers=header, data= json.dumps(payload) , verify=True)

if res.status_code == 200:
    print("your message was successfully posted to Webex Teams")
else:
    print('failed with status code: %d' %res.status_code)

    if res.status_code == 404:
        print("please check the bot is in the space you're attempting to post to...")
    elif res.status_code == 400:
        print("please check the identifier of the space you're attempting to post to...")
    elif res.status_code == 401:
        print("please check if the access token is correct...")
```

	PUT	payload
	Authorization	status
Bearer %s	json(payload)	code

QUESTION 2

DRAG DROP

Drag and drop the correct items from the right to complete this Python script to automate the creation of Cisco Webex Teams spaces and memberships. Not all options are used.

Select and Place:

Answer Area

```
import requests

def makePost(apiKey, url, body):
    headers = {
        'Authorization': "Bearer " + apiKey,
        'Content-Type': "application/json"
    }
    response = requests.post(url = url, headers = headers, [redacted] = body)
    return response

def createSpace(apiKey, roomName):
    url = "https://api.ciscospark.com/v1/rooms/"
    body = {
        "title": roomName
    }
    response = makePost(apiKey, url, body)
    roomId = [redacted]
    return roomId

def addMembership(apiKey, roomId, membershipEmail):
    url = "https://api.ciscospark.com/v1/memberships/"
    body = {
        "roomId": roomId,
        "[redacted]": membershipEmail
    }

    makePost(apikey, url, body)

def main():
    apiKey = input("What is your personal access token? ")
    spacename = input("What is the name of the space you want to create? ")
    membershipEmail = input("What is the email address of the person to add? ")
    roomId = createSpace(apiKey, spacename)
    addMembership(apiKey, roomId, membershipEmail)

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

-
-
-
-
-
-
-
-
-
-
-

Correct Answer:

Answer Area

```
import requests

def makePost(apiKey, url, body):
    headers = {
        'Authorization': "Bearer " + apiKey,
        'Content-Type': "application/json"
    }
    response = requests.post(url = url, headers = headers, data = body)
    return response

def createSpace(apiKey, roomName):
    url = "https://api.ciscospark.com/v1/rooms/"
    body = {
        "title": roomName
    }
    response = makePost(apiKey, url, body)
    roomId = response.id
    return roomId

def addMembership(apiKey, roomId, membershipEmail):
    url = "https://api.ciscospark.com/v1/memberships/"
    body = {
        "roomId": roomId,
        "personEmail": membershipEmail
    }

    makePost(apikey, url, body)

def main():
    apiKey = input("What is your personal access token? ")
    spacename = input("What is the name of the space you want to create? ")
    membershipEmail = input("What is the email address of the person to add? ")
    roomId = createSpace(apiKey, spacename)
    addMembership(apiKey, roomId, membershipEmail)

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

-
-
-
-
-
-
-

QUESTION 3


```
import xows
import asyncio

async def start(ip, usr, pw):
    async with xows.XoWSClient(ip, username=usr, password=pw) as client:
        async def callback(data, id_):
            print(f'Feedback {id}(Id {id_}): {data}')
            
        await client.wait_until_closed()

async def task():
    await start('10.10.10.1', 'admin', 'T357c45e')

asyncio.run(task())
```

Refer to the exhibit. This Python script uses the websocket-based, xAPI library "pyxows" to monitor and print event details to the console when users interact with UI Extension/In-Room Control `Action Buttons` deployed to the Touch-10 interface of a Webex room device. Which two code snippets successfully capture such events? (Choose two.)

- A. `await client.xEvent(['Event', '**'], callback)`
- B. `await client.subscribe(['Event', `ActionButton`, `Clicked`], callback)`
- C. `await client.subscribe(['Event', '**'], callback)`
- D. `await client.xEvent(['Event', `UserInterface`, `Extensions`, `Panel`, `Clicked`], callback)`
- E. `await client.subscribe(['Event', `UserInterface`, `Extensions`, `Panel`, `Clicked`], callback)`

Correct Answer: DE

QUESTION 4

What are two key features of Cisco Unified Communications Manager Serviceability APIs? (Choose two.)

- A. save alarms and events for troubleshooting
- B. connect to serial console on phone hardware
- C. configure firewall rules on phone hardware
- D. scan servers and devices for malware
- E. real-time monitoring of components

Correct Answer: AE

QUESTION 5

What is a benefit of using Python virtual environments?

- A. It isolates dependencies of every project from the system and each other.
- B. It allows Python to differentiate between package versions.
- C. It frees the developer from installing the project dependencies.
- D. It puts dependent packages in a common site-packages directory.

Correct Answer: A

[300-835 PDF Dumps](#)

[300-835 Exam Questions](#)

[300-835 Braindumps](#)