

300-835 Valid Braindumps Files | 300-835 Flexible Learning Mode



The advertisement features a purple background. At the top, the word "TOPIK" is written in large, pink, block letters. To its right is a pink starburst containing the number "5" and a blue star. Below this, there are two images of women: one in a classroom setting and another wearing glasses with a speech bubble that says "HELLO WORLD!". To the right of these images, the text reads "REVIEW CLASSES", "Preparing for TOPIK 2025?", "Be ready to ace your Test of Proficiency in Korean! Achieve your desired TOPIK level by preparing in advance!", and "ENROLL NOW!" in a pink box.

2026 Latest ExamBoosts 300-835 PDF Dumps and 300-835 Exam Engine Free Share: https://drive.google.com/open?id=1oNWPTFAxOi-E6DLyzWb1AY2_rHN4EkrR

We recognize that preparing for the Cisco Certification Exams can be challenging, and that's why we provide Cisco 300-835 practice material with three formats that take your individual needs into account. Our team of experts is dedicated to helping you succeed by providing you with the support you need while using the product.

ExamBoosts 300-835 exam dumps in three different formats has 300-835 questions PDF and the facility of Cisco 300-835 dumps. We have made these Cisco 300-835 questions after counseling a lot of experts and getting their feedback. The 24/7 customer support team is available at ExamBoosts for Cisco 300-835 Dumps users so that they don't get stuck in any hitch.

300-835 Flexible Learning Mode & Reliable 300-835 Test Cost

If you prefer to prepare for your exam on paper, then our 300-835 exam materials will be your best choice. 300-835 PDF version is convenient to read and printable, and you can take them with you, and you can practice them anywhere and anyplace. Besides, free demo for 300-835 PDF version is available, and you can try before buying. We are pass guarantee and money back guarantee and if you fail to pass the exam. You can receive the downloading link and password for 300-835 Training Materials within ten minutes for 300-835 exam materials, if you don't receive, you can contact with us, and we will solve the problem for you.

Cisco Automating and Programming Cisco Collaboration Solutions Sample Questions (Q75-Q80):

NEW QUESTION # 75

Which two files must be downloaded from a Cisco Unified Communications Manager server in order to build requests for the AXL API? (Choose two.)

- A. AXLSOap.xsd
- B. CUCM_AXL.json
- C. AXLEnums.xsd
- D. AXL_CUCM.xsd
- E. AXLWSDL.xml

Answer: A,C

Explanation:

Section: Unified Communication

NEW QUESTION # 76

Drag and Drop Question

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.

Answer Area

```
import requests

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

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

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

Answer:

Explanation:

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()
```

response['id']

json

data

membershipEmail

apiKey

Authentication

OAuth

response[body]['id']

NEW QUESTION # 77

Refer to the exhibit. Which code snippet must be added to the blank in the script to create a new Cisco Webex space?

```
import requests
import json

url = 'https://api.ciscospark.com/v1/rooms'
access_token = 'TQxMjnJMWQyOTC4ZDUONzJmOTEtNjIxPF84_leb6Sfdf-9643-417f-9974-ad72caede10f'
refresh_token = 'PF84_leb6Sfdf-9643-417f-9974-ad72cae0e10f_TQxMjnJMWQyOTC4ZDUONzJmOTEtNjIx'

room_title = 'My new Teams room'
```

```
response = requests.post(
    url,
    headers={'Authorization': 'Bearer ' + refresh_token,
            'Content-type': 'application/json;charset=utf-8'},
    data = json.dumps({'title': room_title})
)
```

- A.

- B.

```
response = requests.post(
    url,
    headers={'Authorization': 'Bearer ' + access_token,
            'Content-type': 'application/json;charset=utf-8'},
    data = json.dumps({'title': room_title})
)
```

- C.

```
response = requests.put(
    url,
    headers={'Authorization': 'Bearer ' + access_token,
            'Content-type': 'application/json;charset=utf-8'},
    data = json.dumps({'title': room_title})
)
```

- D.

```
response = requests.post(
    url,
    headers={'Authorization': 'Bearer ',
            'Content-type': 'application/json;charset=utf-8'},
    data = json.dumps({'title': room_title})
)
```

Answer: B

Explanation:

To create a new Webex space (room) using the Webex REST API, you must send a POST request to the /v1/rooms endpoint using a valid access token for authorization.

The correct code snippet is:

```
response = requests.post(
    url,
    headers={'Authorization': 'Bearer ' + access_token,
            'Content-type': 'application/json;charset=utf-8'},
    data = json.dumps({'title': room_title})
)
```

This uses the correct HTTP method (POST), authenticates with the correct token (access_token), and formats the JSON body properly with the required title field.

NEW QUESTION # 78

Refer to the exhibit. A Webex Meetings XML API HTTP request message with several invalid portions is shown. Which reference points to a line in the exhibit that correctly indicates that this is a LstsummaryUser request?

```

1 POST /WEBService/XMLService/LstsummaryUser HTTP/1.1
2 Content-Type: application/xml
3 Host: api.webex.com
4 Content-Length: 974
5
6 <?xml version="1.0" encoding="UTF-8"?>2 <serv:message xmlns:serv="http://www.webex.com/schemas/2002/06/service"
7   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8   <header>
9     <securityContext>
10      <siteName>apidemoeu</siteName>
11      <webExID>janedoe@example.com</webExID>
12      <sessionTicket>AAABb5LTcGcAABUYA0gAKEGyU</sessionTicket>
13    </securityContext>
14    <action>java.com.webex.service.binding.meeting.LstsummaryUser</action>
15  </header>
16  <body>
17    <method>java.com.webex.service.binding.meeting.LstsummaryUser</method>
18    <bodyContent xsi:type="java.com.webex.service.binding.meeting.LstsummaryUser">
19      <webExId>{{WEBEXID}}</webExId>
20    </bodyContent>
21  </body>
22 </serv:message>

```

- A. line 17
- B. line 14
- C. line 18
- D. line 1

Answer: C

NEW QUESTION # 79

Refer to the exhibit.

```

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_):
            if id_ == 0:
                <SNIPPET>

            await client.subscribe(['Status', 'Audio', 'Volume'], callback, True)
            await client.wait_until_closed()

asyncio.run(start('10.10.10.1', 'admin', 'T357c45e'))

```

An administrator is creating a script using the Python xAPI over WebSockets(pyxows) library. The goal of this script is to monitor the volume of the endpoint and set the volume to 60 whenever the volume has been set higher than that amount. Which code snippet accomplishes this task when it is added?

- A. `if data ['Status']['Audio']['Volume'] > 60:
 await client.xCommand(['Audio', 'Volume', 'Set'], Level=60)`
- B. `if data ['Status']['Audio']['Volume'] > 60:
 await client.xGet(['Status', 'Audio', 'Volume'])`
- C. `if data ['Status']['Audio']['Volume'] > 60:
 await client.subscribe(['Status', 'Audio', 'Volume'].callback, Level=60))`
- D. `if data ['Status']['Audio']['Volume'] > 60:
 await client.xSet(['Configuration', 'Audio', 'DefaultVolume'], 60))`

- A. Option A
- B. Option B
- C. Option D
- D. Option C

myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw,
www.stes.tyc.edu.tw, Disposable vapes

P.S. Free & New 300-835 dumps are available on Google Drive shared by ExamBoosts: https://drive.google.com/open?id=1oNWPTFAxOi-E6DLyzWb1AY2_rHN4EkrR