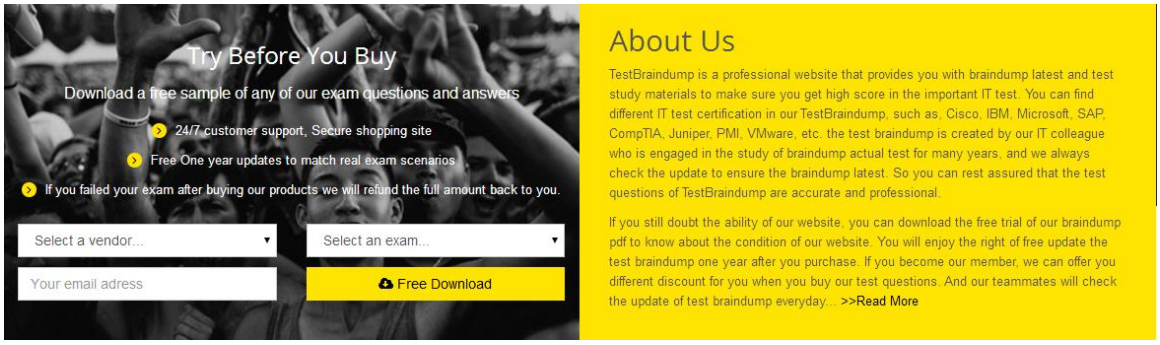
TestBraindump



http://www.testbraindump.com

Latest test braindump, braindump actual test

Exam : **300-835**

Title : Automating and Programming Cisco Collaboration Solutions

Vendor : Cisco

Version : DEMO

NO.1 What is the primary purpose of the POST /v1/meetings function of the Webex Meetings REST API?

A. to join an ongoing meeting

B. to cancel a meeting

C. to schedule a new meeting

D. to start an instant meeting

Answer: C

Explanation:

The POST /v1/meetingsfunction of the Webex Meetings REST API is used to create or schedule a new meeting. You can define details such as title, agenda, start time, duration, password, and invitees when calling this endpoint.

NO.2 Which two scenarios could be automated by combining the Webex Meetings XML API with other API-enabled systems? (Choose two.)

A. scheduling new-hire orientation meetings as part of an HR system workflow

B. triggering Webex meeting recording when the host disconnect from the meeting.

C. automatically launching the weekly Webex scrum meeting on Mondays at 9 AM

D. reassigning an employee's scheduled Webex meeting to their manager then they leave the company

E. muting users in a Webex meeting when their Cisco Jabber presence status transitions to Away

Answer: A,E

NO.3 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.

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

Answer:

Answer Area

```
import requests, json, os

header = { "Authorization"  :  "Bearer %s"  % 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= 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...")
```

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

NO.4 When the behavior of a Cisco collaboration device is customized, which use case requires an external control system because implementing JavaScript macro does not suffice?

A. Add a Join Webex meeting button to the touch panel.

B. Move the shutters up and down.

C. Trigger a "room-reset" to restore default configurations.

D. Implement an in-room control panel for speed-dialing.

Answer: B

NO.5 Which two capabilities can be implemented in a custom application using the Cisco Unified IP Phone Services API? (Choose two.)

A. Authenticate the phone to the network.

B. Display corporate directory information.

C. Play multicast messages.

D. Upgrade phone firmware.

E. Create new phone devices.

Answer: B,C

NO.6 Which Cisco Finesse API can control the communication between agents and servers?

A. serviceability

B. desktop

C. configuration
D. notifications
Answer: B
Explanation:
The Cisco Finesse desktop API is responsible for handling real-time communication between agents and servers, including call control, state changes, and agent actions on the desktop. It enables agents to interact with Cisco Finesse directly through browser-based or custom applications.

NO.7 Drag and drop the code snippets from the bottom onto the boxes where the code is missing to listen for Call History events using the xAPI Python SDK.
Not all options are used.
Select and Place:

Answer Area

```
import xows
import asyncio

async def start():
    async with xows. [          ] ('10.10.20.160', username='admin') as client:
        def callback(data, id_):
            print(f'Feedback (Id {id_}): {data}')

        await client. [          ] ([' [          ] ',
                          ' [          ] ', 'Updated'], callback, True)
        await client.wait_until_closed()

asyncio.run(start())
```

XAPIClient	Event
XoWSClient	CallHistory
subscribe	Updated
listen	

Answer:

Answer Area

```
import xows
import asyncio

async def start():
    async with xows.[XoWSClient] ('10.10.20.160', username='admin') as client:
        def callback(data, id_):
            print(f'Feedback (Id {id_}): {data}')

        await client.[subscribe] (['[Event]',
                                   '[CallHistory]', 'Updated'], callback, True)
        await client.wait_until_closed()

asyncio.run(start())
```

XAPIClient
XoWSClient
subscribe
listen

Event
CallHistory
Updated

NO.8 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_):
         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())
```

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)

Answer: D,E

NO.9 Which Git command sends local changes from every branch to the remote repository?

A. git add-all

B. git commit-m "All"

C. git push origin master

D. git push-all

Answer: D

Explanation:

Explanation:

The command git push--all (or git push-all) pushes all local branches to the remote repository. This ensures that changes from every branch - not just the current one - are sent to the remote.

NO.10 Refer to the exhibit.

```
import requests

def createUser(apiKey, email, firstName, lastName, displayName):
     url = "https://api.ciscospark.com/v1/people"
     headers = {
        'Authorization': "Bearer " + apiKey,
        'Content-Type': "application/json"
     }
     body = {
     [                                    ]
     }
     response = requests.post(url = url, headers = headers, json=body)
     return response
```

Which snippet of code does a user with the administrator role use in the missing "body" section to create a new user in a Webex Teams organization?

A.
```
"displayName": displayName,
"firstName": firstName,
"lastName": lastName
```

B.
```
"emails": {
    email
},
"displayName": displayName,
"firstName": firstName,
"lastName": lastName
```

C.
```
"email": email,
"displayName": displayName,
"firstName": firstName,
"lastName": lastName
```

D.
```
"emails": [
    email
],
"displayName": displayName,
"firstName": firstName,
"lastName": lastName
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer: D

NO.11 Which two characteristics of REST APIs? (Choose two.)
A. Resources are accessed using Uniform Resource Identifiers.
B. API operations for Create/Read/Update/Delete are mapped to standard HTTP methods.
C. Cookies are used for the duration of the session.
D. REST API extends Remote Procedure Call.
E. The server manages the session state.
Answer: A,B