KillTest

高品質　より良いサービス



練習問題

https://www.killtest.jp

1年間無料アップデート

Exam : **300-835**

Title : Automating and Programming Cisco Collaboration Solutions (CLAUTO)

Version : DEMO

1.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.
Answer: A

2.DRAG DROP
Drag and drop the botkit functions from the left onto the correct actions on the right.

Answer Area

hears()	Adds a question to the thread and evaluate the answer.
ask()	Sends a message with no pause for response.
say()	Replies to an incoming message.
reply()	Listens for keyword, phrases, or patterns in messages from users.

Answer:

Answer Area

hears()	ask()
ask()	say()
say()	reply()
reply()	hears()

3.Refer to the exhibit.

```
#!/usr/bin/env python
import os
import requests

token = os.environ.get('ACCESS_TOKEN')
url = "https://api.ciscospark.com/v1/messages"
roomId = "abcde12345"

mentionName = "                    "

markdownTxt = mentionName + ", your registration is confirmed"

payload = { "roomId: roomId, "markdown": markdownTxt }

headers = {
     "Authorization": "Bearer " + token,
     "Content-type": "application/json"
}

response = requests.request("POST", url, data=payload, headers = headers)
```

Which code snippet is used to mention the bot CiscoDevNet@webex.bot?

A. <@personEmail.CiscoDevNet@webex.bot>

B. @CiscoDevNet@webex.bot

C. @personEmail.CiscoDevNet@webex.bot

D. (@personEmail.CiscoDevNet@webex.bot)

Answer: B

4.DRAG DROP

An administrator is creating a script using the Python xAPI over WebSockets (py xows) library. The goal of the script is to capture an event generated by a UI Extensions action button (former In-Room Control Panel).

When the action button is clicked, the script displays an alert that says that the button ID was clicked.

Drag and drop the code snippets into the locations to complete the script:

Answer Area

```
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_):
            [                                                    ]
            [                                                    ]
        [                                                        ]
        await client.wait_until_closed()
async def task():
    [                                                            ]
[                                                                ]
```

```
await client.xCommand(['UserInterface','Message','Alert','Display'], Title=panelid, Text= "Was clicked")
panelId = data['Event']['UserInterface']['Extensions']['Panel']['Clicked']['PanelId']
await start('10.10.10.1','admin','T357c45e')
await client.subscribe(['Event','UserInterface','Extensions','Panel','Clicked'], callback, True)
asyncio.run(task())
```

Answer:

Answer Area

```
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_):
            panelId = data['Event']['UserInterface']['Extensions']['Panel']['Clicked']['PanelId']
            await client.xCommand(['UserInterface','Message','Alert','Display'], Title=panelid, Text= "Was clicked")
        await client.subscribe(['Event','UserInterface','Extensions','Panel','Clicked'], callback, True)
        await client.wait_until_closed()
async def task():
    await start('10.10.10.1','admin','T357c45e')
asyncio.run(task())
```

```
await client.xCommand(['UserInterface','Message','Alert','Display'], Title=panelid, Text= "Was clicked")
panelId = data['Event']['UserInterface']['Extensions']['Panel']['Clicked']['PanelId']
await start('10.10.10.1','admin','T357c45e')
await client.subscribe(['Event','UserInterface','Extensions','Panel','Clicked'], callback, True)
asyncio.run(task())
```

5.DRAG DROP

Drag and drop the code to complete the curl command that creates a halfwake brand logo using the xAPI HTTP API. Not all options are used. Code can be used more than one time.

```
curl -X POST "http://192.168.1.32/ [          ] "                              \
  -H "Content-Type: text/xml"                                                  \
  -H "Authorization: [          ] cmVtb3Rlc3VwcG9ydDo2US82a0JDYnJ0DQo="      \
  -d "<Command>
 <UserInterface>
         < [          ] >
              <Upload>
                        <Type>HalfwakeBranding</Type>
                        < [          ] >iVBORw0KGgoAAAANSUhEUgAAARAAAAEQCAYAAA
B4CisVAAAAAXNSR0IArs4c6QAAACBjSFJNAAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8
AAAACXBIWXMAABJ0AAASdAHeZh94AAABy2lUWHRYTUw6Y29CC</ [          ] >
              </Upload>
         </ [          ] >
 <UserInterface>
</Command>"
```

body	Branding	getxml
Basic	putxml	Brand
Configuration	Digest	Bearer

Answer:

```
curl -X POST "http://192.168.1.32/ [ putxml ] "                               \
  -H "Content-Type: text/xml"                                                 \
  -H "Authorization: [ Basic ] cmVtb3Rlc3VwcG9ydDo2US82a0JDYnJ0DQo="          \
  -d "<Command>
 <UserInterface>
         < [ Branding ] >
              <Upload>
                        <Type>HalfwakeBranding</Type>
                        < [ body ] >iVBORw0KGgoAAAANSUhEUgAAARAAAAEQCAYAAA
B4CisVAAAAAXNSR0IArs4c6QAAACBjSFJNAAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8
AAAACXBIWXMAABJ0AAASdAHeZh94AAABy2lUWHRYTUw6Y29CC</ [ body ] >
              </Upload>
         </ [ Branding ] >
 <UserInterface>
</Command>"
```

body	Branding	getxml
Basic	putxml	Brand
Configuration	Digest	Bearer