

# *KillTest*

품질은 좋고 서비스도 더욱 좋습니다



# 덤프

<http://www.killtest.kr>

우리는 고객에게 년 동안 무상업데이트 서비스를 제공합니다

**Exam** : **300-435**

**Title** : Automating and  
Programming Cisco  
Enterprise Solutions  
(ENAUTO)

**Version** : DEMO

1.Which two API calls are used to trigger a device configuration sync in Cisco DNA Center? (Choose two.)

- A. PUT /dna/intent/api/v1/network-device
- B. PUT /dna/intent/api/v1/network-device/sync-all
- C. PUT /dna/intent/api/v1/network-device/{networkDeviceId}/sync
- D. PUT /dna/intent/api/v1/network-device/sync
- E. POST /dna/intent/api/v1/network-device/{networkDeviceId}/sync

**Answer:** A,D

**Explanation:**

Reference: <https://github.com/CiscoDevNet/DNAC-JAVA-SDK/tree/master/DnacAppApi>

2.Which two Cisco DNA center features are needs to add legacy on the platform? (Choose two.)

- A. Multivendor SDK support
- B. Trusted device profile update
- C. Device package creation
- D. Device package download
- E. Device profile replication

**Answer:** A,D

3.Refer to the exhibit.

```
neighbors = ['s1', 's2', 's3']
switch = {'hostname':'nexus','os':'7.0.3','neighbors':neighbors}
print(switch['neighbors'][1])
```

What is the result when running the Python scripts?

- A. s1
- B. s2
- C. s1, s2, s3
- D. s3

**Answer:** B

4.Refer to the exhibit.

```
- name: Create VRFs as defined by local_vrfs
  ios_vrf:
    vrfs: "{{ local_vrfs }}"
    state: 
  register: addvrf
```

An engineer creates an Ansible playbook to configure VRF information using a local\_vrfs variable. The code must be completed so that it can be tested.

Which string completes the code?

- A. present
- B. up

- C. on
- D. active

**Answer: A**

**Explanation:**

Reference: [https://docs.ansible.com/ansible/latest/modules/ios\\_vrf\\_module.html](https://docs.ansible.com/ansible/latest/modules/ios_vrf_module.html)

**5.DRAG DROP**

Drag and drop the code from the bottom onto the box where the code is missing to construct an noiliest request that shuts down an interface on a Cisco IOS XE device. Not all options are used.

```

from ncclient import manager
import xml.dom.minidom
USERNAME = 'cisco'
PASSWORD = 'cisco'
HOST = '10.10.20.181'
data = ''
<config>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <interface>
      <GigabitEthernet>
        <name>{INTF_NAME}</name>
        <shutdown/>
      </GigabitEthernet>
    </interface>
  </native>
</config>
'''

with manager.connect(host=HOST, password=PASSWORD, port=830,
                    username=USERNAME, hostkey_verify=False,
                    [redacted] ) as m:

    c = m. [redacted] (data.format(INTF_NAME='3'),
                    format='xml',
                    [redacted] )

print(c)

```

device_params={'name':'iosxe'}
edit_config
target = 'running'

conn_params={'name':'cisco_iosxe'}
send_cmds
dst = 'running-config'

**Answer:**

```

from ncclient import manager
import xml.dom.minidom
USERNAME = 'cisco'
PASSWORD = 'cisco'
HOST = '10.10.20.181'
data = ''
<config>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <interface>
      <GigabitEthernet>
        <name>{INTF_NAME}</name>
        <shutdown/>
      </GigabitEthernet>
    </interface>
  </native>
</config>
'''

with manager.connect(host=HOST, password=PASSWORD, port=830,
                    username=USERNAME, hostkey_verify=False,
                    device_params={'name':'iosxe'}) as m:
    c = m. edit_config (data.format(INTF_NAME='3'),
                      format='xml',
                      target = 'running' )

print(c)

```

```
device_params={'name':'iosxe'}
```

```
edit_config
```

```
target = 'running'
```

```
conn_params={'name':'cisco_iosxe'}
```

```
send_cmds
```

```
dst = 'running-config'
```