

The Cisco Live! logo, featuring the word "CISCO" in a dark blue, sans-serif font, followed by "Live!" in a dark blue, script font.

CISCO *Live!*

The text "Let's go" in a large, dark blue, sans-serif font, positioned to the left of a bright, multi-colored sunburst graphic that radiates from the right side of the image.

Let's go

#CiscoLiveAPJC



The bridge to possible

Preparing for the DevNet Associate Certification

George Koukis, Leader Exam Program Manager, CCIEW#42079
Kareem Iskander, Lead Technical Advocate

BRKCRT-1148

CISCO *Live!*

#CiscoLiveAPJC

Agenda

- Introduction to Cisco Certifications
- What to expect on the exam and how to prepare.
- Exam vs Real life

Introduction to Cisco Certifications



“DevNet professional, DEVASC and Automation skills Required”



The image shows a newspaper clipping with a large, bold, black-bordered sign overlaid in the center. The sign has a thick red circle around it, with a diagonal line crossing from the top-left to the bottom-right. The text on the sign reads: **NOW HIRING**
The PERFECT
position for
YOU !!

The newspaper text is partially obscured by the sign. Visible text includes:

...time this meeting was
productive and has
brought major changes on
Earth. We will visit several
places of strategic interest
and will discuss possible
collaborations nationally.

...ssion for global warm-
and terrorism issues.
Among other things will
discuss new measures
global security. Last
time this meeting was very
productive and has brought
major changes on Earth. We
will visit several places of
strategic interest and will
discuss possible collabora-
tions nationally.

...Will also discuss new mea-
sures on global security
Last time this meeting was
very productive and has
brought major changes on
Earth. We will visit several
places of strategic interest
and will discuss possible
collaborations nationally.

...Discu... global warm
ing... orism issues
Ar... r things wil

Network skills or Developer skills

200-301 CCNA

1.0 Network Fundamentals

2.0 Network Access

3.0 IP Connectivity

4.0 IP Services

5.0 Security Fundamentals

6.0 Automation and Programmability

VS

200-901 DEVASC

1.0 Software Development and Design

2.0 Understanding and Using APIs

3.0 Cisco Platforms and Development

4.0 Application Deployment and Security

5.0 Infrastructure and Automation

6.0 Network Fundamentals

Cisco Certifications

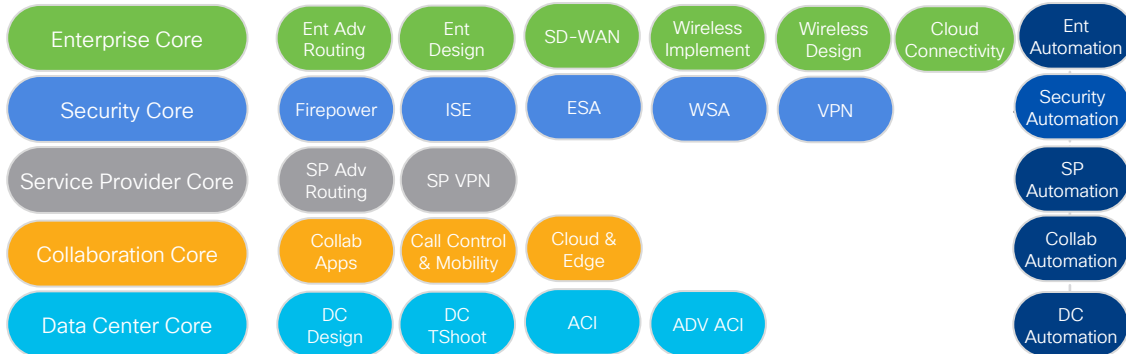


One Exam

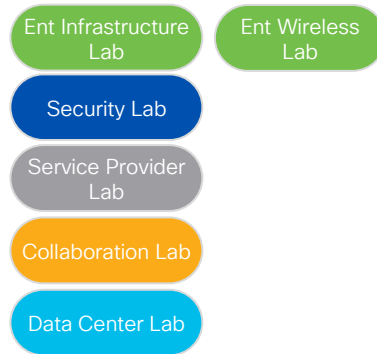
CCNA



Two Exams: Core + 1 Concentration



Core + Lab

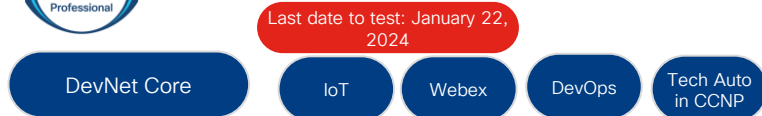


One Exam

DevNet Associate



Two Exams: Core + 1 Concentration



Core + Lab

DevNet Expert



One Exam

CyberOps Associate



Two Exams: Core + 1 Concentration

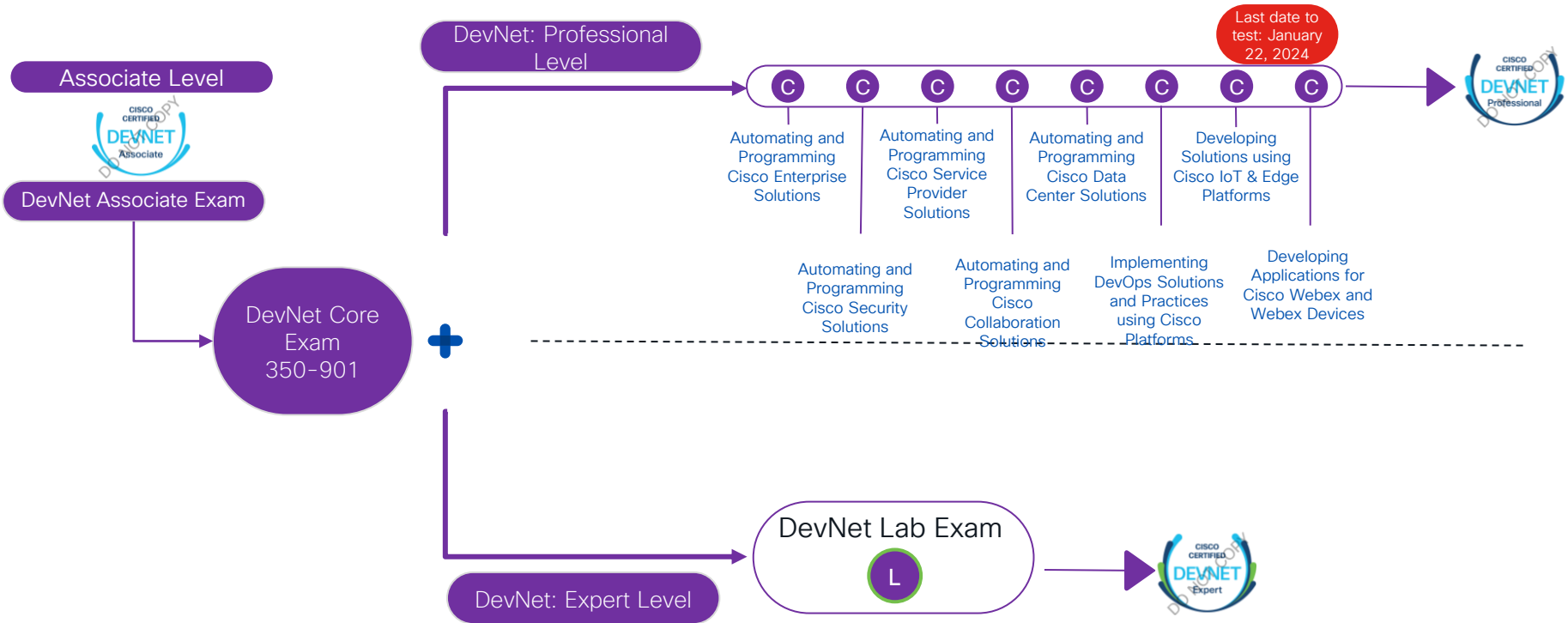


CyberOps Expert



CISCO *Live!*

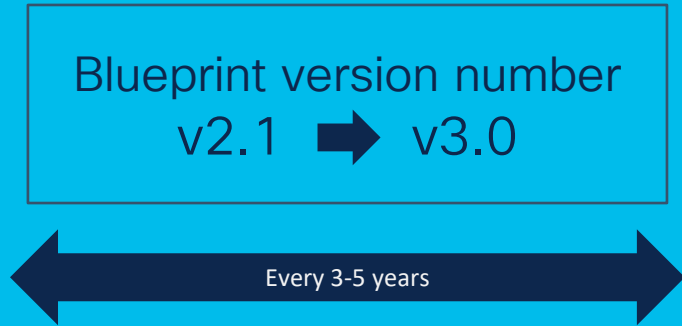
Cisco DevNet Certification Track



Revision Framework

Major Revision

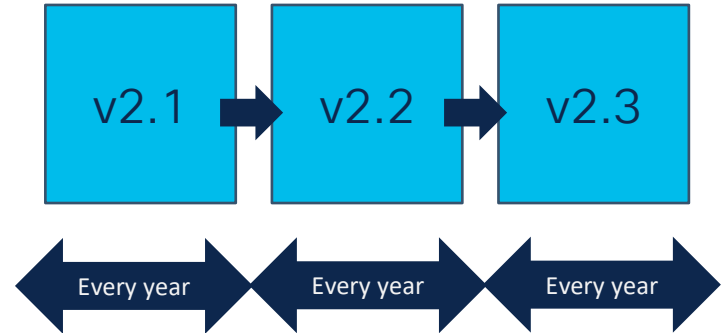
(Traditional Revision Model)



- Large revisions
- Major changes
- Steep learning curve
- Wider Alignment (Product & Technology)

Minor Revision

(Agile Revision Model)

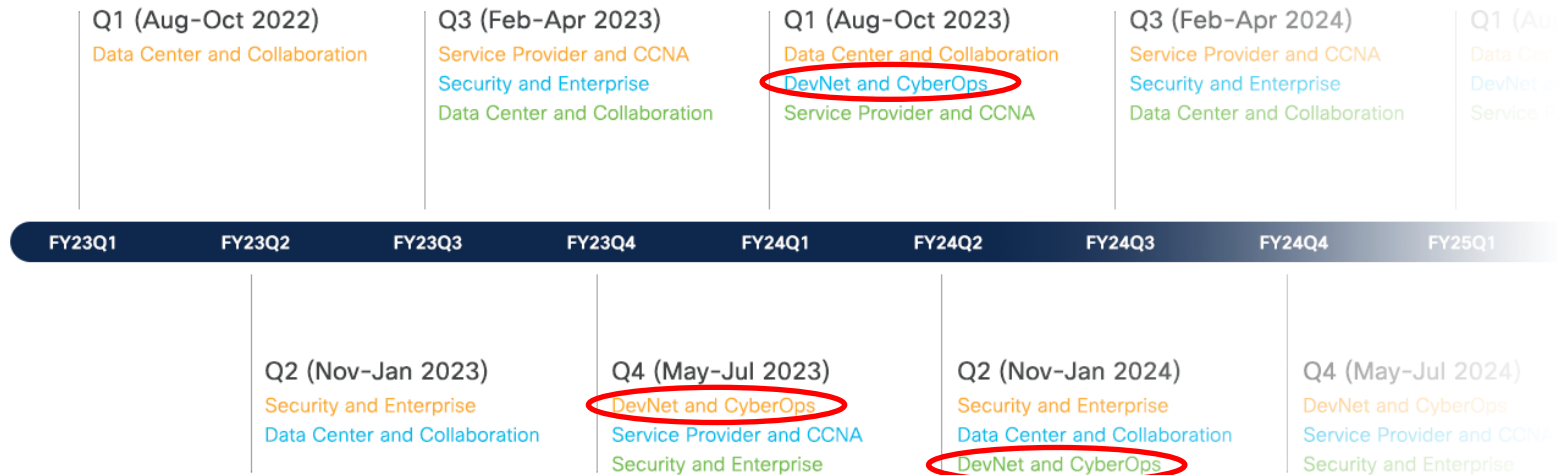


- Smaller modular revisions
- Incremental changes
- Easy bite-size learning model
- Frequent alignment (Product & Technology)

Certification Roadmap

How it works:

1. Cisco **reviews** each technology on the same quarterly schedule each year to make sure our exams align with the latest Cisco technologies.
2. We **announce** blueprint changes 3-6 months in advance along with revised exam topics and release notes.
3. We **publish** the updated exam 3-6 months after the exam blueprint publication.



Last date to test: January 22, 2024

DEVASC v1.0 VS DEVASC v1.1

3.3 Describe the capabilities of Cisco compute management platforms and APIs (UCS Manager, ~~UCS Director~~, and Intersight)

3.4 Describe the capabilities of Cisco collaboration platforms and APIs (Webex ~~Teams~~, Webex devices, Cisco Unified Communication Manager including AXL and UDS interfaces, and Finesse)

3.5 Describe the capabilities of Cisco security platforms and APIs (~~XDR~~, Firepower, Umbrella, ~~Secure endpointAMP~~, ISE, and Secure ~~Malware AnalyticsThreatGrid~~)

3.9.b Manage spaces, participants, and messages in Webex ~~Teams~~

5.3 Describe the use and roles of network simulation and test tools (such as ~~Cisco Modeling LabsVIRL~~ and pyATS)

5.6 Describe the capabilities of automation tools such as Ansible, ~~TerraformChef~~, and Cisco NSO

<https://learningnetwork.cisco.com/s/cisco-certification-roadmaps>

DevNet

CyberOps

Service Provider

CCNA

Security

Enterprise

Collaboration

Data Center

FAQs

Exam Number

Release Notes

Exam Topic Blueprint

Learning Matrix

DevNet Exams

Release Notes

DevNet v1.1
Learning Matrix

DevNet Associate Exam Updates

First date to test: January 22, 2024

200-901 DEVASC v1.1

Exam Topics

DevNet Professional Core Exam Updates

First date to test: January 22, 2024

350-901 DEVCOR v1.1

Exam Topics

DevNet Professional Concentration Exam Updates

What to expect on the exam

Exam Blueprint

<https://learningnetwork.cisco.com>

The Cisco Learning Network



Certifications ▾

Communities ▾

Webinars & Videos ▾

Study Resources ▾

[Certifications](#) / [DevNet Associate Certification and Training Program](#) / [200-901 DEVASC Exam Topics](#)

200-901 DEVASC Exam: DevNet Associate

Exam Description

The DevNet Associate Exam v1.0 (DEVASC 200-901) exam is a 120-minute exam associated with the Cisco Certified DevNet Associate certification. This exam tests a candidate's knowledge of software development and design including understanding and using APIs, Cisco platforms and development, application development and security, and infrastructure and automation. The course, *Developing Applications and Automating Workflows Using Cisco Core Platforms*, helps candidates to prepare for this exam.

The following topics are general guidelines for the content likely to be included on the exam. However, other related topics may also appear on any specific delivery of the exam. To better reflect the contents of the exam and for clarity purposes, the guidelines below may change at any time without notice.

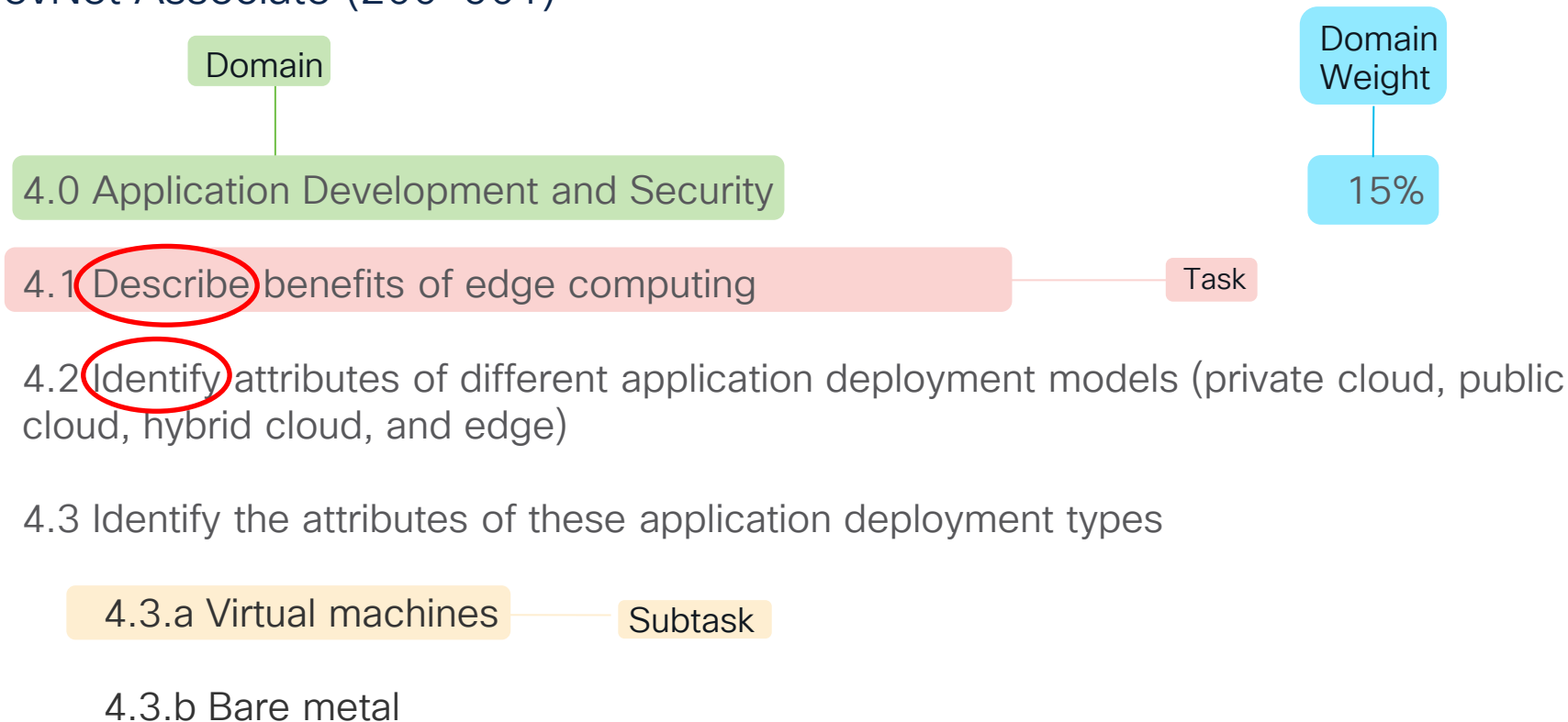
Download Complete List of Topics in PDF format

1.0 Software Development and Design	15%	▾
2.0 Understanding and Using APIs	20%	▾
3.0 Cisco Platforms and Development	15%	▾
4.0 Application Deployment and Security	15%	▾
5.0 Infrastructure and Automation	20%	▾
6.0 Network Fundamentals	15%	▾



Interpret the Blueprint:

DevNet Associate (200-901)



Blueprint Verbs

Describe/Explain

Compare

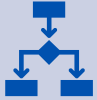
Construct/Utilise/Apply/Interpret

Troubleshoot/Identify

Depth of Knowledge



Types of questions



Multiple choice



Drag and drop



Fill in the blanks

Describe question

Example: Describe the functionality of these IP Services: DHCP, DNS, NAT, SNMP, NTP

Which protocol synchronizes the clock between computer systems?

- A. NTP
- B. NAT
- C. DNS
- D. SNMP

Describe question

Example: Describe the functionality of these IP Services: DHCP, DNS, NAT, SNMP, NTP

Which protocol synchronizes the clock between computer systems?

- A. NTP
- B. NAT
- C. DNS
- D. SNMP

Compare question

Example: Compare common API styles (REST and RPC)

What is the difference between REST and RPC APIs?

- A. REST APIs are stateless, and RPC APIs are stateful.
- B. REST APIs are vendor-specific, and RPC APIs are vendor-neutral.
- C. REST APIs are...
- D. REST APIs are...

OR like this

Drag and drop the characteristics from the left to the API style on the right.	
Options	Categories
stateful	Rest APIs
stateless	stateless
uses XML	uses XML
uses YAML	RPC APIs
	stateful
	uses YAML

Compare question

Example: Compare common API styles (REST and RPC)

What is the difference between REST and RPC APIs?

- A. REST APIs are stateless, and RPC APIs are stateful.
- B. REST APIs are vendor-specific, and RPC APIs are vendor-neutral.
- C. REST APIs are...
- D. REST APIs are...

OR like this

Drag and drop the characteristics from the left to the API style on the right.	
Options	Categories
stateful	Rest APIs
stateless	stateless
uses XML	uses XML
uses YAML	RPC APIs
	stateful
	uses YAML

How to prepare

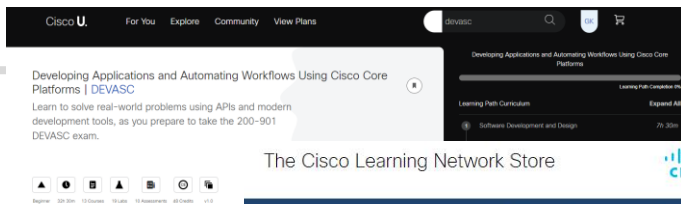
Describe/Explain

Compare

Construct/Utilize/Apply/Interpret

Troubleshoot/Identify

Depth of Knowledge



The Cisco Learning Network Store



Certification Training Technology Training All Training



DEVASC Preparation Bundle

Continuing Education Credits: 48



Access Duration: 180 days

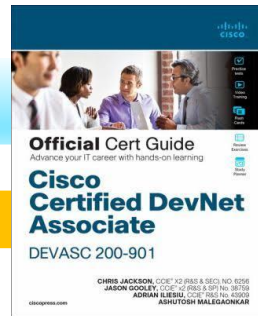
DevNet Associate Exam v1.0 (200-901)

DevNet Associate Exam v1.0 (DEVASC 200-901) is a 120-minute exam associated with the DevNet Associate - Developer Certification. This exam tests a candidate's knowledge of all the associate level in software development and design, understanding and using APIs, application development and security, and infrastructure and automation on Cisco platforms. The course, Developing Applications and Automating Workflows Using Cisco Core Platforms, helps candidates to prepare for this exam.

Sign up for updates

Exam overview

15% 1.0 Software Development and Design	
Exam Topics	Study Material
1.1 Compare data formats (XML, JSON, YAML)	These resources are meant to supplement your learning experience and exam preparation. They are NOT designed to serve as a complete self-study program, but intended only as a suggested starting point. Login to access these materials.
1.2 Describe parsing of common data format (XML, JSON, YAML) to Python data structures	<ul style="list-style-type: none">Setting up your Linux (Ubuntu) workstation as a development environmentSetting up your Windows workstation as a development environmentSetting up your macOS workstation as a development environment
1.3 Describe the concepts of test-driven development	
1.4 Compare software development methods (agile, lean, waterfall)	



Construct question

Usually a code snippet with missing parts.

Example: Construct a Python unit test

Drag and the drop the code on the snippet to complete the Python unit test. Not all options are used.

```
import   
import logging  
  
log = logging.getLogger(__name__)  
log.setLevel(logging.DEBUG)  
  
 uniTest(unittest.TestCase):  
    def setUp(self):  
        log.info('I am doing the setUp')  
  
    def test_one(self):  
        log.info('one')  
  
    @classmethod  
    def tearDownClass(self):  
        log.info('tearDown class')  
  
     test_two(self):  
        log.info('two')  
  
    @classmethod  
    def setUpClass(self):  
        log.info('set up Class')
```

class

unittest

log

def

Construct question

Usually a code snippet with missing parts.

Example: Construct a Python unit test

Drag and the drop the code on the snippet to complete the Python unit test. Not all options are used.

```
import 
import logging

log = logging.getLogger(__name__)
log.setLevel(logging.DEBUG)

 uniFest(unittest.TestCase):
    def setUp(self):
        log.info('I am doing the setUp')

    def test_one(self):
        log.info('one')

    @classmethod
    def tearDownClass(self):
        log.info('tear down class')

 test_two(self):
    log.info('two')

    @classmethod
    def setUpClass(self):
        log.info('set up Class')
```

class

unittest

log

def

class

How to prepare

Describe/Explain

Compare

Construct/Utilize/Apply/Interpret

Troubleshoot/Identify

Depth

The Cisco Learning Network Store



Certification Training Technology Training All Training

DevNet Associate

DEVASC Preparation Bundle

Continuing Education Credits: 48

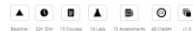


Access Duration: 180 days

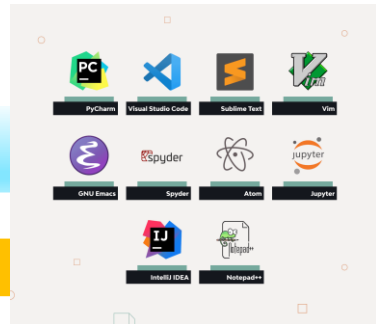
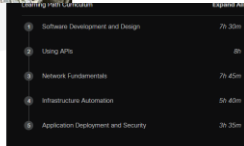
Cisco U. For You Explore

Developing Applications and Automating Platforms | DEVASC

Learn to solve real-world problems using APIs and modern development tools, as you prepare to take the 200-901 DEVASC exam.



DESCRIPTION
The DevOps Applications and Automation Workflows Utilize Cisco Core Platforms



Troubleshoot question

Something is broken and we need to fix it.

Example: Troubleshoot a problem given the HTTP response code, request and API documentation

```
import requests
import json
url = "https://192.168.1.10:443/restconf/data/ietf-interfaces:interfaces"
payload = json.dumps({
    "ietf-interfaces:interface": {"name": "Loopback200", "description":
    "External LOOPBACK200 for VTI Tunnels", "type": "iana-if-type:softwareLoopback",
    "enabled": True, "ietf-ip:ipv4": {"address": [{"ip": "84.16.101.150",
    "netmask": "255.255.255.255"}]}}})
headers = {
    'Authorization': 'Basic e3t1c2VybmFtZX19Ont7cGFzc3dvcmR9fQ==',
    'Accept': 'application/yang-data+json',
    'Content-Type': 'application/yang-data+json'}
response = requests.request("POST", url, headers=headers, data=payload)
print(r.status_code)
print(response.text)
```

Refer to the exhibit. A network engineer has developed a script to automate the provision of newly added devices on the network. In the lab environment where the script was tested, it was running with no issues. When applied to the customer side, it returned a **404 error**. What needs to be changed on the script to fix the error?

- A. The IP address on the script must be set to the customer's range.
- B. The device that needs to be provisioned must connect to the network.
- C. The script must be written in Yang instead of JSON for the devices to listen.
- D. The server running the script in a different range than the device to provision.

Troubleshoot question

Something is broken and we need to fix it.

Example: Troubleshoot a problem given the HTTP response code, request and API documentation

```
import requests
import json
url = "https://192.168.1.10:443/restconf/data/ietf-interfaces:interfaces"
payload = json.dumps({
    "ietf-interfaces:interface": {"name": "Loopback200", "description":
        "External LOOPBACK200 for VTI Tunnels", "type": "iana-if-type:softwareLoopback",
        "enabled": True, "ietf-ip:ipv4": {"address": [{"ip": "84.16.101.150",
            "netmask": "255.255.255.255"}]}}}
headers = {
    'Authorization': 'Basic e3t1c2VybmFtZX19Ont7cGFzc3dvcmR9fQ==',
    'Accept': 'application/yang-data+json',
    'Content-Type': 'application/yang-data+json'}
response = requests.request("POST", url, headers=headers, data=payload)
print(r.status_code)
print(response.text)
```

Refer to the exhibit. A network engineer has developed a script to automate the provision of newly added devices on the network. In the lab environment where the script was tested, it was running with no issues. When applied to the customer side, it returned a **404 error**. What needs to be changed on the script to fix the error?

- A. The IP address on the script must be set to the customer's range.
- B. The device that needs to be provisioned must connect to the network.
- C. The script must be written in Yang instead of JSON for the devices to listen.
- D. The server running the script in a different range than the device to provision.

How to prepare

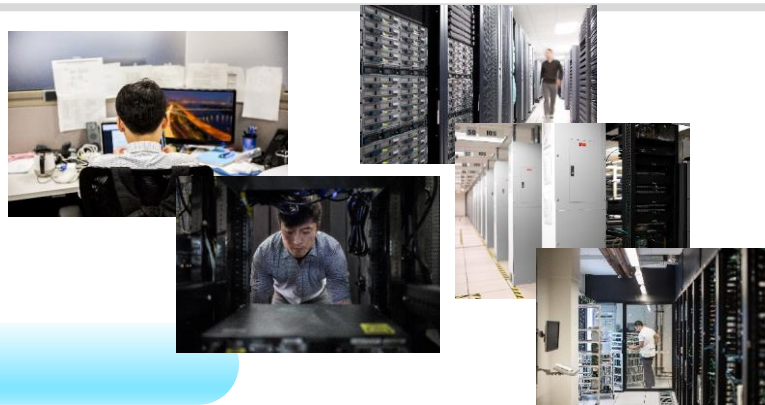
Describe/Explain

Compare

Construct/Utilize/Apply/Interpret

Troubleshoot/Identify

Depth of Knowledge



Exam vs Real life

What will be covered

- 1.8 Utilize common version control operations with Git
- 2.9 Construct a Python script that calls a REST API using the requests library
- 3.9 Construct code to perform a specific operation based on a set of requirements and given API reference documentation such as these:
 - 3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Center, ACI, Cisco SD-WAN, or NSO
- 4.11 Utilize Bash commands (file management, directory navigation, and environmental variables)



Sample Code:

- Scan QR code

or

- <https://github.com/qsnyder/brkcrt-2080>

What will be covered

- 1.8 Utilize common version control operations with Git
- 2.9 Construct a Python script that calls a REST API using the requests library
- 3.9 Construct code to perform a specific operation based on a set of requirements and given API reference documentation such as these:
 - 3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Center, ACI, Cisco SD-WAN, or NSO
- 4.11 Utilize Bash commands (file management, directory navigation, and environmental variables)

```
git  ssh://user@example.com/path/to/application.git
```

Utilize question

1.8 Utilize common version control operations with Git

1.8.a Clone

Refer to the exhibit. An engineer is working on a new feature for an existing application. The application is already live and the development must take place locally on the engineer's machine. Which Git command must the engineer use to create a copy of the application on the local machine?

- A. clone
- B. copy
- C. pull
- D. add

```
git [ ] ssh://user@example.com/path/to/application.git
```

Utilize question

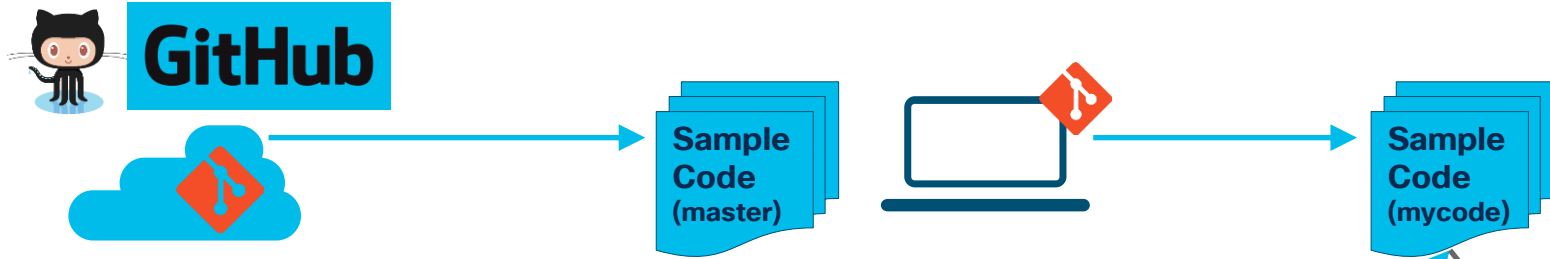
1.8 Utilize common version control operations with Git

1.8.a Clone

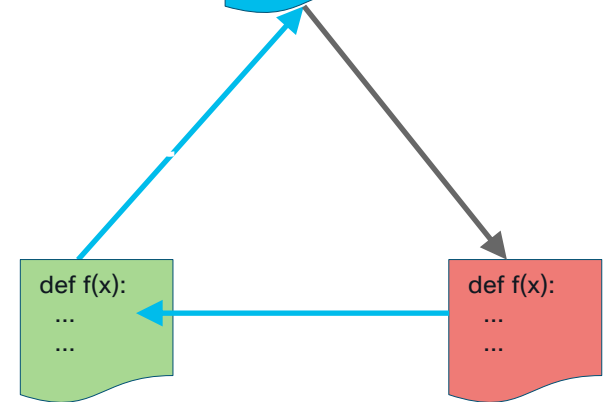
Refer to the exhibit. An engineer is working on a new feature for an existing application. The application is already live and the development must take place locally on the engineer's machine. Which Git command must the engineer use to create a copy of the application on the local machine?

- A. clone
- B. copy
- C. pull
- D. add

DevNet Sample-Code Workflow



Step	Action	Git Command
1.	Clone the Remote Repository	<code>git clone url</code>
2.	Create and Checkout a Local Branch	<code>git checkout -b new-branch-name</code>
3.	Incrementally Commit Changes	<code>git add filename</code> <code>git commit -m "Commit message"</code>



Git Hands-on Walkthrough



What will be covered

- 1.8 Utilize common version control operations with Git
- **2.9 Construct a Python script that calls a REST API using the requests library**
- 3.9 Construct code to perform a specific operation based on a set of requirements and given API reference documentation such as these:
 - 3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Center, ACI, Cisco SD-WAN, or NSO
- 4.11 Utilize Bash commands (file management, directory navigation, and environmental variables)

Construct question

2.9 Construct a Python script that calls a REST API using the requests library

Drag and drop the code from the bottom onto the box where the code is missing to retrieve a list of organizations configured in the Meraki organization. Not all options are used.

```
import   
MERAKI_API_KEY = '6bec40cf957de430a6f1f2baa056b99a4fac9ea0'  
base_url = "https://api.meraki.com/api/v1"  
endpoint = "/organizations"  
header = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}  
orgs = requests. (url=base_url+endpoint, headers=header)  
 = json.loads(orgs.text)  
pprint(orgs)
```

orgs

get

patch

requests

org

json

Construct question

2.9 Construct a Python script that calls a REST API using the requests library

Drag and drop the code from the bottom onto the box where the code is missing to retrieve a list of organizations configured in the Meraki organization. Not all options are used.

```
import   
MERAKI_API_KEY = '6bec40cf957de430a6f1f2baa056b99a4fac9ea0'  
base_url = "https://api.meraki.com/api/v1"  
endpoint = "/organizations"  
header = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}  
orgs = requests.(url=base_url+endpoint, headers=header)  
 = json.loads(orgs.text)  
pprint(orgs)
```

orgs

get

patch

requests

org

json

Demo Overview – Python & REST APIs

```
1 import requests
2 import json
3 from pprint import pprint
4 # DevNet Sandbox Information
5 from requests import Response
6 '''
7 Username: devnetmeraki@cisco.com
8 Password: Admin123!
9 You can also use this API key for the Dashboard API: 6bec40cf957de430a6f1f2baa056b99a4fac9ea0
10 '''
11 MERAKI_API_KEY = '6bec40cf957de430a6f1f2baa056b99a4fac9ea0'
12 # MERAKI BASE URL
13 base_url = "https://api.meraki.com/api/v1"
14 # MERAKI Endpoint
15 endpoint = "/organizations"
16 # requests header information
17 header = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}
18 # Make request
19 orgs: Response = requests.get(url=base_url+endpoint, headers=header)
20 orgs = json.loads(orgs.text)
21 pprint(orgs)
22
23
24
```

Let's Test it!

- **Requests** is an elegant and simple HTTP **library** for **Python**, built for human beings
- Define requests building blocks
- Make API call to service
- Capture returned data and doing something with it

What will be covered

- 1.8 Utilize common version control operations with Git
- 2.9 Construct a Python script that calls a REST API using the requests library
- 3.9 Construct code to perform a specific operation based on a set of requirements and given API reference documentation such as these:
 - 3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Center, ACI, Cisco SD-WAN, or NSO
- 4.1.1 Utilize Bash commands (file management, directory navigation, and environmental variables)

Construct question

3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Centre, ACI, Cisco SD-WAN, or NSO SDWAN

Drag and drop the code from the bottom onto the box where the code is missing to retrieve a list of devices in the Meraki organization. Not all options are used.

```
import requests
base_url = "https://api.meraki.com/api/v1"
[ ] = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}
def get_device_list():
    for org in orgs:
        endpoint = "/organizations/{}/devices".format(org['id'])
        print([ ])
        resp = requests.get(url=base_url + endpoint, headers=header)
        [ ] = json.loads(resp.text)
        pprint(device)
if __name__ == "__main__":
    get_device_list()
```

header

device

auth

endpoint

orgs

list

Construct question

3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Centre, ACI, Cisco SD-WAN, or NSO SDWAN

Drag and drop the code from the bottom onto the box where the code is missing to retrieve a list of devices in the Meraki organization. Not all

options are used

```
import requests
base_url = "https://api.meraki.com/api/v1"
[ ] = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}
def get_device_list():
    for org in orgs:
        endpoint = "/organizations/{}/devices".format(org['id'])
        print([ ])
        resp = requests.get(url=base_url + endpoint, headers=header)
        [ ] = json.loads(resp.text)
        pprint(device)
if __name__ == "__main__":
    get_device_list()
```

header

device

auth

endpoint

orgs

list

Demo Overview – Meraki Network Devices

```
1 import requests
2 import json
3 from pprint import pprint
4
5 # DevNet Sandbox Information
6 '''
7 Username: devnetmeraki@cisco.com
8 Password: Adm!n123!
9 You can also use this API key for the Dashboard API: 6bec40cf957de430a6f1f2baa056b99a4fac9ea0
10 '''
11
12 MERAKI_API_KEY = '6bec40cf957de430a6f1f2baa056b99a4fac9ea0'
13 # MERAKI BASE URL
14 base_url = "https://api.meraki.com/api/v1"
15 # requests header information
16 header = {"X-Cisco-Meraki-API-Key": MERAKI_API_KEY}
17
18 def get_org_list():
19     endpoint = "/organizations"
20     # Make request
21     resp = requests.get(url=base_url+endpoint, headers=header)
22     orgs = json.loads(resp.text)
23     print(orgs)
24     return orgs
25
26 def get_device_list():
27     for org in orgs:
28         # MERAKI Endpoint
29         endpoint = "/organizations/{}/devices".format(org['id'])
30         print(endpoint)
31         resp = requests.get(url=base_url + endpoint, headers=header)
32         device = json.loads(resp.text)
33         pprint(device)
34
35 if __name__ == "__main__":
36     orgs = get_org_list()
37     get_device_list()
```

Let's Test it!

- Meraki APIs and Python Requests
- Authenticate
- Meraki Hierarchical order
 - Orgs contain Networks
 - Networks contain Devices
- Get list of orgs
- loop through each network in orgs and extract devices

What will be covered

- 1.8 Utilize common version control operations with Git
- 2.9 Construct a Python script that calls a REST API using the requests library
- 3.9 Construct code to perform a specific operation based on a set of requirements and given API reference documentation such as these:
 - 3.9.a Obtain a list of network devices by using Meraki, Cisco DNA Center, ACI, Cisco SD-WAN, or NSO
- **4.11 Utilize Bash commands (file management, directory navigation, and environmental variables)**

Utilize question

4.11 Utilize Bash commands

```
$virtualenv -p python3 <desired-path>  
$source <desired-path>/bin/
```

Refer to the exhibit. An engineer needs to complete the creation and enable a Python virtual environment. Which code needs to be placed on the box where the code is missing?

- A. activate
- B. enable
- C. allow
- D. power

Utilize question

4.11 Utilize Bash commands

```
$virtualenv -p python3 <desired-path>  
$source <desired-path>/bin/
```

Refer to the exhibit. An engineer needs to complete the creation and enable a Python virtual environment. Which code needs to be placed on the box where the code is missing?

- A. activate
- B. enable
- C. allow
- D. power

Virtual Environment

- You can code in Python without using virtual environments.
- The minute you update modules or Python itself, you run the risk of breaking your apps
- A virtual environment allows you to lock in the components and modules you use for your app into a single “package.”

Enabling Python VirtualEnv

- Directory Structure
- Usually associated with a Project
- An isolated environment for installing and working with Python Packages

```
$ python3 -m venv myvenv
```

```
$ source myvenv/bin/activate
```

source environment-name/bin/activate

- ✓ The activation script will modify your prompt.
- ✓ Inside a virtual environment, your interpreter will always be **`python``**.

CISCO *Live!*

Did you know?

You can have a
one-on-one session with
a technical expert!

Visit Meet the Expert in The HUB
to meet, greet, whiteboard & gain
insights about your unique questions
with the best of the best.



Meet the Expert Opening Hours:

Tuesday	3:00pm – 7:00pm
Wednesday	11:15am – 7:00pm
Thursday	9:30am – 4:00pm
Friday	10:30am – 1:30pm

Session Surveys

We would love to know your feedback on this session!

- Complete a minimum of four session surveys and the overall event surveys to claim a Cisco Live T-Shirt



Continue your education



- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Expert meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand



The bridge to possible

Thank you

CISCO *Live!*

#CiscoLiveAPJC

The background features a vibrant, multi-colored abstract design. On the left, there are horizontal, wavy bands of color in shades of red, orange, yellow, and green. On the right, a bright white light source emits a series of colorful rays in shades of blue, green, and yellow, creating a sunburst effect.

cisco *Live!*

Let's go

#CiscoLiveAPJC