

300-920^{Q&As}

Developing Applications for Cisco Webex and Webex Devices
(DEVWBX)

Pass Cisco 300-920 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.leads4pass.com/300-920.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Cisco
Official Exam Center

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers



QUESTION 1

DRAG DROP

Drag and drop the code to complete the JavaScript snippet so that it:

1. retrieves the details of an individual user
2. checks what licenses they have already
3. updates their account with a new license Options can be used more than once.

Select and Place:

```
const request = require ('require');
var adminToken = 'VALID_ADMIN_TOKEN';
var personId = "VALID_PERSON_ID";
var licenseId = "VALID_LICENSE_ID";
function buildOptions(sendURL) {
    return (
        url:sendURL,
        headers: {
            'Content-Type': 'application/json',
            'Authorization': 'Bearer $ (adminToken)'}
        )
    }
}
var peopleOptions = buildOptions('https://api.ciscopark.com/v1/ [ ] /${personId}');
request.get(peopleOptions, function(error, response) {
    var json = JSON.parse(response.body);

    if(json.[ ] .indexOf( [ ] ) < 0) {
        json.[ ] .push( [ ] ) :
        peopleOptions['body'] = JSON.stringify(json) :
        request.put(peopleOptions, function(error, response) {
            var json = JSON.parse(response.body) ;
            console.log (json) ;
        }) ;
    }
}) ;
```

licenseId	person
people	licenses

Correct Answer:

```
const request = require ('require');
var adminToken = 'VALID_ADMIN_TOKEN';
var personId = "VALID_PERSON_ID";
var licenseId = "VALID_LICENSE_ID";
function buildOptions(sendURL) {
    return (
        url:sendURL,
        headers: {
            'Content-Type': 'application/json',
            'Authorization': 'Bearer $ (adminToken)'}
        )
    }
}
var peopleOptions = buildOptions('https://api.ciscospark.com/v1/ people /${personId}');
/${personId}');
request.get(peopleOptions, function(error, response) {
    var json = JSON.parse(response.body);

    if(json. licenses .indexOf( licenseId ) < 0) {
    json. people .push( person ) :
    peopleOptions['body'] = JSON.stringify(json) :
    request.put(peopleOptions, function(error, response) {
        var json = JSON.parse(response.body) ;
        console.log (json) ;
        }) ;
    }
}) ;
```

QUESTION 2

DRAG DROP

Drag and drop the expressions to create a Webex Teams widget that uses Guest Issuer to enable customers to chat and meet with agents. Not all options are used.

Select and Place:

```

<script>
$(document).ready(function () {
  $("#btn").click(function () {
    var widgetEL = document.getElementById('supportChart');
    $.ajax({
      url: '/api/getSession',
      type: 'POST',
      contentType: 'application/json',
      data: JSON.stringify({
        name: $("#name").val(),
        email: $("#email").val()
      })
    }).done(function (session) {
      ciscopark.widget(widgetEL).spaceWidget({
        guestToken: ,
        destinationId: ,
        destinationType: ,
        activities: {"files": false, "meet": true, "message": true, "people": false},
        initialActivity: ,
        secondaryActivitiesFullWidth: false
      });
    })
  })
})
</script>

```

"userId"
"message"
session.token
Direct
session.agent
guest

```

app.post('/api/getSession', function(req, res) {
  res.status(200).send({
    agent: 'johndoe@example.com',
    token: jwt.sign({
      "sub": "${req.body.email}",
      "name": "${req.body.name}",
      "iss": "${issuer}"
    }, Buffer.from("${secret}", 'base64'), {})
  });
})

```

Correct Answer:

```
<script>
$(document).ready(function () {
  $("#btn").click(function () {
    var widgetEL = document.getElementById('supportChart');
    $.ajax({
      url: '/api/getSession',
      type: 'POST',
      contentType: 'application/json',
      data: JSON.stringify({
        name: $("#name").val(),
        email: $("#email").val()
      })
    }).done(function (session) {
      ciscopark.widget(widgetEL).spaceWidget({
        guestToken: 
        destinationId: 
        destinationType: 
        activities: {"files": false, "meet": true, "message": true, "people": false},
        initialActivity: 
        secondaryActivitiesFullWidth: false
      });
    })
  })
})
</script>

app.post('/api/getSession', function(req, res) {
  res.status(200).send({
    agent: 'johndoe@example.com',
    token: jwt.sign({
      "sub": "${req.body.email}",
      "name": "${req.body.name}",
      "iss": "${issuer}"
    }, Buffer.from("${secret}", 'base64'), {})
  });
})
```

QUESTION 3

DRAG DROP

Drag and drop the code snippets onto the exhibit to create a valid Webex Meetings API request allowing Jane (an admin) to reset John's PMR pin. Not all options are used.

Select and Place:

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message xmlns:serv= "http://www.webex.com/schemas/2002/06/service »
  xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>apidemoeu</siteName>

      <webExID> [redacted] </webExID>
      [redacted]
    </securityContext>
  </header>
  <body>
    <bodyContent xsi :type= "java:com.webex.xmlapi.service.binding.user.
      [redacted] ">
      <webExID> [redacted] </webExID>
      <personalMeetingRoom>
        <hostPIN>1337</hostPIN>
      </personalMeetingRoom>
    </bodyContent>
  </body>
</serv:message>
```

-
-
-
-
-
-

Correct Answer:

```
<?xml version="1.0" encoding="UTF-8"?>
<serv:message xmlns:serv= "http://www.webex.com/schemas/2002/06/service »
  xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance">
  <header>
    <securityContext>
      <siteName>apidemoeu</siteName>

      <webExID> johndoe </webExID>
      SetUser
    </securityContext>
  </header>
  <body>
    <bodyContent xsi :type= "java:com.webex.xmlapi.service.binding.user.
      <token>AAABb5HuHWUAABUY</token> ">
      <webExID> UpdateUser </webExID>
      <personalMeetingRoom>
        <hostPIN>1337</hostPIN>
      </personalMeetingRoom>
    </bodyContent>
  </body>
</serv:message>
```

janedoe

<sessionTicket>AAABb5HuHWUAABUY</sessionTicket>

QUESTION 4

```
1 const request = require('request-promise').defaults({
2   json: true,
3   baseUrl: 'https://api.ciscospark.com/v1/'
4 })
5
6 const dontsay = ['swag', 'fleek', 'stay', 'shade', 'bae', 'yas', 'lit', 'ratchet']
7
8 function isSensitive(text) {
9   sensitive = false
10  for (j = 0 ; j < dontsay.length ; j++) {
11    if (text && text.includes(dontsay[j])) {
12      sensitive = true
13    }
14  }
15  return sensitive
16 }
17
18 function getMessages() {
19   request.get({
20     url: ".....",
21     headers: {Authorization: 'Bearer ${compliance_token}' }
22   }).then(events => {
23     for (j = 0; i < events.items.length; i++) {
24       if (events.items[i].data.text && isSensitive(events.items[i].data.text)) {
25         request.post({
26           uri: 'messages',
27           headers: {Authorization: 'Bearer ${bot_token}'},
28           body: {
29             'toPersonId' : '${events.items[i].actorId}',
30             'markdown': 'Please don't say **${events.items[i].data.text}** ever again..'
31           },
32         })
33       }
34     }
35   })
36 }
37 getMessages()
38
39
40
```

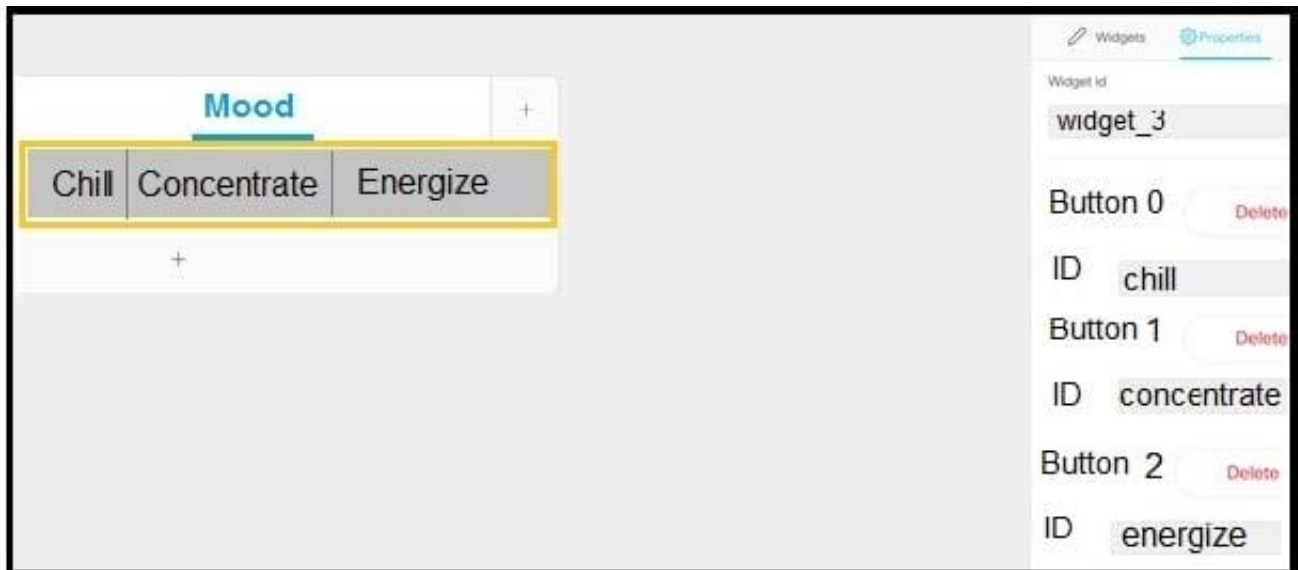
Refer to the exhibit. A company uses Webex Teams extensively for communications involving customers, and want to enforce a consistent messaging policy. Which code completes line 20 to send a notification when noncompliant messages are detected?

- A. events?resource=people
- B. compliance?resource=messages
- C. events?resource=messages
- D. compliance?resource=people

Correct Answer: C

The function is getmessages. Then a request is made so the get comes in action and does events?Resource=messages to get the data for the events.

QUESTION 5



```
function lights(mood) {
  if (mood === 'chill'){
    Hue({"on" : true, "bri": 254, "hue": 7676, "sat": 199});
  }
  else {
    Hue({"on": false, "bri": 255, "hue": 255, "sat": 255});
  }
}
xapi.on('ready', () => {
  xapi.event
    .on('UserInterface Extensions Widget Action', (x) => {
      if (x.WidgetId === 'widget_3' && x.Type === 'released'){
        lights(x)
      }
    })
});
```

Refer to the exhibit. Certain lighting conditions are needed when hosting Webex meetings for a particular department in a company. A split button that integrates with the lighting controls is added to the Touch 10. However, when the user selects Chill, the lights turn off completely. Which code change resolves this issue?

- A. Change `xapi.event` to `xapi.httpfeedback`.
- B. Set `x.Type` to `'pressed'` instead of `'released'`.
- C. Set `x.Type` to `'clicked'` instead of `'released'`.
- D. Pass `x.Value` instead of `x` to the `lights` function.

Correct Answer: B

Reference: <https://www.cisco.com/c/dam/en/us/td/docs/telepresence/endpoint/ce81/sx-mx-in-room-control-guide->

ce81.pdf

[300-920 PDF Dumps](#)

[300-920 VCE Dumps](#)

[300-920 Braindumps](#)