

# Free PDF Quiz Salesforce - Salesforce-MuleSoft-Developer-I High Hit-Rate Reliable Practice Questions



P.S. Free 2025 Salesforce Salesforce-MuleSoft-Developer-I dumps are available on Google Drive shared by Prep4sures: <https://drive.google.com/open?id=1ozO7eSxRAeLr4ClhrHmrQHYuMxU4Zli>

Our Salesforce-MuleSoft-Developer-I learning guide beckons exam candidates around the world with our attractive characters. Our experts made significant contribution to their excellence. So we can say bluntly that our Salesforce-MuleSoft-Developer-Isimulating exam is the best. Our effort in building the content of our Salesforce-MuleSoft-Developer-I Study Materials lead to the development of learning guide and strengthen their perfection. You may find that there are always the latest information in our Salesforce-MuleSoft-Developer-I practice engine and the content is very accurate.

Our Salesforce-MuleSoft-Developer-I exam questions will be the easiest access to success without accident for you. Besides, we are punctually meeting commitments to offer help on Salesforce-MuleSoft-Developer-I study materials. So there is no doubt any information you provide will be treated as strictly serious and spare you from any loss of personal loss. There are so many success examples by choosing our Salesforce-MuleSoft-Developer-I Guide quiz, so we believe you can be one of them.

>> Reliable Salesforce-MuleSoft-Developer-I Practice Questions <<

## Salesforce-MuleSoft-Developer-I Latest Study Guide, Salesforce-MuleSoft-Developer-I Study Plan

Salesforce-MuleSoft-Developer-I practice dumps offers you more than 99% pass guarantee, which means that if you study our Salesforce-MuleSoft-Developer-I learning guide by heart and take our suggestion into consideration, you will absolutely get the certificate and achieve your goal. Meanwhile, if you want to keep studying this course , you can still enjoy the well-rounded services by Salesforce-MuleSoft-Developer-I Test Prep, our after-sale services can update your existing Salesforce-MuleSoft-Developer-I study quiz within a year and a discount more than one year.

## Salesforce Salesforce-MuleSoft-Developer-I Exam Syllabus Topics:

Topic	Details
Topic 1	<ul style="list-style-type: none"><li>Designing APIs: Designing APIs involves describing the lifecycle of the modern API and using RAML to define various aspects of an API. It includes identifying when to use query parameters vs URI parameters, and defining API parameters.</li></ul>
Topic 2	<ul style="list-style-type: none"><li>Using Connectors: It focuses on retrieving data from REST services using HTTP Request or REST Connector. Moreover, the topic covers using a Web Service Consumer connector for SOAP web services and the Transform Message component.</li></ul>

Topic 3	<ul style="list-style-type: none"> <li>Debugging and Troubleshooting Mule Applications: Using breakpoints to inspect a Mule event during runtime, installing missing Maven dependencies, and reading and deciphering Mule log error messages are sub-topics of this topic.</li> </ul>
Topic 4	<ul style="list-style-type: none"> <li>Creating Application Networks: The topic of creating Application Networks encompasses understanding MuleSoft's proposal for closing the IT delivery gap and describing the role and characteristics of the modern API. It also includes the purpose and roles of a Center for Enablement (C4E), and the benefits of API-led.</li> </ul>
Topic 5	<ul style="list-style-type: none"> <li>Structuring Mule Applications: Structuring Mule applications covers parameterizing an application and defining and reusing global configurations. It includes breaking an application into multiple flows using private flows, subflows, and the Flow Reference component.</li> </ul>
Topic 6	<ul style="list-style-type: none"> <li>Routing Events: It focuses on using the Choice router for conditional logic and the Scatter-Gather router to multicast events. This topic also involves validating data by using the Validation module.</li> </ul>
Topic 7	<ul style="list-style-type: none"> <li>Transforming Data with DataWeave: It involves writing DataWeave scripts and using DataWeave functions. This topic also includes defining and using DataWeave variables, functions, and modules, and applying correct syntax.</li> </ul>
Topic 8	<ul style="list-style-type: none"> <li>Handling Errors: Handling errors includes describing default error handling in Mule applications and defining custom global default error handlers. It involves comparing On Error Continue and On Error Propagate scopes, creating error handlers for a flow, using the Try scope, and mapping errors to custom application errors.</li> </ul>
Topic 9	<ul style="list-style-type: none"> <li>Building API Implementation Interfaces: This topic involves manually creating a RESTful interface for a Mule application and generating a REST Connector from a RAML specification. It also includes describing the features and benefits of APIkit.</li> </ul>
Topic 10	<ul style="list-style-type: none"> <li>Deploying and Managing APIs and Integrations: It includes packaging Mule applications for deployment and deploying them to CloudHub. This topic also involves using CloudHub properties, creating and deploying API proxies, connecting an API implementation to API Manager, and applying policies to secure an API.</li> </ul>

## **Salesforce Certified MuleSoft Developer (Mule-Dev-201) Sample Questions (Q16-Q21):**

### **NEW QUESTION # 16**

Refer to the exhibits.

```

#%RAML 1.0
title: ACME Airlines
version: 1.0

/flights:
  get:
    responses:
      200:
      404:

/airline:
  get:
    queryParameters:
      code: string
    responses:
      200:
      404:

/accounts:
  get:
    responses:
      200:
      404:
  post:
    responses:
      201:

```

salesforce

How many private flows does APIKit generate from RAML specification?

- A. 0
- B. 1
- C. 2
- D. 3

**Answer: C**

Explanation:

APIKit Creates a separate flow for each HTTP method. Hence 4 private flows would be generated.  
 MuleSoft Documentation Reference: <https://docs.mulesoft.com/mule-runtime/4.3/build-application-from-api>

## NEW QUESTION # 17

Refer to the exhibits.

The diagram illustrates a Mule flow named 'main'. It starts with an 'HTTP: POST /order/items' Listener. The payload is then processed by a 'Shipping Address' connector, which has a 'Set shippingAddress target' step. Finally, a 'Set Payload' connector merges the order items with the shipping address details. The configuration code below the diagram shows the XML for this flow.

```
<flow name="main" >
    <http:listener doc:name="HTTP: POST /order/items" config-ref="HTTP Listener config" path="/order/items">
        <http:response>
            <http:body><![CDATA[#{output application/json --- payload}]]></http:body>
        </http:response>
    </http:listener>
    <shipping:shipping-address doc:name="Set shippingAddress target" config-ref="Shipping_Config" target="shippingAddress">
        <shipping:shipping-address-request-data><![CDATA[#{payload.user}]]></shipping:shipping-address-request-data>
    </shipping:shipping-address>
    <set-payload value="#{output application/json" />

```

```
{
    items: payload.items,
    shippingInfo: payload.shippingAddress
}]" doc:name="Merge order items with shipping address details" />
</flow>
</flow>
```

A Mule application is being developed to process web client POST requests with payloads containing order information including the user name and purchased items. The Shipping connector returns a shipping address for the input payloads user name. The Shipping connector's Shipping Address operation is configured with a target named shippingAddress.

The Set Payload transformer needs to set an item key equal to the items value from the original received payload and a shippingInfo key equal to the the ShippingAddress operation's response. What is a straightforward way to property configure the Set Payload transformer with the required data?

- A. 

```
{
    items: payload.items,
    shippingInfo: shippingAddress
}
```
- B. 

```
{
    items: payload.items,
    shippingInfo: vars.shippingAddress
}
```
- C. 

```
{
    items: vars.shippingAddress.items
    shippingInfo: payload
}
```
- D. 

```
{
    items: attributes.shippingAddress.items
    shippingInfo: payload
}
```

**Answer: B**

## NEW QUESTION # 18

What is the correct way to format the decimal 200.1234 as a string to two decimal places?

- A. 200.1234 as string {format: ".0#"}  
• B. 200.1234 as String as format: ".0#"  
• C. 200.1234 as string as format: ".0#"  
• D. 200.1234 as String {format: ".0#"} (highlighted in red)

**Answer: D**

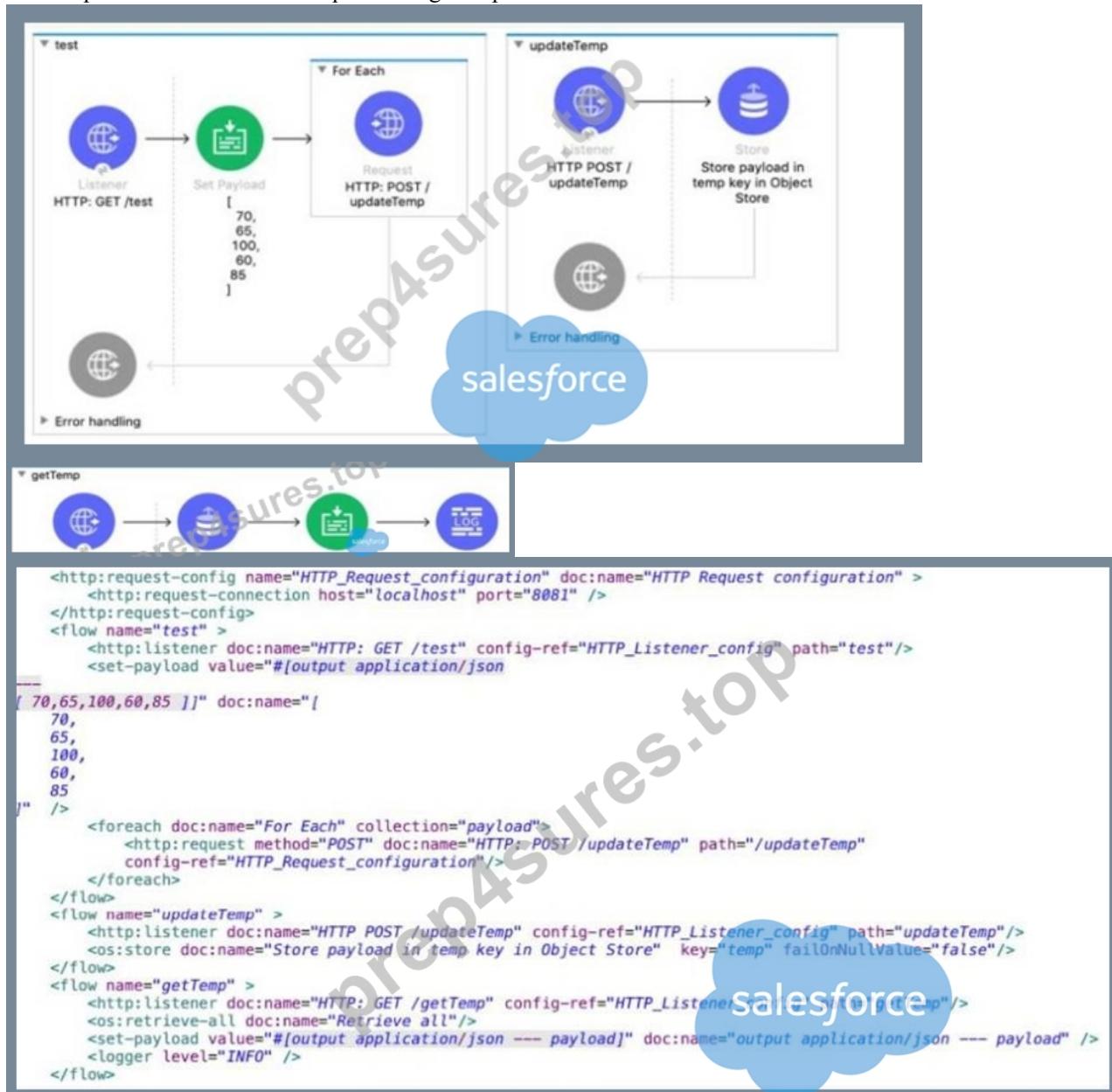
Explanation:

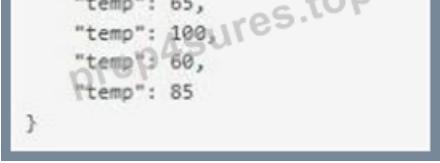
Correct answer is 200.1234 as String {format: ".0#"} . Rest all options are not syntactically correct.

## NEW QUESTION # 19

A web client sends one GET request to the test flow's HTTP Listener, which causes the test flow to call the updateTemp flow. After the test flow returns a response, the web client then sends a different GET request to the getTemp flow's HTTP Listener. The test flow is not called a second time.

What response is returned from the request to the getTemp flow's HTTP Listener?



- A. 
- B. 
- C. 
- D. 

**Answer: A**

#### NEW QUESTION # 20

A shopping API contains a method to look up store details by department.

To get the information for a particular store, web clients will submit requests with a query parameter named department and uri parameter named storeId. What is valid RAML snippet that supports requests from a web client to get a data for a specific storeId and department name?

- A. 1.get:2.queryParameter:3.department:4.uriParameter:5.{storeId}:
- B. 1./{storeId} 2.get:3.queryParameter:4.department:
- C. 1.get:2.uriParameter:3.{storeId} 4.queryParameter:5.department:
- D. 1./department:2.get:3.uriParameter:4.storeId:

**Answer: B**

Explanation:

Lets revise few concepts RAML which can help us to find the answer of this question.

URI Parameters

Lets have a look at below example.

```
* /foos:  
* /{id}:  
* /name/{name}:
```

Here, the braces {} around property names define URI parameters. They represent placeholders in each URI and do not reference root-level RAML file properties as we saw above in the baseUri declaration. The added lines represent the resources /foos/{id} and /foos/name/{name}.

Query Parameters

Now we'll define a way to query the foos collection using query parameters. Note that query parameters are defined using the same syntax that we used above for data types:

```
* /foos:  
* get:  
* description:List all Foos matching query criteria, if provided;  
* otherwise list all Foos  
* queryParameters:  
* name?:string  
* ownerName?:string
```

Based on the above information, below is the only option which defines storeid as uri parameter and department as query parameter.

```
* /{storeId}:  
* get:  
* queryParameter:  
* department:
```

## NEW QUESTION # 21

If you are curious or doubtful about the proficiency of our Salesforce-MuleSoft-Developer-I practice materials, we can explain the painstakingly word we did behind the light. By abstracting most useful content into the Salesforce-MuleSoft-Developer-I practice materials, they have help former customers gain success easily and smoothly. The most important part is that all contents were being sifted with diligent attention. No errors or mistakes will be found within our Salesforce-MuleSoft-Developer-I practice materials. We stress the primacy of customers' interests, and make all the preoccupation based on your needs.

**Salesforce-MuleSoft-Developer-I Latest Study Guide:** <https://www.prep4sures.top/Salesforce-MuleSoft-Developer-I-exam-dumps-torrent.html>

P.S. Free & New Salesforce-MuleSoft-Developer-I dumps are available on Google Drive shared by Prep4sures: <https://drive.google.com/open?id=1ozO7eSxRAeLr4ClhrHmrQHYuMxU4Zli>