MCD-LEVEL1 Q&As

MuleSoft Certified Developer - Level 1 (Mule 4)

Pass Mulesoft MCD-LEVEL1 Exam with 100% Guarantee

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

https://www.passapply.com/mcd-level1.html

100% Passing Guarantee
100% Money Back Assurance

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

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

QUESTION 1

Refer to the exhibit.

```
#%RAML 1.0
title: ACME Telecom API
version: 1.0

/plans:
  get:
    responses:
      200:
        body:
          application/json:
            example: |
              [
                {
                  "plan_type": "Super Saver 500",
                  "plan_details": "all-inclusive",
                  "monthly_discount": 0.10
                },
                {
                  "plan_type": "Business Plus 1000",
                  "plan_details": "business package",
                  "monthly_discount": 0.20
                }
              ]
```

The API needs to be updated using the company-wide standard for the Plan data type. The Object data type has already been published in Anypoint Exchange with the global reference . ACME/DataTypes/PlanDataType.raml. What is a valid RAML specification that reuses the Plan data type?

A.

```
#%RAML 1.0
title: ACME Telecom API
version: 1.0

dataTypes:
  Plan: !include ACME/DataTypes/PlanDataType.raml

/plans:
  get:
    responses:
      200:
        body:
          application/json:
            type: Plan[]
            example: !include ACME/Examples/PlanExamples.raml
```

B.

```
#%RAML 1.0
title: ACME Telecom API
version: 1.0

types:
  Plan: !reference ACME/DataTypes/PlanDataType.raml

/plans:
  get:
    responses:
      200:
        body:
          application/json:
            type: Plan[]
            example: !reference ACME/Examples/PlanExamples.raml
```

C.

```
#%RAML 1.0
title: ACME Telecom API
version: 1.0

dataTypes:
  Plan: !reference ACME/DataTypes/PlanDataType.raml

/plans:
  get:
    responses:
      200:
        body:
          application/json:
            type: Plan[]
            example: !reference ACME/Examples/PlanExamples.raml
```

D.

```
#%RAML 1.0
title: ACME Telecom API
version: 1.0

types:
  Plan: !include ACME/DataTypes/PlanDataType.raml

/plans:
  get:
    responses:
      200:
        body:
          application/json:
            type: Plan[]
            example: !include ACME/Examples/PlanExamples.raml
```

A. Option A

B. Option B

C. Option C

D. Option D

Correct Answer: D

As can be seen in RAML, POST expects input in application/json format which eliminates two of the options as two options are in xml format. Now out of the two remaining options, one has id field in request which is only mentioned for get

response and not for POST request. Hence id field is not expected in POST request.

Hence correct answer is

{

"name": "GoerdiLa Forge",

"address": "1 Westland CA",

"customer_since": "2014-01-04",

"balance": "4829.29",

"bank_agend_id": "12556"

}

QUESTION 2

A Utlility.dwl is located in a Mule project at src/main/resources/modules. The Utility.dwl file defines a function named encryptString that encrypts a String What is the correct DataWeave to call the encryptString function in a Transform Message component?

A. 1. %dw 2.0

2.

output application/json

3.

import modules::Utility

4.

--

5.

Utility::encryptString("John Smith")

B. 1. %dw 2.0

2.

output application/json

3.

import modules::Utility

4.

--

5.

encryptString("John Smith")

C. 1. %dw 2.0

2.

output application/json

3.

import modules.Utility

4.

--

5.

encryptString("John Smith")

D. 1. %dw 2.0

2.

output application/json

3.

import modules.Utility

4.

--

5.

Utility.encryptString("John Smith")

Correct Answer: B

Correct answer is %dw 2.0 output application/json import modules::Utility

Utility::encryptString("John Smith") DataWeave 2.0 functions are packaged in modules. Before you begin, note that DataWeave 2.0 is for Mule 4 apps. For Mule 3 apps, refer to DataWeave Operators in the Mule 3.9 documentation. For other Mule versions, you can use the version selector for the Mule Runtime table of contents. Functions in the Core (dw::Core) module are imported automatically into your DataWeave scripts. To use other modules, you need to import the module or functions you want to use by adding the import directive to the head of your DataWeave script, for example: import dw::core::Strings import camelize, capitalize from dw::core::Strings import * from dw::core::Strings The way you import a module impacts the way you need to call its functions from a DataWeave script. If the directive does not list specific functions to import or use * from to import all functions from a function module, you need to specify the module when you call the function from your script. For example, this import directive does not identify any functions to import from the String module, so it calls the pluralize function like this: Strings::pluralize("box"). Transform %dw 2.0 import dw::core::Strings output application/json

{ \\'plural\\': Strings::pluralize("box") }

QUESTION 3

What is the object type returned by the File List operation?

A. Object of String file names

B. Array of String file names

C. Object of Mule event objects

D. Array of Mule event objects

Correct Answer: D

The List operation returns an array of messages in which: Each message holds the file\\'s content in its payload. The file\\'s attributes section carries the file\\'s metadata (such as name, creation time, and size). The payload is empty if the element is a folder.

QUESTION 4

A Database On Table Row listener retrieves data from a CUSTOMER table that contains a primary key userjd column and an increasing kxjin_date_time column. Neither column allows duplicate values.

How should the listener be configured so it retrieves each row at most one time?

A. Set the watermark column to the bgin_date_time column

B. Set the target value to the last retrieved login_date_time value

C. Set the target value to the last retrieved user_jd value

D. Set the watermark column to the user_Id column

Correct Answer: A

1.

Watermark allows the poll scope to poll for new resources instead of getting the same resource over and over again.

2.

The database table must be ordered so that the "watermark functionality" can move effectively in the ordered list. Watermark stores the current/last picked up "record id."

3.

If the Mule application is shut down, it will store the last picked up "record id" in the Java Object Store and the data will continue to exist in the file. This watermark functionality is valuable and enables developers to have increased transparency.

4.

Developers do not need to create code to handle caching; it is all configurable!

5.

There are two columns and both are unique but user_id can\\'t guaranty sequence whereas date_time will always be in increasing order and table content can easily be ordered on the basis of last processed date_time. So correct answer is: Set the watermark column to the date_time column

QUESTION 5

How are query parameters dynamically passed to an outbound REST request using an HTTP Request operation?

A. As query parameters in the HTTP Request operation

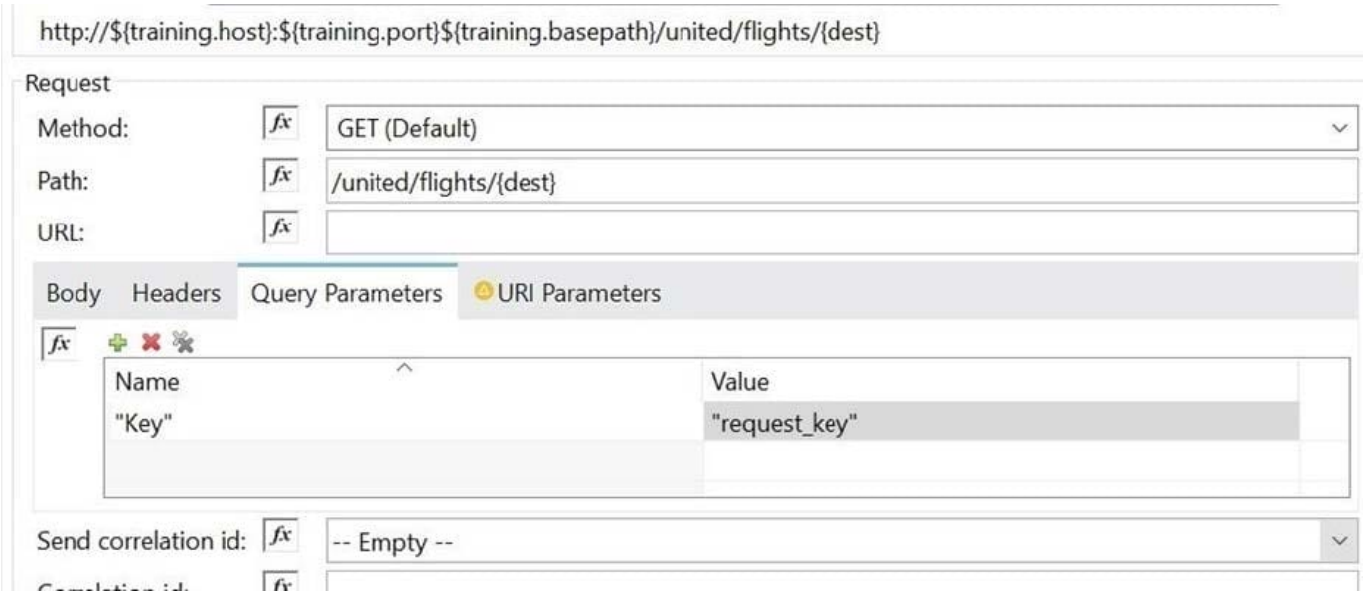
B. As URI parameters in the HTTP Request operation

C. In the Mule event\\'s payload

D. As flow variables

Correct Answer: A

In General > Request > Query Parameters, click the plus icon (+) to add a parameter to a request. Type a name and value for the parameter or use a DataWeave expression to define the name and value.



[Latest MCD-LEVEL1 Dumps](#) [MCD-LEVEL1 Practice Test](#) [MCD-LEVEL1 Study Guide](#)