MuleSoft Certified Developer - Integration Professional Exam Preparation Guide

Mule Runtime 3.7
February 9, 2016

Table of Contents

MuleSoft Certified Developer – Integration Professional Exam Preparation Guide

Preparation guide purpose

The purpose of this guide is to help you understand what knowledge and skills are needed to successfully pass the *MCD – Integration Professional* exam. It does not provide that knowledge, but includes information on how to obtain that knowledge.

Exam objective

The *MCD – Integration Professional* exam is a highly technical exam, designed to assess true expertise as a MuleSoft integration developer. The exam verifies strong software development skills, broad and deep Anypoint Platform knowledge, and appreciable experience using MuleSoft solutions on multiple types of projects.

Preparation recommendations

To prepare for taking the *MCD – Integration Professional* exam, you need to learn and have development experience with the topics listed in the Exam Topics section of this guide. This knowledge is best achieved by following this path:

- Take the *Anypoint Platform Essentials* training course, the self-paced *MuleSoft.U Developer Essentials* course, or gain equivalent knowledge.
- Pass the *MCD – Integration and API Associate* exam (this is suggested but not mandatory).
- Take the *Advanced Anypoint Platform Development* training course or gain equivalent knowledge.
- Get 6-months to several years of hands-on Mule project experience.

Note that actual development experience is required in addition to attending the training courses. Real project experience is needed to pass this challenging exam.

Required experience

This test is designed to verify product expertise gained through significant experience using MuleSoft products on a variety of projects. There is no official requirement for the amount of experience you need to pass the exam, but history shows that people who pass this challenging exam have months or years of real MuleSoft project experience.

Intended audience

The target audience for the exam are people that have true expertise as MuleSoft integration developers. This typically maps to roles including:

- Developers (various languages)
- System Integrators
- Solution Consultants
- Technical Architects
- Business Architects
- Some Program Managers
- Some Project Managers

Exam cost

The exam fee is $250 USD per person per attempt. Other purchase bundles may also be available that include this exam; to check, see http://training.mulesoft.com/catalog.

Exam registration

The exam is administered by the Kryterion Webassessor testing platform. To register for the exam, go to https://www.webassessor.com/home.do?page=PUBLIC&branding=MULESOFT, create a user profile, log in, select Register for an Exam, and select the *MCD – Integration Professional* exam. On the payment screen, you will have the option of paying by credit card and/or entering a voucher code.

Exam format

Format:	Multiple-choice, closed-book, proctored
Number of questions:	100
Duration:	Up to 2 hours are permitted
Language:	English
Delivery method:	In a Kryterion testing center location or online
Availability:	Both in person and online exams need to be scheduled in advance

Testing location

There are two options for taking the exam:

- In any testing center location within Kryterion's worldwide network
- Online from any internet-connected computer using Kryterion Webassessor's Online Proctoring service and a qualifying webcam

Exam pass requirement

A score of 80% or higher is required to pass the *MCD – Integration Professional* exam.

Certification validity

The *MCD – Integration Professional* accreditation expires two years from the date of passing.

Exam topics

The topics covered in the *MCD – Integration Professional* exam are listed here.

Basics

- Understanding Mule applications, flows, messages, and message processors
- Building, running, and debugging Mule applications
- Configuring and using HTTP components
- Configuring transport protocols and endpoints
- Configuring and using Java components
- Manipulating POJOs

Batch

- Understanding the various phases of batch processing
- Configuring and using batch processing

Cache

- Using the Cache Scope to store and reuse frequently called data
- Adding and configuring a Cache Scope
- Creating and managing caching strategies

Clusters

- Managing server clusters
- Understanding the general concepts and benefits for building Mule clusters
- Using queues to distribute application flows for processing in clusters
- Understanding how clustering supports various Mule transport mechanisms

Common Error Scenarios

- Configuring global application exception handling
- Debugging flows and expression handlers

DataWeave

- Using the DataWeave Transform Message component
- Writing DataWeave expressions
- Using DataWeave with data sources that have associated metadata
- Adding custom metadata to data sources

Deployment and Management Consoles

- Deploying applications on-prem using Mule Management Console
- Deploying applications to CloudHub
- Organizing Spring properties and Spring property file configuration
- Using the MMC Flow Analyzer to review application flows and understanding why this is critical during configuration and application deployment

Enricher

- Using Enrichers to enhance a Mule message
- Understanding how the Message Enricher processes data and relevant use cases
- Understanding complex enrichment

Exceptions

- Using exception strategies and understanding how they affect flows and sub-flows
- Understanding the different exception strategies that are available
- Changing and returning a message from an exception strategy
- Using routers (like First Successful and Until Successful) to handle potential error conditions

Expression Routing

- Writing Mule expressions
- Using expressions to route messages

Expressions

- Referencing inbound and outbound properties
- Using flow variables and session variables
- Manipulating Maps and Lists with expressions
- Setting variable and property elements with expressions
- Setting and testing payload attributes with expressions
- Using Expression Transformers, Filters, Choice Router, and Loggers

Flows and Testing

- Building various flow types
- Coding and testing exchange patterns (like request-response and one-way)
- Understanding the differences between inbound and outbound endpoints
- Testing using JUnit cases with Mule applications
- Sending a Mule message from a test class to a Mule application

General Knowledge Questions

- Understanding basic MuleSoft implementation and design concepts
- Creating flows and using message processors
- Debugging flows and common error scenarios
- Understanding data movement through an application

HTTP

- Creating and configuring inbound and outbound HTTP endpoints
- Using HTTP and HTTPS
- Building independent HTTP flows in Anypoint Studio
- Understanding HTTP content-type and the effect on browser types

Java Custom Components

- Creating and testing Java custom components and integrating them into flows
- Using advanced Java concepts to invoke service calls for passing Mule messages
- Creating custom transformers with Java
- Configuring Java components to be prototypes or singletons
- Using the default entry point resolver with Java components

JDBC

- Configuring and using JDBC connectors
- Understanding how JDBC inbound and outbound endpoints differ and their limitations
- Using expressions to access the contents of JDBC return objects
- Using SQL with DataMappers and expressions

JMS

- Configuring JMS connectors for two-way communications, temporary queues, and object serialization over transports
- Configuring and using JMS inbound and outbound endpoints
- Using JMS queues and topics
- Using back channels and creating two-way communication through JMS connections
- Understanding how JMS uses correlation IDs

MuleSoft Solutions' Place in the Enterprise

- Understanding how systems, data, and applications can be integrated within an enterprise and the potential business impact
- Understanding integration points for Mule within an enterprise
- Understanding the benefits of using the Mule runtime

Properties

- Defining Mule properties and creating properties files
- Understanding the benefits of using global properties

REST

- Implementing REST services with GET, POST, PUT, and DELETE methods
- Using annotations on REST methods to create unique signatures
- Using REST blocks to absorb resources
- Creating REST clients and working with dynamic endpoints

Routing

- Using splitters, aggregators, and multi-cast routers
- Using the For-each scope
- Using basic filters with routing
- Using flows, sub-flows, and flow references

SOAP Web Services

- Publishing and/or consuming SOAP messages
- Using CXF interfaces to create service definitions
- Extending interfaces to create CXF implementations

Transactions

- Understanding transaction management and which endpoints support transactions
- Managing and configuring resource transactions for inbound and outbound messages
- Understanding the various transaction types and usage techniques

VM Transport

- Using VM Transport to control how messages are sent and received by components in a system
- Using VM Transport for communication between Mule flows
- Understanding queue usage with VM Transport and configuration structure

Sample exam questions

Fourteen sample exam questions are provided here. Answers are provided at the end.

1. Can a flow have more than one message source?

A. Yes, using the composite source
B. Yes, if declared sequentially at the beginning of the flow
C. No, Mule validates on startup that the next element after a message source is a message processor
D. It depends on if the two message sources have compatible transport types

2. Which of the following transformations is not possible when using DataMapper?

A. Excel Spread Sheet to a CSV file
B. CSV file to a POJO
C. XML file to Map
D. JSON to HTML

3. Fill in the Blank: Mule ESB and CloudHub can have multiple _____, each of which can be transmitting concurrent _____.

A. Applications, Messages
B. Nodes, Loads
C. Flows, Endpoints
D. Endpoints, Workers

4. Where can't I define an Exception Strategy?

A. Sub-flow
B. Flow
C. Private flow
D. Global configuration

5. Expressions can be used in MuleSoft integration:

A. To extract information from the current message
B. With routers and filters for defining routing logic
C. For filtering out unwanted messages
D. All of the above

6. In order to Unit test a MuleSoft application it's necessary to extend what class?

A. org.mule.tck.junit4.FunctionalTestCase
B. org.mule.tck.functional.FunctionalTestComponent
C. org.mule.tck.exceptions.FunctionalTestException
D. org.junit.Test

7. One can use Mule Expression Language in which of the following contexts?

A. Application
B. Message
C. Server
D. All

8. When receiving HTTP responses, the payload of the MuleMessage will always be:

A. A HashMap of all query parameters
B. An InputStream
C. Dependent on the URL and how it comes to Mule (webserver, proxy, etc.)
D. Either JSON, a string or a HashMap depending on the request

9. If the component implements Callable lifecycle interface, what method would you use to receive the message?

A. onCall()
B. initialize()
C. Start()
D. None of the above

10. What are the databases that MuleSoft integrations can connect to?

A. Any JDBC compliant database
B. Oracle, MySQL, Postgres, Derby, DB2, MSSQL
C. Any database
D. Oracle and Postgres

11. What is the purpose of setting the maxRedelivery attribute on the connector URL?

A. To deliver each message in the queue up to the value set in maxRedelivery, without differentiating between messages that generated errors and others
B. That attribute is not possible to set on a JMS connector
C. To avoid an infinite loop trying to process a message that will always generate an error
D. To define the maximum number of receivers for the message

12. What's the correct way to define a dynamic inbound endpoint?

A. Dynamic endpoints are only supported for outbounds
B. <http:inbound-endpoint host="0.0.0.0" path="/#[expresion.for.path]" port="port" />
C. <http:inbound-endpoint address="http://host:port/#[expresion.for.path]" />
D. <dynamic-source expression=""[expresion.for.path]">
<http:inbound-endpoint rootAddress="http://host:port/" />
</dynamic-source>

13. What is a Scope?

A. It is a construct that contains message processors and limits the scope of flow variables
B. It is another name for flows and sub-flows
C. It is a construct that contains message processors, and changes the way they execute
D. It is a construct that executes message processors asynchronously to the container flow

14. What is the appropriate way to log SOAP messages in CXF?

A. Logging the payload as a String
B. Using cxf:inInterceptors
C. Enabling logging in the CXF configuration

Sample exam question answers

Question	Correct Answer
1	A
2	D
3	A
4	A
5	D
6	A
7	D
8	B
9	A
10	A
11	C
12	A
13	C
14	B