

MuleSoft Certified Developer – Level 1

Certification Exam

Summary

A *MuleSoft Certified Developer – Level 1* should be able to successfully work on basic Mule 4 projects with guidance and supervision. The *MCD – Level 1 (Mule 4)* exam validates that a developer has the required knowledge and skills to design, build, test and debug, deploy, and manage basic APIs and integrations: moving from Anypoint Platform to Anypoint Studio and back. Certified candidates should be able to:

- Use MuleSoft-hosted Anypoint Platform to take a basic API through all the steps of its lifecycle: design, build, deploy, manage, and govern.
- Use Anypoint Studio to build, test, and debug basic integrations and API implementations.
- Connect to a range of resources including databases, files, web services, SaaS applications, and JMS queues.
- Perform basic data transformations using DataWeave.
- Control event flow and handle errors.
- Process batch records.

Format

- Format: Multiple-choice, closed book, proctored, online
- Length: 60 questions
- Duration: 120 minutes (2 hours)
- Pass score: 70%
- Language: English

You can take the exam a maximum of 5 times, with a 24-hour wait between each attempt.

Cost

You can purchase the exam for \$400. This includes one free retake. Additional retakes (i.e. attempts 3 to 5) are 50% off and do not come with a free retake.

You can also get two exam attempts with the successful completion of the *Anypoint Platform Development: Fundamentals (Mule 4)* course or the *Anypoint Platform Development: Mule 4 for Mule 3 Users* course.

Validity

The certification expires two years from the date you pass the exam. To extend the certification validity after this date, you can take the *MuleSoft Certified Developer – Level 1 (Mule 4) MAINTENANCE* exam.

Preparation

You can best prepare for the exam by taking the instructor-led *Anypoint Platform Development: Fundamentals (Mule 4)* course and completing the accompanying Do-It-Yourself (DIY) exercises and practice exam. Candidates should be familiar with all of the content in the course and be able to apply the concepts in actual projects.

The following resources are available to help you prepare:

- **Instructor-led training: *Anypoint Platform Development: Fundamentals (Mule 4)***
 - Recommended as the most effective and efficient method of preparation
 - 5-day class
 - Private, public, onsite, and online classes available
 - Includes two attempts for this exam
- **Self-study training: *Anypoint Platform Development: Fundamentals (Mule 4)***
 - 60+ step-by-step exercises to teach you the basics
 - All content available instantly for you to complete at your own pace
 - Supported by the peer-to-peer MuleSoft training forum
 - Successful completion of the course includes two MuleSoft-sponsored attempts for this exam
- **Do-it-yourself exercises**
 - 10+ DIY exercises to get experience with and apply the knowledge gained in class
 - Starting code and solutions provided
 - Can be completed in any order
- **Practice exam**
 - Same format, length, and duration as the certification exam
 - Questions are the same style and degree of difficulty as the exam

Topics

The exam validates that you can perform the following tasks.

Note: DEV: FUN4 is the acronym for the Anypoint Platform Development: Fundamentals (Mule 4) course. DEV: DIY4 is the acronym for the MCD - Level 1 / Development Fundamentals DIY Exercises & Practice Exam materials.

Explaining application network basics	Resources
<ul style="list-style-type: none"> Explain MuleSoft's proposal for closing the IT delivery gap. Describe the role and characteristics of the "modern API." Describe the purpose and roles of a Center for Enablement (C4E). Define and describe the benefits of API-led connectivity and application networks. Define and correctly use the terms API, API implementation, API interface, API consumer, and API invocation. Describe the basics of the HTTP protocol and the characteristics of requests and responses. Describe the capabilities and high-level components of Anypoint Platform for the API lifecycle. 	<ul style="list-style-type: none"> DEV: FUN4 Module 1 DEV: FUN4 Module 2
Designing and consuming APIs	
<ul style="list-style-type: none"> Describe the lifecycle of the "modern API." Use RAML to define API resources, nested resources, and methods. Identify when and how to define query parameters vs URI parameters. Use RAML to define API parameters, requests, and responses. Use RAML to define reusable data types and format-independent examples. Read a RAML spec and formulate RESTful requests with query parameters and/or headers as appropriate. 	<ul style="list-style-type: none"> DEV: FUN4 Module 3 DEV: DIY4 Exercise 3-1 and 4-1
Accessing and modifying Mule events	
<ul style="list-style-type: none"> Describe the Mule event data structure. Use transformers to set event payloads, attributes, and variables. Write DataWeave expressions to access and modify event payloads, attributes, and variables. Enrich Mule events using target parameters. 	<ul style="list-style-type: none"> DEV: FUN4 Module 6 DEV: DIY4 Exercise 6-1, 7-1, and 7-2 <u>Enriching Data with Target Parameters</u>

Structuring Mule applications	
<ul style="list-style-type: none"> • Parameterize an application using property placeholders. • Define and reuse global configurations in an application. • Break an application into multiple flows using private flows, subflows, and the Flow Reference component. • Specify what data (payload, attributes, variables) is persisted between flows when a Flow Reference is used. • Specify what data (payload, attributes, variables) is persisted between flows when a Mule event crosses a connection boundary. • Specify what data (payload, attributes, variables) exists in a flow before and after a call in the middle of a flow to an external resource. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 7 • DEV: DIY4 Exercise 7-1 and 7-2
Building API implementation interfaces	
<ul style="list-style-type: none"> • Manually create a RESTful interface for a Mule application. • Generate a REST Connector from a RAML specification. • Describe the features and benefits of APIkit . • Use APIkit to create implementation flows from a RAML file. • Describe how requests are routed through flows generated by APIkit. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 4 • DEV: FUN4 Module 8 • DEV: DIY4 Exercise 4-1
Routing events	
<ul style="list-style-type: none"> • Use the Choice router to route events based on conditional logic. • Use the Scatter-Gather router to multicast events. • Validate data using the Validation module. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 9 • DEV: DIY4 Exercise 9-1
Handling errors	
<ul style="list-style-type: none"> • Describe the default error handling in a Mule application. • Define a custom global default error handler for an application and identify in what situations it will be used. • Compare and contrast how the On Error Continue and On Error Propagate scopes work. • Create one or more error handlers for a flow. • Use the Try scope to specify error handlers for one or more event processors. • Describe the data structure of the Mule Error object. • Map errors to custom application errors. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 10 • DEV: DIY4 Exercise 10-1

Transforming data with DataWeave	
<ul style="list-style-type: none"> • Write DataWeave scripts to convert JSON, XML, and Java data structures to different data structures and data types. • Use DataWeave functions. • Define and use DataWeave variables, functions, and modules. • Define and use custom data types. • Apply correct DataWeave syntax to coerce data types. • Apply correct DataWeave syntax to format strings, numbers, and dates. • Call Mule flows from a DataWeave script. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 11 • DEV: DIY4 Exercise 11-1
Using Connectors	
<ul style="list-style-type: none"> • Retrieve data from a Database using the Database connector. • Create parameterized SQL queries for the Database connector. • Retrieve data from a REST service using the HTTP Request operation or a REST Connector. • Use a Web Service Consumer connector to consume a SOAP web service. • Use the Transform Message component to pass arguments to a SOAP web service. • List, read, and write local files using the File connector. • List, read, and write remote files using the FTP connector. • Use the JMS connector to publish and listen for JMS messages. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 4 • DEV: FUN4 Module 8 • DEV: FUN4 Module 12 • DEV: DIY4 Exercise 4-1, 8-1, 12-1, and 12-2
Processing records	
<ul style="list-style-type: none"> • List and compare and contrast the methods for processing individual records in a collection. • Explain how Mule events are processed by the For Each scope. • Use the For Each scope to process records. • Explain how Mule events are processed by the Batch Job scope. • Use a Batch Job with Batch Steps and a Batch Aggregator to process records. • Use the Scheduler component to trigger a flow. • Use connector listeners to trigger flows. • Describe the features, benefits, and process to use automatic watermarking vs. manual watermarking. 	<ul style="list-style-type: none"> • DEV: FUN4 Module 12 • DEV: FUN4 Module 13 • DEV: DIY4 Exercise 13-1

<ul style="list-style-type: none">• Use connectors with automatic watermarking capabilities.• Persist data between flow executions using the Object Store.	
Debugging and troubleshooting Mule applications	
<ul style="list-style-type: none">• Use breakpoints to inspect a Mule event during runtime.• Install missing Maven dependencies.• Read and decipher Mule log error messages.	<ul style="list-style-type: none">• DEV: FUN4 Module 6• DEV: FUN4 all WTs• DEV: DIY4 Exercise 6-1 and Walkthrough• DEV: DIY4 all exercises
Deploying and managing APIs and integrations	
<ul style="list-style-type: none">• Package Mule applications for deployment.• Deploy applications to CloudHub.• Use CloudHub properties to ensure deployment success.• Create and deploy API proxies.• Connect an API implementation to API Manager using autodiscovery.• Use policies, including client ID enforcement, to secure an API.• Create SLA tiers and apply SLA based policies.	<ul style="list-style-type: none">• DEV: FUN4 Module 5• DEV: DIY4 Exercise 5-1 and 5-2• <u>Configuring API Autodiscovery in a Mule 4 Application</u>

More information

For more information, visit <http://help.learn.mulesoft.com>.