

# MuleSoft Certified Developer – Level 1 (Mule 4) 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 2.0.
- Control event flow and handle errors.
- Process batch records.

## Format

- Format: Multiple-choice, closed book
- 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 with one of the following. Each includes one free retake.

- \$250
- 1 Flexible Training Credit (FTC)

Additional retakes (i.e. attempts 3 to 5) are 50% off and do not come with a free retake.

Two free exam attempts are also included with the purchase of the instructor-led *Anypoint Platform Development: Fundamentals (Mule 4)* course and the *Anypoint Platform Development: Mule 4 for Mule 3 Users* course.

## Validity

The certification expires two years from the date you pass the 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. **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 free 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](#)
  - Does not include any free attempts for this exam
- **Self-assessment quiz**
  - 5+ multiple-choice questions for each course module
  - Identifies strengths and weaknesses
- **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

## Topics

The exam validates that the candidate 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 (Mule 4) Self-Assessment Quiz & DIY Exercises materials.*

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

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

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

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

## More information

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