

**Error Handling in MuleSoft**

**Course Number:** MULE-112
**Duration:** 1.5 days

**Overview**

This hands-on, online MuleSoft Error Handling training course teaches students how to capture and handle errors in Mule applications at the application, flow, and processor levels. Attendees also learn how to create custom error messages and use HTTP response codes.

**Prerequisites**

Students must have experience with creating, deploying, and managing APIs in Anypoint Studio and Anypoint Platform. Students must also be able to read and interpret information in the Logger console and the Mule debugger.

**Materials**

All MuleSoft training students receive comprehensive courseware.

**Software Needed on Each Student PC**

Students will not need to install any software on their computers for this class. The class will be conducted in a remote environment. Students need a local computer with a web browser (preferably Chrome), stable internet, two monitors, and a headset/microphone.

**Objectives**

* Observe the behavior of the Mule default error handler
* Detect different types of errors and their hierarchy
* Compare the On-Error Propagate Scope and the On-Error Continue Scope
* Handle errors at the application, flow, and processor levels
* Create and modify error response scopes, settings, and messages
* Use the Try scope to handle errors at the processor level
* Review auto-generated error handling in the interface
* Connect the implementation with the interface
* Handle system errors by setting a reconnection strategy
* Reinforce error handling behaviors and error messages

**Outline**

* Menacing the Mule
	+ Mule Default Error Handler
	+ Error Object Properties
	+ Types of Errors
	+ HTTP Default Settings
* The Application Level
	+ The Global Error Handler
		- On-Error Propagate
		- On-Error Continue
	+ Create and Configure Error Handlers
	+ Modify HTTP Default Settings
	+ Catch Distinct Error Types
	+ Map Custom Error Types
* The Flow and Processor Levels
	+ Track Errors in Flows and Subflows
	+ Global Error Handler Ignored?
	+ The Try Scope at the Processor Level
	+ On-Error Propagate
	+ On-Error Continue
	+ Create and Configure Error Handlers
* Our API
	+ Add Error Handling
	+ Connect the Implementation and the Interface
	+ Adjust Error Scopes
* Your API
	+ Add Error Handling
	+ Connect the Implementation and the Interface
	+ Adjust Error Scopes
* Predict Behaviors
	+ Follow On-Error Propagate Flows
	+ Follow On-Error Continue Flows
	+ Predict the Paths of Flows
	+ Predict Error Message Responses
* Conclusion