Anypoint Platform Development: DataWeave (Mule 3)

Summary
This course is for developers interested in advancing their DataWeave 1.0 skills beyond those taught in the Anypoint Platform Development: Fundamentals (Mule 3) course.

Duration
1 day (in-person or online)

Objectives
At the end of this course, students should be able to:

- Write generalized and reusable transformations using functions, variables, and operators.
- Build complex transformations from smaller testable steps.
- Transform, combine, and reduce complex data structures which include nested arrays, objects, and arrays of objects.
- Recursively apply formatting rules to every element of a nested schema.

Prerequisites
- Experience developing Mule 3 applications as demonstrated by one of the following:
  - Passing the MuleSoft Certified Developer - Integration and API Associate (Mule 3) exam
  - Completion of the Anypoint Platform Development: Fundamentals (Mule 3) course
- If new to functional programming, read An introduction to functional programming in JavaScript

Setup requirements
- A computer with at least 4GB available RAM, 2GHz CPU, and 10GB available hard drive space
- Unrestricted internet access to port 80 (with > 5Mbps download and > 2Mbps upload)
- JDK 1.8
- Anypoint Studio 6.4 or later with embedded Mule 3.9 runtime
- Postman REST client (or any other REST client application)

Get a detailed setup document here.

Outline

Module 1: Organizing and formatting DataWeave code
- Review DataWeave fundamentals
- Match DataWeave types and conditions
• Organize and reuse DataWeave code with variables and functions

Module 2: Constructing arrays and objects with DataWeave
• Add components to and remove elements from arrays and objects
• Construct objects from lists of DataWeave expressions using object constructor curly braces { }
• Troubleshoot common issues when using object constructor curly braces { }
• Join data together using map operators

Module 3: Iteratively transforming data using mapping operators
• Transform elements of arrays and objects into a new array using the map operator
• Transform elements of objects into a new object using the mapObject operator
• Combine map and mapObject operators to transform complex schema

Module 4: Recursively transforming complex schema
• Write recursive functions to transform complex schema
• Combine match and mapping operators to recursively transform every element of a complex schema