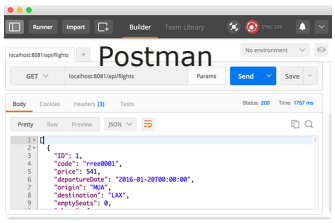
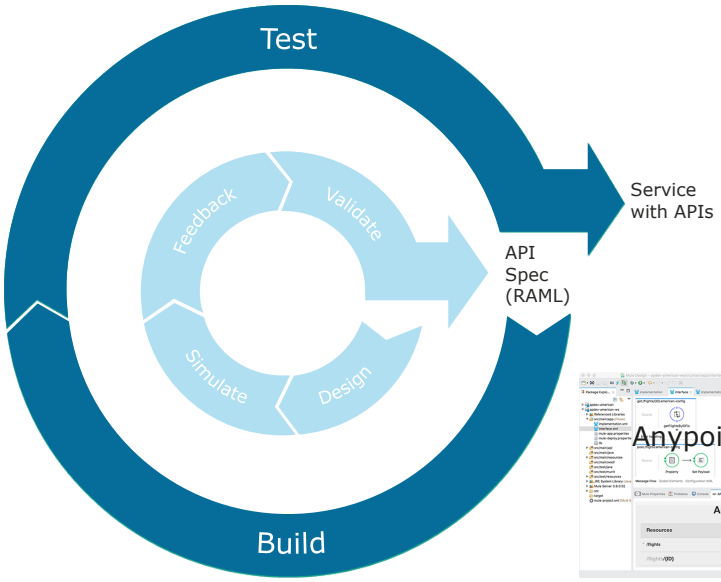
MuleSoft

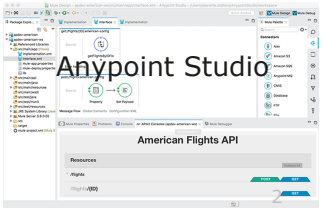
Module 4: Building APIs

Goal

MuleSoft







At the end of this module, you should be able to

MuleSoft

- Use Anypoint Studio to build, run, and test Mule applications
- Use a connector to connect to databases
- Use the graphical DataWeave editor to transform data
- Create RESTful interfaces for applications from RAML files
- Connect API interfaces to API implementations

Comparing Mule 3 and Mule 4 applications

Creating Mule 3 and Mule 4 applications

MuleSoft

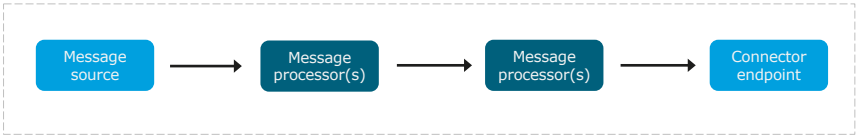
- In Module 2, you created **Mule 4** applications with **flow designer**
  - Flow designer apps can only be built using Mule 4, currently an early access version
- In this module, you are going to create **Mule 3** applications with **Anypoint Studio** 6.X and Mule 3.9.X
  - These are the current GA versions of the Mule runtime and Anypoint Studio
- Get early access to Studio 7 and Mule 4
  - https://www.mulesoft.com/lp/dl/mule-studio-beta

Mule 3 applications

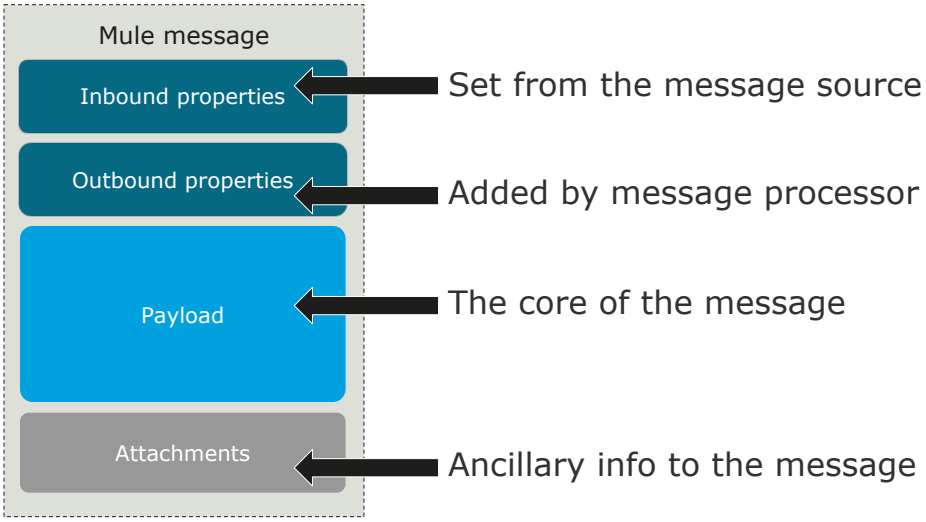
MuleSoft

- **Mule 4** applications are built using components that are **Mule event processors**
- **Mule 3** applications are built using elements that are **Mule message processors**
  - Accept and process **messages** through a series of **message processors** plugged together in a **flow**

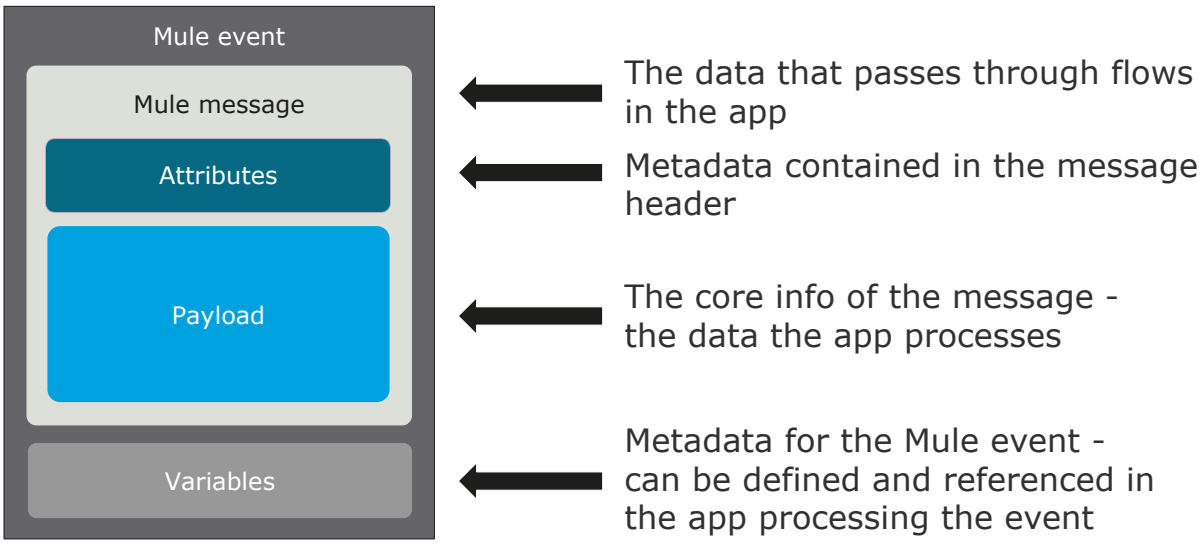
Flow



Mule 3 message structure



Mule 4 event structure

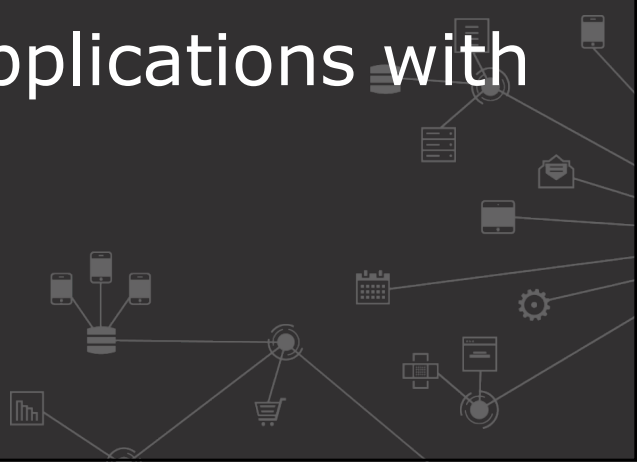


Some differences between Mule 4 and Mule 3

MuleSoft

- Mule 4 has a **simplified Mule event and message model**
- The building blocks of Mule 4 applications are **components** that are Mule event processors
  - The building blocks of Mule 3 apps are **elements** that are Mule message processors
- Mule 4 uses **DataWeave 2.0**, Mule 3.7+ uses **DataWeave 1.0**

Creating Mule applications with Anypoint Studio



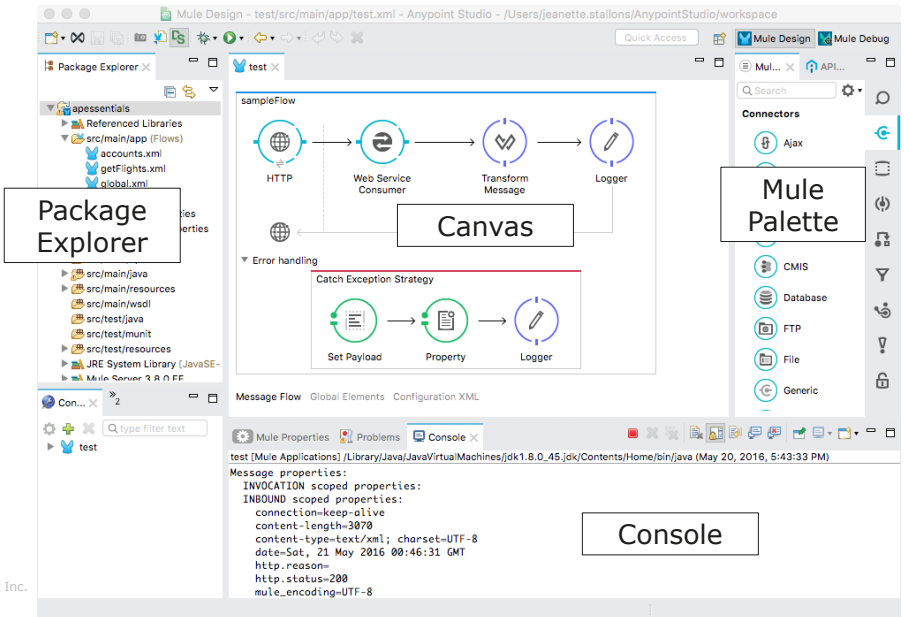
Creating Mule applications with Anypoint Studio

MuleSoft

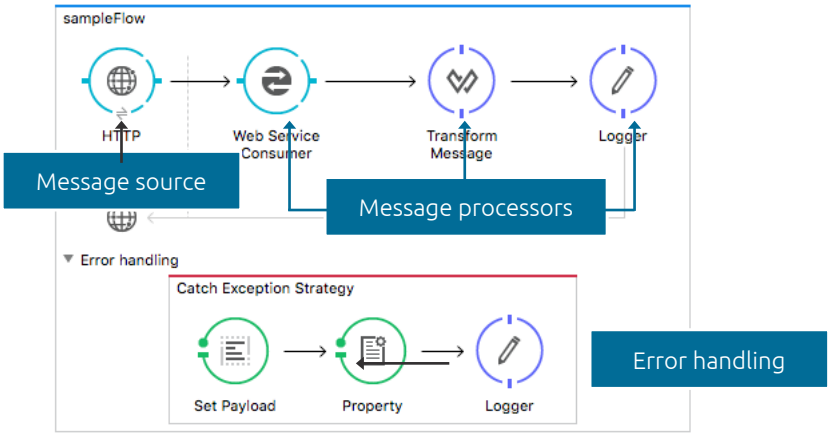
- Studio is an Eclipse-based integration development environment
  - Two-way editing between graphical and XML views
  - An embedded Mule runtime to test applications without leaving it
  - Visual debugging (EE)
  - Pre-built tooling to connect to
    - Many popular services (Salesforce, Workday, Facebook, more!)
    - Many standard protocols (HTTP, HTTPS, FTP, SMTP, more!)
    - Any SOAP or RESTful API
  - A data transformation framework and language (EE)
  - One-click deployment of applications
  - Templates for common integration patterns (EE)
  - Integration with Maven for continuous build processes

Anypoint Studio anatomy

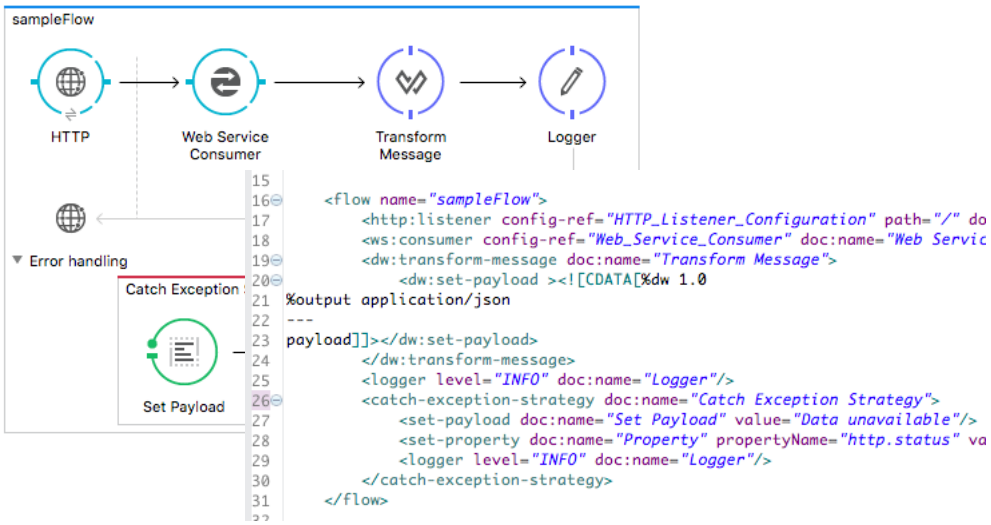
MuleSoft



Anatomy of a flow: Visual



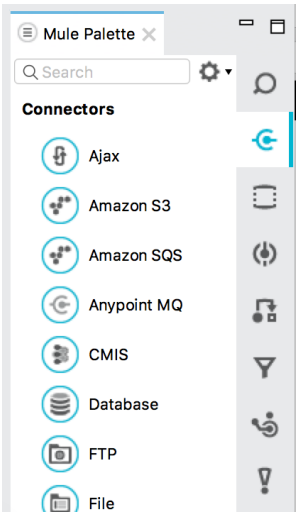
Anatomy of a flow: XML



Mule application building blocks

MuleSoft

- Are separated into categories in the Mule Palette

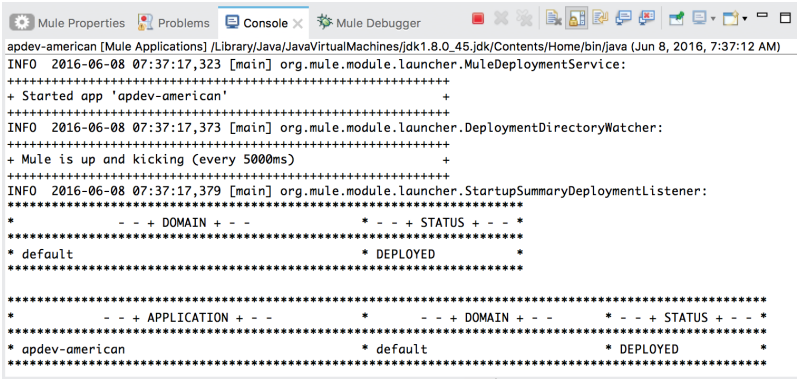


Connectors
Scopes
Components
Transformers
Filters
Flow Control
Error Handling

Running applications

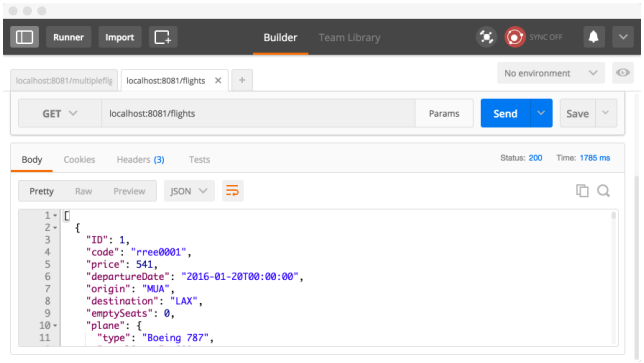
MuleSoft

- Anypoint Studio comes with an embedded Mule runtime to test applications without leaving it
- The console outputs application logs and information



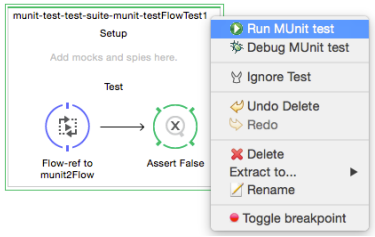
Testing applications by making requests to endpoints

- Some options
  - A browser
  - A cURL command-line utility
  - A browser extension like Postman (for Google Chrome)



Automating testing of applications

- You can automate testing of Mule applications using MUnit
- MUnit is a Mule application testing framework for building automated tests
- MUnit is fully integrated with Anypoint Studio
  - You can create, design, and run MUnit tests just like you do Mule applications

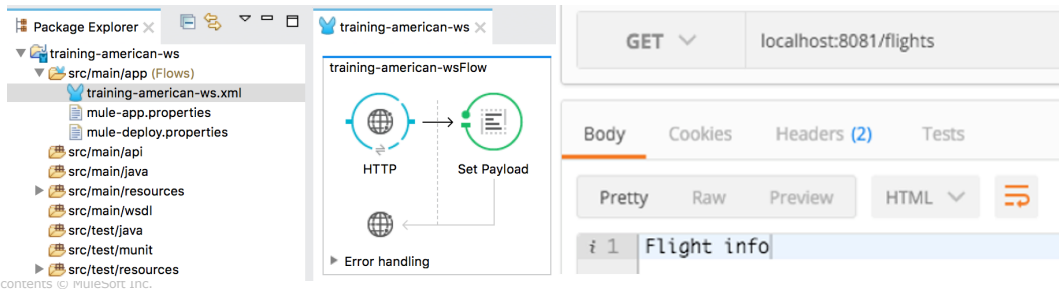


- MUnit is covered in the *Anypoint Platform Development: Advanced* course

Walkthrough 4-1: Create a Mule application with Anypoint Studio

MuleSoft

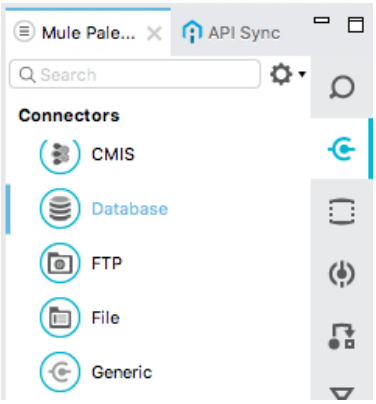
- Create a new Mule project with Anypoint Studio
- Add a connector to receive requests at an endpoint
- Set the message payload
- Run a Mule application using the embedded Mule runtime
- Make an HTTP request to the endpoint using Postman



Connecting to data

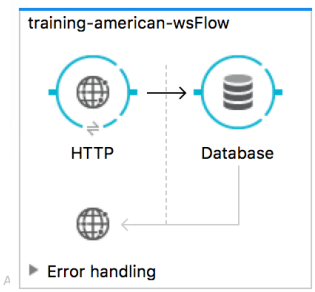
The Database connector

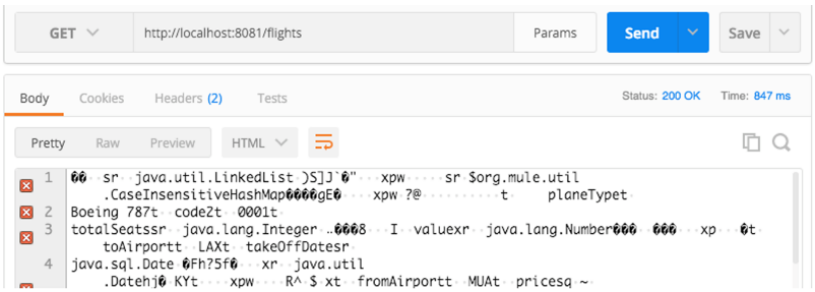
- Can connect to almost any JDBC relational database
  - Any database engine for which you have a driver
- Supports operations including
  - SELECT, INSERT, UPDATE, DELETE
  - Stored Procedures
  - Bulk Execute
  - Data Definition Language (DDL) requests like CREATE and ALTER



Walkthrough 4-2: Connect to data (MySQL database)

- Add a Database connector endpoint
- Configure a Database connector that connects to a MySQL database
  - Or optionally an in-memory Derby database if you do not have access to port 3306
- Configure the Database endpoint to use that Database connector
- Write a query to select data from a table in the database





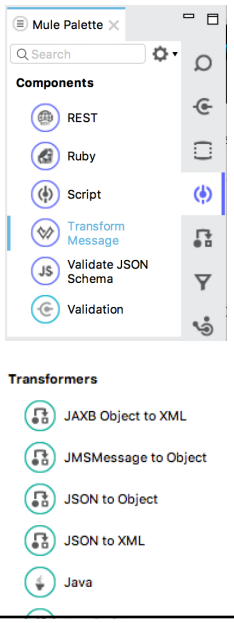
Transforming data

Transforming data

MuleSoft

- Use **DataWeave** and **Transform Message** component for all simple and complex transformations
  - DataWeave was introduced and used in Module 2
  - DataWeave is a JSON-like language that's built just for data query and transformation use cases

- Note: The main way to transform data before Mule 3.7 was to use a set of transformers



DataWeave 1.0

MuleSoft

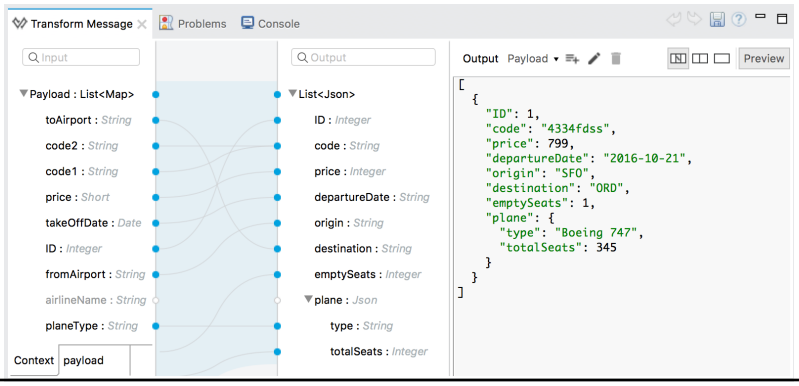
- **DataWeave 1.0** is the expression language for Mule to access, query, and transform **Mule 3** event data

- DataWeave 1.0 vs DataWeave 2.0
  - In Mule 4, DataWeave is the default expression language for everything
  - In Mule 3, there is the Mule Expression Language (MEL) and DataWeave is just for transformations
  - DataWeave 2.0 is simpler - everything is now a function
- Fully integrated with Anypoint Studio 6.4.X
  - Graphical interface with payload-aware development

Walkthrough 4-3: Transform data

MuleSoft

- Use the Object to JSON transformer
- Replace it with a Transform Message component
- Use the DataWeave visual mapper to change the response to a different JSON structure

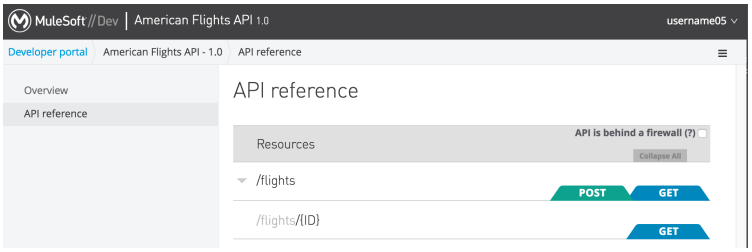


Creating RESTful interfaces manually for Mule applications

Creating RESTful interfaces

MuleSoft

- A RESTful interface for an application will have listeners for each resource / method pairing defined by the API
  - GET: /flights
  - GET: /flights/{ID}
  - POST: /flights

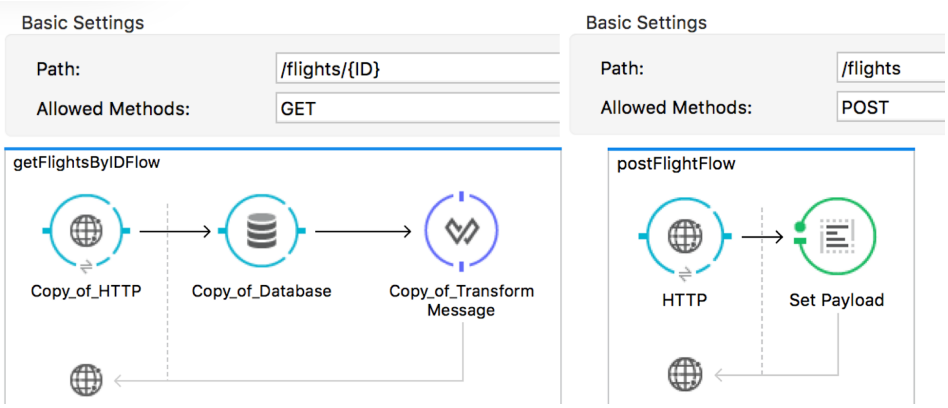


- You can create the interface manually or have it generated from the API definition
  - We will do both in next two walkthroughs

Walkthrough 4-4: Create a RESTful interface for a Mule application

MuleSoft

- Route based on path
- Add a URI parameter to a new HTTP Listener endpoint path
- Route based on HTTP method

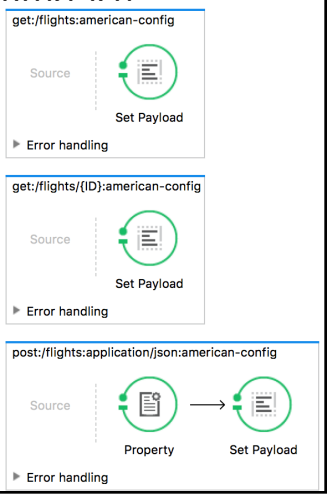


36

Generating RESTful interfaces automatically using APIkit

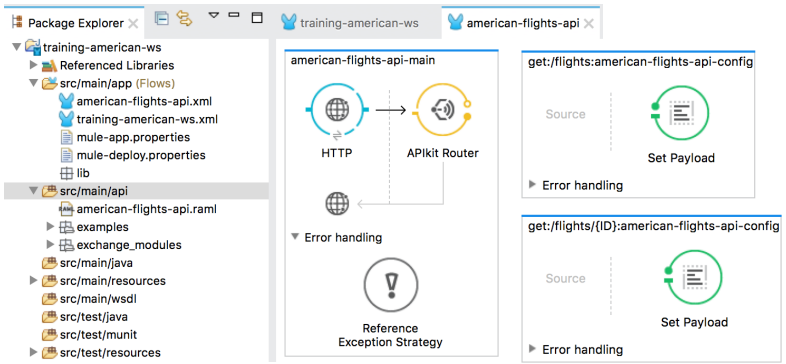
Creating RESTful interfaces automatically using APIkit

- **APIkit** is an open-source toolkit that includes an Anypoint Studio plugin
- The Anypoint Studio **APIkit plugin** can generate an interface automatically from a RAML API definition
  - For new or existing projects
- It generates a main routing flow and flows for each of the API resource / method pairs
- You add processors to the resource flows to hook up to your backend logic



Walkthrough 4-5: Use Anypoint Studio to create a RESTful API interface from a RAML file

- Add Anypoint Platform credentials to Anypoint Studio
- Import an API from Design Center into an Anypoint Studio project
- Use APIkit to generate a RESTful web service interface from an API
- Test the web service using Postman

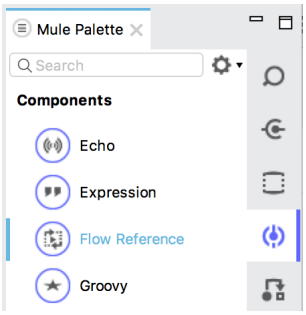


Connecting interfaces to implementations

Passing messages to other flows

MuleSoft

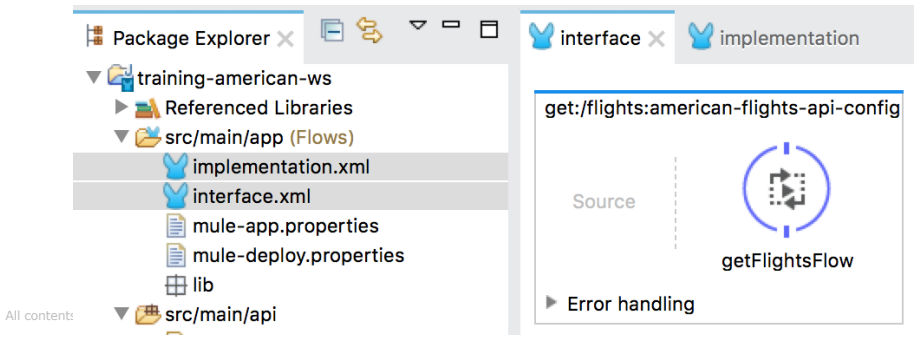
- Flows can be broken into multiple flows
  - Makes the graphical view more intuitive and the XML code easier to read
  - Promotes code reuse
- All flows are identified by name and can be called via **Flow Reference** components in other flows



Walkthrough 4-6: Implement a RESTful web service

MuleSoft

- Pass a message from one flow to another
- Call the backend flows
- Create new logic for the nested resource call
- Test the web service using Postman



Summary

Summary

MuleSoft

- **Anypoint Studio** can be used to build Mule applications for integrations and API implementations
  - Two-way editing between graphical and XML views
  - An embedded Mule runtime for testing applications
- **Mule applications** accept and process messages through a series of message processors plugged together in a flow
  - Use the **HTTP Listener** as an inbound endpoint to trigger a flow with an HTTP request
  - Use the **Set Payload** transformer to set the payload
  - Use the **Database** connector to connect to JDBC databases
  - Use DataWeave and the **Transform Message** component to transform messages from one data type and structure to another

Summary: API design-to-implementation

MuleSoft

- Create RESTful interfaces for applications
  - **Manually** by creating flows with listeners for each resource/method pairing
  - **Automatically** using Anypoint Studio and **APIkit**
- Connect web service interfaces to implementations using the **Flow Reference** component to pass messages to other flows

Mule 4 resources

MuleSoft

- Studio 7 and Mule 4 download page
  - www.mulesoft.com/lp/dl/mule-studio-beta
- Mule 4 EA docs
  - mule4-docs.mulesoft.com
- Blogs
  - blogs.mulesoft.com/result/?as_q=Mule+4
- Webinars
  - www.mulesoft.com/demo/beta/mule-4