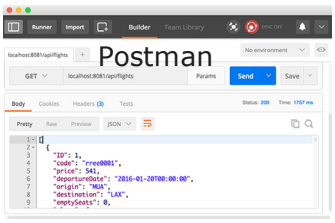
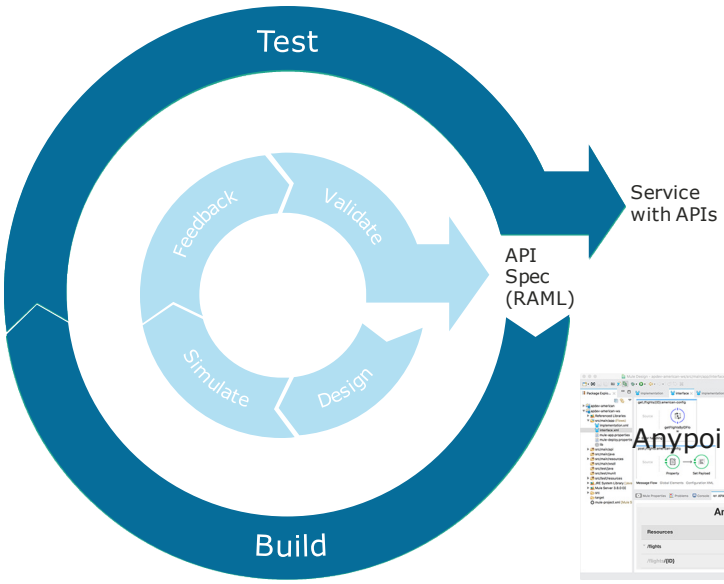
MuleSoft

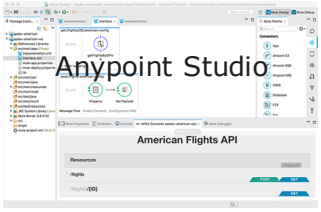
Module 3: Building APIs

Goal

MuleSoft







Objectives

MuleSoft

- Introduce Mule applications, flows, messages, and message processors
- Use Anypoint Studio to create a flow graphically
- Build, run, and test a Mule application
- Use a connector to connect to a database
- Use the graphical DataWeave editor to transform data
- Create a RESTful interface for an application from a RAML file
- Connect an API interface to the implementation

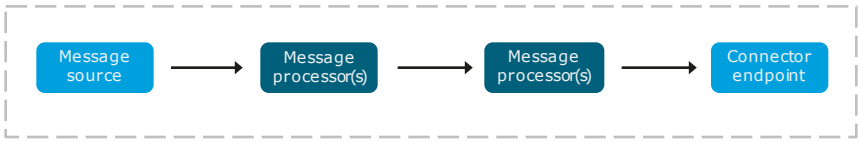
Introducing Mule applications

Mule applications

MuleSoft

- Accept and process **messages** through a series of **message processors** plugged together in a **flow**
  - A message can be initiated by an event like
    - A consumer request from a mobile device
    - A change to data in a database
    - The creation of a new customer ID in a SaaS application

Flow



Mule applications

MuleSoft

- Are written in XML (primarily)
  - Under the hood, are Java applications using Spring
- Can be created and tested visually with Anypoint Studio
  - Available as a stand-alone or as an Eclipse plug-in
- Are deployed to a Mule runtime

Mule runtime

MuleSoft

- A JVM server that
  - Can handle many concurrent requests for different Java (Mule) applications in a single JVM
  - Decouples point-to-point integrations by having all (non-Mule) applications talk to the bus (to a Mule runtime) instead of directly to each other
  - Enforces policies for API governance

Mule runtime editions

MuleSoft

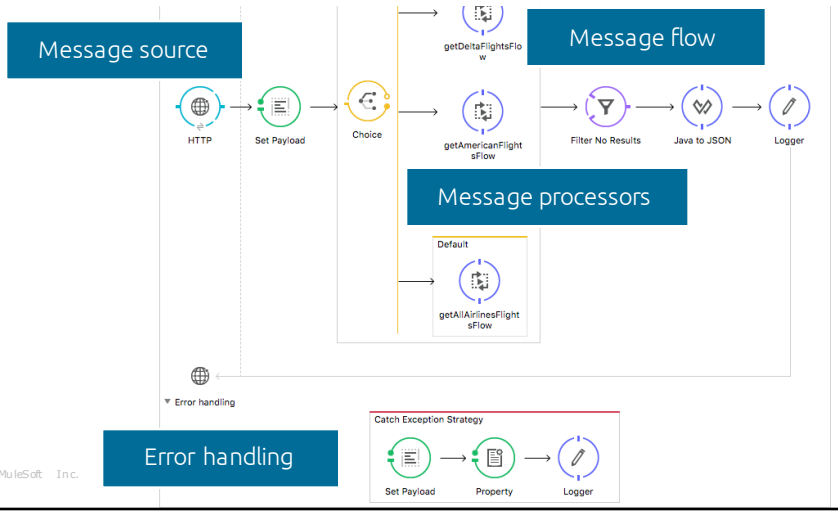
- Enterprise edition (EE) and community edition (CE)
- CE is open-source
- EE is a hardened code line with support and additional capabilities
- By default, Anypoint Studio uses EE
  - You can install other versions and select which one to use
- http://www.mulesoft.com/platform/soa/mule-esb-enterprise

Mule runtime EE

- 24/7 global support
- Additional connectors
- ✧Visual debugging
- ✧DataWeave and DataSense
- ✧Batch module
- Caching and transaction support
- Performance monitoring
- Security module
- Templates
- ✧Deployment and performance management

Mule applications and flows

- Mule applications accept and process **messages** through a series of **message processors** plugged together in a **flow**



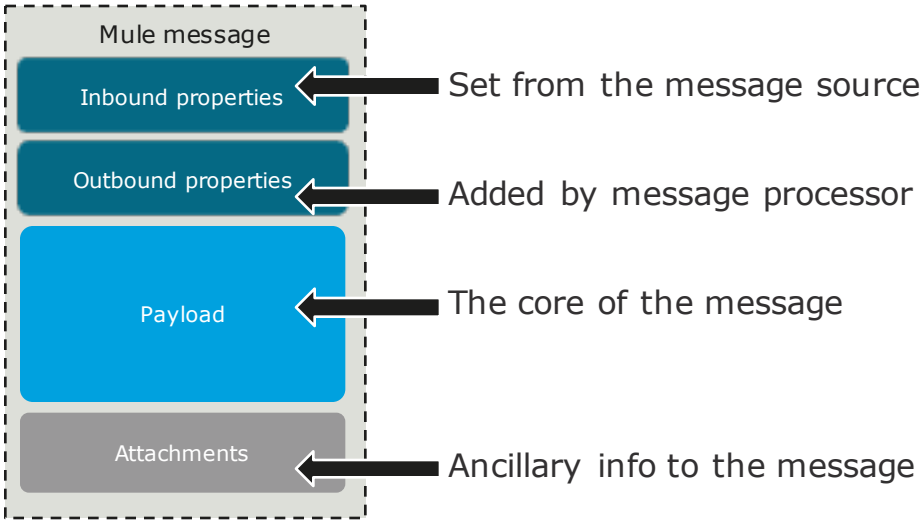
Mule flows

MuleSoft

- A typical flow has
  - A message source
    - Accepts a message from an external source triggering the execution of the flow
  - Message processors
    - Transform, filter, enrich, and process the message
- An application can consist of
  - A single flow
  - Multiple flows
  - Multiple flows connected together

Mule messages

MuleSoft



Creating Mule applications with Anypoint Studio

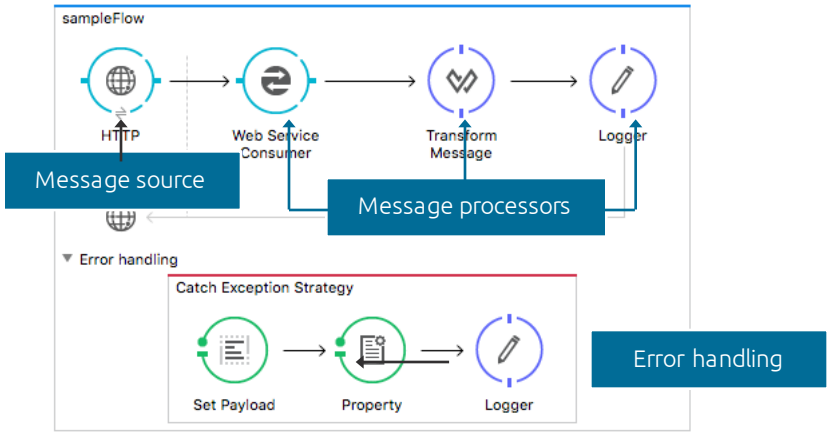
Creating Mule applications with Anypoint Studio

MuleSoft

- Anypoint Studio is an Eclipse-based integration development environment
  - Two-way editing between graphical and XML views
  - 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

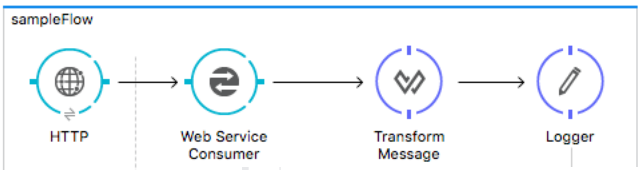
Anatomy of a flow: Visual

MuleSoft



Anatomy of a flow: XML

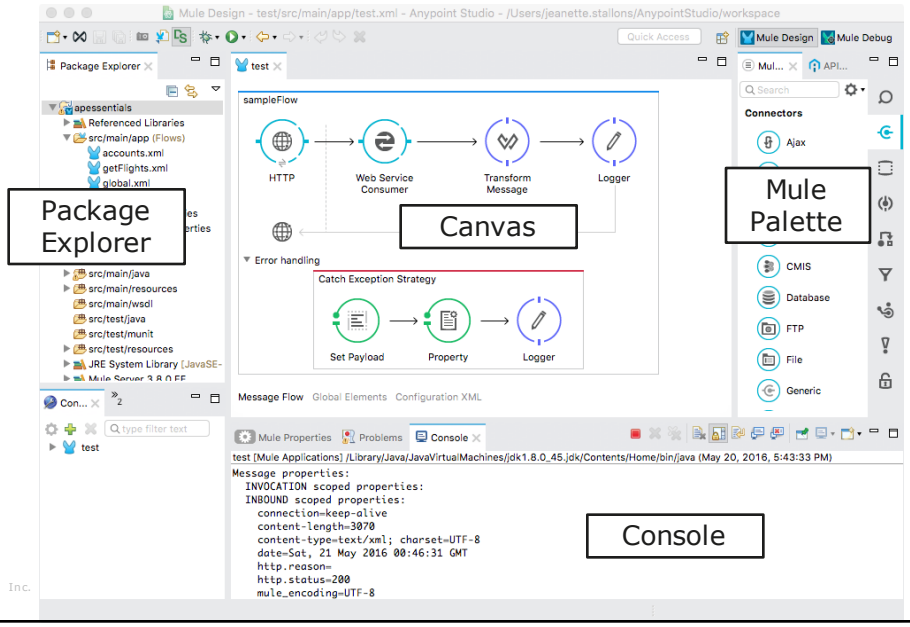
MuleSoft




```
15
16    <flow name="sampleFlow">
17        <http:listener config-ref="HTTP_Listener_Configuration" path="/" do
18        <ws:consumer config-ref="Web_Service_Consumer" doc:name="Web Servic
19        <dw:transform-message doc:name="Transform Message">
20            <dw:set-payload ><![CDATA[%dw 1.0
21 %output application/json
22 ---
23 payload]]></dw:set-payload>
24        </dw:transform-message>
25        <logger level="INFO" doc:name="Logger"/>
26        <catch-exception-strategy doc:name="Catch Exception Strategy">
27            <set-payload doc:name="Set Payload" value="Data unavailable"/>
28            <set-property doc:name="Property" propertyName="http.status" va
29            <logger level="INFO" doc:name="Logger"/>
30        </catch-exception-strategy>
31    </flow>
```

Anypoint Studio anatomy

MuleSoft



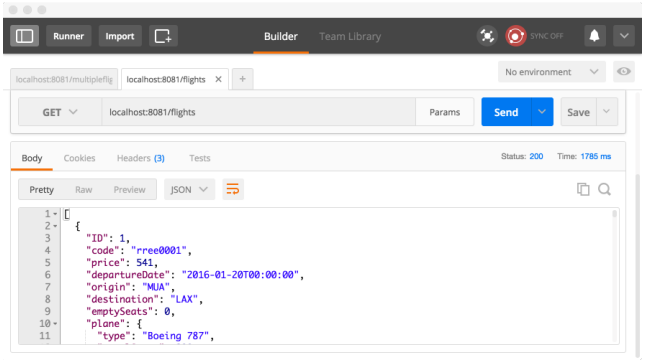
Running applications

MuleSoft

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

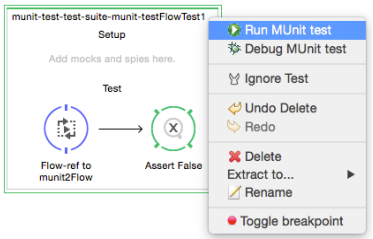
Testing applications by making requests to endpoints

- 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

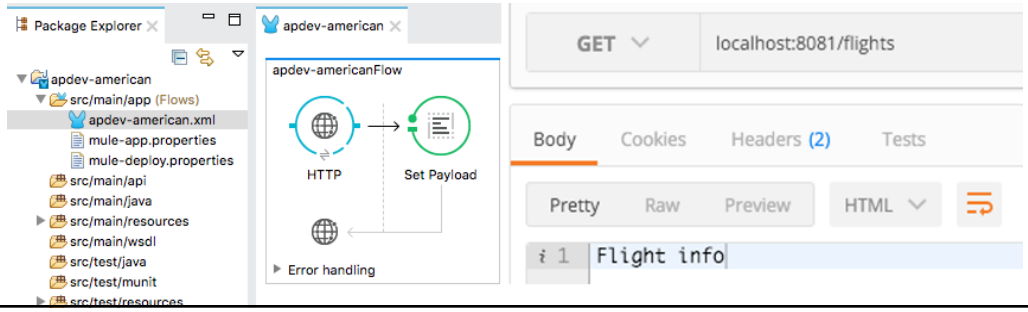


- MUnit is covered in the *Anypoint Platform Development: Advanced* course

Walkthrough 3-1: 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



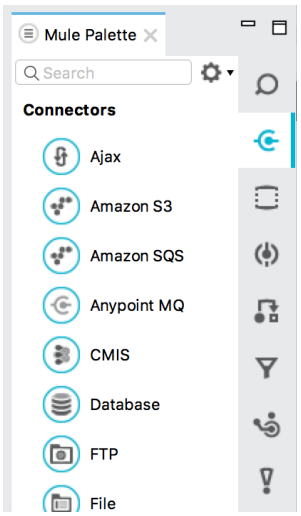
21

Understanding Mule application building blocks

Mule application building blocks

MuleSoft

- Are separated into categories in the Mule Palette



Connectors
Scopes
Components
Transformers
Filters
Flow Control
Error Handling

Message sources

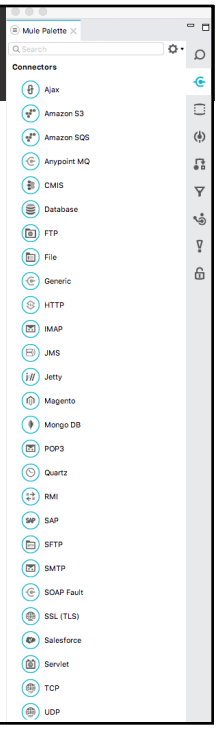
MuleSoft

- The first building block of most flows is a receiver that receives new messages and places them in the queue for processing
- Message sources are usually Anypoint Connectors

- Connectors provide connectivity to external resources
  - Such as databases, protocols, or APIs
  - Standard protocols like HTTP, FTP, SMTP, AMQP
  - Third-party APIs like Salesforce, Twitter, or MongoDB

Anypoint Connectors

- Anypoint Platform has over 120 pre-built connectors
  - 30 bundled with Anypoint Studio
  - Additional ones available in the Anypoint Exchange
    - Accessible directly from Anypoint Studio or at https://www.mulesoft.com/exchange
- There are 2 main types
  - Endpoint-based connectors
  - Operation-based connectors



Endpoint-based connectors

MuleSoft

- Are either inbound or outbound endpoints in a flow
- Inbound endpoints serve as a message source for a flow
- Outbound endpoints send information to external systems
  - Can occur mid-flow or at the end of flows (or at the beginning in a poll)

Operation-based connectors

- Require the specification of an operation for that connector to perform
- Includes most connectors not based on a standard communication protocol

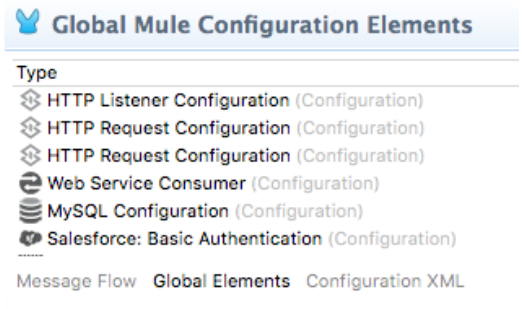
Terminology: Connector vs endpoint

- A **connector** is a Mule-specific connection to an external resource of any kind
- An **endpoint** is a flow-level element that is configured to receive and/or send messages from and/or to external resources

Connectors and endpoints and global elements

MuleSoft

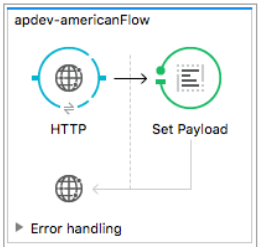
- When you drag a connector from the Mule Palette, an endpoint is created
- For most endpoints, a lot of the configuration is encapsulated in a separate global element
  - A reusable object that can be used by many endpoints
  - Defines a connection to a network resource
- This is a connector configuration
  - Though it is sometimes referred to simply as the connector



HTTP connector

MuleSoft

- Can send and receive HTTP and HTTPS requests over a selected host, port, and address
- Can be either a listener or a requester depending upon where you add it in a flow



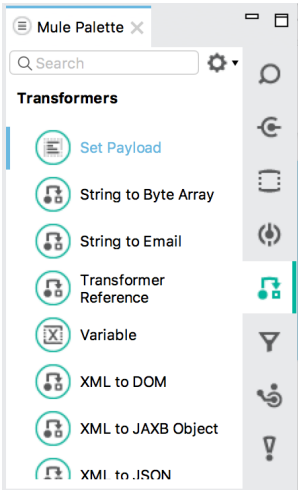
- HTTP Listener connector (inbound)
  - Listens for requests that arrive at a certain address and provides an HTTP response
  - By default, host is set to 0.0.0.0
    - A shortcut to simultaneously listen on all active IP addresses (including localhost)
    - For apps deployed to the cloud, be sure to leave this value in connectors so requests for your application domain URL are routed to the endpoints
- HTTP Request connector (outbound)
  - Sends HTTP requests to a certain address and receives the returned response

Message processors

MuleSoft

- Connectors
  - Connect to an external resources

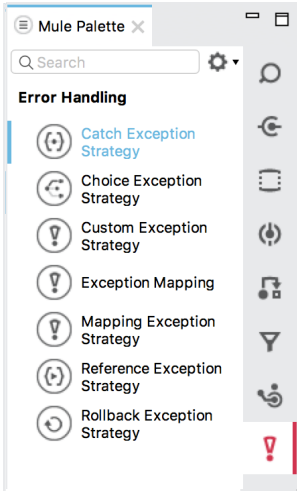
- Scopes
  - Wrap snippets of code to define fine-grained behavior within a flow
- Components
  - Execute specific logic upon a message, including custom-logic in Java, JavaScript, Groovy, Python or Ruby
- Transformers
  - Modify the values of part of a message



Message processors

MuleSoft

- Filters
  - Limit processing of messages based on set criteria
- Flow control
  - Direct messages through different pathways in an application depending upon content or other criteria
- Error handling
  - Handle any errors that occur during message processing

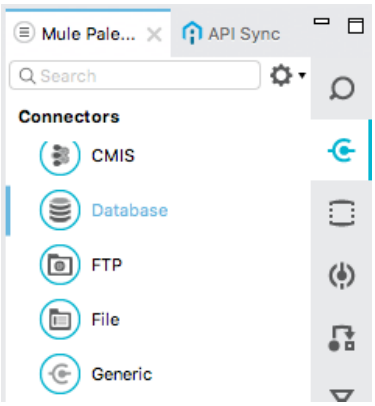


Connecting to data

The Database connector

MuleSoft

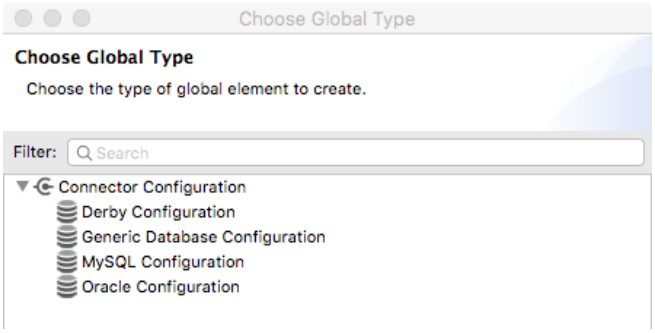
- Can connect to almost any JDBC relational database
  - Any database engine for which you have a driver
- Supports operations including
  - SELECT, INSERT, UPDATE, DELETE
  - Stored Procedures
  - Bulk Execute
  - Data Definition Language (DDL) requests like CREATE and ALTER



Supported database engines

MuleSoft

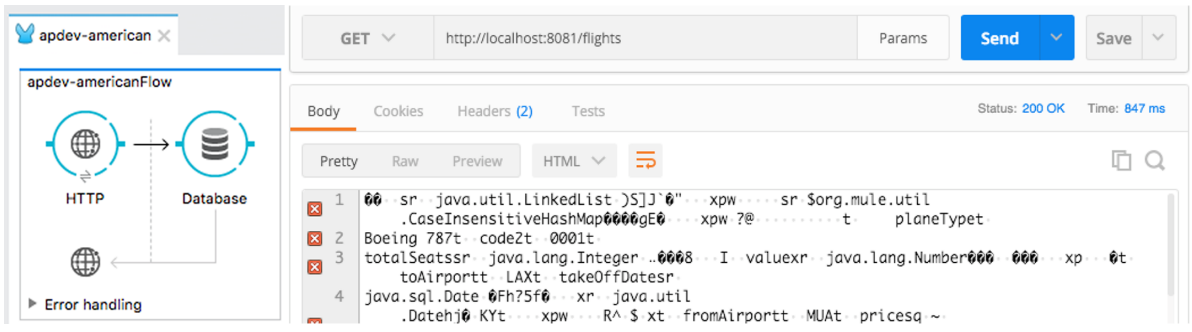
- Out-of-the-box support for
  - MySQL, Oracle, Derby
- Others are supported by a generic database configuration
  - Specify the driver class as one of the connection parameters



Walkthrough 3-2: Connect to data (MySQL database)

MuleSoft

- Add a Database 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

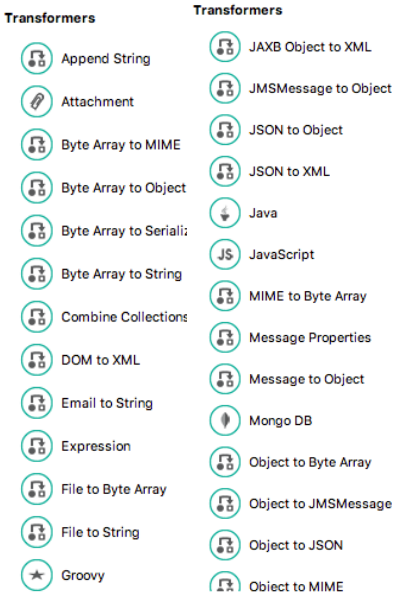


Transforming data

Transforming data using transformers

MuleSoft

- Anypoint Platform provides a set of transformers to handle the most common data transformation scenarios
- Up through Mule 3.6, this was the main way to transform messages



Complex transformations

MuleSoft

- Up through Mule 3.6, complex transformations were handled by
  - Chaining transformers
  - Creating a custom Java transformer
  - Using a Script transformer to write the transformation in Groovy, JavaScript, Python, or Ruby
  - Using the DataMapper transformer
    - Which had a graphical interface to work with many different data formats

Transforming data using DataWeave

MuleSoft

- A new way to transform data was introduced in Mule 3.7
- **DataWeave** is a full-featured and fully native framework for querying and transforming data on Anypoint Platform

- Powered by the DataWeave data transformation language
  - A JSON-like language that's built just for data transformation use cases
- Powered by the core Mule runtime
  - Provides 5x performance vs previous approaches (DataMapper)
- Fully integrated with Anypoint Studio and DataSense
  - Graphical interface with payload-aware development

DataWeave data transformation use cases

MuleSoft

- DataWeave was purposefully built to make it easy to write simple to complex transformations
  - Simple 1-to-1 mappings
  - Transforming hierarchical data models
  - De-duplication of data
  - Filtering data
  - Grouping and partitioning data
  - Joining data across multiple data sources
  - Streaming inbound and outbound data

Using DataWeave for all transformations

MuleSoft

- DataWeave can be used for all your transformations
  - From simple to complex
  - No longer need to use most other transformers unless you want to use specific Java frameworks
    - Like JAXB, Jackson, org.w3c.dom
    - To integrate with existing code bases or leverage existing skill sets

DataWeave under the hood

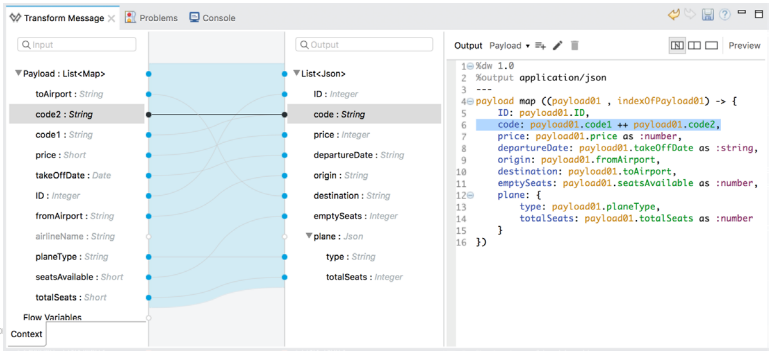
MuleSoft

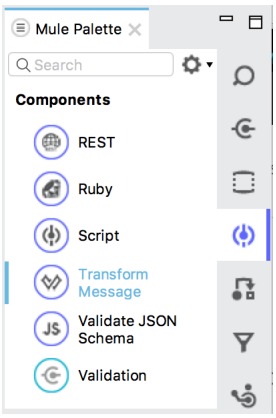
- Underneath, DataWeave includes a connectivity layer and engine that is fundamentally different from other transformation technologies
- It contains a data access layer that indexes content and accesses the binary directly, without costly conversions
  - Enables larger than memory payloads
  - Random access to input documents
  - Very high performance

DataWeave integration with Anypoint Studio

MuleSoft

- The DataWeave Transform Message component provides an Anypoint Studio interface
- The Properties view has input, output, and transform, sections with both drag-and-drop and code editors

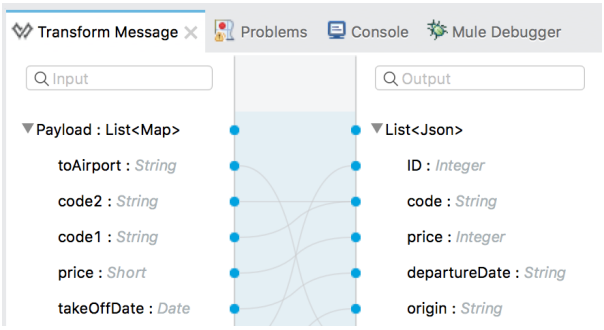




DataWeave integration with DataSense

MuleSoft

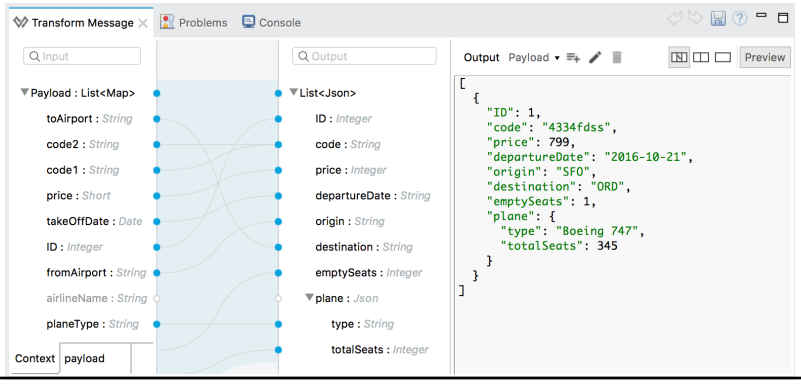
- DataWeave is fully integrated with DataSense allowing payload-aware development
- Metadata from connectors, schemas, and sample documents can be used to more easily build transformations
  - Displayed in input and output sections for incoming and outbound messages
  - Used for drag-and-drop in the graphical editor
  - Used for code auto-completion in the code editor



Walkthrough 3-3: Transform data

MuleSoft

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



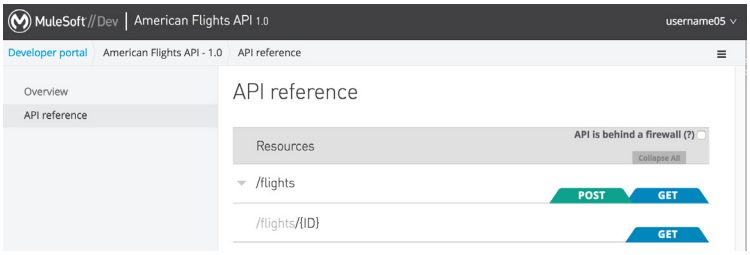
46

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

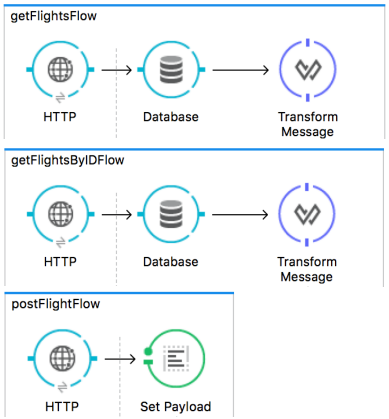


- You can create the interface manually or have it generated from the API definition
  - We will do both

Creating RESTful interfaces manually

MuleSoft

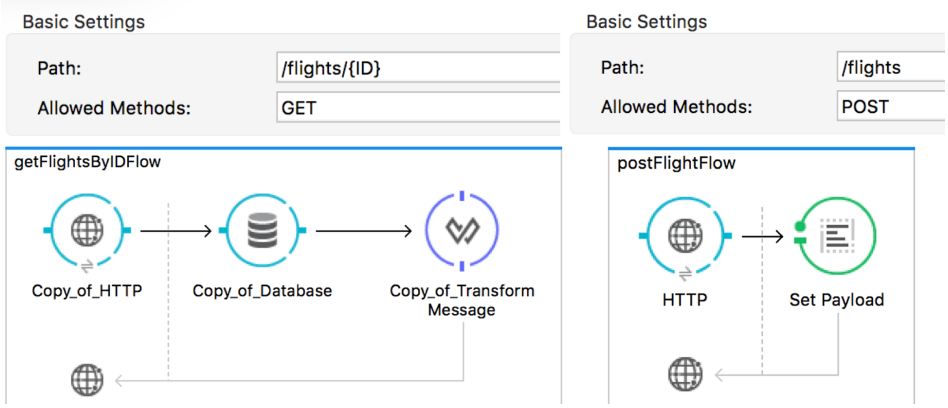
- Create a flow with a listener for each resource / method pair
  - GET: /flights
  - GET: /flights/{ID}
  - POST: /flights
- Or create a flow with a listener that receives all messages and routes them based on URI parameters and HTTP method
  - Path: /* (use a wildcard)
  - Allowed methods: Do not specify



Walkthrough 3-4: 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



Generating RESTful interfaces automatically using APIkit

Introducing APIkit

MuleSoft

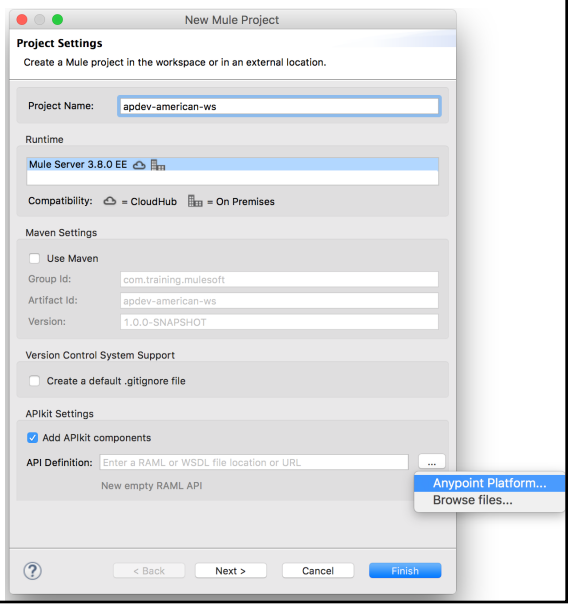
- APIkit is an open-source toolkit that includes
  - A Router element for Mule applications
    - Used in an API implementation
    - Routes inbound API requests and validates requests against RAML
  - A Proxy element for Mule applications
    - Used in a proxy application, not an API implementation
    - Validates requests against RAML
    - Passes valid requests on to a separate API implementation
  - APIkit Mapping Exception Strategy
  - APIkit Anypoint Studio plugin

Creating RESTful interfaces automatically

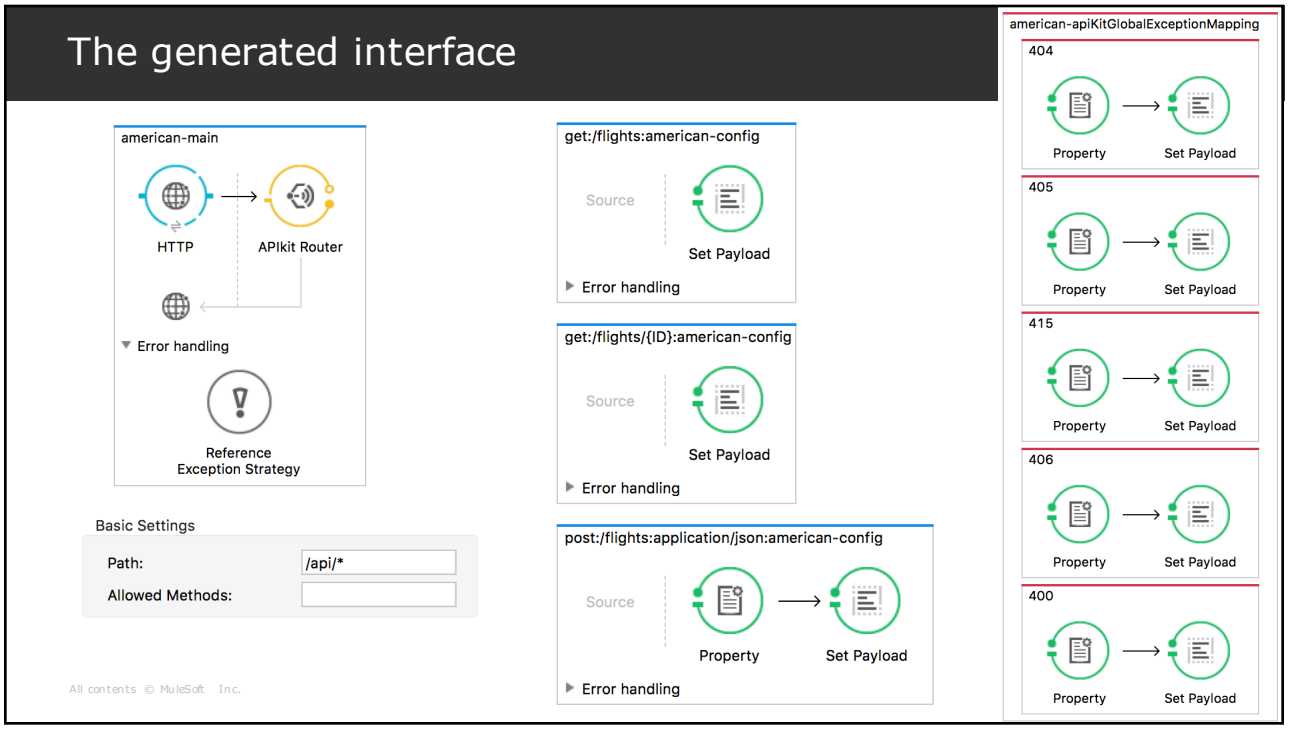
- The Anypoint Studio APIkit plugin can generate an interface automatically from a RAML API definition
- It generates a main routing flow (that uses APIkit Router) and flows for each of the resource / method pairs
- You add processors to the resource flows (usually Flow References) to hook up to your backend logic

Using APIkit to create RESTful interfaces

- In an existing project
  - Add a RAML file to the project
  - Right-click and select Mule > Generate Flows from RAML
- In a new project
  - Specify a RAML file when you create the project
  - Can be local or on Anypoint Platform



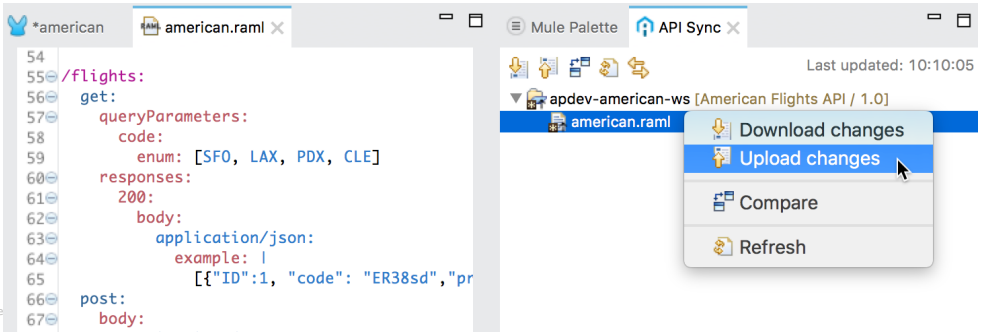
The generated interface



A seamless API design-to-implementation experience

MuleSoft

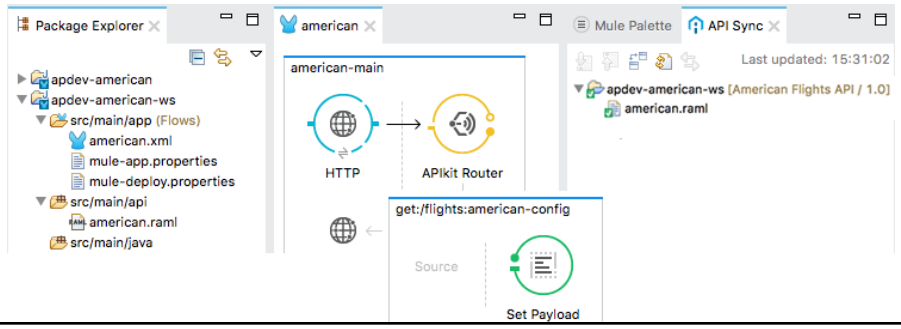
- When you create a project with APIkit components and a RAML file on Anypoint Platform, a connection is made to that RAML file
  - The remote RAML is polled periodically and compared to the local version
- Use the APISync view to pull, push, and compare between Anypoint Studio and API Designer



Walkthrough 3-5: use Anypoint Studio to create a RESTful API interface from a RAML file

MuleSoft

- Add Anypoint Platform credentials to Anypoint Studio
- Add a RAML file from Anypoint Platform to an Anypoint Studio project
- Use Anypoint Studio and APIkit to generate a RESTful web service interface from a RAML file
- Test the web service in the APIkit Consoles view and Postman



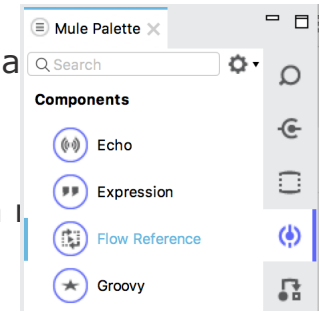
57

Connecting the interface to the implementation

Passing messages to other flows

MuleSoft

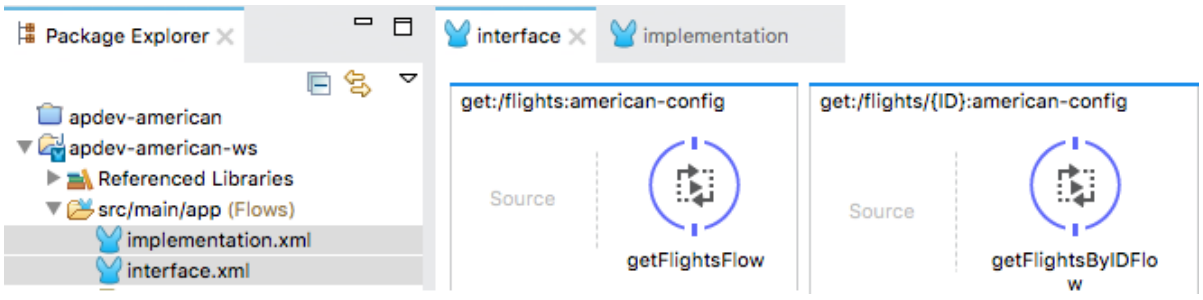
- Flows can be broken into multiple flows
  - Makes the graphical view more intuitive and the XML code easier to read
  - Promotes code reuse
- All flows are identified by name and can be called via components in other flows
- Note: In the developer courses, you will learn much flows and subflows in a later module



Walkthrough 3-6: Implement a RESTful web service

MuleSoft

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



Summary

Summary: Mule applications

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
- Mule messages have inbound properties, outbound properties, a payload, and attachments
- Message processors include connectors, scopes, components, transformers, filters, flow control, and error handling elements

Summary: Application building blocks

- Connectors are inbound/outbound and endpoint/operation based
- When you drag out a connector, an endpoint is created
- For most endpoints, a lot of the configuration is encapsulated in a separate, reusable global element
- 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

- 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
- For a seamless API design-to-implementation experience
  - Create a project with APIkit components and a RAML file on Anypoint Platform
  - Use the APISync view to pull, push and compare between Anypoint Studio and API Designer