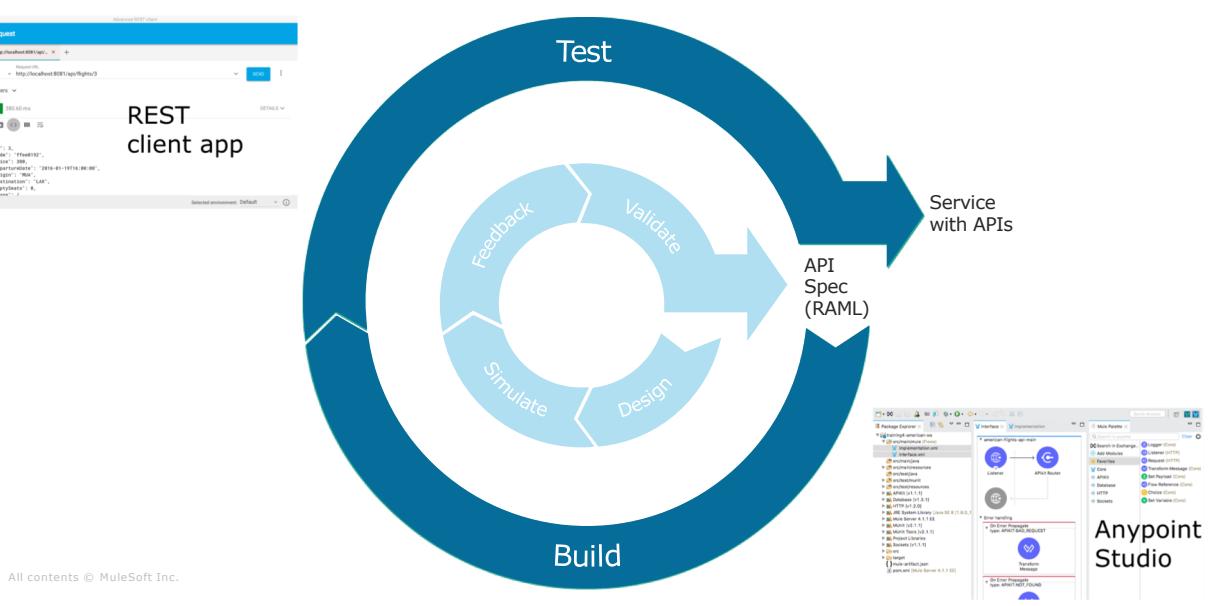




Module 4: Building APIs



Goal



All contents © MuleSoft Inc.

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



- 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

All contents © MuleSoft Inc.

3

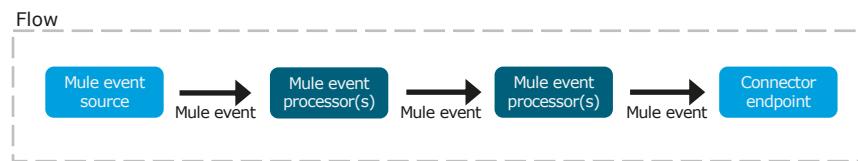
Reviewing Mule 4 applications



Review: Mule 4 applications and flows



- Mule applications receive events, process them, and route them to other endpoints
- Mule applications accept and process a Mule event through a series of Mule event processors plugged together in a flow with
 - A **Mule event source** that initiates the execution of the flow
 - **Mule event processors** that transform, filter, enrich, and process the event and its message



All contents © MuleSoft Inc.

5

Review: Mule 4 event structure



- The data that passes through flows in the app
- Metadata contained in the message header
- The core info of the message - the data the app processes
- Metadata for the Mule event - can be defined and referenced in the app processing the event

All contents © MuleSoft Inc.

6

Creating Mule applications with Anypoint Studio



Introducing Anypoint Studio



- Based on Eclipse, a common Java integrated development environment
- Features include
 - Two-way editing between graphical and XML views
 - Pre-built tooling to connect to APIs (REST, SOAP), protocols (HTTP, FTP, SMTP, more), and popular services (Salesforce, Workday, more!)
 - A data transformation framework and language
 - An embedded Mule runtime to test applications without leaving it
 - Visual debugging
 - One-click deployment of applications to CloudHub
 - Templates for common integration patterns
 - Integration with Maven for continuous build processes

Anypoint Studio anatomy

The screenshot shows the Anypoint Studio interface with several labeled components:

- Package Explorer**: Located on the left, showing the project structure for "apdev-flights-ws".
- Canvas**: The central workspace where Mule flows are designed.
- Mule Palette**: A palette on the right containing various Mule components and modules.
- Console**: A window at the bottom showing Mule runtime logs.

All contents © MuleSoft Inc. 9

Anatomy of a flow: Visual

The diagram illustrates a visual flow named `getFlightsFlow`:

- Event source**: Represented by a Listener icon labeled `Listener GET flights`.
- Event processors**: A sequence of components:
 - A database icon labeled `Select All flights`.
 - A transform icon labeled `Transform Message JSON to JSON`.
 - A logger icon labeled `Logger`.
- Error handling**: A separate section containing an `On Error Propagate` component with two steps: `Set Payload` and `Set Variable`.

All contents © MuleSoft Inc. 10

Anatomy of a flow: XML



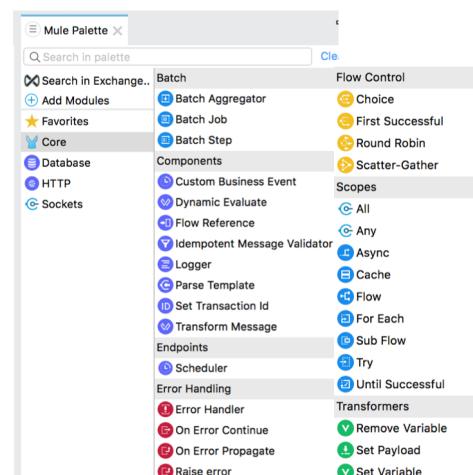
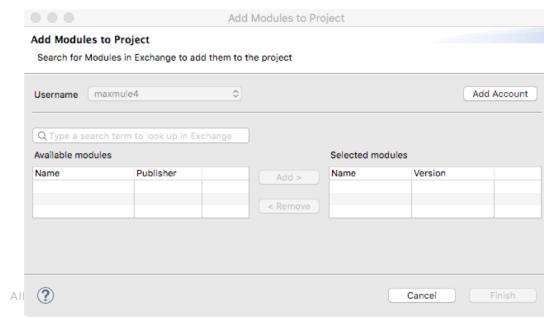
All contents © MuleSoft Inc.

11

Mule application building blocks



- Are separated into categories in the Core section of the Mule Palette
 - By default, projects include HTTP and Sockets modules
 - Can add additional modules



12

Running applications



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

```

Mule Properties Problems Console X
training4-american-ws [Mule Applications] Mule Server 4.1.1 EE
*****
* Started app 'training4-american-ws'
* Application plugins:
* - Sockets
* - HTTP
*****
INFO 2018-04-18 07:38:48,160 [WrapperListener.start_runner] com.mulesoft.agent.configuration.postconfigure.DefaultPostConfig
INFO 2018-04-18 07:38:48,161 [WrapperListener.start_runner] com.mulesoft.agent.configuration.postconfigure.DefaultPostConfig
INFO 2018-04-18 07:38:48,163 [WrapperListener.start_runner] org.mule.runtime.module.deployment.internal.DeploymentDirectoryW
+-----+
+ Mule is up and kicking (every 5000ms) +
+-----+
INFO 2018-04-18 07:38:48,173 [WrapperListener.start_runner] org.eclipse.jetty.server.AbstractConnector: Started ServerConnec
INFO 2018-04-18 07:38:48,174 [WrapperListener.start_runner] org.mule.runtime.module.deployment.internal.StartupSummaryDeploy
+-----+
* - + DOMAIN + - - * - - + STATUS + - - *
* default * DEPLOYED *
+-----+
* - - + APPLICATION + - - * - - + DOMAIN + - - * - - + STATUS + - - *
* training4-american-ws * default * DEPLOYED *
+-----+

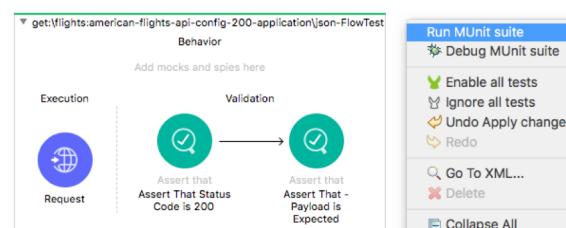
```

All c 13

Automating testing of applications



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



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

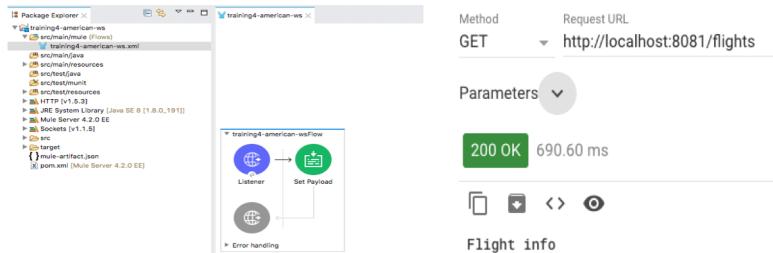
All contents © MuleSoft Inc.

14

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



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



All contents © MuleSoft Inc.

15

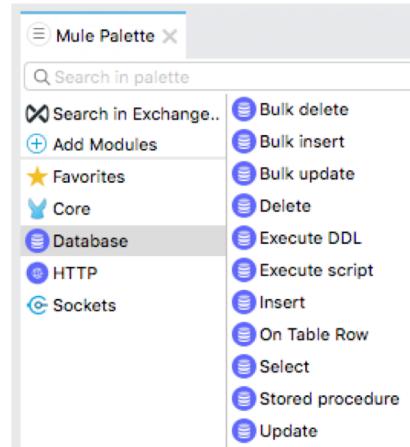
Connecting to data



The Database connector



- Can connect to almost any JDBC relational database
 - Any database engine for which you have a driver
- To use
 - Add the Database module to your project
 - Add a database operation to a flow
 - Configure the connection to the database



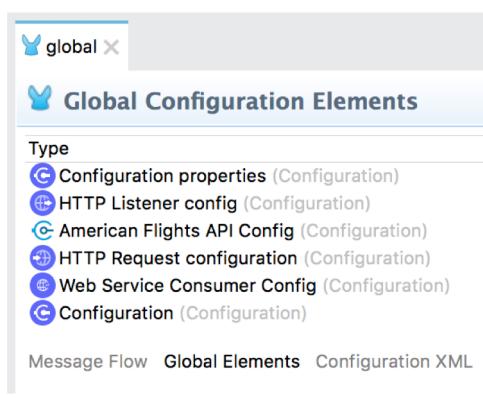
All contents © MuleSoft Inc.

17

Global configuration elements



- For most operations, a lot of the configuration is encapsulated in a separate global element
 - A reusable configuration that can be used by many operations
 - Defines a connection to a network resource
- This is a connector configuration
 - Though it is sometimes referred to simply as the connector

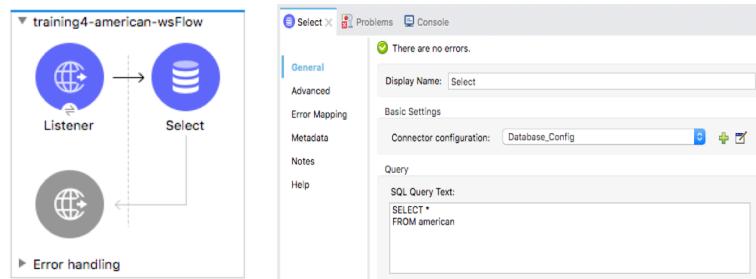


All contents © MuleSoft Inc.

18

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

- Add a Database Select operation
- 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 Select operation to use that Database connector
- Write a query to select data from a table in the database



All contents © MuleSoft Inc.

19

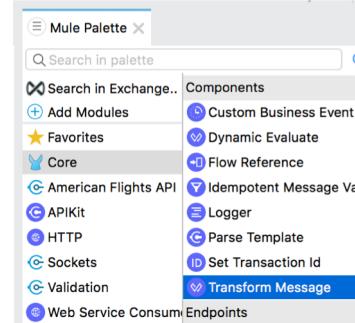
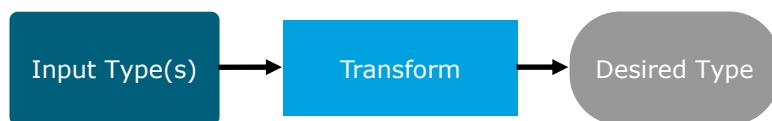
Transforming data



Transforming data



- **DataWeave 2.0** is the expression language for Mule to access, query, and transform **Mule 4** event data
 - DataWeave was introduced and used in Module 2
- In Studio, use **Transform Message** component for transformations
 - Graphical interface with payload-aware development



All contents © MuleSoft Inc.

21

Walkthrough 4-3: Transform data



- Use the Transform Message component
- Use the DataWeave visual mapper to change the response to a different JSON structure

The screenshot shows the 'Transform Message' component in Mule Studio. The 'Payload' tab is selected. On the left, the 'list_object.dwl' script is shown:

```

[{"ID": 2, "code": "4334fdss", "price": 2, "departureDate": "2003-10-01", "origin": "SFO", "destination": "ORD", "emptySeats": 2, "plane": {"type": "Boeing 747", "totalSeats": 2}}
]
  
```

On the right, the resulting JSON output is displayed:

```

{
  "ID": 2,
  "code": "4334fdss",
  "price": 2,
  "departureDate": "2003-10-01",
  "origin": "SFO",
  "destination": "ORD",
  "emptySeats": 2,
  "plane": {
    "type": "Boeing 747",
    "totalSeats": 2
  }
}
  
```

All contents © MuleSoft Inc.

22

Creating RESTful interfaces manually for Mule applications



Creating RESTful interfaces



- A RESTful interface for an application will have listeners for each resource / method pairing defined by the API

- GET: /flights
 - POST: /flights
 - GET: /flights/{ID}
 - DELETE: /flights/{ID}
 - PUT: /flights/{ID}

/flights
GET POST

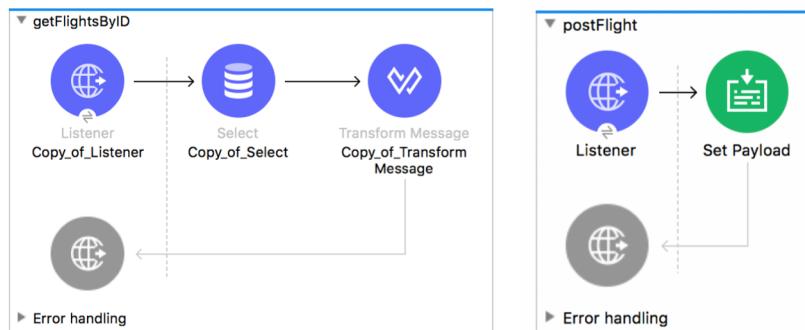
/flights/{ID}
GET DELETE PUT

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

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



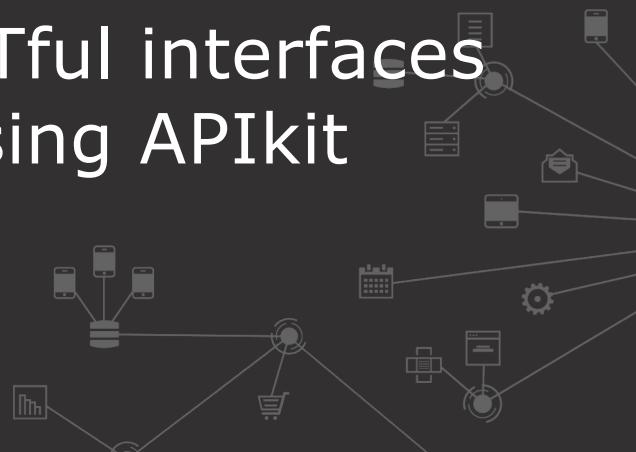
- Route based on path
- Use a URI parameter in the path of a new HTTP Listener
- Route based on HTTP method



All contents © MuleSoft Inc.

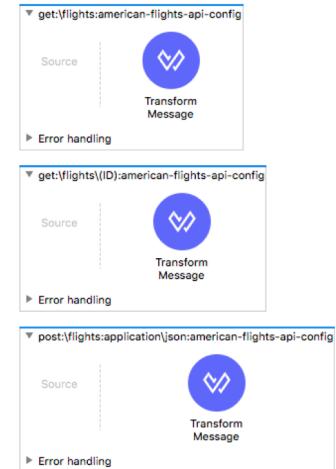
25

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



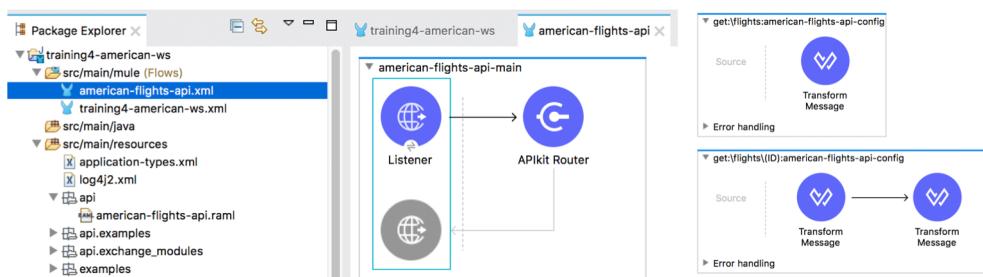
All contents © MuleSoft Inc.

27

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 Exchange into an Anypoint Studio project
- Use APIkit to generate a RESTful web service interface from an API
- Test a web service using APIkit console and Advanced REST Client



All contents © MuleSoft Inc.

28

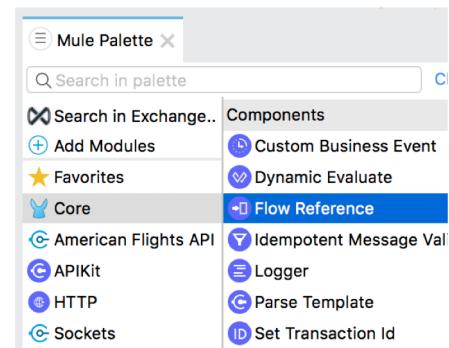
Connecting interfaces to implementations



Passing messages to other flows



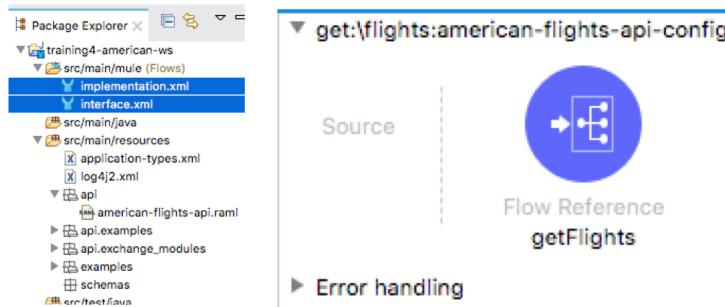
- Flows can be broken into multiple flows
 - Makes the graphical view more intuitive and the XML code easier to read
 - Promotes code reuse
- All flows are identified by name and can be called via **Flow Reference** components in other flows



Walkthrough 4-6: Implement a RESTful web service

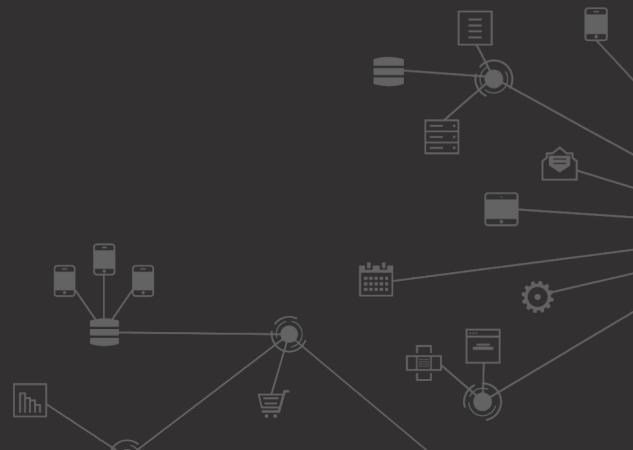


- Pass an event from one flow to another
- Call the backend flows
- Create new logic for the nested resource call
- Test the web service using Advanced REST Client



39

Summary



Summary



- **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 events through a series of event 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



- 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