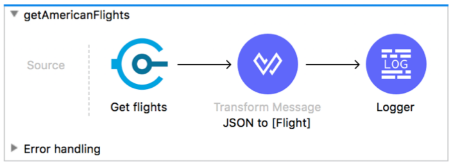
MuleSoft

Module 8: Consuming Web Services

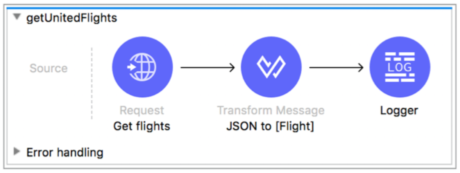
Goal

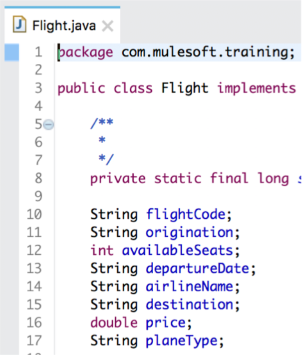
MuleSoft

Call an operation of a API in Exchange



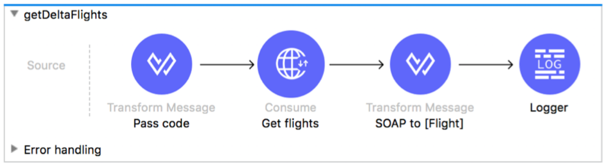
Call a RESTful web service

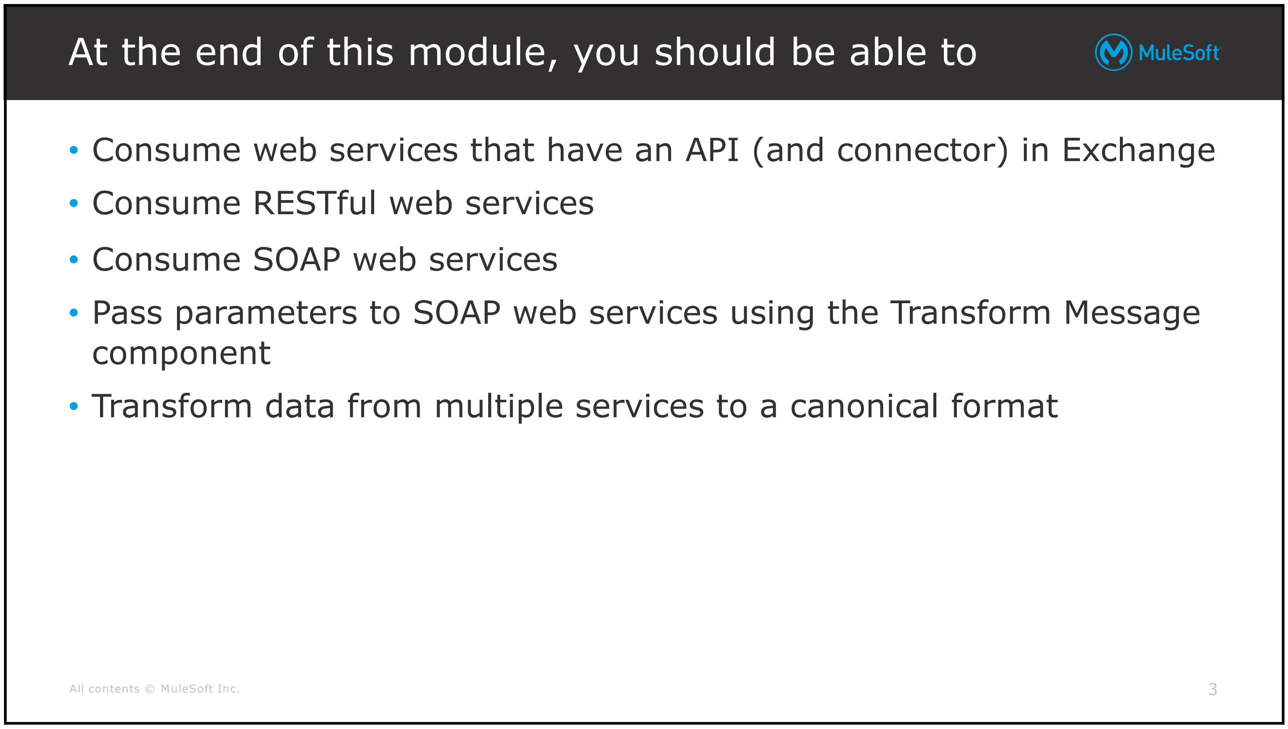




```
package com.mulesoft.training;

public class Flight implements

    /**
     *
     */
    private static final long

    String flightCode;
    String origination;
    int availableSeats;
    String departureDate;
    String airlineName;
    String destination;
    double price;
    String planeType;
```

Call a SOAP web service





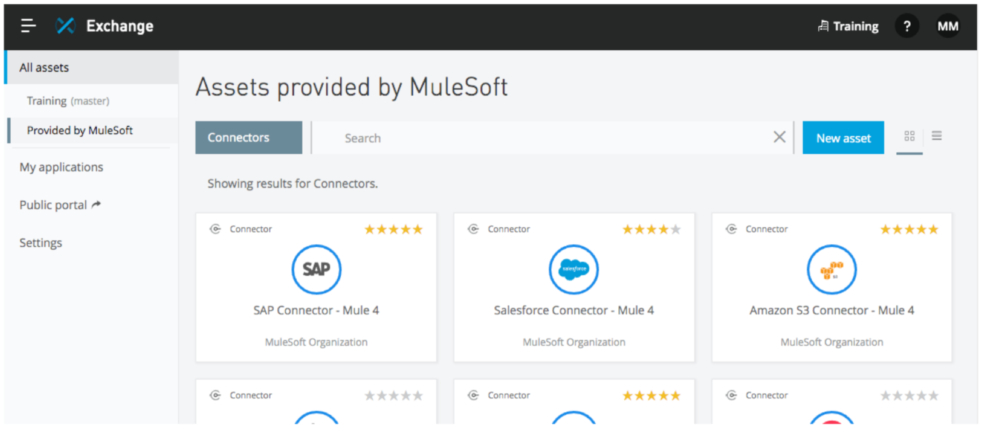
At the end of this module, you should be able to

- Consume web services that have an API (and connector) in Exchange
- Consume RESTful web services
- Consume SOAP web services
- Pass parameters to SOAP web services using the Transform Message component
- Transform data from multiple services to a canonical format

Consuming web services that have an API (and connector) in Exchange

Connectors in Anypoint Exchange

- Many connectors in Exchange package an easy way to make calls to APIs



Assets provided by MuleSoft

Showing results for Connectors.

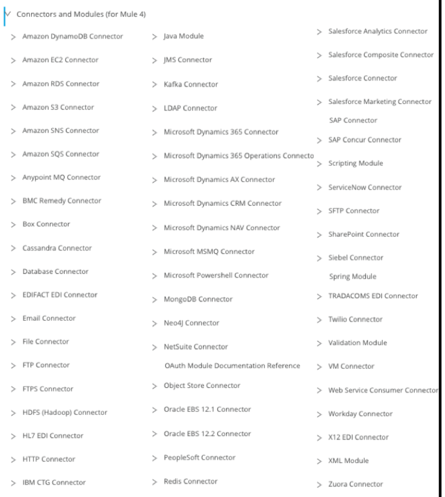
SAP Connector - Mule 4
MuleSoft Organization

Salesforce Connector - Mule 4
MuleSoft Organization

Amazon S3 Connector - Mule 4
MuleSoft Organization

Connectors and modules

- **Modules** are extensions to the Mule runtime that you can use when building a Mule app
  - HTTP, Database, Salesforce, SAP, Slack, Validation, Java, and many more
- **Connectors** are modules that connect to an external server
  - HTTP, Database, Salesforce, SAP, Slack
- For module reference
  - https://docs.mulesoft.com/connectors/

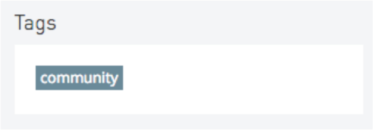


Connectors and Modules (for Mule 4)

- Amazon DynamoDB Connector
- Amazon EC2 Connector
- Amazon RDS Connector
- Amazon S3 Connector
- Amazon SNS Connector
- Amazon SQS Connector
- Anypoint MQ Connector
- BMC Remedy Connector
- Box Connector
- Cassandra Connector
- Database Connector
- EDIFACT EDI Connector
- Email Connector
- File Connector
- FTP Connector
- FTPS Connector
- HDFS (Hadoop) Connector
- HL7 EDI Connector
- HTTP Connector
- IBM CTG Connector
- Java Module
- JMS Connector
- Kafka Connector
- LDAP Connector
- Microsoft Dynamics 365 Connector
- Microsoft Dynamics 365 Operations Connecto
- Microsoft Dynamics AX Connector
- Microsoft Dynamics CRM Connector
- Microsoft Dynamics NAV Connector
- Microsoft MSMQ Connector
- Microsoft Powershell Connector
- MongoDB Connector
- Neo4J Connector
- NetSuite Connector
- OAuth Module Documentation Reference
- Object Store Connector
- Oracle EBS 12.1 Connector
- Oracle EBS 12.2 Connector
- PeopleSoft Connector
- Redis Connector
- Salesforce Analytics Connector
- Salesforce Composite Connector
- Salesforce Connector
- Salesforce Marketing Connector
- SAP Connector
- SAP Concur Connector
- Scripting Module
- ServiceNow Connector
- SFTP Connector
- SharePoint Connector
- Siebel Connector
- Spring Module
- TRADACOMS EDI Connector
- Twilio Connector
- Validation Module
- VM Connector
- Web Service Consumer Connector
- Workday Connector
- X12 EDI Connector
- XML Module
- Zuora Connector

Connector types specify creator and support level

- The type of selector is specified in its tags on Exchange



	Premium	Select	MuleSoft Certified	Community
Additional cost	x			
Updated APIs	x	x		
Fully tested	x	x		
MuleSoft Support	Tier 1-3	Tier 1-3	Tier 1 (From developer: T2/T3)	Tier 1
Connector examples	HL7 SAP Siebel	Salesforce Workday	AS/400 Oracle JD Edwards Microsoft Azure Storage	LinkedIn Slack

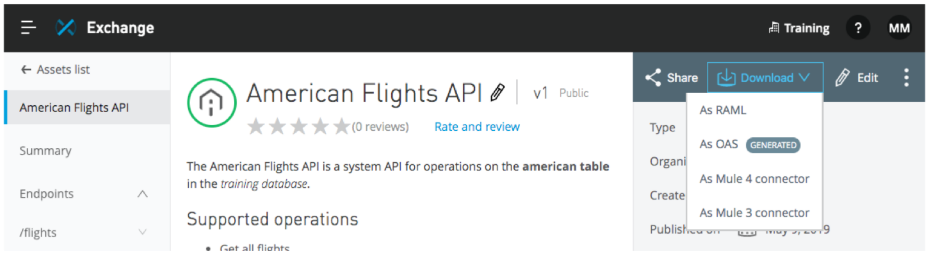
Connector support levels

- Tier 1
  - MuleSoft will isolate the problem and diagnose it
- Tier 2
  - MuleSoft will find a workaround
- Tier 3
  - MuleSoft will fix the code

	Premium	Select	MuleSoft Certified	Community
Not included in Platform license	x			
Tier 2-3 Support	x	x		
Tier 1 Support	x	x	x	x

Connectors are automatically generated for APIs added to Exchange

- **REST CONNECT** is the tool that generates connectors for RESTful APIs
  - Resides in Exchange and automatically invoked when an API is added to Exchange
  - Works for both RAML 1.0 or OAS API specifications added to Exchange
  - Both Mule 3 and Mule 4 connectors are generated
- You can use the connector in Anypoint Studio or Flow Designer
- Download connectors from the download drop-down menu in Exchange



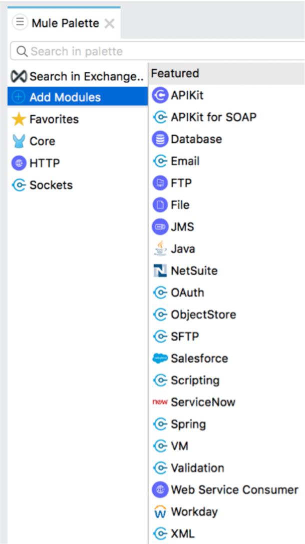
The American Flights API is a system API for operations on the **american table** in the *training database*.

Supported operations

- Get all flights

Connectors in Anypoint Studio

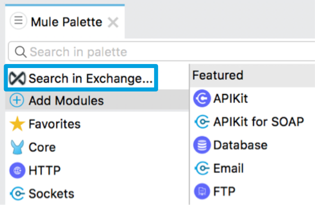
- Some modules are **pre-installed** in Studio
  - HTTP, Database, Salesforce, Validation, Java
- Some modules are **not pre-installed** in Studio
  - SAP, Slack
  - Generated REST connectors

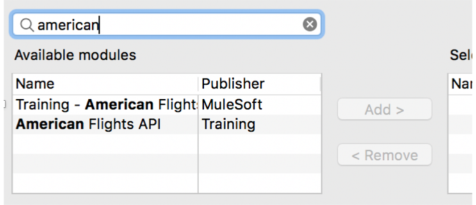


Adding connectors from Exchange

MuleSoft

- If connectors are not pre-installed in Anypoint Studio, you can search Exchange and add them to a project

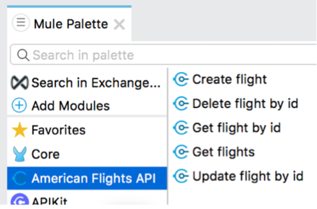




american

Available modules

Name	Publisher
Training - **American** Flight	MuleSoft
American Flights API	Training

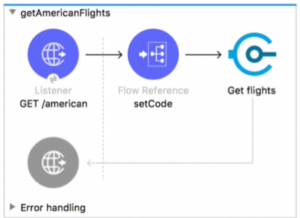


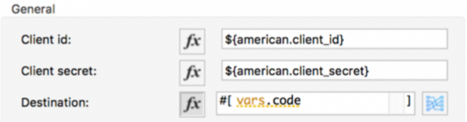
All contents © MuleSoft Inc.

Walkthrough 8-1: Consume a RESTful web service that has an API (and connector) in Exchange

MuleSoft

- Create a new flow to call the American RESTful web service
- Add a REST connector from Exchange to an Anypoint Studio project
- Configure and use a REST connector to make a call to a web service
- Dynamically set a query parameter for a web service call





General

Client id: ${american.client_id}

Client secret: ${american.client_secret}

Destination: #[vars.code]

All contents © MuleSoft Inc.

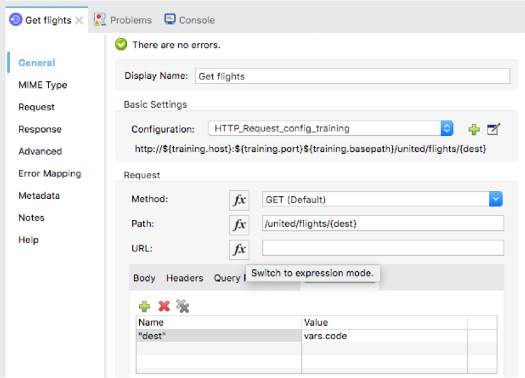
Consuming RESTful web services

Consuming RESTful web services

MuleSoft

- First check and see if there is an existing Anypoint Connector in Studio or Exchange to connect to the service provider
- If there is not, use the **HTTP** connector and its **Request** operation
  - Configure the operation and/or global element configuration
  - Specify any headers, query parameters, or URI parameters to pass to the call

Request



Get flights | Problems | Console

There are no errors.

General
MIME Type
Request
Response
Advanced
Error Mapping
Metadata
Notes
Help

Display Name: Get flights

Basic Settings

Configuration: HTTP_Request_config_training

http://${training.host}:${training.port}${training.basepath}/united/flights/{dest}

Request

Method: GET (Default)

Path: /united/flights/{dest}

URL:

Body | Headers | Query P

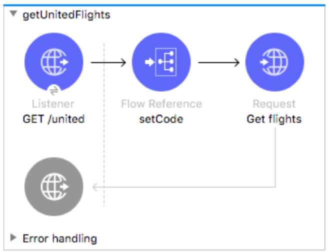
Switch to expression mode.

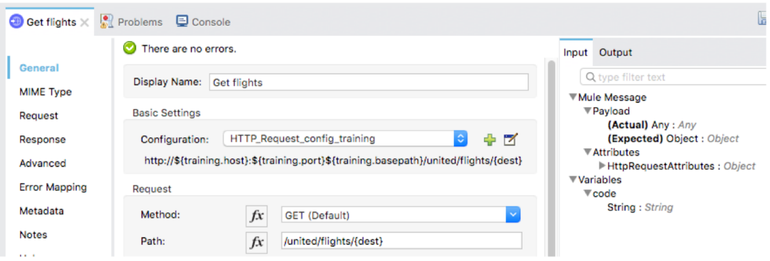
Name	Value
"dest"	vars.code

Walkthrough 8-2: Consume a RESTful web service

MuleSoft

- Create a new flow to call the United RESTful web service
- Use the HTTP Request operation to call a RESTful web service
- Dynamically set a URI parameter for a web service call
- Add metadata for an HTTP Request operation's response





There are no errors.

Display Name: Get flights

Basic Settings

Configuration: HTTP_Request_config_training

http://${training.host}:${training.port}${training.basepath}/united/flights/{dest}

Request

Method: GET (Default)

Path: /united/flights/{dest}

```
▼Mule Message
  ▼Payload
      (Actual) Any : Any
      (Expected) Object : Object
  ▼Attributes
    ▶HttpRequestAttributes : Object
▼Variables
  ▼code
      String : String
```

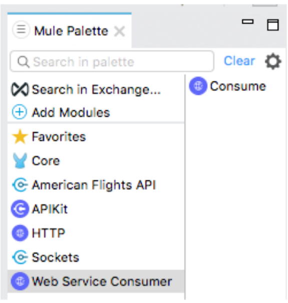
Consuming SOAP web services

Consuming SOAP web services

MuleSoft

- First check and see if there is an existing Anypoint Connector in Studio or Exchange to connect to the service provider
- If there is not, use the **Web Service Consumer** connector
  - Add the Web Service Consumer module to the project
  - Configure a global element configuration, which includes the location of the WSDL
  - Use the Consume operation
  - Select the SOAP operation to invoke

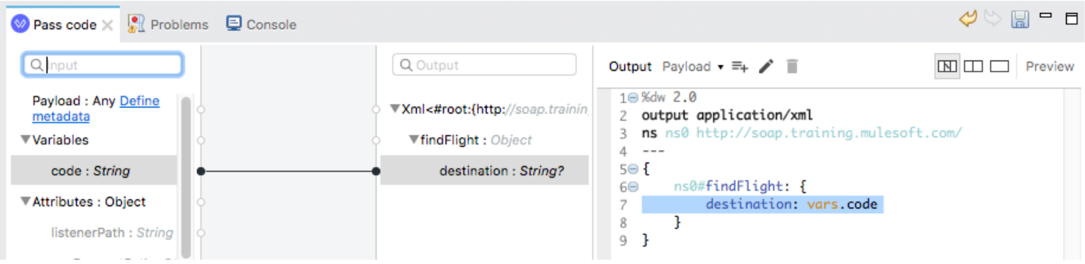
Consume



Passing data to a SOAP web service

MuleSoft

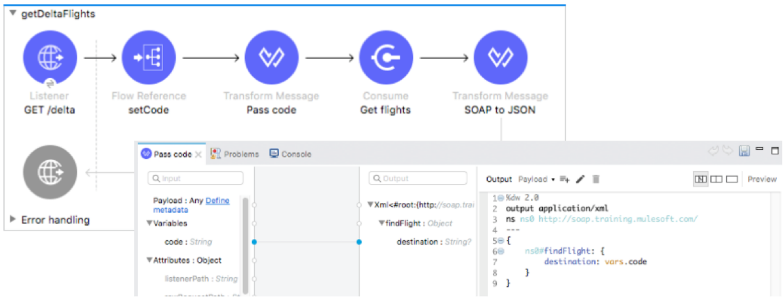
- Use the **Transform Message** component to pass arguments to a SOAP web service
- When you add it before the Consume operation, DataSense is used to create metadata for the input that includes the arguments



```
%dw 2.0
output application/xml
ns ns0 http://soap.training.mulesoft.com/
---
{
    ns0#findFlight: {
        destination: vars.code
    }
}
```

Walkthrough 8-3: Consume a SOAP web service

- Create a new flow to call the Delta SOAP web service
- Use the Web Service Consumer connector to call a SOAP web service
- Use the Transform Message component to pass arguments to a SOAP web service



```
%dw 2.0
output application/xml
ns ns0 http://soap.training.mulesoft.com/
---
{
    ns0#findFlight: {
        destination: vars.code
    }
}
```

Combining data from multiple services

Combining data from multiple services

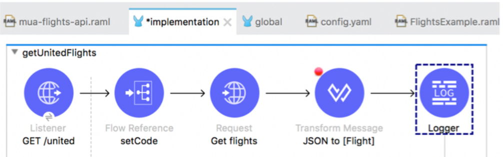
MuleSoft

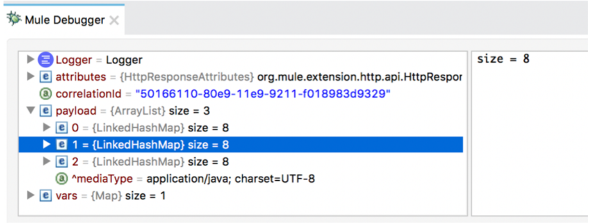
- Data from different services is pretty much always going to be in different formats
- To combine the data sets, you need to transform each of them to a canonical, or standard format
  - In this module, you will use a Java class as the canonical format
  - In module 11, you will learn to use the DataWeave format as a canonical format

Walkthrough 8-4: Transform data from multiple services to a canonical format

MuleSoft

- Define a metadata type for the Flight Java class
- Transform the results from RESTful and SOAP web service calls to a collection of Flight objects

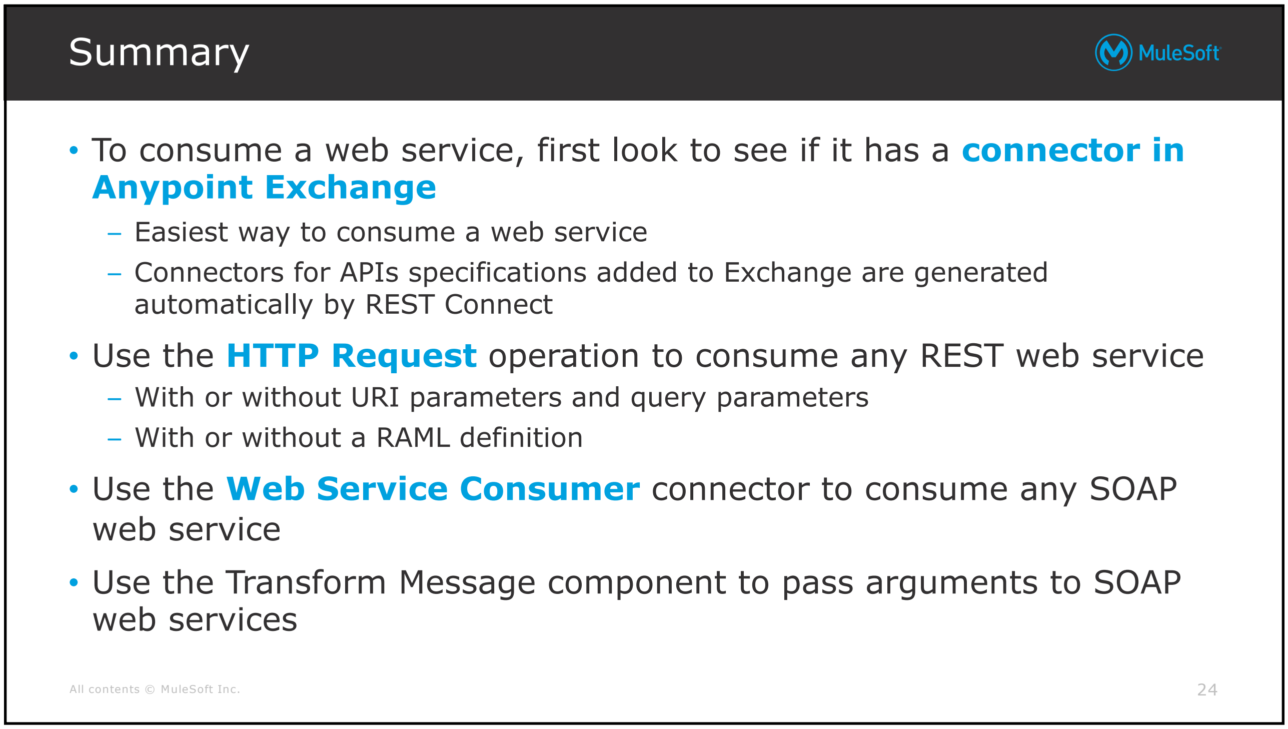




```
▶ Logger = Logger
▶ attributes = {HttpResponseAttributes} org.mule.extension.http.api.HttpRespor
  correlationId = "50166110-80e9-11e9-9211-f018983d9329"
▼ payload = {ArrayList} size = 3
  ▶ 0 = {LinkedHashMap} size = 8
  ▶ 1 = {LinkedHashMap} size = 8
  ▶ 2 = {LinkedHashMap} size = 8
    ^mediaType = application/java; charset=UTF-8
▶ vars = {Map} size = 1
```

```
size = 8
```

Summary



Summary

- To consume a web service, first look to see if it has a **connector in Anypoint Exchange**
  - Easiest way to consume a web service
  - Connectors for APIs specifications added to Exchange are generated automatically by REST Connect
- Use the **HTTP Request** operation to consume any REST web service
  - With or without URI parameters and query parameters
  - With or without a RAML definition
- Use the **Web Service Consumer** connector to consume any SOAP web service
- Use the Transform Message component to pass arguments to SOAP web services