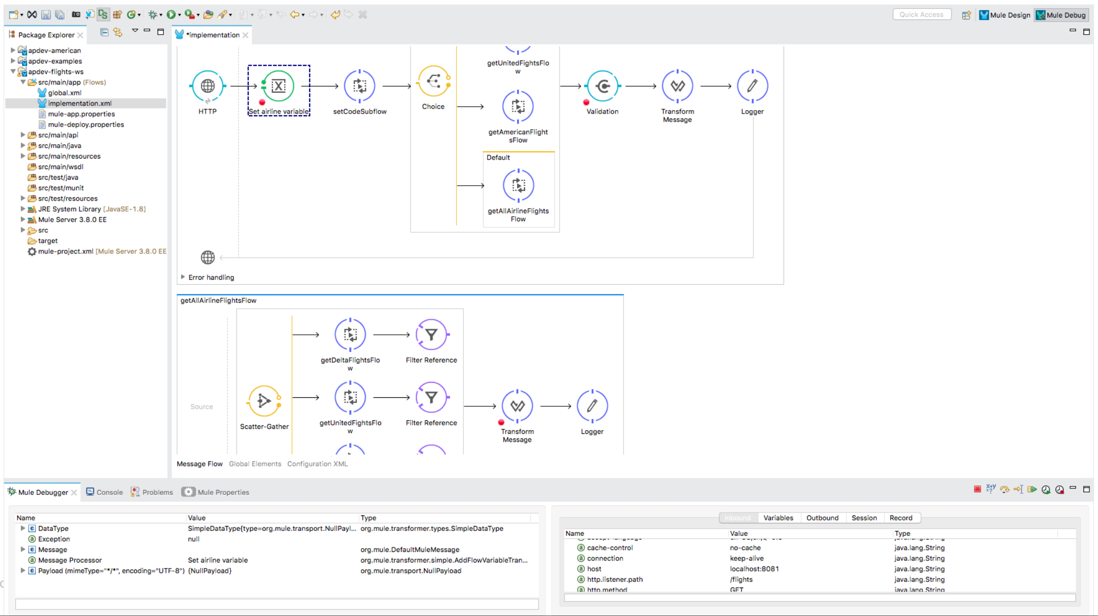
MuleSoft

PART 2: Building Applications with Anypoint Studio

Goal

MuleSoft



Name	Value	Type
DataType	SimpleDataType{type=org.mule.transport.NullPayl...	org.mule.transformer.types.SimpleDataType
Exception	null	
Message		org.mule.DefaultMuleMessage
Message Processor	Set airline variable	org.mule.transformer.simple.AddFlowVariableTran...
Payload (mimeType="*/*", encoding="UTF-8")	{NullPayload}	org.mule.transport.NullPayload

Name	Value	Type
cache-control	no-cache	java.lang.String
connection	keep-alive	java.lang.String
host	localhost:8081	java.lang.String
http.listener.path	/flights	java.lang.String
http.method	GET	java.lang.String

All c

2

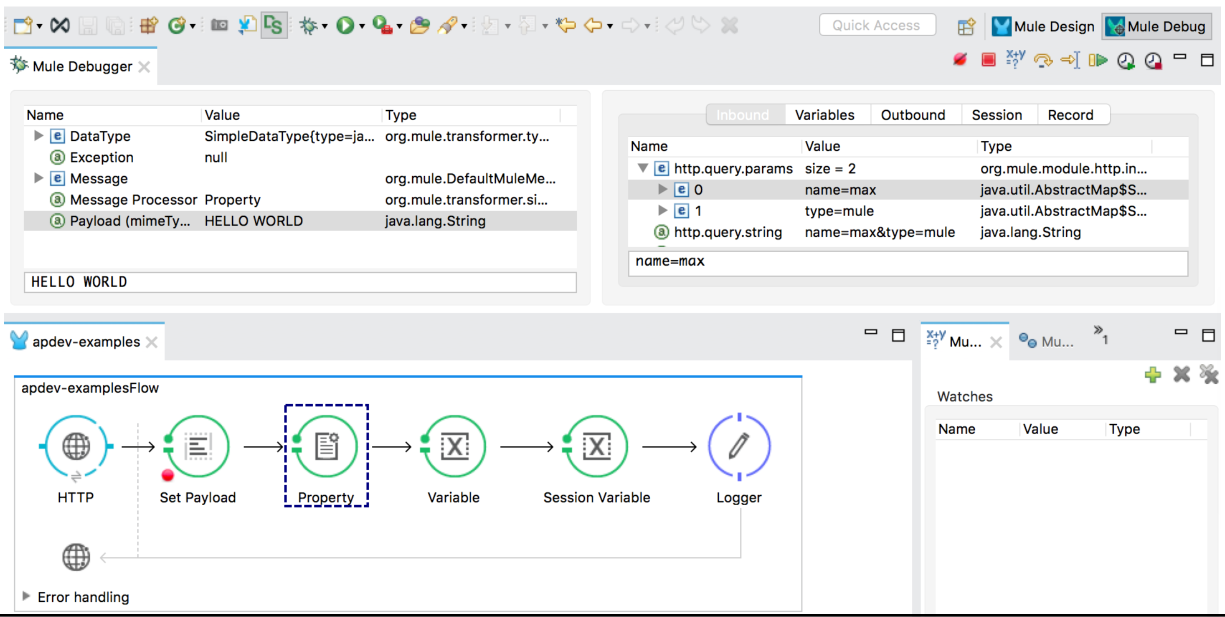
At the end of this part, you should be able to

- Debug Mule applications
- Read and write message payloads, properties, and variables using the Mule Expression Language
- Structure Mule applications using flows, subflows, in-memory message queues, properties files, and configuration files
- Connect to web services, SaaS applications, files, polled resources, JMS queues, and more
- Route, filter, and validate messages and handle message exceptions
- Write DataWeave expressions for more complicated transformations
- Process individual records in a collection and synchronize data in databases to SaaS applications

MuleSoft

Module 6: Accessing and Modifying Mule Messages

Goal



Name	Value	Type
DataType	SimpleDataType{type=ja...	org.mule.transformer.ty...
Exception	null	
Message		org.mule.DefaultMuleMe...
Message Processor	Property	org.mule.transformer.si...
Payload (mimeTy...	HELLO WORLD	java.lang.String

```
HELLO WORLD
```

Inbound | Variables | Outbound | Session | Record

Name	Value	Type
http.query.params	size = 2	org.mule.module.http.in...
0	name=max	java.util.AbstractMap$S...
1	type=mule	java.util.AbstractMap$S...
http.query.string	name=max&type=mule	java.lang.String

```
name=max
```

Name	Value	Type

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

- Log message data
- Debug Mule applications
- Read and write message properties
- Write expressions with Mule Expression Language (MEL)
- Create variables

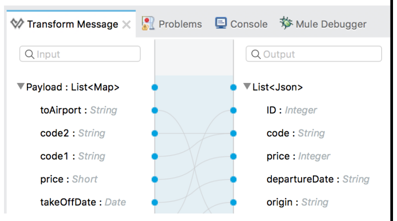
Accessing information about Mule 3 messages

View message info using DataSense

MuleSoft

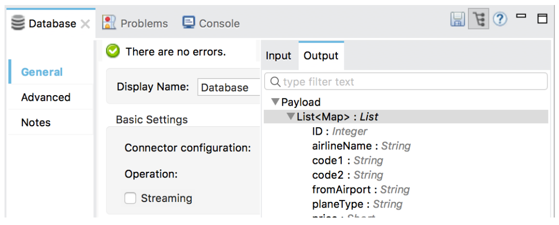
- **We saw this already using the Transform Message component**

- **DataSense** is the ability to proactively discover metadata from internal and external resources
  - Keeps you from having to manually discover information about the data
  - Facilitates transformations by providing DataWeave expected input or output



```
▼Payload : List<Map>          ▼List<Json>
    toAirport : String            ID : Integer
    code2 : String                code : String
    code1 : String                price : Integer
    price : Short                 departureDate : String
    takeOffDate : Date            origin : String
```

- There is also a DataSense Explorer in the Properties view
  - Lets you see message data structure throughout a flow at design time



There are no errors.

Display Name: Database

Basic Settings

Connector configuration:

Operation:

☐ Streaming

```
▼Payload
  ▼List<Map> : List
      ID : Integer
      airlineName : String
      code1 : String
      code2 : String
      fromAirport : String
      planeType : String
```

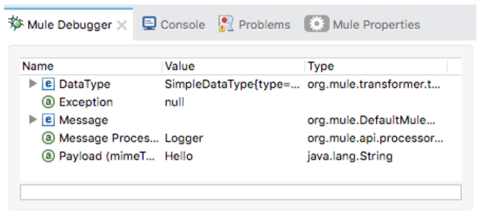
Other ways to view message information

MuleSoft

- Add a Logger component to a flow and view its output in the Anypoint Studio console

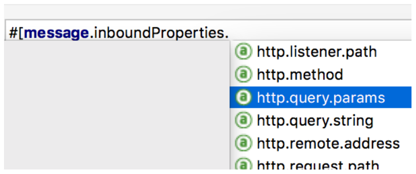
Logger

- Use the Anypoint Studio Visual Debugger
  - Most comprehensive way
  - Also has a Mule Expression Evaluator



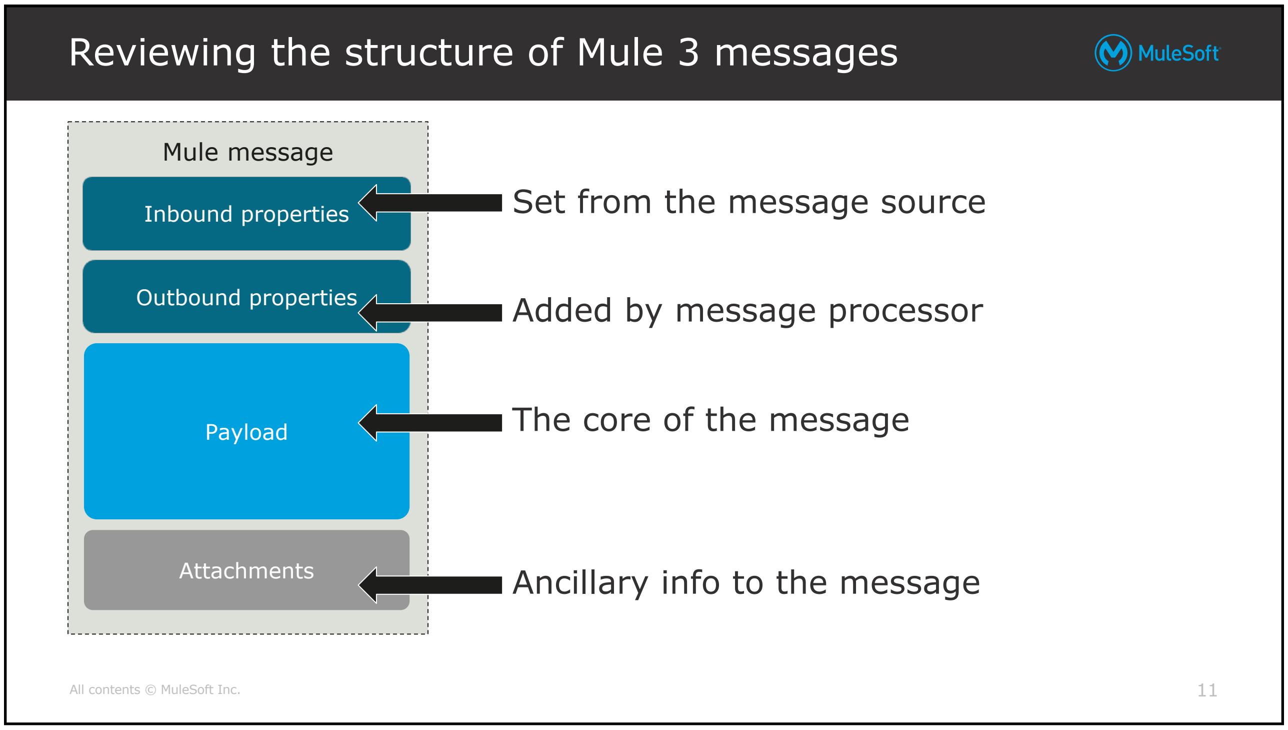
Name	Value	Type
DataType	SimpleDataType{type=...	org.mule.transformer.t...
Exception	null	
Message		org.mule.DefaultMule...
Message Proces...	Logger	org.mule.api.processor...
Payload (mimeT...	Hello	java.lang.String

- Use autocomplete when writing expressions in the Anypoint Studio Visual Editor



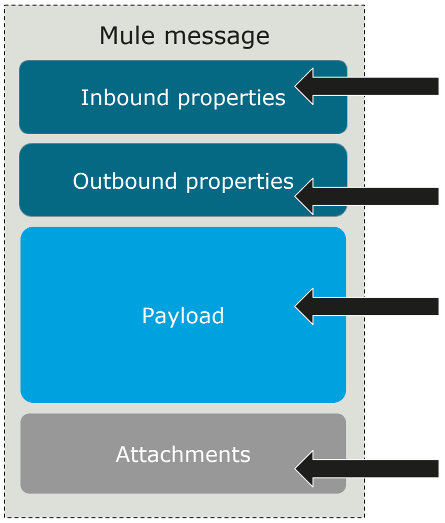
```
#[message.inboundProperties.
http.listener.path
http.method
http.query.params
http.query.string
http.remote.address
http.request.path
```

Setting message data



Reviewing the structure of Mule 3 messages

MuleSoft

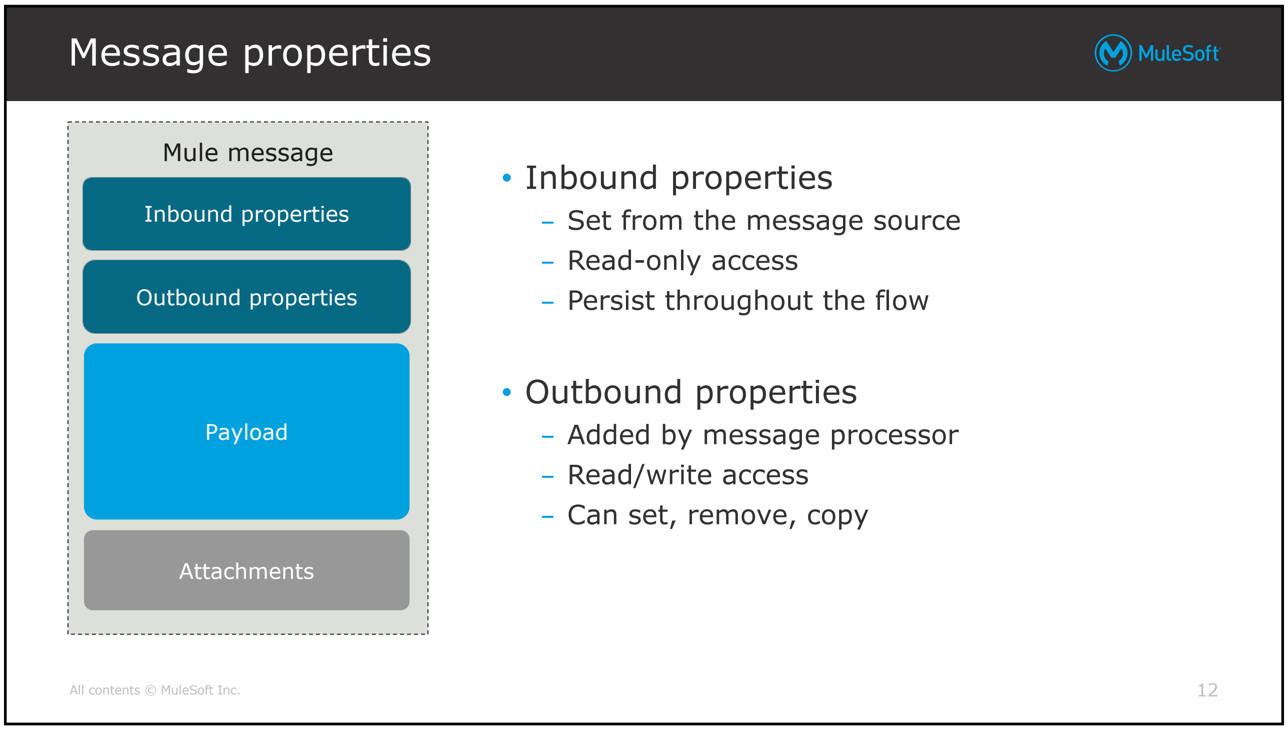


Set from the message source

Added by message processor

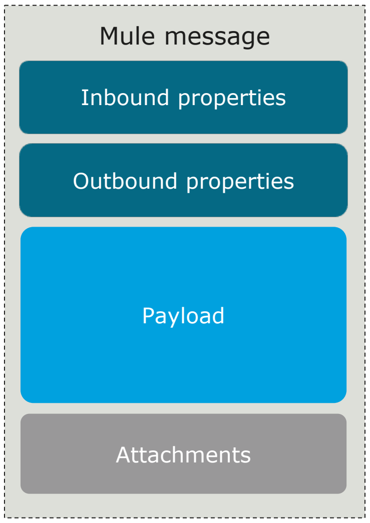
The core of the message

Ancillary info to the message



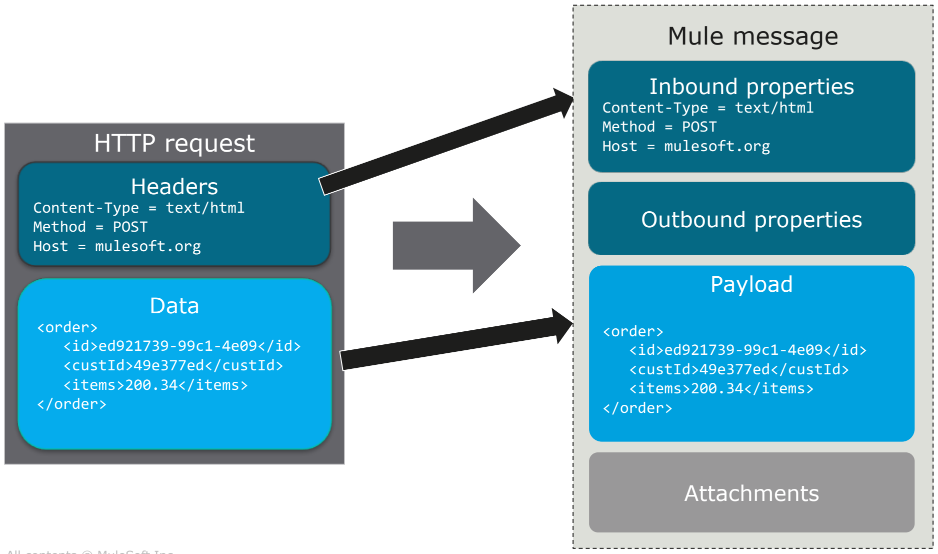
Message properties

MuleSoft

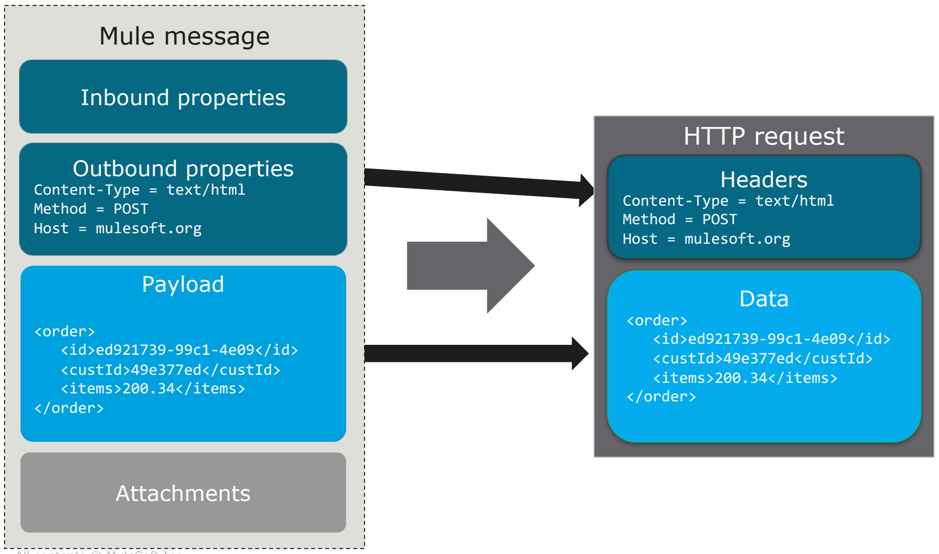


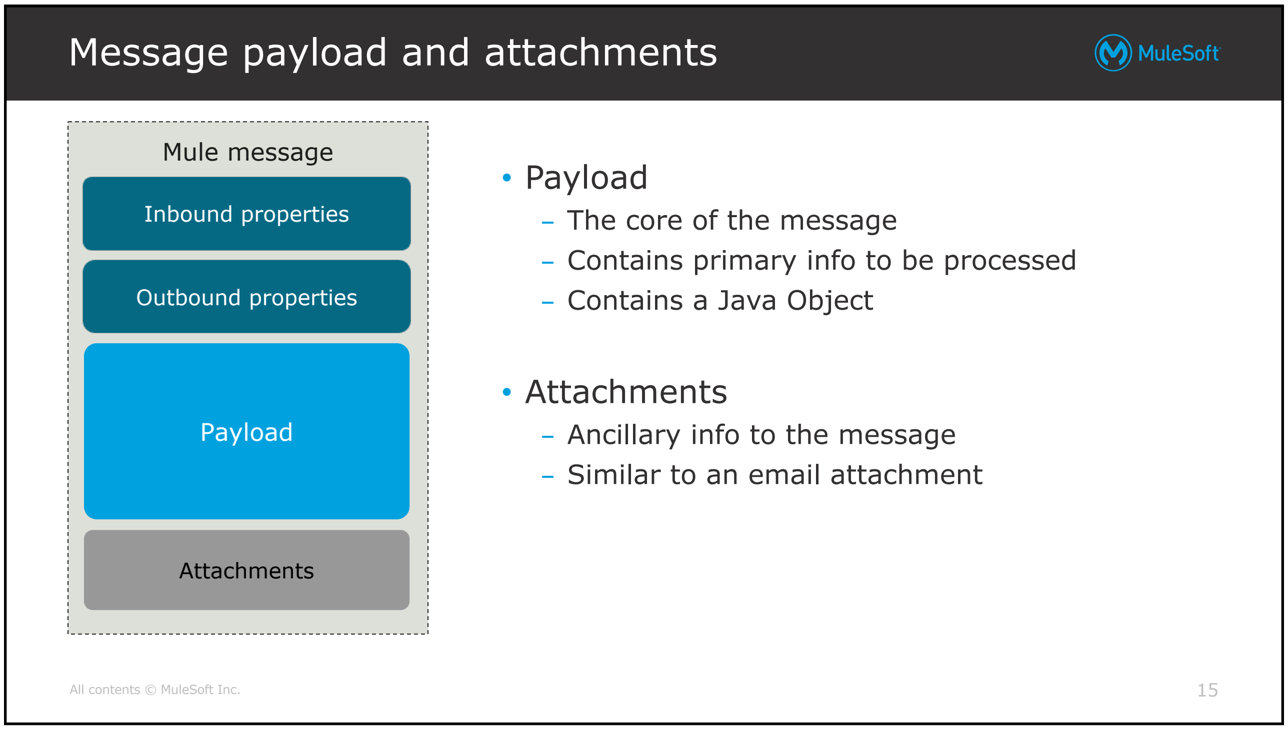
- Inbound properties
  - Set from the message source
  - Read-only access
  - Persist throughout the flow
- Outbound properties
  - Added by message processor
  - Read/write access
  - Can set, remove, copy

Inbound message properties



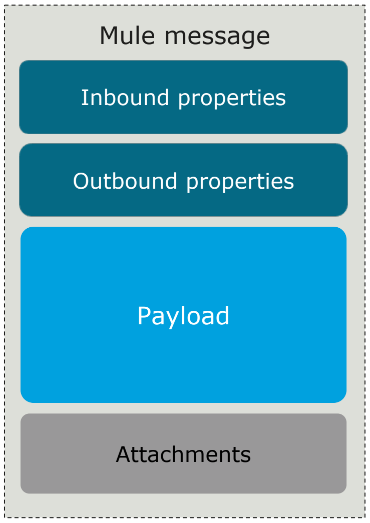
Outbound message properties





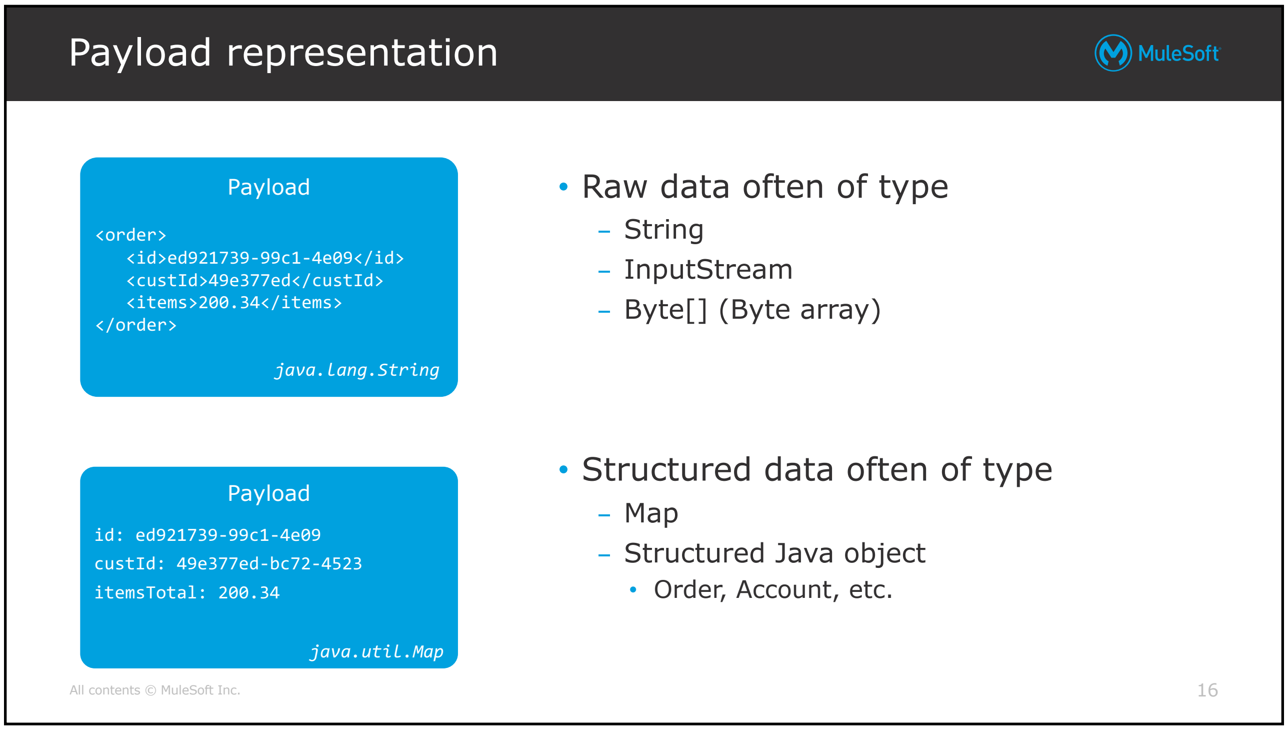
Message payload and attachments

MuleSoft



- Payload
  - The core of the message
  - Contains primary info to be processed
  - Contains a Java Object
- Attachments
  - Ancillary info to the message
  - Similar to an email attachment

All contents © MuleSoft Inc. 15



Payload representation

MuleSoft

Payload

```
<order>
   <id>ed921739-99c1-4e09</id>
   <custId>49e377ed</custId>
   <items>200.34</items>
</order>
```

java.lang.String

Payload

```
id: ed921739-99c1-4e09
custId: 49e377ed-bc72-4523
itemsTotal: 200.34
```

java.util.Map

- Raw data often of type
  - String
  - InputStream
  - Byte[] (Byte array)
- Structured data often of type
  - Map
  - Structured Java object
    - Order, Account, etc.

All contents © MuleSoft Inc. 16

Setting message properties

MuleSoft

Set Payload

- Sets the value of the message payload
  - message.payload

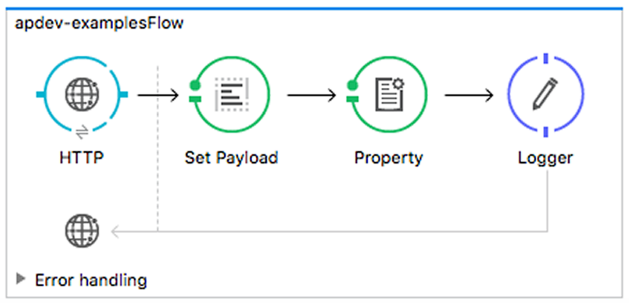
Property

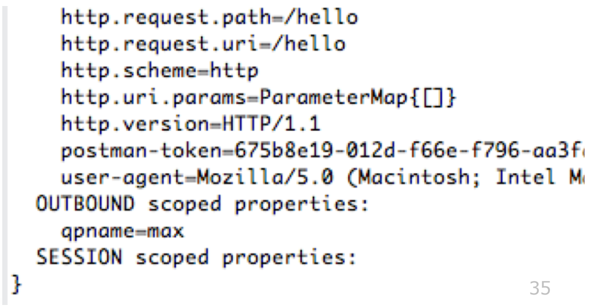
- Sets, removes, or copies properties on the outbound scope of a message
  - message.outboundProperties
- These are transformers in the Mule Palette in Studio

Walkthrough 6-1: Set and log message data

MuleSoft

- Create a new project
- Set the message payload
- Set message outbound properties
- Log the message to the console





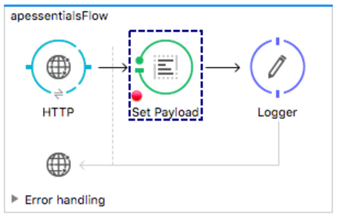
```
   http.request.path=/hello
   http.request.uri=/hello
   http.scheme=http
   http.uri.params=ParameterMap{[]}
   http.version=HTTP/1.1
   postman-token=675b8e19-012d-f66e-f796-aa3f
   user-agent=Mozilla/5.0 (Macintosh; Intel M
 OUTBOUND scoped properties:
   qpname=max
 SESSION scoped properties:
}
```

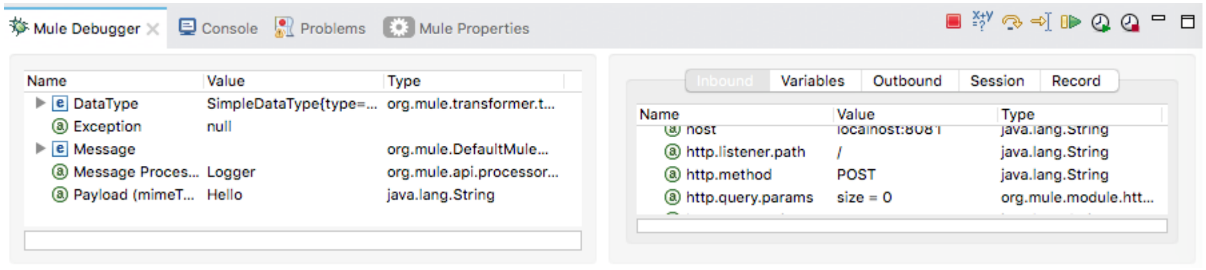
Debugging Mule applications

Debugging applications with the Mule Debugger

MuleSoft

- Can add breakpoints to processors and step through the application
  - Watch message and variable values
  - Watch and evaluate expressions
- By default, Debugger listens for incoming TCP connections on localhost port 6666
  - Can change this in a project’s run configuration



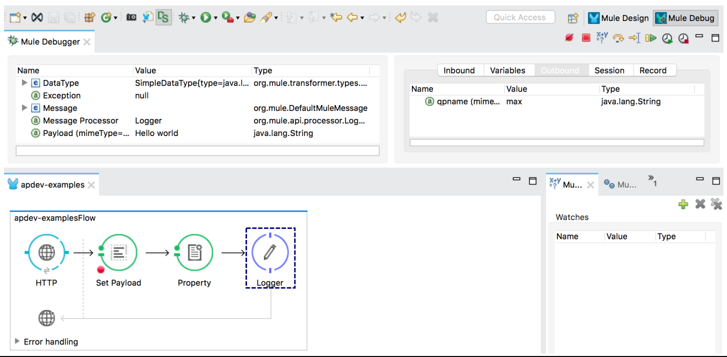


Name	Value	Type
DataType	SimpleDataType{type=...	org.mule.transformer.t...
Exception	null	
Message		org.mule.DefaultMule...
Message Proces...	Logger	org.mule.api.processor...
Payload (mimeT...	Hello	java.lang.String

Name	Value	Type
host	localhost:8081	java.lang.String
http.listener.path	/	java.lang.String
http.method	POST	java.lang.String
http.query.params	size = 0	org.mule.module.htt...

Walkthrough 6-2: Debug a Mule application

- Locate the port used by the Mule Debugger
- Add a breakpoint, debug an application, and step through the code
- Use the Mule Debugger to view message properties
- Pass query parameters to a request and locate them in the Debugger

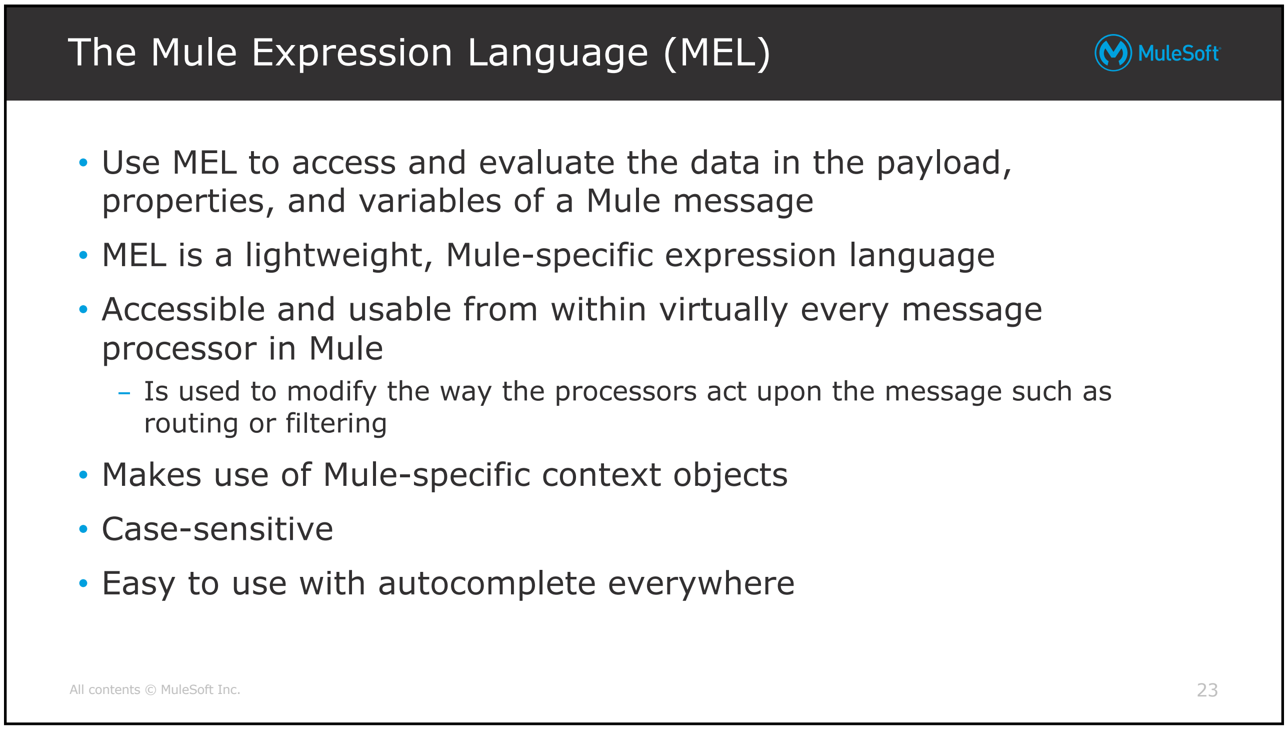


Name	Value	Type
DataType	SimpleDataType{type=java.l...	org.mule.transformer.types....
Exception	null	
Message		org.mule.DefaultMuleMessage
Message Processor	Logger	org.mule.api.processor.Log...
Payload (mimeType=...	Hello world	java.lang.String

Inbound | Variables | Outbound | Session | Record

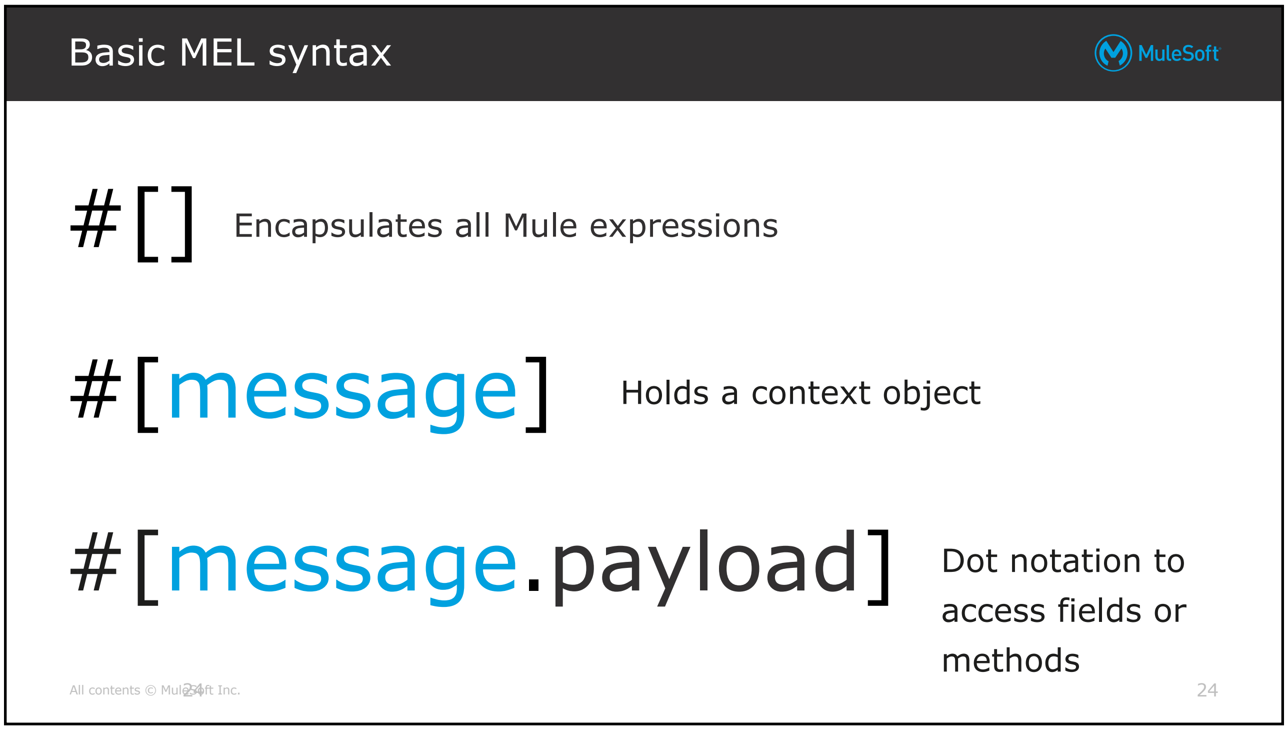
Name	Value	Type
qpname (mime...	max	java.lang.String

Using expressions to read and write message data



The Mule Expression Language (MEL)

- Use MEL to access and evaluate the data in the payload, properties, and variables of a Mule message
- MEL is a lightweight, Mule-specific expression language
- Accessible and usable from within virtually every message processor in Mule
  - Is used to modify the way the processors act upon the message such as routing or filtering
- Makes use of Mule-specific context objects
- Case-sensitive
- Easy to use with autocomplete everywhere



Basic MEL syntax

```
#[]
```
Encapsulates all Mule expressions

```
#[message]
```
Holds a context object

```
#[message.payload]
```
Dot notation to access fields or methods

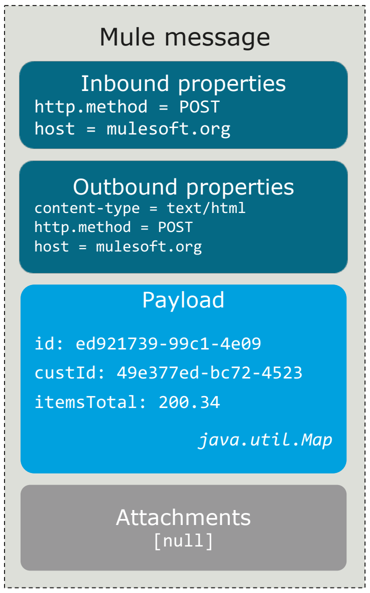
Context objects

MuleSoft

server	Operating system that message processor is running
mule	The Mule instance that the application is running
app	User application the current flow is deployed in
message	The Mule message that the message processer is processing

Accessing message data

MuleSoft



```
http.method = POST
host = mulesoft.org
```

```
content-type = text/html
http.method = POST
host = mulesoft.org
```

```
id: ed921739-99c1-4e09
custId: 49e377ed-bc72-4523
itemsTotal: 200.34
java.util.Map
```

```
[null]
```

`#[message.inboundProperties.host]`

mulesoft.org

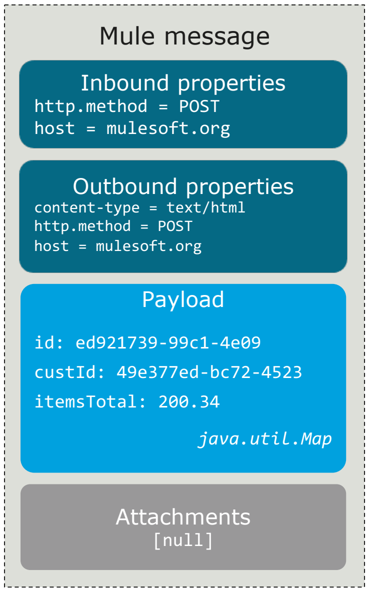
`#[message.inboundProperties['http.method']]`

POST

`#[message.inboundProperties['content-type']]`

text/html

Accessing message payload data



```
#[message.payload.id]
#[message.payload['id']]
```
ed921739-99c1-4e09

```
#[message.payload.itemsTotal]
```
200.34

```
#[message.payload.toString()]
```

`#[payload]` is a shortcut for `#[message.payload]`

This shortcut only works with payload

27

Accessing relational map data

FirstName	LastName	City	State
John	Muley	Boston	Ohio
Mark	Dailer	Cleveland	Ohio
Bill	Muley	Avon	Ohio

```
#[message.payload[1]['LastName']]
```
Dailer

```
#[message.payload[0].City]
```
Boston

28

Accessing relational map data

MuleSoft

- Operators
  - Arithmetic: +, -, /, *, %
  - Evaluation: ==, !=, >, <, >=, <=, contains, is

    #[message.inboundProperties.'http.query.params'.lastname != null]

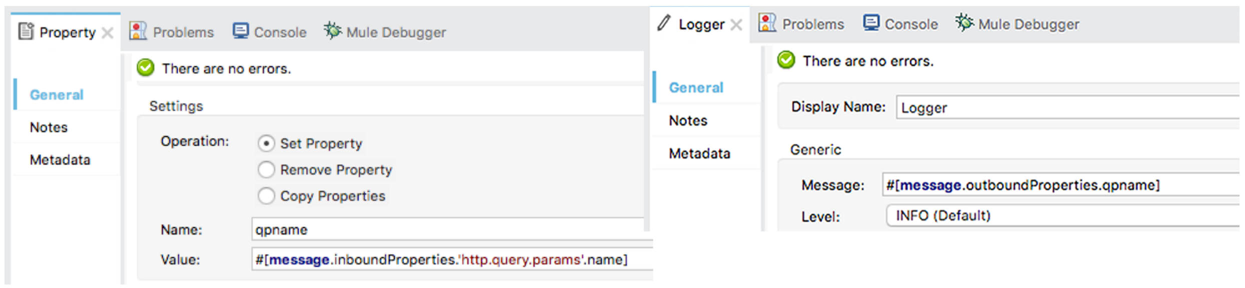
- Testing for emptiness
  - The literal **empty** tests the emptiness of a value
    - Null, boolean false, "", " ", zero, empty collections

- Data extraction
  - XPath: #[xpath('expression')]
  - RegEx: #[regex('expression')]

Walkthrough 6-3: Read and write message properties using MEL expressions

MuleSoft

- Use an expression to set the payload
- Use an expression to display specific info to the console.
- Use an expression to set an outbound property
- Use an expression to read an outbound property



There are no errors.

Settings

Operation: [✓ Set Property] Remove Property Copy Properties

Name: qpname

Value: #[message.inboundProperties.'http.query.params'.name]

There are no errors.

Display Name: Logger

Generic

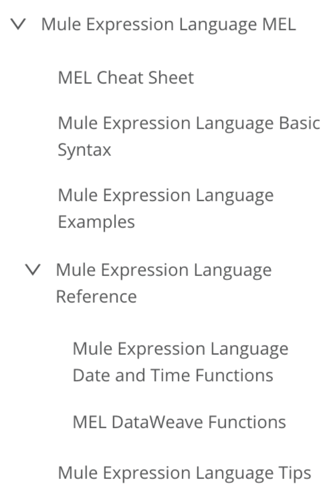
Message: #[message.outboundProperties.qpname]

Level: INFO (Default)

MEL references

MuleSoft

- MEL expression reference
  - https://docs.mulesoft.com/mule-user-guide/v/3.8/mule-expression-language-reference
- MEL language tips
  - https://docs.mulesoft.com/mule-user-guide/v/3.8/mule-expression-language-tips



∨ Mule Expression Language MEL

MEL Cheat Sheet

Mule Expression Language Basic Syntax

Mule Expression Language Examples

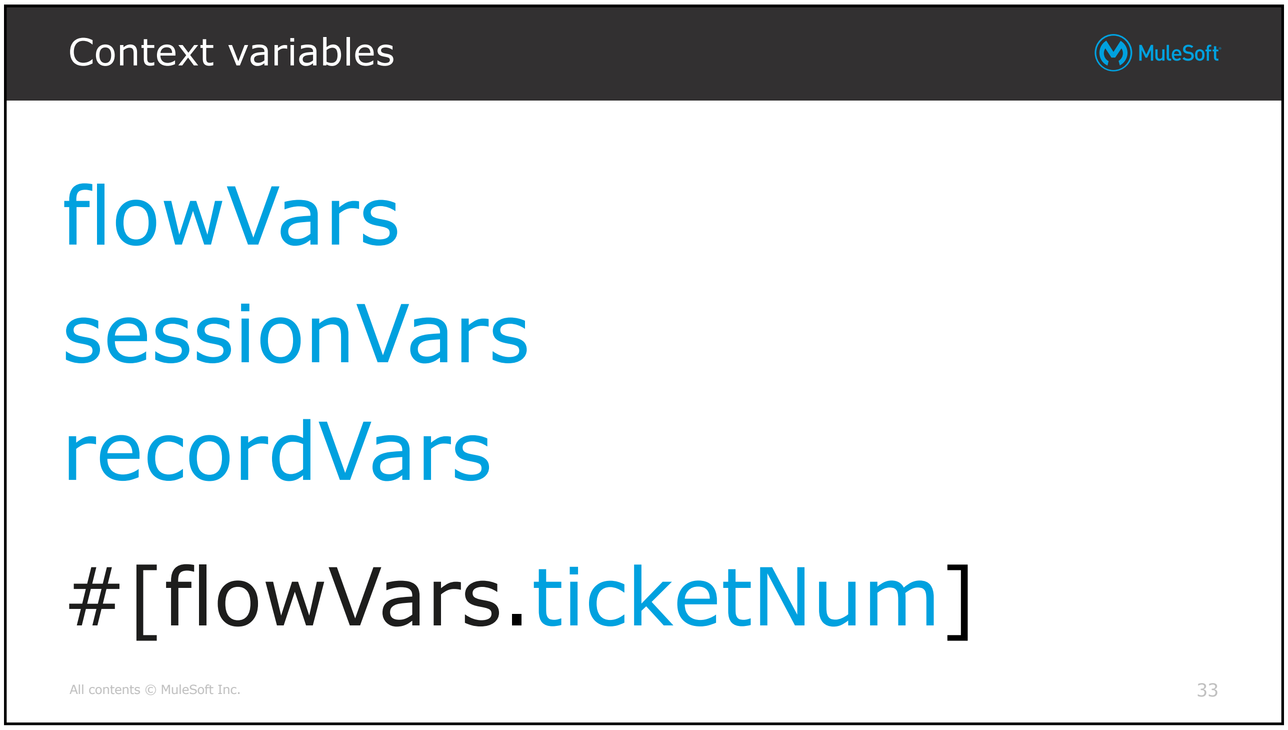
∨ Mule Expression Language Reference

Mule Expression Language Date and Time Functions

MEL DataWeave Functions

Mule Expression Language Tips

Creating variables



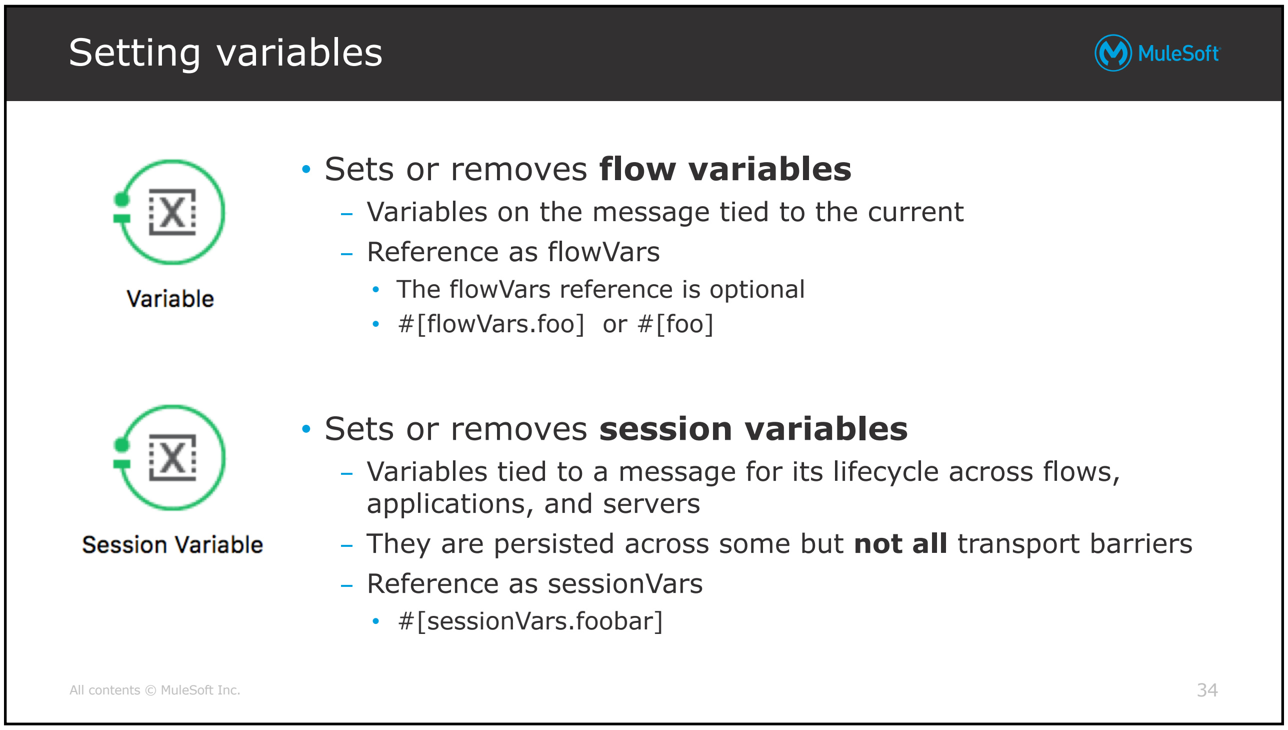
Context variables

flowVars

sessionVars

recordVars

#[flowVars.ticketNum]



Setting variables

Variable

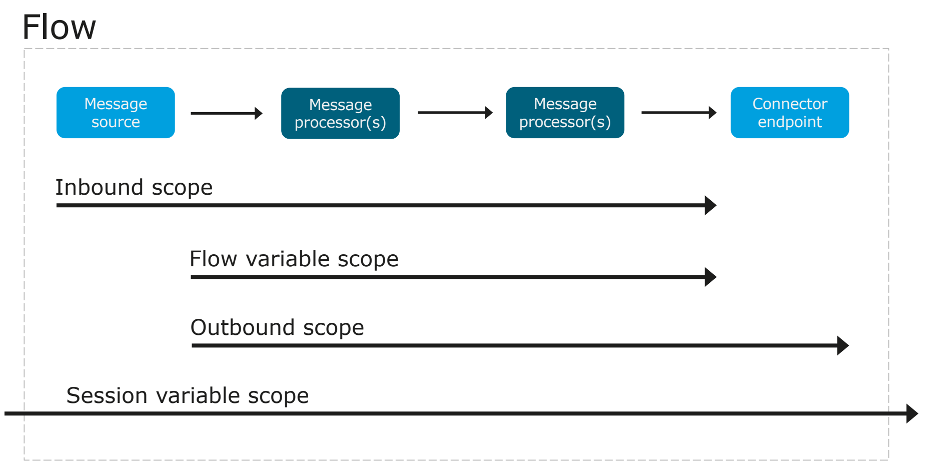
- Sets or removes **flow variables**
  - Variables on the message tied to the current
  - Reference as flowVars
    - The flowVars reference is optional
    - #[flowVars.foo] or #[foo]

Session Variable

- Sets or removes **session variables**
  - Variables tied to a message for its lifecycle across flows, applications, and servers
  - They are persisted across some but **not all** transport barriers
  - Reference as sessionVars
    - #[sessionVars.foobar]

Variable persistence

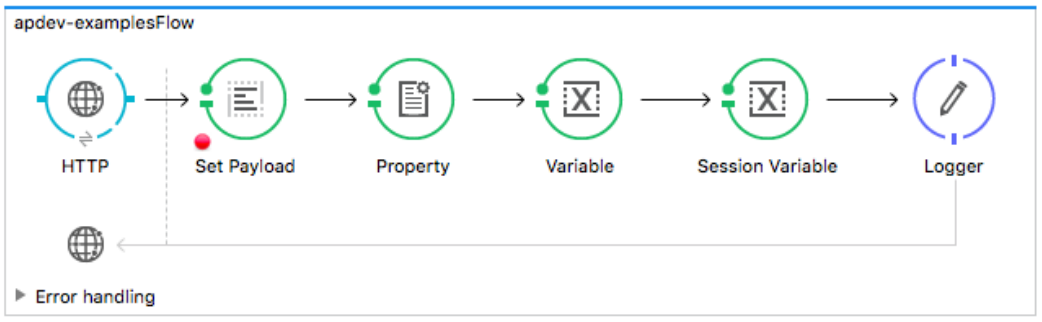
MuleSoft

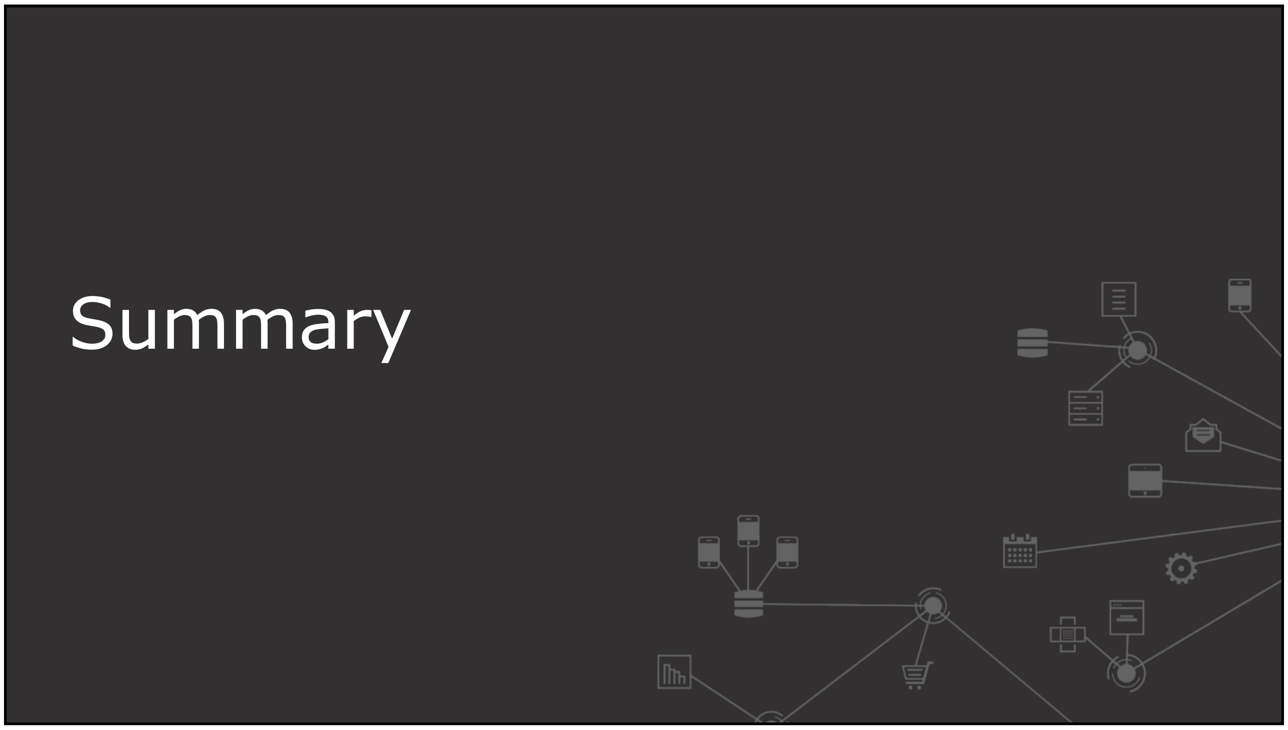


Walkthrough 6-4: Read and write variables

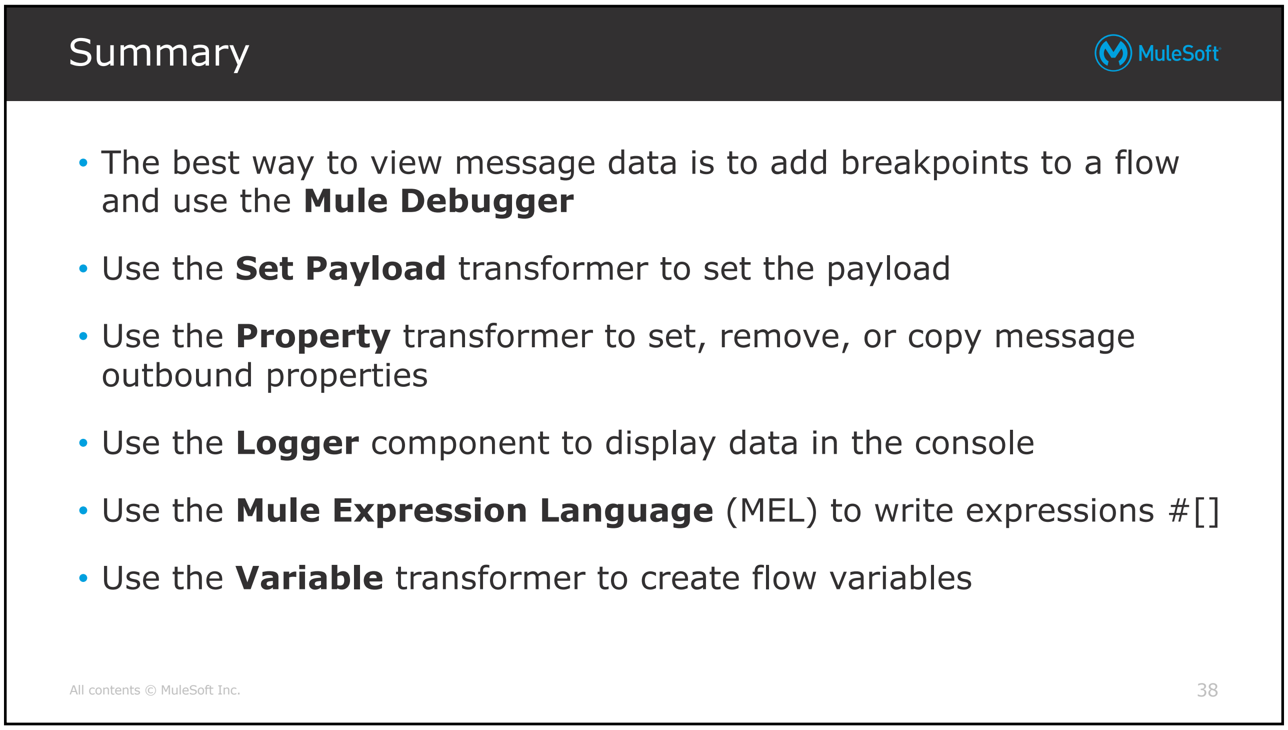
MuleSoft

- Use the Variable transformer to create a flow variable
- Use the Session transformer to create a session variable
- Use the Mule Debugger to see their values





Summary



Summary

- The best way to view message data is to add breakpoints to a flow and use the **Mule Debugger**
- Use the **Set Payload** transformer to set the payload
- Use the **Property** transformer to set, remove, or copy message outbound properties
- Use the **Logger** component to display data in the console
- Use the **Mule Expression Language** (MEL) to write expressions #[]
- Use the **Variable** transformer to create flow variables