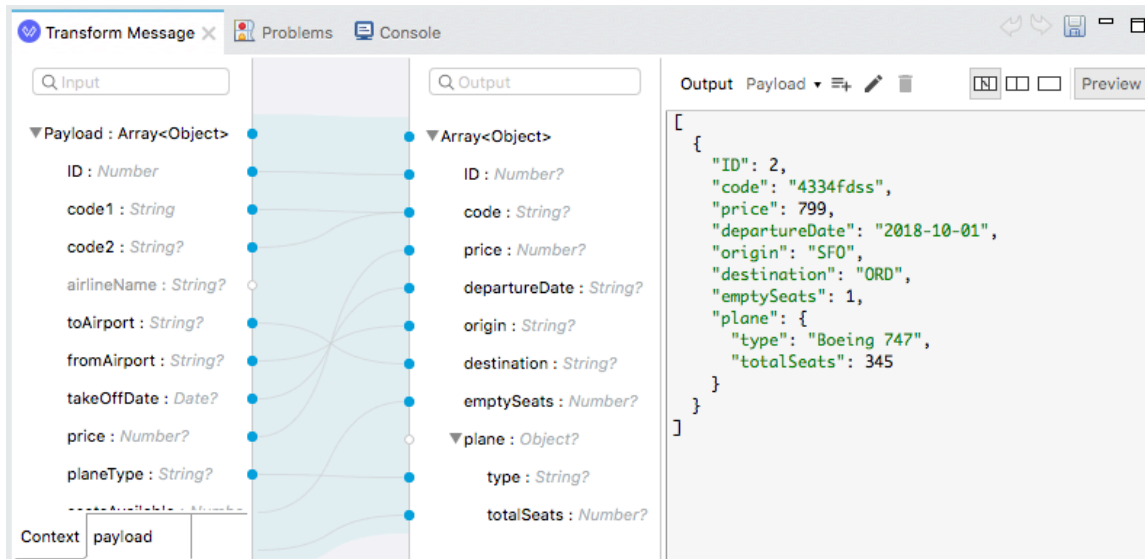


Walkthrough 4-3: Transform data

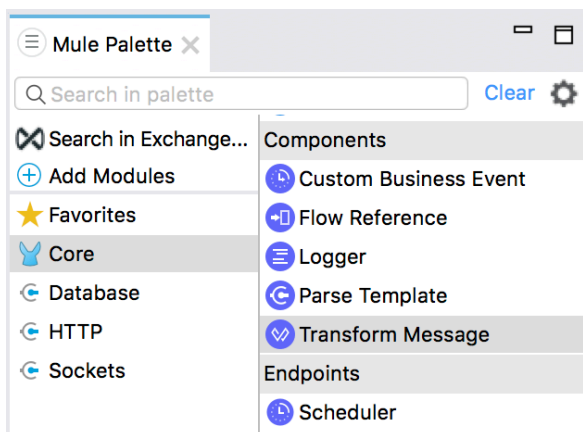
In this walkthrough, you transform and display the flight data into JSON. You will:

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

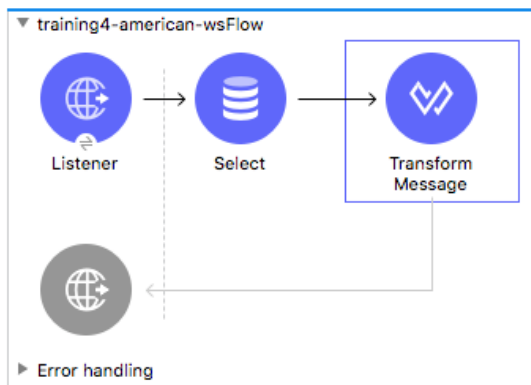


Add a Transform Message component

1. Return to Anypoint Studio.
2. In the Mule Palette, select Core and locate the Transform Message component in the Components section.



3. Drag the Transform Message and drop it after the Select processor.



Review metadata for the transformation input

4. In the Transform Message properties view, look at the input section and review the payload metadata.

Note: If you are using the local Derby database, the properties will be uppercase instead.

Transform Message x Problems Console

Input

▼ Payload : Array<Object>

toAirport : String?

code2 : String?

code1 : String?

price : Number?

Context

Drag-and-Drop fields to build the transform

Define metadata

Output Payload ▾

```
1 %dw 2.0
2 output application/java
3 ---
4 {
5 }
```

Return the payload as JSON

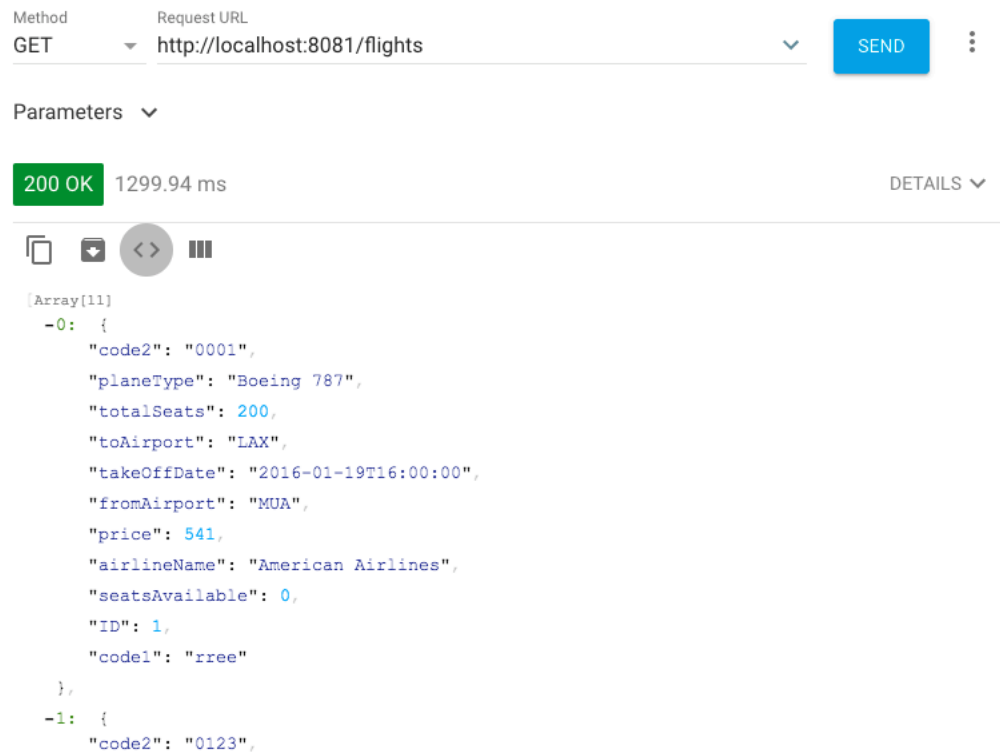
5. In the Transform Message properties view, change the output type from application/java to application/json and change the {} to payload.

Output Payload ▾

```
1 %dw 2.0
2 output application/json
3 ---
4 payload
```

Test the application

6. Save the file to redeploy the project.
7. In Advanced REST Client, send the same request; you should see the American flight data represented as JSON.

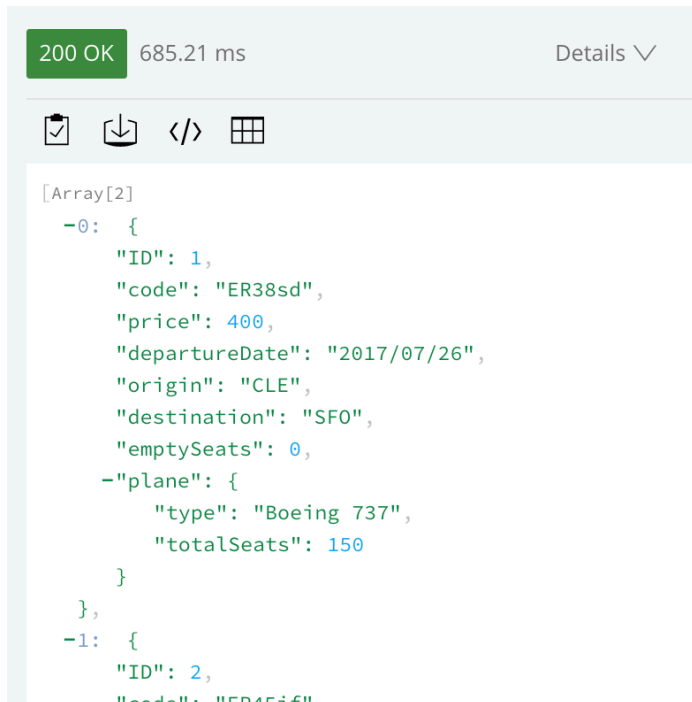


Note: If you are using the local Derby database, the properties will be uppercase instead.

Review the data structure to be returned by the American flights API

8. Return to your American Flights API in Exchange.

- Look at the example data returned for the /flights GET method.



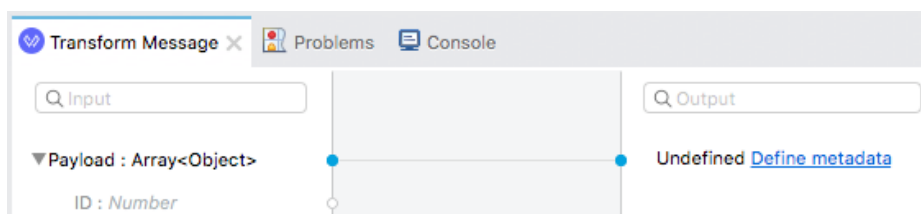
The screenshot shows a REST client interface with a status bar at the top indicating "200 OK" and a response time of "685.21 ms". Below the status bar, there are icons for a clipboard, download, code editor, and table view. The main area displays a JSON array of two flight objects. The first object has an ID of 1, code "ER38sd", price 400, departure date "2017/07/26", origin "CLE", destination "SF0", 0 empty seats, and a plane object with type "Boeing 737" and 150 total seats. The second object has an ID of 2 and code "ER45if".

```
[Array[2]
  -0: {
    "ID": 1,
    "code": "ER38sd",
    "price": 400,
    "departureDate": "2017/07/26",
    "origin": "CLE",
    "destination": "SF0",
    "emptySeats": 0,
    -"plane": {
      "type": "Boeing 737",
      "totalSeats": 150
    }
  },
  -1: {
    "ID": 2,
    "code": "ER45if"
```

- Notice that the structure of the JSON being returned by the Mule application does not match this JSON.

Define metadata for the data structure to be returned by the American flights API

- Return to Anypoint Studio.
- In the Transform Message properties view, click the Define metadata link in the output section.



- In the Select metadata type dialog box, click the Add button.
- In the Create new type dialog box, set the type id to american_flights_json.
- Click Create type.

16. Back in the Set metadata type dialog box, set the type to JSON.

17. Change the Schema selection to Example.

Select metadata type

The selected example file does not exist

+ Add - Delete

type filter text

Type JSON

▼ User Defined

american-flights-json : String

Example Select your JSON example file ...

training-american-ws will be copied to src/main/resources/examples

☐ Wrap element in a collection

Close Select

18. Click the browse button and navigate to the course student files.

19. Select american-flights-example.json in the examples folder and click Open; you should see the example data for the metadata type.

Select metadata type

Choose metadata type from tree and click Select

+ Add - Delete

type filter text

Type JSON

▼ User Defined

american-flights-json : String

Example amples/american-flights-example.json ...

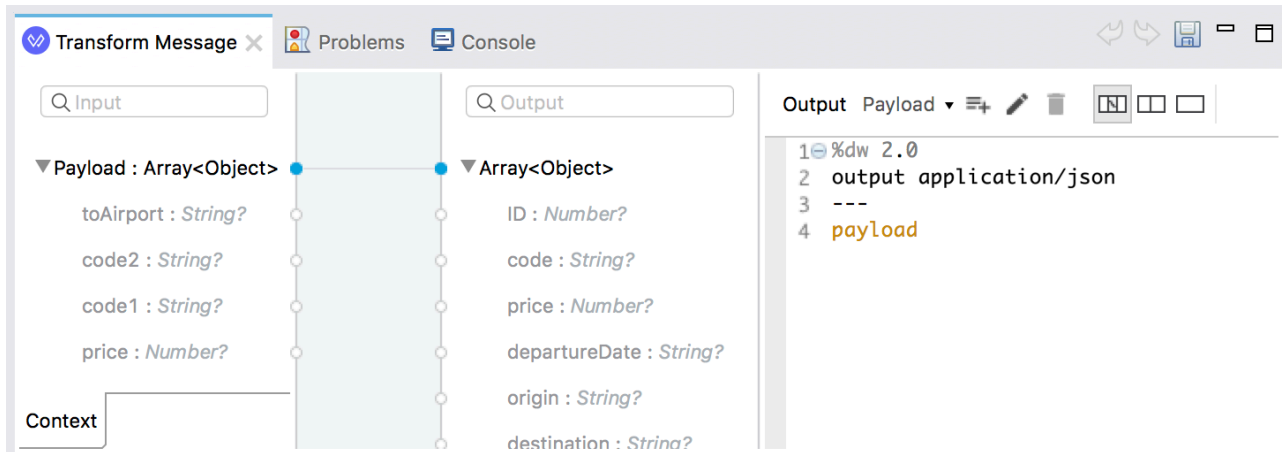
american-flights-example.json will be copied to src/main/resources/examples

ID : Number?
code : String?
price : Number?
departureDate : String?
origin : String?
destination : String?
emptySeats : Number?
▼ plane : Object?
type : String?
totalSeats : Number?

☐ Wrap element in a collection

Close Select

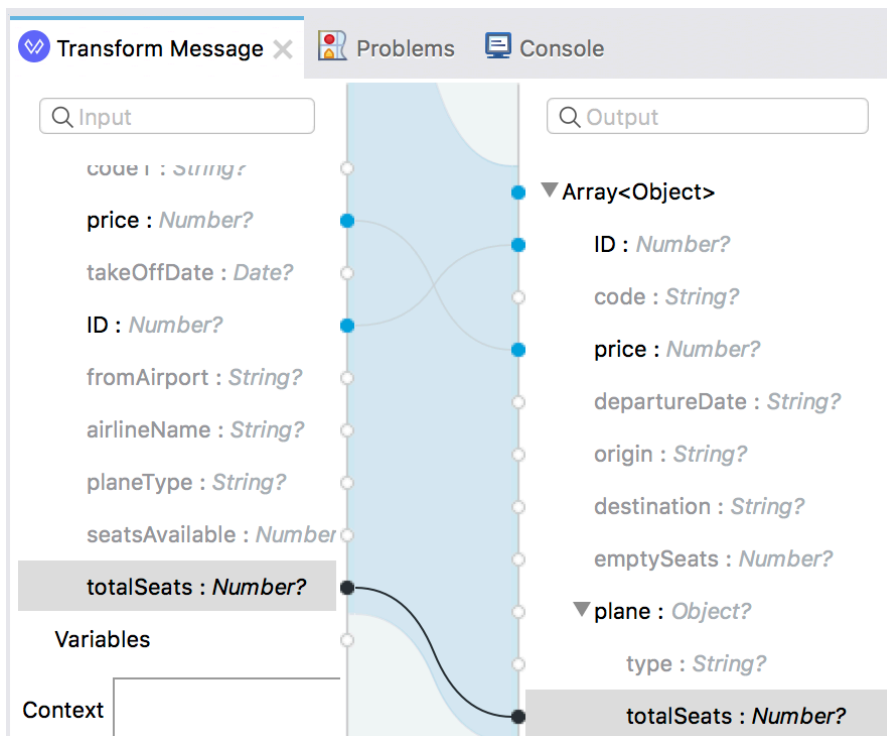
20. Click Select; you should now see output metadata in the output section of the Transform Message properties view.



Create the transformation

21. Map fields with the same names by dragging them from the input section and dropping them on the corresponding field in the output section.

- ID to ID
- price to price
- totalSeats to plane > totalSeats

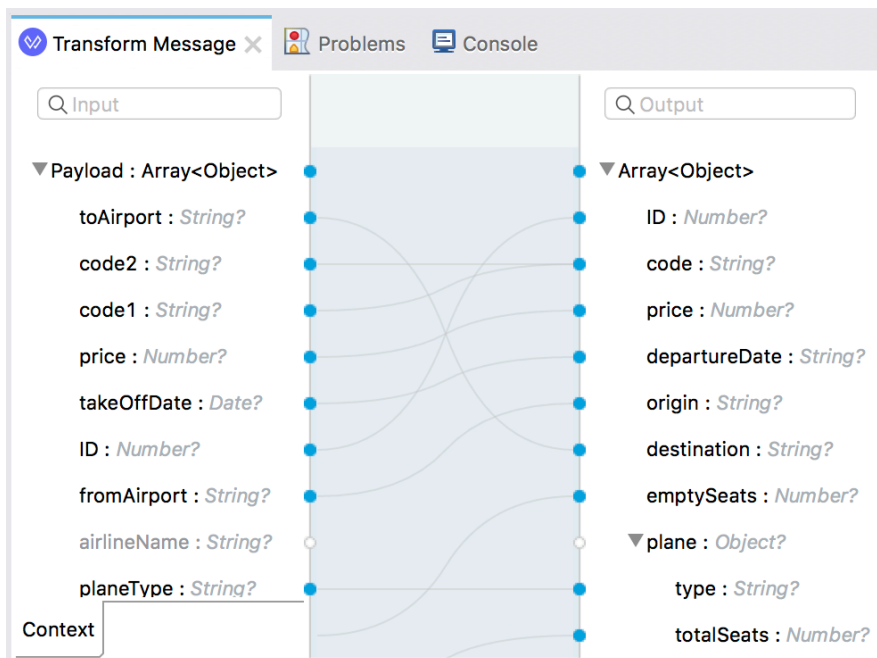


22. Map fields with different names by dragging them from the input section and dropping them on the corresponding field in the output section.

- toAirport to destination
- takeOffDate to departureDate
- fromAirport to origin
- seatsAvailable to emptySeats
- planeType to plane > type

23. Concatenate two fields by dragging them from the input section and dropping them on the same field in the output section.

- code1 to code
- code2 to code



Add sample data (optional)

24. Click the Preview button in the output section.

25. In the preview section, click the Create required sample data to execute preview link.

The screenshot shows the MuleSoft IDE's output section. On the left, a JSON payload is displayed with line numbers 1 through 16. The payload is a map with keys for ID, code, price, departureDate, origin, destination, emptySeats, and plane. The plane key has a nested object with type and totalSeats. On the right, there is a 'Preview' button and a link that says 'Create required sample data to execute preview'.

```
1 %dw 2.0
2   output application/json
3   ---
4   payload map ( payload01 , index0fPa
5     ID: payload01.ID,
6     code: (payload01.code1 default
7     price: payload01.price,
8     departureDate: payload01.takeOf
9     origin: payload01.fromAirport,
10    destination: payload01.toAirpor
11    emptySeats: payload01.seatsAvai
12    plane: {
13      "type": payload01.planeType
14      totalSeats: payload01.total
15    }
16  }
```

26. Look at the input section, you should see a new tab called payload with sample data generated from the input metadata.

27. Look at the output section, you should see a sample response for the transformation.

The screenshot shows the MuleSoft IDE's input and output sections. On the left, the input section shows a JSON payload with line numbers 1 through 16. The payload is a map with keys for toAirport, code2, code1, price, takeOffDate, ID, fromAirport, airlineName, planeType, seatsAvailable, and totalSeats. On the right, the output section shows a sample response for the transformation, which is a JSON array containing one object with keys for ID, code, price, departureDate, origin, destination, emptySeats, plane, and totalSeats. The output section also has a 'Preview' button and a link that says 'Create required sample data to execute preview'.

```
%dw 2.0
output application/java
---
[
  {
    toAirport: "????",
    code2: "????",
    code1: "????",
    price: 2,
    takeOffDate: |2003-10-01|,
    ID: 2,
    fromAirport: "????",
    airlineName: "????",
    planeType: "????",
    seatsAvailable: 2,
    totalSeats: 2
  }
]
```

Context: payload

Output: Payload

Array<Object>

- ID : Number?
- code : String?
- price : Number?
- departureDate : String?
- origin : String?
- destination : String?
- emptySeats : Number?
- plane : Object?
- type : String?
- totalSeats : Number?

```
[
  {
    "ID": 2,
    "code": "????????",
    "price": 2,
    "departureDate": "2003-10-01",
    "origin": "????",
    "destination": "????",
    "emptySeats": 2,
    "plane": {
      "type": "????",
      "totalSeats": 2
    }
  }
]
```


28. In the input section, replace all the ??? with sample values.

29. Look at the output section, you should see the sample values in the transformed data.

Transform Message x Problems Console

list_dwl

%dw 2.0
output application/java

[
 toAirport: "ORD",
 code2: "fdss",
 code1: "4334",
 price: 799,
 takeOffDate: |2018-10-01|,
 ID: 1,
 fromAirport: "SFO",
 airlineName: "american",
 planeType: "Boeing 747",
 seatsAvailable: 1,
 totalSeats: 345
]

Context payload

Output Payload

▼ Array<Object>

ID : Number?

code : String?

price : Number?

departureDate : String?

origin : String?

destination : String?

emptySeats : Number?

▼ plane : Object?

type : String?

totalSeats : Number?

```
[  
  {  
    "ID": 1,  
    "code": "4334fdss",  
    "price": 799,  
    "departureDate": "2018-10-01",  
    "origin": "SFO",  
    "destination": "ORD",  
    "emptySeats": 1,  
    "plane": {  
      "type": "Boeing 747",  
      "totalSeats": 345  
    }  
  }  
]
```

Test the application

30. Save the file to redeploy the project.

31. In Advanced REST Client, make another request to <http://localhost:8081/flights>; you should see all the flight data as JSON again but now with a different structure.

Method GET Request URL http://localhost:8081/flights SEND

Parameters

200 OK 1123.12 ms DETAILS

```
[Array[11]]  
-0: {  
  "ID": 1,  
  "code": "rree0001",  
  "price": 541,  
  "departureDate": "2016-01-19T16:00:00",  
  "origin": "MUA",  
  "destination": "LAX",  
  "emptySeats": 0,  
  "plane": {  
    "type": "Boeing 787",  
    "totalSeats": 200  
  }  
},
```

Try to retrieve information about a specific flight

32. Add a URI parameter to the URL to make a request to <http://localhost:8081/flights/3>; you should get a 404 Not Found response with a no listener message.

The screenshot shows an HTTP client interface. At the top, there are two input fields: 'Method' with 'GET' selected and 'Request URL' with 'http://localhost:8081/flights/3'. To the right of these fields is a blue 'SEND' button and a vertical ellipsis menu. Below the input fields is a 'Parameters' section with a dropdown arrow. The response area shows a status bar with '404 Not Found' in an orange box and '21.24 ms' in grey. To the right of the status bar is a 'DETAILS' link with a dropdown arrow. Below the status bar is a row of icons: a document icon, a download icon, a code icon (highlighted with a grey circle), and an eye icon. Below the icons is the text 'No listener for endpoint: /flights/3'.

Method GET Request URL http://localhost:8081/flights/3 SEND

Parameters

404 Not Found 21.24 ms DETAILS

No listener for endpoint: /flights/3

33. Return to Anypoint Studio.
34. Look at the console; you should get a no listener found for request (GET)/flights/3.