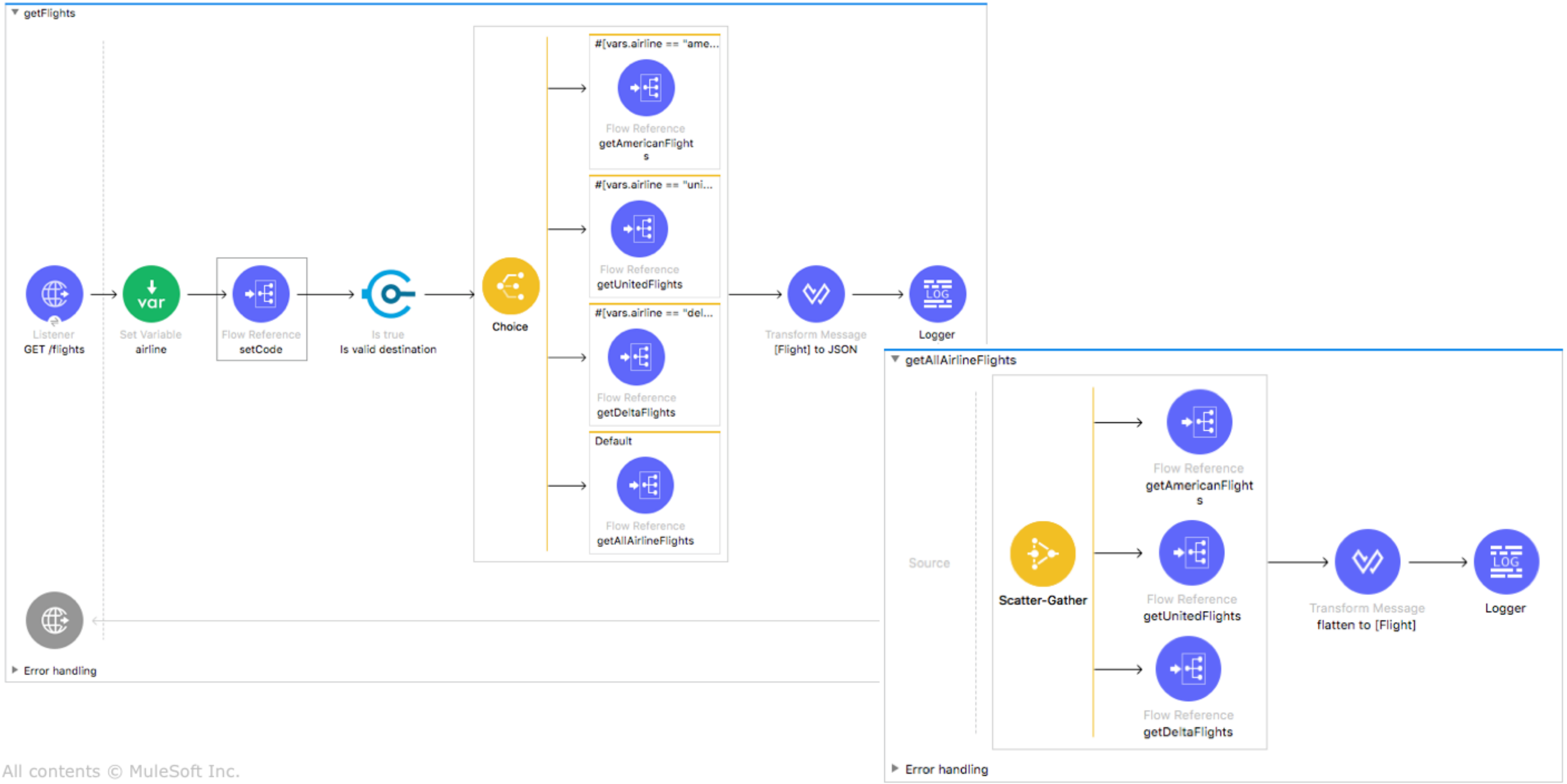
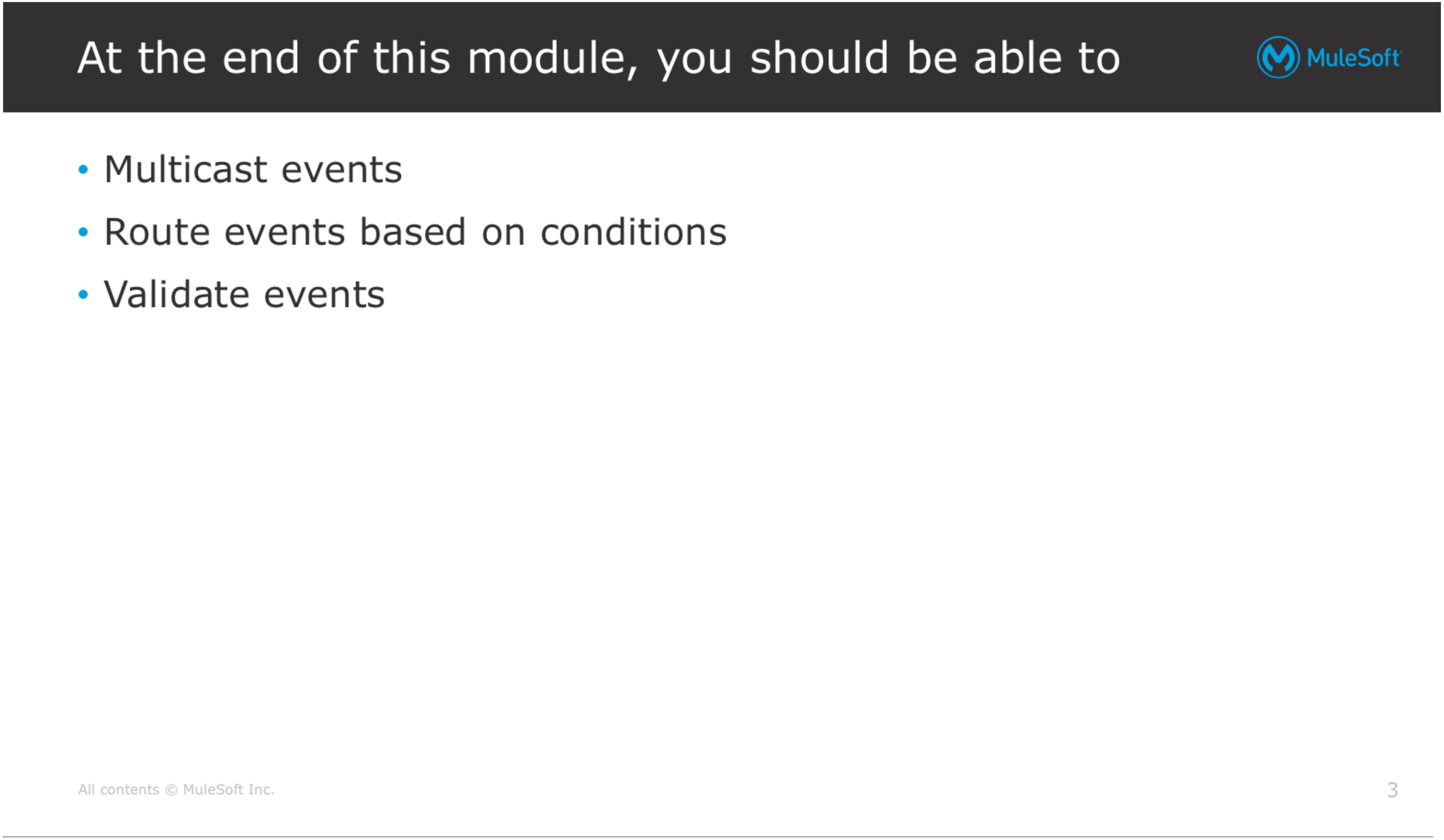
hello

Module 9: Controlling Event Flow

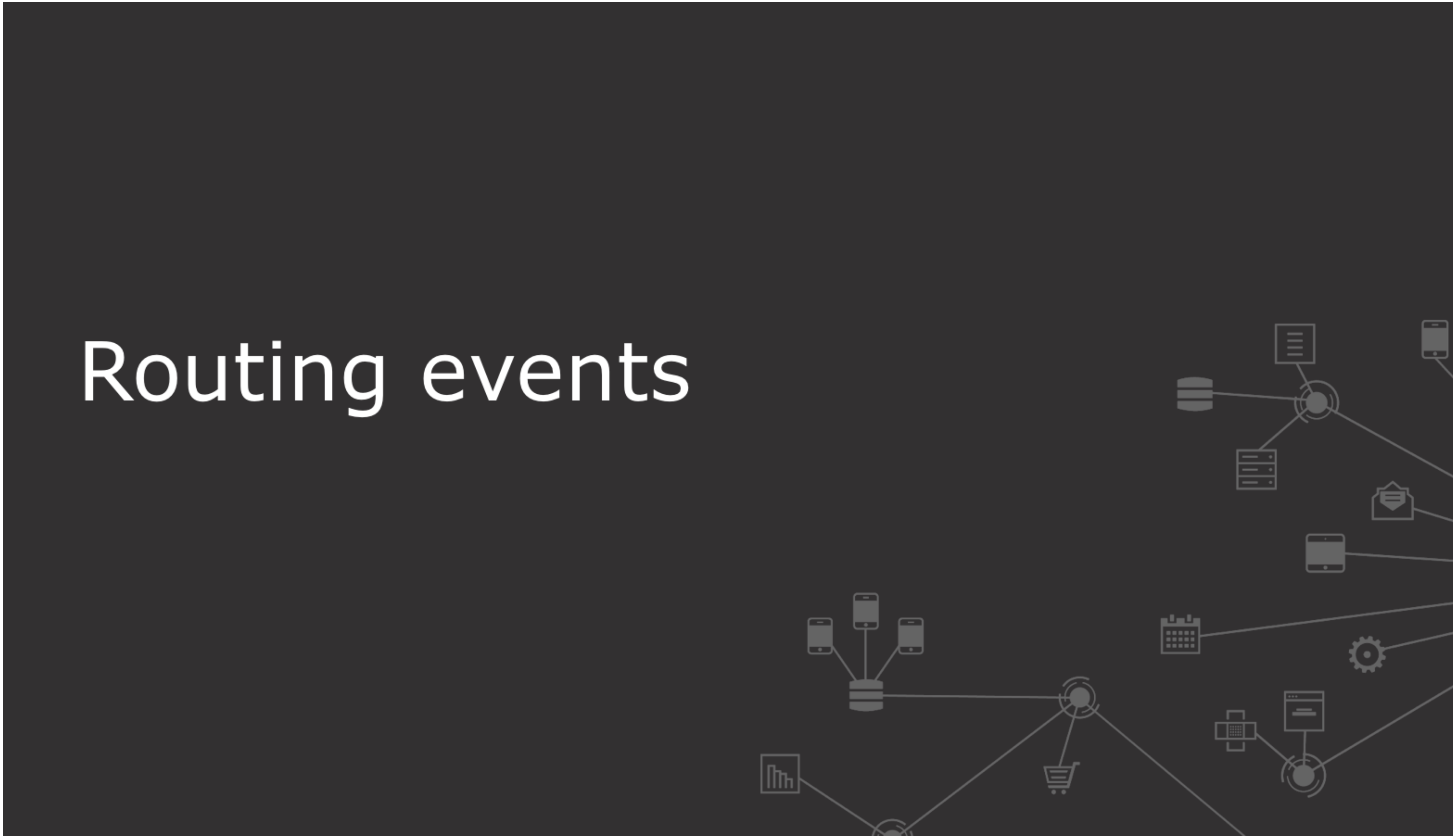
Goal


getFlights
Listener
GET /flights
Set Variable
airline
Flow Reference
setCode
Is true
Is valid destination
Choice
#[vars.airline == "ame...
Flow Reference
getAmericanFlights
#[vars.airline == "uni...
Flow Reference
getUnitedFlights
#[vars.airline == "del...
Flow Reference
getDeltaFlights
Default
Flow Reference
getAllAirlineFlights
Transform Message
[Flight] to JSON
Logger
Error handling
getAllAirlineFlights
Source
Scatter-Gather
Flow Reference
getAmericanFlights
Flow Reference
getUnitedFlights
Flow Reference
getDeltaFlights
Transform Message
flatten to [Flight]
Logger
Error handling



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

- Multicast events
- Route events based on conditions
- Validate events

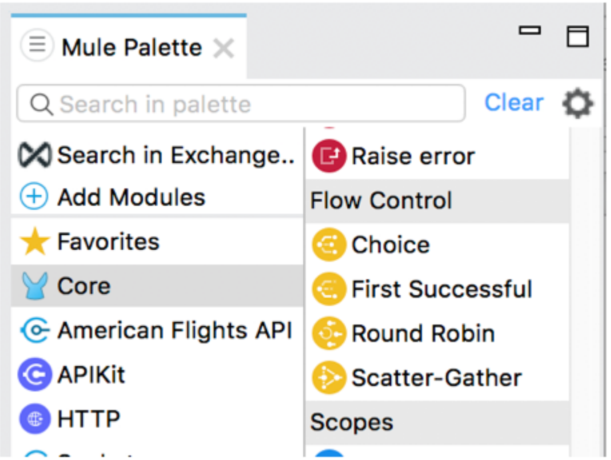


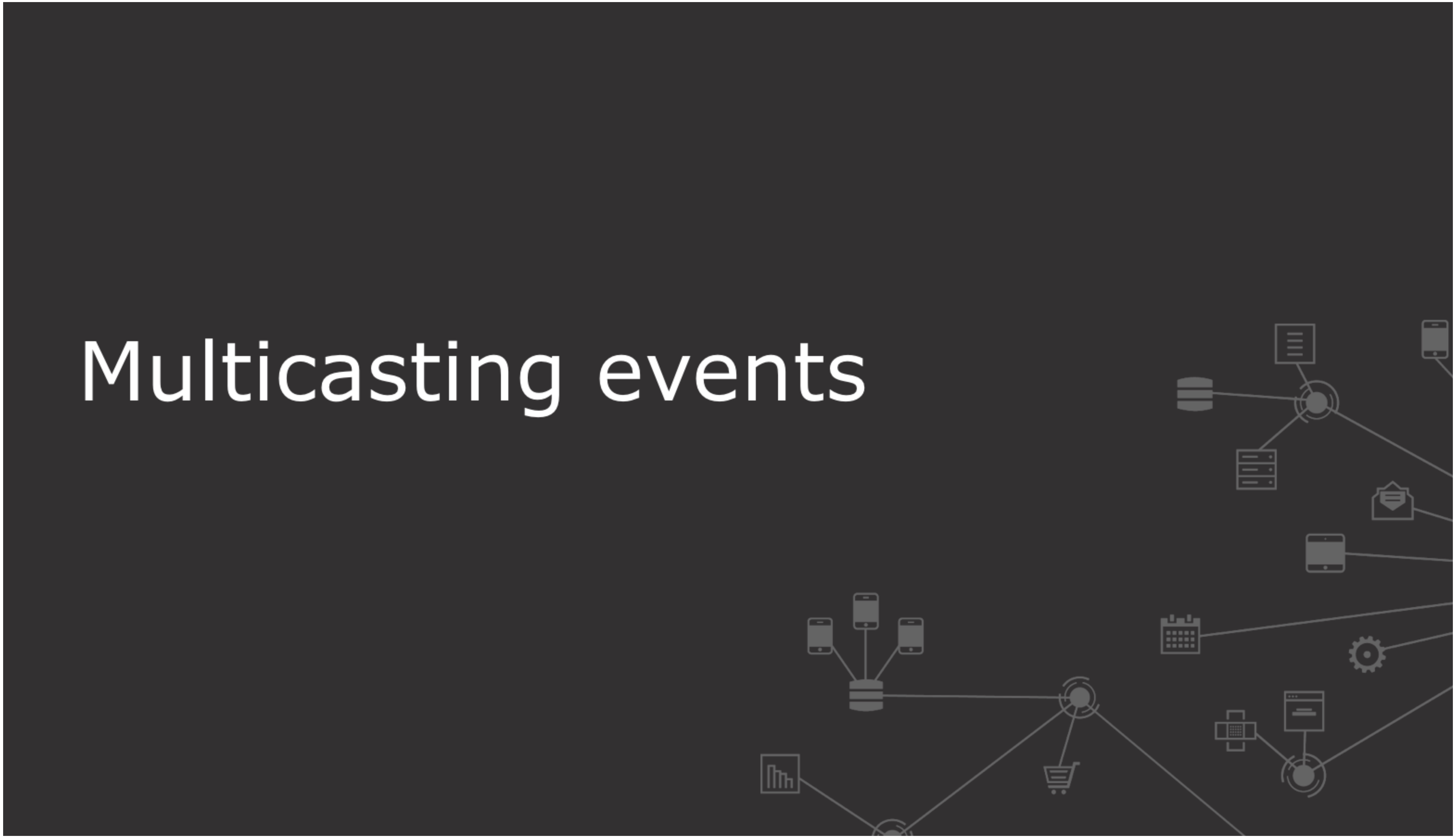
Routing events

Routers

- Routers send events to one or more groups of event processors (routes

- **Choice**
  - One route executed based on conditional logic
- First Successful
  - Routes executed sequentially until one is successfully executed
- Round Robin
  - One route executed, which one is selected by iterating through a list maintained across executions
- **Scatter-Gather**
  - All routes executed concurrently

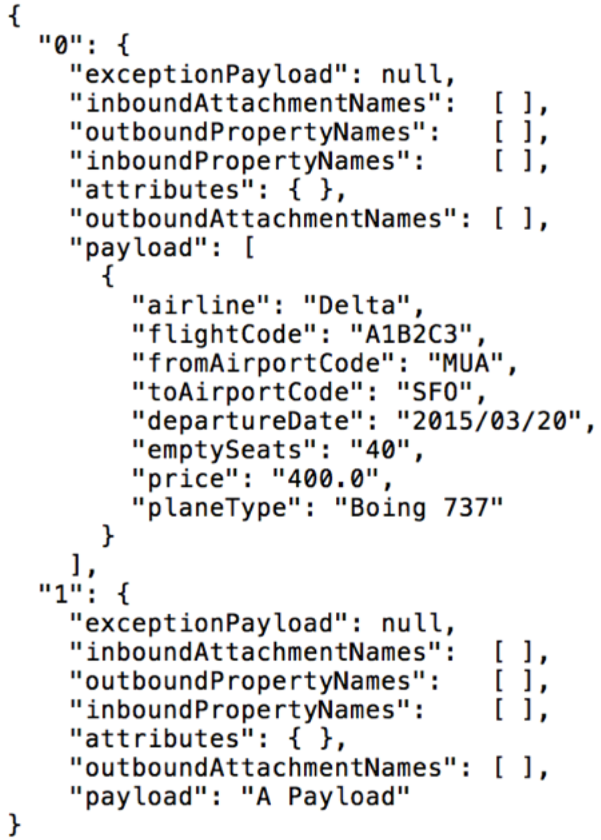




Multicasting events

The Scatter-Gather router

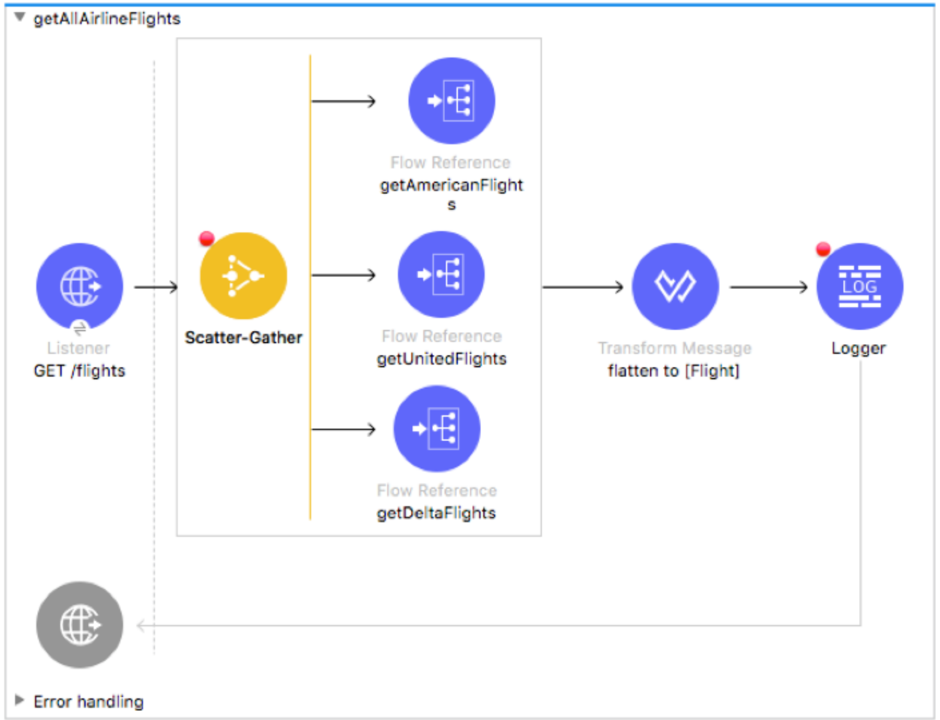
- Scatter-Gather sends the event to each route concurrently and returns a collection of all results
- Collection is an object of objects
  - Each object contains attributes and payload from each Mule event returned from a flow

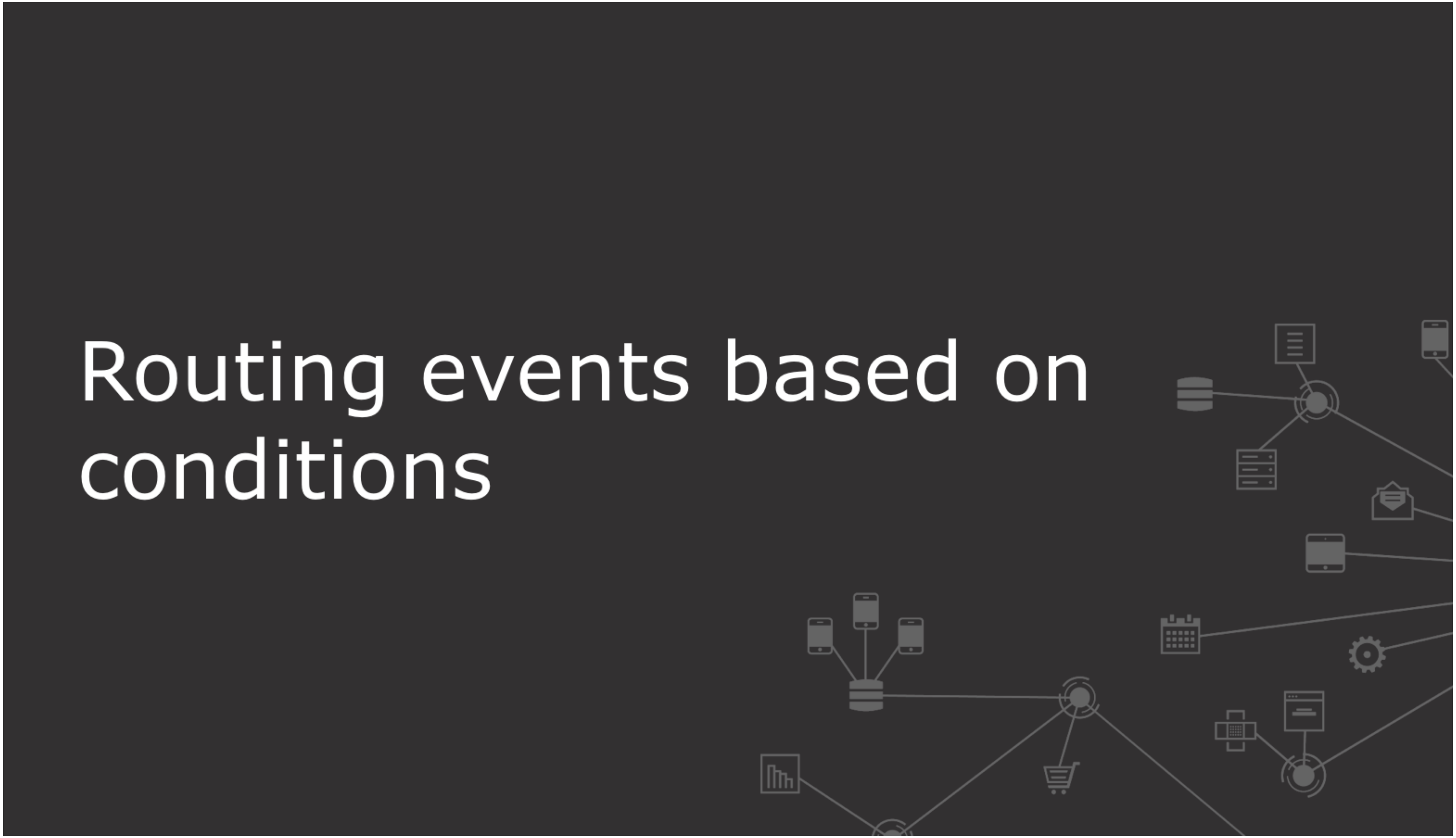


```
{
  "0": {
    "exceptionPayload": null,
    "inboundAttachmentNames":  [ ],
    "outboundPropertyNames":   [ ],
    "inboundPropertyNames":    [ ],
    "attributes": { },
    "outboundAttachmentNames": [ ],
    "payload": [
      {
        "airline": "Delta",
        "flightCode": "A1B2C3",
        "fromAirportCode": "MUA",
        "toAirportCode": "SFO",
        "departureDate": "2015/03/20",
        "emptySeats": "40",
        "price": "400.0",
        "planeType": "Boing 737"
      }
    ],
  "1": {
    "exceptionPayload": null,
    "inboundAttachmentNames":  [ ],
    "outboundPropertyNames":   [ ],
    "inboundPropertyNames":    [ ],
    "attributes": { },
    "outboundAttachmentNames": [ ],
    "payload": "A Payload"
}
```

Walkthrough 9-1: Multicast an event

- Use a Scatter-Gather router to concurrently call all three flight services
- Use DataWeave to flatten multiple collections into one collection

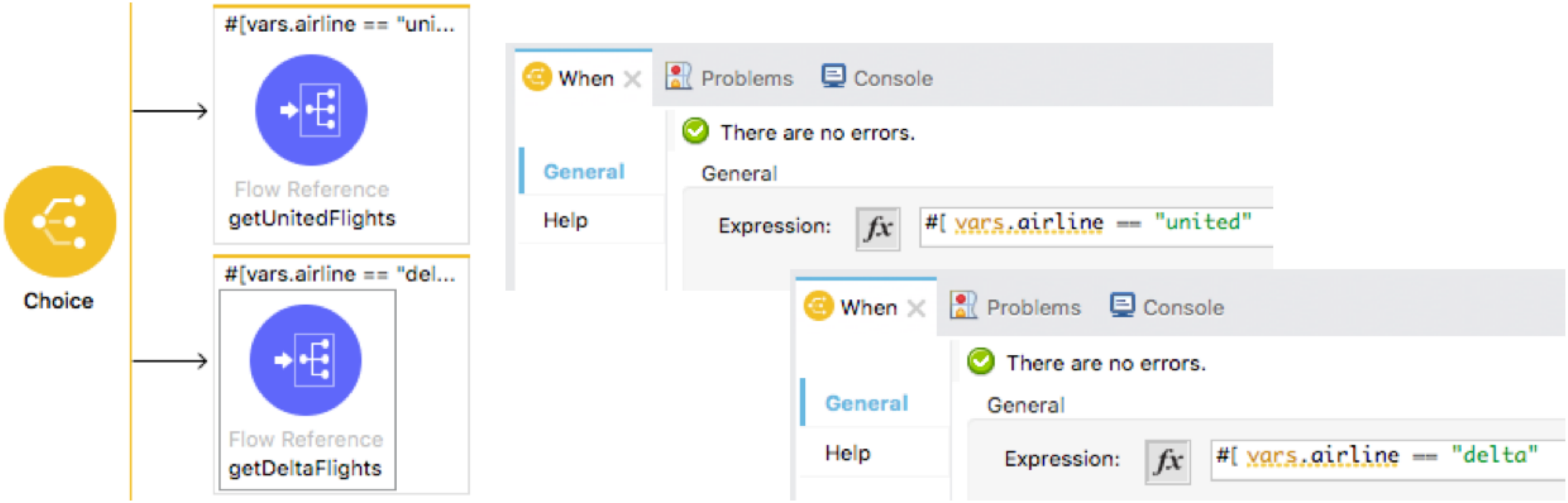




Routing events based on conditions

The Choice router

- Sends the event to one route based on conditional logic
- The conditions are written with DataWeave



There are no errors.

General

Expression: #[vars.airline == "united"

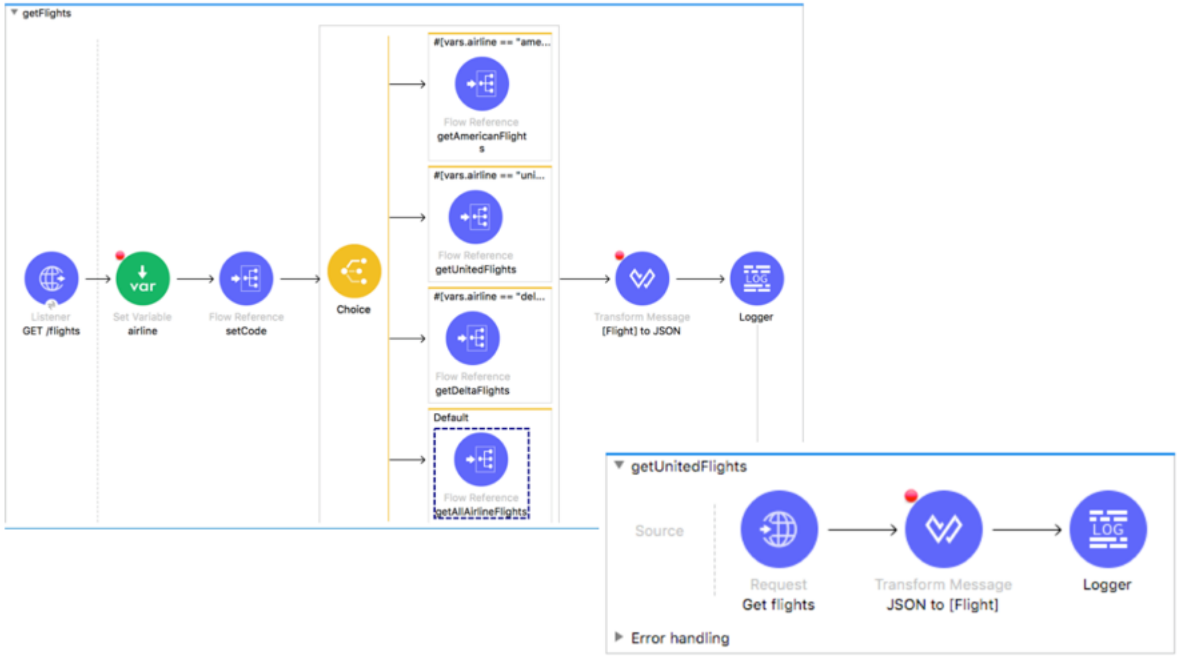
There are no errors.

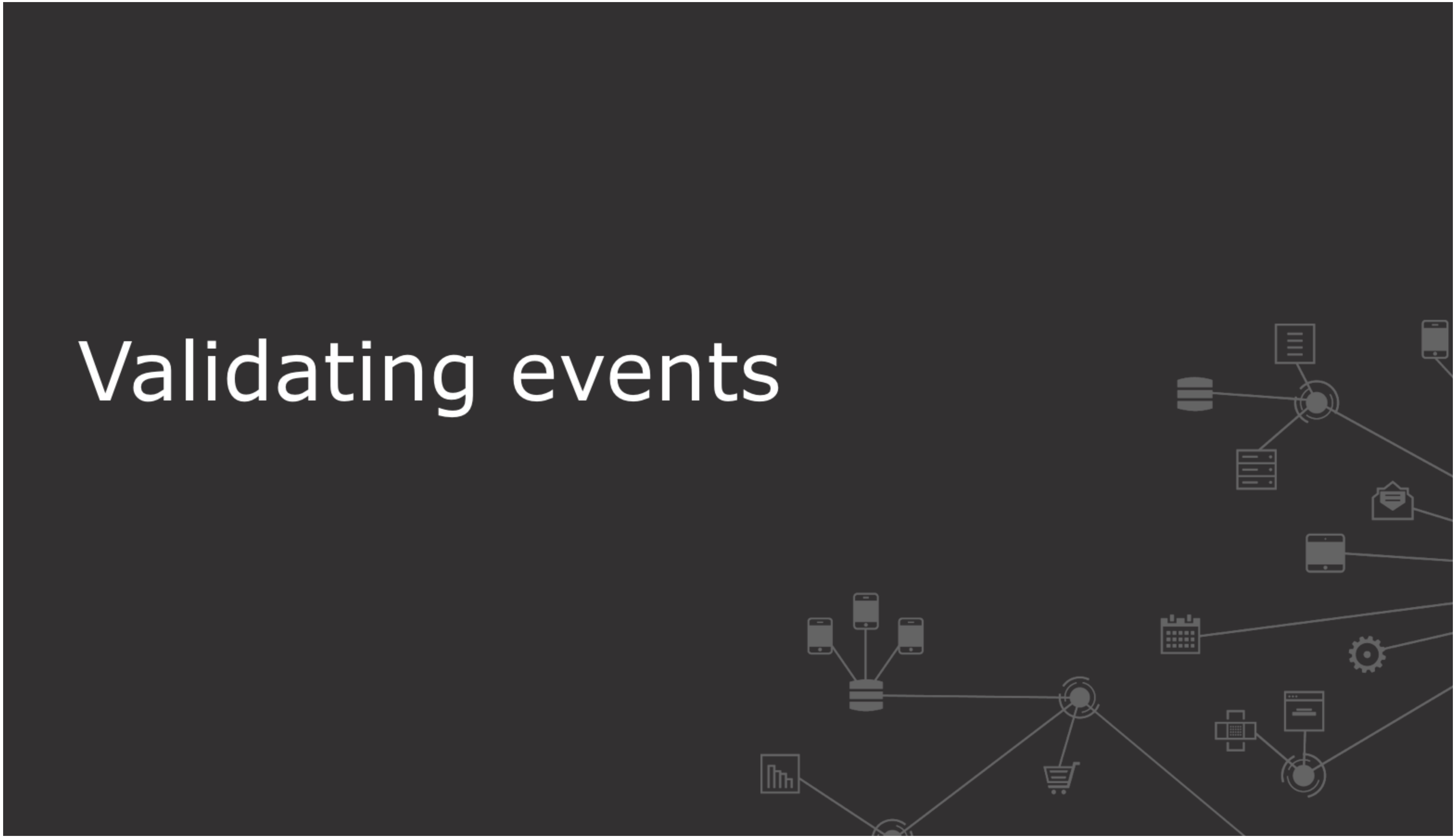
General

Expression: #[vars.airline == "delta"

Walkthrough 9-2: Route events based on conditions

- Use a Choice router
- Use DataWeave expressions to set the router paths
- Route all flight requests through the router



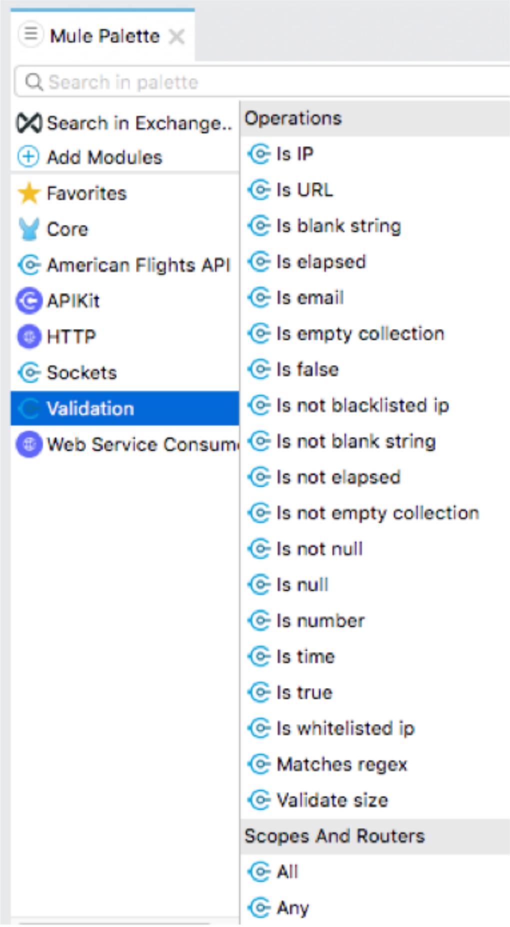


Validating events

Validators

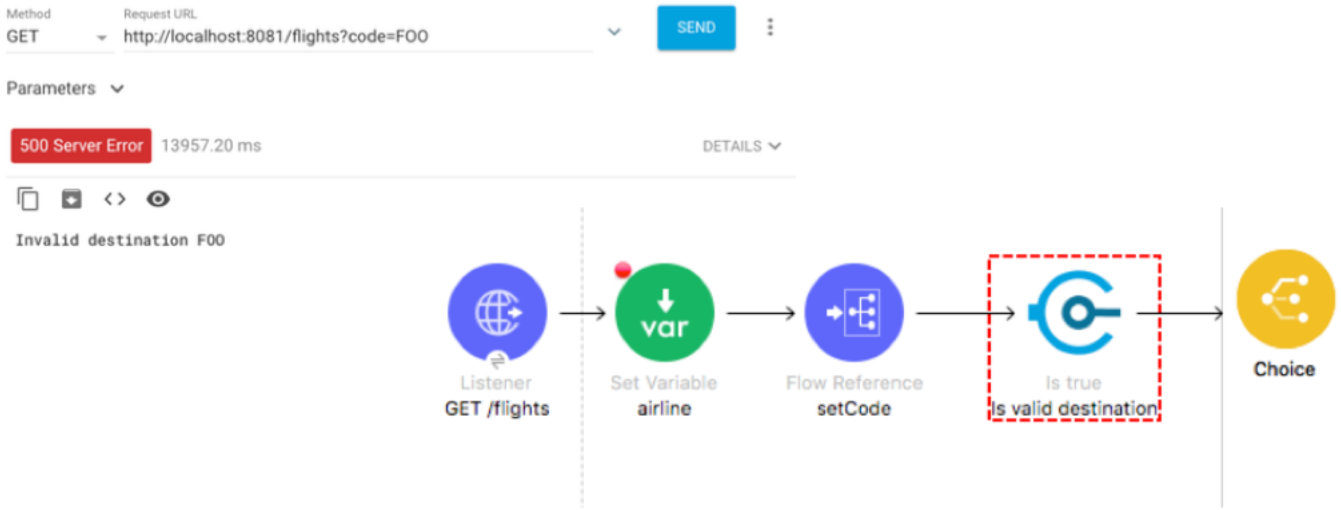
- Provide a way to test some conditions are met and throw an error if the validation fails

- To use
  - Add the Validation module to a project
  - Select a validation operation



Walkthrough 9-3: Validate events

- Add the Validation module to a project
- Use an Is true validator to check if a query parameter called code with a value of SFO, LAX, CLE, PDX, or PDF is sent with a request
- Return a custom error message if the condition is not met



Method: GET

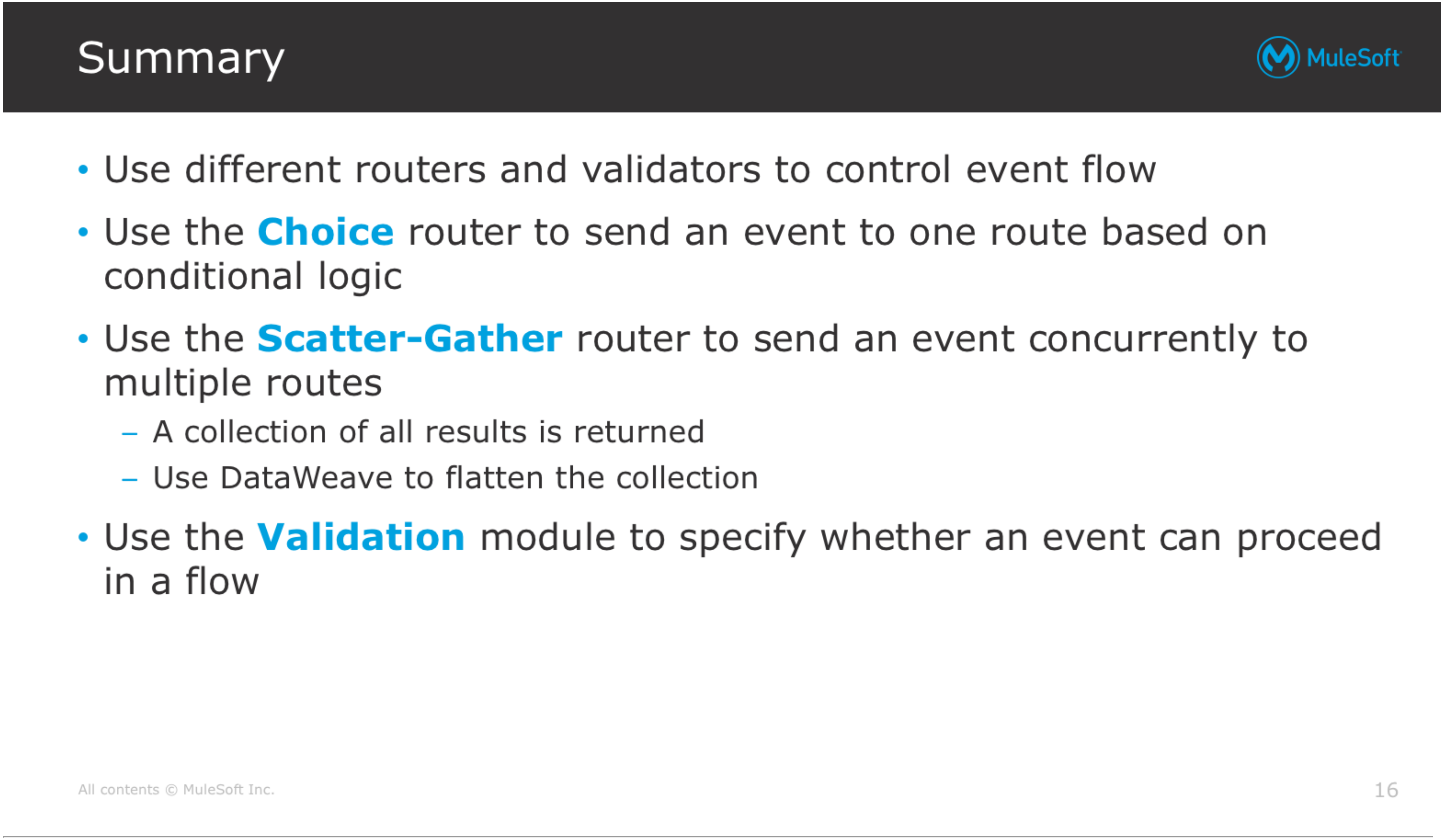
Request URL: http://localhost:8081/flights?code=FOO

500 Server Error 13957.20 ms

```
Invalid destination FOO
```

All contents © MuleSoft Inc.

Summary



Summary

- Use different routers and validators to control event flow
- Use the **Choice** router to send an event to one route based on conditional logic
- Use the **Scatter-Gather** router to send an event concurrently to multiple routes
  - A collection of all results is returned
  - Use DataWeave to flatten the collection
- Use the **Validation** module to specify whether an event can proceed in a flow