Introduction to AsyncAPI for Kafka

Lorna Mitchell, Aiven

Introducing: AsyncAPI

Specification for describing event-driven and data streaming systems.

- https://asyncapi.com
- Open standard
- Active community
- Supports Kafka! And MQTT, AMQP, WS ....

Why use AsyncAPI?

Describe your event-driven systems in a useful and reusable way.

- Clearly track changes in a text-based description
- Generate documentation and code
- Verify system operations against description
- Enclose existing payload descriptions (CloudEvents, Avro) within AsyncAPI

Bunch of JSON/YAML

```yaml
asyncapi: '2.0.0'
id: 'urn:com.github.lornajane.example1'
info:
  contact:
  license:
servers:
  local-kafka:
channels:
  factorysensor:
    subscribe:
      message:
        payload:
        examples:
components:
  securitySchemes:
  messages:
```

Generated Docs

Operations

SUB `door-sensor`

Door sensors (external and internal)

Open/closed state information from the doors.

Operation Bindings >

#sensor

Accepts the following message:

Door Sensor Reading door-sensor-data

Door sensor data

Payload > Object

Examples

Payload

Example #1

```
{
  "location": "Car park",
  "state": "open"
}
```

Example #2

```
{
  "location": "Roof-level fire exit",
  "state": "closed"
}
```

@lornajane

AsyncAPI Descriptions

AsyncAPI Structure

Top-level elements:

- `asyncapi` **and** `id`
- `info`
- `servers`
- `channels`
- `tags`
- `components`

Info Section

Valuable metadata is held in `info`.

```yaml
info:
  title: Thingum Industries Sensors
  description: Keeping the factory and all the machines running nicely
  version: 1.0.0
  contact:
    name: Lorna
    email: lornajane@aiven.io
    url: https://github.com/aiven/thingum-industries
  license:
    name: Apache 2.0
    url: http://www.apache.org/licenses/LICENSE-2.0.html
```

Channels Section

Main operations are described here

```yaml
channels:
  door-sensor:
    description: Door sensors (external and internal)
    subscribe:
      operationId: DoorSensor
      description: Open/closed state information from the doors.
      tags:
        - name: sensor
      bindings:
        kafka:
          clientId:
            type: string
      message:
        $ref: '#/components/messages/DoorData'
```

$ref Reusable Content

Refer to content in the components section

```
message:
  $ref: '#/components/messages/DoorData'
```

Useful for reuse and readability.

We can also refer to other files:

```
message:
  $ref: 'doors-publish.yaml#/components/messages/DoorData'
```

Components Section

A collection of reusable components

```yaml
components:
  messages:
    DoorData:
      name: door-sensor-data
      title: Door Sensor Reading
      description: Door sensor data
      payload:
        type: object
        properties:
          location:
            type: string
          state:
            enum: ["open", "closed"]
```

AsyncAPI Loves Standards

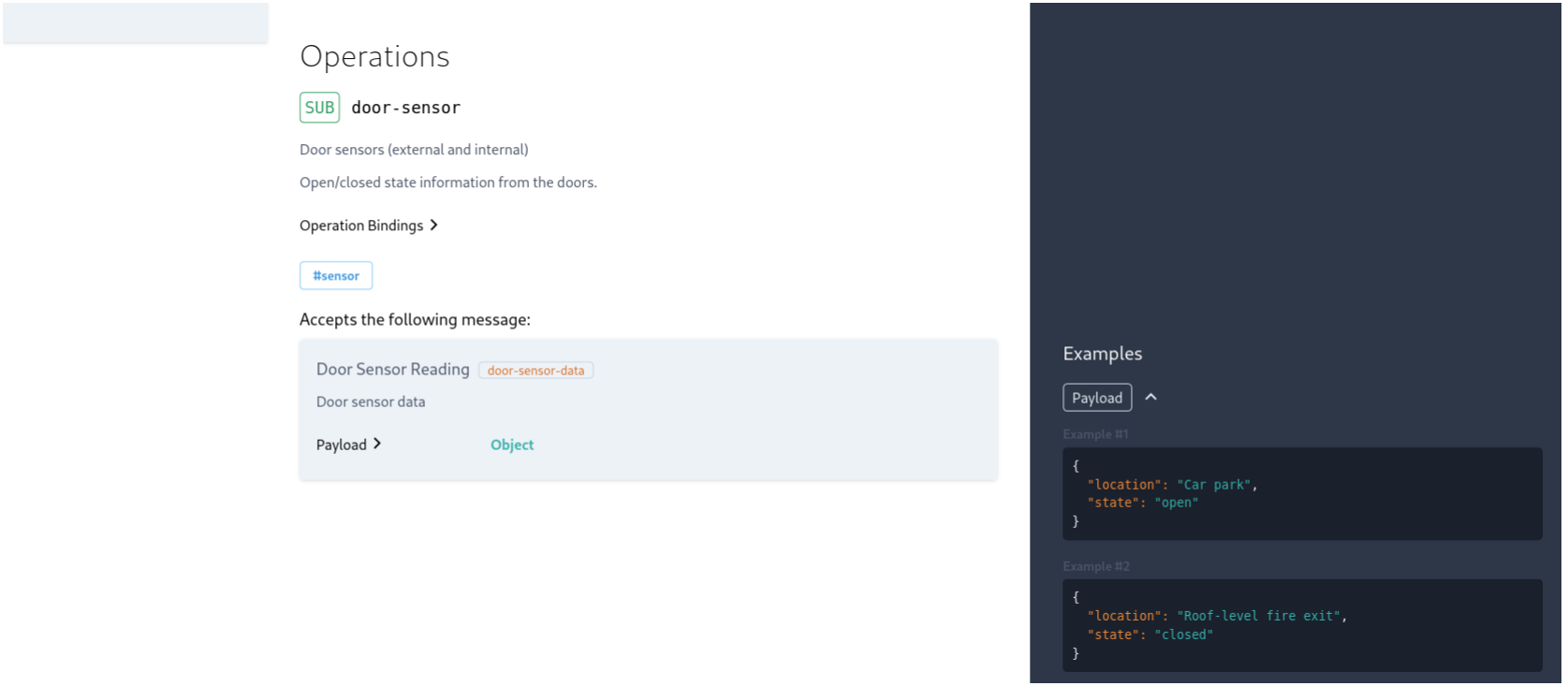
AsyncAPI is compatible with other standards:

- Avro or CloudEvent payload descriptions can be accessed with `$ref`
- JSONSchema is supported
- Based on and builds on OpenAPI

AsyncAPI Tools

Generate Documentation

Documentation with https://www.asyncapi.com/generator



Generate Code

Also using the generator:

```
ag thingum/doors-publish.yaml \
@asyncapi/nodejs-template \
-o thingum/nodejs \
-p server=development
```

Tools Landscape

- VSCode extension: https://github.com/asyncapi/vs-asyncapi-preview
- Validation/linting: https://stoplight.io/open-source/spectral/
- Generate test data: https://microcks.io/

- GitHub action: https://github.com/WaleedAshraf/asyncapi-github-action

AsyncAPI and You

Open standards, seamless integrations

Resources

- AsyncAPI: https://asyncapi.com
- Aiven: https://aiven.io - try the free trial
- Examples: https://github.com/aiven/thingum-industries

@lornajane