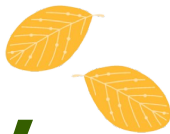


The State of OpenAPI Specification: What's New in 3.1

Lorna Mitchell, Vonage/OpenAPI Initiative

Quick Summary



- An OpenAPI minor release: 3.1
- Full JSON Schema compatibility
- Webhooks are supported
- Many quality-of-life improvements



Version Numbering

- From 3.1, OpenAPI does not follow Semantic Versioning
- Enables edge-case breakage for JSON Schema support



JSON Schema

OpenAPI 3.1 will support JSON Schema 2020-12

JSON Schema Data Types

OpenAPI types are aligned with JSON Schema types

- string
- number
- object
- array
- boolean
- null



JSON Schema Array of Types

- For fields that can have more than one type
- Arrays of types are supported
- Including **null**
- Breaking change: replaces **nullable** keyword

```
parameters:  
  - name: friendly-label  
    in: query  
    schema:  
      type: string  
      nullable: true
```

```
parameters:  
  - name: friendly-label  
    in: query  
    schema:  
      type:  
        - string  
        - null
```

JSON Schema Examples

- In schemas, recommend **examples** array, rather than **example** with a single element
- Both are valid in 3.1

```
schema: # 3.0 but valid in 3.1
  required:
    - name
  properties:
    name:
      description: "Item name"
      type: "string"
      example: "Item One"
```

```
schema: # Recommended in 3.1
  required:
    - name
  properties:
    name:
      description: "Item name"
      type: "string"
      examples:
        - "Item One"
```

Examples and Examples

- Look out for existing OpenAPI examples
- Used in Media Types section
- A map with named keys, rather than an array of values





Examples and Examples

```
responses:
  "400":
    content:
      application/json:
        examples:
          throttled:
            summary: Limit exceeded
            value:
              status: "1"
              error_text: Throttled
          concurrent:
            summary: Request in progress
            value:
              request_id: abcdef0123456789abcdef0123456789
              status: "10"
              error_text: Cannot start request, already in progress
```



JSON Schema ... Schemas

- Arbitrary keywords are supported

```
components:  
  schemas:  
    style:  
      type: object  
      properties:  
        template_id:  
          type: number  
        hue:  
          type: string  
        decoration: sparkly
```



JSON Schema ... Schemas

- Siblings are allowed alongside `$ref`

```
content:
  'application/json':
    schema:
      $ref: '#/components/schemas/style'
      required:
        - hue
```



Webhooks

Built like callbacks

Callbacks Example

- Valid in 3.0 and 3.1

```
parameters:
  - $ref: "#/components/parameters/callback"
responses: ...
callbacks:
  onData:
    "{$request.query.callback}":
      post:
        operationId: asyncCallback
        requestBody:
          content:
            application/json: ...
        responses: ...
```

Webhooks Example

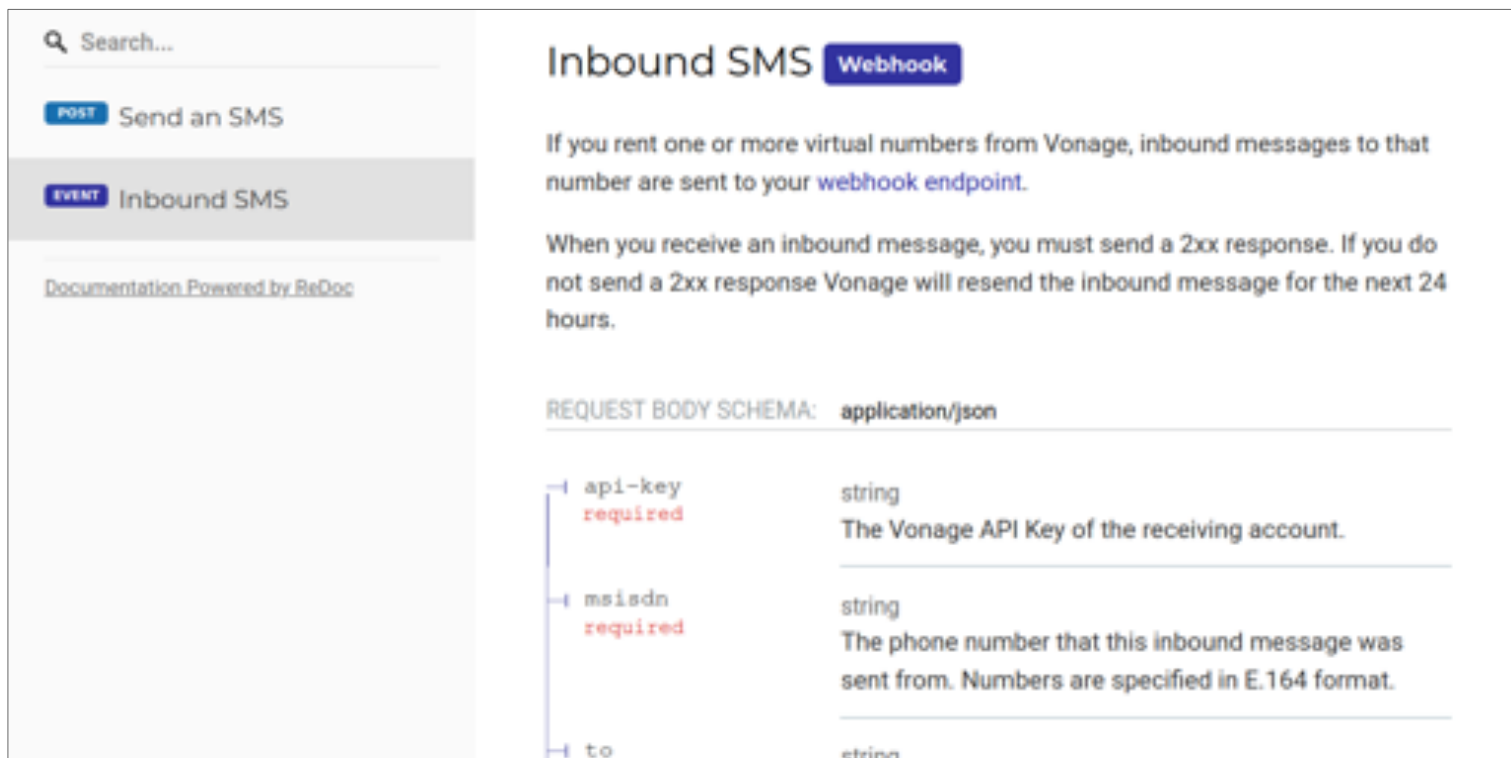
- Newly introduced in 3.1

```
webhooks:  
  inbound-sms:  
    post:  
      operationId: inbound-sms  
      requestBody:  
        content:  
          application/json: ...  
      responses: ...
```

Webhooks Preview



- Use **x-webhooks** in 3.0 to see it working in Redoc



The screenshot shows the Redoc documentation for the 'Inbound SMS' endpoint. The left sidebar contains a search bar and two menu items: 'POST Send an SMS' and 'EVENT Inbound SMS', with the latter being selected. The main content area is titled 'Inbound SMS' with a 'Webhook' tag. It includes a description of the endpoint's purpose, a note about the required 2xx response, and a 'REQUEST BODY SCHEMA' section showing a JSON schema with three required fields: 'api-key', 'msisdn', and 'to'.

Search...

POST Send an SMS

EVENT Inbound SMS

Documentation Powered by Redoc

Inbound SMS Webhook

If you rent one or more virtual numbers from Vonage, inbound messages to that number are sent to your [webhook endpoint](#).

When you receive an inbound message, you must send a 2xx response. If you do not send a 2xx response Vonage will resend the inbound message for the next 24 hours.

REQUEST BODY SCHEMA: `application/json`

api-key	string
required	The Vonage API Key of the receiving account.
msisdn	string
required	The phone number that this inbound message was sent from. Numbers are specified in E.164 format.
to	string

OpenAPI Top-Level Elements 3.0

Required:

- `openapi`
- `info`
- `paths`

Optional:

- `servers`
- `tags`
- `security`
- `externalDocs`
- `components`



OpenAPI Top-Level Elements 3.0

Required:

- openapi
- info
- paths

Optional:

- servers
- tags
- security
- externalDocs
- components



OpenAPI Top-Level Elements 3.1

Required:

- openapi
- info

One of:

- paths
- webhooks
- components

Optional:

- servers
- tags
- security
- externalDocs



OpenAPI Top-Level Elements 3.1

Required:

- openapi
- info

One of:

- paths
- webhooks
- components

Optional:

- servers
- tags
- security
- externalDocs



Callback or Webhook?

- Overlapping concepts
- Example: subscription API call, later push events
 - a callback described within the subscription/configuration API call
 - a webhook alongside the paths that are functionally similar
 - many incoming HTTP requests could be described either way



Components Section

- 3.1 supports **pathItems** in **components**, to support webhooks and more use of callbacks
- Supported components section keys:
 - schemas
 - responses
 - parameters
 - examples
 - requestBodies
 - headers
 - securitySchemas
 - links
 - callbacks
 - pathItems



OpenAPI Proposal Process

- Open proposal process
- Submit detailed proposal
- Attend meetings for feedback and champion your idea



OpenAPI 3.1: Minor Improvements

Better \$ref Benefits

- \$ref can now have summary and description as siblings

```
paths:
  /items:
    post:
      parameters:
        - $ref: '#/components/parameters/item'
          description: The specific item in question
```


Info Section Upgrade

- Add **summary** alongside **title** and **description**
- **Summary** is shown in list view, a brief summary
- **Description** is used in detail view and is more detailed



License SPDX Identifier

- 3.0: `license` includes `name` and optional `url`
- 3.1: `license` includes `name` and optional `url` or `identifier` fields
- <https://spdx.org/licenses/>



MutualTLS Security Scheme

- 3.1 adds support for `mutualTLS` as a value for the security scheme type field

```
components:  
  securitySchemes:  
    bearerAuth:  
      type: mutualTLS  
      scheme: mutual
```



Request Bodies for any Method

- 3.0 does not allow `requestBody` with GET or HEAD requests
- 3.1 allows it with any request
- *Just because you can, doesn't mean you should*



Responses are Optional

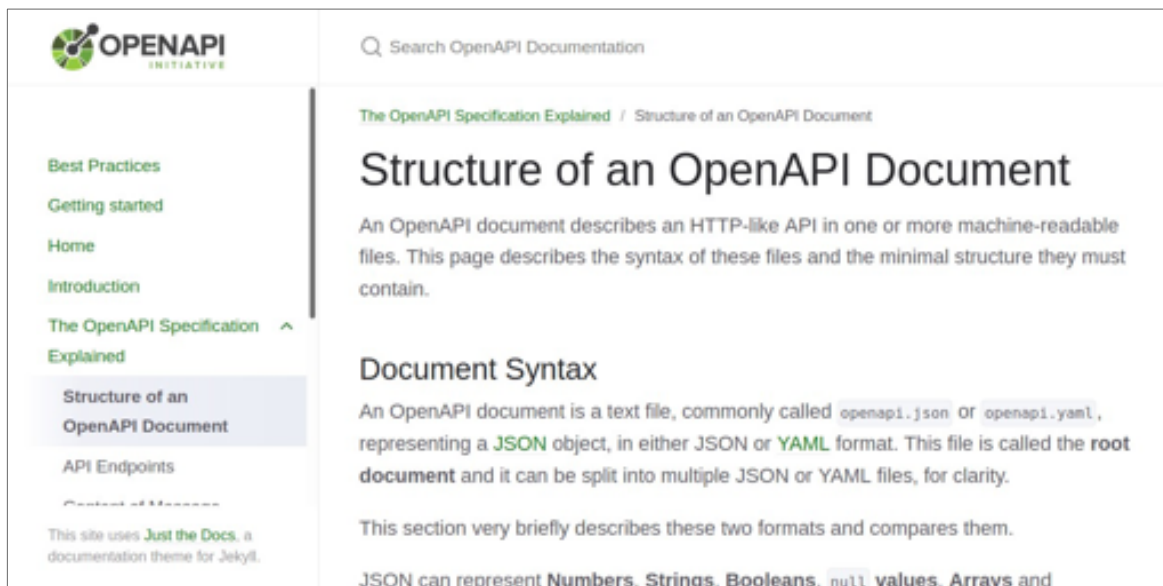
- Your OpenAPI 3.1 descriptions are valid without a **responses** field for every endpoint
- Useful during development



OpenAPI: Community Update

OpenAPI Documentation Project

- Getting started needs more than the spec
- Documentation initiative is well underway
 - <https://oai.github.io/Documentation/>



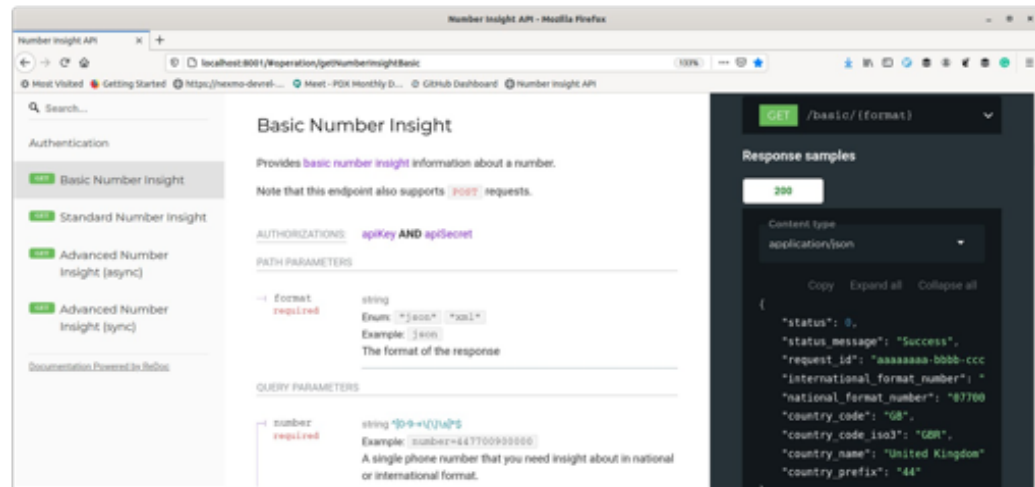
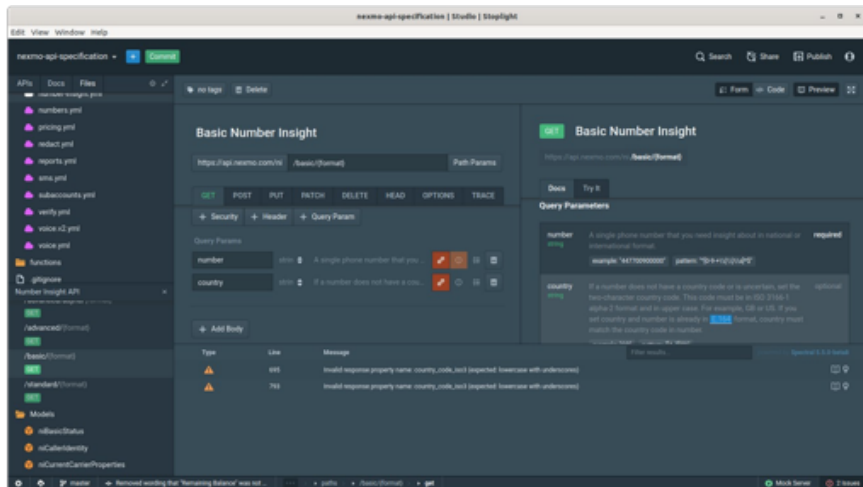
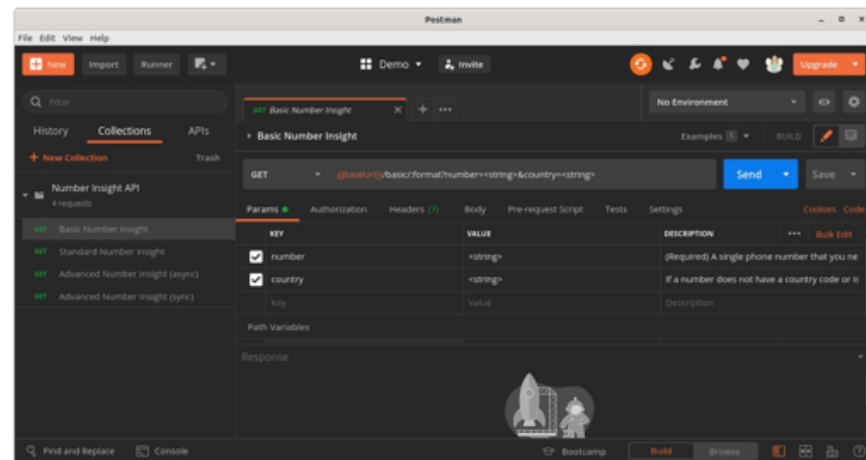
OpenAPI Involvement

- Everything happens in the open
 - Issues and PRs
- <https://github.com/OAI/OpenAPI-Specification>
- Regular meetings
 - Meeting agenda is a GitHub issue
- Proposal process



OpenAPI Tooling

- Tools improving all the time
 - <https://openapi.tools/>



State of OpenAPI

- **Spec:** Innovating, new release 3.1
- **Docs:** Coming along nicely
- **Tools:** Levelling up, many more integrations
- **Events:** API Specifications Conference was in September
- **Future:** More meetings, get involved



Resources

- <https://github.com/OAI/OpenAPI-Specification/blob/3.1.0-rc1/versions/3.1.0.md>
- <https://github.com/OAI/OpenAPI-Specification/releases>
- <http://json-schema.org/>
- <https://oai.github.io/Documentation/>
- <https://github.com/OAI/OpenAPI-Specification/>
- https://www.youtube.com/playlist?list=PLcx_iGeB-Nxil4S7-0Y1Y5r0oLahy3f0Y

