Ben Ramsey • PHP Tek • May 20, 2026

Describe Your API with OpenAPI

Austin Burleson / Unsplash

You shipped an API.

Someone needs to use it.

“Where’s the documentation?”

Josh Marty / Unsplash

Hans Vivek / Unsplash

Well, there’s a Confluence page.

It’s from two years ago.

Half of it is wrong.

Or you’re the *consumer*.

Guessing at field names.

Hoping the `400` response tells you something useful.

Lee Cartledge / Unsplash

What if your API had a **contract**?

- Machine-readable
- Human-readable
- Single source of truth for docs, code generation, testing, and validation


OPENAPI

A brief history

Year	Event
2011	Swagger 1.0 (SmartBear Software)
2014	Swagger 1.2 published; Swagger 2.0 follows
2015	SmartBear donates Swagger to the OpenAPI Initiative
2017	OpenAPI Specification 3.0.0
2021	OpenAPI Specification 3.1.0
2025	OpenAPI 3.1.2 and 3.2.0

“Swagger” is not the spec.

Swagger is a set of tools from SmartBear:

- Swagger UI
- Swagger Editor
- Swagger Codegen

The spec is **OpenAPI**.

Heidi Kaden / Unsplash

What is an OpenAPI Description?

JiaJia Yan / Pexels

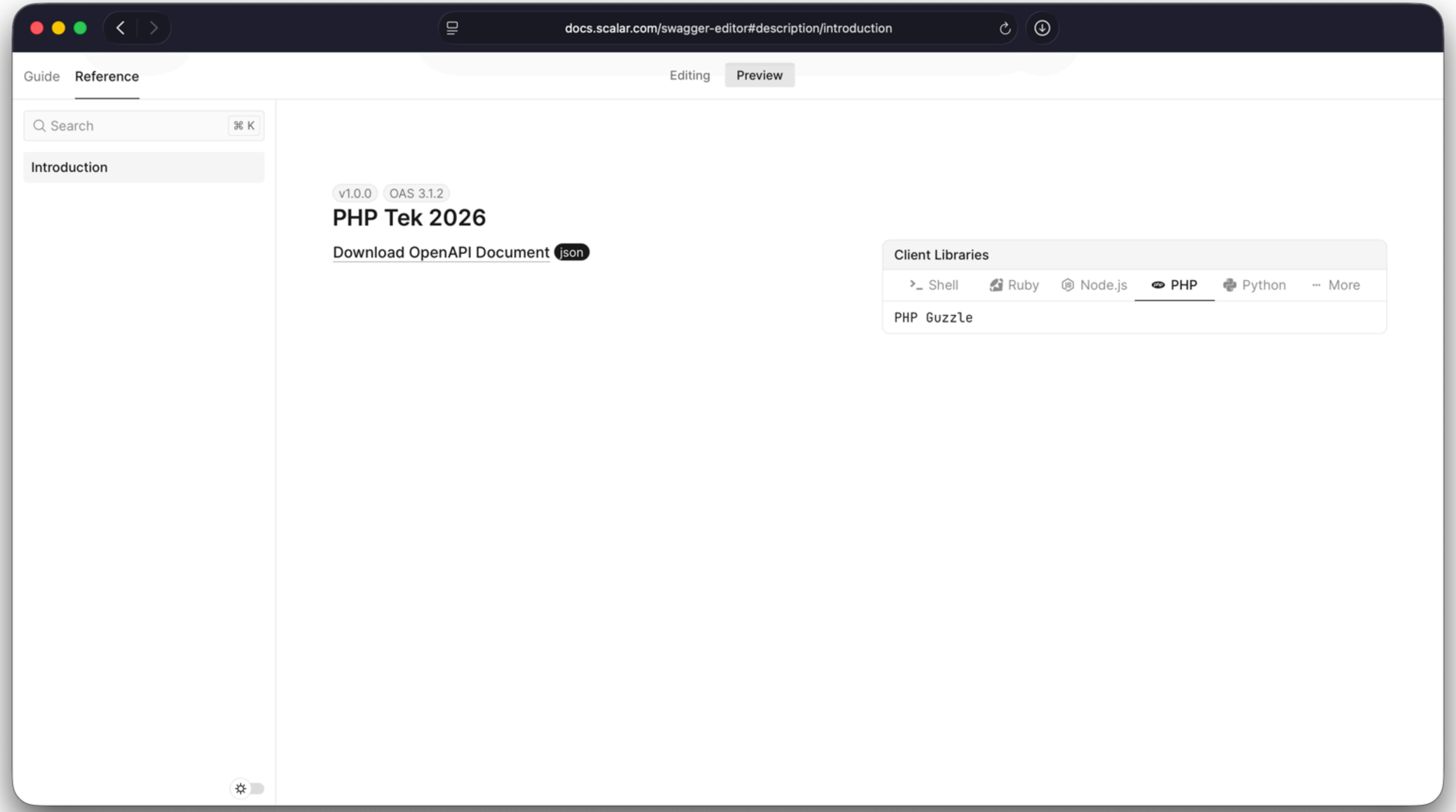
Connor Scott McManus / *Pexels*

An OpenAPI Description (or OAD) is a **text file**.

YAML or JSON.

That’s it.

```yaml
openapi: "3.1.2"
info:
  title: PHP Tek 2026
  version: "1.0.0"
paths: {}
```


docs.scalar.com/swagger-editor#description/introduction
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
v1.0.0
OAS 3.1.2
PHP Tek 2026
Download OpenAPI Document
json
Client Libraries
Shell
Ruby
Node.js
PHP
Python
More
PHP Guzzle

YAML vs. JSON

	YAML	JSON
Hand-authoring	Preferred	Verbose
Generated output	Fine	Fine
Recommend filename	openapi.yaml	openapi.json

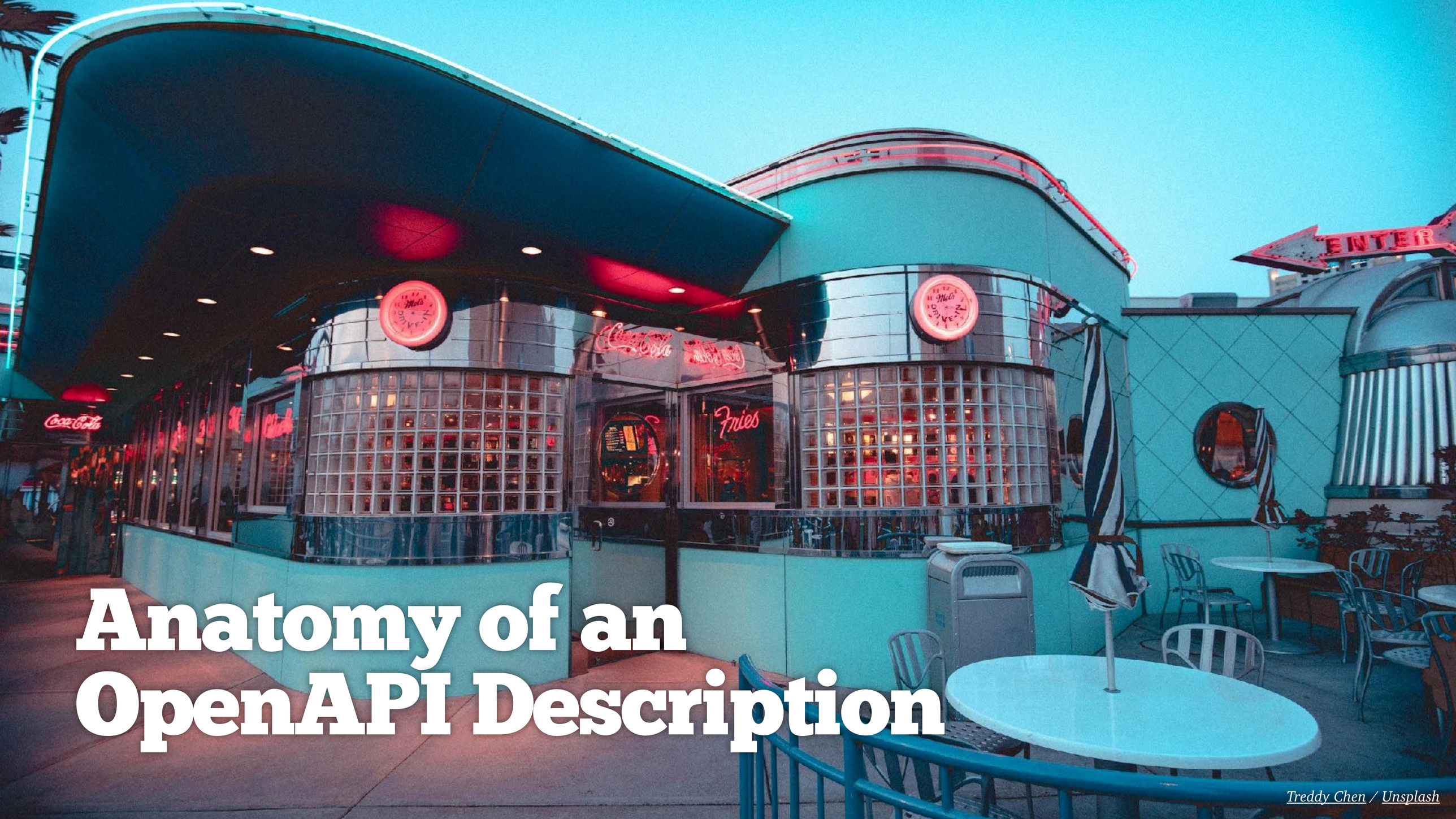
정규송 Nui MALAMA / Pexels

One File or Many

An OpenAPI Description can live in **one file** or **be split across many**.

References (`$ref`) connect them.

Either way, it describes the same API.



Anatomy of an OpenAPI Description

Treddy Chen / Unsplash

```yaml
openapi: "3.1.2"
info:  # required
  ...
servers:
  - ...
tags:
  - ...
paths:  # required (or webhooks/components)
  /sessions:
    ...
components:
  ...
```

```yaml
info:
  title: PHP Tek Conference API
  version: "2026-05"
  summary: Sessions, speakers, and schedule
  description: |
    Provides access to **session** schedules,
    **speaker** profiles, and **room** assignments.
  contact:
    name: PHP Tek Support
    url: https://phptek.io
    email: support@phparch.com
```

```yaml
info:
  title: PHP Tek Conference API
  version: "2026-05"
  summary: Sessions, speakers, and schedule
  description: |
    Provides access to **session** schedules,
    **speaker** profiles, and **room** assignments.
  contact:
    name: PHP Tek Support
    url: https://phptek.io
    email: support@phparch.com
```

```yaml
info:
  title: PHP Tek Conference API
  version: "2026-05"
  summary: Sessions, speakers, and schedule
  description: |
    Provides access to **session** schedules,
    **speaker** profiles, and **room** assignments.
  contact:
    name: PHP Tek Support
    url: https://phptek.io
    email: support@phparch.com
```

```yaml
info:
  title: PHP Tek Conference API
  version: "2026-05"
  summary: Sessions, speakers, and schedule
  description: |
    Provides access to **session** schedules,
    **speaker** profiles, and **room** assignments.
  contact:
    name: PHP Tek Support
    url: https://phptek.io
    email: support@phparch.com
```

```yaml
servers:
  - url: https://api.tek.phparch.com/v1
    description: Production

  - url: https://staging.api.tek.phparch.com/v1
    description: Staging

  - url: http://localhost:8080/v1
    description: Local development
```

```yaml
tags:
  - name: Sessions
    description: Conference sessions & schedule

  - name: Speakers
    description: Speaker profiles and bios
```

```yaml
paths:
  /sessions:
    get:
      summary: List all sessions
      operationId: listSessions
      tags: [Sessions]
      parameters:
        - name: day
          in: query
          description: Filter by day (1, 2, or 3)
          schema:
            type: integer
            maximum: 3
            minimum: 1
      responses:
        "200":
          description: A list of conference sessions
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: "#/components/schemas/Session"
```

```yaml
paths:
  /sessions:
    get:
      summary: List all sessions
      operationId: listSessions
      tags: [Sessions]
```

```yaml
parameters:
  - name: day
    in: query
    description: Filter by day (1, 2, or 3)
    schema:
      type: integer
      maximum: 3
      minimum: 1
```

```yaml
responses:
  "200":
    description: A list of conference sessions
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: "#/components/schemas/Session"
```

```yaml
components:
  schemas:
    Session: ...
    Speaker: ...
  responses:
    NotFound: ...
  parameters:
    SessionId: ...
  securitySchemes:
    BearerAuth: ...
```

Components are the reuse engine of OpenAPI. Define once. Reference anywhere.

```yaml
$ref: "#/components/schemas/Session"
```

Schemas: describing your data

Airam Dato-on / Pexels

In OpenAPI 3.1, the Schema Object is **JSON Schema**

JSON Schema Draft 2020-12

Yu Ma / Pexels

```yaml
type: string
type: integer
type: number
type: boolean
type: array
type: object
type: "null"   # quoted string — not YAML null
```

In 3.1, `type` can also be an **array**:

```yaml
type: [string, "null"]
```

```yaml
type: integer
format: int32          # signed 32-bit
format: int64          # signed 64-bit

type: string
format: date           # 2026-05-20
format: date-time      # 2026-05-20T21:00:00Z
format: uuid
format: email
format: uri
format: password       # hints to UIs to obscure the value
```

```yaml
components:
  schemas:
    Session:
      type: object
      required: [id, title, startsAt]
      properties:
        id:
          type: integer
          format: int64
        title:
          type: string
          minLength: 1
          maxLength: 200
        abstract:
          type: string
        startsAt:
          type: string
          format: date-time
        room:
          type: string
        speaker:
          $ref: "#/components/schemas/Speaker"
```

```yaml
components:
  schemas:
    Session:
      type: object
      required: [id, title, startsAt]
      properties:
        ...
```

```yaml
title:
  type: string
  minLength: 1
  maxLength: 200
```

```
speaker:
  $ref: "#/components/schemas/Speaker"
```

```yaml
# 3.0 — nullable: true (DO NOT use in 3.1)
type: string
nullable: true

# 3.1 — type is a JSON Schema keyword; use a type array
type: [string, "null"]

# Also valid in 3.1
oneOf:
  - type: string
  - type: "null"
```

```yaml
# 3.0 — nullable: true (DO NOT use in 3.1)
type: string
nullable: true

# 3.1 — type is a JSON Schema keyword; use a type array
type: [string, "null"]

# Also valid in 3.1
oneOf:
  - type: string
  - type: "null"
```

```yaml
# 3.0 — nullable: true (DO NOT use in 3.1)
type: string
nullable: true

# 3.1 — type is a JSON Schema keyword; use a type array
type: [string, "null"]

# Also valid in 3.1
oneOf:
  - type: string
  - type: "null"
```

Composition

Keyword	Meaning
`allOf`	Valid against **all** listed schemas (intersection / extends)
`anyOf`	Valid against **one or more** listed schemas
`oneOf`	Valid against **exactly one** listed schema

```yaml
Session:
  oneOf:
    - $ref: "#/components/schemas/Talk"
    - $ref: "#/components/schemas/Workshop"
```

```yaml
Talk:
  allOf:
    - $ref: "#/components/schemas/BaseSession"
    - type: object
      properties:
        type:
          type: string
          enum: [talk]
        slides:
          type: string
          format: uri
```

```yaml
Workshop:
  allOf:
    - $ref: "#/components/schemas/BaseSession"
    - type: object
      properties:
        type:
          type: string
          enum: [workshop]
        maxAttendees:
          type: integer
```

```yaml
Session:
  oneOf:
    - $ref: "#/components/schemas/Talk"
    - $ref: "#/components/schemas/Workshop"
  discriminator:
    propertyName: type
    mapping:
      talk: "#/components/schemas/Talk"
      workshop: "#/components/schemas/Workshop"
```

Reuse

```yaml
# Reference within the same document
$ref: "#/components/schemas/Session"

# Reference an external file
$ref: "./schemas/speaker.yaml"

# External file with a specific fragment
$ref: "./common.yaml#/components/schemas/Error"
```

Reuse

```yaml
# Reference within the same document
$ref: "#/components/schemas/Session"

# Reference an external file
$ref: "./schemas/speaker.yaml"

# External file with a specific fragment
$ref: "./common.yaml#/components/schemas/Error"
```

Reuse

```
# Reference within the same document
$ref: "#/components/schemas/Session"

# Reference an external file
$ref: "./schemas/speaker.yaml"

# External file with a specific fragment
$ref: "./common.yaml#/components/schemas/Error"
```

Reuse

```
# Reference within the same document
$ref: "#/components/schemas/Session"

# Reference an external file
$ref: "./schemas/speaker.yaml"

# External file with a specific fragment
$ref: "./common.yaml#/components/schemas/Error"
```

$ref siblings

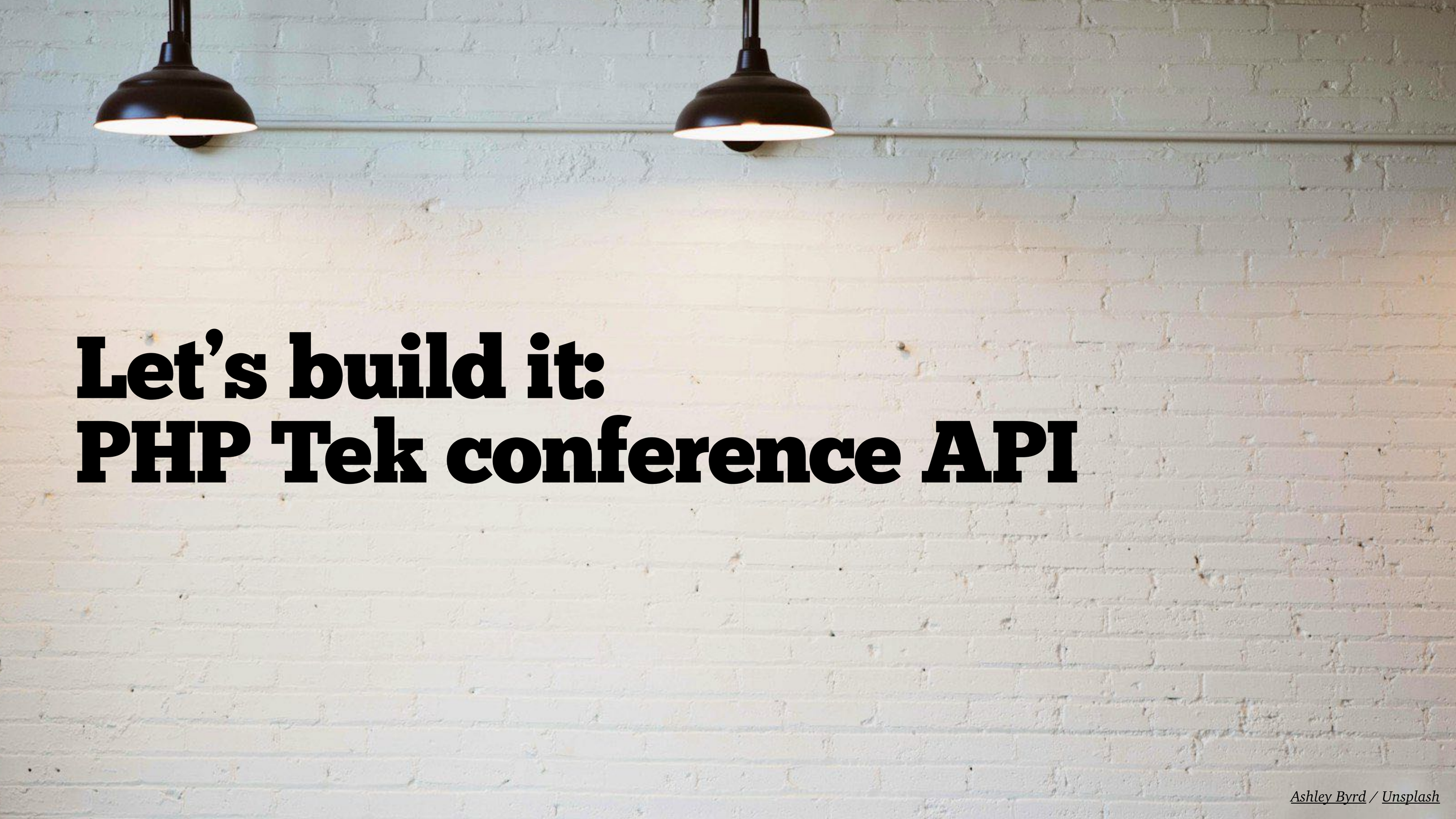
```yaml
# 3.0 — sibling fields were silently ignored
$ref: "#/components/schemas/Session"
description: "This was ignored in 3.0"

# 3.1 — sibling fields are meaningful
$ref: "#/components/schemas/Session"
description: The session being registered for
```

$ref siblings

```yaml
# 3.0 — sibling fields were silently ignored
$ref: "#/components/schemas/Session"
description: "This was ignored in 3.0"

# 3.1 — sibling fields are meaningful
$ref: "#/components/schemas/Session"
description: The session being registered for
```

$ref siblings

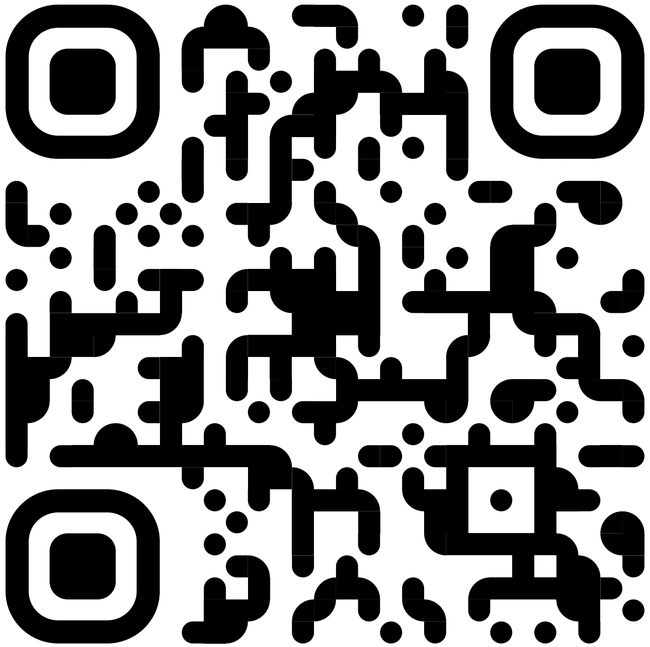
```
# 3.0 — sibling fields were silently ignored
$ref: "#/components/schemas/Session"
description: "This was ignored in 3.0"

# 3.1 — sibling fields are meaningful
$ref: "#/components/schemas/Session"
description: The session being registered for
```



Let's build it: PHP Tek conference API

Ashley Byrd / Unsplash

bram.se/phptek-openapi-yaml



Tanya Barrow / Unsplash

PHP Tek

Conference API

`GET /sessions`
list all sessions

`GET /sessions/{id}`
get a single session

`GET /speakers`
list speakers

Step 1

Getting started

```yaml
openapi: "3.1.2"
info:
  title: PHP Tek Conference API
  version: "2026-05"
servers:
  - url: https://api.tek.phparch.com/
    description: Production
  - url: http://localhost:8080/
    description: Local development
```

Step 2

Add the first path

```yaml
paths:
  /sessions:
    get:
      summary: List all sessions
      operationId: listSessions
      tags: [Sessions]
      parameters:
        - name: day
          in: query
          description: Filter by day (1, 2, or 3)
          schema:
            type: integer
            maximum: 3
            minimum: 1
      responses:
        "200":
          description: A list of sessions
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: "#/components/schemas/Session"
```

```yaml
components:
  schemas:
    Session:
      type: object
      required: [id, title, startsAt, endsAt]
      properties:
        id:
          type: integer
          format: int64
        title:
          type: string
        abstract:
          type: string
        startsAt:
          type: string
          format: date-time
        endsAt:
          type: string
          format: date-time
        room:
          type: string
        speaker:
          $ref: "#/components/schemas/Speaker"
    Speaker:
      type: object
      required: [id, name]
      properties:
        id:
          type: integer
          format: int64
        name:
          type: string
        bio:
          type: [string, "null"]
```

Step 3

Define schemas

```yaml
Session:
  type: object
  required: [id, title, startsAt, endsAt]
  properties:
    id:
      type: integer
      format: int64
    title:
      type: string
    abstract:
      type: string
    startsAt:
      type: string
      format: date-time
    endsAt:
      type: string
      format: date-time
    room:
      type: string
    speaker:
      $ref: "#/components/schemas/Speaker"
```

```yaml
Session:
  type: object
  required: [id, title, startsAt, endsAt]
  properties:
    ...
```

```yaml
properties:
  id:
    type: integer
    format: int64
  ...
  startsAt:
    type: string
    format: date-time
  endsAt:
    type: string
    format: date-time
  ...
```

```yaml
properties:
  ...
  speaker:
    $ref: "#/components/schemas/Speaker"
  ...
```

```yaml
Speaker:
  type: object
  required: [id, name]
  properties:
    id:
      type: integer
      format: int64
    name:
      type: string
    bio:
      type: [string, "null"]
```

Step 4

Second path with params

```yaml
paths:
  /sessions/{id}:
    get:
      summary: Get a session by ID
      operationId: getSession
      tags: [Sessions]
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
            format: int64
      responses:
        "200":
          description: The session
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Session"
        "404":
          $ref: "#/components/responses/NotFound"
```

```
/sessions/{id}:
  get:
    parameters:
      - name: id
        in: path
        required: true
```

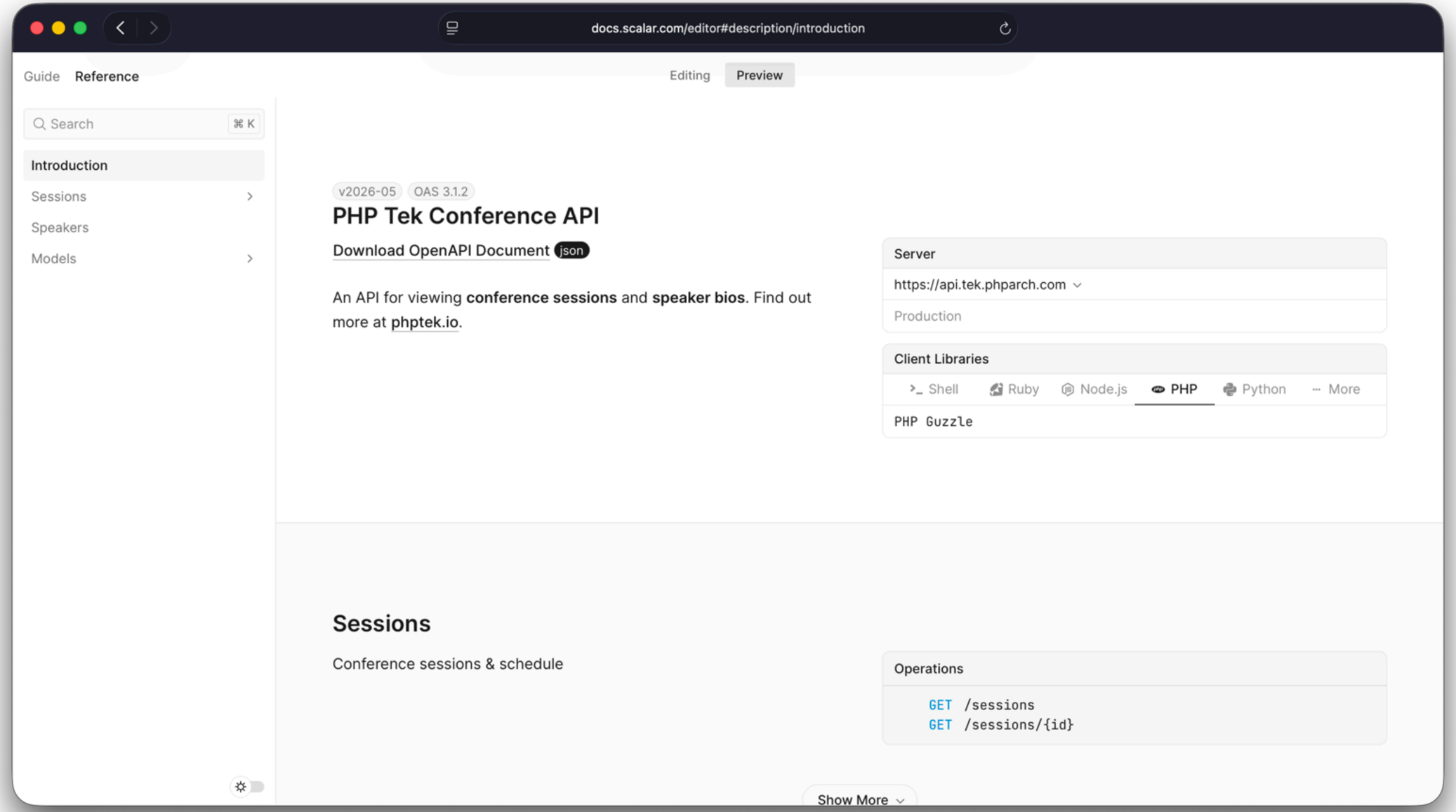
```
responses:
  "200":
    description: The session
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/Session"
  "404":
    $ref: "#/components/responses/NotFound"
```

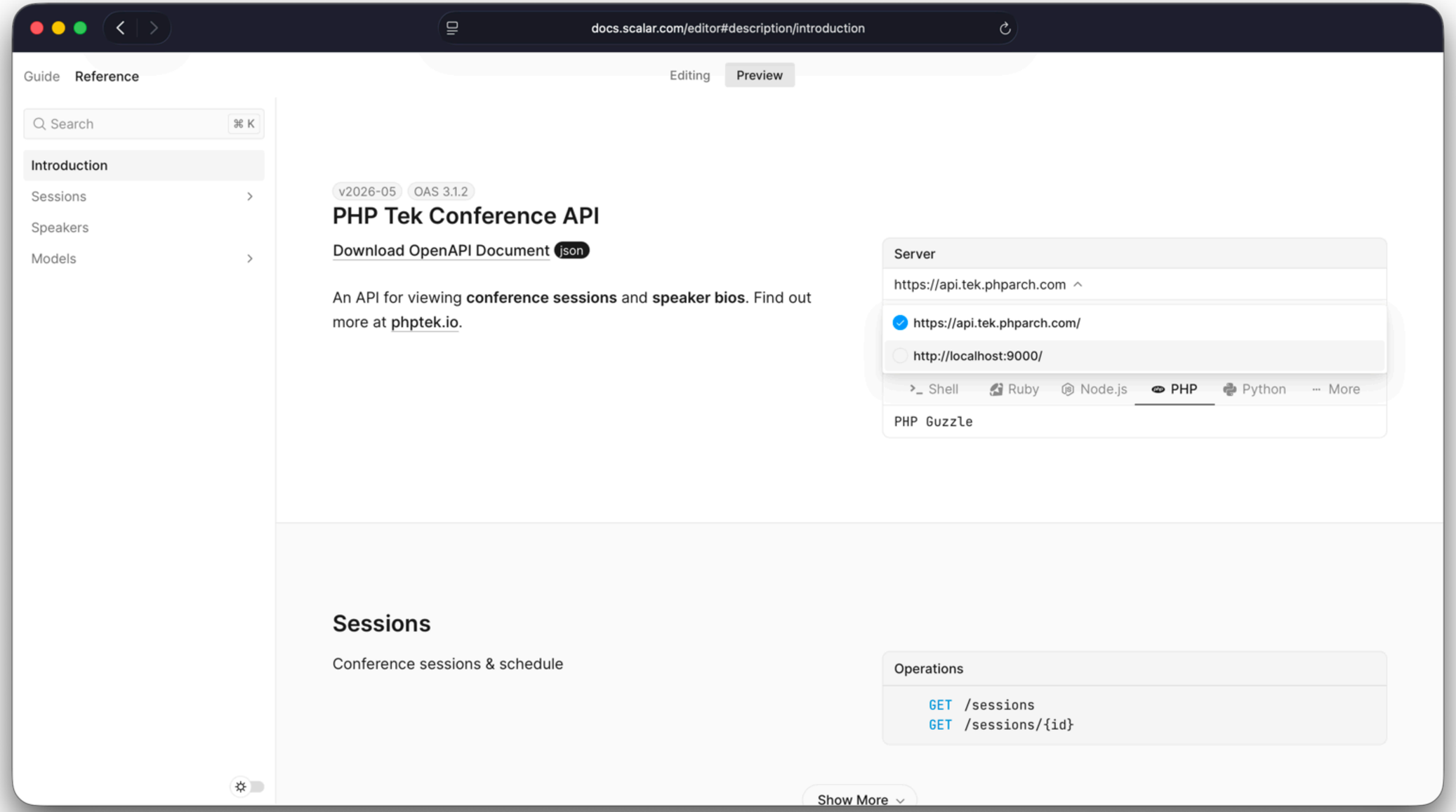
Step 5
Reusable errors

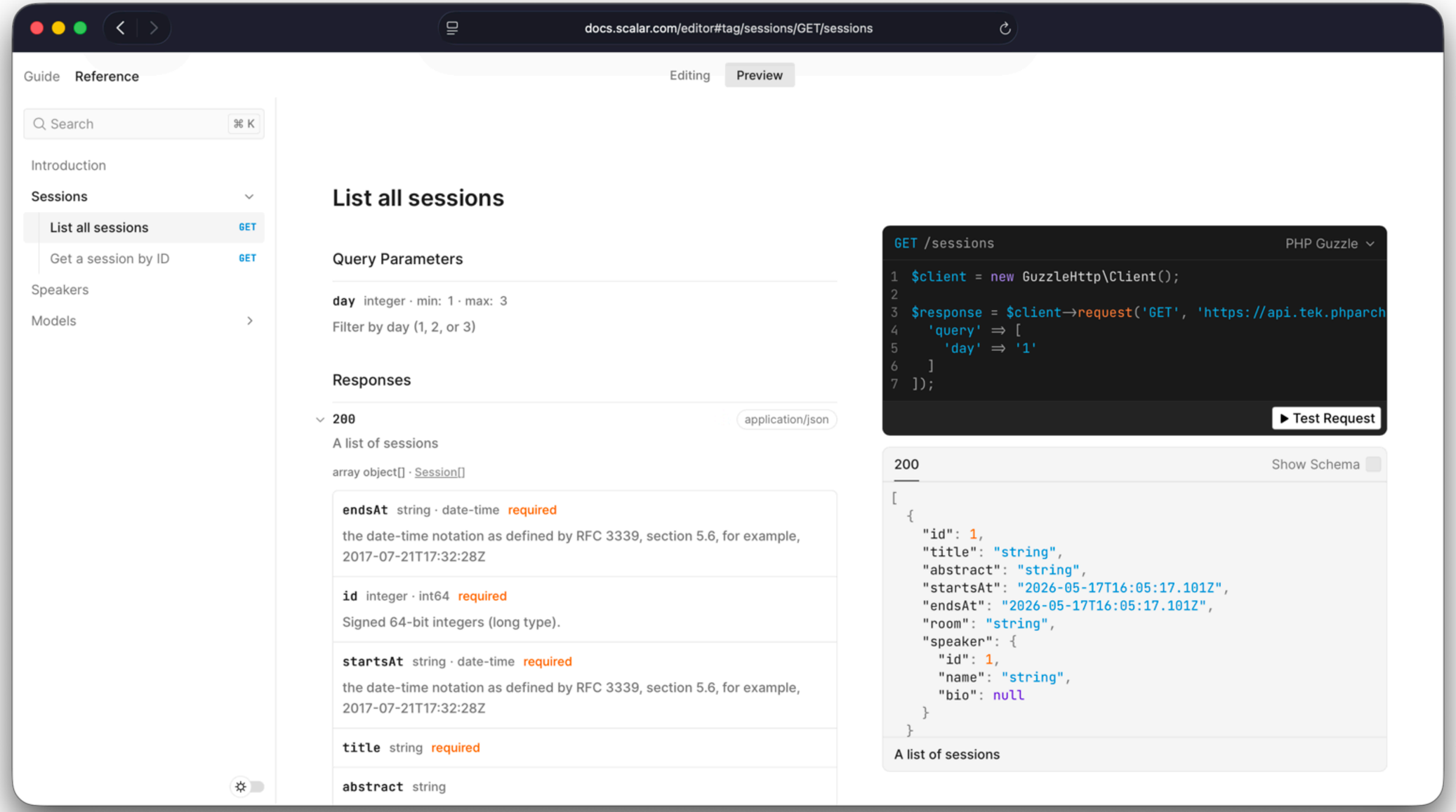
```yaml
components:
  responses:
    NotFound:
      description: Resource not found
      content:
        application/problem+json:
          schema:
            $ref: "#/components/schemas/Problem"
  schemas:
    Problem:
      type: object
      properties:
        type:
          type: string
          format: uri
        title:
          type: string
        status:
          type: integer
        detail:
          type: string
```

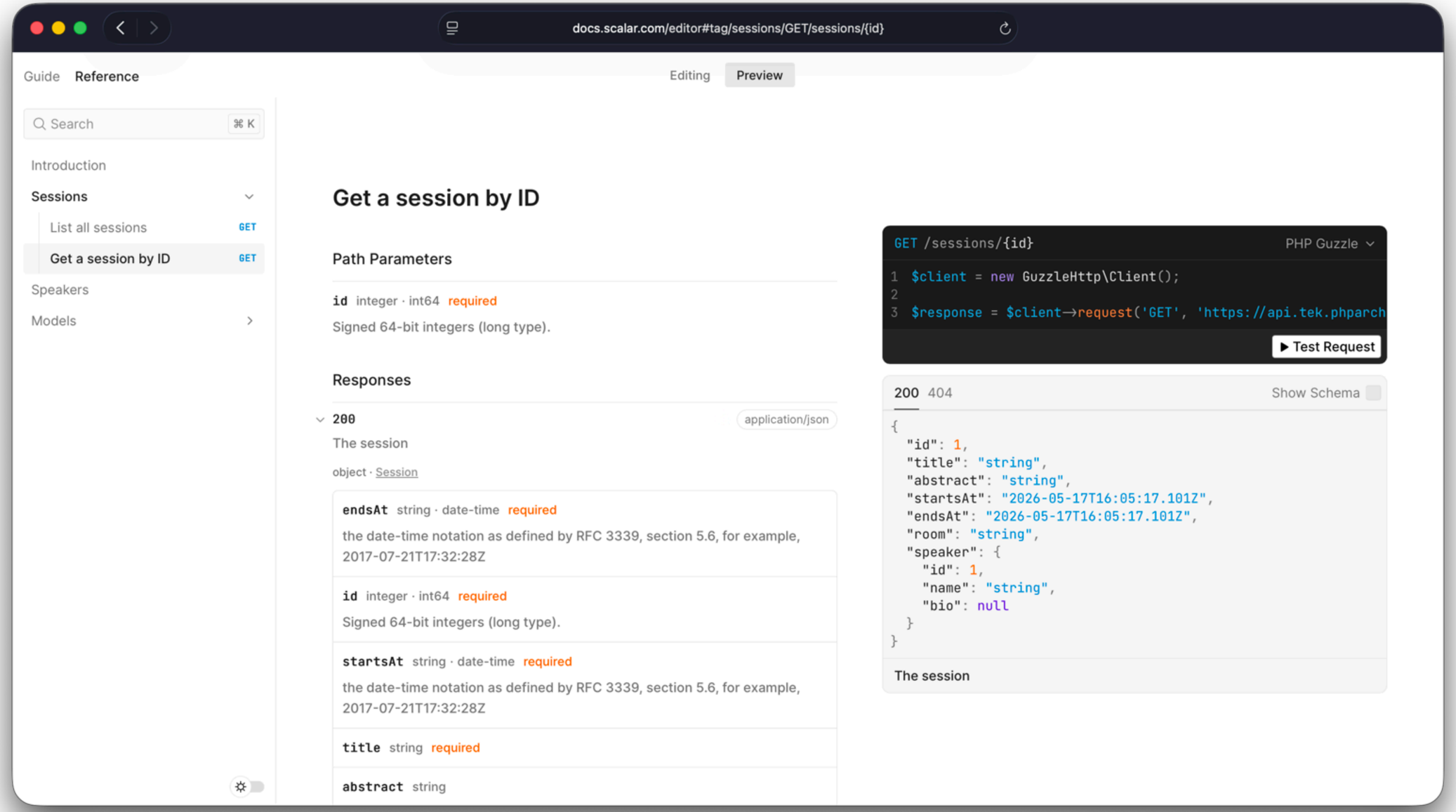
```yaml
responses:
  NotFound:
    description: Resource not found
    content:
      application/problem+json:
        schema:
          $ref: "#/components/schemas/Problem"
```

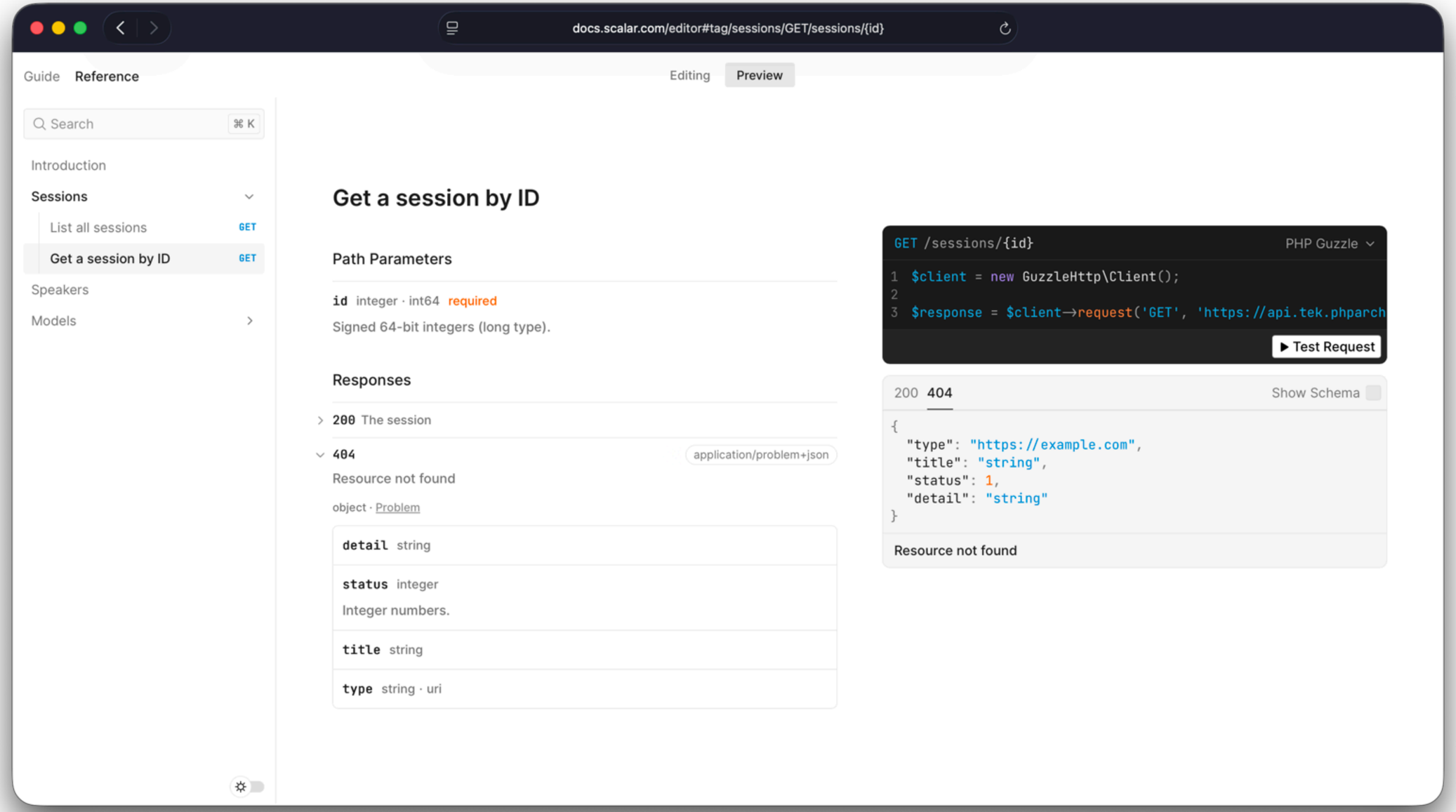
```yaml
schemas:
  Problem:
    type: object
    properties:
      type:
        type: string
        format: uri
      title:
        type: string
      status:
        type: integer
      detail:
        type: string
```


docs.scalar.com/editor#description/introduction
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
Sessions
Speakers
Models
v2026-05
OAS 3.1.2
PHP Tek Conference API
Download OpenAPI Document
json
An API for viewing conference sessions and speaker bios. Find out more at phptek.io.
Server
https://api.tek.phparch.com
Production
Client Libraries
Shell
Ruby
Node.js
PHP
Python
More
PHP Guzzle
Sessions
Conference sessions & schedule
Operations
GET /sessions
GET /sessions/{id}
Show More


docs.scalar.com/editor#description/introduction
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
Sessions
Speakers
Models
v2026-05
OAS 3.1.2
PHP Tek Conference API
Download OpenAPI Document
json
An API for viewing conference sessions and speaker bios. Find out more at phptek.io.
Server
https://api.tek.phparch.com
https://api.tek.phparch.com/
http://localhost:9000/
Shell
Ruby
Node.js
PHP
Python
More
PHP Guzzle
Sessions
Conference sessions & schedule
Operations
GET /sessions
GET /sessions/{id}
Show More


docs.scalar.com/editor#tag/sessions/GET/sessions
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
Sessions
List all sessions
GET
Get a session by ID
GET
Speakers
Models
List all sessions
Query Parameters
day integer · min: 1 · max: 3
Filter by day (1, 2, or 3)
Responses
200
application/json
A list of sessions
array object[] · Session[]
endsAt string · date-time required
the date-time notation as defined by RFC 3339, section 5.6, for example, 2017-07-21T17:32:28Z
id integer · int64 required
Signed 64-bit integers (long type).
startsAt string · date-time required
the date-time notation as defined by RFC 3339, section 5.6, for example, 2017-07-21T17:32:28Z
title string required
abstract string
GET /sessions
PHP Guzzle
$client = new GuzzleHttp\Client();
$response = $client->request('GET', 'https://api.tek.phparch
'query' => [
'day' => '1'
]
]);
▶ Test Request
200
Show Schema
[
{
"id": 1,
"title": "string",
"abstract": "string",
"startsAt": "2026-05-17T16:05:17.101Z",
"endsAt": "2026-05-17T16:05:17.101Z",
"room": "string",
"speaker": {
"id": 1,
"name": "string",
"bio": null
}
}
A list of sessions


docs.scalar.com/editor#tag/sessions/GET/sessions/{id}
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
Sessions
List all sessions
GET
Get a session by ID
GET
Speakers
Models
Get a session by ID
Path Parameters
id integer · int64 required
Signed 64-bit integers (long type).
Responses
200
application/json
The session
object · Session
endsAt string · date-time required
the date-time notation as defined by RFC 3339, section 5.6, for example, 2017-07-21T17:32:28Z
id integer · int64 required
Signed 64-bit integers (long type).
startsAt string · date-time required
the date-time notation as defined by RFC 3339, section 5.6, for example, 2017-07-21T17:32:28Z
title string required
abstract string
GET /sessions/{id}
PHP Guzzle
1 $client = new GuzzleHttp\Client();
2
3 $response = $client->request('GET', 'https://api.tek.phparch
▶ Test Request
200
404
Show Schema
{
"id": 1,
"title": "string",
"abstract": "string",
"startsAt": "2026-05-17T16:05:17.101Z",
"endsAt": "2026-05-17T16:05:17.101Z",
"room": "string",
"speaker": {
"id": 1,
"name": "string",
"bio": null
}
}
The session


docs.scalar.com/editor#tag/sessions/GET/sessions/{id}
Guide
Reference
Editing
Preview
Search
⌘ K
Introduction
Sessions
List all sessions
GET
Get a session by ID
GET
Speakers
Models
Get a session by ID
Path Parameters
id integer · int64 required
Signed 64-bit integers (long type).
Responses
200 The session
404
application/problem+json
Resource not found
object · Problem
detail string
status integer
Integer numbers.
title string
type string · uri
GET /sessions/{id}
PHP Guzzle
1 $client = new GuzzleHttp\Client();
2
3 $response = $client->request('GET', 'https://api.tek.phparch
▶ Test Request
200 404
Show Schema
{
"type": "https://example.com",
"title": "string",
"status": 1,
"detail": "string"
}
Resource not found

Beyond the basics

Jen Dries / Unsplash

Security Schemes

```yaml
components:
  securitySchemes:
    ApiKeyAuth:
      type: apiKey
      in: header
      name: Api-Key
    BearerAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT
    OAuth2:
      type: oauth2
      flows:
        authorizationCode:
          authorizationUrl: https://example.com/oauth/authorize
          tokenUrl: https://example.com/oauth/token
          scopes:
            read:sessions: Read session data
            write:sessions: Register for sessions
    OpenIDConnect:
      type: openIdConnect
      openIdConnectUrl: https://example.com/.well-known/openid-configuration
```

Global default

```yaml
security:
  - BearerAuth: []
```

Per-operation override

```yaml
paths:
  /sessions:
    get:
      security: []   # no auth, public
  /sessions/{id}/register:
    post:
      security:
        - BearerAuth: []
        - OAuth2:
            - write:sessions
```

Webhooks

New in 3.1

```yaml
webhooks:
  sessionRegistered:
    post:
      summary: Fired when an attendee registers for a session
      requestBody:
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/RegistrationEvent"
      responses:
        "202":
          description: Acknowledge receipt
```

Extensions (x-)

```yaml
paths:
  /sessions:
    get:
      x-internal: true
      x-rate-limit: 100
      x-beta: true
      summary: List all sessions
      ...
```

Tips, Tricks & Gotchas

MChe Lee / Unsplash

1. `openapi` must be a quoted string

```yaml
# WRONG — YAML parses 3.1.2 as a float
openapi: 3.1.2

# Correct
openapi: "3.1.2"
```

2. `nullable: true` is not in 3.1

```yaml
# 3.0 — nullable: true (DO NOT use in 3.1)
type: string
nullable: true

# 3.1 — type is an array
type: [string, "null"]

# Also valid in 3.1
oneOf:
  - type: string
  - type: "null"
```

3. Status codes must be strings

```yaml
responses:

  200:    # WRONG — integer key
    description: OK

  "200": # Correct
    description: OK

  "2XX": # Also valid — range wildcard
    description: Any 2xx success
```

4. operationID must be unique

```
paths:

  /sessions:
    post:
      operationId: createSession

  /speakers:
    post:
      operationId: createSpeaker
```

5. Path parameters must be declared

Every `{variable}` in the path **must** have a matching parameter entry:

```yaml
/sessions/{id}:
  get:
    parameters:
      - name: id
        in: path
        required: true    # mandatory for path params
        schema:
          type: integer
          format: int64
```

Missing the entry **or** missing `required: true` is a validation error.

6. `$ref` siblings now work

3.0: siblings next to `$ref` were silently ignored.

3.1: siblings are meaningful — they override or extend the referenced schema.

```
$ref: "#/components/schemas/Session"
description: The session being registered for
```

7. additionalProperties: false and allOf

```yaml
# Problematic — blocks sub-schemas from adding properties
BaseSession:
  type: object
  properties:
    id:
      type: integer
  additionalProperties: false   # rejects Talk's extra properties


# Better — JSON Schema 2020-12 aware
BaseSession:
  type: object
  properties:
    id:
      type: integer
  unevaluatedProperties: false  # respects allOf sub-schemas
```

8. `format` is an annotation, not validation

e.g., `format: email` does **not** guarantee email validation

```yaml
type: string
format: email    # annotation, documents intent
                 # validation depends on tool
```

9. `binary` and `byte` formats are gone

```yaml
# 3.0
type: string
format: binary                    # raw binary file upload/download

type: string
format: byte                      # base64-encoded binary


# 3.1 replacements
contentMediaType: image/png       # raw binary

type: string
contentMediaType: image/png
contentEncoding: base64           # base64-encoded
```

10. Recursion works

```yaml
components:
  schemas:
    Session:
      type: object
      properties:
        id:
          type: integer
        relatedSessions:
          type: array
          items:
            $ref: "#/components/schemas/Session"
```

The ecosystem

Wolfgang Rottmann / *Unsplash*

Editors & design tools

- Swagger Editor
- Stoplight Platform
- OpenAPI Editor by 42Crunch for VS Code
- PhpStorm / IntelliJ

Albert Stoynov / Unsplash

Peter Werkman / Unsplash

Docs renderers

- Swagger UI
- Redoc
- Scalar

PHP Libraries

Code-first (generate spec from code)

- zircote/swagger-php
- darkaonline/l5-swagger
- dedoc/scramble
- API Platform

Tanya Barrow / *Unsplash*

Diane Picchiottino / Unsplash

PHP Libraries

Description-first & validation

- jane-php/open-api-3
- OpenAPI Generator
- league/openapi-psr7-validator

Validators & linters

- Spectral by Stoplight
- vacuum

Tanya Barrow / Unsplash

Description-first vs. code-first

	Description-first	Code-first
Workflow	Write spec then implement	Implement then generate spec
Best for	New APIs, contract-driven dev	Existing APIs, keeping spec in sync
Strength	API can be reviewed & mocked before any code	Spec stays close to implementation
Risk	Spec and code can diverge without discipline	Spec reflects code, not design intent

Wrap up & what's coming

Derick McKinney / Unsplash

Why 3.1.2?

OpenAPI 3.2.0 was published on September 19, 2025

- the same day as 3.1.2

3.2.0 is the **recommended** version going forward

But tooling hasn’t fully caught up

3.1.2 is what works reliably today

wr heustis / Pexels

Shamia Casiano / *Pexels*

What's new in 3.2.0?

`$self`

Improved multi-document handling

Media type improvements

JSON Schema

IETF standards process

May 12, 2026: IETF published draft-ietf-jsonschema-json-schema-00

Merges core and validation specs into a single document

OpenAPI is built directly on JSON Schema

Strong foundation for the long term

Tolga Ahmetler / Pexels

Tolga Ahmetler / Pexels

Where to go from here

- spec.openapis.org
  the specification
- learn.openapis.org
  tutorials and guides
- openapi.tools
  community tool directory
- json-schema.org
  JSON Schema documentation

Write the spec. Render it. Lint it. Version it.

Your API consumers will thank you.

Thank you!

Keep in touch

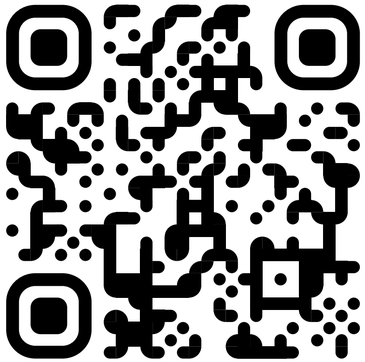

ben.ramsey.dev


phpc.social/@ramsey


github.com/ramsey


www.linkedin.com/in/benramsey


ben@ramsey.dev



bram.se/phptek-openapi

Describe Your API With OpenAPI
Copyright © 2026 Ben Ramsey

This work is licensed under Creative Commons Attribution-ShareAlike 4.0 International.
For uses not covered under this license, please contact the author.

Ramsey, Ben. “Describe Your API With OpenAPI.” PHP Tek Conference, 20 May 2026, Sheraton Suites Chicago O’Hare, Rosemont, IL.