
OPEN
VALUE

Three approaches in API Development

Enis Spahi (@enisspahi)

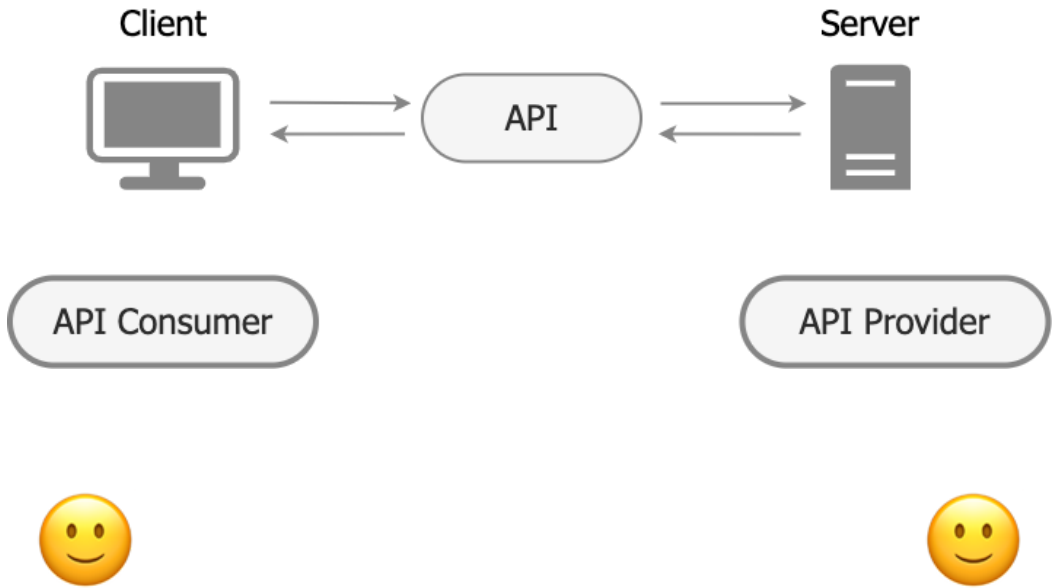
OPENVALUE

API

"An application programming interface (API) is a way for two or more computer programs to communicate with each other."

Source: Wikipedia, "API"

Relationships



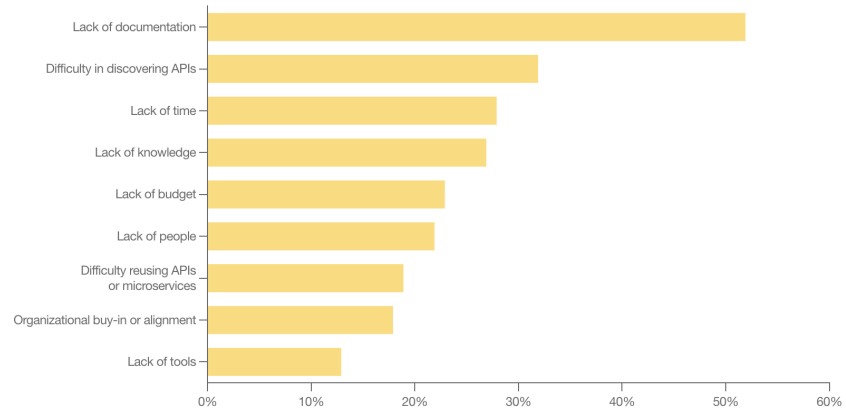
Boundaries

- Private APIs
  - Provider and consumer are developers in the same team or same organisation
- Partner Facing APIs
  - Serving partners (i.e.: Payment Service Providers)
  - Provider and consumer might not communicate directly
- Public APIs
  - Publicly available (i.e… Geo-Location services)
  - Communication at scale: Many Consumers ↔ 1 Provider

Statistics

Top 3 Obstacles to consuming APIs

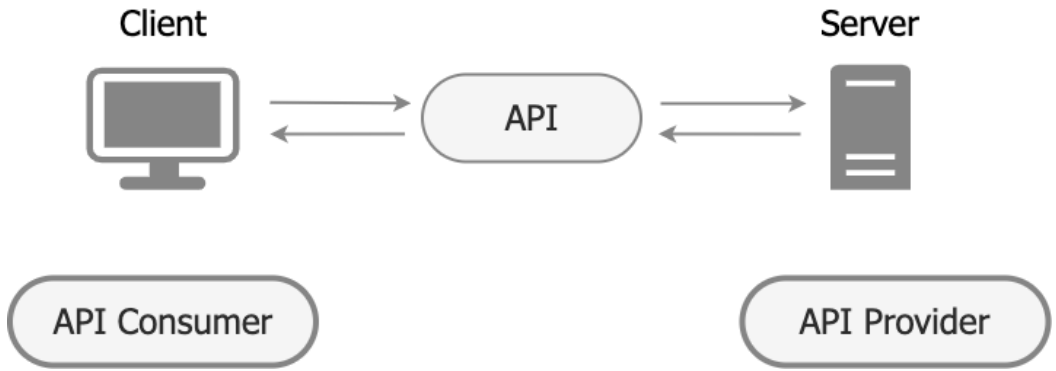
- Lack of documentation
- Difficulty in discovering APIs
- Lack of time



Source: Postman, "2023 State of the API Report"

OPENVALUE

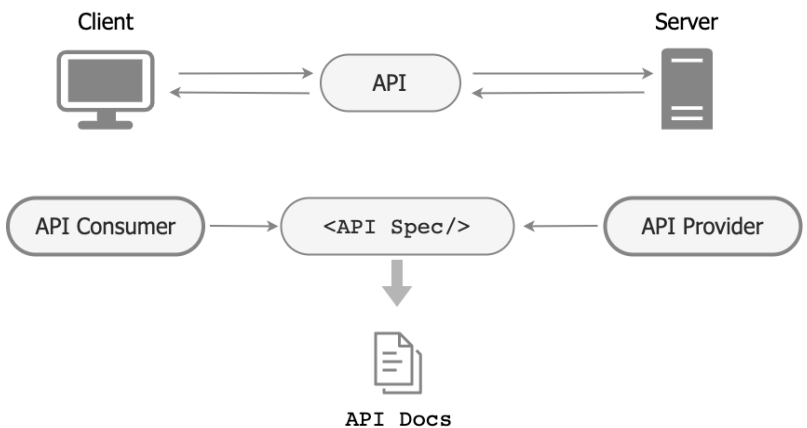
API Communication



Enhancing Communication

Speak common language

Standardized API Communication



- Common Language for API discovery
  - OpenAPI/Swagger for REST APIs
  - AsyncAPI for message-driven APIs
- Foundation for tooling
  - Code generation
  - Documentation
- Community

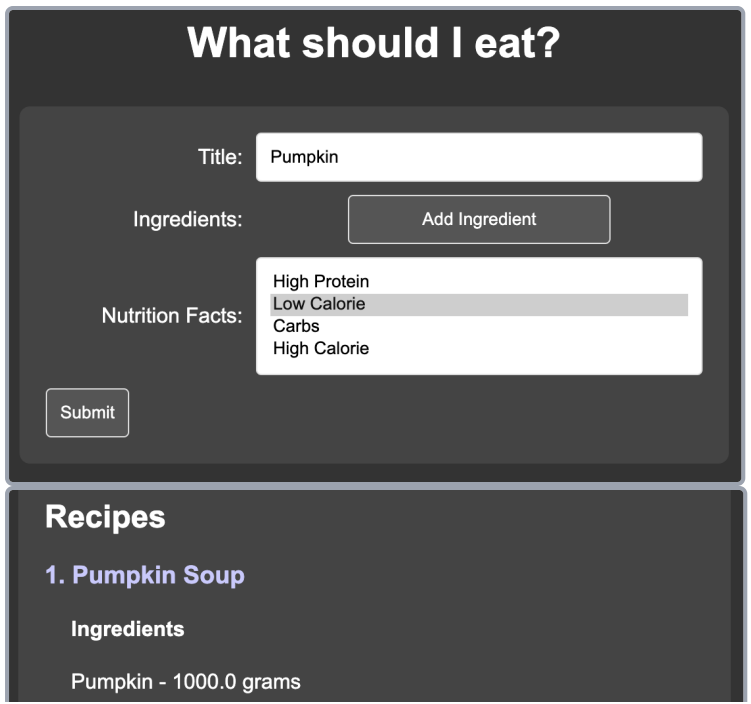
Development Process

Let’s build an API

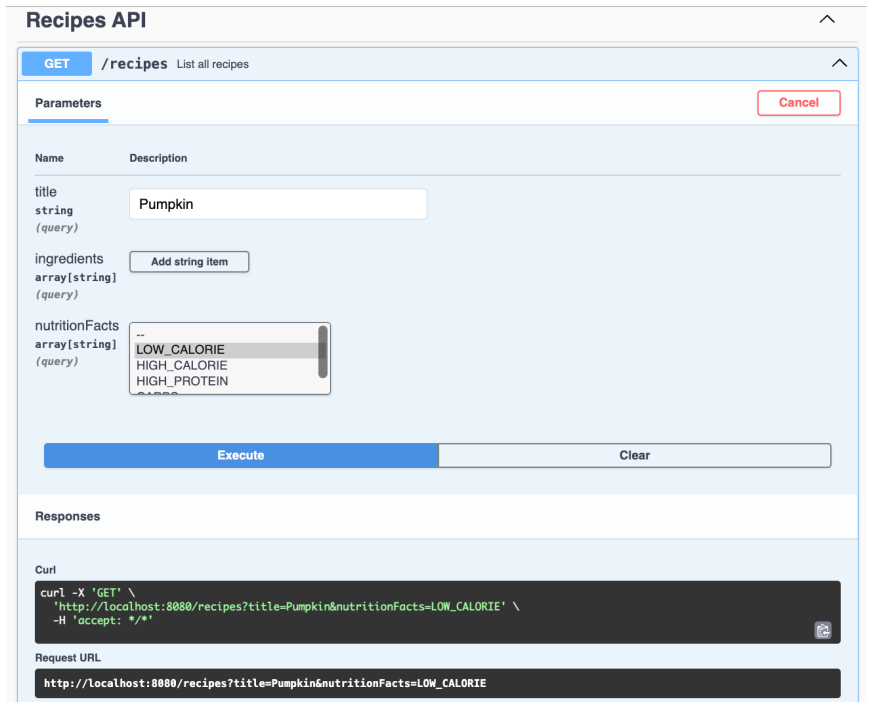
OPENVALUE

Recipes API

Client



Server

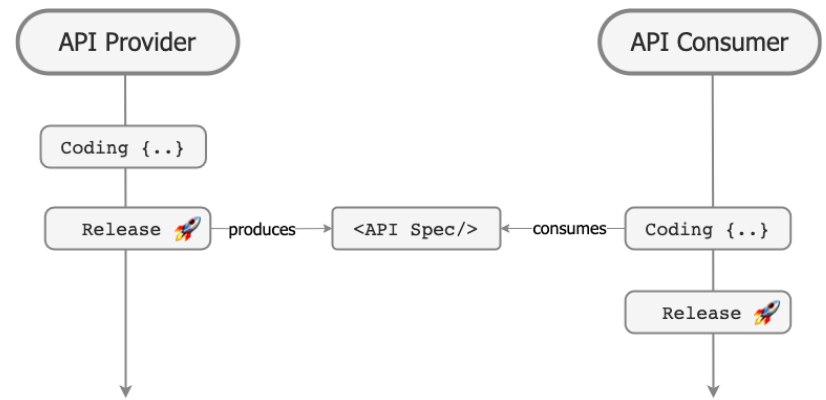


OPENVALUE

API Development - Code First

Communicate API specification once coding has been done

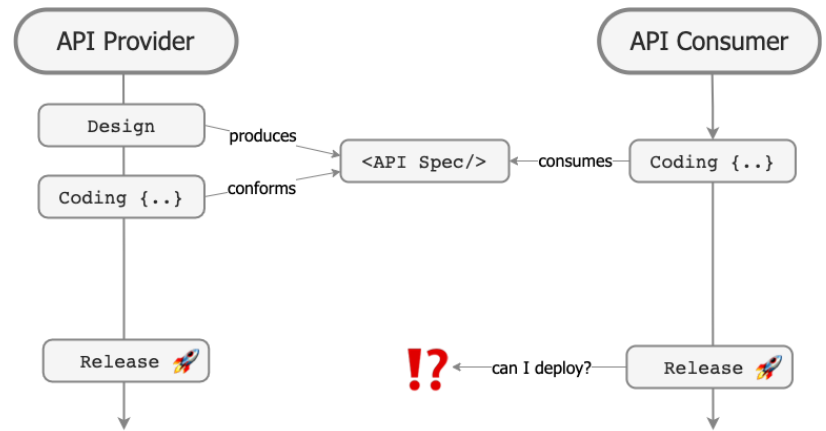
- **Advantages:**
  - Focus on coding
  - Flexibility to change the API design
- **Disadvantages:**
  - Late communication with the consumer
  - Does not enable development in parallel
  - Annotations



API Development - API First

Communicate API specification before coding. Prioritizes API design over implementation.

- **Advantages:**
  - Early communication with the consumer
  - Documentation thought ahead
  - Enables development in parallel
- **Disadvantages:**
  - Less flexibility to change the API design
  - Sometimes bureaucratic for providers



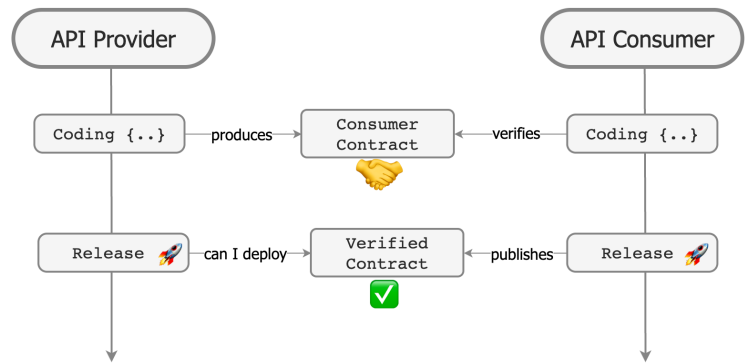
API Development - Consumer First

Consumer dictates the expected API behavior to the provider

Pact: A Code-first consumer contract testing tool that enables consumer driven API development.

Process:

- Consumer produces a pact
- Provider verifies it's API implementation
- Server / Client deployments synced



Summary

When to use Code first?	Provider initially focuses on coding speed Flexibility to change the design
When to use API first?	API design over implementation Early communication with the consumer → Documentation Utilize code generation Large number of consumers
When to use Consumer first?	Provider should conform to consumer needs API consumer and provider test their applications independently To sync provider and consumer deployments Small number of consumers
When to mix & match?	When API first alone is not sufficient to match consumer needs

OpenValue Blog

https://openvalue.blog/posts/2023/11/25/communicating_our_apis_part1/

https://openvalue.blog/posts/2023/11/26/communicating_our_apis_part2/

Code Samples

https://github.com/enisspahi/code-first-api-example

https://github.com/enisspahi/contract-first-api-example

https://github.com/enisspahi/consumer-first-api-example

Q&A