OpenAPI & AsyncAPI: Part 2

Jonathan Beri · jberi@golioth.io · @beriberikix

About me

Product guy

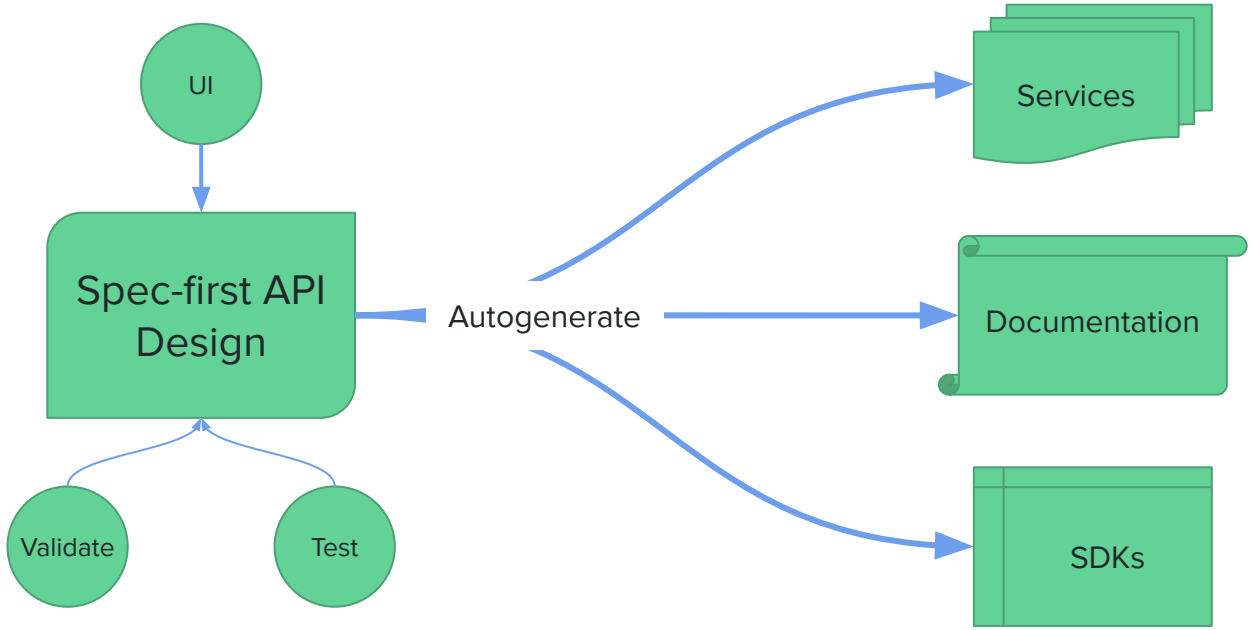
Ex-Google/Nest, startups

Founder @ IoT startup

jberi@golioth.io or @beriberikix

Recap: developer experience

We’re building an IoT platform with an emphasis on devexp.



Recap: OpenAPI/AsyncAPI specifications

- API contract between server developer and client integrator
- Automation a nice side effect of Human & Machine readable specs
- OpenAPI is an *opinionated* spec for *HTTP* APIs
- AsyncAPI is an *extensible* spec for *event-driven* APIs
- A rich ecosystem of tooling exists

Concepts from devs PoV

- Methods
- Responses
- Paths
- Errors
- Options
- Security
- OBSERVE
- Block transfer
- Group messaging
- Discovery
- Proxying

What would a CoRE “pseudo spec” inspired by OpenAPI look like?

Demo of “Pseudo Spec” (WIP)

Implications of pseudo spec

- Leverage existing spec format, toolchain and tools
- Would be more idiomatic and use CoRE terminology
- Requires forking and maintaining format, toolchain and libraries
    - Eventually could be used to incorporate recommendations back into OpenAPI/AsyncAPI

What about a valid OpenAPI CoRE spec look like?

Demo of OpenAPI-flavor (WIP)

Implications of OpenAPI-based spec

- Fully leverage existing spec format, toolchain and tools
- Only able to use terms in OpenAPI, “x-” extensions and tags
    - Tooling might be able to ease this problem
- No forking! Maybe.
    - But existing tools won’t know about CoRE extensions/tags and we’d need to work with them to add support...or fork

Demo of AsyncAPI (Not Started)

Revisiting Our Plan

- Define a pseudo-spec based on CoRE **(in-progress)**
    - Code @ bit.ly/coreoaiwip
- Identify how adapt CoRE spec to OpenAPI/AsyncAPI
- Decide to fork OpenAPI/AsyncAPI or pick adaptation
- Extra Credit: align with WoT

Feedback? Questions? Comments?

OpenAPI & AsyncAPI: Part 2

Jonathan Beri · jberi@golioth.io · @beriberikix