Axway Blog
News, trends and insights on hybrid integration.

Contact

Designing APIs for extensibility

By Erik Wilde - September 24, 2020

The main role of APIs is to serve as the *connective fabric* between different components and capabilities within an organization and across organizations.

One important aspect of that is that an API should be *reusable* so that individual components and capabilities can become building blocks of new processes and value chains.

However, this reusability creates a bit of a challenge. On one hand, it should be possible to improve and evolve an existing API (for example, when feedback from existing users is asking for additional features and capabilities). On the other hand, existing consumers should not be disrupted in their established ways of using the API.

Design APIs for extensibility

A proven way to minimize this problem is to *design APIs for extensibility*. If an API is designed so that it *can be changed without breaking existing consumers*, then API producers and API consumers become more loosely coupled and can evolve independently.

Making managing API versions part of the API lifecycle and using well-defined versioning practices such as *semantic versioning* are key ingredients. Yet, for all of this to work, the API must be designed for extensibility.

We can first look at what extensibility looks like for the five major API styles, and we can see that extensibility translates to the main abstractions of these five API styles:

- *Tunnel style:* The main abstraction are functions, and extensibility often means adding new functions to an API.
- *Resource style:* The main abstractions are resources and their representations, and extensibility often translates to adding new resources, and/or extending the representations of existing resources.
- *Hypermedia style:* The same as for the resource style, but also links and/or link relations may be added.
- *Query style:* The main abstraction is the schema to be queried, so extensibility means extending the schema of an API.
- *Event-based style:* The main abstractions are events and their representations, which means that extensibility means adding new events or evolving the representations of existing events.

Looking at extensibility a little closer, three main factors can be identified that play important roles when it comes to designing for extensibility.

By making these factors part of API practices and patterns, organizations can make their API landscape more stable and can minimize cases where changing an API disrupts existing consumers.

- *Meaningful initial API:* Creating a useful and meaningful API as a starting point creates the initial "promise" of the API that all consumers can depend on. It also creates an early way to start gathering feedback so that the API can be evolved based on consumer feedback.
- *Well-defined extension model:* Each API should clearly document the ways in which it can be extended. This allows consumers to understand in which way the API may change.
- *Well-defined processing model:* Each API should clearly document how extensions have to be processed. APIs should always use the must Ignore processing model, telling consumers that extensions are safe to ignore.

Using these practices, it is possible to design APIs that can be safely extended. This translates into fewer disruptions in the API landscape. It also means that API producers can more easily evolve their APIs because they can depend on consumers being able to gracefully deal with these extensions.

The following video explains these design issues in more detail and contains some tips and tricks for implementing these principles. Check it out if API design and API lifecycle management are important for you.

If you liked this video, why don't you check out Erik's YouTube channel for more "Getting APIs to Work" content?

Click Here

Erik Wilde

Digital Catalyst, Erik works in the Axway Catalyst team and focuses on API strategy, API programs, and API platforms. His main goal is to make sure that organizations make the

right decisions for using APIs as the foundation of their digital transformation initiatives. Erik has a Ph.D. from ETH Zurich, is the author of many articles, papers, and books. He is a frequent speaker at global API events and contributes to standardization activities to help improve the way APIs are designed, managed, and used."