API Resources | Understanding the API-First Approach to Building Products

By Janet Wagner

Web APIs have been around for nearly 20 years, but it is only in the past few years that the concept of “API first” has gained traction with software teams. The number of developers taking an API-first approach to building products is rising. So today we thought we would introduce you to the concept of API first and why this approach is growing in prominence.

What Does an API-First Approach Mean?

An API-first approach means that for any given development project, your APIs are treated as “first-class citizens.” That everything about a project revolves around the idea that the end product will be consumed by mobile devices, and that APIs will be consumed by client applications. An API-first approach involves developing APIs that are consistent and reusable, which can be accomplished by using an API description language to establish a contract for how the API is supposed to behave. Establishing a contract involves spending more time thinking about the design of an API. It also often involves additional planning and collaboration with the stakeholders providing feedback on the design of an API before any code is written.

The Growing Popularity of API-First

Today, both humans and machines are consuming data. Humans consume data through applications, often from many different devices — smartphones, laptops, tablets, and desktops. Many different types of devices mean many different screen sizes. Organizations must build apps that look good and work well across all devices.

APIs allow companies to break down capabilities into individual, autonomous services (aka microservices). Building applications based on microservices can help ensure a good user experience (UX) on all devices. An API-first strategy allows organizations to build APIs that serve all applications, and applications can be developed and maintained efficiently for all devices, platforms, and operating systems.

The Benefits of an API-First Approach

An API-first approach to building products provides many benefits, including but not limited to:

Development teams can work in parallel

API first involves establishing a contract. Creating a contract between services that is followed by teams across an organization allows those teams to work on multiple APIs at the same time. Developers do not have to wait for updates to an API to be released before moving on to the next API. Teams can mock APIs and test API dependencies based on the established API definition.

Reduces the cost of developing apps

APIs and code can be reused on many different projects. When a development team wants to build a new app, they don't have to start from scratch which is time-consuming and costly. API-first design also allows most problems to be solved before any code is even written which helps prevent problems when it is time to integrate APIs with applications.

Increases the speed to market

Much of the process of building APIs can be automated using tools that allow import of API definition files. Tools like SwaggerHub allow import of API definition files, and with those files API tools such as API documentation, SDKs, and mock APIs can be auto-generated. Automation significantly speeds up the development of APIs and applications.

API first also makes it possible to add new services and technologies to applications without having to re-architect the entire system. The competition is fierce when it comes to developing applications, so apps must be developed quickly. Today, applications must not only be well designed but also to market within six months.

Ensures good developer experiences

Consumers of APIs are most often developers, and developer experience (DX) can make or break the success of an API. API first ensures that developers have positive experiences using your APIs. Well-designed, well-documented, consistent APIs provide positive developer experiences because it's easier to reuse code and onboard developers, and it reduces the learning curve.

Reduces the risk of failure

For most companies, APIs are used in nearly every business process — from marketing and sales to communication and consumer-facing applications, which means that APIs can impact every part of your business positively or negatively. API first reduces the risk of failure by ensuring that APIs are reliable, consistent, and easy for developers to use.

Plan Your API-First Program

Now that you know some of the benefits of an API-first approach to product development, how should you go about planning and implementing an API-first approach?
Here are just a few things that should be part of your API-first plan.

1. **Brainstorm -** First, it is necessary to identify key services your business offers and capabilities of the business. Figure out the kinds of APIs that should be built and which services should be offered via APIs.

Also, figure out and write down the use cases for each API. Write down potential endpoints based on those use cases.

2. **Establish API stakeholders -** Who are the stakeholders within your organization? As many people as possible should be involved in your API initiative – you need company-wide buy-in and a vision that is shared by teams within your organization. Also, allow stakeholders to weigh in on the design of the API. Stakeholders can then agree on interactions across the organization so that APIs remain consistent.
3. **Design an API contract -** The contract establishes a set of standards and best practices for designing APIs. Be sure to describe and document all APIs. Ensure that all APIs work the same, from endpoint names and URLs to error codes and versioning. Consistency is key.
4. **Create a style guide -** A comprehensive, cohesive style guide ensures consistency across the teams of an organization who are building services. API status codes, versioning, error handling, and more will be standardized ensuring that APIs are designed the same way. Use a tool like SwaggerHub to create a style guide for all APIs across your organization.
5. **Implement API governance -** An API governance process can help enforce established standards and reinforce desired outcomes. We discuss API governance in an upcoming blog article. Conducting peer code reviews can also help ensure that API design standards are followed and that developers are producing quality code.
6. **Automate processes -** Use tools like SwaggerHub to automate processes like generating API documentation, style validation, API mocking, and versioning. Also, make APIs self-service so that developers can get started building apps with your APIs right away. Provide interactive documentation or a sandbox so that developers can try out API endpoints.
7. **Track and manage your API portfolio -** Avoid duplicating code and building redundant APIs by tracking and managing your API portfolio.

Implement a system that helps you track and manage your APIs. The larger your organization and platform becomes, the harder it gets to track APIs and their dependencies.

8. **Create a portal for internal developers -** Create a central place for internal developers, a place where everything for all your APIs is stored- API specification, documentation, contracts, etc. For example, PayPal has built a portal for its developers and it's "one of the most visited internal apps in PayPal," according to an InfoQ article. PayPal's portal includes an inventory of all APIs, documentation, dashboards, and more.

APIs as First-Class Citizens

An API-first approach to building products can benefit your organization in many ways. And API first approach requires that teams plan, organize, and share a vision of their API program. It also requires adopting tools that support an API first approach.

Thanks for reading! Looking for more API resources? Subscribe to the Swagger newsletter. Receive a monthly email with our best API articles, trainings, tutorials, and more. Subscribe