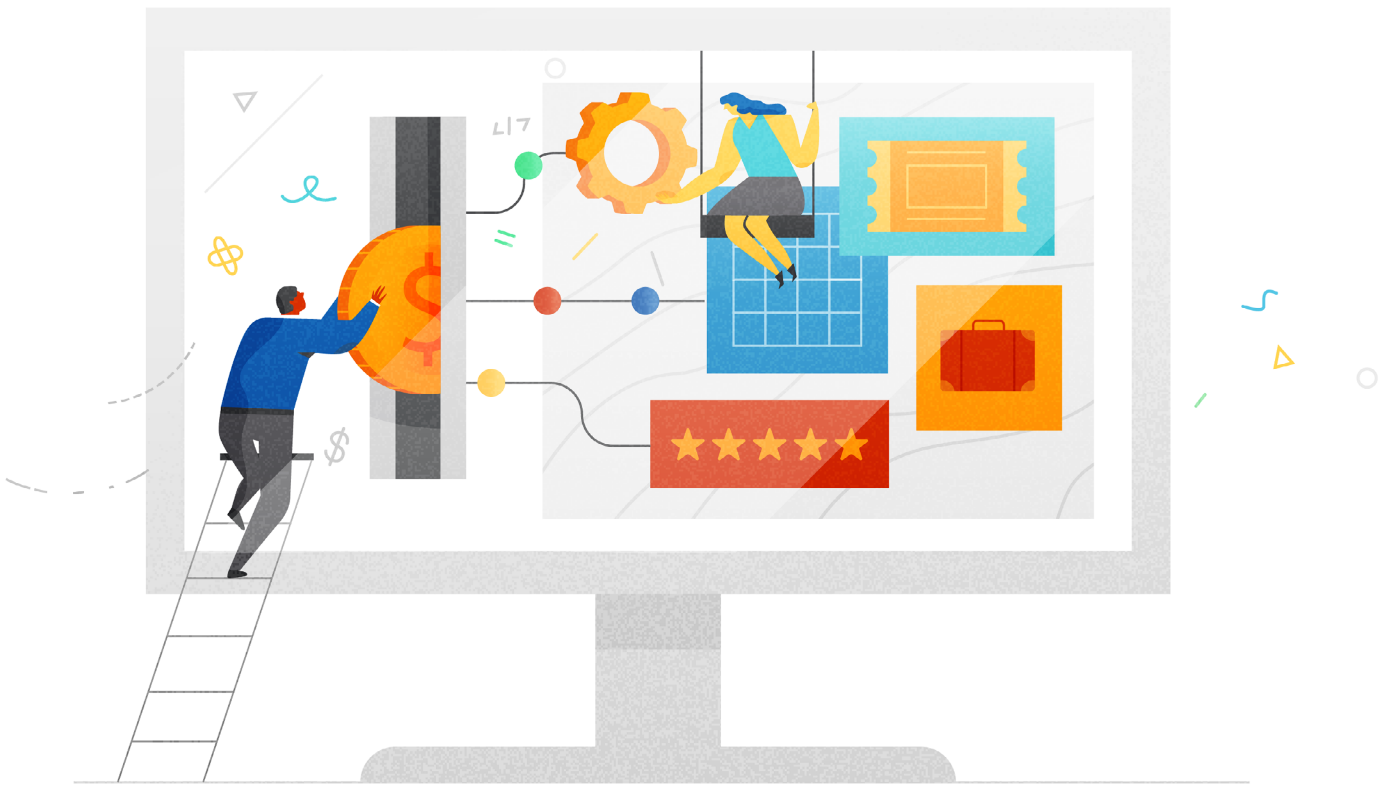
apigee

Google Cloud

Inside the API Product Mindset

Maximizing the Business Value of Digital Assets Through API Monetization



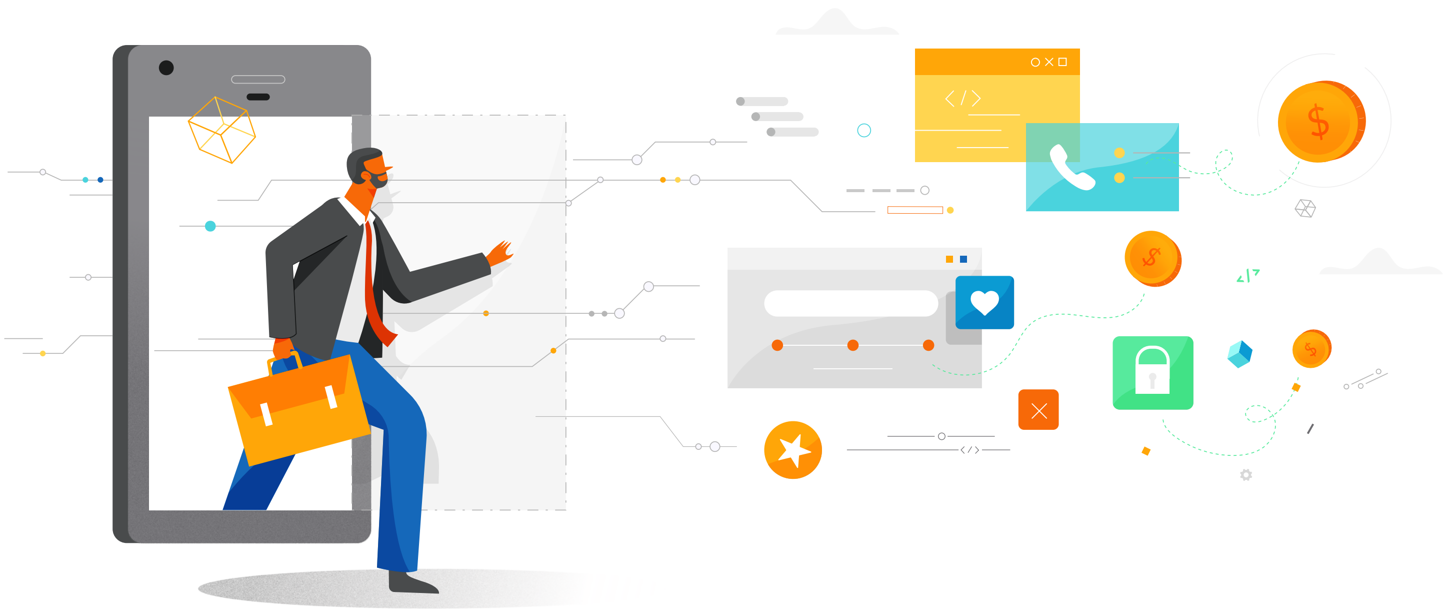
- Field-tested best practices
- Real-world use cases
- Monetization checklist

Table of contents

Inside the API product mindset

All businesses have digital assets: something valuable (functionality, data, access to certain product inventories or user bases, etc.) that can be expressed as software.

Application programming interfaces, or APIs, are the mechanisms that make these digital assets available to developers, both inside and outside an organization, who use them to build and add value to software applications.



An API abstracts a digital asset's complexity into a consistent interface, allowing developers to harness an asset even if they have no expertise in the underlying system that houses it. APIs allow developers to work more efficiently by reusing valuable functionality or data, and they allow assets to be combined more freely in new ways. The increase in developers leveraging these capabilities has not only led to an explosion of richer, more responsive digital experiences but has also positioned APIs in the middle of almost every organization's digital business initiatives.

As a result, many top digital enterprises view APIs not just as simple tools, but as full lifecycle products for developers.

As we explored in The API Product Mindset ebook, typical roles on an API product team include an **API Product Manager** who owns the processes and cross-functional coordination critical to the API's success; an **API Architect** responsible for designing and guiding the creation of APIs; an **API Developer** who builds APIs from the API Architect's designs and implements security policies and other protocols; an **API Evangelist** who serves as the voice of API consumers and owns partner and developer outreach; and an **API Champion** who works closely with internal executive sponsors to communicate the value of the API program to the rest of the organization.

The API product team typically carries several critical responsibilities:

- Design secure and easy-to-use APIs and bring them to market
- Deliver a world-class API developer experience
- Drive ongoing API improvements with monitoring and analytics
- Maximize the business value of APIs through API monetization, ecosystem participation, developer evangelism, etc.

This ebook dives deeper into one important aspect of this process: maximizing the business value of digital assets via API monetization.

Direct and Indirect Value: Not all APIs Should be Monetized

When a business makes its APIs available to external developers, those APIs can drive both direct and indirect value. Distinguishing these is a prerequisite to determining if, let alone how, an API should be monetized.

For example, many retailers make their store location APIs available for free to third-party developers. For both the developers who consume the API and the end users those developers serve, these APIs provide obvious value: better functionality, better applications, and better experiences. An API provider *could* charge for this value—but many decide they produce the most value and best customer experiences by bundling the capability or making it freely available to developers. After all, with more developers leveraging store location APIs, more applications will be delivering store locations to more users, which should theoretically lead to more sales. Is it wise to throw a wrench into this process by trying to monetize the API directly instead of enjoying the indirect benefits? A store locator has value—but perhaps not so much value or such unique value that many developers will pay for the chance to adopt the API.

In other cases, however, an enterprise may possess unique, hard-to-replicate and valuable data, such as the weather data AccuWeather's APIs provide. It may possess proprietary, valuable functionality such as the shipping and logistics capabilities offered by Pitney Bowes' APIs or the machine learning capabilities provided by APIs from a growing number of organizations. When an API provides access to valuable assets that developers are willing to pay for, directly monetizing the API may be more lucrative than giving it away to encourage adoption.

Regardless of whether or not an API is monetized, enterprises should think of developers as the API's customers. APIs should not merely expose digital assets—they should make the asset easy to consume and clearly present its value proposition. APIs that fail this baseline may struggle to gain adoption no matter if they carry a price tag, are given away for free to external audiences or introduced to in-house developers as internal resources.

Field-tested best practices

Many key API monetization best practices are encapsulated in "5 A's": Assets, Audience, Access, Adoption, and Administration and operations.

Assets: What business capabilities or data do you have that people want to use?

An API monetization strategy may start with identifying business capabilities or digital assets that have value outside the organization or even to other teams teams within the organization. Once valuable assets have been identified, they can be exposed as APIs, related assets can be grouped into API products, and related API products can be bundled into API packages, in order to target various developer needs.

SECTION SUMMARY

Define your valuable assets

- Identify valuable digital assets that could be useful to third parties or to other teams throughout the organization.
- Explore how those assets may be exposed as APIs, and bundled into products.

Audience: Who are the people who want to use your business capabilities?

To understand how to package digital assets into APIs that developers will use, an enterprise may need to identify constituencies of target customers and work to understand their needs. In some cases, the constituency is composed of communities of developers, and in some cases, the constituency is a set of business partners. Either way, an API provider should strategize from user needs rather than assumptions formed inside the organization. We call this approach working from the "outside-in."

Though an outside-in approach may suggest consideration of only an API provider's external customers or partners, that is not always the case, as developer communities exist within organizations as well. When thinking about API productization and monetization, APIs intended for internal developers should often be part of the same conversation as those intended for external developers.

Some internal APIs prove themselves so valuable that they become candidates for externalization and even monetization—but if those internal APIs are not managed like products, this opportunity may never be recognized. Moreover, monetization features applied to purely internal APIs can help businesses track finances within internal departments.

Enterprise leaders must also understand, as noted above, whether the API's target audience is more valuable under a direct-monetization model or under a model that seeks indirect benefits. Many people will use a resource, but only some within that group will be willing to pay to use that resource. Among those willing to pay, some will first need to first try an API for free, and some will need to be offered flexible billing that scales to their needs and uses. To find the right monetization mix, enterprises may need tools to tune monetization strategies on the fly and to pursue many models. Popular API monetization models include:

- **Freemium:** The API provider offers a basic level of service, defined in terms of quotas and/or capabilities, for free. Higher tiers of service that offer more capabilities or higher usage quota are offered for a fee.
- **Usage-based:** The API customer is charged for each transaction. Possible variations include
  - ☐ *flat rate models* in which the developer is charged a fixed rate for each transaction;
  - ☐ *volume-banded models* in which the developer is charged a variable rate depending on transaction volume;
  - ☐ usage-based models in which *custom attributes* dictate fees. For example, the developer may be charged for each transaction but the amount charged varies based on the number of bytes transmitted.
- **Revenue sharing:** The API provider shares with the developer a percentage of the revenue generated from each transaction. Possible variations include
  - ☐ *fixed share models* in which the developer receives a set percentage of revenue generated from each transaction;
  - ☐ *flexible share models* in which the developer receives a variable percentage of revenue generated, based on total revenue generated across API transactions over a specific period of time.

SECTION SUMMARY

Know your audience

- Identify your asset's target market, whether a community of developers or B2B customers, and how best to address that market's needs.
- View both internal and external developers as customers and manage both internal-facing and external-facing APIs as products.
- Remember: some APIs can be directly monetized, and many monetization models are possible—but many APIs are more powerful if they're given away to drive adoption and generate indirect value.

Access: How do people find and access your product?

Most products need a store—and in the case of API products aimed at the open market, that store often takes the form of either API marketplaces or a custom developer portal: a one-stop destination where customers can discover, explore, access, and test a provider's APIs.

Providers that build API portals should include a catalog that makes it easy for developers to discover APIs, and they should include documentation, sample code, blogs, and forums that clearly express the APIs' value and make it easy to start working with the APIs within minutes. Developers visiting the portal should be able to access API keys and otherwise onboard themselves via self-service processes, and the portal should include sandbox environments for developers to safely experiment.

SECTION SUMMARY

Make your APIs accessible

- Identify API marketplaces where target developers are already active, and consider investing in a developer portal to serve as a first-party storefront for APIs.
- Go beyond providing access to APIs. Provide sample code, documentation, testing tools, and other support materials to help developers make the most of your APIs and understand the value of your monetization packages.

Adoption: How do you market your product?

Various forms of marketing can also play an important role in driving adoption of APIs. In forming an outside-in viewpoint, enterprises should task evangelists with forming relationships with developer communities and create a presence at conferences and developer gatherings. A developer portal should include forums and other community features that allow users to exchange best practices and provide feedback to the API provider. SEO, SEM, social media, and other mainstream marketing efforts may be important as well.

SECTION SUMMARY
Spread the Word

- To make developers aware of your APIs, consider investing in marketing and appointing an API evangelist.

Administration and operations: How do you service and improve your product?

As with any product, a monetized API that is popular today may not be as lucrative in the future if it isn't updated and supported. API consumers are *especially* likely to have higher expectations for APIs they pay to access, so they will expect a high level of service, support, continuous improvement, and so on.

To meet these evolving expectations, it's important that API providers maintain uptime of their services and be able to change pricing models as developer behavior changes. It should almost go without saying that APIs providers must protect their products from threats, as developers will abandon an unsafe service.

API providers also need a strategy that encompasses both technical operations tasks, such as release management and API monitoring, as well as business operations tasks, including generating insights from analytics and using those insights to form strategies for growth. This means API providers need to continuously monitor their products so they can quickly detect and address service disruptions. Providers should use API analytics to observe how APIs are being used by different developer groups, learn how different monetization models and product features are affecting adoption, and shape iterations so that their APIs become more useful in the future. API providers should invest in tools that allow them to flexibly and responsibly modify pricing and A/B test different offerings.

SECTION SUMMARY

Keep delighting API customers

- Invest in testing and monitoring tools that help detect and resolve API-related issues.
- Adopt analytics tools that provide insight into how APIs are being used and how APIs can be improved.
- Prioritize the ability to flexibly tune API pricing models based on changes in user behavior.
- Protect customers by investing in proactive monitoring and security against bots and other threats.
- Make APIs more useful over time by iterating based on API usage trends, user feedback, and other data.

Telstra: Reaping broad benefits from API revenue

For Telstra, the largest telecommunications service provider in Australia, API monetization has played a key role in not only attracting new business, but also in raising the profile of the company's API program to key decision makers within the company.

When Telstra launched its first public API—a basic SMS messaging API—in 2015, its goal was both to experiment with ways to modernize its legacy systems and products, and to make those systems and products more modular and user-friendly for internal developers, said Steven Cooper, Telstra's API and platform evangelist.

"It was geared toward testing the waters and gauging what was needed internally for publicly available APIs and services," Cooper said.

"As soon as you start monetizing things, it really changes the way the business thinks about APIs"

David Freeman, Telstra

The response was impressive: within a few months, over 3,000 developers registered to use the API. With this initial success under its belt, Telstra began to focus on attracting new audiences. It launched its developer portal in 2017 as an online storefront for its APIs and built a roadmap for future API products.

As Telstra built up its ability to monetize its APIs and distribute them through its portal, new doors opened to new audiences. The opportunity to access and incorporate Telstra's APIs into new applications could be extended not only to enterprise partners but to individual developers and small teams—a dynamic that significantly expanded the range of developers innovating with Telstra's services.

Through its developer portal, developers sign up for limited free trials to experiment with Telstra APIs, including 1,000 free calls to Telstra's SMS and MMS APIs. There's also a rate package based on API usage for developers who need greater call volume.

"Telstra had some great traditional products out there such as SMS and MMS," said David Freeman, Telstra's general manager of API enablement. "What we've seen is that we can actually turn those into real, monetizable API products."

Monetization also provides a new and important metric for the Telstra API team: revenue.

"What are the quality metrics we really need to measure? We've taken a bit of a change on that now—we're looking at things like revenue because our program has matured," Freeman said.

Another very important achievement that Telstra's API team realized by implementing monetization: producing tangible revenue has gotten the attention of Telstra's business leaders and impressed upon them the value that comes from exposing telco services as APIs.

"As soon as you start monetizing things, it really changes the way the business thinks about APIs," Freeman said. "That helps us punch above our weight within the organization and helps us to have a voice."

Monetization checklist

Here are the key capabilities that businesses should consider when forming API monetization strategies and evaluating API monetization solutions:

- **Assets: Package valuable digital assets into monetizable products that can be easily consumed by developers**
  - ☐ Package a collection of related RESTful APIs as an API product
  - ☐ Create rules to define which API calls qualify as monetized transactions and to enforce monetization limits on API proxies
  - ☐ Issue PCI DSS compliance certificate, specify terms and conditions and support multiple currencies
  - ☐ Dig into API traffic analytics to determine which APIs are most popular and which APIs might be ripe for monetization
- **Audience: Manage your customers' experiences and options**
  - ☐ Support for various monetization models: revenue sharing, usage-based billing, subscriptions, freemium, etc.
  - ☐ Create and manage rate plans
  - ☐ Manage renewals and notify customers of updates with notification templates, support webhooks, etc.
  - ☐ Measure developer engagement and funnel analytics to top developers and apps that use your APIs
- **Access: Help your customers find and use your API products**
  - ☐ Create an appealing and intuitive self-service developer portal
  - ☐ Publish API packages and rate plans, and integrate with payment providers
- **Adoption: Help your customers hear and learn about your API products**
  - ☐ Facilitate community among the developers using your APIs and create feedback loops with forums, blogs, and other outreach channels
- **Administration and operations: Track metrics and derive insights that help you improve your API products**
  - ☐ View monetization-specific reports for billing, prepaid balance, variance, and revenue use cases
  - ☐ Set up alerts to notify administrators of API issues
  - ☐ Investigate API issues to identify the root cause of problems, regardless of whether the use case involves a developer app, proxy or backend target

About Apigee API Management

The Apigee API management platform delivers full lifecycle API management to help businesses unlock the value of data and securely deliver modern applications. Apigee offers a rich set of capabilities to enable enterprises to monetize their APIs, including tools to test and flexibly tune pricing models, robust monitoring tools, and API monetization.

Now that you've finished reading, why stop learning?
Visit the Apigee website for more.

apigee

Share this eBook

on social

with a colleague

Google Cloud