10 MUST HAVES FOR A GREAT API

Having a great API is more than a feeling - here are the top ten things you need to know.

The tldr version? Right here

10 An Up-to-Date Knowledge Base and Online Documentation with Examples
9 The Ability to Interactively Try it Out Online
8 Internal Consistency
7 Consistent Simplicity
6 Client Libraries for Popular Languages and Frameworks
5 Easy to Use Directly as RAW JSON
4 Meaningful Error Codes and Messages
3 Logging Within the Application
2 Common Up/Down Objects
1 The Number One Feature of a Great API? (Look below)

The important details? Catch them below.

10 An Up-to-Date Knowledge Base and Online Documentation with Examples



You're jumping into a new API, and you need documentation - documentation on how to get started. You'll find this in the form of a knowledge base. Ideally, your knowledge base gives example code for common use cases you are building.

After you are up and working, you'll be ready to dig into all the ins and outs of the API. At this point, you need a full reference that gives all the details, including the resources, methods, and the level of every field.

09 The Ability to Interactively Try it Out Online



You know you have a leading-edge online documentation tool when you can make requests to the API through the documentation page itself. Instead of cycling between reading the page and editing your code to try something new, you can play with the API live - right there on the page. This option drastically shortens cycle-time when you're testing new changes or resources.

08 Internal Consistency



The best user guide is one that you never have to crack open - because you already know what to do, thanks to past experience. An internally consistent API gives you this ability.

In the world of API, internal consistency means that every endpoint exposes the same operations with the same syntax and the same kind of output. It means that a field has the same name on each resource it's exposed on. You'll never be surprised at the responses you receive. And you will be confident using the full API without scouring the documentation first.

07 Consistent Simplicity



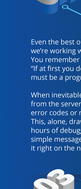
As engineers, we tend to fall into the trap of complexity. Writing a complex piece of logic is alluring - it's something you can brag about to your colleagues. It's also a sure-fire way to antagonize the users of your API. APIs need to be simple to be loved, as simple as they can be while still working at top capacity. Own that challenge and you'll succeed as an API designer.

06 Client Libraries for Popular Languages and Frameworks



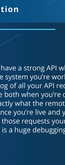
Ideally, all you need to take off with a new API is to download its library, add it to your package manager, import the objects, and go. So having a vendor who provides a primed-and-ready library is crucial. It allows you to ignore so many questions during implementation: "Is it JSON or XML?" "How are you doing authentication?" "What error codes do you return?" A pre-built library allows you to perform at your highest capacity since these details are already covered.

05 Easy to Use Directly as RAW JSON



On the other hand, if the ONLY way to use an API is through its library, it's functionality is compromised. There are so many great tools out there (Postman, Paw, and the ubiquitous command line with curl), and your ability to make ad-hoc calls to an API is a huge advantage. Say you need to see what all possible values from a resource look like. Don't build it into your app; just drop it into Paw and get an instant result. Or maybe you need to make a one-off call to update a record; five minutes in the shell can save you an application build, deploy, and release cycle.

04 Meaningful Error Codes and Messages



Even the best of us get errors when we're working with a partner's API. You remember the old saying... "If at first you don't succeed, you must be a programmer."

When inevitable errors bounce back from the server, the quality of the error codes or messages is crucial. This, alone, draws the line between hours of debugging or reading a simple message and then getting it right on the next try.

03 Logging Within the Application



You know you have a strong API when you can log into the system you're working with and see a full log of all your API requests. This is valuable both when you're developing - to see exactly what the remote end is getting - and once you're live and you're no longer making those requests yourself. A built-in API log is a huge debugging tool.

02 Common Up/Down Objects

This point concurs with the internal consistency point above. An internally consistent API equips you to download an object (i.e., from a GET), modify the fields you need to update, then send it back to the server (i.e., PUT or PATCH) to update it. Or, maybe clear out its ID and send it back (POST)



to create a new duplicate record. A robust API gives you that same object, no matter what direction it's moving.

01 The Number One Feature of a Great API - An Extraordinary Support Team



So you've got all the functionality, dexterity, and resources at hand. But you've got an inkling that these aren't all you'll need. You need grit, compassion, intuition. It's the extraordinary people behind a great API that make all the difference. You need to be able to ask questions when you're

legitimately stuck. You need to be able to talk with someone who can truly understand what you're aiming for and who will give you what you need, not just what you're saying you want. You need a partner whose goal is your success.

infoplus