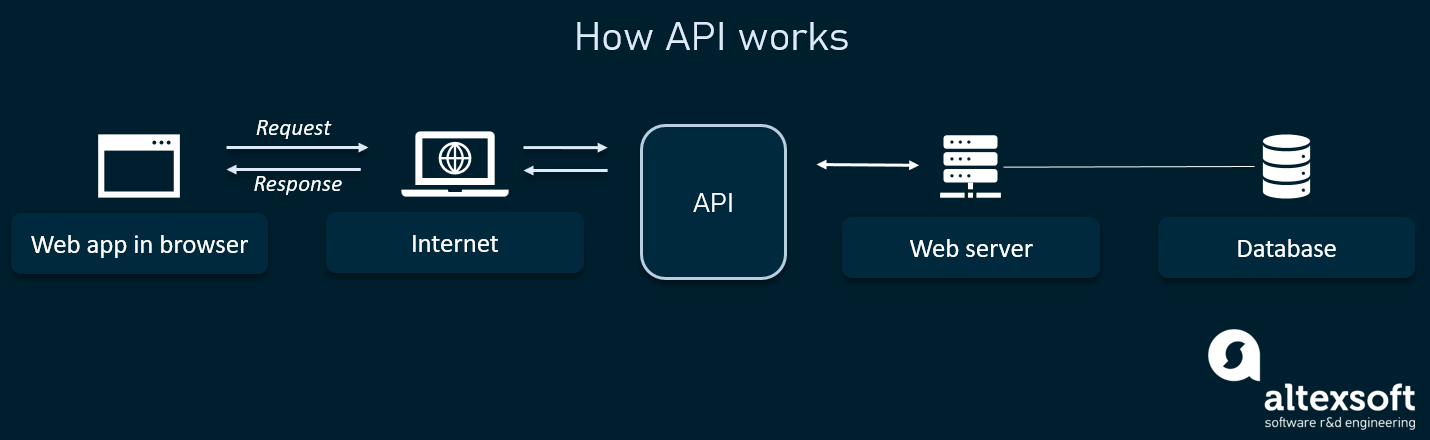
**Q. What is an API?**

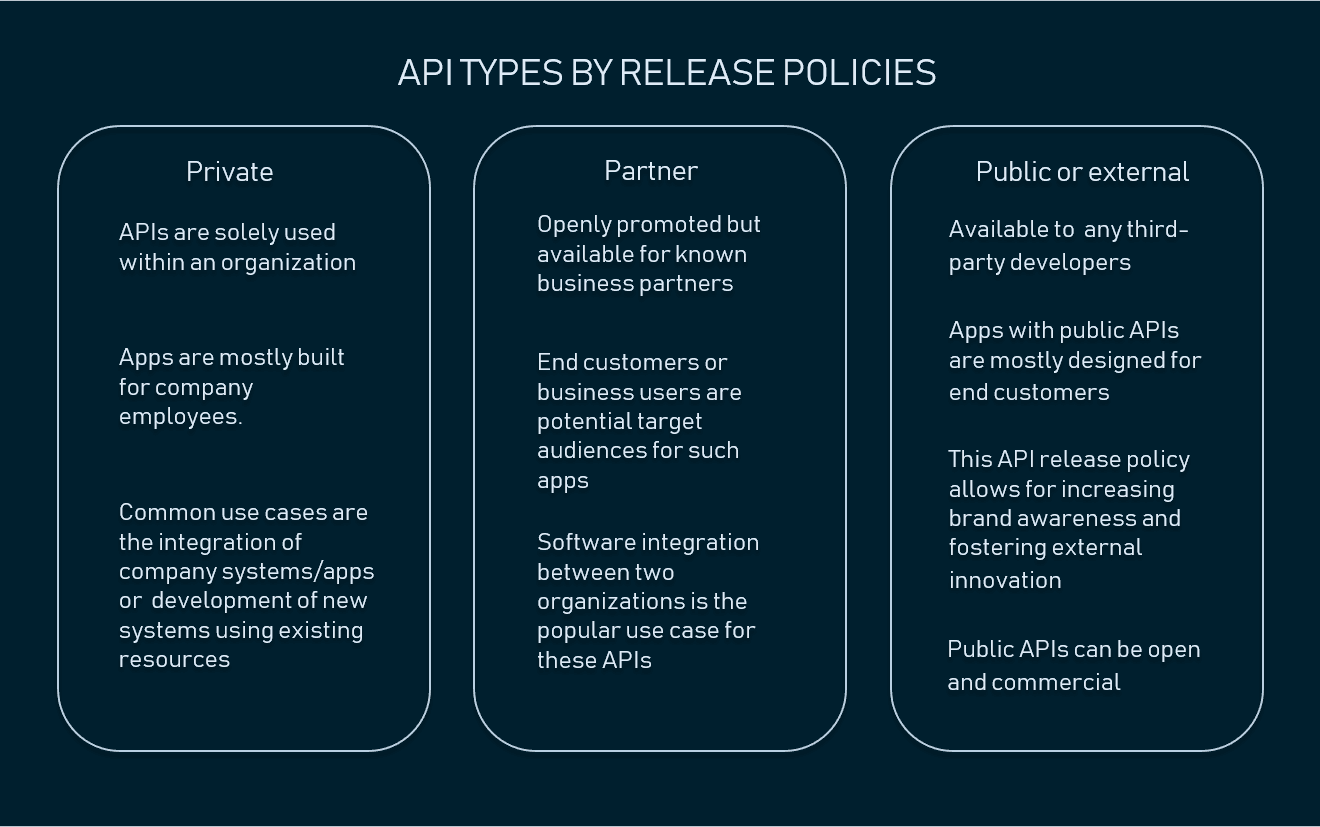
An**API**is a set of programming code that enables data transmission between one software product and another. It also contains the terms of this data exchange. [API stands for Application Programming Interface](https://rapidapi.com/blog/api-glossary/api/), which is a mechanism that allows the interaction between two applications using a**set of rules**.



APIs are beneficial because they allow developers to add specific functionality to an application, without having to write all of the code themselves. APIs also allow developers to access data from other applications.

# For Example: Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you’re using an API.

**# Basic Types of API:**



## # Main types of Web APIs:

There are 4 main types of APIs:

1. **Public / Open APIs:**Also known as Public API, there are no restrictions to access these types of APIs because they are publicly available.
2. **Partner APIs:**A developer needs specific rights or licenses in order to access this type of API because they are not available to the public.
3. **Private / Internal APIs:**Also known as Private APIs, only internal systems expose this type of API. These are usually designed for internal use within a company. The company uses this type of API among the different internal teams to be able to improve its products and services.
4. **Composite APIs:** This type of API combines different data and service APIs. It is a sequence of tasks that run synchronously as a result of the execution, and not at the request of a task. Its main uses are to speed up the process of execution and improve the performance of the listeners in the web interfaces.

## # APIs By Use Cases:

APIs can be classified in 4 categories according to the systems for which they are designed :

1. **Database APIs. :**Database APIs enable communication between an application and a database management system. Developers work with databases by writing queries to access data, change tables, etc. [The Drupal 7 Database API](https://www.drupal.org/docs/7/api/database-api/database-api-overview), for example, allows users to write unified queries for different databases, both proprietary and [open source](https://www.altexsoft.com/blog/engineering/how-to-use-open-source-software-features-main-software-types-and-selection-advice/) (Oracle, MongoDB, PostgreSQL, MySQL, CouchDB, and MSSQL).

Another example is [ORDS database API](https://docs.oracle.com/en/database/oracle/oracle-rest-data-services/19.1/aelig/enabling-ords-database-api.html#GUID-8730051B-7C03-487B-954A-7D6786B7EC74), which is embedded into Oracle REST Data Services.

1. **Operating systems APIs. :**This group of APIs defines how applications use the resources and services of operating systems. Every OS has its set of APIs, for instance, [Windows API](https://docs.microsoft.com/en-us/windows/desktop/apiindex/windows-api-list) or Linux API ([kernel–user space API](https://www.kernel.org/doc/html/v4.15/userspace-api/index.html)and [kernel internal API](https://www.kernel.org/doc/html/latest/driver-api/media/index.html)).

Apple provides API reference for macOS and iOS in its [developer documentation](https://developer.apple.com/documentation/). APIs for building applications for Apple’s macOS desktop operating system are included in the Cocoa set of developer tools. Those building apps for the iOS mobile operating system use Cocoa Touch – a modified version of Cocoa.

1. **Remote APIs. :**Remote APIs define standards of interaction for applications running on different machines. In other words, one software product accesses resources located outside the device that requests them, which explains the name. Since two remotely located applications are connected over a communications network, particularly the internet, most remote APIs are written based on web standards. [Java Database Connectivity API](https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/) and [Java Remote Method Invocation API](https://docs.oracle.com/javase/8/docs/technotes/guides/rmi/index.html) are two examples of remote application programming interfaces.
2. **Web APIs. : This API class is the most common.** Web APIs provide machine-readable data and functionality transfer between web-based systems which represent [client-server architecture](https://www.techopedia.com/definition/438/clientserver-architecture). These APIs mainly deliver requests from web applications and responses from servers using Hypertext Transfer Protocol (HTTP).

Developers can use web APIs to extend the functionality of their apps or sites. For instance, the [Pinterest API](https://developers.pinterest.com/) comes with tools for adding users’ Pinterest data like boards or Pins to a website. [Google Maps API](https://developers.google.com/maps/) enables the addition of a map with an organization’s location.

Most businesses use more than one API to connect applications and share information.Some end up needing an API management tool to help them control, distribute, and analyze different APIs.

## # Web service APIs :

Apart from the main web APIs, there are also web service APIs:

1. [SOAP](https://rapidapi.com/blog/soap-vs-rest-api/) (Simple Object Access Protocol).
2. XML-RPC (Remote Procedural Calls).
3. [JSON](https://rapidapi.com/blog/api-glossary/json/)-RPC (Remote Procedural Calls).
4. REST (Representational State Transfer).

A web service is a system or software that uses an address, i.e., URL on the World Wide Web, to provide access to its services.

The following are the most common types of web service APIs:

* **SOAP (Simple Object Access Protocol):**This is a protocol that uses only XML as a format to transfer data. Its main function is to define the structure of the messages and methods of communication. It also uses WSDL(Web Services Definition Language), in a machine-readable document to publish a definition of its interface.
* **XML-RPC** (Remote Procedural Calls)**:**This is a protocol that uses a specific XML format to transfer data compared to SOAP that uses a proprietary XML format. It is also older than SOAP. XML-RPC uses minimum bandwidth and is much simpler than SOAP. Example

**<employees>**

**<employee>**

**<firstName>Becky</firstName> <lastName>Smith</lastName>**

* **JSON-RPC** (Remote Procedural Calls)**:** This protocol is similar to XML-RPC but instead of using XML format to transfer data it uses JSON. Example

**{"employees":[**

**{ "firstName":"Becky", "lastName":"Smith" }]},**

* **REST (Representational State Transfer):** REST is not a protocol like the other web services, instead, it is a set of [architectural](https://rapidapi.com/blog/api-architecture/) principles. The REST service needs to have certain characteristics, including simple interfaces, which are resources identified easily within the request and manipulation of resources using the interface.

### # What are the differences between [SOAP and REST](https://rapidapi.com/blog/soap-vs-rest-api/) ?

|  |  |
| --- | --- |
| **SOAP** | **REST** |
| It has strict rules and advanced security to follow. | There are loose guidelines to follow allowing developers to make recommendations easily |
| It is driven by Function | It is driven by Data |
| It requires more Bandwidth | It requires minimum Bandwidth |

### # What are the differences between JSON and XML ?

|  |  |
| --- | --- |
| **JSON** | **XML** |
| Supports only text and numbers. | Supports various types of data for example text, numbers, images, graphs, charts etc. |
| Focuses mainly on Data | Focuses mainly on Document. |
| It has low security | It has more security |

**# Choosing the Right Type of API :**

To summarize, we can group web APIs into four broad categories:

* **Open APIs**, which any developer can access.
* **Partner APIs**, which only authorized developers may access.
* **Internal APIs**, which only internal teams may access.
* **Composite APIs**, which combine multiple APIs.

There are also three common types of API architectures:

* **REST**, a collection of guidelines for lightweight, scalable web APIs.
* **SOAP**, a stricter protocol for more secure APIs.
* **RPC**, a protocol for invoking processes that can be written with XML (**XML-RPC**) or JSON (**JSON-RPC**).

If you’re not someone who needs to know the gritty, this is a good baseline understanding of the types of APIs out there and how developers use them. By harnessing the right API, you’ll empower your business to partner with other applications, expanding your reach and influence.

## # API specifications/protocols :

### **Remote Procedure Call (RPC):** Today, the majority of web APIs are built on REST. REST, which stands for representational state transfer, is a set of guidelines for scalable, lightweight, and easy-to-use APIs. A REST API (or “RESTful” API) is an API that follows REST guidelines and is used for transferring data from a server to a requesting client.

For a more in-depth look at REST guidelines, see our full [**guide to REST APIs**](https://blog.hubspot.com/website/what-is-rest-api). Briefly, these guidelines are:

* **Client-Server Separation:** All client-server interactions must be in the form of a request from the client, followed by a response from the server. Servers can’t request and clients can’t respond.
* **Uniform Interface:** All requests and responses must use HTTP as the communication protocol and be formatted in a specific way to ensure compatibility between any client and any server. Server responses are formatted in JavaScript Object Notation (JSON).
* **Stateless:** Each client-server interaction is independent of every other interaction. The server stores no data from client requests and remembers nothing from past interactions.
* **Layered system:** Requests and responses must always be formatted the same way, even when passed through intermediate servers between the client and the API.
* **Cacheable:** Server responses should indicate whether a provided resource can be cached by the client and for how long.

By following these guidelines, REST APIs can be used for quick, easy, secure data transfers, making them a popular choice among developers.

### **# Service Object Access Protocol (SOAP) :**

SOAP (Simple Object Access Protocol) is a protocol for transmitting data across networks and can be used to build APIs. SOAP is standardized by the [**World Wide Web Consortium (W3C)**](https://www.w3.org/) and utilizes [**XML**](https://blog.hubspot.com/website/what-is-xml-file) to encode information.

SOAP strictly defines how messages should be sent and what must be included in them. This makes SOAP APIs more secure than REST APIs, although the rigid guidelines also make them more code-heavy and harder to implement in general.

For this reason, SOAP is often implemented for internal data transfers that require high security, and the more flexible REST architecture is deployed more commonly everywhere else. But, one more advantage to SOAP is that it works over any communication protocol (not just HTTP, as is the case with REST).

### **# Representational State Transfer (REST) :**

### The RPC (Remote Procedural Call) protocol is the most straightforward of the three architectures. Unlike REST and SOAP that facilitate the transfer of data, RPC APIs invoke processes. In other words, they execute scripts on a server.

RPC APIs may employ either JSON (a JSON-RPC protocol) or XML (an XML-RPC protocol) in their calls. XML is more secure and more accommodating than JSON, but these two protocols are otherwise similar. Though the RPC protocol is strict, it's a relatively simple and easy way to execute code on remote networks.

RPC APIs are limited in their security and capabilities, so you likely won’t see them as often as REST or SOAP APIs on the web. However, it can be used for internal systems for making basic process requests, especially many at once.