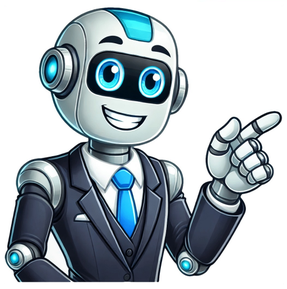
Continue



The API guideDavid FatehUpdated: January 15, 2025In this guide, you'll find everything you need to understand the fundamentals of APIs, how they enable seamless integration between systems, and why they are essential for modern digital experiences.Well explore topics like how APIs work, what API endpoints and keys are, how API calls function, and the role of headless APIs in powering flexible, scalable solutions. By the end, youll have a solid foundation to leverage APIs effectively in your projects.An application programming interface, typically shortened to API, is a set of rules which determines how one software program is able to access the data or functionality provided by another software program.Use of APIs dates back to the 1940s, when British computer scientists Maurice Wilkes and David Wheeler were programming the Electronic Delay Storage Automatic Calculator (EDSAC) to accept instructions like add, subtract, print, load, and store. This programmed behavior is similar to how modern web APIs interface with data for example, add blog post, delete blog post, or get blog post information.Today, APIs are an essential part of software development. They allow different systems and applications to communicate, exchange data, and share functionality in a flexible, efficient way.Theyre used in a wide variety of contexts, including web development, mobile apps, and Internet of Things (IoT) applications.Read on belowAPIs enable applications to exchange data as part of a request and response process. They typically come in the form of a library that a software developer can include in their applications code. This library contains a set of functions that facilitate various tasks between one application and another. The API specifies the function calls, the inputs that the applications accept, and the outputs that they return.Heres a simplified overview of the process:A client sends a request or call to an API server, usually over the internet or a local network. The request is made using a specific protocol, such as hypertext transfer protocol (HTTP), and includes information about the operation that the client wants to perform, e.g., retrieving data or updating a resource.The API server receives and processes the request. That process may involve validating the request, authorizing the request, authenticating the client, or performing other necessary operations.The API server sends a response back to the client, which may include data, an error message, or a status code indicating the result of the operation.The client receives the response and processes it.Lets go deeper with a real-world example. Imagine youre building an app that displays the weather in your location and in other locations around the world. For your app to work, youll need a way to get the coordinates (latitude and longitude) of your location or of a chosen destination, and then match that information against the relevant weather forecast.You could do that by manually entering the information when a user requests it but thats likely a time-consuming and error-prone process. As an alternative, you could use an API provided by a company that maintains a database of meteorological data to search for the relevant forecast, and deliver the requested information to the user. The API would access and query the database, and then your app would simply display the information that it returned in a matter of seconds.For this kind of project, using an API saves a lot of time and effort, while also ensuring that the data is accurate and up to date. Beyond the possibilities for weather apps, you can use APIs in a multitude of scenarios to access a wide variety of data and functionality, including retrieving the latest news headlines, looking up local restaurants, or finding information about a specific product.Heres a step-by-step guide to finding and using an API: StepDescription1. Find an APIFind an API that provides the functionality you need. There are many APIs available on the internet, and you can search for APIs using search engines, developer portals, or API marketplaces.2. Read the documentationOnce you have found an API that you want to use, read the API documentation. The documentation should provide information on the APIs capabilities, the structure of the requests and responses, and any authentication or authorization requirements.3. Understand the API's terms of useBefore you begin using an API, make sure you understand its terms of use. The terms typically specify limitations on how you can use the API, and how you can use and share the APIs data. They may also explain whether the API is free or requires a paid subscription.4. Obtain an API keyMany APIs require you to obtain an API key before you can start using them. API keys are usually obtained via the API provider. They can be generated and managed through the API providers developer portal, and are often specific to a particular API or application. Youll typically receive your key after you sign up or after you agree to the APIs terms of use.5. Make API requestsOnce you have an API key and have read the API documentation, you can start making API requests also known as API calls. You can make API requests using a tool such as a command-line utility, or a library in your programming language of choice.6. Process the API responseWhen you make an API request, the API server will send a response back to you. The response may include data, an error message, or a status code indicating the result of the operation. Its important to process the API response and handle any errors or exceptions that may occur.7. Follow best practicesLearn and apply the APIs best practices, such as caching data to reduce the number of API requests, handling errors and exceptions gracefully, and respecting the APIs terms of use. By following best practices like these, youll ensure that your use of the API is efficient, reliable, and compliant.APIs and webhooks both enable different software components to communicate with each other over the internet, and both facilitate transfers of data. However, they differ in terms of functionality and the contexts in which they are normally used.APIs represent a communication layer between applications that facilitates an ongoing series of requests and responses. One app sends a request for data, another responds by sending that data, and the API defines the specific format and structure of that two-way, data-transfer conversation.While webhooks also send data between applications, that transfer isnt initiated by a request, but when a specific predefined event takes place. When the predefined event occurs, the server sends a message to a specified URL, usually in the form of an HTTP POST request. The client application processes the message and acts based on the data received. Webhooks enable real-time communication and event-driven architectures, and can be used to build more efficient and responsive systems.Both APIs and webhooks are incredibly useful tools for building interconnected systems, but are not interchangeable. To learn more about the differences, see our post comparing webhooks vs. APIs.APIs are extremely useful in situations where communication and interoperability between different systems and applications is needed. Here are a few examples:Web developmentAPIs are a core component of web-based applications which are accessible from any device with an internet connection. For example, an API would allow an app in your web browser to communicate with a server-side application or, alternatively, enable a web-based application to retrieve data from a database.Mobile appsYou can use APIs to build dedicated mobile apps that can access data and functionality from a server-side application or cloud service. For example, a social media app on your smartphone will use an API to retrieve a users feed or to publish a new post.Internet of ThingsAPIs also enable communication between IoT devices and other systems and applications. For example, an API can facilitate communication between a smart thermostat and a home automation system, or enable an IoT device to send data to a cloud service for analysis.Third-party servicesAPIs allow different applications and web services to work together and share data. For example, a customer relationship management (CRM) system would use an API to integrate with a marketing automation platform, or a project management tool would use an API to integrate with a time-tracking application.MicroservicesAPIs are extremely popular for building microservices architectures. A microservice is the result of breaking down a large application into smaller, independent services that communicate with each other through APIs. As distinct parts of a larger entity, the different services can be developed, tested, and deployed independently, which makes it easier to scale and maintain the overall application.Although there are many different types of API, some common categories include: TypeDescriptionOpen APIsThese are made available to developers outside the organization that created them. Open APIs provide access to a specific product or service, and are available to anyone who agrees to their terms of use. They are also known as external or public APIs.Internal APIsThese are used within an organization to share resources and functionality between different teams or systems. They arent usually available to developers outside the organization.Partner APIsThese APIs are available to a specific group of developers, usually partners or third-party developers, who have a business relationship with the organization that created the API. Users may need to complete an onboarding process to gain access to partner APIs.Composite APIsThese are APIs that combine multiple underlying APIs into a single interface. They provide a simplified way for developers to access multiple resources or functionality in a single call.Its worth noting that these categories arent mutually exclusive, and an API may fall into more than one depending on how its used.An API architecture refers to the overall design and structure of an API, including the way its organized, the types of requests it supports, the data formats and protocols it uses, and any security or authentication mechanisms it employs. Well-designed API architecture is flexible, scalable, and easy to use, with clear documentation and a robust support system for developers.Some of the most common API architectures include:Representational State Transfer (REST) APIsREST APIs are a type of web API that uses HTTP requests to manipulate data. Designed to be lightweight and flexible, REST APIs are great for building web services that are scalable and easy to maintain. REST APIs use a fixed set of HTTP verbs to perform operations on resources that are identified using a URI. Examples of these verbs are GET, POST, PUT, and DELETE. REST APIs are also useful for CRUD (Create, Read, Update, Delete) applications.Simple Object Access Protocol (SOAP) APIsSOAP APIs are a type of web API that uses XML (Extensible Markup Language) to encode messages. Theyre designed to be extensible and neutral, and are useful for building web services that are interoperable between different programming languages and platforms. A SOAP API uses a standardized messaging format and transport protocol (usually HTTP) to send and receive messages.Remote Procedure Call (RPC) APIsRPC APIs are a type of API designed to make it easy to invoke methods on remote objects, which is useful for building distributed systems. RPC APIs are based on a client-server model. The client sends a request to the server to execute a specific procedure or function, and the server returns the result to the client.GraphQL APIsDeveloped by Facebook as an alternative to REST APIs and SOAP APIs, GraphQL is a query language and runtime that creates a representation of data that is designed to feel familiar and natural, like a visual graph. The graph describes a data structure of a collection of objects or nodes that are connected to each other through a set of links, or edges. This visual approach makes it possible to display relationships between different objects in a user interface and, because the data structure is non-linear, one object can be connected to more than one other object, and relationships can be circular.GraphQL makes it possible for clients to request only the data they need, rather than receiving a fixed set of data from a server. The language is flexible and efficient while also being easy to use and learn, with a simple syntax and powerful tools for building and testing queries.A web API is an API thats accessed over the internet using HTTP. Web applications built with web APIs are accessible from any device with an internet connection.Its easy to assume that all APIs are web APIs, but this isnt the case. Some APIs function only within the confines of a computer operating system, allowing local programs and routines to interact with one another.APIs can be created by any group or individual, including software companies, government agencies, non-profit organizations, and open-source projects. Well-known companies that create APIs include:Google: Google is a multinational technology company that creates APIs for cloud computing, maps, search, machine learning applications, and more.Amazon: Amazon is a global ecommerce company that provides a variety of APIs for uses including ecommerce, fulfillment, and payment processing.Microsoft: Microsoft is a technology company that creates APIs for purposes such as cloud computing, productivity, and machine learning.Contentful: Contentful is a company that provides a composable content platform with an API-first content management system (CMS) at its heart. The platform includes tools and features to create, manage, and deliver content.Many other companies and organizations create APIs, for a spectrum of purposes, including social media, finance, healthcare, and education.An API-first CMS is a type of CMS that exposes its content and functionality through APIs. These types of CMS platforms provide a robust and flexible set of APIs to build a range of applications and integrations.Modular by design, an API-first CMS leverages a microservices-based architecture that allows different components of the CMS to be developed, tested, and deployed independently. Its also typically designed to be scalable and resilient, with the ability to handle large amounts of traffic and data without performance degradation.API-first CMSes are often used to build modern web and mobile applications that require flexible and scalable content management capabilities. They can also build integrations with other systems and applications, or enable real-time communication and event-driven architectures.Contentful provides a set of APIs to build and manage content-driven applications and websites.To use Contentfuls APIs, youll need to sign up for a Contentful account and create an API key. You can then use the API key to make API requests in order to access and manipulate content stored in Contentful.Heres a step-by-step guide to using Contentfuls APIs: StepDescription1. Sign up for a Contentful accountCreate a Contentful account and sign up for a free trial or a paid plan depending on your needs.2. Create an API keyNavigate to the API Keys section in your Contentful account settings. Here you can create multiple API keys for different purposes, including for different environments or applications.3. Read our API documentationContentful provides extensive documentation for our APIs, including detailed information on the structure of the requests and responses, and examples of how to use the APIs. Familiarize yourself with the documentation before you start using the APIs.4. Make API requestsYou can use a variety of tools and technologies to make API requests to Contentfuls APIs. Common options include command-line utilities such as cURL, libraries in programming languages such as JavaScript or Python, or web-based APIs such as GraphQL.5. Process the API responseWhen you make an API request to Contentful, the API server will send a response back to you. The response may include data, an error message, or a status code indicating the result of the operation. Its important to process the API response and handle any errors or exceptions that may occur.To learn more about APIs, integration, or how to use Contentful APIs for creating and managing your content, sign up for a free Contentful account and begin creating your space.Written byDavid Fateh is a software engineer with a penchant for web development. He helped build the Contentful App Framework and now works with developers that want to take advantage of it.Unlock the power of digital contentBuild experiences that drive your business forward with the composable content platform.Sign up freeContact sales When I started learning to code, the term API would always haunt me. I couldn't make sense of what it actually meant because I would hear people talking about APIs in different contexts. The biggest challenge was that I couldn't find resources to learn about APIs in simple terms. Now that I know how APIs work, I decided to write this guide for any newbies out there who are struggling to make sense of this not-so-complicated but still confusing topic in web development and software engineering. API stands for Application Programming Interface. The application can be any software that performs a specific task and the interface is a point where two applications communicate. One application acts as a client and the other acts as a server. A client asks for some resource, say for example a photo, and the server sends that photo to the client. The client here can be your mobile phone, desktop or laptop computer, or any device you use to surf the internet. And the server is a bigger computer that stores the data you want (a photo in our case).Suppose I want a nature photograph to upload to my travel blog. I might go onto the Unsplash website, type "nature: in the search bar, and it would return a large number of nature photographs. That's an API working behind the scenes to make the conversation between Unsplash and me happen. Computers follow a protocol to communicate with each other. A protocol is nothing but a set of rules that computers follow to communicate. Any computer that doesn't follow the protocol breaks the communication thread. You might have used Bluetooth to share data back in the day. Bluetooth is nothing but a protocol for mobile devices to communicate with each other at a shorter distance. When you ask your friend to send you photos of their last trip, your device acts as a client, and your friend's device (the one that sends photos) is the server. This is just an example of a protocol. We have a large number of protocols in the world of computer science one for almost anything.On the web, we use the HTTP protocol (which stands for Hyper Text Transfer Protocol). APIs available on the web use the HTTP protocol for a number of reasons - it's easy to use and it's popular, for example. Communications that take place over the HTTP protocol are also known as the request-response cycle because this is exactly how the protocol works. The client sends a request to the server and the server responds to the client regarding that request. Unlike humans, computers have to be rigid to communicate with each other or they break the communication. For this reason, a client (requesting computer/ device) needs a set of information to send with the request so the server responds accordingly. This information includes:URL a web address where you want to make a requestMethod whether you want data already stored somewhere or want to save new data in a databaseHeader all the relevant information about your request including in what format the client device expects to receive the dataBody the body contains the actual request data In our Unsplash example, the URL is . The method is GET because we want the server to get nature images back. The header includes information like the format our computer expects to get and accept like language meaning, the language of the device, our operating system, and so on. The body includes the data we need to send to the server, the nature keyword for example.There are four types of methods for HTTP requests which we will get back to in a moment. For now, just know that a method indicates what you want to do with the data available on the server. For example, whether you want that data as documents or you want to save a new entry in data saved somewhere.When a client makes a request, the server responds to that request. The response might be the data the client requested or an error. Just like a response, a request has a structure including a URL, status code, header and body. In a request, we have a method, which has four types. And in the response, we have a status code which indicates whether a request has been accepted or declined. There are four available HTTP methods, and each has its unique functionality.GET: as already discussed, this indicates that the client is requesting data to be sent from the server.POST: this method tells the server that the client wants to create a new entry in a database. For example, saving a new blog post in a database of all previous blogs.DELETE: as the name suggests, the client wants to delete a data record from a database.PUT: this method is used when a client wants to update or edit a data record. For example, changing your Facebook password. There is a huge list of HTTP status codes, but let's look at a few of the most common:200 OK: this indicates that the request was successfully fulfilled by the server201 CREATED: the data entry that you requested to create was created404 NOT FOUND: this indicates that the resource you requested wasn't found by the server500 INTERNAL SERVER ERROR: this means that an error occured at the server's end and it couldn't fulfill your requestThere is no need to memorize these status codes, as the list is huge and you will subconsciously learn them as you encounter them in your development journey.Still, there is a range of status codes that indicates a generic response, as you can see here:100s: Informational responses, indicating the request's progress200s: Success, indicating the request's success300s: Redirection, indicating the request had to redirect somewhere else400s: Client errors, indicating errors that occurred on the client side500s: Server errors, when the server fails to respond to a valid client request Remember how I told you that I got confused when people would talk about APIs in different contexts? That's because we have different types of APIs available as well. The ones we talked about in this article are web APIs that use the HTTP protocol. Developers can use them to create a better user experience for their users. Other types include internal APIs that are hidden from external users and that are used within a company only. There are also open APIs that are available to be used by anyone for free (like the open weather map API). You can have partner APIs that are shared among business partners only to carry out their business tasks, and composite APIs that sequentially combine multiple API requests into a single API call to reduce server load and create a faster experience. If you want to learn more about how to design APIs, here's a full book for you to get started.And you can learn more about types of APIs, testing tools, and documentation here.Here's a tutorial that'll teach you all about REST APIs.And here's a Fetch API cheatsheet to get you started learning about Fetch. An API is an interface for two computers to communicate in order to carry out tasks on the internet. APIs follow the HTTP protocol to communicate, which has a specific request and response structure. Different methods exist to perform different tasks and numerous status codes are available that indicate whether the request is successful, declined, or in a pending state. Interested in connecting on LinkedIn? Hit me up at Tooba Jamal. In order to fully understand the role that APIs play in our lives, it's important to understand how they have evolved. APIs have been around for decades, with modern web APIs first taking shape in the early 2000s. The history of APIs since that period can be roughly broken down into the following five phases:Phase 1: Commercial APIsIn the early 2000s, web APIs emerged as a new method for emerging startups to not only make products and services available online, but to also enable partners and third-party resellers to extend the reach of their platforms. This era of APIs was defined by Salesforce, eBay, and Amazon, and these companies continue to dominate the API playing field today.Phase 2: Social media APIsA shift in the API landscape occurred in the mid-2000s, as a new group of companiessuch as Flickr, Facebook, and Twitterrealized that APIs could change the way we share information with one another. While these APIs weren't as intrinsically linked to revenue as their commercial predecessors, they nevertheless provided significant value to their organizations. For instance, Facebook launched version 1.0 of its API in August of 2006, which allowed developers to access Facebook users' friends, photos, events, and profile information. This API played a crucial role in establishing Facebook as one of the most popular social networks in the world.Phase 3: Cloud APIsIn 2006, Amazon introduced Amazon Simple Storage (S3), which marked yet another turning point in the history of APIs. S3 is a basic storage service in which resources are accessible via API and CLI, and its pay-as-you-go model provides a cost-efficient way for organizations to monetize digital assets in the online economy. Just six months later, Amazon released Amazon Elastic Compute (EC2), which enabled developers to use web APIs to deploy infrastructure that would power the next generation of applications. Both S3 and EC2 continue to play an essential role in application development today.Phase 4: APIs for mobile applicationsThe world was introduced to Apple's iPhone and Google's Android in 2007. The ability to carry the web in our pockets radically changed how we liveand spurred a massive investment in mobile applications that are powered by APIs.For instance, Twilio launched its API-as-a-product platform in 2007, which allowed developers to make and receive phone calls from any cloud application. Instagram then launched its photo-sharing iPhone application in October 2010, and it had one million users just three months later. Instagram did not initially provide an API, but it began work on one in early 2011 in response to user demand. These API-first companies played an essential role in creating the blueprint for how APIs are delivered today.Phase 5: APIs for connected devicesAround 2010, some developers began using APIs to connect everyday objectssuch as cameras, thermostats, speakers, microphones, and sensorsto the cloud. This next generation of devices, which includes Fitbit, Nest, Alexa, can send and receive data, content, media, and other digital resources, further changing the way we interact with the world around us.Skip to contentAn API, or application programming interface, is a set of rules that lets two software systems share data and functionality. It acts as a bridge between applications, without exposing their internal workings.While the idea originated in the 1940s, the API economy exploded as microservices became more popular in the 2010s. In a microservice architecture, a suite of smaller, loosely coupled services work together to act as a single application, driving digital transformation. To do this, however, those services must interact seamlessly.Well walk you through the ins and outs of APIs, including how they work, what types exist, what purposes they serve, how to integrate them, and more.Pretty much anyone who knows anything about APIs will tell you theyre the software equivalent of a restaurant waiter taking your order, bringing it to the kitchen, and then returning to your table with the food you ordered.In more technical terms, an API generally serves as a set of rules between an application and a server that governs how they communicate with each other. The defined communication protocol allows apps to be developed, connected, and integrated more quickly and at scale.API request: A request originates from a client application and is processed via the APIs management system or the API gateway.API call: The APIs management system or the API gateway calls the web server or external program.API response: The server sends the requested information to the API.API data transfer: The API communicates the data to the client application that requested it.The term API is often used generically to refer to any API. But the truth is there are several different varieties. They can be broken down by management strategy and purpose.Public API: Also known as an open API or an external API, a public API is accessible to any outside developer. Some public APIs are free, but others have a subscription fee. Public APIs often restrict shared assets and have lower authentication measures.Partner API: Partner APIs are reserved for authorized business partners who have completed an onboarding process and received an official license. This gives the API owner more control over who uses the API and how they use it. Because of their more restricted access, partner APIs tend to have stronger security measures than public APIs.Private API: Also known as internal APIs, private APIs are used by internal software development teams to enhance productivity by connecting systems and data. They are a scalable solution that makes it easy to incorporate new systems with existing systems. Private APIs are not available to outside users, but some companies eventually make them public.Composite API: A composite API consists of two or more data or service APIs, allowing access to several endpoints per call. Bundling calls and responses reduces the server load to improve system speed. This is especially useful for a microservice architecture since a single task may need data from multiple sources.System API: A system API extracts data from core systems, such as an ERP or billing system.Process API: A process API combines data from multiple system APIs to break down data silos and improve agility.Experience API: An experience API targets user experience by reconfiguring data based on audience and context.A remote API interacts through a communications network. While not all remote APIs use the internet and an HTTP protocol, many do, which qualifies them as web APIs. That means that while web APIs are also remote APIs, some remote APIs are not web APIs. That said, most APIs fall into both categories.Similarly, a web service is a type of API, but not all APIs qualify as web services. A web service must communicate over a network and typically limits access to approved partners. It often uses the SOAP protocol, which well unpack that in the next section. API architectures and protocols control what data the API shares and how it shares that information. Three main options are available:The most common API architecture. REST APIs are lightweight, scalable, and designed for simplicity.Client-server separation: Clients and servers operate independently.Stateless: Every request is self-contained no memory of previous interactions.Uniform interface: Standard rules govern communication.Layered system: API calls can pass through multiple layers without changing format.Cacheable: Responses can be stored to improve performance.RESTful APIs are widely used in web services due to their flexibility and ease of integration.A protocol that defines exactly how APIs exchange messages. It uses XML and often runs over HTTP.What makes SOAP different:Strict formatting: Every message follows a fixed XML structure.Built-in security: SOAP includes standards for authentication and access control.More overhead: Powerful, but heavier and harder to implement than REST.SOAP is still popular in enterprise systems that require formal contracts and secure transactions.A protocol that lets one system trigger a function on another system, like calling a script remotely.Minimalist format: Uses JSON-RPC or XML-RPC to send instructions.Low complexity: Fast and lightweight, but lacks modern security features.Use case: Good for internal tools or simple systems, but rarely used for public-facing APIs.The uses of APIs are limited only by your creativity. However, many businesses use them for the following purposes:Expand reach: APIs can help their owners expand their reach and brand recognition. This is particularly true for a public API, around which an entire ecosystem of apps may spring up.Generate revenue: While some companies offer APIs for free to build relationships, others sell an API key, creating a direct revenue stream. In some cases, APIs can also provide revenue in other ways, such as by creating a sales funnel or distribution channel.Improve collaboration: The average enterprise uses a staggering number of cloud apps: nearly 1,200. Without APIs to connect them, the information in each application can become siloed, limiting collaboration.Enhance efficiency and scalability: Since APIs automate data handoff between applications, employees dont need to go through a time-consuming, error-prone manual process. API integrations can also improve scalability by simplifying the process of connecting systems and applications.Promote innovation: Because of their flexibility, APIs support innovation. They may allow companies to connect with new business partners, launch new services, and add value.Improve visibility: A streamlined system improves visibility into processes, allowing for better monitoring and reporting.APIs are so standard that you may have utilized one today without even realizing it. While APIs have widespread uses, here are a few common instances youre probably familiar with:If youve ever looked up airfare on an airlines website, you interacted with the airlines API. You requested flight information, and the API retrieved it from the companys database. Travel booking platforms use even more APIs to access information from multiple hotels and airlines.The Google Maps API gives developers access to a wealth of geographic data, including interactive maps. Businesses often leverage this tool to give potential customers clear driving directions to their locations.Many websites and apps allow users to sign in through other platforms and social media, like Google, Facebook, and Twitter. They do this by authenticating users through the services APIs, thereby saving users the trouble of setting up a new account and remembering yet another password.Pay with PayPal has become ubiquitous on e-commerce platforms. This API allows users to pay for items without exposing sensitive data.Depending on your companys needs, timeframe, and budget, several methods are available for integrating APIs. Review the API documentation and assess your goals to decide what is best for your company.Custom integrations are developed by a skilled expert to connect existing third-party systems for greater functionality. Since custom integrations tend to be costly and time consuming, they are fading in popularity.Connector applications serve as intermediaries between popular software platforms via their APIs to help manage and maintain integrations. Because connector applications tend to be affordable and easy to use, theyre quite popular. They may be classified by the app or protocol they support, so you can find options like Salesforce connectors, HTTP connectors, and JDBC connectors.API integration management platforms are typically SaaS applications used to connect other SaaS applications and cloud-based systems. With one request, they can collect and combine data from multiple sources in a single query. They act as Integration Platform as a Service (iPaaS) solutions.Most API errors break down into two main categories: request errors and server errors. You can often determine the cause of the error and how to fix it by the HTTP status code.Request errors are generally client-side errors. They have 400-level HTTP status codes, such as the following:400 Bad Request: This infuriatingly vague message just means something went wrong. Check your request and cross your fingers that you find an issue with your header, query, or body field.401 Unauthorized: If youre not authenticated, you could get this error. You may need to sign up and get an API key.403 Forbidden: While youre authenticated, youre not allowed to make the request based on your permissions. This may occur if you try to access features beyond your subscription plan or you use the wrong API key.404 Not Found: This is the most common error code, and it can have several potential meanings. While it usually indicates something is wrong on the client side, it can also result from a server problem. The first troubleshooting step should be checking the client code for typos.429 Too Many Requests: API subscriptions often have limits per minute, per hour, and/or per day. If you exceed those limits, you may get this message. Be aware of your limits and consider adjusting your subscription if necessary.If you see a 500-level HTTP status code, youre probably experiencing a server error. You may need to contact API support to resolve the issue.500 Internal Server Error: This generally means the API server crashed, but it may be worth checking your docs to make sure everything is right on your end.501 Not Implemented: If the API developer has not yet implemented the HTTP method you used to request a URL, you may get this status. Trying a different method may yield better results.502 Bad Gateway: Instead of reaching the API server, you contacted a gateway or a proxy server. While that server attempted to call the API server, the API server did not respond.503 Service Unavailable: If a server receives too many requests and is overloaded, youll get this message. You might try again later.504 Gateway Timed Out: This status is similar to 502 in that it means you contacted a gateway or a proxy server. However, in this case, the API server didnt respond quickly enough. If you are requesting a high volume of data, breaking it down into smaller chunks may help.APIs are the unsung heroes behind the digital tools we rely on every day. Whether you're connecting systems, automating tasks, or just trying to fix a mysterious 401 error, a solid grasp of API fundamentals can save you time and sanity. If you're looking to dig deeper into the tech that powers your environment, check out our Sysadmin Glossary for quick-hit definitions on everything from ACLs to XML. And if you ever hit a wall, the PDQ community is always a good place to trade tips, troubleshoot, or just swap war stories. PDQPDQ is the best way to have healthy, up-to-date machines automatically. Scan your environment for vulnerabilities and streamline your patch management and software deployment processes whether you manage 15 machines or 15,000. You've likely seen the term "API" come up. Operating system, web browser, and app updates often announce new APIs for developers. But what is an API and how do developers utilize them? What Is an Application Programming Interface The term API is an acronym, and it stands for "Application Programming Interface." Think of an API like a menu in a restaurant. The menu provides a list of dishes you can order, along with a description of each dish. When you specify what menu items you want, the restaurant's kitchen does the work and provides you with some finished dishes. You don't know exactly how the restaurant prepares that food, and you don't really need to. Similarly, an API lists a bunch of operations that developers can use, along with a description of what they do. The developer doesn't necessarily need to know how, for example, an operating system builds and presents a "Save As" dialog box. They just need to know that it's available for use in their app. This isn't a perfect metaphor, as developers may have to provide their own data to the API to get the results, so perhaps it's more like a fancy restaurant where you can provide some of your own ingredients the kitchen will work with. But it's broadly accurate. APIs allow developers to save time by taking advantage of a platform's implementation to do the nitty-gritty work. This helps reduce the amount of code developers need to create, and also helps create more consistency across apps for the same platform. APIs can control access to hardware and software resources. Let's say you want to develop an app for an iPhone. Apple's iOS operating system provides a large number of APIs---as every other operating system does---to make this easier on you. If you want to embed a web browser to show one or more web pages, for example, you don't have to program your own web browser from scratch just for your application. You use the WKWebView API to embed a WebKit (Safari) browser object in your application. If you want to capture photos or video from the iPhone's camera, you don't have to write your own camera interface. You use the camera API to embed the iPhone's built-in camera in your app. If APIs didn't exist to make this easy, app developers would have to create their own camera software and interpret the camera hardware's inputs. But Apple's operating system developers have done all this hard work so the developers can just use the camera API to embed a camera, and then get on with building their app. And, when Apple improves the camera API, all the apps that rely on it will take advantage of that improvement automatically. This applies to every platform. For example, do you want to create a dialog box on Windows? There's an API for that. Want to support fingerprint authentication on Android? There's an API for that, too, so you don't have to test every different Android manufacturer's fingerprint sensor. Developers don't have to reinvent the wheel over and over. APIs are also used to control access to hardware devices and software functions that an application may not necessarily have permission to use. That's why APIs often play a big role in security. For example, if you've ever visited a website and seen a message in your browser that the website is asking to see your precise location, that website is attempting to use the geolocation API in your web browser. Web browsers expose APIs like this to make it easy for web developers to access your location---they can just ask "where are you?" and the browser does the hard work of accessing GPS or nearby Wi-Fi networks to find your physical location. However, browsers also expose this information via an API because it's possible to control access to it. When a website wants access to your exact physical location, the only way they can get it is via the location API. And, when a website tries to use it, you---the user---can choose to allow or deny this request. The only way to access hardware resources like the GPS sensor is through the API, so the browser can control access to the hardware and limit what apps can do. This same principle is used on modern mobile operating systems like iOS and Android, where mobile apps have permissions that can be enforced by controlling access to APIs. For example, if a developer tries to access the camera via the camera API, you can deny the permission request and the app has no way of accessing your device's camera. File systems that use permissions---as they do on Windows, Mac, and Linux---have those permissions enforced by the file system API. A typical application doesn't have direct access to the raw physical hard disk. Instead, the app must access files through an API. APIs are used for all kinds of other reasons, too. For example, if you've ever seen a Google Maps object embedded on a website, that website is using the Google Maps API to embed that map. Google exposes APIs like this to web developers, who can then use the APIs to plop complex objects right on their website. If APIs like this didn't exist, developers might have to create their own maps and provide their own map data just to put a little interactive map on a website. And, because it's an API, Google can control access to Google Maps on third-party websites, ensuring they use it in a consistent way rather than attempting to messily embed a frame that shows the Google Maps website, for example. This applies to many different online services. There are APIs for requesting text translation from Google Translate, or embedding Facebook comments or tweets from Twitter on a website. The OAuth standard also defines a number of APIs that allow you to sign into a website with another service---for example, to use your Facebook, Google, or Twitter accounts to sign into a new website without creating a new user account just for that site. APIs are standard contracts that define how developers communicate with a service, and the kind of output those developers should expect to receive back. If you've gotten through this, you'll have a better idea of what an API is. Ultimately, you don't really need to know what an API is unless you're a developer. But, if you see a software platform or service has added new APIs for various hardware or services, it should be easier for developers to take advantage of such features. Skip to main content APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols. For example, the weather bureaus software system contains daily weather data. The weather app on your phone talks to this system via APIs and shows you daily weather updates on your phone. API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses. Their API documentation contains information on how developers are to structure those requests and responses. API architecture is usually explained in terms of client and server. The application sending the request is called the client, and the application sending the response is called the server. So in the weather example, the bureaus weather database is the server, and the mobile app is the client. There are four different ways that APIs can work depending on when and why they were created. SOAP APIs These APIs use Simple Object Access Protocol. Client and server exchange messages using XML. This is a less flexible API that was more popular in the past. RPC APIs These APIs are called Remote Procedure Calls. The client completes a function (or procedure) on the server, and the server sends the output back to the client. Websocket APIs Websocket APIis another modern web API development that uses JSON objects to pass data. A WebSocket API supports two-way communication between client apps and the server. The server can send callback messages to connected clients, making it more efficient than REST API. REST APIs These are the most popular and flexible APIs found on the web today. The client sends requests to the server as data. The server uses this client input to start internal functions and returns output data back to the client. Lets look at REST APIs in more detail below. REST stands for Representational State Transfer. REST defines a set of functions like GET, PUT, DELETE, etc. that clients can use to access server data. Clients and servers exchange data using HTTP. The main feature ofREST APIis statelessness. Statelessness means that servers do not save client data between requests. Client requests to the server are similar to URLs you type in your browser to visit a website. The response from the server is plain data, without the typical graphical rendering of a web page. Learn more about Rest API A Web API or Web Service API is an application processing interface between a web server and web browser. All web services are APIs but not all APIs are web services. REST API is a special type of Web API that uses the standard architectural style explained above. The different terms around APIs, like Java API or service APIs, exist because historically, APIs were created before the world wide web. Modern web APIs are REST APIs and the terms can be used interchangeably. API integrations are software components that automatically update data between clients and servers. Some examples of API integrations are when automatic data sync to the cloud from your phone image gallery, or the time and date automatically sync on your laptop when you travel to another time zone. Enterprises can also use them to efficiently automate many system functions. REST APIs offer four main benefits: 1. Integration APIs are used to integrate new applications with existing software systems. This increases development speed because each functionality doesnt have to be written from scratch. You can use APIs to leverage existing code. 2. Innovation Entire industries can change with the arrival of a new app. Businesses need to respond quickly and support the rapid deployment of innovative services. They can do this by making changes at the API level without having to re-write the whole code. 3. Expansion APIs present a unique opportunity for businesses to meet their clients needs across different platforms. For example, maps API allows map information integration via websites, Android,iOS, etc. Any business can give similar access to their internal databases by using free or paid APIs. 4. Ease of maintenance The API acts as a gateway between two systems. Each system is obliged to make internal changes so that the API is not impacted. This way, any future code changes by one party do not impact the other party. APIs are classified both according to their architecture and scope of use. We have already explored the main types of API architectures so lets take a look at the scope of use. Private APIs These are internal to an enterprise and only used for connecting systems and data within the business. Public APIs These are open to the public and may be used by anyone. There may or not be some authorization and cost associated with these types of APIs. Partner APIs These are only accessible by authorized external developers to aid business-to-business partnerships. Composite APIs These combine two or more different APIs to address complex system requirements or behaviors. API endpoints are the final touchpoints in the API communication system. These include server URLs, services, and other specific digital locations from where information is sent and received between systems. API endpoints are critical to enterprises for two main reasons: 1. Security API endpoints make the system vulnerable to attack. API monitoring is crucial for preventing misuse. 2. Performance API endpoints, especially high traffic ones, can cause bottlenecks and affect system performance. All APIs must be secured through proper authentication and monitoring. The two main ways to secure REST APIs include: 1. Authentication tokens These are used to authorize users to make the API call. Authentication tokens check that the users are who they claim to be and that they have access rights for that particular API call. For example, when you log in to your email server, your email client uses authentication tokens for secure access. 2. API keys API keys verify the program or application making the API call. They identify the application and ensure it has the access rights required to make the particular API call. API keys are not as secure as tokens but they allow API monitoring in order to gather data on usage. You may have noticed a long string of characters and numbers in your browser URL when you visit different websites. This string is an API key the website uses to make internal API calls. Due diligence and effort are required to build an API that other developers will want to work with and trust. These are the five steps required for high-quality API design: 1. Plan the API API specifications, like OpenAPI, provide the blueprint for your API design. It is better to think about different use cases in advance and ensure the API adheres to current API development standards. 2. Build the API API designers prototype APIs using boilerplate code. Once the prototype is tested, developers can customize it to internal specifications. 3. Test the API API testing is the same as software testing and must be done to prevent bugs and defects. API testing tools can be used to strength test the API against cyber attacks. 4. Document the API While APIs are self-explanatory, API documentation acts as a guide to improve usability. Well-documented APIs that offer a range of functions and use cases tend to be more popular in a service-oriented architecture. 5. Market the API Just as Amazon is an online marketplace for retail, API marketplaces exist for developers to buy and sell other APIs. Listing your API can allow you to monetize it. API testing strategies are similar to other software testing methodologies. The main focus is on validating server responses. API testing includes: Making multiple requests to API endpoints for performance testing. Writing unit tests for checking business logic and functional correctness. Security testing by simulating system attacks. Writing comprehensive API documentation is part of theAPI management process. API documentation can be auto-generated using tools or written manually. Some best practices include: Writing explanations in simple, easy-to-read English. Documents generated by tools can become wordy and require editing. Using code samples to explain functionality. Maintaining the documentation so it is accurate and up-to-date. Aiming the writing style at beginners Covering all the problems the API can solve for the users. The steps to implement a new API include: Obtaining an API key. This is done by creating a verified account with the API provider. Set up an HTTP API client. This tool allows you to structure API requests easily using the API keys received. If you dont have an API client, you can try to structure the request yourself in your browser by referring to the API documentation. Once you are comfortable with the new API syntax, you can start using it in your code. New web APIs can be found on API marketplaces and API directories. API marketplaces are open platforms where anyone can list an API for sale. API directories are controlled repositories regulated by the directory owner. Expert API designers may assess and test a new API before adding it to their directory. Some popular API websites include: Rapid API The largest global API market with over 10,000 public APIs and 1 million active developers on site. RapidAPI allows users to test APIs directly on the platform before committing to purchase. Public APIs The platform groups remote APIs into 40 niche categories, making it easier to browse and find the right one to meet your needs. APIForThat and APIList Both these websites have lists of 500+ web APIs, along with in-depth information on how to use them. An API Gateway is an API management tool for enterprise clients that use a broad range of back-end services. API gateways typically handle common tasks like user authentication, statistics, and rate management that are applicable across all API calls. Amazon API Gatewayis a fully managed service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale. It handles all the tasks involved in accepting and processing thousands of concurrent API calls, including traffic management, CORS support, authorization, and access control, throttling, monitoring, and API version management. GraphQL is a query language that was developed specifically for APIs. It prioritizes giving clients exactly the data they request and no more. It is designed to make APIs fast, flexible, and developer-friendly. As an alternative toREST, GraphQL

and personal applications. Here are some API examples that users often encounter: A popular API example is the function that enables people to log in to websites by using their Facebook, X, or Google profile login details. This convenient feature allows any website to use an API from one of the more popular services for quick authentication. This capability helps save users the time and hassle of setting up a new profile for every web application or new membership. These smart devices offer added functionality, such as internet-enabled touchscreens and data collection, through APIs. For example, a smart fridge can connect to recipe applications or take and send notes to mobile phones through text message. Internal cameras connect to various applications so that users can see the contents of the refrigerator from anywhere. Travel booking sites aggregate thousands of flights, showcasing the cheapest options for every date and destination. APIs enable this service by providing application users access to the latest information about availability from hotels and airlines. This access is available either through a web browser or the travel booking companys own application. With an autonomous exchange of data and requests, APIs dramatically reduce the time and effort involved in checking for available flights or accommodation. Navigation apps use core APIs that display static or interactive maps. These apps also use other APIs and features to provide users with directions, speed limits, points of interest, traffic warnings and more. Users communicate with an API when plotting travel routes or tracking items on the move, such as a delivery vehicle. Social media companies use APIs to allow other entities to share and embed content featured on social media apps to their own sites. For example, the Instagram API enables businesses to embed their Instagram grid on their website and for the grid to update automatically as users add new posts. APIs are an integral part of the growth in software as a service (SaaS) products. Platforms like CRMs (customer relationship management tools) often include several built-in APIs that let companies integrate with applications they already use, such as messaging, social media and email apps. This integration drastically reduces time spent switching between applications for sales and marketing tasks. It also helps reduce or prevent data silos that might exist between departments that use different applications. Traditionally, API referred to an interface connected to an application created with any of the low-level programming languages, such as JavaScript. However, modern APIs vary in their architectures and use of data formats. They are typically built for HTTP, resulting in developer-friendly interfaces that are easily accessible and widely understood by applications written in Java, Ruby, Python and many other languages. As the use of web APIs has increased, it has led to the development and use of certain protocols, styles, standards and languages. These structures provide users with a set of defined rules, or API specifications, that create accepted data types, commands and syntax. In effect, these API protocols facilitate standardized information exchange. SOAP is a lightweight XML-based messaging protocol specification that enables endpoints to send and receive data through a range of communication protocols including SMTP (simple mail transfer protocol) and HTTP (hypertext transfer protocol.) SOAP is independent, which allows SOAP APIs to share information between apps or software components running in different environments or written in different languages. Remote procedure call (RPC) is a protocol that provides the high-level communications paradigm used in the operating system. RPC presumes the existence of a low-level transport protocol, such as transmission control protocol/internet protocol (TCP/IP) or user datagram protocol (UDP), for carrying the message data between communicating programs. RPC implements a logical client-to-server communications system designed specifically for the support of network applications. The RPC protocol enables users to work with remote procedures as if the procedures were local.1 The XML-RPC protocol relies on a specific XML format to transfer data. XML-RPC is older than SOAP, but simpler, and relatively lightweight in that it uses minimum bandwidth. Like XML-RPC, JSON-RPC is a remote procedure call, that uses JSON (JavaScript Object Notation) instead of XML. JSON is a lightweight format for data exchange that is simple to parse and uses name/value pairs and ordered lists of values. Because JSON uses universal data structures, it can be used with any programming language. gRPC is a high-performance, open-source RPC framework initially developed by Google. gRPC uses the network protocol HTTP/2 and Protocol Buffers data format and is commonly used to connect services in a microservices architecture. WebSocket APIs enable bidirectional communication between client and server. This type of API does not require a new connection to be established for each communication, once the connection is established it allows for continuous exchange. This makes Web Socket APIs ideal for real-time communication. REST is a set of web API architecture principles. REST APIs,also known as RESTful APIs, are APIs that adhere to certain REST architectural constraints. REST APIs use HTTP requests such as GET, PUT, HEAD and DELETE to interact with resources. REST makes data available as resources, with each resource represented by a unique URI. Clients request a resource by providing its URI. REST APIs are stateless, they do not save client data between requests. Its possible to build RESTful APIs with SOAP protocols, but practitioners usually view the two standards as competing specifications. GraphQL is an open-source query language and server-side runtime that specifies how clients should interact with APIs.2GraphQL allows users to make API requests with just a few lines, rather than having to access complex endpoints with many parameters. This capability can make it easier to generate and respond to API queries, particularly more complex or specific requests that target multiple resources. SOAP and REST represent different approaches to API design, describing rules and standards for how an API should interact with other applications. SOAP is a protocol while REST is a set of constraints that constitute an architectural style. Both use HTTP to exchange information. REST is often considered a simpler alternative to SOAP because it is lightweight, flexible, transparent and relatively easy to use; SOAP requires users to write more code to complete each task than REST requires. SOAP is more deterministic and robust (due to type checking), and proponents make the case that it is easier to use because of the SOAP support built into many development tools.3 SOAP features built-in compliance, and developers often consider it a more secure protocol, better suited for situations with strict data integrity requirements. RESTful systems support messaging in different formats, such as plain text, HTML, YAML, XML and JSON, while SOAP only allows XML. Each has their strengths, and the right choice, might depend on use case. However, the ability to support multiple formats for storing and exchanging data is one of the reasons REST is a prevailing choice for building public APIs. GraphQL is a query language and API runtime that Facebook developed internally in 2012 before it becameopen sourcein 2015. GraphQL and REST are both stateless, use a client/server model and use HTTP. GraphQL solves for some limitations of REST, for example, providing the ability to more accurately target wanted resources with a single request. REST APIs follows a fixed structure, and always return a whole data set for a specified object. If the request is more complex, spanning multiple resources, for example, the client must submit separate requests for each resource. These limitations can lead to under or over-fetching issues. Neither REST nor GraphQL APIs are inherently superior. Theyre different tools that are suited to different tasks. REST is generally easier to implement and can be a good choice when a straightforward, cacheable communication protocol with stringent access controls is preferred (for public-facing e-commerce sites like Shopify and GitHub, as one example). GraphQL APIs enable more flexible, efficient data fetching, which can improve system performance and ease-of-use for developers. These features make GraphQL especially useful for building APIs in complex environments with rapidly changing front-end requirements.4 A web service is an internet software component that facilitates data transfers over a network. Because a web service exposes an applications data and functionality to other applications, in effect, every web service is an API. However, not every API is a web service. APIs are any software component that serves as an intermediary between two disconnected applications. While web services also connect applications, they require a network to do so. Web services are typically private and only approved users can access them. Microservicesis an architectural style that divides an application into smaller, independent components, often connected by using REST APIs. Building an application as a collection of separate services enables developers to work on one application component independent of the others, and makes applications easier to test, maintain and scale. Microservices architecture has become more prevalent with the rise ofcloud computingand, together withcontainersandKubernetes, is foundational to cloud-native application development. APIs simplify the design and development of new applications and services, and the integration and management of existing ones. They also offer significant benefits to developers and organizations at large. Improved collaboration The average enterprise uses almost1,200 cloud applications, many of which are disconnected. APIs enable integration so that these platforms and apps can seamlessly communicate with one another. Through this integration, companies can automate workflows and improve workplace collaboration. Without APIs, many enterprises would lack connectivity, causing information silos that compromise productivity and performance. Accelerated innovation APIs offer flexibility, allowing companies to make connections with new business partners and offer new services to their existing market. This flexibility also enables companies to access new markets that can boost returns and drive digital transformation. For example, the company Stripe began as an API with just seven lines of code. The company has since worked with many of the biggest enterprises in the world. Stripe has diversified to offer loans and corporate cards, and received a recent valuationofUSD 65 billion . Data monetization Many companies choose to offer APIs for free, at least initially, so that they can build an audience of developers around their brand and forge relationships with potential partners. If the API grants access to valuable digital assets, a business monetizes it by selling access. This practice is referred to as the API economy. WhenAccuWeatherstarted its self-service developer portal to sell a wide range of API packages, it took just 10 months to attract 24,000 developers, selling 11,000 API keys. This move helped to build a thriving community in the process. System security APIs separate the requesting application from the infrastructure of the responding service and offer layers of security between the two as they communicate. For example, API calls typically require authentication credentials. HTTP headers, cookies or query strings can provide additional security during data exchange. AnAPI gatewaycan control access to further minimize security threats. User security and privacy APIs provide added protection within a network. They can also provide another layer of protection for personal users. When a website requests a users location (a location API provides this information), the user can decide whether to allow or deny this request. Many web browsers and desktop and mobile operating systems have built-in permission structures. When an app must access files through an API, operating systems such as iOS, macOS, Windows and Linux use permissions for that access. For other uses, see API (disambiguation)."Api.php" redirects here. For the Wikipedia API, see Special:ApiHelp.An application programming interface (API) is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software.[1] A document or standard that describes how to build such a connection or interface is called an API specification. A computer system that meets this standard is said to implement or expose an API. The term API may refer either to the specification or to the implementation.In contrast to a user interface, which connects a computer to a person, an application programming interface connects computers or pieces of software to each other. It is not intended to be used directly by a person (the end user) other than a computer programmer[1] who is incorporating it into software. An API is often made up of different parts which act as tools or services that are available to the programmer. A program or a programmer that uses one of these parts is said to call that portion of the API. The calls that make up the API are also known as subroutines, methods, requests, or endpoints. An API specification defines these calls, meaning that it explains how to use or implement them.One purpose of APIs is to hide the internal details of how a system works, exposing only those parts a programmer will find useful and keeping them consistent even if the internal details later change. An API may be custom-built for a particular pair of systems, or it may be a shared standard allowing interoperability among many systems.The term API is often used to refer to web APIs,[2] which allow communication between computers that are joined by the internet. There are also APIs for programming languages, software libraries, computer operating systems, and computer hardware. APIs originated in the 1940s, though the term did not emerge until the 1960s and 70s.An API opens a software system to interactions from the outside. It allows two software systems to communicate across a boundary an interface using mutually agreed-upon signals.[3] In other words, an API connects software entities together. Unlike a user interface, an API is typically not visible to users. It is an "under the hood" portion of a software system, used for machine-to-machine communication.[4]A well-designed API exposes only objects or actions needed by software or software developers. It hides details that have no use. This abstraction simplifies programming.[5]Metaphorically, APIs connect software like interlocking blocks. Building software using APIs has been compared to using building-block toys, such as Lego bricks. Software services or software libraries are analogous to the bricks; they may be joined together via their APIs, composing a new software product.[6] The process of joining is called integration.[3]As an example, consider a weather sensor that offers an API. When a certain message is transmitted to the sensor, it will detect the current weather conditions and reply with a weather report. The message that activates the sensor is an API call, and the weather report is an API response.[7] A weather forecasting app might integrate with a number of weather sensor APIs, gathering weather data from throughout a geographical area.An API is often compared to a contract. It represents an agreement between parties: a service provider who offers the API and the software developers who rely upon it. If the API remains stable, or if it changes only in predictable ways, developers' confidence in the API will increase. This may increase their use of the API.[8]A diagram from 1978 proposing the expansion of the idea of the API to become a general programming interface, beyond application programs alone[9]The term API initially described an interface only for end-user-facing programs, known as application programs. This origin is still reflected in the name "application programming interface." Today, the term is broader, including also utility software and even hardware interfaces.[10]The idea of the API is much older than the term itself. British computer scientists Maurice Wilkes and David Wheeler worked on a modular software library in the 1940s for EDSAC, an early computer. The subroutines in this library were stored on punched paper tape organized in a filing cabinet. This cabinet also contained what Wilkes and Wheeler called a "library catalog" of notes about each subroutine and how to incorporate it into a program. Today, such a catalog would be called an API (or an API specification or API documentation) because it instructs a programmer on how to use (or "call") each subroutine that the programmer needs.[10]Wilkes and Wheeler's book The Preparation of Programs for an Electronic Digital Computer contains the first published API specification. Joshua Bloch considers that Wilkes and Wheeler "latently invented" the API, because it is more of a concept that is discovered than invented.[10]Although the people who coined the term API were implementing software on a Univac 1108, the goal of their API was to make hardware independent programs possible.[11]The term "application program interface" (without an -ing suffix) is first recorded in a paper called Data structures and techniques for remote computer graphics presented at an AFIPS conference in 1968.[12][10] The authors of this paper use the term to describe the interaction of an applicationa graphics program in this casewith the rest of the computer system. A consistent application interface (consisting of Fortran subroutine calls) was intended to free the programmer from dealing with idiosyncrasies of the graphics display device, and to provide hardware independence if the computer or the display were replaced.[11]The term was introduced to the field of databases by C. J. Date[13] in a 1974 paper called The Relational and Network Approaches: Comparison of the Application Programming Interface.[14] An API became a part of the ANSI/SPARC framework for database management systems. This framework treated the application programming interface separately from other interfaces, such as the query interface. Database professionals in the 1970s observed these different interfaces could be combined; a sufficiently rich application interface could support the other interfaces as well.[9]This observation led to APIs that supported all types of programming, not just application programming. By 1990, the API was defined simply as "a set of services available to a programmer for performing certain tasks" by technologist Carl Malamud.[15]Screenshot of Web API documentation written by NASAThe idea of the API was expanded again with the dawn of remote procedure calls and web APIs. As computer networks became common in the 1970s and 80s, programmers wanted to call libraries located not only on their local computers, but on computers located elsewhere. These remote procedure calls were well supported by the Java language in particular. In the 1990s, with the spread of the internet, standards like CORBA, COM, and DCOM competed to become the most common way to expose API services.[16]Roy Fielding's dissertation Architectural Styles and the Design of Network-based Software Architectures at UC Irvine in 2000 outlined Representational state transfer (REST) and described the idea of a "network-based Application Programming Interface" that Fielding contrasted with traditional "library-based" APIs.[17] XML and JSON web APIs saw widespread commercial adoption beginning in 2000 and continuing as of 2021. The web API is now the most common meaning of the term API.[2]The Semantic Web proposed by Tim Berners-Lee in 2001 included "semantic APIs" that recast the API as an open, distributed data interface rather than a software behavior interface.[18] Proprietary interfaces and agents became more widespread than open ones, but the idea of the API as a data interface took hold. Because web APIs are widely used to exchange data of all kinds online, API has become a broad term describing much of the communication on the internet.[16] When used in this way, the term API has overlap in meaning with the term communication protocol.The interface to a software library is one type of API. The API describes and prescribes the "expected behavior" (a specification) while the library is an "actual implementation" of this set of rules.A single API can have multiple implementations (or none, being abstract) in the form of different libraries that share the same programming interface.The separation of the API from its implementation can allow programs written in one language to use a library written in another. For example, because Scala and Java compile to compatible bytecode, Scala developers can take advantage of any Java API.[19]API use can vary depending on the type of programming language involved.An API for a procedural language such as Lua could consist primarily of basic routines to execute code, manipulate data or handle errors while an API for an object-oriented language, such as Java, would provide a specification of classes and its class methods.[20][21] Hyrum's law states that "With a sufficient number of users of an API, it does not matter what you promise in the contract: all observable behaviors of your system will be depended on by somebody."[22] Meanwhile, several studies show that most applications that use an API tend to use a small part of the API.[23]Language bindings are also APIs. By mapping the features and capabilities of one language to an interface implemented in another language, a language binding allows a library or service written in one language to be used when developing in another language.[24] Tools such as SWIG and F2PY, a Fortran-to-Python interface generator, facilitate the creation of such interfaces.[25]An API can also be related to a software framework: a framework can be based on several libraries implementing several APIs, but unlike the normal use of an API, the access to the behavior built into the framework is mediated by extending its content with new classes plugged into the framework itself.Moreover, the overall program flow of control can be out of the control of the caller and in the framework's hands by inversion of control or a similar mechanism.[26][27]An API can specify the interface between an application and the operating system.[28] POSIX, for example, specifies a set of common APIs that aim to enable an application written for a POSIX conformant operating system to be compiled for another POSIX conformant operating system.Linux and Berkeley Software Distribution are examples of operating systems that implement the POSIX APIs.[29]Microsoft has shown a strong commitment to a backward-compatible API, particularly within its Windows API (Win32) library, so older applications may run on newer versions of Windows using an executable-specific setting called "Compatibility Mode".[30] How much Microsoft developers' access to the company's operating systems' internal APIs is an advantage is unclear. Richard A. Shaffer of Technologic Computer Letter in 1987 compared the situation to a baseball game in which "Microsoft owns all the bats and the field",[31] but Ed Esber of Ashton-Tate said in an interview that year that Bill Gates told him that his developers sometimes had to rewrite software based on early APIs. Gates noted in the interview that Microsoft's Apple Macintosh applications were more successful than those for MS-DOS, because his company did not have to also devote resources to Mac OS.[32]An API differs from an application binary interface (ABI) in that an API is source code based while an ABI is binary based. For instance, POSIX provides APIs while the Linux Standard Base provides an ABI.[33][34]Remote APIs allow developers to manipulate remote resources through protocols, specific standards for communication that allow different technologies to work together, regardless of language or platform.For example, the Java Database Connectivity API allows developers to query many different types of databases with the same set of functions, while the Java remote method invocation API uses the Java Remote Method Protocol to allow invocation of functions that operate remotely, but appear local to the developer.[35][36]Therefore, remote APIs are useful in maintaining the object abstraction in object-oriented programming; a method call, executed locally on a proxy object, invokes the corresponding method on the remote object, using the remoting protocol, and acquires the result to be used locally as a return value.A modification of the proxy object will also result in a corresponding modification of the remote object.[37]Main article: Web APIWeb APIs are the defined interfaces through which interactions happen between an enterprise and applications that use its assets, which also is a Service Level Agreement (SLA) to specify the functional provider and expose the service path or URL for its API users. An API approach is an architectural approach that revolves around providing a program interface to a set of services to different applications serving different types of consumers.[38]When used in the context of web development, an API is typically defined as a set of specifications, such as Hypertext Transfer Protocol (HTTP) request messages, along with a definition of the structure of response messages, usually in an Extensible Markup Language (XML) or JavaScript Object Notation (JSON) format. An example might be a shipping company API that can be added to an eCommerce-focused website to facilitate ordering shipping services and automatically include current shipping rates, without the site developer having to enter the shipper's rate table into a web database. While "web API" historically has been virtually synonymous with web service, the recent trend (so-called Web 2.0) has been moving away from Simple Object Access Protocol (SOAP) based web services and service-oriented architecture (SOA) towards more direct representational state transfer (REST) style web resources and resource-oriented architecture (ROA).[39] Part of this trend is related to the Semantic Web movement toward Resource Description Framework (RDF), a concept to promote web-based ontology engineering technologies. Web APIs allow the combination of multiple APIs into new applications known as mashups.[40]In the social media space, web APIs have allowed web communities to facilitate sharing content and data between communities and applications. In this way, content that is created in one place dynamically can be posted and updated to multiple locations on the web.[41] For example, Twitter's REST API allows developers to access core Twitter data and the Search API provides methods for developers to interact with Twitter Search and trends data.[42]The design of an API has significant impact on its usage.[5] The principle of information hiding describes the role of programming interfaces as enabling modular programming by hiding the implementation details of the modules so that users of modules need not understand the complexities inside the modules.[43] Thus, the design of an API attempts to provide only the tools a user would expect.[5] The design of programming interfaces represents an important part of software architecture, the organization of a complex piece of software.[44]APIs are one of the more common ways technology companies integrate. Those that provide and use APIs are considered as being members of a business ecosystem.[45]The main policies for releasing an API are:[46]Private: The API is for internal company use only.Partner: Only specific business partners can use the API. For example, vehicle for hire companies such as Uber and Lyft allow approved third-party developers to directly order rides from within their apps. This allows the companies to exercise quality control by curating which apps have access to the API, and provides them with an additional revenue stream.[47]Public: The API is available for use by the public. For example, Microsoft makes the Windows API public, and Apple releases its API Cocoa, so that software can be written for their platforms. Not all public APIs are generally accessible by everybody. For example, Internet service providers like Cloudflare or Voxility, use RESTful APIs to allow customers and resellers access to their infrastructure information, DDoS stats, network performance or dashboard controls.[48] Access to such APIs is granted either by API tokens, or customer status validations.[49]An important factor when an API becomes public is its "interface stability". Changes to the APIfor example adding new parameters to a function callcould break compatibility with the clients that depend on that API.[50]When parts of a publicly presented API are subject to change and thus not stable, such parts of a particular API should be documented explicitly as "unstable". For example, in the Google Guava library, the parts that are considered unstable, and that might change soon, are marked with the Java annotation @Beta.[51]A public API can sometimes declare parts of itself as deprecated or rescinded. This usually means that part of the API should be considered a candidate for being removed, or modified in a backward incompatible way. Therefore, these changes allow developers to transition away from parts of the API that will be removed or not supported in the future.[52]Client code may contain innovative or opportunistic usages that were not intended by the API designers. In other words, for a library with a significant user base, when an element becomes part of the public API, it may be used in diverse ways.[53]On February 19, 2020, Akamai published their annual State of the Internet report, showcasing the growing trend of cybercriminals targeting public API platforms at financial services worldwide. From December 2017 through November 2019, Akamai witnessed 85.42 billion credential violation attacks. About 20%, or 16.55 billion, were against hostnames defined as API endpoints. Of these, 473.5 million have targeted financial services sector organizations.[54]API documentation describes what services an API offers and how to use those services, aiming to cover everything a client would need to know for practical purposes.Documentation is crucial for the development and maintenance of applications using the API.[55]API documentation is traditionally found in documentation files but can also be found in social media such as blogs, forums, and Q&A websites.[56]Traditional documentation files are often presented via a documentation system, such as Javadoc or Pydoc, that has a consistent appearance and structure.However, the types of content included in the documentation differs from API to API.[57]In the interest of clarity, API documentation may include a description of classes and methods in the API as well as "typical usage scenarios, code snippets, design rationales, performance discussions, and contracts", but implementation details of the API services themselves are usually omitted. It can take a number of forms, including instructional documents, tutorials, and reference works. It'll also include a variety of information types, including guides and functionalities.Restrictions and limitations on how the API can be used are also covered by the documentation. For instance, documentation for an API function could note that its parameters cannot be null, that the function itself is not thread safe.[58] Because API documentation tends to be comprehensive, it is a challenge for writers to keep the documentation updated and for users to read it carefully, potentially yielding bugs.[50]API documentation can be enriched with metadata information like Java annotations. This metadata can be used by the compiler, tools, and by the run-time environment to implement custom behaviors or custom handling.[59]It is possible to generate API documentation in a data-driven manner. By observing many programs that use a given API, it is possible to infer the typical usages, as well the required contracts and directives.[60] Then, templates can be used to generate natural language from the mined data.Main article: Google LLC v. Oracle America, Inc.In 2010, Oracle Corporation sued Google for having distributed a new implementation of Java embedded in the Android operating system.[61] Google had not acquired any permission to reproduce the Java API, although permission had been given to the similar OpenJDK project. Judge William Alsup ruled in the Oracle v. Google case that APIs cannot be copyrighted in the U.S. and that a victory for Oracle would have widely expanded copyright protection to a "functional set of symbols" and allowed the copyrighting of simple software commands:To accept Oracle's claim would be to allow anyone to copyright one version of code to carry out a system of commands and thereby bar all others from writing its different versions to carry out all or part of the same commands.[62][63]Alsup's ruling was overturned in 2014 on appeal to the Court of Appeals for the Federal Circuit, though the question of whether such use of APIs constitutes fair use was left unresolved.[64][65]In 2016, following a two-week trial, a jury determined that Google's reimplementation of the Java API constituted fair use, but Oracle vowed to appeal the decision.[66] Oracle won on its appeal, with the Court of Appeals for the Federal Circuit ruling that Google's use of the APIs did not qualify for fair use.[67] In 2019, Google appealed to the Supreme Court of the United States over both the copyrightability and fair use rulings, and the Supreme Court granted review.[68] Due to the COVID-19 pandemic, the oral hearings in the case were delayed until October 2020.[69]The case was decided by the Supreme Court in Google's favor.[70]Main category: Application programming interfacesASPI for SCSI device interfacingCocoa and Carbon for the MacintoshDirectX for Microsoft WindowsEHLLAPIJava APIsODBC for Microsoft WindowsOpenAL cross-platform sound APIOpenCL cross-platform API for general-purpose computing for CPUs & GPUsOpenGL cross-platform graphics APIOpenMP API that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran on many architectures, including Unix and Microsoft Windows platforms.Server application programming interface (SAPI)Simple DirectMedia Layer (SDL)API testingAPI writerAugmented webCalling conventionCommon Object Request Broker Architecture (CORBA)Comparison of application virtual machinesDocument Object Model (DOM)Double-chance functionForeign function interfaceFront and back endsInterface (computing)Interface control documentList of 3D graphics APIsMicroservicesName manglingOpen APIOpen Service Interface DefinitionsParsingPluginRAML (software)Software development kit (SDK)Web APIWeb content vendorXPCOM^ a b Reddy, Martin (2011). API Design for C++. Elsevier Science. p.1. ISBN9780123850041.^ a b Lane, Kin (October 10, 2019). "Intro to APIs: History of APIs". Postman. Retrieved September 18, 2020. When you hear the acronym "API" or its expanded version "Application Programming Interface," it is almost always in reference to our modern approach, in that we use HTTP to provide access to machine readable data in a JSON or XML format, often simply referred to as "web APIs." APIs have been around almost as long as computing, but modern web APIs began taking shape in the early 2000s.^ a b Pedro, Bruno (2024). Building an API Product: Design, Implement, Release, and Maintain API Products that Meet User Needs. Packt Publishing. p.4. ISBN9781837638536.^ Biehl, Matthias (2016). RESTful API Design. API-University Press. p.10. ISBN9781514735169.^ a b c Clarke, Steven (2004). "Measuring API Usability". Dr. Dobb's. Retrieved 29 July 2016.^ Jin, Brenda; Sahni, Saurabh; Shevat, Amir (2018). "Preface". Designing Web APIs: Building APIs That Developers Love. O'Reilly Media. ISBN9781492026877.^ Geewax, JJ (2021). API Design Patterns. Manning. p.6. ISBN9781638350330.^ Jacobson, Daniel; Brail, Greg; Woods, Dan (2011). APIs: A Strategy Guide. O'Reilly Media. p.4. ISBN9781449321642.^ a b Database architectures a feasibility workshop (Report). Washington, DC: U.S. Department of Commerce, National Bureau of Standards. April 1981. pp.4547. hdl:2027/mdp.39015077587742. LCCN81600004. NBS special publication 500-76. Retrieved September 18, 2020.^ a b c d Bloch, Joshua (August 8, 2018). A Brief, Opinionated History of the API (Speech). QCon. San Francisco: InfoQ. Retrieved September 18, 2020.^ a b Cotton, Ira W.; Greatorex, Frank S. (December 1968). "Data structures and techniques for remote computer graphics". AFIPS '68: Proceedings of the December 911, 1968, Fall Joint Computer Conference. AFIPS 1968 Fall Joint Computer Conference. Vol.I. San Francisco, California: Association for Computing Machinery. pp.533544. doi:10.1145/1476589.1476661. ISBN978-1450378994. OCLC1175621908.^ "application program interface". Oxford English Dictionary (Onlineed.). Oxford University Press. (Subscription or participating institution membership required.)^ Date, C. J. (2019). E. F. Codd and Relational Theory: A Detailed Review and Analysis of Codd's Major Database Writings. Lulu.com. p.135. ISBN978-1684705276.^ Date, C. J.; Codd, E. F. (January 1975). "The relational and network approaches: Comparison of the application programming interfaces". In Randall Rustin (ed.). Proceedings of 1974 ACM-SIGMOD Workshop on Data Description, Access and Control. SIGMOD Workshop 1974. Vol.2. Ann Arbor, Michigan: Association for Computing Machinery. pp.83113. doi:10.1145/800297.811532. ISBN978-1450374187. OCLC1175623233.^ Carl, Malamud (1990). Analyzing Novell Networks. Van Nostrand Reinhold. p.294. ISBN978-0442003647.^ a b Jin, Brenda; Sahni, Saurabh; Shevat, Amir (2018). Designing Web APIs. O'Reilly Media. ISBN9781492026877.^ Fielding, Roy (2000). Architectural Styles and the Design of Network-based Software Architectures (PhD). Retrieved September 18, 2020.^ Dotsika, Fefie (August 2010). "Semantic APIs: Scaling up towards the Semantic Web". International Journal of Information Management. 30 (4): 335342. doi:10.1016/j.ijinfomgt.2009.12.003.^ Odersky, Martin; Spoon, Lex; Venners, Bill (10 December 2008). "Combining Scala and Java". www.artima.com. Retrieved 29 July 2016.^ de Figueiredo, Luiz Henrique; Ierusalimschy, Roberto; Filho, Waldemar Celes (1994). "The design and implementation of a language for extending applications". Proceedings of XXI Brazilian Seminar on Software and Hardware. pp.273284. CiteSeerX10.1.1.47.5194. S2CID59833827. Retrieved 29 July 2016.^ Sintes, Tony (13 July 2001). "Just what is the Java API anyway?". JavaWorld. Retrieved 2020-07-18.^ Winters, Titus; Tom Manshreck; Hyrum Wright, eds. (2020). Software engineering at Google: lessons learned from programming over time. Sebastopol, CA: O'Reilly Media. ISBN9781492082798. OCLC1144086840.^ Mastrangelo, Luis; Ponzanelli, Luca; Mocci, Andrea; Lanza, Michele; Hauswirth, Matthias; Nystrom, Nathaniel (2015-10-23). "Use at your own risk: the Java unsafe API in the wild". Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications. New York, New York, U.S.: Association for Computing Machinery. pp.695710. doi:10.1145/2814270.2814313. ISBN978-1-4503-3689-5.^ Emery, David. "Standards, APIs, Interfaces and Bindings". Acm.org. Archived from the original on 2015-01-16. Retrieved 2016-08-08.^ "F2PY.org". F2PY.org. Retrieved 2011-12-18.^ Fowler, Martin. "Inversion Of Control".^ Fayad, Mohamed. "Object-Oriented Application Frameworks".^ Lewine, Donald A. (1991). POSIX Programmer's Guide. O'Reilly & Associates, Inc. p.1. ISBN9780937175736. Retrieved 2 August 2016.^ West, Joel; Dedrick, Jason (2001). "Open source standardization: the rise of Linux in the network era" (PDF). Knowledge, Technology & Policy. 14 (2): 88112. doi:10.1007/PL00022278. Retrieved 2 August 2016.^ Microsoft (October 2001). "Support for Windows XP". Microsoft. p.4. Archived from the original on 2009-09-26.^ Barney, Douglas (1987-11-02). "Balancing on the high wire of Microsoft's success". Computerworld. Vol.XXI, no.44. p.SR15. Retrieved 2025-06-08.^ Gates, Bill; Manzi, Jim; Esber, Ed (1987-11-02). "The great software debate". Computerworld (Interview). Vol.XXI, no.44. Interviewed by Paul Gillin. p.SR7. Retrieved 2025-06-08.^ "LSB Introduction". Linux Foundation. 21 June 2012. Archived from the original on 2015-04-02. Retrieved 2015-03-27.^ Stoughton, Nick (April 2005). "Update on Standards" (PDF). USENIX. Retrieved 2009-06-04.^ Bierhoff, Kevin (23 April 2009). "API Protocol Compliance in Object-Oriented Software" (PDF). CMU Institute for Software Research. Retrieved 29 July 2016.^ Wilson, M. Jeff (10 November 2000). "Get smart with proxies and RMI". JavaWorld. Retrieved 2020-07-18.^ Henning, Michi; Vinoski, Steve (1999). Advanced CORBA Programming with C++. Addison-Wesley. ISBN978-0201379273. Retrieved 16 June 2015.^ "API-fication" (PDF). www.hcltech.com. August 2014.^ Benslimane, Djamal; Schahram Dustdar; Amit Sheth (2008). "Services Mashups: The New Generation of Web Applications". IEEE Internet Computing. 12 (5). IEEE: 1315. doi:10.1109/MIC.2008.110. Retrieved 2019-10-01.^ Niccolai, James (2008-04-23), "So What Is an Enterprise Mashup, Anyway?", PC World, archived from the original on 2017-10-10, retrieved 2017-09-17^ Parr, Ben (21 May 2009). "The Evolution of the Social Media API". Mashable. Retrieved 26 July 2016.^ "GET trends/place". developer.twitter.com. Retrieved 2020-04-30.^ Parnas, D.L. (1972). "On the Criteria To Be Used in Decomposing Systems into Modules" (PDF). Communications of the ACM. 15 (12): 10531058. doi:10.1145/361598.361623. S2CID53856438.^ Garlan, David; Shaw, Mary (January 1994). "An Introduction to Software Architecture" (PDF). Advances in Software Engineering and Knowledge Engineering. 1. Retrieved 8 August 2016.^ de Ternay, Guerric (Oct 10, 2015). "Business Ecosystem: Creating an Economic Moat". BoostCompanies. Archived from the original on 2016-09-17. Retrieved 2016-02-01.^ Boyd, Mark (2014-02-21). "Private, Partner or Public: Which API Strategy Is Best for Business?". ProgrammableWeb. Retrieved 2 August 2016.^ Weissbrot, Alison (7 July 2016). "Car Service APIs Are Everywhere, But What's In It For Partner Apps?". AdExchanger.^ "Cloudflare API v4 Documentation". cloudflare. 25 February 2020. Retrieved 27 February 2020.^ Liew, Zell (17 January 2018). "Car Service APIs Are Everywhere, But What's In It For Partner Apps". Smashing Magazine. Retrieved 27 February 2020.^ a b Shi, Lin; Zhong, Hao; Xie, Tao; Li, Mingshu (2011). An Empirical Study on Evolution of API Documentation. International Conference on Fundamental Approaches to Software Engineering. Lecture Notes in Computer Science. Vol.6603. pp.416431. doi:10.1007/978-3-642-19811-3_29. ISBN978-3-642-19810-6. Retrieved 22 July 2016.^ google/guava: Google Core Libraries for Java on GitHub^ Oracle. "How and When to Deprecate APIs". Java SE Documentation. Retrieved 2 August 2016.^ Mendez, Diego; Baudry, Benoit; Monperrus, Martin (2013). Empirical evidence of large-scale diversity in API usage of object-oriented software. 2013 IEEE 13th International Working Conference on Source Code Analysis and Manipulation (SCAM). pp.4352. arXiv:1307.4062. doi:10.1109/SCAM.2013.6648183. ISBN978-1-4673-5739-5. S2CID6890739.^ Takanashi, Dean (19 February 2020). "Akamai: Cybercriminals are attacking APIs at financial services firms". Venture Beat. Retrieved 27 February 2020.^ Dekel, Uri; Herbsleb, James D. (May 2009). "Improving API Documentation Usability with Knowledge Pushing". Institute for Software Research, School of Computer Science. CiteSeerX10.1.1.446.4214.^ Parnin, Chris; Treude, Cristoph (May 2011). "Measuring API documentation on the web". Proceedings of the 2nd International Workshop on Web 2.0 for Software Engineering. pp.2530. doi:10.1145/1984701.1984706. ISBN9781450305952. S2CID17751901. Retrieved 22 July 2016.^ Maalej, Waleed; Robillard, Martin P. (September 2012). "Patterns of Knowledge in API Reference Documentation" (PDF). IEEE Transactions on Software Engineering. 39 (9): 12641282. doi:10.1109/TSE.2013.12. Retrieved 22 July 2016.^ Monperrus, Martin; Eichberg, Michael; Tekes, Elif; Mezini, Mira (3 December 2011). "What should developers be aware of? An empirical study on the directives of API documentation". Empirical Software Engineering. 17 (6): 703737. arXiv:1205.6363. doi:10.1007/s10664-011-9186-4. S2CID8174618.^ "Annotations". Sun Microsystems. Archived from the original on 2011-09-25. Retrieved 2011-09-30..^ Bruch, Marcel; Mezini, Mira; Monperrus, Martin (2010). Mining subclassing directives to improve framework reuse. 2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010). pp.141150. CiteSeerX10.1.1.434.15. doi:10.1109/msr.2010.5463347. ISBN978-1-4244-6802-7. S2CID1026918.^ "Oracle and the End of Programming As We Know It". DrDobbs. 2012-05-01. Retrieved 2012-05-09.^ "APIs Can't be Copyrighted Says Judge in Oracle Case". TGDaily. 2012-06-01. Retrieved 2012-12-06.^ "Oracle America, Inc. vs. Google Inc" (PDF). Wired. 2012-05-31. Retrieved 2013-09-22.^ "Oracle Am., Inc. v. Google Inc., No. 13-1021, Fed. Cir. 2014".^ Rosenblatt, Seth (May 9, 2014). "Court sides with Oracle over Android in Java patent appeal". CNET. Retrieved 2014-05-10.^ "Google beats Oracle Android makes "fair use" of Java APIs". Ars Technica. 2016-05-26. Retrieved 2016-07-28.^ Decker, Susan (March 27, 2018). "Oracle Wins Revival of Billion-Dollar Case Against Google". Bloomberg Businessweek. Retrieved March 27, 2018.^ Lee, Timothy (January 25, 2019). "Google asks Supreme Court to overrule disastrous ruling on API copyrights". Ars Technica. Retrieved February 8, 2019.^ vkimber (2020-09-28). "Google LLC v. Oracle America, Inc". LII / Legal Information Institute. Retrieved 2021-03-06.^ "Supreme Court of the United States, No. 18956, GOOGLE LLC, PETITIONER v. ORACLE AMERICA, INC" (PDF). April 5, 2021.Taina Bucher (16 November 2013). "Objects of Intense Feeling: The Case of the Twitter API". Computational Culture (3). ISSN2047-2390. Argues that "APIs are far from neutral tools" and form a key part of contemporary programming, understood as a fundamental part of culture.What is an API? in the U.S. Supreme Court opinion, Google v. Oracle 2021, pp.37 "For each task, there is computer code; API (also known as Application Program Interface) is the method for calling that 'computer code' (instruction like a recipe rather than cooking instruction, this is machine instruction) to be carry out"Maury, Innovation and Change Cory Ondrejka \ February 28, 2014 \ " ...proposed a public API to let computers talk to each other". (Textise URL)Forrester: IT industry: API Case: Google v. Oracle May 20, 2021 content format: Audio with text length 26:41Retrieved from " In the tech world, APIs (Application Programming Interfaces) are crucial. If you're interested in becoming a web developer or want to understand how websites work, you'll need to famil