[OSAKamp] ASOCIACIÓN DE EMPRESAS DE TECNOLOGÍAS LIBRES Y CONOCIMIENTO ABIERTO DE EUSKADI

04 - APIs

Teoria

CODE4JOBS

Bizkaia
foru aldundia
diputación foral

[esle]

¿Qué es una API?

El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.

Así pues, podemos hablar de una API como una especificación formal que establece cómo un módulo de un software se comunica o interactúa con otro para cumplir una o muchas funciones. Todo dependiendo de las aplicaciones que las vayan a utilizar, y de los permisos que les dé el propietario de la API a los desarrolladores de terceros.

No son la parte visible, sino los circuitos internos que sólo los desarrolladores ven y conectan para hacer funcionar una herramienta. De cara a un usuario normal, lo único que vas a ver de una API son los resultados, cómo abres un juego para el móvil y puedes conectarte a tu cuenta de Facebook para iniciar sesión, o cómo puedes publicar los resultados de una partida en Twitter. O cuando esa aplicación te manda notificaciones al móvil o al ordenador.

Como hemos dicho antes, las API pueden tener tanto una como varias funciones, pudiendo llegar a ser auténticos kits de herramientas. Cuando esto pasa, tu aplicación puede enviarle una solicitud con una estructura particular, y esta estructura determinará cómo responderá el servicio o el software al que le estés enviando esa solicitud.

Pueden ser privadas para el uso de una empresa, abiertas sólo para partners, o públicas para que cualquier desarrollador interactuar con ellas o crear sus propias API para que lo hagan. También pueden ser API locales para aplicaciones que se comunican dentro de un mismo ambiente o dispositivo, o remotas para cuando hay que acceder a otro punto diferente.

¿Para qué sirve una API?

Una de las principales funciones de las API es poder facilitarles el trabajo a los desarrolladores y ahorrarles tiempo y dinero. Por ejemplo, si estás creando una aplicación que es una tienda online, no necesitarás crear desde cero un sistema de pagos u otro para verificar si hay stock disponible de un producto. Podrás utilizar la API de un servicio de pago ya existente, por ejemplo PayPal, y pedirle a tu distribuidor una API que te permita saber el stock que ellos tienen.

Con ello, no será necesario tener que reinventar la rueda con cada servicio que se crea, ya que podrás utilizar piezas o funciones que otros ya han creado. Imagínate que cada tienda online tuviera que tener su propio sistema de pago, para los usuarios normales es mucho más cómodo poder hacerlo con los principales servicios que casi todos utilizan.

Otro ejemplo clásico es el de las aplicaciones de terceros para conectarse a un servicio. Por ejemplo, redes sociales como Twitter o Reddit permiten que se creen aplicaciones diferentes a las oficiales para conectarse a ellas. Estas aplicaciones necesitarán las API de las redes sociales para poder mostrarte la información y los mecanismos internos que las hacen funcionar.

¿Cómo funciona una API?

Una API (las que veremos nosotros) se encarga de pedir datos a una base de datos, y de añadir, modificar o borrar dichos datos. Cuando nosotros hacemos una petición a la API, ésta nos devolverá el resultado que hayamos pedido en formato JSON. JSON hace referencia a JavaScript Object Notation y es un estándar para pasar datos. Veamos por qué necesitamos un JSON.

Cuando se hacen peticiones entre el servidor y el navegador, los datos que viajan de uno a otro y viceversa sólo pueden ser texto. JSON nos ayuda aquí convirtiendo un objeto en texto, con un marcado muy particular, de forma que cuando llegue a su destino pueda volver a convertirse en objeto con una instrucción. La instrucción para pasar un objeto a texto es JSON.stringify(objetoAConvertir) y la instrucción para pasar de texto a objeto es JSON.parse(objetoEnStringAConvertir).

Para hacer esas peticiones de las que hablamos, vamos a utilizar un método llamado **fetch**, el cual es nativo de la API de javascript.

Una petición **fetch** tendrá la siguiente estructura:

```
fetch('http://example.com/movies.json')
  .then(function(response) {
    return response.json();
  })
  .then(function(res) {
    console.log(res);
  });
```

En esta parte del bootcamp sólo haremos peticiones que pidan datos a una base de datos, por lo que no entraremos mucho en hablar acerca de los diferentes métodos que existen y nos centraremos en el GET. Las peticiones de tipo GET sirven para pedir datos ya existentes en la base de datos. A veces tendremos que pasar un parámetro en la url, y otras veces haremos una petición a una url genérica, la cual no necesita de un parámetro. Dicha url va entre los paréntesis del método fetch. Tras eso vemos que la función continúa con dos **then**. Los **then** hacen el trabajo de esperar a que la petición se resuelva porque las peticiones no son instantáneas, son lo que se conoce como **asíncronas**. El primer **then** espera a que la respuesta llegue desde la API hasta el lugar donde estamos haciendo la petición (el front casi siempre), y el segundo **then** utilizará los datos tratados en ese primer **then**.

Con eso ya estaríamos haciendo peticiones a una API y pudiendo acceder a los datos que esa petición a la API nos devuelve.