# **API de JavaScript**

## **Introducción**

En esta sección, explorará las API de JavaScript y el estilo arquitectónico REST.

**Tiempo estimado:** 15 minutos

## **Objetivos**

Después de completar esta sección de lectura, usted podrá:

1. Definir una API y una API de JavaScript
2. Describir la arquitectura REST
3. Explicar las operaciones CRUD
4. Identificar las API de JavaScript más populares

## **¿Qué son las API de JavaScript?**

Para entender qué es una API de JavaScript, es importante saber primero qué es una API. Una API, o interfaz de programación de aplicaciones, es una forma en la que dos aplicaciones se comunican entre sí. Entrega su solicitud a otro dispositivo, como una base de datos, y le devuelve la respuesta.

Imagina que estás sentado en un restaurante y has seleccionado tu pedido. El menú describe una lista de platos y las comidas correspondientes se preparan en la cocina. Tu camarero es el vínculo entre tú y la cocina, quien comunica tu pedido a la cocina y te devuelve la comida. Esta comunicación es similar a cómo funcionan las API. Una API es análoga a un camarero, ya que comunica una solicitud de un dispositivo a otro y devuelve la respuesta al primer dispositivo.

El menú del ejemplo es la documentación de la API. En un restaurante, si pides un artículo de comida que no existe en el menú, el camarero te informará que es una opción no válida y no podrá entregarte el artículo. De manera similar, cada API tiene documentación que describe las solicitudes que puedes hacer y el tipo de respuesta que deberías esperar recibir. Si intentas hacer una solicitud no válida, te aparecerá un error.

Las API de JavaScript son interfaces de programación de aplicaciones que utilizan scripts de JavaScript para acceder y modificar contenido de forma dinámica.

## **Arquitectura REST**

La mayoría de las API de JavaScript siguen el estilo arquitectónico de transferencia de estado representacional (REST). Se las conoce como API RESTful y siguen el paradigma CRUD. CRUD significa crear, leer, actualizar y eliminar, y modela las cuatro funcionalidades básicas necesarias para la comunicación entre servicios y con una base de datos. En un entorno REST, estas operaciones CRUD suelen tener los siguientes alias:

* Crear → PUBLICAR
* Leer → OBTENER
* Actualizar → PONER
* Eliminar → ELIMINAR

A modo de ejemplo, imaginemos una API que se comunica con un servicio bancario para procesar pagos en línea. Es posible utilizar las cuatro operaciones CRUD para esta API. A continuación, se ofrece un ejemplo de cada una de ellas.

|  |  |  |
| --- | --- | --- |
| **Método** | **URL** | **Descripción** |
| CORREO | api/cliente | Crear un nuevo cliente bancario |
| CONSEGUIR | api/clientes/{id} | Recuperar la información de un cliente |
| PONER | api/clientes/{id} | Actualizar información de un cliente específico |
| BORRAR | api/clientes/{id} | Eliminar un cliente bancario |

Las URL en cada ejemplo son importantes, ya que determinan el artículo/cliente específico al que se accede.

En la solicitud POST, puede ver que se crea un nuevo cliente con la API. Según las especificaciones de la API, esto puede incluir opciones para proporcionar datos de este cliente, como un nombre o detalles de la tarjeta de crédito. También puede generar automáticamente nueva información al momento de la creación, como un ID.

La solicitud GET le permite recuperar toda la información asociada con un cliente. La API supone un ID único para cada cliente, que se utiliza en la URL para especificar la información de qué cliente está buscando. La respuesta a esta solicitud puede venir en muchos formatos diferentes, como JSON o XML, según la API.

En la solicitud PUT, puede actualizar la información de un cliente específico. Esto sobrescribirá los datos actuales con datos nuevos. De manera similar a la solicitud GET, puede especificar un cliente específico mediante el ID. Puede proporcionar datos nuevos para actualizar de diferentes maneras que son específicas de la API. Algunas API le permiten incluir un "cuerpo" en el que puede especificar una carga de datos que se enviará con la solicitud. Por ejemplo, puede adjuntar el siguiente cuerpo a la solicitud PUT para actualizar el nombre y apellido de un cliente:

1

2

3

4

1. {
2. "first\_name": "Thomas",
3. "last\_name": "Watson"
4. }

Copiado!

En la solicitud DELETE dada en el ejemplo, puede eliminar un cliente por completo proporcionando nuevamente su identificación.

## **API del modelo de objetos de documento (DOM)**

La API del modelo de objetos de documento (DOM) es una de las API de JavaScript más básicas que existen. Conecta páginas web con scripts al representar la estructura de un documento (por ejemplo, una página web HTML) en la memoria, lo que lo hace accesible para realizar modificaciones según sea necesario.

La API DOM se cubre con más detalle en los videos y no se revisará aquí.

## **Solicitud XMLHttp**

Una API de JavaScript popular es XMLHttpRequest (XHR), que permite recuperar datos sin actualizar toda la página. Esto es importante cuando se desea actualizar solo una parte de una página sin interrumpir lo que el usuario está haciendo en ese momento en la página.

XMLHttpRequest se utiliza mucho en la programación asincrónica en JavaScript y XML (AJAX). Puede encontrar documentación completa sobre su uso en (esta página)  
<https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest>

## **API avanzadas**

Hay API de JavaScript más avanzadas disponibles, cada una con un uso y una especificación diferentes. Muchas de estas API se pueden encontrar en el [sitio web oficial para desarrolladores de Mozilla](https://developer.mozilla.org/en-US/docs/Web/API) o puedes buscar en Internet la API que necesitas.