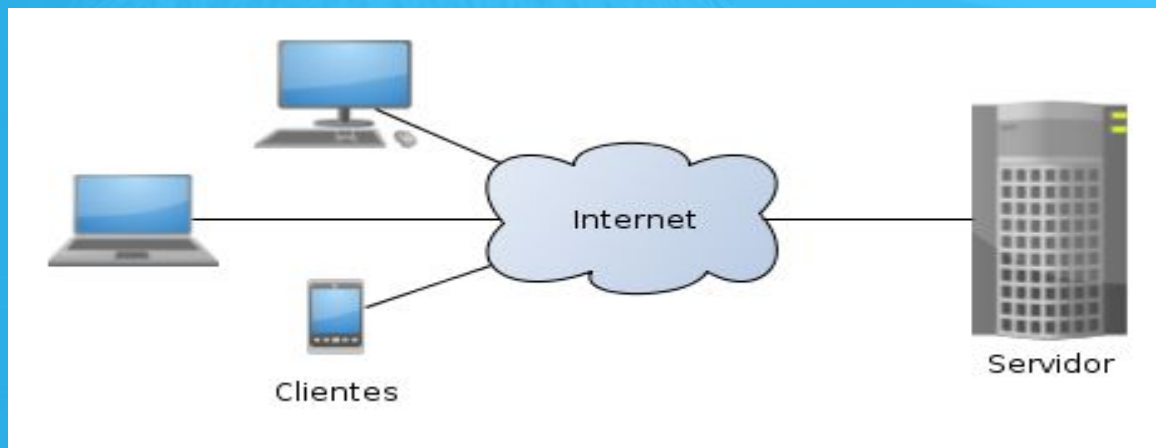




BackEnd

¿Que es el Back End?

- ❑ Contiene la lógica, las reglas de negocio, de la aplicación.
- ❑ Es la capa de acceso a datos de una aplicación o cualquier dispositivo
- ❑ No es directamente accesible por los usuarios.





Microservicios

La arquitectura de microservicios (MSA) es una aproximación para el desarrollo de software que consiste en:

- ❑ Construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros (normalmente una API de recursos HTTP).
- ❑ Cada servicio se encarga de implementar una funcionalidad completa del negocio.
- ❑ Cada servicio es desplegado de forma independiente y puede estar programado en distintos lenguajes y usar diferentes tecnologías de almacenamiento de datos.

Se suele considerar la arquitectura de microservicios como una forma específica de realizar una arquitectura orientada a servicios.

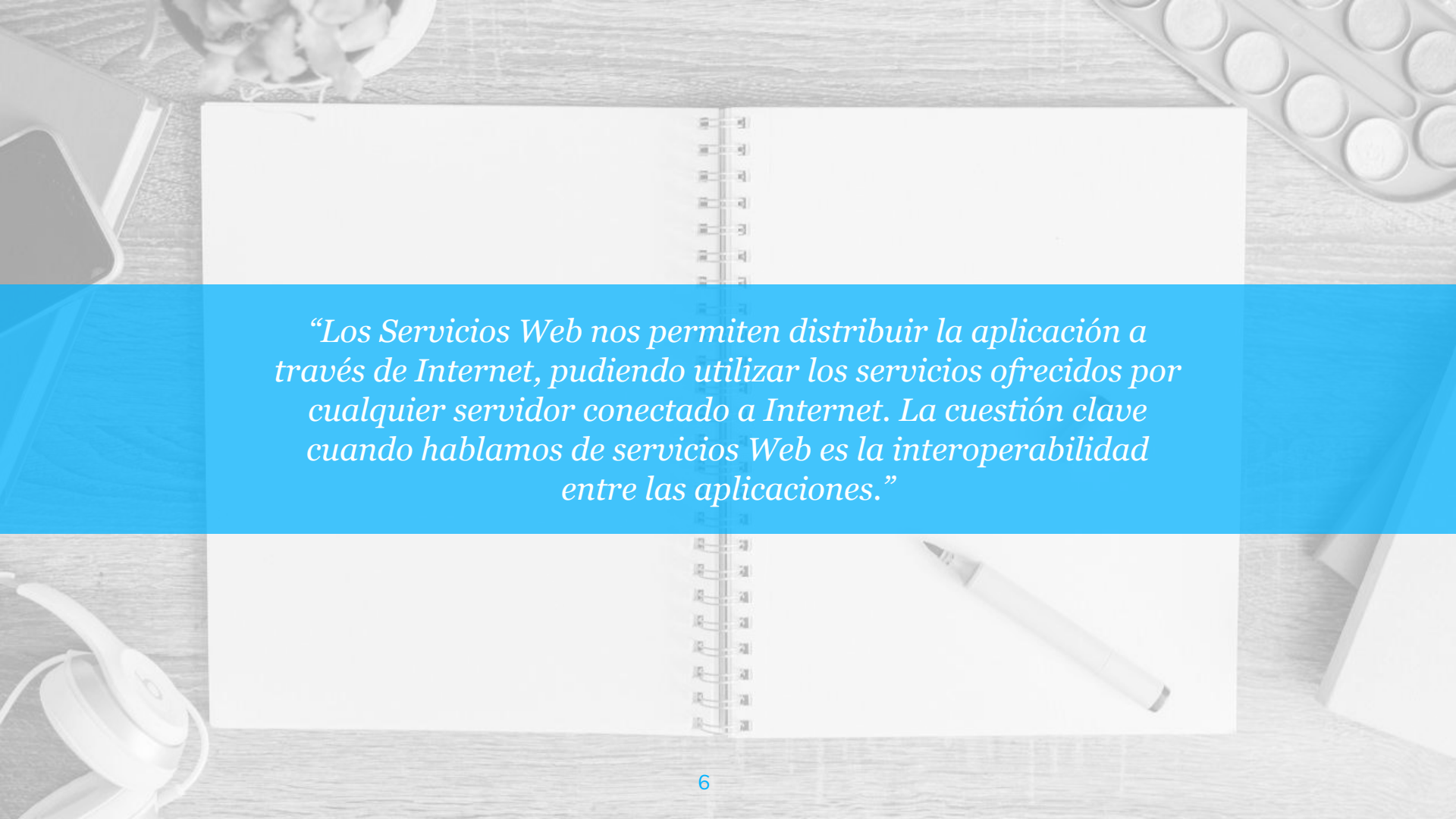
Memo de Jeff Bezos a sus empleados (+-2002)

- ❑ All teams will henceforth expose their data and functionality through service interfaces.
- ❑ Teams must communicate with each other through these interfaces.
- ❑ There will be no other form of inter-process communication allowed: no direct linking, no direct reads of another team's data store, no shared-memory model, no back-doors whatsoever. The only communication allowed is via service interface calls over the network.
- ❑ It doesn't matter what technology they use.
- ❑ All service interfaces, without exception, must be designed from the ground up to be externalizable. That is to say, the team must plan and design to be able to expose the interface to developers in the outside world. No exceptions.

Anyone who doesn't do this will be fired. Thank you; have a nice day!

Hello!

Web Services



“Los Servicios Web nos permiten distribuir la aplicación a través de Internet, pudiendo utilizar los servicios ofrecidos por cualquier servidor conectado a Internet. La cuestión clave cuando hablamos de servicios Web es la interoperabilidad entre las aplicaciones.”

¿Que es un servicio web?

Un Servicio Web es un componente al que podemos acceder mediante protocolos Web estándar. En sus orígenes, utilizando XML para el intercambio de información.

Normalmente nos referimos a una colección de procedimientos (métodos) a los que podemos llamar desde cualquier lugar de Internet o de nuestra intranet, siendo este mecanismo de invocación totalmente independiente de la plataforma que utilizemos y del lenguaje de programación en el que se haya implementado internamente el servicio.

Historia y Estándares

Historia de los servicios Web

Los servicios Web fueron "inventados" para solucionar el problema de la **interoperabilidad** entre las aplicaciones. Al principio de los 90, con el desarrollo de Internet/LAN/WAN, apareció el gran problema de integrar aplicaciones diferentes. Una aplicación podía haber sido desarrollada en C++ o Java, y ejecutarse bajo Unix, un PC, o un computador *mainframe*. No había una forma fácil de intercomunicar dichas aplicaciones. Fué el desarrollo de XML el que hizo posible compartir datos entre aplicaciones con diferentes plataformas hardware a través de la red, o incluso a través de Internet. La razón de que se llamasen servicios Web es que fueron diseñados para residir en un servidor Web, y ser llamados a través de Internet, típicamente via protocolos HTTP, o HTTPS. De esta forma se asegura que un servicio puede ser llamado por cualquier aplicación, usando cualquier lenguaje de programación, y bajo cualquier sistema operativo, siempre y cuándo, por supuesto, la conexión a Internet esté activa, y tenga un puerto abierto HTTP/HTTPS, lo cual es cierto para casi cualquier computador que disponga de acceso a Internet.

Los servicios Web son componentes de aplicaciones distribuidas que están disponibles de forma externa. Se pueden utilizar para integrar aplicaciones escritas en diferentes lenguajes y que se ejecutan en plataformas diferentes. Los servicios Web son independientes de lenguaje y de la plataforma gracias a que los vendedores han admitido estándares comunes de Servicios Web.

El WC3 (World Wide Web Consortium) define un servicio Web como un sistema software diseñado para soportar interacciones máquina a máquina a través de la red.



Características de un WebServices

- ❑ Un servicio debe poder ser accesible a través de la Web. Para ello debe utilizar protocolos de transporte estándares como HTTP, y codificar los mensajes en un lenguaje estándar que pueda conocer cualquier cliente que quiera utilizar el servicio.
- ❑ Un servicio debe contener una descripción de sí mismo. De esta forma, una aplicación podrá saber cuál es la función de un determinado Servicio Web, y cuál es su interfaz, de manera que pueda ser utilizado de forma automática por cualquier aplicación, sin la intervención del usuario.
- ❑ Debe poder ser localizado. Deberemos tener algún mecanismo que nos permita encontrar un Servicio Web que realice una determinada función. De esta forma tendremos la posibilidad de que una aplicación localice el servicio que necesite de forma automática, sin tener que conocerlo previamente el usuario.



ENDPOINT

Los servicios pueden interconectarse a través de la red. En una arquitectura orientada a servicios, cualquier interacción punto a punto implica dos endpoints:

- uno que proporciona un servicio,
- y otro de lo consume.

Es decir, un endpoint es cada uno de los "elementos", que se sitúan en ambos "extremos" de la red. Cuando hablamos de servicios Web, un endpoint se especifica mediante una URI (Identificador de Recursos Uniforme)



Tipos de servicios web

Web SOAP

Servicios grandes
complejos.

Web RESTful

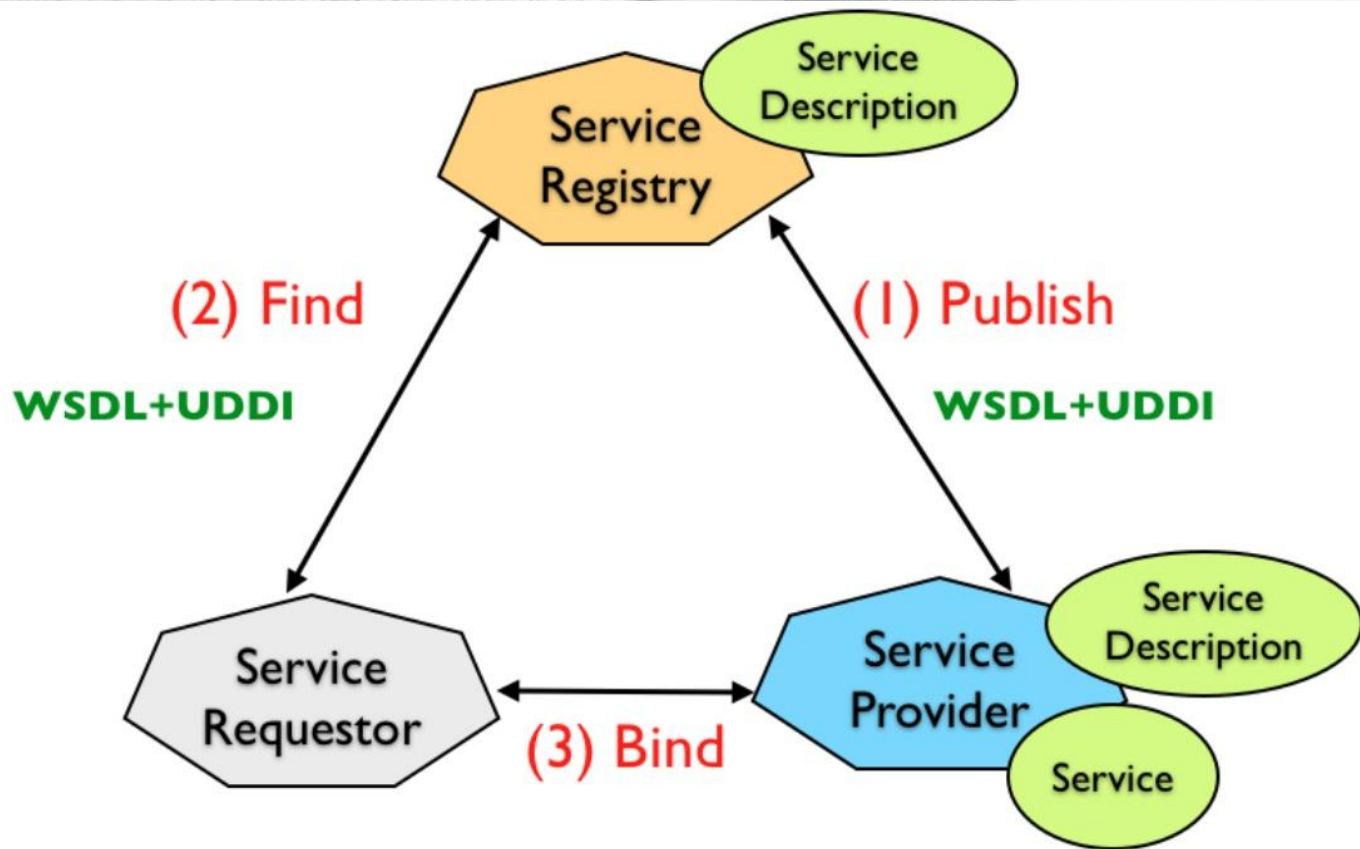
Servicios básicos
ad-hoc.



Servicios Web SOAP

Estos servicios utilizan mensajes **XML** para comunicarse que siguen el estándar **SOAP** (Simple Object Access Protocol), un lenguaje XML que define la arquitectura y formato de los mensajes.

Dichos sistemas normalmente contienen una descripción legible por la máquina de la descripción de las operaciones ofrecidas por el servicio, escrita en **WSDL** (Web Services Description Language), que es un lenguaje basado en XML para definir las interfaces sintácticamente.





Servicios Web RESTful

RESTful (Representational State Transfer Web Services) son adecuados para escenarios de integración básicos.

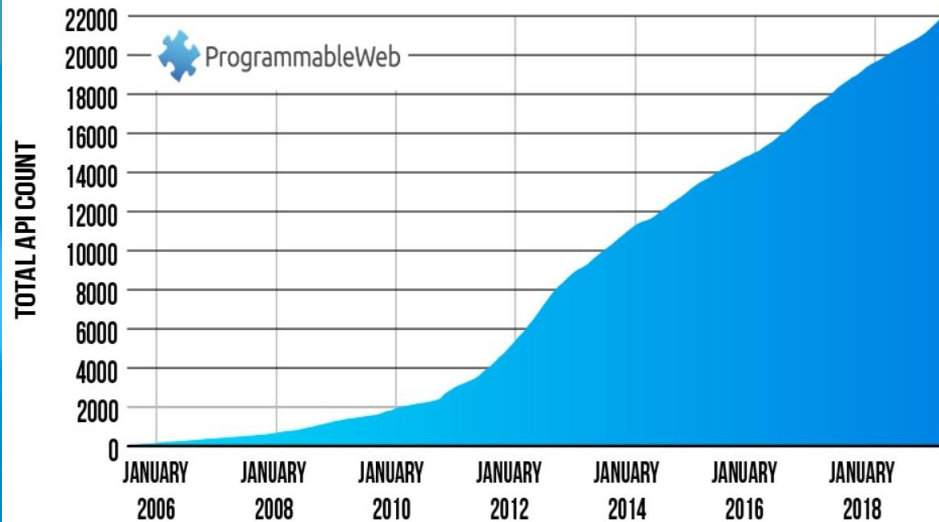
Dichos servicios Web se suelen integrar mejor con HTTP que los servicios basados en SOAP, ya que no requieren mensajes XML o definiciones del servicio en forma de fichero WSDL.

Los servicios Web REST utilizan estándares muy conocidos como HTTP, SML, URI, MIME, y tienen una infraestructura "ligera" que permite que los servicios se construyan utilizando herramientas de forma mínima. Gracias a ello, el desarrollo de servicios RESTful es barato y tiene muy pocas "barreras" para su adopción.



Desarrollo de APIs

GROWTH IN WEB APIS SINCE 2005



A grayscale photograph of a person's hand holding a white coffee cup with a lid. The person is wearing a textured, possibly knitted, garment. The background is blurred, suggesting an indoor setting with other people and objects.

API (Application Programming Interface) Management

- ❑ Especifica cómo deberían interactuar los diferentes componentes de software.
- ❑ Facilita el acceso a componentes hardware o bases de datos.
- ❑ Puede utilizarse para facilitar el trabajo de desarrollo.

En la práctica, a menudo incluyen dentro de sus librerías especificaciones para manejar subrutinas, tipos de datos, clases y variables. En algunos casos, especialmente en servicios web, es únicamente una especificación para que los usuarios remotos puedan consumir los servicios.

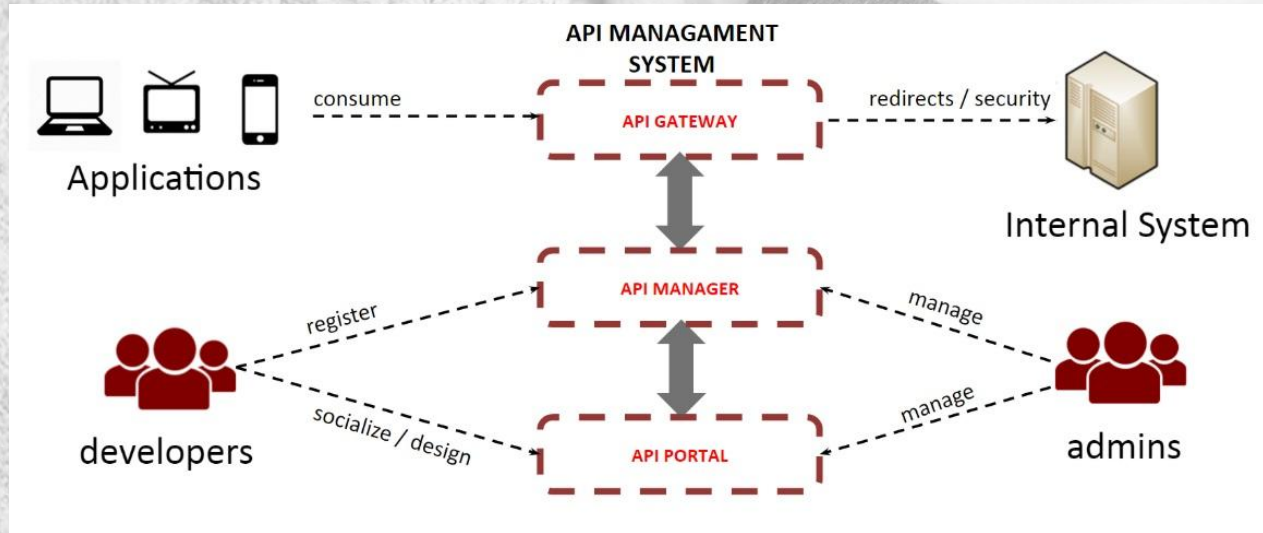


API (Application Programming Interface) Management

En términos generales, hacen posible la interconexión de módulos y aplicaciones, facilitando el acceso a sus backends y permitiendo la reutilización de servicios.

Haciendo una analogía con un ejemplo cotidiano, una API podría ser el enchufe de nuestra casa y el servicio la electricidad que nos proporciona la empresa distribuidora.

Un gran número de organizaciones, está centrando sus esfuerzos en la publicación de APIs. La estrategia empresarial en la gestión de sus APIs debe ser la palanca sobre la que pivotar otro tipo de estrategias IT: Mobility, IoT, Cloud...



- ❑ **API Gateway:** principal función es la de habilitar la interconexión entre los servicios y los consumidores, a través de las APIs publicadas en él.
- ❑ **API Manager:** cuya principal responsabilidad es la de ofrecer a los proveedores capacidades de alta configuración y publicación de sus APIs en el componente API Gateway.
- ❑ **API Portal:** dedicado a recopilar toda la información necesaria para los consumidores sobre las APIs publicadas en el API Gateway.

Elementos

API Gateway

- ❑ Routing
- ❑ Soporte multi-protocolo
- ❑ Soporte multi-formato
- ❑ Monitoring trafico
- ❑ Políticas de seguridad
- ❑ Políticas de uso

API Manager

- ❑ Publicación
- ❑ Edición
- ❑ Gestor del ciclo de vida
- ❑ Gestor de políticas de uso
- ❑ Consumo
- ❑ Gestor de políticas de seguridad

API Portal

- ❑ Comunidad de desarrollo
- ❑ Navegador interno
- ❑ Tienda
- ❑ Probador
- ❑ Documentación.
- ❑ Estadísticas de uso

A top-down view of a wooden desk. In the center is a spiral-bound notebook with blank white pages. A silver pen lies on the bottom right page. To the top right is a paint palette with several wells of paint. To the bottom left is a pair of white headphones. The background is a light-colored wooden surface.

NODE.JS y EXPRESS



¿Que es Node.js?

Es un entorno que trabaja en tiempo de ejecución, de código abierto, multi-plataforma, que permite a los desarrolladores crear toda clase de herramientas del lado del servidor en JavaScript.

La ejecución en tiempo real está pensada para usarse fuera del contexto de un explorador web. Como tal, el entorno omite las APIs de JavaScript específicas del explorador web y añade soporte para APIs de sistema operativo más tradicionales que incluyen HTTP y bibliotecas de sistemas de ficheros.



¿Que es Express?

Express.js es un framework para Node.js que nos proporciona funcionalidades como el enrutamiento, opciones para gestionar sesiones y cookies, y un largo etc.

Express.js está basado en Connect, que a su vez es un framework basado en http para Node.js. Podemos decir que Connect tiene todas las opciones del módulo http que viene por defecto con Node y le suma funcionalidades.



Servidor Web y HTTP

Los browser se comunican con los servidores web usando el Protocolo de Transferencia de HyperTexto (HTTP). Cuando seleccionas un enlace en una página web, envías un formulario o ejecutas una búsqueda, el explorador envía una petición (Request) HTTP al servidor.

Esta petición incluye:

- ❑ Una URL que identifica el servidor de destino y un recurso.
- ❑ Un método que define la acción requerida.
- ❑ Información adicional

Métodos petición HTTP

Los diferentes métodos/verbos y sus acciones asociadas se listan debajo:

- ❑ GET: Obtener un recurso específico .
- ❑ POST: Crear un nuevo recurso .
- ❑ HEAD: Obtener la información de los metadatos sobre un recurso específico sin obtener el cuerpo entero tal como haría GET.
- ❑ PUT: Actualizar un recurso existente (o crear uno nuevo si no existe).
- ❑ DELETE: Borrar el recurso específico.
- ❑ TRACE, OPTIONS, CONNECT, PATCH: Estos verbos son para tareas menos comunes/avanzadas.



Códigos respuesta HTTP

- ❑ Informational responses (100–199),
- ❑ Successful responses (200–299),
- ❑ Redirects (300–399),
- ❑ Client errors (400–499),
- ❑ Server errors (500–599).

Ahora Manos a la Obra

Thanks!

