

Aplicación Web-Webapplication

Definición

A distributed application, a kind of client-server computer program, hosted in a web server and that runs in a web browser.

Características

- *Portables:* No están comprometidas a una arquitectura o sistema operativo
- *Usan estándares:* Es decir que son soportados en una variedad de navegadores, algunos ejemplos son: HTML, CSS y ECMAScript

Algunas aplicaciones móviles son desarrolladas como aplicaciones web (no son nativas) pero, ¿qué opinas de las aplicaciones nativas: iOS, android? ¿presentan características similares a las de una aplicación web actual? ¿en qué son diferentes?

Los resultados obtenidos al hacer una aplicación web y una nativa pueden ser los mismos en cuanto a interfaz de usuario, sin embargo hay diferencias que no se aprecian a simple vista, como el rendimiento que cada una tiene en diferentes dispositivos. Por un lado, desarrollar una aplicación web es sencillo porque es multiplataforma y el tiempo que toma es relativamente corto. Asimismo es posible hacer aplicaciones web que se “vean bien” en dispositivos móviles (responsive web design). Por otro lado, las aplicaciones nativas permiten adaptarse al dispositivo en el que serán utilizadas por lo que la experiencia que dan al usuario es mejor.

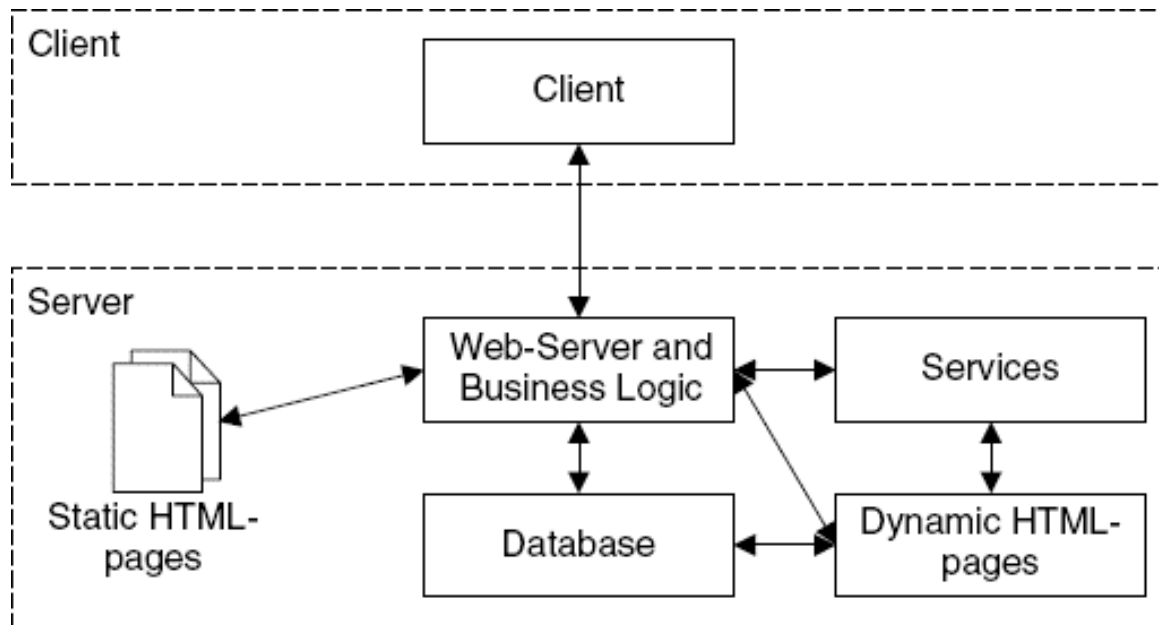
Historia

Anteriormente, el contenido era en su mayoría estático y solo era interactivo por contener hiper vínculos. Sin embargo cualquier cambio por más insignificante, significaba una ida y vuelta al servidor para refrescar el sitio completo.

JavaScript agregó elementos dinámicos para agregar contenido, sin embargo aún se necesitaba refrescar ciertos segmentos del sitio web.

Empezando en 2001 hasta 2005, se agregaron nuevos paradigmas de interacción con usuarios tal como XMLHttpRequest y AJAX. Estos permitieron enviar o recibir información sin recargar la página. Entonces pasamos de tener sitios web a aplicaciones web.

Arquitectura



La mayoría de los recursos en la Web se basan en el modelo cliente servidor y usando lenguajes de marcado para transferir o representar información: XML y HTML. Debajo de esa primera capa, existen varios lenguajes de programación o scripting que procesan, modifican o generan información dinámicamente e incluso proveen una interfaz de usuario.

El desarrollo de las aplicaciones web es multidisciplinario, ¿por qué crees que se dice eso?

Porque hoy en día no solo basta con saber desarrollar las aplicaciones. También es necesario considerar lo que el usuario espera de la aplicación, cómo interactuará con ella. Asimismo, es importante considerar aspectos de seguridad, ya que el usuario introduce información personal a algunas de las aplicaciones que utiliza a diario. Además, una aplicación web puede cumplir con distintos propósitos de diferentes áreas (económica, social, etc)

El desarrollo de aplicaciones web se encuentra en un ambiente cambiante y los requerimientos evolucionan a medida que la comunidad crece.

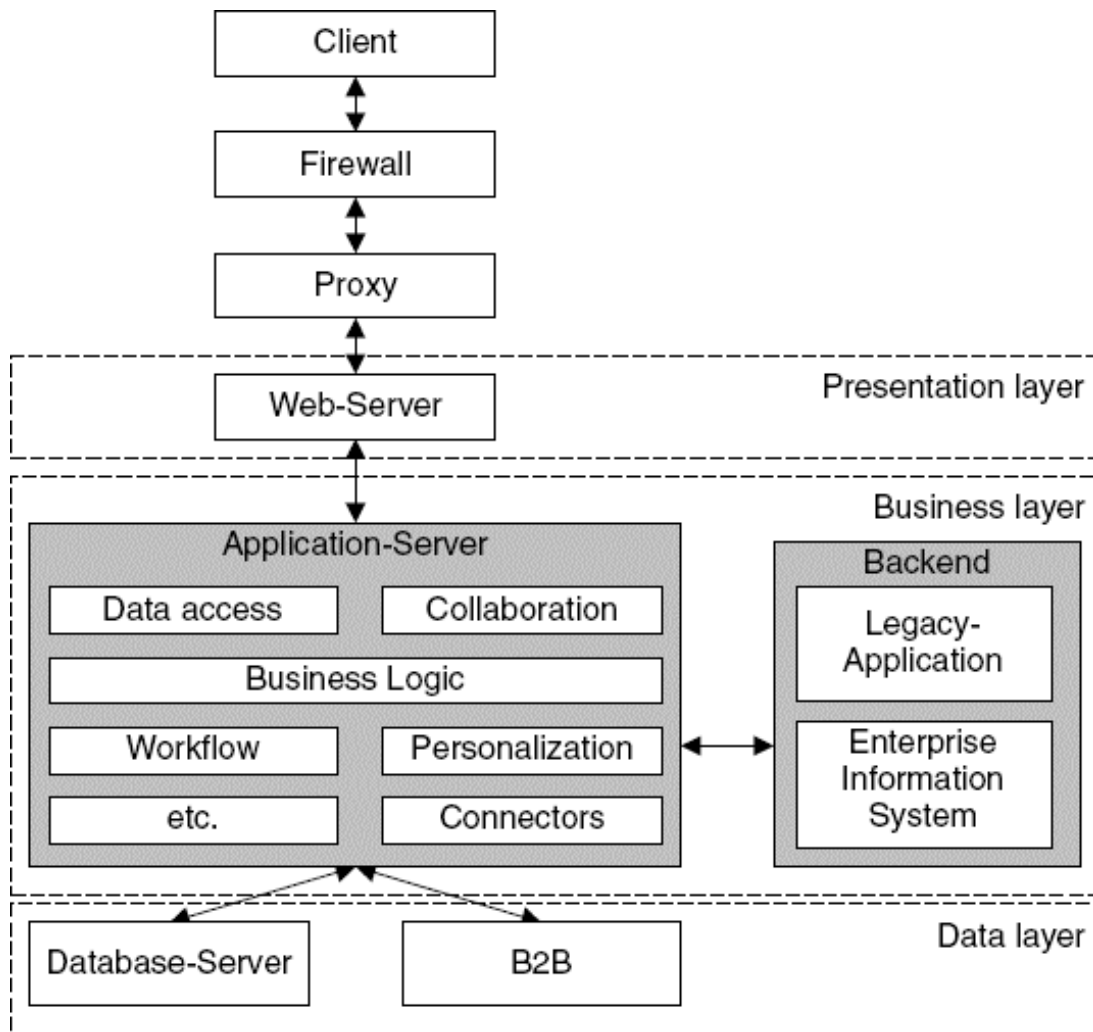
Las aplicaciones web manejan información de varios tipos: texto, gráficos, video, audio; por lo que hay retos al estructurar, procesar, almacenar y presentar la información.

En tu opinión (y experiencia), ¿cuál crees que es el reto más difícil de resolver: estructurar, procesar, almacenar o presentar información? y ¿por qué?

En mi opinión el reto más difícil es estructurar la información, ya que si no se hace bien desde un principio los cambios en un futuro son más complicados. Además, el desarrollo de una aplicación requiere de varias personas que deben estar de acuerdo en la estructuración y el ponerse de acuerdo puede llevar mucho tiempo. Sin embargo, presentar información también puede llegar a ser problemático porque involucra al usuario y sus expectativas y no siempre se puede llegar a satisfacer todas sus necesidades.

Modelo de capas:

Arquitectura escalable a cualquier tamaño o propósito de una aplicación web, es el estándar en la mayoría de las aplicaciones web. Apodada 3-tier architecture o n-tier architecture aunque en otros contextos layer se refiere a la distinción lógica entre las partes integrales de una aplicación y tier puede significar las estructuras físicas donde corre una aplicación (red, computadoras, servidores).



La parte más visible y la que interactúa con el cliente: the View layer. La capa intermedia es aquella que sirve como intermediario entre la presentación (View Layer) y la información (Data Layer), contiene reglas, personalización, entre otros: the Business Layer. La última capa se encarga de almacenar la información utilizada en la aplicación web: the Data Layer.

Una arquitectura dividida en capas, hace más sencillo su construcción y mantenimiento, así como rehusar código y rastrear errores (debug).

The View Layer

W3C: "A user wants to see the weather in her trip destination"

UI: "see weather in different periods of time or another places, and see pictures of it"

Conocida como vista, presentación, UI; esta capa se encarga de presentar la información necesaria al usuario y las herramientas para interactuar con esa información.

Las tecnologías utilizadas en esta capa son:

- Markup procesado por el navegador: HTML _____
- Estilo de la página: CSS _____
- Scripts del lado del cliente: JavaScript _____

Aunque la generación de contenido puede provenir de la capa de lógica del negocio, la capa de presentación se encarga de presentarla y puede modificarla: para implementar paradigmas interesantes de interacción con el usuario, para ser adaptable a cualquier dispositivo, ser accesible a cualquier persona, etc.

Después de presentar información, puede ser que el usuario a su vez responda. En ese caso, la capa de presentación pasa esta información de regreso a la capa de lógica del negocio, no dejando decisiones a la capa de presentación. El traspaso de información (enviar peticiones y recibir respuestas) puede ser asíncrono o síncrono.

¿Cuál es la diferencia entre peticiones síncronas y asíncronas?, ¿para qué utilizarías cada tipo de petición?

Una petición síncrona bloquea la ejecución del código y el usuario no puede hacer nada mientras se procesa la solicitud, mientras que en una petición asíncrona se procesa la solicitud dejando libre al navegador para que pueda hacer otras operaciones. Hay pocos casos en los que se utiliza una petición síncrona, como cuando se requiere hacer un proceso con varios pasos que siguen una secuencia y uno de esos pasos sucede en otra máquina.

The Business Logic Layer

W3C: "A user wants to see the weather in her trip

destination"UI: *takes the request and sends it to the server*

Business Logic layer: *takes the request and makes calls for the lower layer* ...

WAITS..."The weather in Pachuca is faircurrently,with a high temperature of 22° and a low of 7°."

Se encarga de la lógica del programa, recibe información de la capa de presentación y la transforma de acuerdo a reglas establecidas y recibe información de la capa de datos y la utiliza para formar dichas reglas.

Contiene la parte determinante de la lógica de la aplicación:

- Ejecuta todos los cálculos y validaciones
- Administra el flujo de trabajo:
 - Administración del estado: mantener registro de la ejecución de la aplicación
 - Administración de la sesión: distinguir diferentes instancias de la aplicación
 - Identificación del usuario
 - Acceso del servicio: proveer servicios de la aplicación de manera consistente
- Administra los accesos a información solicitados en la capa de presentación

La capa de la lógica del negocio se encuentra implementada, por lo general, en un servidor de aplicación que automatiza los servicios como transacciones, seguridad, persistencia, *connection pooling*, mensajes, DNS.

Las herramientas utilizadas en esta capa son:

- Server-side scripts: Ruby, Node.js (JS para el servidor) PHP, ASP.NET, CGI scripts like Perl

The Data Layer

W3C: "A user wants to see the weather in her trip

destination"UI: *takes the request and sends it to the server*

Business Logic layer: *takes the request and makes calls for the lower

layer*Data layer: *searches for the data using the given parameters and returns it*

Se encarga de recuperar información de sus fuentes. En un buen diseño de capas, forma una parte esencial del intercambio y reusabilidad de las tecnologías ya que se podría cambiar la fuente de los datos por otra pero la información genuina debería ser la misma.

Las tecnologías utilizadas en esta capa son:

- Database management systems: base de datos relacionales como MySQL, PostgreSQL, base de datos no relacionales como MongoDB, Apache Cassandra, plain XML o archivos de texto
- Mecanismo para realizar queries y recuperar la información:
 - API para los sistemas de administración
 - Scripts para los archivos

KAHOOT!

Genera dos preguntas acerca del tema con cuatro respuestas posibles:

¿Qué capa se encarga de recuperar información de sus fuentes?

- a) *Information Layer*
- b) *Business Layer*
- c) **Data Layer**
- d) *Presentation Layer*

¿Cuál no es una ventaja de las aplicaciones web?

- a) *Menor costo*
- b) **Buen rendimiento**
- c) *Multiplataforma*
- d) *Menor tiempo de desarrollo*

Bibliografía(agrega bibliografía extra)

- W3C (2016) Webdesign and applications. <https://www.w3.org/standards/webdesign/>
- Dr. Adamkó, Attila (2014) Layered Architecture of Web Applications. <https://gyires.inf.unideb.hu/GyBITT/08/ch04.html>
- Raona (2017). ¿App nativa, web o híbrida? <https://www.raona.com/aplicacion-nativa-web-hibrida/>
- ceviche.js (s.f.). Capítulo 6: Peticiones asíncronas. <http://cevichejs.com/6-xhr>
- Mozilla. (s.f.). Solicitudes síncronas y asíncronas. https://developer.mozilla.org/es/docs/Web/API/XMLHttpRequest/Synchronous_and_Asyncronous_Requests
- stackoverflow (2017). Asynchronous vs synchronous execution, what does it really mean? <https://stackoverflow.com/questions/748175/asynchronous-vs-synchronous-execution-what-does-it-really-mean>