¿Qué es una aplicación web?

Definición de **aplicación**: software que permite realizar al usuario una determinada tarea. Ejemplos:

- Aplicaciones de escritorio: usan recursos de la computadora. Lenguajes como C, C++,...

  Ejemplos: ¿?

- Aplicaciones web: es una aplicación que se ejecutan en un navegador (Google Chrome, Mozilla Firefox, Internet Explorer…).

  Ejemplos: ¿?

Definición de **navegador**: software que permite el acceso a internet, interpretando la información de distintos tipos de archivos y sitios web para que éstos puedan ser visualizados. Los navegadores están presentes incluso en dispositivos más pequeños como tabletas o smartphones.

Funcionalidad: visualizar documentos, visitar páginas web, enlazar un sitio con otro, imprimir, enviar, recibir correos, etc...

Conceptos:

Hipervínculos: enlazan texto o imagen a otro documento

Navegación: seguimiento de enlaces de una página a otra

URL (Uniform Resource Locator): Identificador único para acceder a los recursos:

Formato: protocolo://máquina/directorio/archivo

Ejemplo: https://www.google.es

La comunicación entre el servidor web y el navegador se realiza mediante el protocolo HTTP, aunque la mayoría de los navegadores soportan otros protocolos como FTP y HTTPS (una versión cifrada de HTTP basada en Secure Socket Layer o Capa de Conexión Segura (SSL)

La función principal del navegador es obtener documentos HTML e interpretarlos para mostrarlos en pantalla. En la actualidad, no solamente descargan este tipo de documentos sino que muestran con el documento sus imágenes, sonidos e incluso vídeos streaming en diferentes formatos y protocolos. Además, permiten almacenar la información en el disco o crear marcadores (bookmarks) de las páginas más visitadas.

Proceso de desarrollo de aplicaciones web

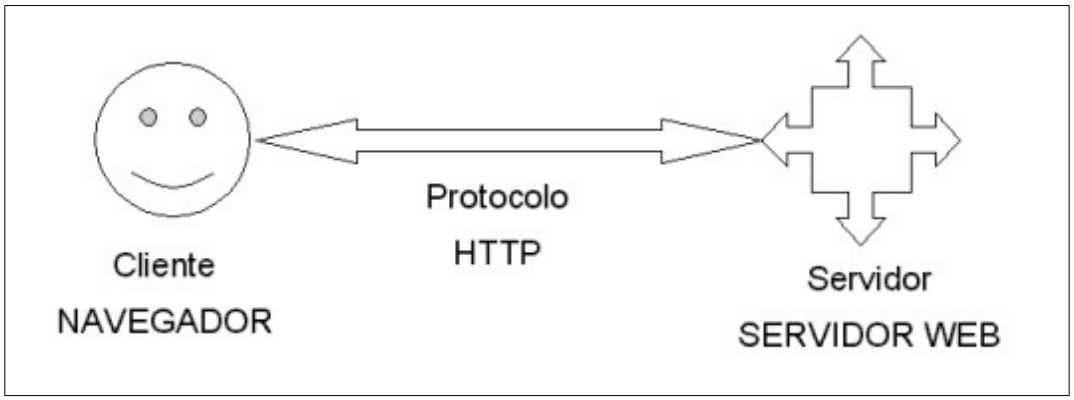
Cuando creamos aplicaciones web, se pueden utilizar diversos lenguajes y tecnologías. Principalmente hay dos tipos de tecnologías a tener en cuenta:

Tecnologías en el lado del cliente

Elementos que se incorporan junto al código HTML y son interpretados por un navegador. El código puede ser examinado por el usuario.

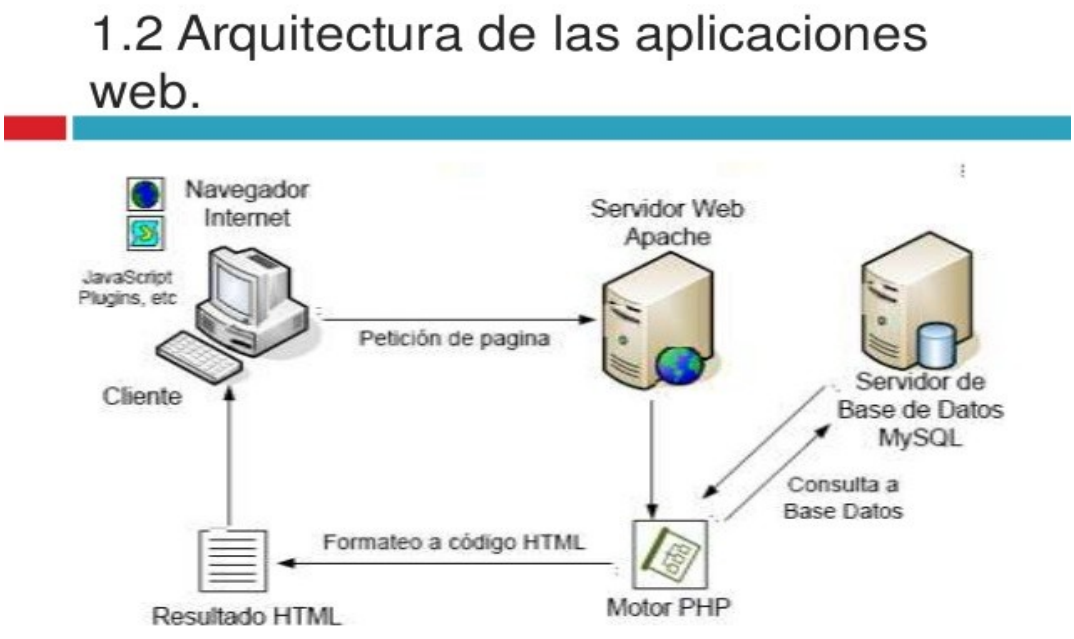
Tecnologías en el lado del servidor

La programación que es interpretada por un servidor web. El código queda oculto al usuario ya que se ejecuta de forma externa.



Esquema básico de una aplicación web

Ejemplo de la arquitectura para una aplicación web con php y base de datos mysql.



1.- El usuario especifica en el cliente web la dirección de la página que desea consultar. Escribe en el navegador la dirección (URL) de la página a visitar: **https://www.ceuandalucia.es**

2.- El cliente establece una conexión con el servidor web.

3. El cliente solicita la página o el objeto deseado.

4. El servidor envía dicha página u objeto (o, si no existe, devuelve un código de error).

5. Si se trata de una página HTML, el cliente inicia sus labores de interpretación de los códigos HTML. Si el cliente web encuentra instrucciones que hacen referencia a otros objetos que se tienen que mostrar con la página (imágenes, sonidos, animaciones multimedia, etc.), establece automáticamente comunicación con el servidor web para solicitar dichos objetos.

6. Se cierra la conexión entre el cliente y el servidor.

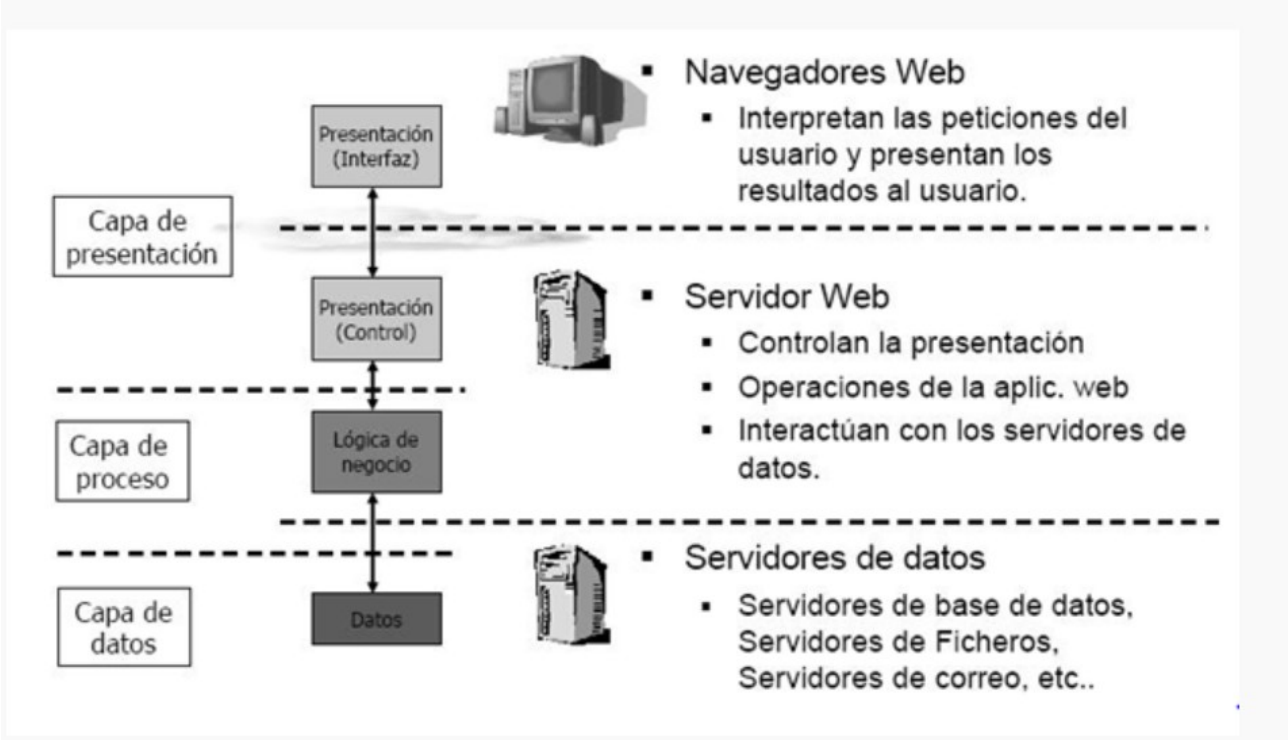
7. Se muestra la página al usuario

Arquitectura en tres niveles

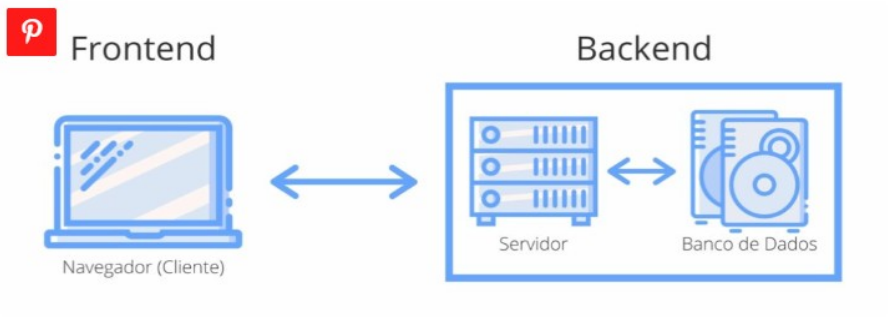
- **Capa de presentación.**
  Conocida como capa del cliente o el navegador.
  Controla la parte de la aplicación web que llega al navegador.
  Front-end. Ejemplos: HTML, CSS, Javascript

- **Capa lógica o proceso**
  Conocida como capa del servidor o de aplicación
  Gestiona el funcionamiento interno de la aplicación.
  Back-end. Ejemplos: PHP, ASP, Java, Javascript (lado servidor)

- **Capa de negocio o datos**
  Conocida como capa de datos.
  SGBD, recursos correo, audio, certificados..
  Back-end.



Por lo tanto…



Flujo web simple

¿Tecnologías del lado del cliente y servidor?

Investiguemos… Buscar en Internet 5 tecnologías en lado cliente y 5 en lado servidor.

Ejemplos de tecnologías lado cliente:
HTML, CSS, JavaScript, XML, JSON, SVG, Flash, Applets

JavaScript (JS). Un poco de historia

JavaScript es un lenguaje de programación creado por Brendan Eich en 1995 con el objetivo de hacer la navegación web más dinámica e interactiva. Se creó para el navegador Netscape Navigator. Se convertió en standard en el año 1997.

Los programas escritos en JS, se ejecutan por el navegador web y no en el servidor donde se encuentra alojado el sitio.

Por otro lado, Microsoft creó en el años 1996 un lenguaje para Internet Explorer 3, jScript. Con características específicas y comportamientos distintos a los de JavaScript. Por lo tanto los sitios desarrollados con una de las dos versiones del lenguaje no eran compatibles con todos los navegadores.

Esta competitividad y guerra de tecnologías provocó la estandarización del lenguaje: ECMA (European Computer Manufacturers Association).

ECMA: https://www.w3schools.com/js/js_versions.asp

ECMA es una asociación fundada en Ginebra en 1961 para crear estándares del sector informático y de las telecomunicaciones. El primer estándar de JS que creó se denominó ECMA-262, en 1997, en el que se definió por primera vez el lenguaje ECMAScript. (dicta la base del lenguaje ECMAScript a través de su sintaxis, tipos, sentencias, palabras clave y reservadas, operadores y objetos)

Por este motivo, algunos programadores prefieren la denominación ECMAScript para referirse al lenguaje JavaScript. De hecho, JavaScript no es más que la implementación que realizó la empresa Netscape del estándar ECMAScript.

Desde entonces se han ido sucediendo distintas versiones de ECMAScript de forma anual:

EN 1998 se realizaron pequeñas modificaciones para adaptarlo al estándar ISO/IEC-16262 y se creó la segunda edición. *ECMAScript 2*

La tercera edición del estándar ECMA-262 (publicada en Diciembre de 1999) *ECMAScript 3* con soporte de expresiones regulares, nuevas sentencias de control, manejo de excepciones (bloque try-catch), definición de errores más precisa, formateo de salidas numéricas de datos, manejo de strings más avanzado...

ECMAScript 4, que aparece en 2004. (Nunca se lanzó)

Las versiones de referencia son la 5 y la 6.

Las características que se incluyen en ECMAScript5 o ES5 (2009) (Versión actual de la mayoría de los navegadores con la aceptación de HTML 5 como estándar) son las siguientes: Getters y setters, Array extras y reductions, Rediseño de los atributos internos de las propiedades, Introducción de métodos estáticos de Object, cambios en el objeto Date, soporte nativo de JSON...

ECMAScript 6 o ES6 (2015): ES2015: Módulos, Ámbito a nivel de bloque (sentencia let),Iterators, String templates, hash tables..

Así sucesivamente...

Diferentes versiones de JavaScript junto con su nombre oficial y el año de su lanzamiento:

Nombre	Versión	Fecha
ES14	ES2023	Junio del 2023
ES13	ES2022	Junio del 2022
ES12	ES2021	Junio del 2021
ES11	ES2020	Junio del 2020
ES10	ES2019	Junio del 2019
ES9	ES2018	Junio del 2018
ES8	ES2017	Junio del 2017
ES7	ES2016	Junio del 2016
ES6	**ES2015**	**Junio del 2015**
ES5.1	ES5.1	Junio del 2011
ES5	ES5	Diciembre del 2009
ES4	ES4	Descartado
ES3	ES3	Diciembre de 1999
ES2	ES2	Junio de 1998
ES1	ES1	Junio de 1997

JavaScript (JS). Ventajas

- Respuesta a la interacción del usuario con elementos de formulario y enlaces hipertexto
- Interfaz amigable
- Controlar múltiples ventanas, marcos, applets java en la elección del usuario en la página HTML
- Pre-procesar datos en el clientes antes de enviarlos al servidor
- Modificar estilos y contenido dinámicamente
- Solicitar ficheros del servidor y enviar peticiones de lectura y escritura al servidor

JavaScript (JS). Limitaciones

- Compatibilidad con los navegadores: los distintos navegadores web interpretan el código JavaScript de forma diferente, lo que provoca incoherencias. Por lo tanto, debes probar tu script JavaScript en todos los navegadores web populares, incluyendo sus versiones más antiguas, para evitar perjudicar la experiencia del usuario.
- Seguridad: el código JavaScript que se ejecuta en el lado del cliente es vulnerable a la explotación por parte de usuarios irresponsables.
- Depuración: aunque algunos editores de HTML admiten la depuración, son menos eficaces que otros editores. Dado que los navegadores no muestran ninguna advertencia sobre los errores, encontrar el problema puede ser un reto.

¿Cómo integramos código JavaScript en HTML?

Hay tres maneras de poder integrar código JS en HTML

1.- Embebido dentro de la etiqueta <script>

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
   <title>Ejemplo de código JavaScript en el propio documento</title>
  <script type="text/javascript">
     console.log("Un mensaje de prueba");
   </script>
</head>
<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

2.- Instrucciones dentro del código

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Ejemplo de código JavaScript en el propio documento</title>
</head>
<body>
    <p onclick="console.log('Un mensaje de prueba')">Un párrafo de texto.</p>
</body>
</html>
```

3.- Archivo externo

```
<head>
    <script type="text/javascript" src="/js/codigo.js"></script>
</head>
```

Tendremos un fichero: codigo.js por ejemplo con el código: console.log("Un mensaje de prueba");

¿Cual pensáis que es la mejor de las tres?

Ventajas de utilizar fichero externo:

- Carga más rápida de páginas
- Se separa el comportamiento de la estructura
- Se comparte código entre páginas
- Es más sencillo depurar errores
- Modularidad
- Seguridad

Para finalizar...

1. ¿Sabríais explicar con vuestras palabras del funcionamiento de las aplicaciones web?
2. ¿Sabríais encontrar que versiones de los navegadores empezaron a soportar el ES2015?