Tecnologías y Aplicaciones Web

2021-1

Apuntes by Ariadna Camino

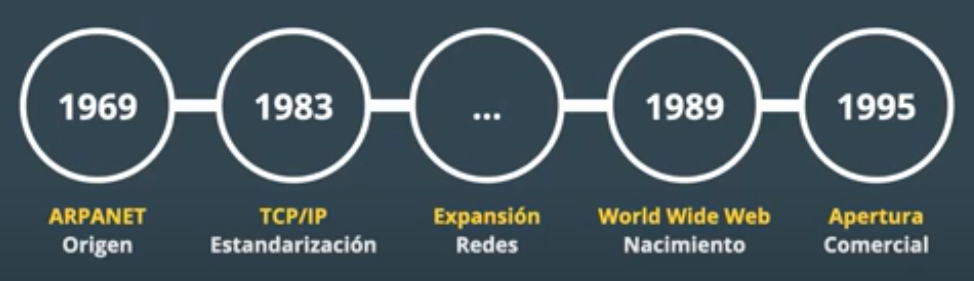
Profesor: Sebastián Vicencio

16-03-2021

**Resumen primera clase**

Fundamentos de la web

* Internet **no** es lo mismo que la Web.
* **Internet:** Infraestructura sobre la cual está construida la Web. Es una red interconectada de computadores mediante cables (físico).
* TCP/IP: Protocolo que permite la transmisión de datos a través de la infraestructura de internet.
* Paquete: Grupo de datos (bits) enviados a través de internet.
* Analogía con una ciudad:
* La dirección de cada domicilio es una IP.
* Los domicilios son los computadores.
* Domicilios interconectados mediante calles = Computadores interconectados mediante cables.
* Medio de transporte para moverse de un domicilio a otro: Protocolo TCP/IP.
* Autos llevan paquetes.
* A veces los paquetes no caben en un solo auto.
* **Historia Internet:** Hitos importantes.

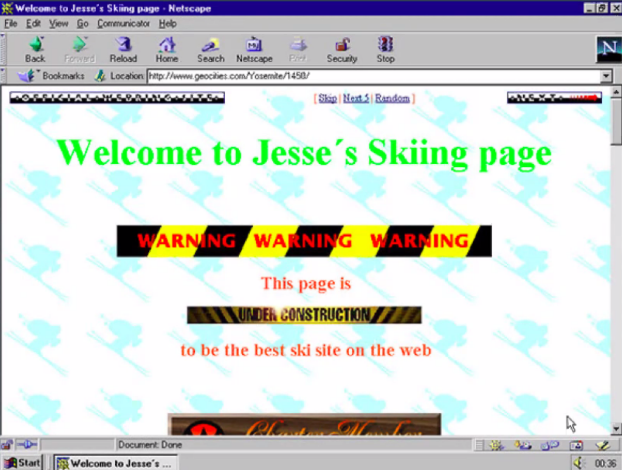


18-03-2021

**Resumen segunda clase**

World Wide Web

* Creada en 1989 (Tim Berners-Lee). Su paper:
* **Information Managment: A proposal**
* Compartir documentos académicos.
* **Primer sitio web**: Finales de 1990 -> **World Wide Web**.
* Se **libera** el código fuente (Web es **Open Source**): 1993

Web 1.0 (Antes)

* Sitios web solo de lectura, no interactiva.
* Unidireccional.

Web 2.0 (Ahora)

* Usuarios pueden crear.
* Existe participación.
* La Web como plataforma (ej. Facebook).

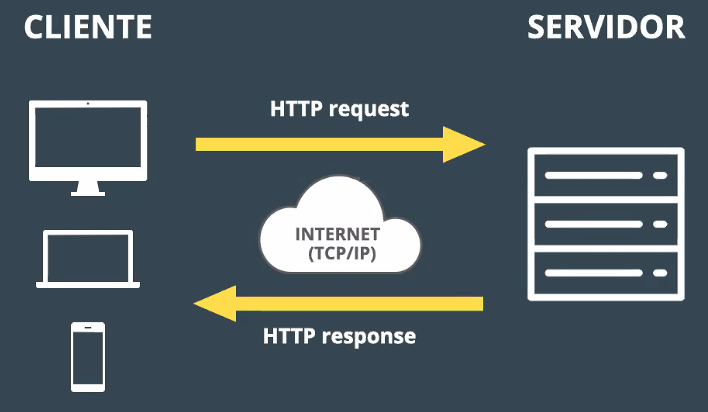
¿Qué sigue?

* **Web semántica**: Pueden acceder robots además de humanos.

Cómo funciona la Web

Conceptos básicos

Arquitectura **Cliente – Servidor**

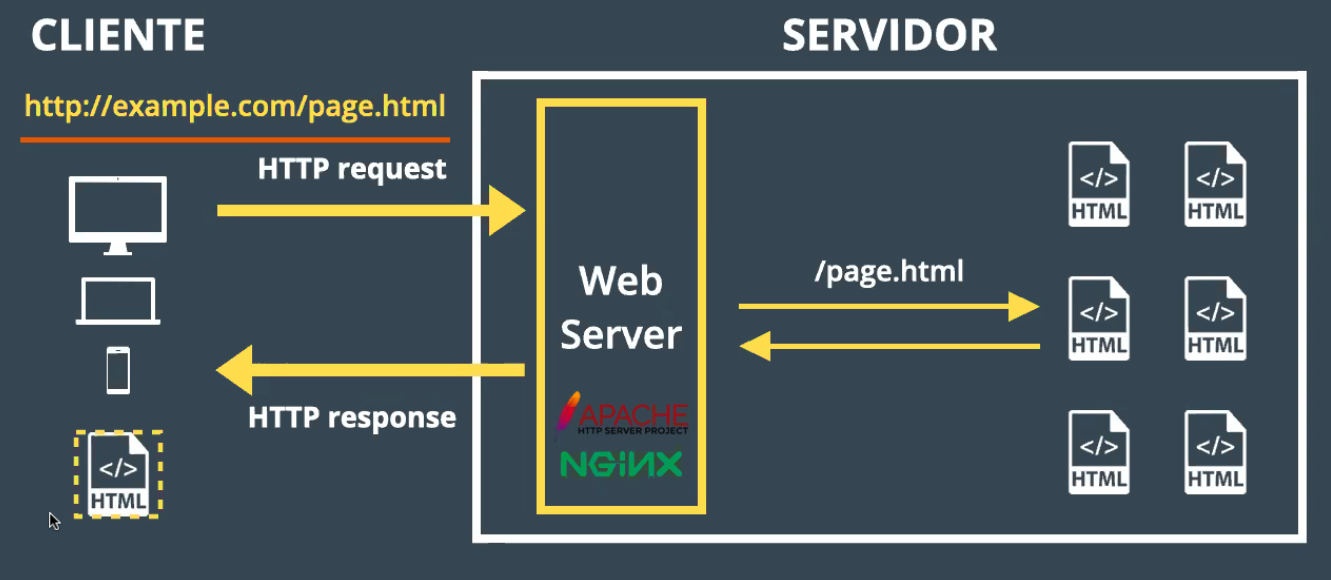


* El intermediario es un protocolo que es un HTTP.
* HTTP permite que cliente y servidor puedan entenderse entre sí.
* Funciona con solicitudes y respuestas.
* **Protocolo HTTP**: Permite que cliente y servidor puedan entenderse. Funciona con: Request/response.
* **Recurso web:** Identificado por una URL (dónde está alojado el recurso y refleja el recurso en sí), lo que uno quiere buscar en el servidor.

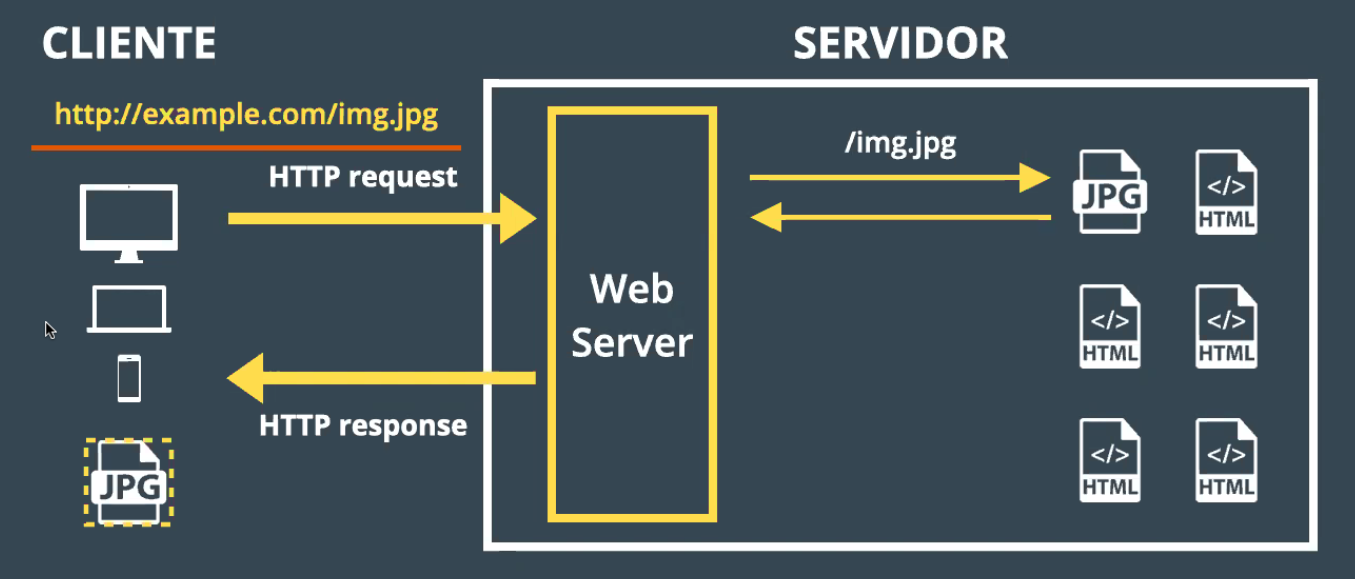
**Aplicaciones web tradicionales**

Evolución cliente – servidor

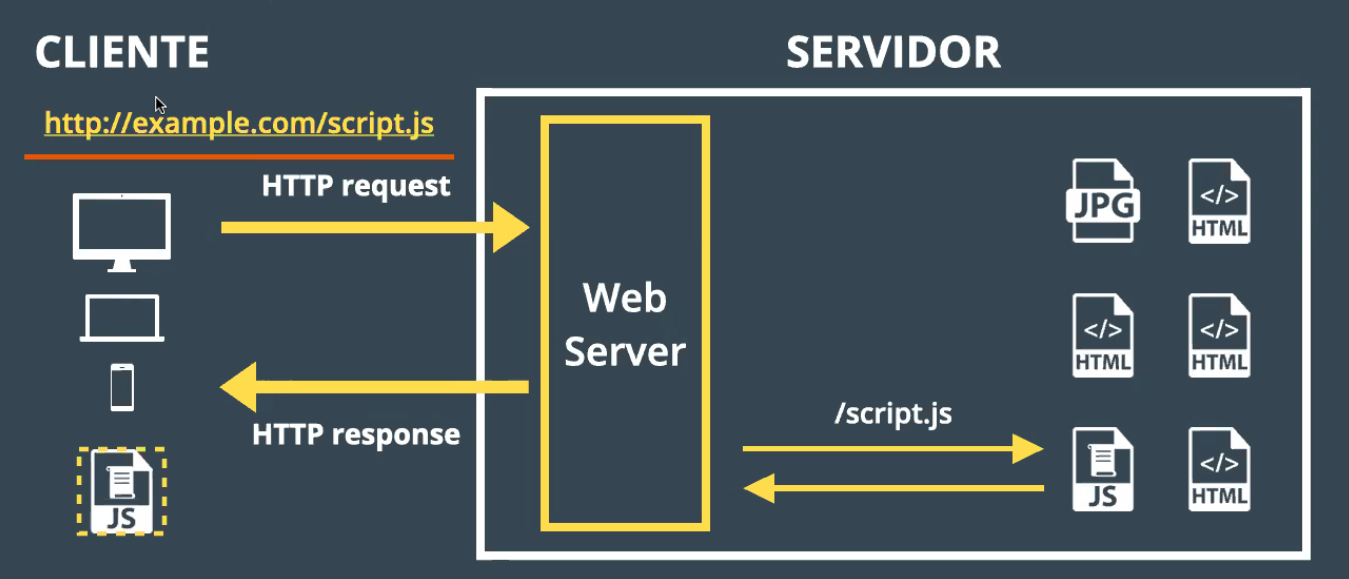
* Inicios de la web (1990 – 1993):



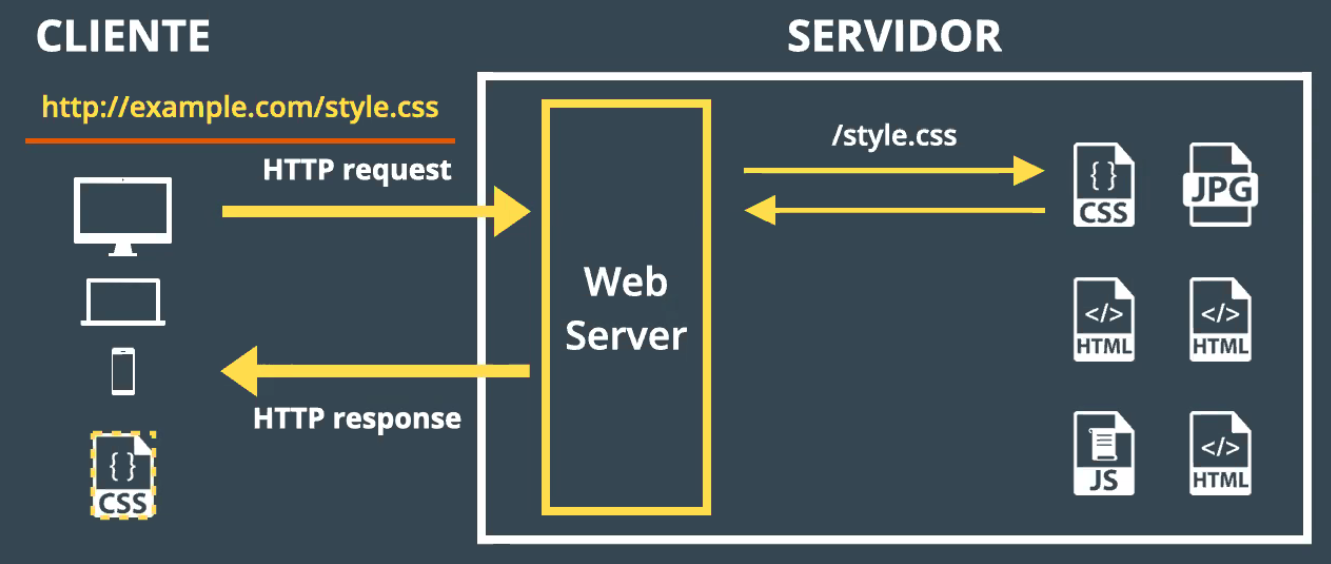
* Después podían enviarse más documentos que solo HTML (por ejemplo, imágenes).



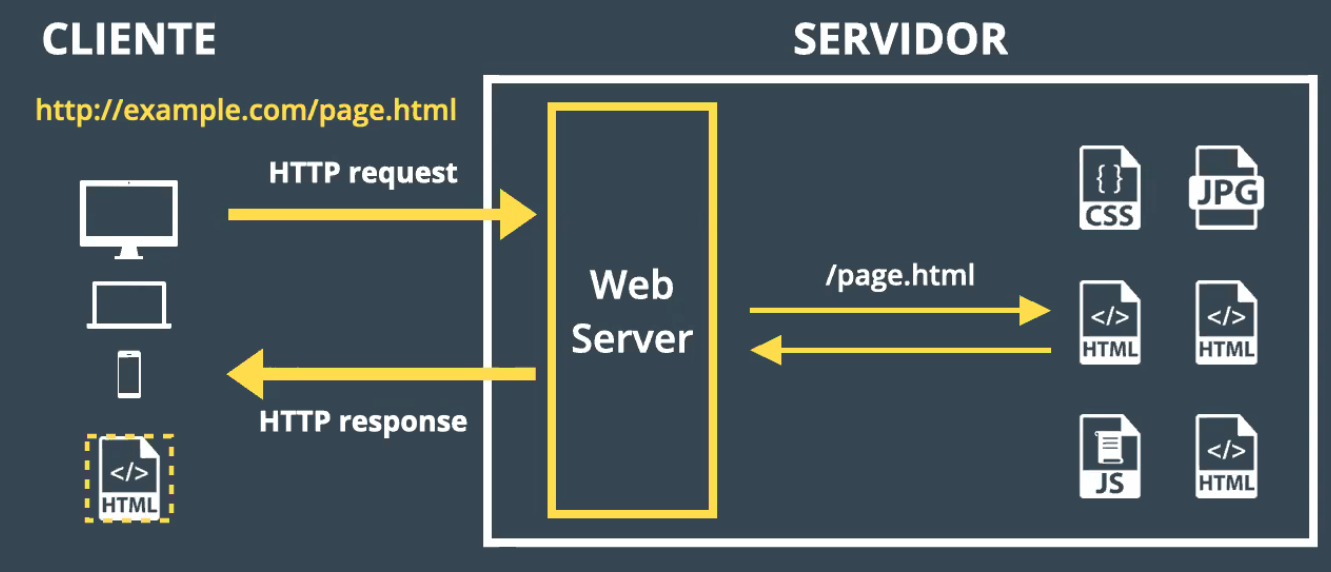
* En 1995 surge JavaScript (se ejecutaba en el browser, no en el servidor).



* Surge CSS entre 1994 – 1996 (Embellece las páginas).
* Los tres lenguajes vistos hasta ahora se ejecutan en el cliente.



* Sitio web estático: Un documento por página.

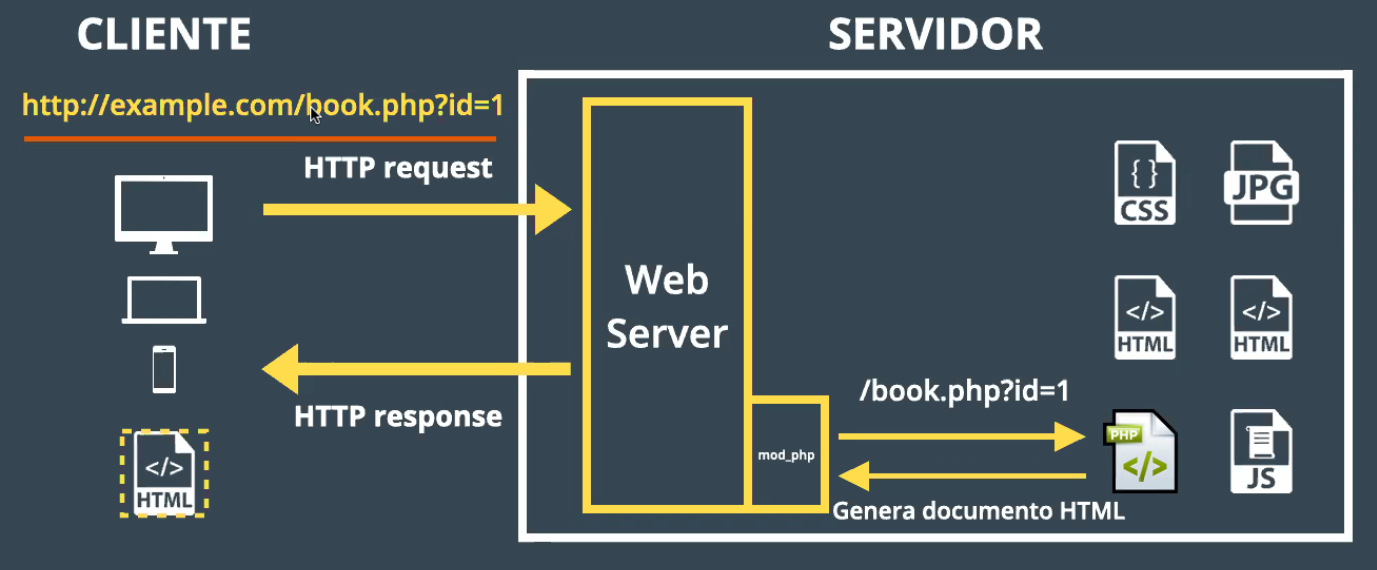


Sitio web estático

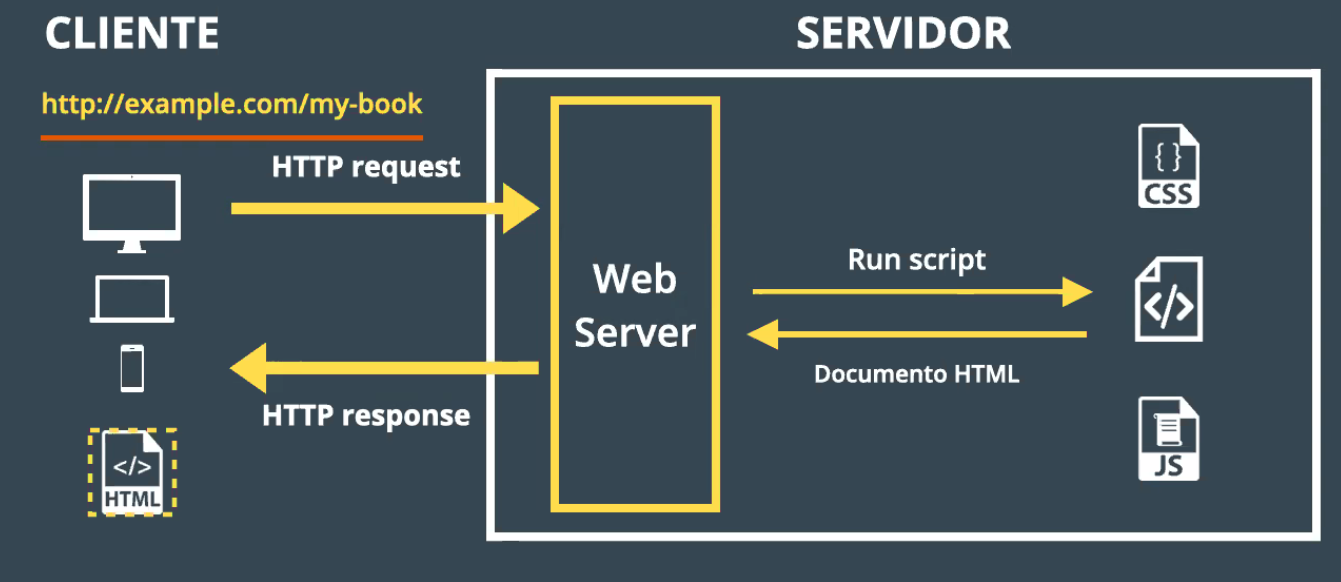
* Librería con veinte libros = veinte documentos HTML.
* ¿Y si quisiéramos tener cien libros? ¿Y si decidiéramos cambiar la forma de mostrar los libros?

Sitio web dinámico

* Ya no solo entregar documentos, también ejecutar scripts.



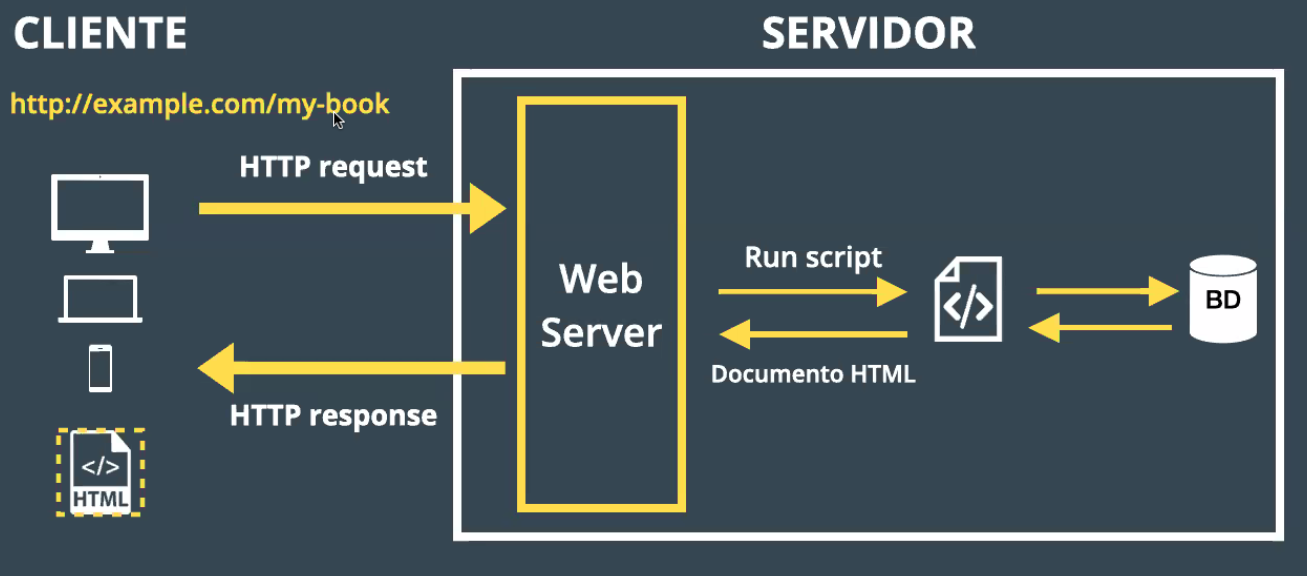
* **Sitio web dinámico**: documento HTML generado.



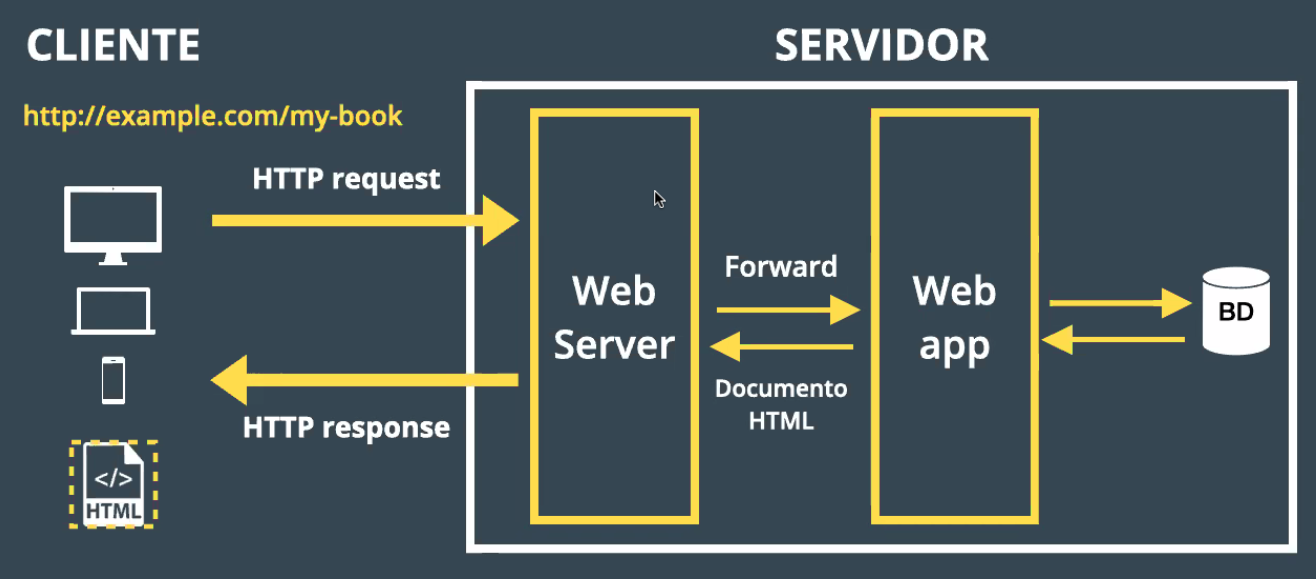
23-03-2021

**Resumen tercera clase**

* Complejidad crece (fines 90 – principios 2000). Ahora puede consultar a una base de datos.



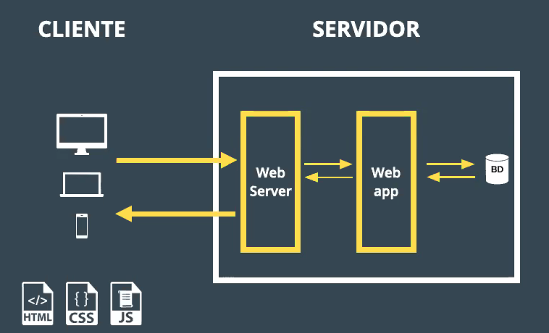
* Surge la aplicación web.
* Recibe request HTTP, decide qué hacer (consultar base de datos, escribir correo, etc.) y se entrega la respuesta.



Aplicación web

* Aplicación que **realiza acciones en el servidor** en base a un request HTTP.
* **Construye de forma dinámica una respuesta**, la cual se envía de vuelta como un response HTTP.
* Coordinación entre cliente y servidor:

**Aplicación web tradicional**

1. Request recurso web.
2. Servidor consulta BD.
3. Servidor genera documento HTML.
4. Cliente recibe e interpreta documento HTML.
5. Documento HTML tiene referencia a CSS.
6. Request al servidor del documento CSS.
7. Servidor devuelve documento CSS.
8. Cliente despliega página web con estructura y estilo.
9. Documento HTML (inicial) tiene referencia a script JS.
10. Request al servidor script JS.
11. Servidor devuelve script JS (tal como está.
12. Cliente recibe y ejecuta script JS.

* En términos de tecnologías



Lenguajes y frameworks web



Backend

* Buscar y/o persistir información en una BD.
* Controlar acceso a contenido.
* Manejar sesiones de usuarios.
* Conexión con servicios (correo, transacciones).

Frontend

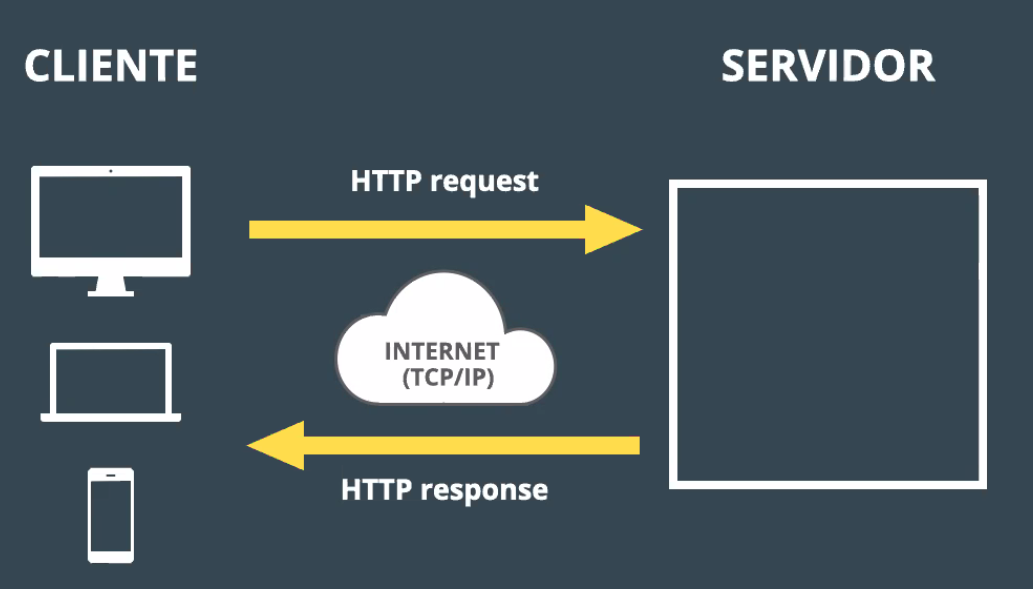
* **Estructura (HTML):** Contenido de una página web.
* **Presentación (CSS):** Aplicar estilos y layout.
* **Comportamiento (JavaScript):** Generar cambios visuales en base a interacción con el usuario.

25-03-2021

**Resumen cuarta clase**

Protocolo HTTP

* **Interacción cliente – servidor** (siempre iniciada por el cliente):



* **HTTP (HyperText Transfer Protocol):** Protocolo que permite intercambio de información en la Web.
* **Hipertexto:** Acceso a documentos mediante **links**.
* **Link:** Texto que contiene **referencia a otro documento** (hipertexto).
* **Recursos**:
* Motivo de la comunicación.
* “Cualquier cosa” que se quiera transmitir. Documento, foto, texto plano.
* Identificado por una URI (o URL).
* **HTTP: Interfaz para interactuar con recursos.**
* Se transmite una **representación del recurso:**

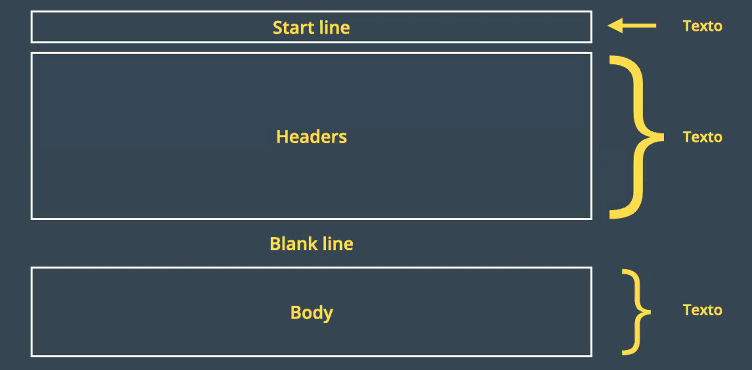
1. Recurso: abstracto.
2. Representación: concreto.

* **HTTP: Características principales**

1. Simple: Posible de ser leído por humanos.
2. Extensible: Información arbitraria por medio de headers.
3. Stateless: Dos request consecutivos son independientes.

Mensajes HTTP (La base de la comunicación)

* Texto compuesto por múltiples líneas.
* Se dividen en dos tipos, request y response.
* Estructura general:



* Body es opcional.
* Ejemplo:



* Los headers tiene la estructura Llave: Valor.
* Request header: Automáticos por el browser.
* Entity header: Relacionado al cuerpo.

Métodos HTTP (Propósito del request)

* Acción que cliente espera que servidor realice.
* **GET:** Obtener representación de un recurso específico.
* **POST:** Crear un nuevo recurso en base a entidad adjunta.
* **PUT:** Reemplazar recurso específico con entidad adjunta.
* **PATCH:** Modificar recurso específico en base a entidad adjunta.
* **DELETE:** Eliminar recurso específico.

30-03-2021

**Resumen quinta clase**

Response HTTP (Mensaje del servidor)

* Estructura:

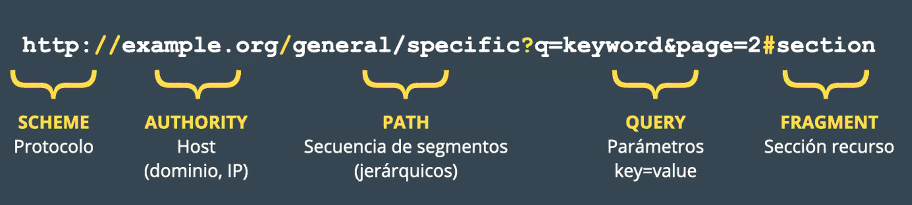


Status codes

* Tipo de respuesta del servidor.
* Tres dígitos.
* **1xx**: Informativo -> Request recibido y continuando proceso.
* **2xx**: Éxito -> Request recibido, entendido y aceptado.
* **3xx**: Redirección -> Acciones necesarias para completar request.
* **4xx**: Error del cliente -> Request con sintaxis errónea o no válido.
* **5xx**: Error del servidor -> Servidor falló en completar request.

URI (Uniform Resource Identified)

* Texto que permite **identificar** de forma única un **recurso web**.
* **URL**: Tipo de URI que incorpora información de la **ubicación del recurso web**.
* **Estructura de URI**:



* **Host**: Servidor en el cual está alojado el recurso al cual se quiere hacer referencia.
* **Dominio**: Nombre asociado a una IP.

Paths

* Convención para orden jerárquico.
* **/users** -> Colección.
* **/users/:id** -> Un elemento de la colección.
* **/users/:id/posts** -> Colección como sub-recurso.
* **/useres/:userId/posts/:id** -> Sub-recurso.

Paths + método HTTP

* Acciones sobre recursos.
* **GET /users**
* **POST /users**
* **GET /users/2**
* **PATCH /users/2**
* **PUT /users/2**
* **DELETE /users/2**

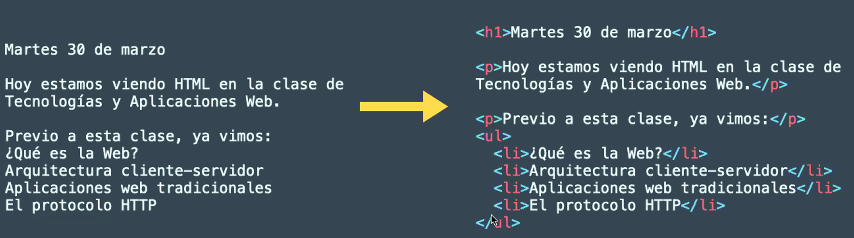
HTML (HyperText Markup Language)

¿Qué es?

* Lenguaje de marcado
* Permite **representar contenido y estructura** de una página web.
* Nos permite darle **significado** **al contenido web.**
* Títulos, párrafos, listas, imágenes, links, etc.

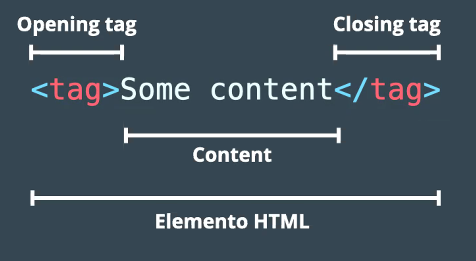
Markup

* El contenido se debe **marcar con etiquetas**.



Elemento HTML

* Estructura:



* Existen muchos elementos HTML, cada uno con un **propósito específico**.

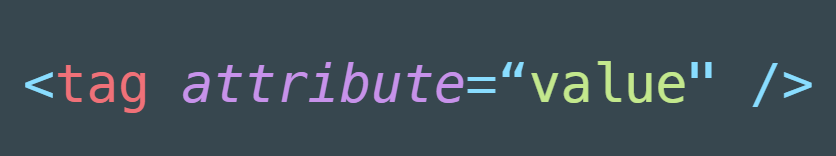
Atributos HTML

* Información extra de un elemento.
* En general, **no tiene efecto visible.**
* Múltiples atributos **separados por espacios.**
* Estructura:



Self-closing tags

* **No marcan** contenido visible.
* Especifican otro tipo de contenido.



Etiquetas más utilizadas

