Licenciatura en Sistemas de la Información

Cátedra: Programación Avanzada

Profesor: Lic. Mg Ledesma Ernesto

Alumno: Rojas Fernando

Trabajo practico N°1

2024

Historia de la WWW

1. ¿Como surgió el sistema de documentos distribuidos WWW?

2. ¿En qué año se creó la www? ¿Quién la creo? ¿Cuál fue el propósito inicial?

3. ¿Sobre qué red funcionaba? ¿Cuáles fueron los componentes iniciales?

1- El sistema de documentos distribuidos conocido como la World Wide Web (WWW) surgió para solucionar la necesidad de compartir información entre investigadores de todo el mundo. Fue desarrollado por Tim Berners Lee en el CERN a finales de los 80 y principios de los 90. Su propuesta incluyó el uso de hipertexto para enlazar documentos, el protocolo HTTP para la transferencia de datos, y las direcciones URL para localizar los documentos, facilitando así el acceso y la interconexión de la información a nivel global.

2- La World Wide Web (WWW) se creó en 1989. Fue desarrollada por Tim Berners-Lee en el CERN (Organización Europea para la Investigación Nuclear). El propósito inicial de la WWW era facilitar la comunicación y el intercambio de información entre científicos de distintas partes del mundo, utilizando un sistema de documentos enlazados por hipertexto accesibles a través de internet.

3- La World Wide Web (WWW) funcionaba sobre la red de Internet. Los componentes iniciales clave incluían el protocolo **HTTP** (Hypertext Transfer Protocol), el sistema de direcciones **URL** (Uniform Resource Locator) para localizar recursos, y el lenguaje **HTML** (Hypertext Markup Language) para crear y enlazar documentos. Estos elementos permitieron que los usuarios accedieran a información distribuida a través de diferentes servidores mediante navegadores web.

WWW en la actualidad

4. Nombre y describa las distintas etapas de evolución en la web. Realice una cronología de tecnologías que surgieron en la web hasta la actualidad.

Cronología de la evolución de la web y sus tecnologías

La web ha experimentado una evolución constante desde su creación en 1989, con la aparición de nuevas tecnologías que han transformado la forma en que la usamos. A continuación, se presenta una cronología de las etapas más importantes de la evolución de la web, junto con las tecnologías que surgieron en cada etapa:

1. Web 1.0 (1989-1995): La web estática

- **Características:**
  - Páginas web estáticas, creadas principalmente con HTML.
  - Interactividad limitada, principalmente a través de formularios.
  - Acceso a la información limitado a un número reducido de usuarios.
- **Tecnologías:**
  - HTML (HyperText Markup Language)
  - HTTP (HyperText Transfer Protocol)
  - Gopher

2. Web 2.0 (2004-actualidad): La web social y participativa

- **Características:**
  - Páginas web dinámicas y contenido generado por el usuario (UGC).
  - Interactividad y colaboración entre usuarios a través de redes sociales, blogs y foros.

- Mayor accesibilidad y uso generalizado de la web.
- **Tecnologías:**
  - XML (Extensible Markup Language)
  - CSS (Cascading Style Sheets)
  - JavaScript
  - AJAX (Asynchronous JavaScript and XML)
  - Web Services
  - Redes sociales (Facebook, Twitter, etc.)
  - Blogs (WordPress, Blogger, etc.)
  - Wikis (Wikipedia)

3. Web 3.0 (en desarrollo): La web semántica

- **Características:**
  - Web inteligente y contextualizada.
  - Comprensión del significado del contenido web por parte de las máquinas.
  - Interacción más natural entre humanos y máquinas.
- **Tecnologías:**
  - RDF (Resource Description Framework)
  - OWL (Web Ontology Language)
  - SPARQL (SPARQL Protocol and RDF Query Language)
  - Inteligencia artificial
  - Machine learning

Esta cronología es una simplificación de la compleja evolución de la web. Cada etapa ha sido marcada por la aparición de nuevas tecnologías que han transformado la forma en que usamos la web y la forma en que se crea y consume la información

5- ¿Qué es un Servidor WWW?

Un servidor WWW, o servidor web, es como un bibliotecario digital que guarda y entrega libros (páginas web) a sus lectores (navegadores web) cuando se los solicitan.

6- ¿Qué es un cliente web? explique detalladamente

Un cliente web es un software o aplicación que te permite acceder y visualizar contenido de la World Wide Web. Es como una puerta de entrada a Internet que te conecta con los servidores que almacenan páginas web, imágenes, videos y otros tipos de información.

Imagina que la web es una biblioteca gigante. Los libros de esta biblioteca son las páginas web y los estantes son los servidores. Tú, como usuario, necesitas una herramienta para encontrar los libros que te interesan y leerlos. Esa herramienta es el cliente web, más comúnmente conocido como navegador web.

7- ¿Como está compuesto el sistema WWW actual?

El sistema WWW, o World Wide Web, es una vasta red de información interconectada que ha evolucionado significativamente desde sus inicios. Si bien su funcionamiento puede parecer complejo, en esencia, se compone de varios elementos clave que trabajan en conjunto para brindarnos la experiencia de navegación que conocemos hoy en día.

Componentes Fundamentales

Clientes (Navegadores):

Interfaz de usuario: La parte visible del navegador que permite a los usuarios interactuar con la web, como la barra de direcciones, pestañas, y botones.

Motor de renderizado: Interpreta el código HTML, CSS y JavaScript de las páginas web y las muestra en la pantalla de forma visual.

Motor de JavaScript: Ejecuta el código JavaScript para agregar interactividad a las páginas web.

Red: Se encarga de enviar y recibir datos a través de Internet.

Servidores:

Hardware: Las computadoras físicas que almacenan los archivos de los sitios web.

Software: Sistemas operativos y software de servidor (como Apache, Nginx) que gestionan las solicitudes de los clientes y envían las páginas web correspondientes.

Bases de datos: Almacenan la información dinámica de los sitios web, como los datos de los usuarios o los productos de una tienda en línea.

Protocolos:

HTTP (HyperText Transfer Protocol): El protocolo principal utilizado para la comunicación entre clientes y servidores. Define cómo se envían y reciben las solicitudes y respuestas.

HTTPS (HyperText Transfer Protocol Secure): Una versión segura de HTTP que utiliza encriptación para proteger la comunicación entre el cliente y el servidor.

DNS (Domain Name System): Un sistema que traduce los nombres de dominio (como www.ejemplo.com) a direcciones IP numéricas que las computadoras pueden entender.

Lenguajes de Marcado:

HTML (HyperText Markup Language): Define la estructura y el contenido de una página web.

CSS (Cascading Style Sheets): Se utiliza para dar estilo y formato a las páginas web.

JavaScript: Un lenguaje de programación que permite agregar interactividad y dinamismo a las páginas web.

Contenidos:

Textos: El contenido principal de las páginas web.

Imágenes: Gráficos y fotografías.

Videos: Contenido audiovisual.

Audio: Archivos de sonido.

Datos estructurados: Información organizada en formatos como JSON o XML, utilizada por aplicaciones web.

9. Realiza un diagrama conceptual de composición de la web. ¿Cuáles son los Componentes de la www?

```
Usuario
  ↓
Navegador (Cliente)
  ↓ HTTP
  ↓
Servidor Web
  ↓
Bases de Datos
  ↓
Infraestructura de Red
```

10- ¿Cuál es la Arquitectura del sistema distribuido de documentos?

Componentes Clave

Clientes:

Interfaz de usuario: Permite a los usuarios interactuar con el sistema para buscar, crear, editar y compartir documentos.

Lógica de cliente: Realiza tareas locales como validación de datos y almacenamiento en caché.

Servidores:

Servidor de archivos: Almacena los documentos en un formato estructurado y proporciona acceso a los mismos.

Servidor de metadatos: Almacena información sobre los documentos, como títulos, autores, fechas de creación, etiquetas, etc., lo que permite realizar búsquedas y filtrado.

Servidor de índices: Crea índices para permitir búsquedas rápidas en grandes volúmenes de documentos.

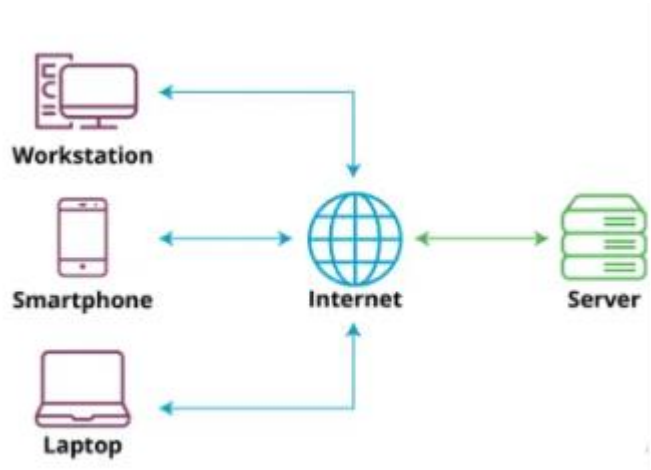
Servidor de autenticación y autorización: Gestiona el acceso a los documentos y controla los permisos de los usuarios.

Red:

Conecta los diferentes componentes del sistema y permite la comunicación entre ellos.

Base de datos:

Almacena los metadatos de los documentos y, en algunos casos, los documentos completos.



11. ¿Cómo funciona el sistema www?

¿Cómo se conecta todo?

Tu navegador (cliente): Es la herramienta que usas para solicitar información. Cuando escribes una

dirección web (URL) en la barra de direcciones, estás enviando una solicitud a un servidor.

Servidor: Es una computadora que almacena los archivos de las páginas web. Cuando recibe tu solicitud, busca el archivo correspondiente y te lo envía.

Protocolos: Son las reglas que gobiernan la comunicación entre tu navegador y el servidor. El protocolo principal es el HTTP (HyperText Transfer Protocol).

Proceso simplificado:

Solicitud: Tú escribes una URL en tu navegador.

Traducción: Tu navegador convierte esa URL en una dirección IP (un número que identifica un dispositivo en Internet).

Conexión: Se establece una conexión entre tu computadora y el servidor.

Respuesta: El servidor envía el código HTML de la página a tu navegador.

Visualización: Tu navegador interpreta el código HTML y te muestra la página en la pantalla.

Componentes clave:

HTML (HyperText Markup Language): Es el lenguaje utilizado para estructurar el contenido de una páginaK web.

CSS (Cascading Style Sheets): Define el estilo y la apariencia de una página web.

JavaScript: Agrega interactividad a las páginas web.

Tecnologías, Servidores y Clientes.

12- HTML (HyperText Markup Language) es el lenguaje estándar para crear páginas web. Fue desarrollado por Tim Berners-Lee en 1991 mientras trabajaba en el CERN, el laboratorio de física de partículas en Suiza. HTML se utiliza para estructurar contenido en la web, permitiendo la inserción de texto, imágenes, enlaces, y otros elementos multimedia.

13- ¿A qué se le llama página web estática? ¿y pagina dinámica? ¿y página activa? 14. ¿Qué es una URL? ¿qué función cumple en la Web? ¿Qué es un servidor de archivos basado en URL?

- **Página web estática**: Es una página web cuyo contenido no cambia en función de las interacciones del usuario. Cada vez que se accede a una página estática, el contenido mostrado es siempre el mismo, ya que está predefinido en el código HTML.

- **Página web dinámica**: Es una página web cuyo contenido puede cambiar según la interacción del usuario, la hora del día, el idioma o cualquier otra variable. Las páginas dinámicas suelen estar conectadas a bases de datos o scripts que generan el contenido en tiempo real.

- **Página web activa**: Este término a veces se usa para referirse a páginas que incluyen scripts (como JavaScript) que permiten interacciones directas sin necesidad de recargar la página. Sin embargo, no es un término técnico tan común como los anteriores.

- **URL (Uniform Resource Locator)**: Es la dirección específica que se utiliza para acceder a un recurso en la web, como una página web. Una URL indica la ubicación de un recurso y el método para recuperarlo.

- **Servidor de archivos basado en URL**: Es un servidor que proporciona acceso a archivos a través de URL. Esto permite a los usuarios descargar o ver archivos directamente desde un navegador web, utilizando una URL específica.

14 ¿Dónde y cómo se ejecuta el lenguaje HTML?

- Dónde: HTML se interpreta en el **navegador web** (como Chrome, Firefox, Safari, etc.) del dispositivo del usuario. No importa si estás usando una computadora, un teléfono móvil o una tablet; el navegador es el encargado de leer el archivo HTML.

- Cómo: Cuando un navegador carga una página web, descarga el archivo HTML desde un servidor web. Luego, el navegador interpreta el código HTML para estructurar y mostrar el contenido en la pantalla del usuario. El HTML define la estructura de la página, como los encabezados, párrafos, imágenes, enlaces, etc. El navegador también puede combinar HTML con otros lenguajes como CSS (para el diseño) y JavaScript (para la interactividad).

15- ¿Qué es un lenguaje script en la tecnología WWW?

Un lenguaje de script es una herramienta que permite crear páginas web dinámicas e interactivas. Esto significa que las páginas pueden cambiar y responder a las acciones del usuario, como hacer clic en un botón o llenar un formulario.

Los lenguajes de script más comunes son:

JavaScript: El rey de la web, se ejecuta directamente en el navegador.

PHP: Principalmente usado en el servidor para generar contenido dinámico.

Python y Ruby: También populares en el servidor, ofrecen frameworks poderosos.

En resumen, los lenguajes de script son esenciales para crear experiencias web modernas y personalizadas.

Plugin:

- **Definición:** Un plugin es una extensión o complemento que se añade a un software existente para ampliar sus funcionalidades. Es como agregar una nueva herramienta a un programa que ya tienes.
- **Función:** Los plugins permiten personalizar y adaptar un software a las necesidades específicas de un usuario o una aplicación. Por ejemplo, en WordPress, los plugins se utilizan para agregar nuevas características como formularios de contacto, galerías de imágenes, tiendas en línea, etc.
- **Características:**
    - **Modularidad:** Los plugins son independientes y se pueden activar o desactivar según sea necesario.
    - **Personalización:** Permiten adaptar el software a diferentes usos y preferencias.
    - **Desarrollo:** Pueden ser creados por desarrolladores independientes o por la comunidad de usuarios.

Cookie:

- **Definición:** Una cookie es un pequeño archivo de texto que un sitio web almacena en el dispositivo del usuario (ordenador, teléfono, tablet).
- **Función:** Las cookies sirven para recordar información sobre la visita del usuario, como sus preferencias, el contenido de su carrito de compras o el idioma que ha seleccionado. Esta información se utiliza para personalizar la experiencia del usuario y mejorar la funcionalidad del sitio web.
- **Características:**
    - **Temporales:** Algunas cookies se eliminan al cerrar el navegador, mientras que otras pueden permanecer durante más tiempo.
    - **Seguridad:** Las cookies no pueden ejecutar código y no pueden acceder a otros archivos de tu dispositivo.
    - **Privacidad:** Aunque las cookies son generalmente seguras, es importante ser consciente de cómo se utilizan y tener la opción de gestionarlas.

En resumen:

- **Los plugins añaden nuevas funcionalidades a un software.**
- **Las cookies almacenan información sobre la navegación del usuario.**

16- ¿Qué es JavaScript? ¿Dónde y quien ejecuta JavaScript? ¿Cómo nació JavaScript?

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web interactivas. Esto significa que, gracias a JavaScript, las páginas web pueden hacer mucho más que simplemente mostrar información estática. Pueden responder a tus acciones, como hacer clic en un botón, llenar un formulario o desplazarte por una página.

Imagina una página web como un lienzo en blanco. JavaScript es como el pincel que te permite pintar sobre ese lienzo, añadiendo color, movimiento y vida.

¿Dónde y quién ejecuta JavaScript?

- **Dónde:** JavaScript se ejecuta principalmente en el **navegador web** del usuario. Cuando visitas una página web con JavaScript, el navegador lee el código y lo ejecuta, creando la experiencia interactiva que ves en pantalla.
- **Quién:** El **motor JavaScript** del navegador es el encargado de interpretar y ejecutar el código JavaScript. Cada navegador tiene su propio motor JavaScript (por ejemplo, V8 en Chrome, SpiderMonkey en Firefox), pero todos cumplen la misma función básica.
- ¿Cómo nació JavaScript?
-

JavaScript fue creado por **Brendan Eich** en **1995**, en tan solo 10 días. Originalmente se llamó **Mocha**, luego **LiveScript**, y finalmente se adoptó el nombre de JavaScript para aprovechar la popularidad de Java en ese momento.

La idea detrás de JavaScript era crear un lenguaje de programación que permitiera a los desarrolladores web crear páginas web más dinámicas y atractivas. Antes de JavaScript, las páginas web eran bastante estáticas y aburridas.

¿Por qué se hizo tan popular?

- **Facilidad de uso:** JavaScript es relativamente fácil de aprender y usar, especialmente para aquellos con conocimientos básicos de programación.
- **Amplia compatibilidad:** Funciona en todos los navegadores modernos, lo que lo convierte en una opción versátil para cualquier desarrollador web.
- **Gran comunidad:** Tiene una comunidad de desarrolladores muy activa, lo que significa que hay muchos recursos, tutoriales y bibliotecas disponibles para aprender y utilizar JavaScript.

18- ¿Qué es un Applet de Java? ¿Cómo puede un Browser ejecutar código Java?

Un applet de Java era un pequeño programa escrito en el lenguaje de programación Java que podía incrustarse directamente en una página web HTML. Imagina una pequeña aplicación que se ejecutaba dentro de tu navegador, capaz de realizar tareas más allá de lo que permitía el HTML estático de aquellos tiempos.

Para que un navegador pudiera ejecutar un applet, necesitaba tener instalada una máquina virtual de Java (JVM). Esta JVM era como un intérprete que "entendía" el código Java y lo ejecutaba dentro del navegador.

El proceso era el siguiente:

Incrustación en HTML: Se insertaba una etiqueta especial <applet> en el código HTML de la página web, indicando la ubicación del applet y sus parámetros.

Descarga del applet: Cuando el navegador cargaba la página, descargaba el código del applet desde el servidor.

Ejecución por la JVM: La JVM del navegador tomaba el código del applet y lo ejecutaba en un entorno seguro y aislado.

La Tecnología Flash de Macromedia: Un Retrospecto

Adobe Flash, anteriormente conocido como Macromedia Flash, fue una plataforma multimedia que revolucionó la forma en que experimentábamos el contenido en línea a finales de los 90 y principios de los 2000.

¿Cómo funcionaba?

Creación de contenido:

Entorno de desarrollo: Los diseñadores utilizaban herramientas como Adobe Flash Professional para crear animaciones, gráficos vectoriales, interfaces de usuario y contenido interactivo.

Lenguaje de scripting: El contenido se programaba utilizando ActionScript, un lenguaje de scripting basado en ECMAScript (el predecesor de JavaScript), lo que permitía crear animaciones complejas, interacciones con el usuario y lógica de juego.

Archivos SWF:

Compilación: Una vez creado el contenido, se compilaba en un archivo con extensión SWF (Small Web Format). Este archivo contenía todo el código, los gráficos y los recursos necesarios para ejecutar la animación.

Formato vectorial: Los gráficos en Flash se almacenaban en formato vectorial, lo que permitía escalarlos sin pérdida de calidad.

Reproducción en el navegador:

Flash Player: Para visualizar el contenido Flash, los usuarios necesitaban tener instalado un plugin en su navegador llamado Flash Player. Este plugin era responsable de interpretar el archivo SWF y renderizarlo en la pantalla.

Entorno de ejecución: El Flash Player proporcionaba un entorno de ejecución aislado dentro del navegador, donde el contenido Flash podía ejecutarse de forma segura.

Características clave de Flash:

Animación vectorial: Permitía crear animaciones suaves y escalables.

Interactividad: Facilitaba la creación de interfaces de usuario interactivas y juegos.

Rendimiento: Ofrecía un buen rendimiento en la mayoría de las computadoras.

Alcance: Fue ampliamente adoptado por diseñadores web y desarrolladores de juegos.

¿Por qué Flash perdió popularidad?

Problemas de seguridad: Flash fue blanco de numerosos ataques de seguridad, lo que llevó a preocupaciones sobre la seguridad de los sistemas.

Rendimiento en dispositivos móviles: Flash no se optimizó bien para dispositivos móviles, lo que limitó su uso en este tipo de dispositivos.

Estándares web abiertos: El surgimiento de HTML5, CSS3 y JavaScript proporcionó alternativas más abiertas y flexibles para crear contenido interactivo en la web.

Fin del soporte: Adobe anunció en 2020 que dejaría de dar soporte a Flash, lo que llevó a su eventual desaparición.

En resumen, Flash fue una tecnología innovadora que transformó la web, pero con el tiempo quedó obsoleta debido a problemas de seguridad, rendimiento y la evolución de los estándares web. Hoy en día, las tecnologías web modernas ofrecen alternativas más seguras y flexibles para crear experiencias interactivas en línea.

19- ¿Para qué se utiliza la tecnología CGI? ¿Qué lenguajes de programación se pueden utilizar para los scripts CGI?

CGI (Common Gateway Interface) es una interfaz estándar que permite a un servidor web ejecutar programas externos, generalmente scripts, y devolver el resultado al navegador del cliente. En términos más simples, es el puente que conecta las páginas web estáticas con la capacidad de generar contenido dinámico.

Usos comunes de CGI:

Formularios: Procesar datos enviados a través de formularios HTML, como información de registro, pedidos en línea o encuestas.

Bases de datos: Interactuar con bases de datos para recuperar y mostrar información personalizada, como resultados de búsquedas o noticias.

Generación de contenido dinámico: Crear páginas web que se actualizan en tiempo real, como resultados deportivos, cotizaciones bursátiles o chat en vivo.

Procesamiento de imágenes: Realizar operaciones sobre imágenes, como redimensionar, aplicar filtros o crear miniaturas.

Autentificación: Verificar la identidad de los usuarios y controlar el acceso a áreas restringidas de un sitio web.

20- ¿Qué es un "form" HTML? ¿para que se utiliza?

Un **formulario HTML** es un elemento fundamental en las páginas web que permite a los usuarios introducir datos y enviarlos a un servidor para su procesamiento. Imagina un formulario de contacto en un sitio web: los campos para tu nombre, correo electrónico y mensaje son parte de un formulario.

En términos técnicos, un formulario HTML es una sección de una página web que contiene controles interactivos, como campos de texto, botones de radio, casillas de verificación y botones de envío. Estos controles permiten al usuario ingresar información que luego puede ser enviada a un servidor para su almacenamiento, procesamiento o cualquier otra acción.

¿Para qué se utiliza un formulario HTML?

Recopilación de datos: Se utilizan para recopilar información de los usuarios, como nombres, direcciones de correo electrónico, fechas de nacimiento, etc. Esto es esencial para realizar registros, suscripciones, encuestas y otros procesos que requieren la entrada del usuario.
Búsquedas: Los motores de búsqueda utilizan formularios para permitir a los usuarios introducir palabras clave y realizar búsquedas.
Compras en línea: Los formularios son la columna vertebral de las tiendas en línea, ya que permiten a los usuarios agregar productos a sus carritos de compra, completar información de facturación y envío, y realizar pagos.
Login y registro: Los formularios se utilizan para autenticar a los usuarios y permitirles acceder a áreas restringidas de un sitio web.
Envío de correos electrónicos: Los formularios pueden configurarse para enviar correos electrónicos automáticamente cuando un usuario envía un formulario, como en el caso de un formulario de contacto.

21- ¿Qué son y cómo funcionan las tecnologías ASP, PHP y JSP?

ASP, PHP y JSP: Los pilares de la web dinámica

ASP, PHP y JSP son tecnologías de servidor que permiten crear páginas web dinámicas. Estas tecnologías han sido fundamentales en el desarrollo de la web tal como la conocemos hoy, permitiendo la creación de sitios web interactivos y personalizados.

¿Cómo funcionan estas tecnologías?

El cliente envía una solicitud: Cuando un usuario introduce una dirección web en su navegador, se envía una solicitud al servidor web.

El servidor procesa la solicitud: El servidor recibe la solicitud y la pasa al programa correspondiente (ASP,PHP,JSP).

El programa genera contenido: El programa analiza la solicitud, accede a la base de datos si es necesario, y genera el código HTML que será enviado al navegador del cliente.

El cliente recibe el contenido: El navegador recibe el código HTML generado y lo muestra en la pantalla.

22 ¿Qué es DHTML? ¿y XHTML?

DHTML y XHTML: Evolución de las páginas web

DHTML (Dynamic HTML) y XHTML son términos que han sido fundamentales en la evolución de las páginas web, aunque hoy en día se utilizan con menos frecuencia debido a los avances en tecnologías web.

DHTML (Dynamic HTML)

¿Qué es? DHTML es un término general que se refiere a la combinación de HTML, CSS y JavaScript para crear páginas web dinámicas e interactivas. En otras palabras, es la forma de hacer que las páginas web "cobren vida" y respondan a las acciones del usuario.

23 XML: Un Lenguaje de Marcado Extensible

XML (eXtensible Markup Language) es un metalenguaje diseñado para almacenar y transportar datos de forma estructurada. A diferencia de HTML, que está diseñado específicamente para presentar información en un navegador web, XML es más flexible y permite definir tus propias etiquetas para representar cualquier tipo de datos.

¿Por qué es importante XML?

Estructura: XML proporciona una forma clara y concisa de organizar la información, lo que facilita su procesamiento por parte de computadoras y personas.

Flexibilidad: Al permitir definir tus propias etiquetas, XML se adapta a una amplia variedad de aplicaciones, desde la configuración de aplicaciones hasta el intercambio de datos entre diferentes sistemas.

Extensibilidad: Puedes crear tus propios esquemas XML para definir la estructura y el contenido de los documentos.

Independencia de la plataforma: XML es un estándar abierto, lo que significa que puede ser utilizado en cualquier plataforma y sistema operativo

24 ¿Que son los Servicios Web? De un ejemplo de aplicación utilizando Servicios WEB

Un servicio web es una tecnología que permite a diferentes aplicaciones comunicarse e intercambiar datos a través de una red, generalmente Internet. Es como una interfaz o punto de acceso que permite a una aplicación solicitar un servicio a otra aplicación, sin importar el lenguaje de programación o la plataforma en la que estén desarrolladas.

¿Cómo funcionan?

Imagina que tienes una tienda en línea y quieres integrar las funciones de pago con un procesador de pagos externo. En lugar de desarrollar todo el sistema de pago tú mismo, puedes utilizar un servicio web proporcionado

por el procesador de pagos. Tu aplicación enviará una solicitud al servicio web con los datos de la compra (productos, cantidades, información del cliente), y el servicio web procesará el pago y te devolverá una respuesta indicando si la transacción fue exitosa o no.

URL´s, Dominios y Direcciones

25 ¿Qué es una URL? ¿qué función cumple en la web? ¿Cuál es la estructura de una URL?

Una **URL** (Uniform Resource Locator, o Localizador Uniforme de Recursos en español) es como la dirección de una casa en Internet. Es una secuencia única de caracteres que identifica de manera precisa un recurso específico en la web. Este recurso puede ser una página web, una imagen, un video, un documento o cualquier otro tipo de archivo disponible en línea.

¿Qué función cumple en la web?

La principal función de una URL es **permitir que los usuarios y las aplicaciones encuentren y accedan a recursos específicos en Internet**. Al escribir una URL en la barra de direcciones de un navegador, estás indicando al navegador qué recurso deseas ver.

26 ¿Qué es una URL? ¿qué función cumple en la web? ¿Cuál es la estructura de una URL?

Estructura de una URL

Una URL típica se compone de varias partes:

- **Protocolo:** Indica el método utilizado para acceder al recurso. Los protocolos más comunes son HTTP (HyperText Transfer Protocol) para páginas web y HTTPS (HyperText Transfer Protocol Secure) para páginas web seguras.
- **Dominio:** Identifica el servidor web donde se encuentra el recurso. Por ejemplo, "[se quitó una URL no válida]".
- **Ruta:** Especifica la ubicación exacta del recurso dentro del servidor. Puede incluir subdirectorios y archivos.
- **Parámetros:** Opcionalmente, se pueden incluir parámetros para pasar información adicional al servidor.

Ejemplo:

```
https://www.ejemplo.com/articulos/como-crear-una-pagina-web.html
```

https: Indica que se utiliza una conexión segura.

www.ejemplo.com: Es el dominio del sitio web.

artículos/como-crear-una-pagina-web.html: Es la ruta al archivo HTML que contiene el artículo.

27 ¿Qué es un servidor de archivos basado en URL?

Un servidor de archivos basado en URL es un sistema que te permite almacenar y compartir archivos a través del Internet utilizando una dirección web o URL. En lugar de acceder a tus archivos a través de una red local o un cliente FTP tradicional, puedes hacerlo directamente desde cualquier dispositivo conectado a Internet, simplemente escribiendo la URL correspondiente en la barra de direcciones de tu navegador.

28 ¿Cuál es el formato de una URL en IPv6?

La estructura básica de una URL (Uniform Resource Locator) permanece igual tanto en IPv4 como en IPv6. La principal diferencia radica en la dirección IP que se utiliza para identificar el servidor.

Estructura General de una URL

Una URL típica se compone de:

Protocolo: Indica el método utilizado para acceder al recurso (http, https, ftp, etc.).

Dominio: Identifica el servidor donde se encuentra el recurso.

Ruta: Especifica la ubicación exacta del recurso dentro del servidor.

Parámetros: Opcionalmente, se pueden incluir parámetros para pasar información adicional al servidor.

Ejemplo:

`https://www.ejemplo.com/articulos/como-crear-una-pagina-web.html`

URL con Dirección IPv6

Cuando se utiliza una dirección IPv6 en lugar de un nombre de dominio, la URL tendrá un aspecto similar a este:

`http://[2001:db8:85a3:0042:1000:8a2e:0370:7334]/index.html`

Diferencias clave:

Corchetes: La dirección IPv6 siempre va encerrada entre corchetes `[]` para diferenciarla del nombre de dominio y otros componentes de la URL.

Formato de la dirección IPv6: La dirección IPv6 consta de ocho grupos de cuatro dígitos hexadecimales separados por dos puntos.

Compresión: Para simplificar la escritura, se pueden omitir grupos de ceros consecutivos. Por ejemplo, `2001:db8:85a3:0000:0000:8a2e:0370:7334` se puede escribir como `2001:db8:85a3::8a2e:0370:7334`.

Ejemplo con compresión:

`http://[2001:db8:85a3::8a2e:0370:7334]/index.html`

29 ¿Qué es un Dominio www? ¿Qué relación tiene con una URL?
El Dominio www: Tu dirección en Internet

Imagina Internet como una ciudad gigante. Cada casa en esta ciudad tiene una dirección única que te permite encontrarla. En el mundo digital, esa dirección es el **dominio**.

Un **dominio** es el nombre único que identifica a un sitio web en Internet. Es la parte de la URL que sigue al protocolo (http:// o https://) y que precede a la ruta. Por ejemplo, en la URL https://www.ejemplo.com, "www.ejemplo.com" es el dominio.

www es un subdominio común, que significa "World Wide Web". Se utiliza para indicar que estás accediendo a la parte pública de un sitio web. Sin embargo, no es obligatorio. Puedes tener otros subdominios como "blog", "tienda" o "correo".

Relación entre Dominio y URL

La relación entre un dominio y una URL es similar a la relación entre una dirección y una habitación dentro de una casa.

Dominio: Es la dirección general de la casa (el sitio web).

URL: Es la dirección específica de una habitación dentro de la casa (una página web en particular).

Por ejemplo:

https://www.ejemplo.com**:** Esta URL te lleva a la página principal del sitio web "ejemplo.com".

[se quitó una URL no válida]: Esta URL te lleva a la página de contacto del mismo sitio web.

En resumen, el dominio es parte de la URL y proporciona la dirección general del sitio web. La URL completa, incluyendo la ruta, te lleva a un recurso específico dentro del sitio web.

30 ¿Cuál es la función del DNS en la WEB?

El DNS (Domain Name System o Sistema de Nombres de Dominio) es como un traductor universal que nos permite navegar por Internet de forma sencilla. Imagina que en lugar de recordar largas secuencias de números (direcciones IP), pudiéramos recordar nombres fáciles como "[se quitó una URL no válida]" o "facebook.com". ¡Eso es exactamente lo que hace el DNS!

Protocolos utilizados en la www

31 **HTML (HyperText Markup Language)** y **HTTP (HyperText Transfer Protocol)** son dos tecnologías fundamentales que trabajan en conjunto para permitirnos navegar por la web de manera interactiva. Aunque son conceptos distintos, están estrechamente relacionados y desempeñan roles complementarios en la creación y visualización de páginas web.

HTML: El contenido de la página

Estructura: HTML es un lenguaje de marcado utilizado para estructurar el contenido de una página web. Define la apariencia y la organización del texto, imágenes, enlaces y otros elementos que componen una página.

Etiquetas: Utiliza etiquetas (como <h1>, <p>, <img>) para indicar cómo debe interpretarse el contenido por el navegador.

Contenido estático: Principalmente se encarga de la presentación visual de la información.

HTTP: El protocolo de comunicación

Transmisión: HTTP es un protocolo de comunicación que se utiliza para enviar y recibir datos entre un servidor web y un navegador.

Peticiones y respuestas: Cuando ingresas una URL en tu navegador, estás enviando una petición HTTP al servidor. El servidor procesa la petición y envía una respuesta, que incluye el código HTML de la página solicitada.

Interacción: Permite la interacción entre el usuario y el servidor, por ejemplo, al enviar un formulario o hacer clic en un enlace.

¿Cómo se relacionan HTML y HTTP?

HTML viaja a través de HTTP: Cuando solicitas una página web, el código HTML de esa página se envía desde el servidor a tu navegador a través de HTTP.

HTTP interpreta las etiquetas HTML: El navegador interpreta las etiquetas HTML y construye la página web en la pantalla.

HTTP permite la actualización dinámica: HTTP permite que las páginas web se actualicen dinámicamente sin recargar toda la página, lo que hace posible las aplicaciones web interactivas.

En resumen, HTML proporciona el contenido y la estructura de una página web, mientras que HTTP es el mecanismo que permite que ese contenido se transmita desde un servidor web a un navegador.

32 ¿Cuáles son las versiones de HTTP y cuales sus diferencias?

HTTP/0.9

La versión más básica: Solo permitía solicitudes GET para obtener un recurso.

Sin encabezados: No incluía información adicional en las solicitudes o respuestas

HTTP/1.0

Introducción de encabezados: Permitió agregar información adicional a las solicitudes y respuestas, como el tipo de contenido y el código de estado.

Conexiones persistentes: Introdujo la idea de mantener una conexión abierta para múltiples solicitudes, mejorando la eficiencia.

Limitaciones: Sufría de problemas de rendimiento como el bloqueo de cabeza (head-of-line blocking), donde una solicitud lenta podía bloquear todas las siguientes.

HTTP/1.1

Versión estándar: Se convirtió en el estándar de facto durante muchos años.**Mejoras en las conexiones persistentes:** Introdujo conexiones persistentes por defecto y mecanismos para cerrar conexiones inactivas.

Pipelining: Permitió enviar múltiples solicitudes sin esperar una respuesta completa para cada una, mejorando el rendimiento.

Cacheado: Incluyó mecanismos para el almacenamiento en caché de recursos, reduciendo la cantidad de datos que deben transferirse.

HTTP/2

Mayor rendimiento: Introdujo características como multiplexación, compresión de encabezados y servidor push para mejorar significativamente el rendimiento de las páginas web.

Binario: A diferencia de las versiones anteriores que eran textuales, HTTP/2 utiliza un formato binario más eficiente.**Única conexión:** Utiliza una única conexión TCP para múltiples recursos, reduciendo la sobrecarga.

HTTP/3 (QUIC)

Basado en UDP: Utiliza el protocolo UDP en lugar de TCP, lo que lo hace más resistente a la congestión de la red y reduce la latencia.

Multiplexación y corrección de errores: Ofrece características similares a HTTP/2, pero con un mejor rendimiento en redes móviles y conexiones con alta latencia.**Cifrado por defecto:** Incluye cifrado de extremo a extremo por defecto, mejorando la seguridad.

33 ¿Qué es MIME? ¿Cómo funciona MIME?

MIME utiliza **tipos MIME** para identificar los diferentes tipos de datos. Un tipo MIME consta de dos partes separadas por una barra oblicua:

Tipo principal: Indica la categoría general de datos (por ejemplo, text, image, audio).

Subtipo: Especifica el formato exacto del dato (por ejemplo, plain, jpeg, mp3)

Ejemplo:

`text/html`: Indica un documento HTML.

`image/jpeg`: Indica una imagen en formato JPEG.

`application/pdf`: Indica un documento PDF.

¿Dónde se utiliza MIME?

MIME se utiliza en muchos protocolos de Internet, pero es especialmente importante en:

Correo electrónico: Para enviar archivos adjuntos de diferentes tipos.

HTTP: Para indicar el tipo de contenido que se está transfiriendo en una solicitud HTTP.

FTP: Para identificar los tipos de archivos que se están transfiriendo.

34. Explique cómo es una solicitud HTTP.

La Solicitud HTTP: El Pedido al Servidor Web

Una solicitud HTTP es como un pedido que haces a un restaurante. Tú eres el cliente (tu navegador), el restaurante es el servidor web y el menú es el contenido disponible en ese servidor

Estructura de una solicitud HTTP:

Una solicitud HTTP se compone de tres partes principales

Línea de solicitud:

Método HTTP: Indica la acción que deseas realizar (GET, POST, PUT, DELETE, etc.). Por ejemplo, GET se utiliza para solicitar un recurso, POST para enviar datos a un servidor.

Recurso: Especifica el recurso que deseas acceder (una página web, una imagen, etc.).

Versión HTTP: Indica la versión del protocolo HTTP que estás utilizando.

35. ¿Cuáles son los métodos de HTTP 1? Enumere y describa el funcionamiento de cada uno

Los métodos HTTP son como verbos que le indican al servidor qué acción deseas realizar sobre un recurso. Cada método tiene una función específica y se utiliza en diferentes escenarios. Métodos HTTP 1 más comunes:

GET:

Función: Se utiliza para solicitar un recurso (una página web, una imagen, etc.).

Ejemplo: Cuando escribes una URL en tu navegador y presionas Enter, estás realizando una solicitud GET.

Características:

Es idempotente: realizar la misma solicitud múltiples veces produce el mismo resultado.

No debería utilizarse para enviar datos sensibles, ya que estos pueden quedar expuestos en la URL o en los logs del servidor.

POST:

Función: Se utiliza para enviar datos a un servidor, típicamente para crear un nuevo recurso o actualizar un recurso existente.

Ejemplo: Al enviar un formulario, los datos del formulario se envían al servidor mediante una solicitud POST.

Características:

No es idempotente: realizar la misma solicitud múltiples veces puede crear múltiples recursos.

Se utiliza para enviar datos sensibles, ya que los datos se incluyen en el cuerpo de la solicitud y no en la URL.

PUT:

Función: Se utiliza para actualizar un recurso existente.

Ejemplo: Al editar un documento en una aplicación web, se utiliza una solicitud PUT para enviar los cambios al servidor.

Características:

Es idempotente: realizar la misma solicitud múltiples veces tiene el mismo efecto que realizarla una sola vez.

Se utiliza para reemplazar completamente un recurso existente.

DELETE:

Función: Se utiliza para eliminar un recurso.

Ejemplo: Al eliminar un archivo en una aplicación en la nube, se utiliza una solicitud DELETE.

Características:

No es idempotente: realizar la misma solicitud múltiples veces puede eliminar el recurso varias veces (aunque el resultado final suele ser el mismo).

HEAD:

Función: Es similar a GET, pero solo devuelve las cabeceras de la respuesta, sin el cuerpo.

Ejemplo: Se utiliza para obtener información sobre un recurso sin descargar todo su contenido.

OPTIONS:

Función: Se utiliza para obtener información sobre las opciones de comunicación que un servidor soporta.

- **Ejemplo:** Se utiliza para descubrir los métodos HTTP que un servidor permite para un recurso específico.

Otros métodos menos comunes:

- **CONNECT:** Se utiliza para establecer un túnel a través de un servidor proxy.
- **TRACE:** Devuelve la solicitud recibida, lo que puede ser útil para la depuración.
- **PATCH:** Se utiliza para aplicar modificaciones parciales a un recurso.