**UNIVERSIDAD NACIONAL DE INGENIERÍA**

**Áreas de Conocimiento: Tecnología de la Información y Comunicación**

**Asignatura: Aplicaciones Gráficas y Multimedia**

**UNIDAD III: DISEÑO WEB FRONT-END**

**Actividad 6**: Teoría - Explicar las sintaxis de marcado.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre:** | **Apellidos:** | | **Carnet:** 00000 |
| **Carrera:** Ingeniería de Sistemas | **Grupo:** 000 | **Grupo de trabajo:** 00 | **Fecha:** 00-00-00 |

**OBJETIVO GENERAL DE LA ASIGNATURA**

Desarrollar la interfaz web de sitios, sistemas y aplicaciones para organizaciones, por medio de lenguajes de marcado, de estilo y de comportamiento basado en los documentos y directrices que contemplan principios de accesibilidad y adaptabilidad, mediante la aplicación de técnicas, procedimientos y herramientas front-end.

**UNIDAD III: OBJETIVOS PARTICULARES**

* Explicar las sintaxis de los lenguajes de marcado, presentación e interacción para la construcción de sitios web interactivos.
* Aplicar los lenguajes de marcado, presentación e interacción para la construcción de sitios web interactivos.
* Valorar la aplicación adecuada de los lenguajes utilizados en el diseño de sitios web interactivos.

**Actividad 6: Analizar la Unidad III: Diseño web front-end**

**Contestar las siguientes preguntas explicando con sus propias palabras y ejemplifique (Explicación breve del software que utilizo):**

**Puede consultar con la IA - ChatGPT (Recuerde personalizar los ejemplos)**

Contestar las siguientes preguntas:

Diseño Web

1. ¿Cuál es el Funcionamiento de la Web?
2. ¿Cuáles son las Métricas de calidad para el diseño de sitios Web?
3. ¿Qué es La Web semántica?
4. ¿Cuál es la Diferencias entre front-end y back-end?
5. ¿Cuáles son los Servidores y navegadores Web?
6. ¿Cuál son los Entornos de trabajo para el diseño Web?
7. ¿Cuáles son los Dominios y Web hosting?
8. ¿Qué es Mock-ups y prototipos?

Estructura de las páginas Web

1. ¿Qué es Estándares W3C en las páginas Web?
2. ¿Qué es HTML y XML?
3. ¿Cuáles son las Etiquetas HTML, HEAD y BODY?
4. ¿Cuáles son los Textos y comentarios las páginas Web?
5. ¿Cuáles son los Enlaces en las páginas Web?
6. ¿Cuáles son las Imágenes en las páginas Web?
7. ¿Cuáles son los Videos en las páginas Web?
8. ¿Cuáles son las Listas en las páginas Web?
9. ¿Cuáles son las Tablas en las páginas Web?
10. ¿Cuáles son los Formularios en las páginas Web?
11. ¿Cuáles son los Elementos de HTML5?
12. ¿Cuáles son las APIs de HTML5?

Presentación de las páginas Web

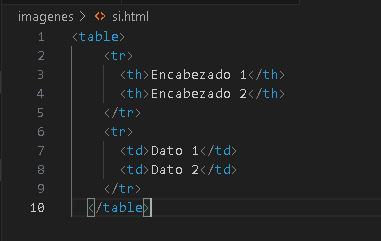
1. ¿Cuáles son las Hojas de estilo en cascada en las páginas Web?
2. ¿Qué es Vinculación de hojas de estilo en las páginas Web?
3. ¿Cuáles son las Reglas y comentarios en CSS?
4. ¿Cuáles son los Selectores en las páginas Web?
5. ¿Qué es Pseudo-selectores en las páginas Web?
6. ¿Qué es Herencia y especificidad en las páginas Web?
7. ¿Cuáles son los Colores y fondos en las páginas Web?
8. ¿Qué es Tipografía en las páginas Web?
9. ¿Qué es El modelo de caja CSS y Flexbox en las páginas Web?
10. ¿Qué es La cuadrícula CSS en las páginas Web?
11. ¿Qué es Pseudo-clases en las páginas Web?
12. ¿Cuáles son las Animaciones con CSS en las páginas Web?

Interactividad de las páginas Web

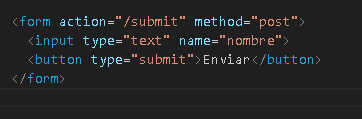
1. ¿Qué Lenguaje Javascript en las páginas Web?
2. ¿Cuáles son las Variables y tipos de datos en las páginas Web?
3. ¿Cuáles son las Estructuras de control y ciclos repetitivos en las páginas Web?
4. ¿Cuáles son las Funciones en las páginas Web?
5. ¿Cuáles son los Arreglos en las páginas Web?
6. ¿Cuáles son los Objetos en las páginas Web?
7. ¿Qué es Manipulación del DOM en las páginas Web?
8. ¿Qué es Orientación a objetos con Javascript en las páginas Web?
9. ¿Qué es Ajax en las páginas Web?
10. **Funcionamiento de la Web** La web funciona mediante la interacción de varios componentes esenciales: servidores web, navegadores, protocolos de comunicación (HTTP/HTTPS) y contenido web. Los servidores web almacenan sitios y envían contenido a los navegadores a través de HTTP/HTTPS. Los navegadores interpretan y muestran el contenido al usuario final. La interacción comienza con una solicitud del navegador, que se envía al servidor, el cual responde con los recursos solicitados (páginas HTML, CSS, JavaScript, imágenes, etc.).
11. **Métricas de calidad para el diseño de sitios Web** Las métricas de calidad incluyen:
    * **Usabilidad**: Facilidad de uso y navegación.
    * **Accesibilidad**: Acceso para personas con discapacidades.
    * **Rendimiento**: Tiempo de carga y velocidad de respuesta.
    * **Compatibilidad**: Funcionamiento en diferentes dispositivos y navegadores.
    * **SEO**: Optimización para motores de búsqueda.
    * **Seguridad**: Protección contra vulnerabilidades y ataques.
12. **La Web semántica** La Web semántica es una extensión de la web actual que permite a los datos ser compartidos y reutilizados a través de diferentes aplicaciones, empresas y comunidades. Utiliza tecnologías como RDF (Resource Description Framework), OWL (Web Ontology Language) y SPARQL (SPARQL Protocol and RDF Query Language) para facilitar la integración y el intercambio de datos con significado.
13. **Diferencias entre front-end y back-end**
    * **Front-end**: Parte de la web que interactúa directamente con el usuario. Incluye HTML, CSS y JavaScript, responsables del diseño, la estructura y la interactividad de una página web.
    * **Back-end**: Parte del servidor que maneja la lógica, las bases de datos y la autenticación. Incluye lenguajes y tecnologías como PHP, Python, Ruby, Node.js, bases de datos (SQL, MongoDB), y frameworks de servidores.
14. **Servidores y navegadores Web**
    * **Servidores web**: Apache, Nginx, Microsoft IIS, LiteSpeed.
    * **Navegadores web**: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, Opera.
15. **Entornos de trabajo para el diseño Web** Incluyen herramientas y software que facilitan el diseño y desarrollo de sitios web:
    * **IDE y editores de texto**: Visual Studio Code, Sublime Text, Atom.
    * **Sistemas de control de versiones**: Git y plataformas como GitHub.
    * **Frameworks y bibliotecas**: Bootstrap, Angular, React, Vue.js.
16. **Dominios y Web hosting**
    * **Dominios**: Dirección única que identifica un sitio web en Internet (ej. [www.ejemplo.com](http://www.ejemplo.com)).
    * **Web hosting**: Servicio que permite a las organizaciones e individuos publicar un sitio web o una página web en Internet. Proveedores populares incluyen Bluehost, HostGator, SiteGround.
17. **Mock-ups y prototipos**
    * **Mock-ups**: Representaciones estáticas de un diseño de página web, mostrando la apariencia final.
    * **Prototipos**: Versiones interactivas de un diseño que simulan la funcionalidad y la navegación del sitio web.

**Estructura de las páginas Web**

1. **Estándares W3C en las páginas Web** El W3C (World Wide Web Consortium) establece estándares para asegurar la accesibilidad, la internacionalización y la interoperabilidad de las tecnologías web. Incluyen especificaciones para HTML, CSS, XML, y otros lenguajes web.
2. **HTML y XML**
   * **HTML (HyperText Markup Language)**: Lenguaje de marcado estándar para crear páginas web.
   * **XML (eXtensible Markup Language)**: Lenguaje de marcado que define reglas para la codificación de documentos en un formato que es legible tanto por humanos como por máquinas.
3. **Etiquetas HTML, HEAD y BODY**
   * **HTML**: Contenedor principal que define el documento como un documento HTML.
   * **HEAD**: Contiene metadatos, enlaces a hojas de estilo, scripts y otras configuraciones.
   * **BODY**: Contiene el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.
4. **Textos y comentarios en las páginas Web**
   * **Textos**: Se definen con etiquetas como <p>, <h1>, <span>, <strong>, etc.
   * **Comentarios**: Añaden notas que no se muestran al usuario y se definen con <!-- comentario -->.
5. **Enlaces en las páginas Web** Los enlaces permiten la navegación entre páginas web y se definen con la etiqueta <a>. Ejemplo: <a href="https://www.ejemplo.com">Enlace</a>.
6. **Imágenes en las páginas Web** Las imágenes se insertan con la etiqueta <img>, que requiere atributos como src (fuente de la imagen) y alt (texto alternativo). Ejemplo: <img src="imagen.jpg" alt="Descripción de la imagen">.
7. **Videos en las páginas Web** Los videos se integran utilizando la etiqueta <video>, con atributos como src, controls, autoplay, y loop. Ejemplo: <video src="video.mp4" controls></video>.
8. **Listas en las páginas Web**
   * **Listas ordenadas**: Utilizan la etiqueta <ol>.
   * **Listas desordenadas**: Utilizan la etiqueta <ul>.
   * **Elementos de lista**: Utilizan la etiqueta <li> dentro de <ol> o <ul>.
9. **Tablas en las páginas Web** Se crean utilizando etiquetas como <table>, <tr> (fila de tabla), <td> (celda de tabla) y <th> (encabezado de celda). Ejemplo:



1. **Formularios en las páginas Web** Los formularios permiten a los usuarios enviar datos a un servidor y se definen con la etiqueta <form>. Contienen elementos de entrada como <input>, <textarea>, <select>, y botones <button>. Ejemplo:



1. **Elementos de HTML5** HTML5 introdujo nuevos elementos que mejoran la semántica y la funcionalidad de las páginas web. Ejemplos incluyen:

<header>: Define la cabecera de una sección o página.

<footer>: Define el pie de una sección o página.

<article>: Representa un contenido independiente que podría ser distribuido por separado, como un artículo de noticias.

<section>: Agrupa contenido temático.

<nav>: Define un conjunto de enlaces de navegación.

<figure> y <figcaption>: Para contener imágenes y sus leyendas asociadas.

<aside>: Contenido relacionado que se coloca al margen del contenido principal.

1. **APIs de HTML5** HTML5 introdujo varias APIs que permiten funcionalidades avanzadas en las páginas web:

**Geolocation API**: Permite obtener la ubicación geográfica del usuario.

**Web Storage API**: Proporciona almacenamiento local y de sesión en el navegador.

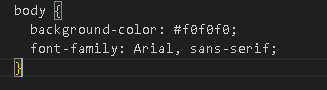
**Canvas API**: Permite el dibujo y la manipulación de gráficos y animaciones.

**WebSockets API**: Permite comunicación bidireccional en tiempo real entre el cliente y el servidor.

**Drag and Drop API**: Facilita la funcionalidad de arrastrar y soltar elementos en la página web.

**Presentación de las páginas Web**

1. **Hojas de estilo en cascada en las páginas Web** CSS (Cascading Style Sheets) es un lenguaje utilizado para describir la presentación de documentos HTML. CSS permite separar el contenido (HTML) de la presentación, facilitando el mantenimiento y la consistencia del diseño. Ejemplo de una regla CSS:

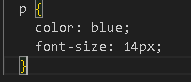


1. **Vinculación de hojas de estilo en las páginas Web** Las hojas de estilo externas se vinculan al documento HTML utilizando la etiqueta <link> en la sección <head>. Esto permite la reutilización de estilos a través de múltiples páginas. Ejemplo:



**Reglas y comentarios en CSS**

1. **Reglas**: Definen cómo se deben presentar los elementos HTML. Cada regla CSS consta de un selector y un bloque de declaraciones. Ejemplo:



**Comentarios**: Se utilizan para añadir notas o explicaciones en el código CSS y se ignoran durante la ejecución. Se escriben entre /\* comentario \*/.

1. **Selectores en las páginas Web** Los selectores identifican los elementos HTML a los que se aplican las reglas de estilo. Tipos comunes de selectores incluyen:

**Selectores de tipo**: Seleccionan todos los elementos de un tipo específico. Ejemplo: p { color: red; }

**Selectores de clase**: Seleccionan elementos con una clase específica. Ejemplo: .clase { color: green; }

**Selectores de ID**: Seleccionan un elemento con un ID específico. Ejemplo: #id { color: blue; }

**Selectores de atributo**: Seleccionan elementos basados en un atributo y su valor. Ejemplo: input[type="text"] { border: 1px solid #ccc; }

1. **Pseudo-selectores en las páginas Web** Los pseudo-selectores permiten aplicar estilos a elementos en estados específicos o basados en su posición en el DOM. Ejemplos:

:hover: Aplica estilos cuando el usuario coloca el cursor sobre un elemento.

:first-child: Selecciona el primer hijo de su elemento padre.

:nth-child(n): Selecciona el enésimo hijo de su elemento padre.

1. **Herencia y especificidad en las páginas Web**

**Herencia**: Algunas propiedades CSS se heredan automáticamente de los elementos padres a sus hijos, como color y font-family.

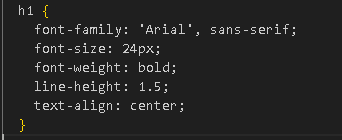
**Especificidad**: Determina qué reglas CSS se aplican cuando múltiples reglas coinciden con un elemento. La especificidad se calcula basado en el tipo de selectores usados: IDs tienen mayor peso que clases, que a su vez tienen mayor peso que los selectores de tipo.

1. **Colores y fondos en las páginas Web**

**Colores**: Los colores se pueden definir usando nombres de colores (red), valores hexadecimales (#ff0000), valores RGB (rgb(255, 0, 0)), y valores HSL (hsl(0, 100%, 50%)).

**Fondos**: La propiedad background permite definir el color de fondo, imágenes de fondo, y configuraciones como repetición (background-repeat), posición (background-position), y tamaño (background-size).

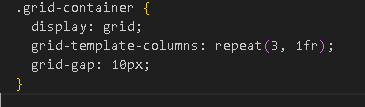
1. **Tipografía en las páginas Web** La tipografía se configura utilizando propiedades CSS como font-family (para definir la fuente), font-size (tamaño de la fuente), font-weight (grosor de la fuente), line-height (altura de línea), y text-align (alineación del texto). Ejemplo:



1. **El modelo de caja CSS y Flexbox en las páginas Web**

* **Modelo de caja**: Todos los elementos HTML se representan como cajas que constan de cuatro áreas: contenido, padding (relleno), border (borde), y margin (margen). El modelo de caja define cómo se calculan las dimensiones y el espacio de los elementos.
* **Flexbox**: Sistema de diseño unidimensional que permite distribuir y alinear elementos dentro de un contenedor flexible. Ejemplo:

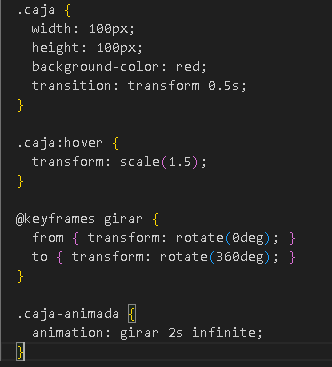
1. **La cuadrícula CSS en las páginas Web** CSS Grid Layout es un sistema de diseño bidimensional que permite crear disposiciones complejas de elementos en filas y columnas. Proporciona una forma poderosa de crear diseños responsivos y estructurados. Ejemplo:



1. **Pseudo-clases en las páginas Web** Las pseudo-clases seleccionan elementos en estados específicos. Ejemplos:

* :hover: Aplica estilos cuando el usuario coloca el cursor sobre un elemento.
* :active: Aplica estilos cuando un elemento está siendo activado por el usuario, como al hacer clic en un enlace.
* :focus: Aplica estilos cuando un elemento, como un campo de entrada, está enfocado.

1. **Animaciones con CSS en las páginas Web** Las animaciones en CSS permiten crear efectos de transición y movimiento utilizando propiedades como transition, transform, y @keyframes. Ejemplo de una animación básica:

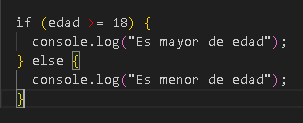


1.  **Lenguaje Javascript en las páginas Web** JavaScript es un lenguaje de programación interpretado que permite crear interactividad dinámica en las páginas web. Es esencial para el desarrollo de aplicaciones web modernas, ya que permite manipular el DOM, gestionar eventos, validar formularios, y comunicarse con servidores mediante Ajax. JavaScript se puede usar tanto en el lado del cliente (front-end) como en el lado del servidor (back-end) con plataformas como Node.js.
2. **Variables y tipos de datos en las páginas Web**

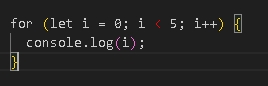
**Variables**: Son contenedores para almacenar datos. Se declaran utilizando var, let, o const. Ejemplo:

1. **Estructuras de control y ciclos repetitivos en las páginas Web**

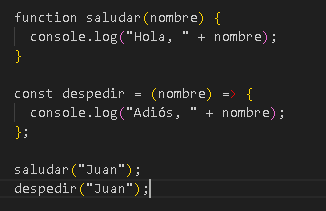
**Estructuras de control**: Permiten la ejecución condicional del código. Ejemplos incluyen if, else if, else, y switch. Ejemplo:



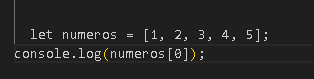
**Ciclos repetitivos**: Permiten la ejecución repetida de bloques de código. Ejemplos incluyen for, while, do...while, for...in, y for...of. Ejemplo:



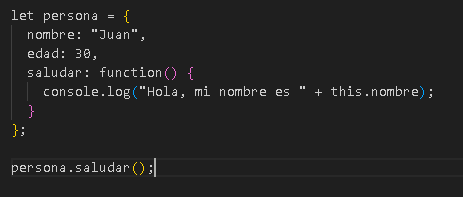
1. **Funciones en las páginas Web** Las funciones son bloques de código reutilizables que realizan una tarea específica. Se pueden declarar utilizando la palabra clave function o como funciones flecha (introducidas en ES6). Ejemplo:



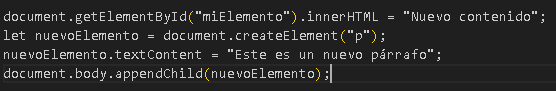
1. **Arreglos en las páginas Web** Los arreglos son colecciones ordenadas de datos. Se definen utilizando corchetes y pueden contener cualquier tipo de dato. Ejemplo:



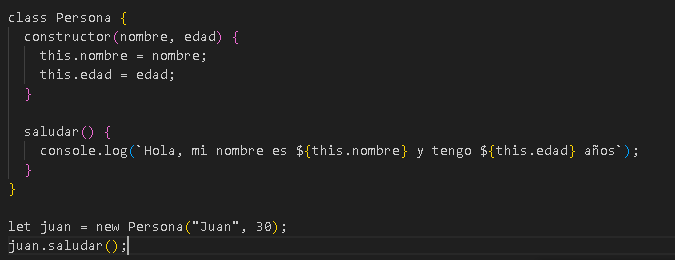
1. **Objetos en las páginas Web** Los objetos son colecciones de pares clave-valor que representan entidades. Los objetos se definen utilizando llaves {} y pueden contener propiedades y métodos. Ejemplo:



1. **Manipulación del DOM en las páginas Web** El DOM (Document Object Model) es una representación estructurada del documento HTML. JavaScript permite manipular el DOM para cambiar el contenido y el estilo de los elementos en tiempo real. Métodos comunes incluyen getElementById, querySelector, createElement, y appendChild. Ejemplo:



1. **Orientación a objetos con Javascript en las páginas Web** JavaScript soporta programación orientada a objetos mediante el uso de prototipos y clases (introducidas en ES6). Las clases permiten definir objetos y herencia de manera más estructurada. Ejemplo:



1. **Ajax en las páginas Web** Ajax (Asynchronous JavaScript and XML) permite la actualización de partes de una página web sin recargar la página completa. Utiliza objetos XMLHttpRequest o la API Fetch para comunicarse con el servidor de manera asíncrona. Ejemplo con Fetch:

