# HTML (HyperText Markup Language):

Sintaxis de HTML: Comprender la estructura básica de un documento HTML, incluyendo elementos, etiquetas y atributos.

Semántica HTML: Conocer el propósito y uso adecuado de elementos semánticos como <header>, <footer>, <nav>, <article>, etc.

## Estructura Básica:

Etiquetas HTML básicas: <html>, <head>, <body>.

Etiquetas semánticas: <header>, <nav>, <section>, <article>, <footer>.

Etiquetas de texto: <p>, <h1> a <h6>, <span>.

Etiquetas de lista: <ul>, <ol>, <li>.

Etiquetas de enlace: <a> y atributos href, target, rel.

Formularios HTML: Saber cómo crear y validar formularios utilizando elementos como <form>, <input>, <select>, <textarea>, etc.

Etiquetas de formulario: <form>, <input>, <textarea>, <select>, <button>.

Atributos de formulario: name, id, type, value, required, placeholder.

Multimedia: Conocer cómo incorporar imágenes, audio y video en documentos HTML utilizando elementos como <img>, <audio>, <video>, etc.

Etiquetas para contenido multimedia: <img>, <audio>, <video>.

Atributos de etiquetas multimedia: src, alt, controls.

## Tablas:

Creación de tablas con <table>, <tr>, <td>, <th>.

Atributos de tablas: colspan, rowspan, scope.

Estilo y diseño: Familiarizarte con CSS para dar estilo a tus documentos HTML y crear diseños atractivos.

Accesibilidad: Entender los principios de accesibilidad web y cómo implementarlos correctamente en HTML para hacer tus sitios accesibles para todos los usuarios.

Enlaces y navegación: Saber cómo crear enlaces internos y externos, así como menús de navegación utilizando <a> y otros elementos relacionados.

Modelo de caja: Entender el modelo de caja de CSS y cómo afecta al diseño y posicionamiento de elementos en una página HTML.

Responsive Design: Conocer las técnicas para crear sitios web que se adapten a diferentes tamaños de pantalla y dispositivos.

Frameworks y bibliotecas: Familiarizarte con el uso de frameworks como Bootstrap o bibliotecas como jQuery para facilitar el desarrollo de sitios web.

# CSS (Cascading Style Sheets):

Selectores y Especificidad: Comprender los diferentes tipos de selectores (como selectores de elemento, de clase, de id, selectores descendientes, etc.) y cómo usarlos para aplicar estilos a elementos específicos en tu HTML.

Selectores de tipo, clase, ID, atributo, combinadores.

Especificidad de los selectores.

Modelo de caja: Entender cómo funciona el modelo de caja en CSS, incluyendo propiedades como width, height, padding, margin, border, y cómo afectan al diseño y la disposición de los elementos en la página.

Propiedades de dimensiones: width, height.

Propiedades de borde: border, border-radius.

Propiedades de relleno: padding.

Propiedades de margen: margin.

Posicionamiento y Diseño: Dominar los diferentes métodos de posicionamiento en CSS, como static, relative, absolute, y fixed, así como el uso de propiedades como top, bottom, left, y right.

Posicionamiento: static, relative, absolute, fixed.

Modelo de caja flexible (Flexbox) y cuadrícula (Grid).

Tipografía y Colores: Comprender cómo controlar la tipografía en CSS, incluyendo propiedades como font-family, font-size, font-weight, line-height, y cómo crear efectos de texto.

Propiedades de texto: font-family, font-size, font-weight, text-align.

Propiedades de color: color, background-color.

Transiciones y Animaciones: Conocer cómo crear efectos de transición y animaciones utilizando CSS para mejorar la experiencia del usuario en tu sitio web.

Propiedades de animación: animation, keyframes.

Propiedades de transición: transition, transition-property, transition-duration.

## Responsive Design:

Media queries para adaptar el diseño a diferentes dispositivos.

Unidades de medida relativas: %, em, rem.

Propiedades CSS: Familiarizarte con una amplia gama de propiedades CSS que controlan aspectos como el color, tamaño, posición, diseño, animación y otros atributos visuales de los elementos HTML.

Flexbox: Conocer el modelo de diseño flexible (Flexbox) en CSS y cómo utilizarlo para crear diseños más complejos y adaptables.

Grid: Dominar el sistema de cuadrícula (Grid) en CSS y cómo utilizarlo para crear diseños de página más avanzados y estructurados.

Media Queries: Entender cómo utilizar media queries para crear diseños responsivos que se adapten a diferentes tamaños de pantalla y dispositivos.

Preprocesadores CSS: Familiarizarte con preprocesadores como Sass o Less, que permiten escribir CSS de manera más eficiente y mantenible utilizando características como variables, mixins, y anidamiento.

# JavaScript:

Fundamentos de JavaScript: Comprender los conceptos básicos del lenguaje, incluyendo variables, tipos de datos, operadores, estructuras de control (bucles y condicionales), funciones, y scope.

Variables y Tipos de Datos:

Variables: var, let, const.

Tipos de datos: números, cadenas, booleanos, objetos, arreglos.

Estructuras de Control:

Condicionales: if, else, else if.

Bucles: for, while, do-while.

Funciones y Ámbito:

Declaración de funciones: function.

Funciones de flecha: () => {}.

Ámbito léxico y ámbito de bloque.

DOM Manipulation: Saber cómo interactuar con el Document Object Model (DOM) para acceder, modificar y manipular elementos HTML dinámicamente.

Manipulación del DOM:

Selección de elementos.

Modificación de contenido, atributos y estilos.

Creación y eliminación de elementos.

Event Handling: Entender cómo manejar eventos del usuario en el navegador y responder a ellos utilizando JavaScript.

Eventos:

Asociación de manejadores de eventos: addEventListener.

Eventos comunes: click, submit, keydown.

Programación Orientada a Objetos en JavaScript: Dominar los conceptos de programación orientada a objetos (POO) en JavaScript, incluyendo la creación de clases, herencia, encapsulamiento y polimorfismo.

Programación Orientada a Objetos (OOP):

Clases y objetos.

Herencia y polimorfismo.

Encapsulamiento y abstracción.

AJAX y Fetch API: Conocer cómo realizar peticiones asíncronas al servidor web para cargar datos de manera dinámica sin necesidad de recargar la página.

ES6+ (ECMAScript 2015+): Familiarizarte con las características introducidas en las versiones recientes de JavaScript, como arrow functions, destructuring, template literals, classes, y otras.

Patrones de Diseño: Conocer los patrones de diseño comunes en JavaScript, como el patrón módulo, patrón constructor, patrón observador, y otros, y saber cuándo y cómo aplicarlos.

Frameworks y Bibliotecas Frontend: Familiarizarte con frameworks y bibliotecas populares como React, Angular o Vue.js, así como herramientas como webpack o Babel.

Testing: Entender los principios de las pruebas unitarias y de integración en JavaScript, y saber cómo utilizar herramientas como Jest, Mocha o Jasmine para escribir y ejecutar pruebas.

Optimización de rendimiento: Saber cómo optimizar el rendimiento de tus aplicaciones frontend, incluyendo técnicas de optimización de carga inicial, reducción del tiempo de carga y mejora de la eficiencia del código.

Accesibilidad: Comprender los principios de accesibilidad web y cómo aplicarlos en el desarrollo frontend para garantizar que tus aplicaciones sean accesibles para todos los usuarios.

Seguridad: Conocer las vulnerabilidades comunes en aplicaciones web y cómo evitarlas mediante buenas prácticas de seguridad en el desarrollo frontend.

Herramientas y Tecnologías Avanzadas:

Frameworks de JavaScript:

Aprende al menos uno de los principales Frameworks:

React.js

Angular

Vue.js

Preprocesadores CSS:

SASS o LESS para mejorar la eficiencia en la escritura de estilos.

Control de Versiones:

Git para gestionar el control de versiones de tu código.

Plataformas como GitHub o GitLab para alojar repositorios y colaborar en proyectos.

Gestión de Paquetes y Dependencias:

NPM (Node Package Manager) para instalar y gestionar paquetes de JavaScript.

Bundler y Task Runners:

Webpack o Parcel para empaquetar y optimizar tu código.

Gulp o Grunt para automatizar tareas repetitivas.

Testing:

Conocimientos básicos de pruebas unitarias y de integración.

Herramientas como Jest, Mocha o Jasmine.

Prácticas Avanzadas y Conceptos Específicos:

Performance Web:

Optimización de carga de páginas.

Minificación de archivos CSS y JS.

Carga diferida y asíncrona de recursos.

Accesibilidad Web:

Diseño y desarrollo inclusivo para personas con discapacidades.

Cumplimiento de estándares como WCAG (Web Content Accessibility Guidelines).

SEO (Search Engine Optimization):

Buenas prácticas para mejorar el posicionamiento en motores de búsqueda.

Optimización de metadatos y etiquetas HTML.

Desarrollo Profesional:

Metodologías de Desarrollo:

Agile, Scrum o Kanban para la gestión de proyectos.

Desarrollo colaborativo con metodologías como Gitflow.

Despliegue y Hosting:

Conocimientos básicos de servidores web y configuración.

Plataformas de alojamiento como Netlify, Vercel o AWS.

Mantenimiento y Depuración:

Uso de herramientas de depuración en navegadores.

Monitoreo y mantenimiento proactivo de aplicaciones.

Aprendizaje Continuo:

Mantente al día con las últimas tendencias y tecnologías en desarrollo frontend.

Participa en comunidades en línea, asiste a conferencias y realiza cursos de actualización.