# Primeros pasos del desarrollo frontend

En este curso vas a obtener los conocimientos necesarios para dar tus primeros pasos en el desarrollo front-end. Tenés tiempo para finalizar la cursada hasta el 1 de febrero. Al finalizar el recorrido del curso, en esta misma sección se te habilitará el botón “entregar proyecto” mediante el cual vas a poder realizar la entrega de tu proyecto final integrador para certificarte.

# Reglamento académico

**Ticmas Academy es una innovadora propuesta educativa autogestionada que ofrece una experiencia que se enfoca en el desarrollo de perfiles digitales.**

Se trata de una plataforma de simple acceso y fácil navegación: el contenido académico se encuentra distribuido en módulos formados por videos, podcast, infografías y textos breves que el estudiante puede recorrer, **a su ritmo y a libre demanda**. Cada uno de ellos se inicia y se termina de acuerdo con el tiempo destinado y según los avances que vaya realizando.

Para acompañar, al cierre de cada módulo se proponen **evaluaciones parciales** con ejercicios prácticos que poseen un feedback inmediato automatizado, lo que permite comprender dificultades, identificar errores y visibilizar la consolidación de aprendizajes.

Luego de terminar el recorrido, el estudiante deberá cumplir con la entrega de un **proyecto final integrador**, que certificará los aprendizajes incorporados y será evaluado por un experto,

Además, teniendo como base el objetivo de la autonomía y protagonismo del alumno, se espera que el producto final resulte ser una creación original. De este modo, si se utilizan fuentes de terceros, tendrán que ser citadas de forma completa.

Y en caso de solicitarse algún ajuste por parte del experto, se recibirá el feedback correspondiente y se tendrá la posibilidad de entregar nuevamente y recuperar el proyecto. La calificación mínima debe ser **6 (seis)**.

El proyecto final de cada estudiante formará parte de su portfolio profesional. El estudiante recibirá un **certificado** que dará cuenta del recorrido realizado.

La fecha límite para la recepción de proyectos será informada en la home del curso, quedando imposibilitada las posteriores entregas luego de esa fecha.

# Modulo 1

# Estructura de un proyecto web

**En este primer módulo aprenderás los conceptos básicos de HTML, su sintaxis y los elementos iniciales, para poder empezar a construir los documentos del lenguaje demarcado que utilizan los navegadores web.**

Editores de Texto y primer Documento HTML

Estructura del HTML y elementos de Titulo

**¿Que debo escribir en código HTML?**

**Básicamente todo el contenido de la página, pero nada que tenga relación con el diseño y los estilos visuales**

**En el HTML debemos incluir los textos, links, llamados a imágenes y videos, y principalmente las etiquetas que estructuran el contenido de toda pagina web. Todas las definiciones estéticas iran en la hoja de estilos CCS, y no en el documento HTML.**

**Nombre de archivos**

**Nombre de archivo de la pagina de inicio de todo sitio web**

**Index.html**

**Esto no significa que sea el único. Los archivos pueden tener cualquier nombre, pero la mayoría de los servidores van a reconocer al que se llame index.html como la pagina de inicio.**

****

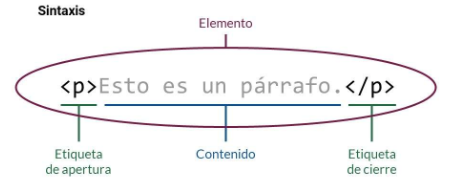
**Estructura básica de un documento HTML**



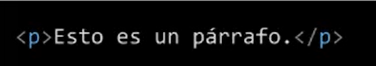
* **HTML: Gran contenedor de todo**
* **HEAD: Contiene toda la meta información o información adicional, por ejemplo el titulo del navegador.**
* **BODY: Lo que va a ver el usuario en el navegador**

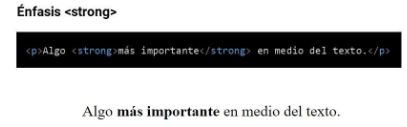
**Cada pieza tiene un sentido en el documento.**

**Sintaxis**

****

**Parrafo**

****

****

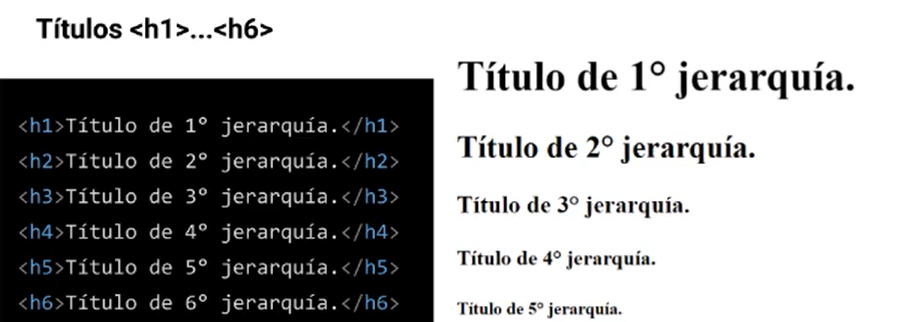
**Body**

**<Bode>…</Body>**

**Todo lo que este dentro de el se visualizara en la pantalla del navegador**

**Titulos**

**<Title>…</ Title >**

****

**Va del H1 al H6, según el nivel de jerarquía**

Como incluir imágenes en nuestro HTML

**Imagen**

**<img>**

**El elemento IMG se cierra en si mismo**

****

**Atributos obligatorios:**

**Src: Ruta y nombre del archivo**

**Alt: Texto alternativo (accesibilidad, buscadores, etc.). En el caso de que la imagen no se cargue por ejemplo. Si no carga la imagen y no esta el alt, no muestra nada. Siempre incluirlo.**

**Atributos opcionales:**

**Heigh: Alto de la imagen en pixeles.**

**Width: Ancho de la imagen en pixeles**

**No es una buena práctica en HTML, el diseño va en CSS.**

**Atributo: Informacion complementaria que se le da a un elemento para poder completarlo. Dan mas información de las que necesita ese elemento para mostrarse en el navegador.**

****

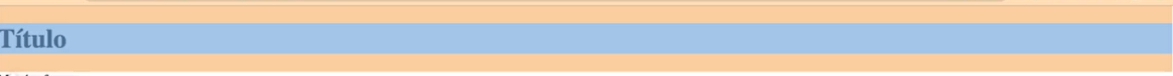
**Inspeccionar: Consola de Desarrollador de Google Chrome**

****Consola de Desarrollador de Google Chrome****

* **Me permite ver mis modificaciones en tiempo real**
* **Entorno para editar en tiempo real**
* **Si recargo la página se pierden los cambios en el caso de que no lo haya actualizado en mi documento.**

Elemento de Bloque – Elemento de línea

**Elemento de Bloque: Ocupa todo el espacio/linea que tiene disponible. Toman todo el espacio disponible de la pantalla. Siempre comienzan con una línea nueva y los navegadores automáticamente agregan algo de espacio antes y después del elemento.**

****

**Celeste: Lo que ocupa H1**

**Naranja: Margen**

**Todos los elementos de bloque en HTML**

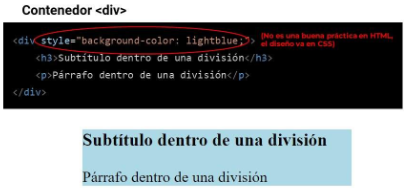


**Elemento de Linea: No empiezan con una nueva línea**

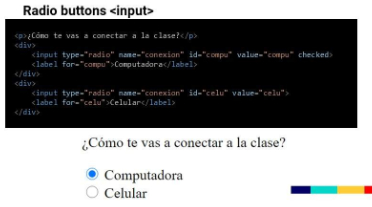
**Ejemplo: Elemento span**

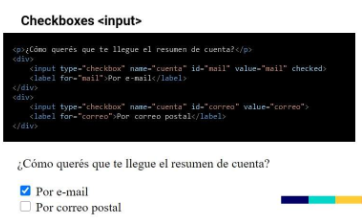


****

****

**** ****

****

****

****Formularios****

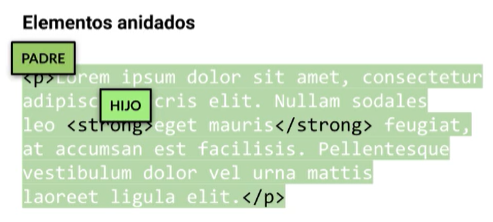
**Formularios <form>**

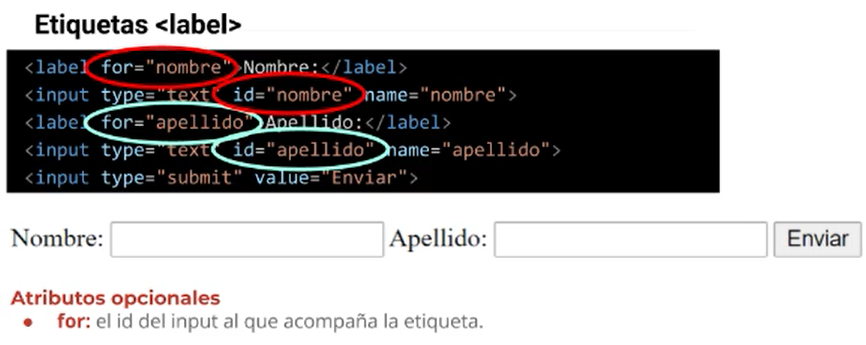
****

**Atributos opcionales**

* **Action: URL hacia donde se enviara la respuesta al formulario**
* **Method: Metodo HTTP que se usara para enviar el formulario.**
* **Name: Nombre del formulario**

**Elementos anidados: Dentro de este introduzco las diferentes etiquetas o campos.**





Resumen



# Modulo 2

# Elementos multimedia

**En** **este módulo veras cómo utilizar los elementos de HTML para incluir audio y video, como también entender el valor semántico de los**tags **y la importancia de maquetar siguiendo estos lineamientos.**

Elemento video

**¿Cuales son los elementos multimedia de HTML?**

Los elementos multimedia vienen en diversos formatos. Puede ser casi

cualquier cosa que puedas escuchar o ver, como imágenes, música,

sonido, videos, discos, películas, animaciones y más.

Las páginas web suelen contener elementos multimedia de diferentes

tipos y formatos.

**Compatibilidad en los navegadores**

**No todos los elementos son compatibles en todos los navegadores,**

**por eso te presentamos la siguiente página (https://caniuse.com/), en**

**la que vas a poder buscar el elemento que quieras aplicar y conocer su**

**compatibilidad con todos los navegadores existentes del mercado. Los usuarios pueden entrar en diferentes navegadores, tengo que ver como queda mi proyecto dentro de otros navegadores y dispositivos.**



Los formatos de video soportados son:

MP4 (video/mp4)

WebM (video/webm)

Ogg (web/ogg)

Elemento audio



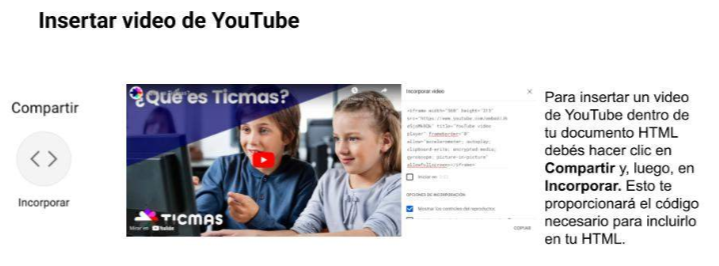
Los formatos de audio soportados son:

MP3 (mpeg/mp3)

Wav (audio/wav)

Ogg (audio/ogg)

Insertar videos de youtube





Web semántica y Open Graph

Elemento semántico = Elementos con un significado

Un elemento semántico describe claramente su significado tanto para el navegador como para el desarrollador.

Pensar en el usuario (amigable, facilidad de usar, etc) y también en definir bien el contenido, dándole un contexto.

Header, nav y footer le dan un valor semántico al sitio, ya que el robot de Google entiende el contenido.

Ejemplo



Un favicon es una pequeña imagen que se muestra al lado del título de la página en la pestaña del navegador.

Protocolo OpenGraph

un protocolo de Internet diseñado por Facebook para estandarizar el uso de los metadatos existentes en un sitio web y señalar los contenidos presentes en una página en específico

Permite previsualizar algo de nuestra pagina.

Las propiedades OG se deben incluir dentro del head

Resumen



[Estándares web](https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/The_web_and_web_standards#est%C3%A1ndares_web)

**Los estándares web** son las tecnologías que utilizamos para crear sitios web. Estos estándares existen como extensos documentos técnicos llamados especificaciones, que detallan exactamente cómo debería funcionar la tecnología. El W3C es el organismo de estándares web más conocido,

Modulo 3

Capa de presentación

**En este módulo aprenderás cómo aplicar estilos a tus documentos HTML, de manera tal que puedas diseñar tus páginas con colores, tipografías y espaciados, según lo requieras.**  
**La capa de presentación, aplicada a través de CSS, te permitirá customizar tus documentos y hacer diseños a medida. Verás las distintas formas de aplicar CSS, buenas prácticas,**box mode**l y conceptos clave para que puedas seguir adelante.**

Introducción a CSS

**Responsable de darle estilo a nuestra pagina web. Corresponde a la capa de presentación, la pagina web se vera de una forma mas amigable.**

**CSS nos permite aplicar los estilos que nosotros querramos a los elementos de HTML.**

****

**Formas de cargar un CSS**

**Formas de incluir CSS en nuestra hoja de HTML**

**Estilos dentro del HTML: Si creo un nuevo documento HTML no puedo reutilizarlo, ya que habrá muchos documentos.**

**Estilo externo al HTML: Lo que corresponde es tener en una hoja externa el CSS. Tener en una carpeta llamada CSS, ahí meteremos todas nuestra hojas de estilo. Vinculo el HTML con una hoja de estilos, que será un archivo dentro de una carpeta. Me permite reutilizar código.**

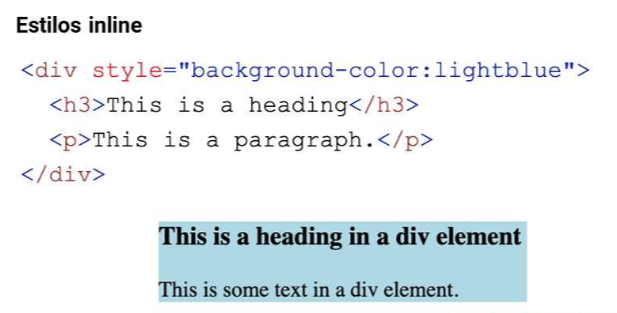
**Estilo Inline Style: Definir dentro de cada elemento. Pesima practica porque estoy mezclando CSS con HTML, aparte que queda atado a cada elemento.**

/\*

Comentarios en CSS

\*/

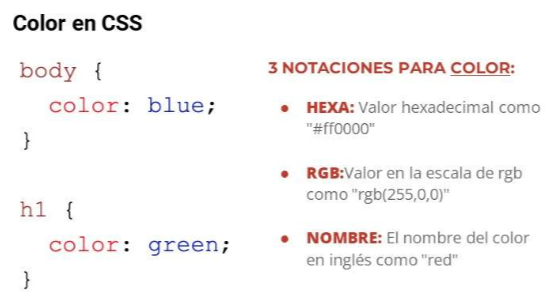
**:**

****

****

****

****

****

****

****

Box Model

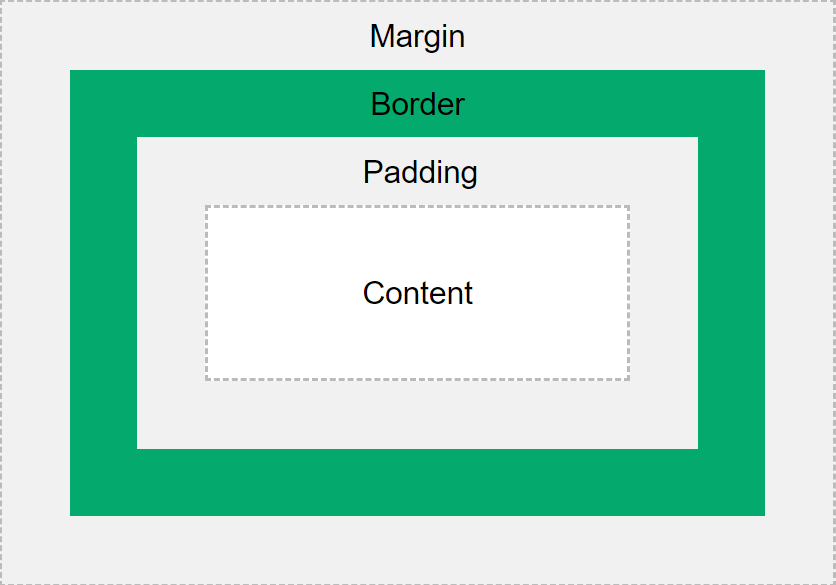
**Modelo de cajas: Como consideramos a todos los elementos dentro de CSS. En CSS, todos los elementos HTML pueden ser consideramos como cajas. Pensar las selecciones como cajas. Las cajas comparten propiedades:**

**Margin: Despeja un área fuera de la frontera. El margen es transparante.**

**Border: Lo que encapsula el contenido. Un borde que rodea el acolchado y el contenido.**

**Padding: Seccion transparanete que limpia el área alrededor del contenido. Despeja un área alrededor del contenido. El acolchado es transparente.**

**Content: Contenido propiamente dicho. El contenido del cuadro. Lo que tiene el elemento. Donde aparecen texto e imágenes.**

****

**Selectores y tipografia**

**Hoja de estilos en cascada: Funciona siempre el que tiene aplicada la propiedad.**

**\*: Comodin, aplica a todos los elementos**

**Selectores: Selecciona el elemento HTML para aplicarle un estilo.**

**Desde CSS se pueden compartir propiedades entre los mismos elementos**

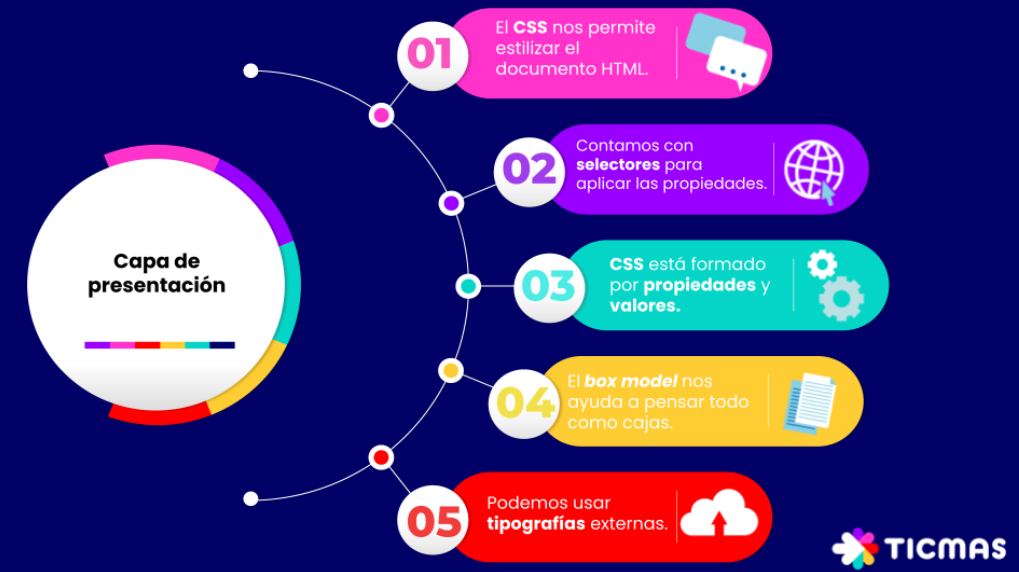
**Atributo ID:** id atributo es usado para especificar una identificación única para un elemento HTML. No puede tener más de un elemento con la misma identificación en un Documento HTML.

**Clases en CSS.**

**.nombre{}**

**Tipografias: Utilizar con Google fonts, ya que ahí se encuentran las permitidas por la web.**

**Resumen**

****

# Modulo 4

# Extendiendo CSS y sus frameworks

**En este módulo trabajarás con los conceptos clave del *responsive web design*. Aprenderás cómo transformar tu página web, originalmente solo para *desktop*, en *full-responsive*; es decir, que se adapte a cualquier dispositivo y que se siga visualizando correctamente.**

**A continuación, avanzarás con los *frameworks*más populares del mercado, de manera tal que, habiendo entendido los conceptos más importantes, puedas utilizarlos para avanzar rápidamente en el desarrollo *full-responsive*.**

Responsive WEB design introduction

Meta ViewPort – Media Queries

Responsive Web Design: Permite a cualquier usuario ver correctamente la pagina web desde cualquier dispositivo. Poder visualizar en escritorio, Tablet o celular, Smart tv, etc. Nos tenemos que encargar que, independiente la pantalla, se tiene que ver como corersponde

Meta viewport (parte en html)

El Viewport es una de las etiquetas más representativas de la web móvil, que nos permite configurar cómo debe interpretar una página el navegador web para móviles. Define el área visible de una pagina web. Desde HTML, nos deja definir el ancho y la escala inicial, según el dispositivo que acceda.

Full-responsive: Que se adapte a cualquier dispositivo y que se siga visualizando correctamente

. 

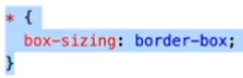


Se distorzionara la imagen según el dispositivo

El visual studio core lo agrega por defecto

Otros elementos

Grid-view: Parte la pantalla en columnas, trabajar con una grilla con x determina columnas, dependiendo el dispostivio (ósea, la grilla total), siempre se adapta al usuario.



**Media queries (parte en ccs)**

Las media queries (en español "consultas de medios") son utiles cuando deseas modificar tu página web o aplicación en función del tipo de dispositivo (como una impresora o una pantalla) o de características y parámetros específicos (como la resolución de la pantalla o el ancho del viewport del navegador). Nos permite aplicar estilos según el tamaño de pantalla. Regla de css que incluye propiedades si una condición es verdadera





Librería full responsive: Boostrap- Foundation – Tailwind

.(Cortan camino y tiempo, ya que hacer un sitio full responsive para los diferentes dispostivios lleva mucho tiempo)

**Framework Boostrap (primeros pasos)**

Bootstrap es una biblioteca multiplataforma milenaria o conjunto de

herramientas de codigo abierto para disefio de sitios y aplicaciones web.

Contiene plantillas de disefio con tipografia, formularios, botones,

cuadros, menus de navegacién y otros elementos de disefio basado en

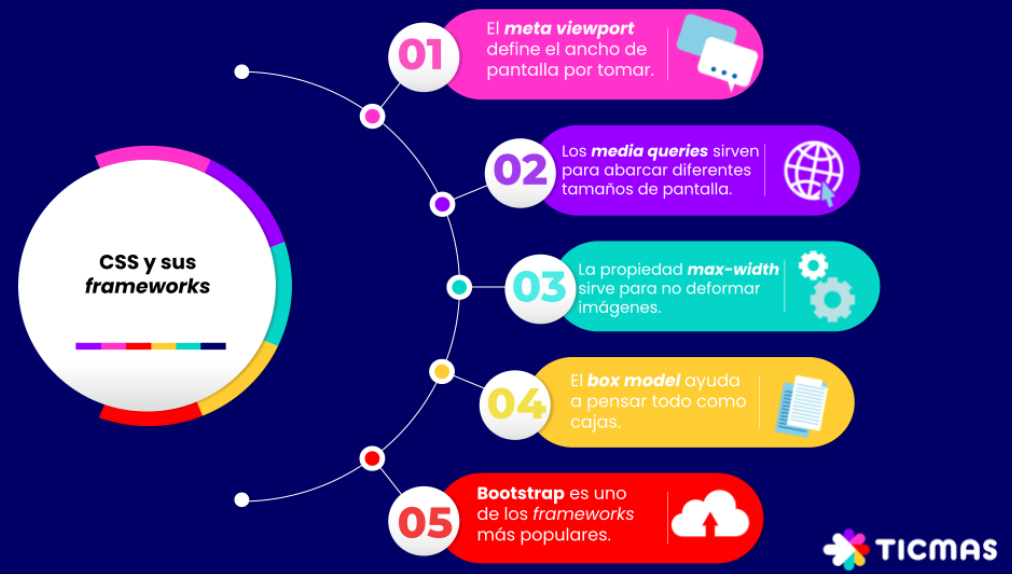
HTML y CSS, asi como extensiones de JavaScript adicionales. A

diferencia de muchos frameworks web, solo se ocupa del desarrollo

front-end. Heredamos propiedades y reglas de HTML ya hechas.

¿Para qué sirven los breakpoints en CSS?   
Para definir los píxeles en los cuales la pantalla debe cambiar el layout según el dispositivo.

Resumen:



# Modulo 5

# Alojamiento y tipo de servidores

**En este módulo aprenderás cómo funciona el modelo cliente-servidor, que es básicamente lo que mantiene operable la web. Entenderás cómo publicar el proyecto, qué debés tener en cuenta, como también qué elecciones tomar al momento de optar por el proveedor de *hosting*y dónde comprar los dominios.**

**Por último, aprenderás cómo utilizar GIT, la herramienta de versionado de código fuente más popular del mercado.**

**Alojamiento y tipos de servidores**

Por ahora estamos trabajando con proyectos localmente. Puerto/archivotrabajo

Modelo cliente-servidor



La arquitectura cliente-servidor es un

modelo de disefio de software en el que

las tareas se reparten entre los

proveedores de recursos o servicios,

llamados servidores, y los demandantes,

llamados clientes. Un cliente realiza

peticiones a otro programa, el servidor,

quien le da respuesta.

Como usuarios, consumimos la web a

través de navegadores web, siendo este

nuestro medio (Cliente) el cual realiza las

solicitudes a destino (Servidor)

Nosotros a traves de un navegador hacemos una petición, que nos lleva a un servidor y nos devuelve una respuesta

Los servidores tienen direcciones ip.(como si fuera la dirección de una casa)

Hostings (como si fuera la cochera de un auto: alquilar un espacio un determinado precio para guardar algo. Pagar un espacio adentro de un servidor): Para poder publicar tu proyecto en internet sera necesario contratar un

espacio dentro de un servidor para que puedas subir los archivos que

has desarroliado.

Existen miltiples empresas destinadas a vender y comercializar

hostings, con diferentes planes, capacidad de disco, memoria, entre

otros.

Para iniciar, podés contratar uno bien pequefio; conforme tu proyecto

vaya creciendo, irs escalando.

Es importante entender que tu hosting te permitira tener disponibles 24

horas al dia, los 365 dias del afio, el proyecto en internet

Dominios: ¿Podés comprar dominios en cualquier página?Debés buscar, según el país y la extensión de dominio, qué empresa te brinda la posibilidad de comprar el dominio por un período determinado.



Introduccion a GIT y versionado

Versionar: Tener las versiones de mis archivos.

Git: Herramientas de versionado de código fuente. Git es un software de control de

versiones disefiado por Linus Torvalds,

pensando en la eficiencia, la confiabilidad

y compatibilidad del mantenimiento de

versiones de aplicaciones cuando estas

tienen un gran numero de archivos de

cédigo fuente. Su propésito es llevar

registro de los cambios en archivos de

computadora incluyendo coordinar el

trabajo que varias personas realizan

sobre archivos compartidos en un

repositorio de cédigo.



Resumen



# Podcasts de cierre

Acontinuacion, encontraras cuatro

podcast elaborados por el profesional

que te acompaiio a lo largo de estos

méddulos.

Te invitamos a escucharlos y

continuar aprendiendo.

Podcast: Como encarar un proyecto web

Podcast: Logica de negocio y algo mas.

# Pensamiento lógico

# **En este primer módulo hablaremos de la lógica de programación y cómo empezar a pensar como una computadora. Entenderemos que la computadora procesa instrucciones y que somos los desarrolladores los responsables de escribir las sentencias.**

Pensamiento lógico en Programacion: Es indispensable a la hora de programar, ya que

nos permite reconocer el problema que querem:

solucionar, buscar una solucién que podai

programar y analizar las salidas que obtengamos para formarnos conclusiones. Capaz de codificar la solución al problema que tengo.

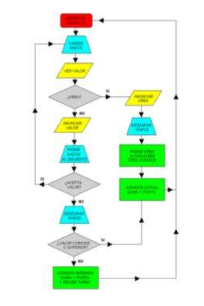
1° Tengo que reconocer el problema

Armar un diagrama de flujo: Es la representacién grafica de un algoritmo o

proceso. Nos muestra visualmente los pasos a

seguir para llegar a la solucién partiendo de un

problema determinado.



Algoritmo: Son un **conjunto de instrucciones definidas,**

**ordenadas y acotadas que sirven para resolver**

**un problema**, realizar un calculo o desarrollar una

tarea.

Se puede pensar como una receta de cocina, la

cual cuenta con una serie de ingredientes y

pasos por seguir, para llevar adelante la comida

que queramos realizar.

Pensar todos los posibles escenarios

**Como pensar lógicamente:**

Tenemos disponibles en el mundo de la

programacion diversas estrategias para poder

resolver un problema; las mas populares son:

¢ DRY (‘No te repitas a vos mismo”): No duplicar código. Cambian me ayuda para el mantemiento.

e Divide y venceras (paradigma mas importante en el diseño de algoritmos) Resolucion recursiva de un problema dividiéndolo en subproblemas

e WET



Computacion tradicional:

Dato + algoritmo = Resultado

Machine Learning

Data + Resultado = (genera un) Algoritomo

# Modulo 2

# Tipos de lenguajes de programación

**En el siguiente módulo, se abordará la definición de un lenguaje de programación, sus principales características, los diferentes paradigmas y los conceptos claves para entender la diferencia entre un lenguaje fuertemente tipado ante uno que no lo es.**

**Introduccion a un lenguaje de programación**

**Lenguaje de programación: Es un lenguaje formal (o artificial, es decir, un**

**lenguaje con reglas gramaticales bien definidas)**

**que le proporcionaa una persona, en este caso**

**el programador, la capacidad de escribir (o**

**programar) una serie de instrucciones o**

**secuencias de ordenes en forma de algoritmos (que produce una salida o ejecuta una determinada tarea)**

**con el fin de controlar el comportamiento fisico o**

**légico de un sistema informatico.**

**No todos los lenguajes de programación son iguales. La idea es entender primero la lógica para luego ir al lenguaje. Un algoritmo genérico se puede aplicar a diferentes lenguajes, cambiando su sintaxis.**

**Caracteristicas:**

[Programar](https://es.wikipedia.org/wiki/Programaci%C3%B3n) viene a ser el proceso de crear un software fiable mediante la escritura, [prueba](https://es.wikipedia.org/wiki/Beta_tester), [depuración](https://es.wikipedia.org/wiki/Depurador) (Debaggear), [compilación](https://es.wikipedia.org/wiki/Compilador) (produce un binario que puede ser ejecutado en un SO) o [interpretación](https://es.wikipedia.org/wiki/Int%C3%A9rprete_(inform%C3%A1tica)) (interpreta y ejecuta)

, y mantenimiento del [código fuente](https://es.wikipedia.org/wiki/C%C3%B3digo_fuente) de dicho [programa informático](https://es.wikipedia.org/wiki/Programa_inform%C3%A1tico).

**Depuraci6on de programas**

**La depuracién de programas es el proceso de**

**identificar y corregir errores de programacion.**

**Es conocido también por el término inglés**

**debugging, cuyo significado es eliminacién de**

**bugs (bichos), que es la manera en que se**

**conoce informaimente a los errores de**

**programación**

**HTML no es un lenguaje de programación, porque no tiene lógica, no se toman decisiones. Solo crea documentos para la web.**

**JavaScript se utiliza para front end y back end (todo lo que vive dentro de un servidor)**

**Interpretes y compiladores**

**Paradigmas de un lenguaje de programación**

**Compilador: Agarra el código fuente y lo baja a código de maquina. Si hay error en alguna parte del código, tira el error y no hay programa desarrollado.**

****

**Interprete: Programa capaz de analizar y ejecutar otros programas. Traducir el programa a medida que se va ejecutando. Interprete el código fuente. Si hay un error de sintaxis genera un problema en el programa.**

**Lenguaje de alto nivel y bajo nivel: Se clasifican según el nivel de abstracción. Que tan cerca estoy de hablarle al procesador.**

**Mas lejos: Alto nivel**

**Mas cerca: Bajo nivel (Ejemplo Assembler)**

**Paradigmas de programacion**

**Se denominan paradigmas de programaciéna**

**las formas de clasificarlos lenguajes de**

**programacién en funcién de sus caracteristicas.**

**Losidiomas se pueden clasificar en multiples**

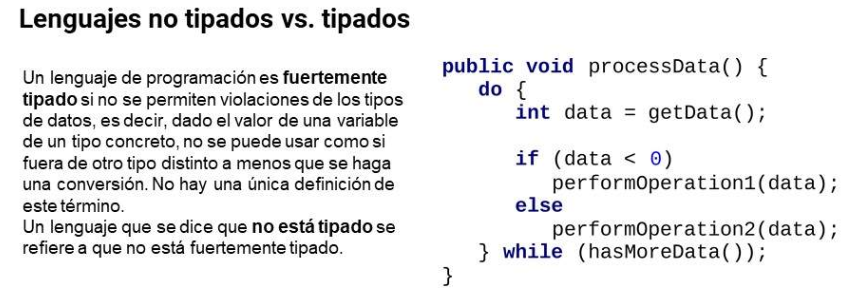
**paradigmas, y los mas comunes son:**

* **Procedural (receta de cocina)**
* **Orientado a Objetos (Todo es un objeto)**
* **Orientado a Eventos (Lo que sucede dentro del sistema)**

**Plataforma: En donde se puede correr mi lenguaje de programación? En que sistema operativo, depende del tipo de arquitectura**

**Tipado de los lenguajes de programación**

**Ejemplos**

****

**Fuertemente tipado: No permite violar el tipo de dato que almacenan las variables**



# Estructuras de la programación

**En este módulo veremos los diferentes elementos necesarios para poder resolver un problema informático. Iniciaremos con variables, sus tipos y entender para qué sirven; avanzaremos con los condicionales, que serán los que nos permitan tomar decisiones en los programas. Y, por último, aprenderemos de funciones para no escribir código repetido.**

**Tipos de variables**

**Uso de variables**

**Variable: Es un elemento dentro del mundo de la programación que me permite guardar valores para usarlos mas adelante en el algoritmo. Tiene un identificador, tipo de dato y el dato en si a guardar. Se almacena en memoria**

**Tipos de datos**

**Veamos a continuación los 5 tipos de datos primitivos que existen en JavaScript:**

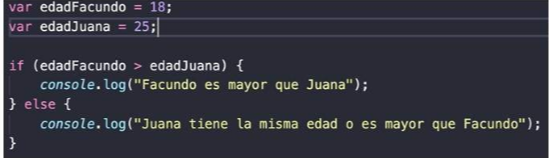
1. Numero: Con punto flotante, decimales y enteros
2. String: Cadena de caracteres.
3. Booleano: Tipo de dato lógico, cuyos posibles valores son true o false.
4. Undefined: Tipo de dato de una variable a la cual aun no se le asigno valor
5. Null Ausencia de valor

Cada tipo de dato tiene ciertas operaciones asociadas, por ejemplo: a los números los puedo sumar, restar, multiplicar, dividir, a los booleanos los puedo operar/comparar usando los operadores/comparadores lógicos.

JavaScript es no tipado, me deja cambiar el tipo de dato de una variable

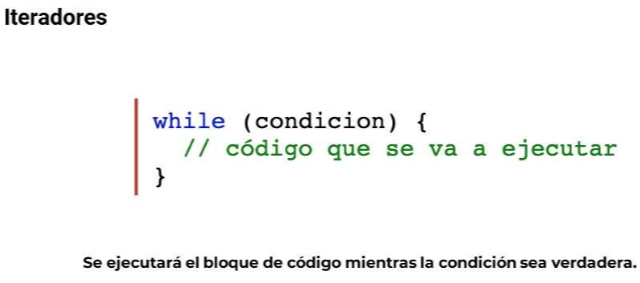
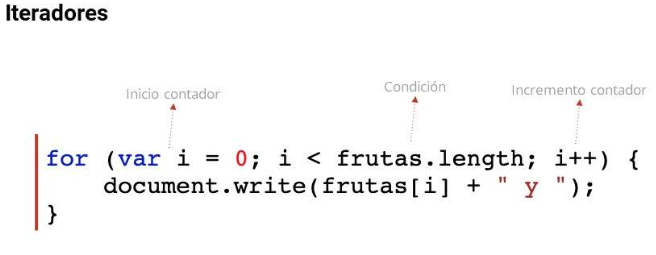
**Condicionales e iteradores**

**Condicionales: Una sentencia condicional es un conjunto de comandos que se ejecutan si una condición es verdadera. JavaScript soporta dos stencias condicionales: if..else y switch**

****

**Se utiliza la sentencia if para comprobar si la condición lógica es verdadera y la opción else, para ejecutar una sentencia si la condición es falsa.**

**Mas de iteradores y funciones para reutilizar código**

****

**Funciones: Encapsula un bloque de código.**

**Es un elemento el cual tiene una entrada y una salida, por lo que genera en el**

**medio una transformacién.**

**Enel colegioustedes vieron que las funciones tenian como entrada nimeros y**

**como salida obtenian nimeros distintos en un principio.**

**En programaciénesto avanza mas y las funciones pueden tener varios tipos**

**distintos de entrada y varios tipos distintos de salida. Por ejemplo, una funcién en**

**donde entra un nimero y sale como resultado un texto.**

****

# Fundamentos JavaScript I

**En este módulo daremos nuestros primeros pasos en JavaScript. Aprenderemos cómo es su sintaxis, cómo integrar lo aprendido hasta aquí junto con HTML y CSS. Por último, mediante los eventos que proporciona este poderoso lenguaje de programación, podremos dejar listos nuestros sitios para que los usuarios puedan interactuar con él.**

**Introducción a JavaScript**

**Sintaxis y como escribir un documento JS**

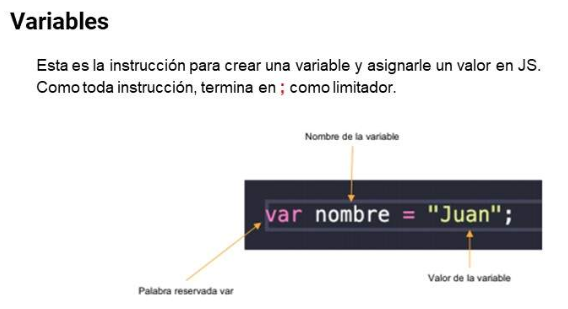
**JavaScript (JS) es un lenguaje ligeroe**

**interpretado, orientado a objetos, mas**

**conocido comoel lenguaje de scriptpara**

**paginas web. Lenguaje mas popular. No es tipado. Este le agrega interactividad al sitio web.**

****

****

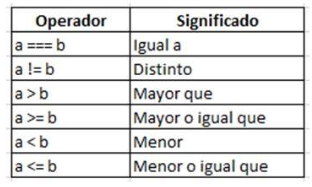
**Var, Let: Si cambia la variable con el tiempo**

**Const: No cambia el valor de la variable**

****

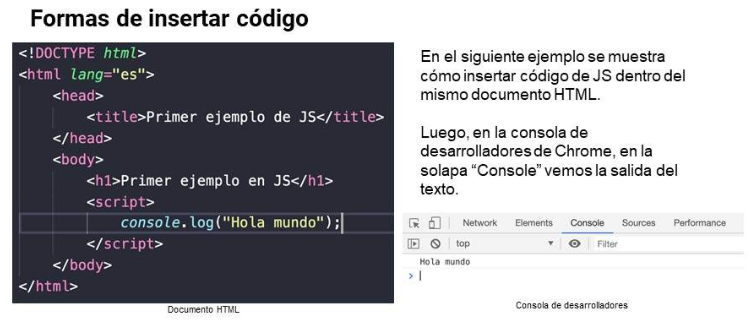
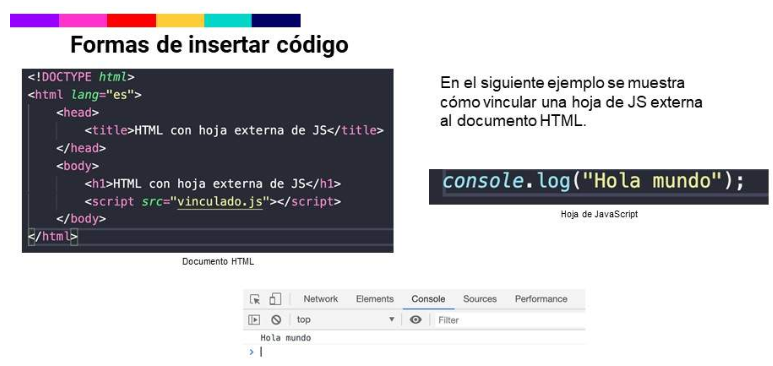
****

****

****

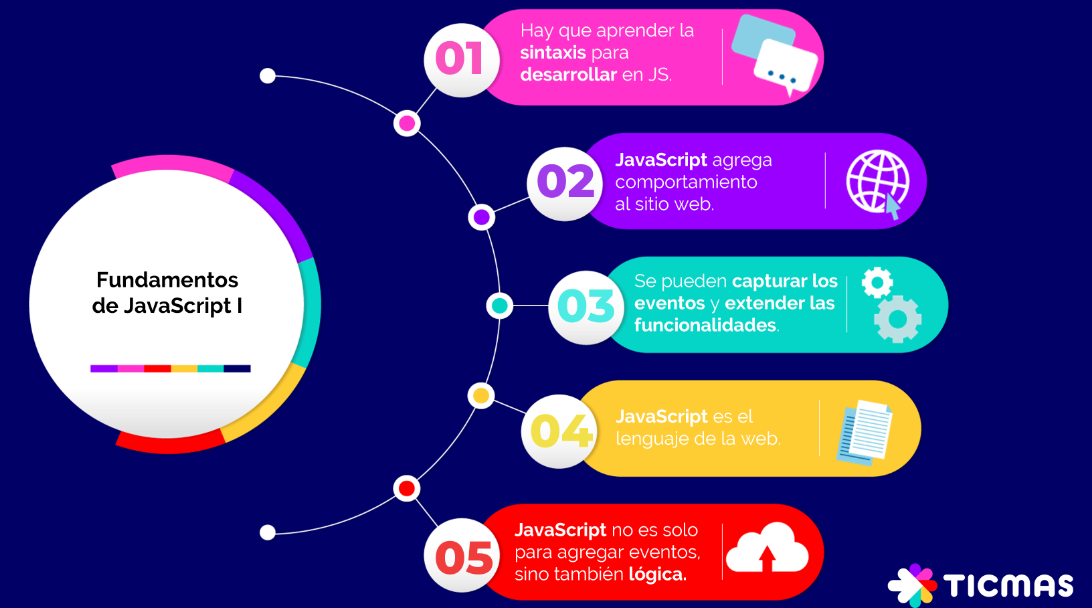
**Agregar código JavaScript a HTML**

**Integrando JavaScript a un documento HTML**

****

**Introduccion a Eventos de JavaScript**

**Resumen**

****

# Pensamiento lógico

**En este módulo veremos el DOM y cómo se debe aplicar con JavaScript, cómo empezar a dar los primeros pasos transportando datos con el formato JSON; y, por último, llegando bastante lejos y siendo muy ambiciosos, iremos por las últimas funcionalidades disponibles de JS, como geolocalización y aplicaciones web progresivas.**