**FULL-STACK BOOTCAMP**

**FULL STACK DENTRO DEL MUNDO DE LA INFORMATICA**

¿QUIENES SOMOS?

web applications

en unñtima instancia lo sprogramadores gull stack son desarrolladores de aplicaciones web

se comparte con back end (bbdd etc)

front edn nterfaces usuario gráficas

a medi ocamino está el full stack, que lo hacemos todo

a veces la especialidad depende del tamaño de la empresa (si es pequeá tal vez sea full stack)

en una gran empresa seria un intermediario entre las dos especialdidades

El desarrollo de las aplicaciones web es solo una única parte dentro de el Desarrollo de Aplicaciones general, ya sea moviles, escritorio, dispositivos electrónicos, etc.

y todo esto dentro de la ingeniería de software, que serái el mundo más genérico dentro de este sector, que contendría también arcquitectura de sogtware, desarrollo de frameworks, etc

En resumen, el desarrollo de aplicaciones web, y más conceratemente al desarrollo full stack, es solo un pequeó punto dentro de la informática, sin contar además el hardware, algoritmia, etc etc, que son otros campos como la ingeniería de software.

Es un area muy cinceta dentro del mundo muy amplio de la informatica

**Progressive web apps**

**DESARROLLO WEB**

¿Qué ahcemos?

INTERNET Y LA WEB

fullstacl->web apps->www->Internet

1.1 Internet

particularidades tcp ip

todos los equipos tienen un id que es la dirección ip

internet solo es la red de cominicación , no el contenido (serían las carreteras qie unen ciudades y pueblos)

red de redes que cominica ordenadores entre si

1.2 La WEB (WWW)

sistema de distribución

sería el equivalente a FEDEX en tema de mensakería de paquetes (y internet las careeteras)

FTP- file transfer protocol

la www, su caracter distintivo, además demandar paquetes dehipertexo, en alto nivel, cuando se pograma una web se trabaja con otros protocolos, específicos dela web, http y https;

¿Quien gestiona la www, controla la evolución actialización, etc.?

w3c--> world wild web consuortium

Son los expertos, los que “hacen” la web

[www.w3schools.com](http://www.w3schools.com)

1.3 Web Apps

aplicaciones que se distribuyen a traves dela web

HTML -ESTRUCTURA: tabiques, tejado, etc

CSS - ESTILOS - Pintado, alfombras, forma ventanas

JS - DINAMISMO -interruptores, agua, sea interactiva, etc

una web puede funcionar sin JS segun cómo, aunque solo en situaciones muy básicas (como vivir en una casa sin luz)

juntados los en una carpeta, esto es un “sitio web”

uedes tener muchos archivos de cada, no solo uno.

Cada archivo html es una “página” de tu sitio web.

aunque tengas 20 archivos css y un html, solo tienes una página, etc

puedes tener un css y 5 html, y tendrías 5 páginas.

1.4 Fullstack developer.

Frameworks

Ya no escribimos a mano el html, css, etc

programamos en una herramienta de más alto nivel;

backbone, angular, react, etc etc

todo vino pq hubo un boon de la web y trajeron ideas de otras areas.

en uñtuma instancia aprenderemos a usar estos frameworks, pero ahora aprenderemos a hacer html etc a mano, para entender qué hay detrás de estos frameworks.

Primer paso para desarrollar es saber qué herramientas vas ausar para hacer la web

este conjunto de herramientas que seleccionas para hacer la web es el STACK.

Hay mil combinaciones posibles según las herramientas que escojas, aunque con el tiempo se ha visto que algunas combinan muy bien entre ellas

por ejemplo:

MEAN: Mongo DB para bbdd, express como framwork de ack end, angular como framework de front end, y node para la infraestructura. Es el que usarems en este bootcamp

MERN, cambiando angular por React.

2. IDE - Integrated developmnet environment

Son los editores

herramientas:

Para mantenerse al día en estemundo que evoluciona tanto puedes

-estar pegado ala comunidad

-mirando informes, como “Stack overflow”

Tienen una base enorme de programadores y ayudan a resolver dudas que puedas tener.

Cada año hacen un estudio y es una manera de estar al día

insight.stackoverflow.com/survey/2019

En el apartado de tecnología, podemos ver ODAES (III. Development environtment and tools

1. Visual Studio CODE: muy ágil, intuitivo, etc.

2. HTML

Lenguaje de marcado, node programación

“Texto con metadatos”

Los metadatos son datos sobre los datos, es decir, información relacionada con ls datos.

<h1>HOla mundo</h1>

el metadato sería decir que este dato, “hola mundo”, es un header. No es simplemente texto plano. Se le da información esxtra y se convierte en hipertexto.

HTML es un lenguaje para generar hiperteto que utiliza etiquetas para dar esa informacón adicional, le da un “marcado” a los datos.

HTML es un lenguaje de marcado.

Muestra un contenido e información sobre ese contenido.

El navegador al ver que es html, te “traduce” esa infromación y te lo muestra como se supone que se debería ver.

Lo que se ve en el navegador no es la web, es la interpretación de la web.

LA web en sí es lo que se ve con “ver el código fuente”. (Control + U)

El navegador “RENDERIZA” este contenido.

2.1 Inspector de elementos

Nos permite ver la estructura html de una página web

También podemos ver los estilos.

TAGS

1. Tags estructurales: definen la estructura de la página
   1. los más básicos son header, body, foote.
2. tags semánticos: definen qué es un título, un texto, etc

1.1. tags estructurales.

cuando se abre uno, se tiene también que cerrar

* head: información relevante sobre el sitio web.
  + codificación de los caracteres, como el utf-8, el más usado hoy en día.
  + metadatos de la web
  + tag BASE. se indica dónde stá la base de nuestra web (por si tiene muchas páginas
* body: contenido de la web
  + HEader: nuevo tag, que nos enseña el header visible de la web (ya que la etiqueta head en realidad es solo par ametadatos etc)
  + div: tag estructural, “contiene” otros elementos,no tiene información de contenido
  + p (paragraph).
  + links: hacer referencias a cosa fuera de al página:
    - img: para cargar imágenes
    - a: para otras páginas, dentro o fuera de nuestro sitio
    - link: para vincular, por ejemplo, a css. Es buena práctica **ponerlo en el head**, para que lo cargue antes de llegar a la estructura. Si el link está después de algo del contenido, este se cargará sin estilos y después se actualizará, de manera que se verá durante un momento sin estilos hasta que se vuelva a cargar.
      * rel: opcional. le ahorras al navegador decirle qué tipo de link es (stylesheet, por ejemplo)
      * href: dónde encontrar el archivo (en este caso el css)
    - script: para vincularlo con el javascript. Tiende a provocar errores si se pone en el head a veces. Si el archivo hace referencia a contenido que no se ha cargado, es decir, que está en el body, “petará”, porque en ese instante de tiempo no existe ese contenido. PAra prevenirlo, ponerlo **en el footer o al final del body**, después de todo el contenido.
      * src: la ruta al archivo js

Se pueden aprovechar plantillas enteras para docs nuevos de la cheatsheet.

* formularios: para que le usuario mande infomración a la página (para registrarse, mensaje, dirección, etc etc)
  + todos empiezan con la etiqueta <form>. Es una etiqueta estructural.
  + action: donde se envía
  + method: cómo se envía
  + input: la etiqueta semántica, que pueden ser de diferentes tipos
    - text
    - textarea:se puede redimensionar
    - password: como texto, pero se ve oculto el texto
    - email (html5): va abuscar que tenga @ y dominio. Si no lo tiene, da error y sale un tooltip indicándolo
    - date: calendario. cuando se clica sobre él, aparece un calendario
  + button
    - en w3school se puede ver todos los tipos de input
  + select: dropdown. es también una etiqueta estructura.
    - se tiene qeu añadir una “option”, que corresponderá a cada una d elas opciones del menú dropdown
  + button: para enviar el contenido (type=”button”)

Atributos de las tags HMTL

LE dan especifidad a la etiqueta. Sirve para hacer algun elemento de una manera particularidad

Hay GENÉRICOS/GLOBALES(class, para asignar un indentificador de clase, que se puede hacer que varios elementos tengan unas caracteristicas similares, por ejemplo “catalogo”; id, para darle un identificador **único** para ese elemento, que no se podrá repetir) y los ESPECÍFICOS (type dentro de imput, src dentro de img, etc)

<div id=”row1” class=”catalog” </div>

<div class=”catalog” </div>

3) ESTILOS (CSS) -Hojas de estilos en cascada

html solo da la estructura y contenido.

CSS les aplica estilos para que se muestren de una manera concreta.

si se introduce directamente en el html, se usa el atributo o tag “style”

1. Estilos en línea (in-line), directamente como atributo

<p style=”color:css”> Testeando </p>

2. como un bloque de código, usando la tag stye con llaves seleccionando el tipo de tag

<style>

p {

}

#miID{

}

.miCLASS{

}

</style>

el HTML es la estructura, así que es feo y desorganizado que estén los estilos dentro del código html.

PAra evitar eso, se cogen los estilos y se hacen en un archivo aparte del CSS.

en el archivo css ya no hace falta tener las etiquetas style, se puede incluir directmante el elemento y los estilos en bloque