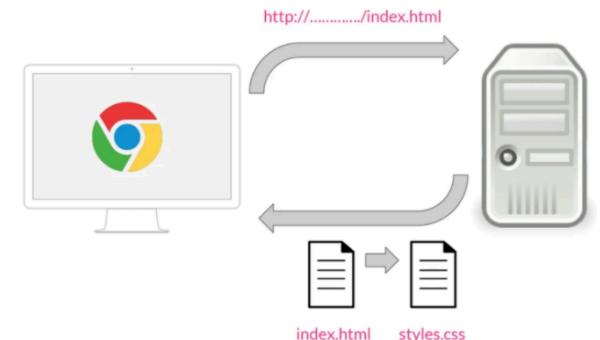


QUE ES CSS?

- CSS es un lenguaje de diseño gráfico que permite definir y crear la presentación de un documento HTML.
- Tecnología que ha tenido una evolución en el tiempo, que actualmente se encuentra en su versión 3.

CÓMO FUNCIONA?

- Cuando desde un navegador, solicitamos una página a través de una dirección, por ejemplo **https://escalab.academy**, esta petición va a un servidor web, que nos devuelve la página que se ha solicitado.
- Para aplicar estilos en las páginas HTML, se utiliza una hoja de estilos con la extensión .css, por lo que cuando estos dos documentos llegan al navegador, va a leer el documento HTML, le aplica los estilos CSS y lo muestra.



CÓMO SE ORDENAN LOS ELEMENTOS?

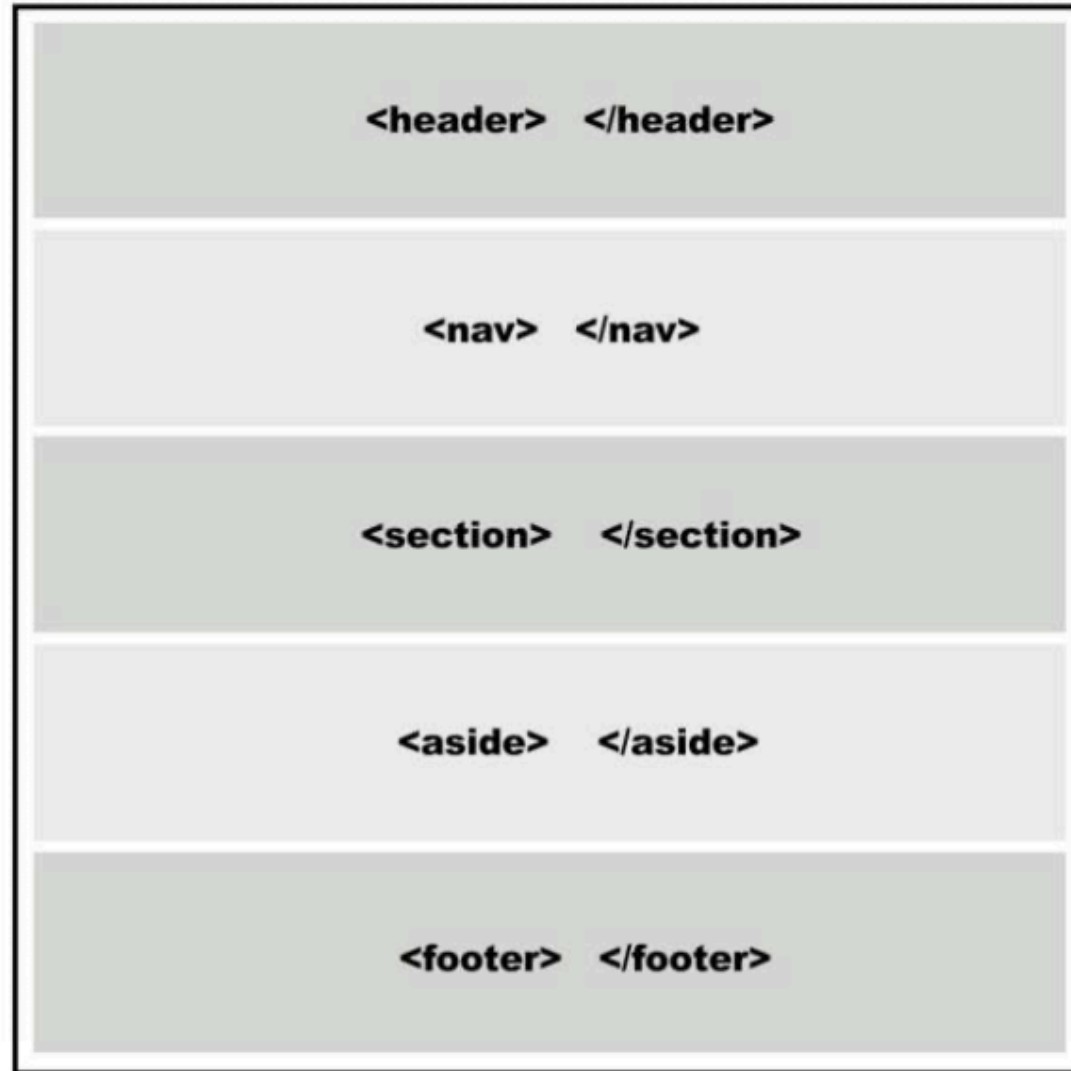
Con respecto a la estructura, básicamente cada navegador ordena los elementos por defecto de acuerdo a su tipo: block (bloque) o inline (en línea). Esta clasificación está asociada con la forma en que los elementos son mostrados en pantalla.

ELEMENTOS BLOCK

Elementos block son posicionados uno sobre otro hacia abajo en la página

ELEMENTOS INLINE

Elementos inline son posicionados lado a lado, uno al lado del otro en la misma línea, sin ningún salto de línea a menos que ya no haya más espacio horizontal para ubicarlos.



DONDE APLICAR LOS ESTILOS?

- Podremos agregar los estilos de distintas formas

ESTILOS EN LÍNEA

Una de las técnicas más simples para incorporar estilos CSS a un documento HTML es la de asignar los estilos dentro de las etiquetas por medio del atributo **style**

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este es el título del documento</title>
  </head>
  <body>
    <p style="font-size: 20px">Mi texto</p>
  </body>
</html>
```

ESTILOS EMBEBIDOS

Una mejor alternativa es insertar los estilos en la cabecera del documento y luego usar referencias para afectar los elementos HTML correspondientes

ARCHIVOS EXTERNOS

Declarar los estilos en la cabecera del documento ahorra espacio y vuelve al código más consistente y actualizable, pero nos requiere hacer una copia de cada grupo de estilos en todos los documentos de nuestro sitio web. La solución es mover todos los estilos a un archivo externo y luego utilizar el elemento **<link>** para insertar este archivo dentro de cada documento que los necesite. Este método nos permite cambiar los estilos por completo simplemente incluyendo un archivo diferente.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este texto es el título del documento</title>
    <style>
      p { font-size: 20px }
    </style>
  </head>
  <body>
    <p>Mi texto</p>
  </body>
</html>
```

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <p>Mi texto</p>
  </body>
</html>
```

COMO REFERENCIAMOS LOS ELEMENTOS

Almacenar todos nuestros estilos en un archivo externo e insertar este archivo dentro de cada documento que lo necesite es muy conveniente, sin embargo no podremos hacerlo sin buenos mecanismos que nos ayuden a establecer una específica relación entre estos estilos y los elementos del documento que van a ser afectados.

REFERENCIANDO CON PALABRA CLAVE

Al declarar las reglas CSS utilizando la palabra clave del elemento afectamos cada elemento de la misma clase en el documento. Por ejemplo, la siguiente regla cambiará los estilos de todos los elementos `<p>`

```
p { font-size: 20px }
```

REFERENCIANDO CON EL ATRIBUTO ID

El **id** es un identificador del elemento. Esto significa que el valor de este atributo no puede ser duplicado. Este nombre debe ser único en todo el documento.

Para referenciar un elemento en particular usando el atributo **id** desde nuestro archivo CSS la regla debe ser declarada con el símbolo **#** al frente del valor que usamos para identificar el elemento:

```
#texto1 { font-size: 20px }
```

REFERENCIANDO CON EL ATRIBUTO CLASS

Este atributo es más flexible y puede ser asignado a cada elemento HTML en el documento que comparte un diseño similar.

Para trabajar con el atributo **class**, debemos declarar la regla CSS con un punto antes del nombre. La ventaja de este método es que insertar el atributo **class** con el valor **texto1** será suficiente para asignar estos estilos a cualquier elemento que queramos:

```
.texto1 { font-size: 20px }
```

REFERENCIANDO CON CUALQUIER ATRIBUTO

Aunque los métodos de referencia estudiados anteriormente cubren un variado espectro de situaciones, a veces no son suficientes para encontrar el elemento exacto. La última versión de CSS ha incorporado nuevas formas de referenciar elementos HTML. Uno de ellas es el Selector de Atributo. Ahora podemos referenciar un elemento no solo por los atributos **id** y **class** sino también a través de cualquier otro atributo.

REFERENCIANDO CON PSEUDO CLASES

Contiene cuatro elementos **<p>** que, considerando la estructura HTML, son hermanos entre sí e hijos del mismo elemento **<div>**. Usando pseudo clases podemos aprovechar esta organización y referenciar un elemento específico sin importar cuánto conocemos sobre sus atributos y el valor de los mismos



```
p[name] { font-size: 20px }
```



```
p[name="mitexto"] { font-size: 20px }
```




```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <div id="wrapper">
      <p class="mitexto1">Mi texto1</p>
      <p class="mitexto2">Mi texto2</p>
      <p class="mitexto3">Mi texto3</p>
      <p class="mitexto4">Mi texto4</p>
    </div>
  </body>
</html>
```


REFERENCIANDO CON PSEUDO CLASES

La pseudo clase **nth-child()** nos permite encontrar un hijo específico. En el ejemplo anterior tiene cuatro elementos **<p>** que son hermanos. Esto significa que todos ellos tienen el mismo padre que es el elemento **<div>**. Lo que esta pseudo clase está realmente indicando es algo como: “el hijo en la posición...” por lo que el número entre paréntesis será el número de la posición del hijo, o índice.

REFERENCIANDO CON PSEUDO CLASES

La palabra clave **odd** para la pseudo clase **nth-child()** afecta los elementos **<p>** que son hijos de otro elemento y tienen un índice impar. La palabra clave **even**, por otro lado, afecta a aquellos que tienen un índice par.




```
p:nth-child(2){  
  background: #999999;  
}
```

```
p:nth-child(odd){  
  background: #999999;  
}  
p:nth-child(even){  
  background: #CCCCCC;  
}
```

REFERENCIANDO CON PSEUDO CLASES

Existen otras importantes pseudo clases relacionadas con esta última, como **first-child**, **last-child** y **only-child**, algunas de ellas recientemente incorporadas. La pseudo clase **first-child** referencia solo el primer hijo, **last-child** referencia solo el último hijo, y **only-child** afecta un elemento siempre y cuando sea el único hijo disponible.

Otra importante pseudo clase llamada **not()** es utilizada realizar una negación



```
p:last-child{  
  background: #999999;  
}
```

```
:not(p){  
  margin: 0px;  
}
```

NUEVOS SELECTORES

Hay algunos selectores más que fueron agregados o que ahora son considerados parte de CSS3 y pueden ser útiles para nuestros diseños. Estos selectores usan los símbolos `>`, `+` y `~` para especificar la relación entre elementos.

SELECTOR `>`

El selector `>` está indicando que el elemento a ser afectado por la regla es el elemento de la derecha cuando tiene al de la izquierda como su padre.

```
div > p.mitexto2{  
    color: #990000;  
}
```

SELECTOR +

Este selector referencia al elemento de la derecha cuando es inmediatamente precedido por el de la izquierda. Ambos elementos deben compartir el mismo padre.



```
p.mitexto2 + p{  
  color: #990000;  
}
```

2

QUE ES GIT?

Herramienta de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.



GIT INIT

Esto crea un subdirectorio nuevo llamado `.git`, el cual contiene todos los archivos necesarios del repositorio – un esqueleto de un repositorio de Git. Todavía no hay nada en tu proyecto que esté bajo seguimiento.



GIT FETCH

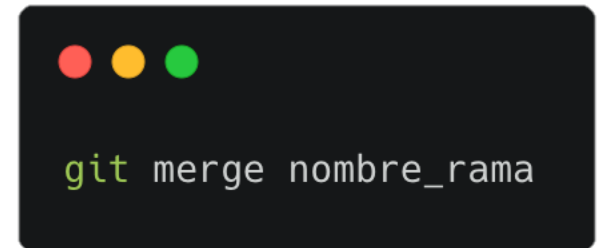
Descarga los cambios realizados en el repositorio remoto.

GIT MERGE <nombre_rama>

Mezcla en la rama en la que te encuentras parado, los cambios realizados en la rama “nombre_rama”.

GIT PULL


Unifica los comandos *fetch* y *merge* en un único comando.



2

GIT COMMIT -m “<mensaje>”

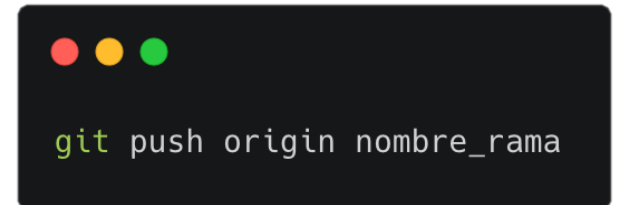
Confirma los cambios realizados. El “mensaje” generalmente se usa para asociar al *commit* una breve descripción de los cambios realizados.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays the command `git commit -m "Primer commit"` in a light green monospace font.

```
git commit -m "Primer commit"
```

GIT PUSH ORIGIN <nombre_rama>

Sube la rama “nombre_rama” al servidor remoto.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays the command `git push origin nombre_rama` in a light green monospace font.

```
git push origin nombre_rama
```

GIT STATUS

Muestra el estado actual de la rama, como los cambios que hay sin commitear.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays the command `git status` in a light green monospace font.

```
git status
```

GIT

ADD <nombre_archivo>

Agregas el archivo “nombre_archivo” para hacerle seguimiento.



```
git add nombre_archivo.txt
```

GIT CHECKOUT -

b <nombre_rama_nueva>

Crea una rama a partir de la que te encuentres parado con el nombre “nombre_rama_nueva”, y luego salta sobre la rama nueva, por lo que quedas parado en esta última.



```
git checkout -b nombre_rama_nueva
```

GIT BRANCH

Lista todas las ramas locales.



```
git branch
```


GIT BRANCH -a

Lista todas las ramas locales y remotas.

GIT PUSH ORIGIN <nombre_rama>

Commitea los cambios desde el branch local origin al branch “nombre_rama”.



```
git branch -a
```



```
git push origin nombre_rama
```

2

GITHUB

Sitio número uno para almacenar repositorios git. El sistema esta diseñado para permitir a los usuarios crear fácilmente sistemas de control de versiones basados en Git. ¿Por que es tan popular? Git admite fusiones y divisiones de versiones uniformes con la ayuda de herramientas de visualización y herramientas para la navegación a través del historial de desarrollo no lineal. Por ahora, GitHub alberga más de 50 millones de proyectos de código abierto.



2

BENEFICIOS

- **Seguimiento de errores:** Esta característica pertenece a las funciones de colaboración y permite mejorar la calidad del código al mantener registro de los errores de software detectados en el proyecto.
- **Búsqueda rápida:** El repositorio proporciona una estructuración conveniente de proyectos que permite una búsqueda y clasificación eficiente. Además una indexación adecuada permite a los usuarios encontrar cualquier cadena de código en los repositorios públicos.

BENEFICIOS

- **Comunidad:** GitHub reporta tener más de 20 millones de usuarios a abril del 2017. Esta enorme comunidad de desarrolladores en todo el mundo es una gran fuente de experiencia y habilidades compartidas.
- **Trabajo conjunto:** GitHub brinda funciones eficientes para la administración de equipos.

DESVENTAJAS

- El servicio no es completamente gratuito. Para acceder a todas las funciones de GitHub, se debe actualizar a un usuario Premium.
- Limitaciones de tamaño. Los archivos no pueden ser mayores a 100 MB mientras que el repositorio puede alojar 1 GB de información.

GITLAB

El servicio también está desarrollado en la base del control de versiones de Git. A pesar de que la funcionalidad de GitLab es similar a su principal competidor, GitHub, existen algunas peculiaridades importantes. GitLab tiene versiones diferentes, como GitLab SAAS que es adecuada para las empresas y GitLab Community Edition, una solución individual para los usuarios.



BENEFICIOS

- Es gratis. Eso significa que los usuarios pueden tener un número ilimitado de repositorios privados. Esto es la versión comunitaria, y los usuarios tendrán que pagar si requieren la versión empresarial. Esta última agrega algunas características adicionales a la funcionalidad básica que mejora la interacción con herramientas en línea, flujo de trabajo y administración de servidores, entre otras.

BENEFICIOS

- GitLab opera bajo una licencia de código abierto.
- Seguimiento de errores y edición de código basado en la web.

DESVENTAJAS

- Interface relativamente lenta
- Frecuentes problemas técnicos con los repositorios

BITBUCKET

Similar a GitHub, orientado a los equipos de desarrollo profesional, ya que proporciona grandes beneficios para ellos, como repositorios privados gratuitos, integración con Jira, revisión de código avanzado y CI. Al mismo tiempo, con el crecimiento del equipo, Bitbucket ofrece condiciones de precios más adecuadas comparadas con GitHub y GitLab. Bitbucket también proporciona una modelo de implementación flexible para equipos.



BENEFICIOS

- Repositorios privados para equipos pequeños. Equipos pequeños, hasta 5 integrantes, pueden obtener un número ilimitado de repositorios y 500 minutos de compilación. En la oferta empresarial, Butbucket cobra \$5 por usuario al mes, mientras que GitHub cobra \$21.

BENEFICIOS

- Bitbucket viene con Trello para que pueda comenzar con un gran rastreador de problemas gratuito o se puede aprovechar la existencia de una instancia de Jira. Al ser propiedad de Atlassian, Jira y Bitbucket, se integran en cada etapa del desarrollo, desde la creación hasta la implementación. Con componentes de seguimiento de errores integrados, Jira actualiza automáticamente la información sobre el problema detectado.
- Importación de proyectos Git existentes desde Excel, Github, entre otros.
- Condiciones especiales para estudiantes y profesores.

DESVENTAJAS

- No es de código abierto, pero admite proyectos de código abierto.

CREACIÓN DE LA CUENTA

- <https://github.com/>
- Datos solicitados:
 - Username
 - Email
 - Password



Join GitHub

Create your account

Username *

Email address *

Password *

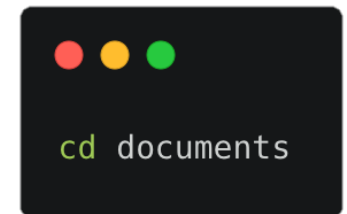
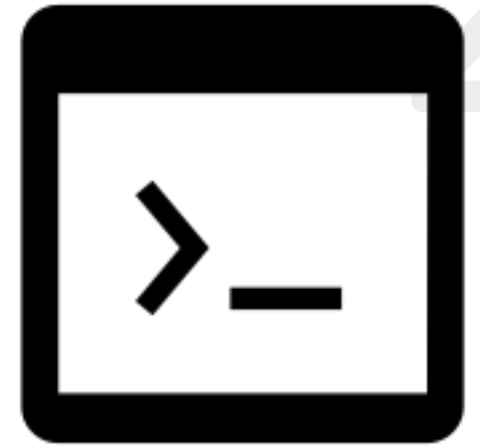
Make sure it's **at least 15 characters** OR **at least 8 characters including a number and a lowercase letter**. [Learn more](#).

NAVEGACIÓN POR CONSOLA

La consola de windows **en palabras humanas es una caja en donde podemos escribir comandos para ejecutar algunas acciones de forma no visual**, podemos navegar dentro de directorios, crear nuevos, borrar y muchas cosas mas. Pero en este tutorial veremos lo mas básico, navegar por el sistema, crear y eliminar directorios

CD [destino]

- Cambiamos de directorio



CD ../

- Regresamos un directorio atrás



CD /

- Regresamos al inicio de nuestra ruta, al disco duro



2

DIR



- Muestra los directorios que hay en nuestra posición actual



```
dir
```

CLS



- Limpiamos pantalla



```
cls
```



LS

- Muestra los directorios que hay en nuestra posición actual



```
ls
```



CLEAR

- Limpiamos pantalla



```
clear
```

A JUGAR

The Kahoot! logo is centered on a square background divided into four quadrants of different colors: red (top-left), blue (top-right), yellow (bottom-left), and green (bottom-right). Each quadrant contains a faint, stylized map of a world region. The word "Kahoot!" is written in a large, white, bold, sans-serif font across the center of the logo.

Kahoot!

2