# CSS Introducción.

## Linkar el archivo de estilos.

En la cabecera del documento (<head>) crear un link

<link rel=”stylesheet” href=”ruta archivo”

## Sintaxis:

selector {

propiedad:valor;

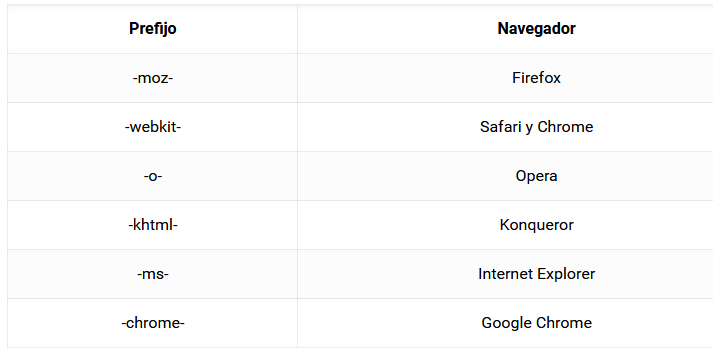
propiedad2: valor2;

……

}

## Prefijos: reglas destinadas a navegadores en concreto.

-webkit-valor

[Can I use](https://caniuse.com/)

Prefijos a las propiedades de CSS experimentales o no estándar y las API de JavaScript

EJEMPLO:

Para transformar un elemento en Firefox, es necesario utilizar la propiedad -moz-transform; en los navegadores basados en WebKit, como Safari y Google Chrome, se utiliza la propiedad -webkit-transform. Por este motivo, no es de extrañar que en algunos casos, tengamos que añadir hasta cuatro líneas de código para una única propiedad CSS.

## Selectores básicos:

1. **Universal :**

**\* {}**

1. **Básicos:**

**Etiqueta html=selector**

Para combinar selectores afectados por las mismas reglas separarlos por comas antes de la apertura de llaves.

*Ejemplo:* p, h1, h2, span{}

1. **Descendente:**

***Selector\_padre* (espacio\_en blanco) selector\_hijo** {

Las reglas afectan a los selectores de tipo hijo cuyo padre sea de tipo padre.

*Ejemplo:* p span a

1. **Clase:**

**. clase** {}

Esta clase tiene que estar definida en el html del selector con el atributo class=””

La misma clase puede ser utilizada por varias etiquetas y un elemento html puede tener más de una clase definida en el atributo class

1. **Restringir el selector clase:**

**selector.clase**

Las reglas afectarán a los selectores de ese tipo cuya clase sea clase

*Ejemplo:* a.destacado {

Las reglas afectarán a los selectores tipo a (enlaces) cuya clase sea destacado.

1. **Restringir afección regla selector clase multiple**

**.clase1.clase2**

Las reglas afectarán a los selectores que tengan al menos ambas clases (podría tener más, pero no menos de esas 2)

*Ejemplo:*

.error {}

.destacado {}

.especial {}

. error.destacado {}

Si una etiqueta html tuviera en class todas las clases, se aplicarían las reglass correspondientes a .error.destacado

1. **id:**

**#id{}**

El id estará asignado en alguna etiqueta html (que debe ser único en todo el documento)

## Selectores avanzados:

1. **Hijo directo:**

**selectorpadre>selector\_hijo**

Depende del selectorpadre (no aplica a un hijo del selector\_hijo)

*Ejemplo:*

p a 🡪 Cualquier enlace que esté dentro de un p (independiente de su situación/relación con el p.

p>a 🡪 Solo los enlaces que sean hijos directos del p

1. **Adyacente**

**selector1+selector2{}**

Selector1 y selector2 deben ser hermanos (mismo padre)

Selector2 debe aparecer inmediatamente después de selector1 (hermanos mellizos)

1. **Hermanos**

**selector1~selector2{}**

Todos los elementos de tipo selector2 que sean hermanos de selector1

1. **Selector de atributo**
2. **[atributo]=valor🡪 Todos los elementos que tengan el atributo = valor**

*Ejemplo*:

[color=”red”] {}

1. **selector [atributo]=valor🡪Todos los elementos de tipo selector que tengan el atributo=valor**

*Ejemplo*:

a[color=”red”] {}

1. **[atributo~=valor🡪 elementos que tienen en el atributo al menos el valor.**

*Ejemplo*:

[class ~=”externo”]{}

En el html puede haber varios elementos así que serían afectados:

<p class=”externo clásico verde”>XXXX</p>

También es válido: selector[atributo]~=valor

1. **[atributo|=”valor”]🡪 elementos que tienen el atributo = a una serie de palabras cuyo valor puede ser exactamente valor o puede comenzar con valor seguido inmediatamente por un guion.**

*Ejemplo:*

[class|=”ext”] 🡪 Todos los elementos cuya clase empiece por ext o ext-\* (ext, ext-erior, ext-erno,…..)

También es válido: selector[atributo]|=”valor”

1. **[atributo^=”valor”]🡪 Elementos cuyo atributo comienza exactamente por la subcadena valor.**

*Ejemplo:*

[class^=”ext”]🡪 Todos los elementos cuya clase empiece por ext (ext, exterior, externo,…)

1. **[atributo$=”valor”]🡪 Elementos con el atributo que termina con la subcadena valor**

*Ejemplo:*

[class$=”ado”]🡪Elementos cuya clase acaba en ado (ado, abollado, acabado, empezado,....)

1. **[atributo\*=”valor”] 🡪 Elementos con el atributo que contiene la subcadena valor**

*Ejemplo:*

[class\*=”ado”]🡪 Elementos cuya clase tiene ado en cualquier posición (ado, prados, ...)

1. **Pseudoclases: estados de los elementos elemento:estado{} (excepto :not)**
2. **:first-child: el primer hijo de un elemento**
3. **:last-child: el ultimo hijo de un elemento**
4. **: focus: se activa cuando el elemento tiene el foco**
5. **: hover: se eactiva al pasar el ratón por encima**
6. **:active: cuando se pulsa el ratón (imperceptible)**
7. **:visited: cuando ha sido visitado (enlaces)**

*Ejemplo1:*

Elemento a:

: link 🡪 estado normal

: visited🡪 visitado

: hover 🡪 cursor encima

: active🡪 click encima

a: link{color:blue}

a: visited{color:red}

*Ejemplo2:*

p em:first-child{}🡪 el primer elemento de tipo em (cursiva) que esté dentro de cualquier p

1. **:not(condición)🡪selecciona los que no cumplen la condición**

:not(#especial)🡪 selecciona el elemento que no tienen como id=especial

:not(p)🡪 Selecciona los elementos que no sean párrafos

:not(.externo)🡪 Selecciona los elementos que no tengan la clase externo

1. **:lang🡪 Seleccionar elementos según el idioma**

\*:lang(es)🡪 Seleccionar elementos que no estén en castellano

1. **:first-child🡪 selecciona el elemento con la condición de que sea el primer hijo (de quien sea de sus descendientes, pero primer hijo)**
2. **:last-child🡪 selecciona el elemento con la condición de que se el ultimo hijo**
3. **:nth-child(n)🡪 selecciona en el caso de que sea el hijo enésimo del padre**
4. **:nth-last-child(n)🡪 selecciona en el caso de que sea el hijo enésimo del padre contando desde el final**
5. **:empty🡪 selecciona en el caso de que no tenga ningún hijo**
6. **:nth-of-type(n)🡪 selecciona el elemento en el caso de que sea el enésimo elemento de este tipo**
7. **:nth-last-of-type(n) )🡪 selecciona el elemento en el caso de que sea el enésimo elemento de este tipo contando desde el final**
8. **Pseudoelementos: selector::pseudoelemento**
9. ::first-line 🡪 Selecciona primera línea del elemento
10. ::first letter🡪 Selecciona primera letra del elemento
11. ::before {content: “texto”}🡪 Añade el texto antes del selector
12. ::after{content: “texto”}🡪 Añade el texto después del selector
13. ::selection 🡪 Selecciona el texto seleccionado por el usuario