

# Width

Hace referencia al ancho de un elemento, es privativa de los elementos de bloque o de los elementos de línea bloque, es decir, puedo aplicar esta propiedad a un div, un header, como así también a una imagen, un elemento de formulario, **pero no a un vínculo, o a un strong o em porque estos son elementos de línea.**

Se puede trabajar con medidas de longitud absolutas, recuerden que son aquellas medidas que no dependen del dispositivo o entorno sino que siempre es lo mismo,

- cm
- mm
- pt
- pc
- in

# Width

También se puede trabajar con medidas relativas, medidas que sí varían dependiendo el dispositivo o entorno,

- px
- %
- em
- ex

En el caso de la propiedad font-size, él % se maneja con referencia a la tipografía base, **sin embargo en este caso, es el %(porcentaje) en referencia al ancho del contenedor padre, por ejemplo,**

```
#encabezado div {  
width: 30%;  
}
```

# Height

**El alto de un elemento** , es fijado a través de esta propiedad, donde se puede tener valores de medidas de longitud tanto absolutos como relativos.

**El height** , no es obligatorio y podemos omitir, siendo que los elementos sin height tienen como valor predeterminado : auto, por tanto eso significa que se adaptarán a su contenido.

```
#encabezado, #pie {  
width: 100%;  
background-color: lightgray;  
height: 90px ; }
```

# Height

De esta forma anterior, establecemos que el **#encabezado**, y el **#pie**, van a tener estas mismas características.

```
#encabezado, #pie {  
width: 100%;  
background-color: lightgray;  
height: 90px ; }
```

Otro detalle no menor, es que si por caso, colocamos un height pero el contenido **desborda al contenedor**, se verá en el navegador de la siguiente manera:

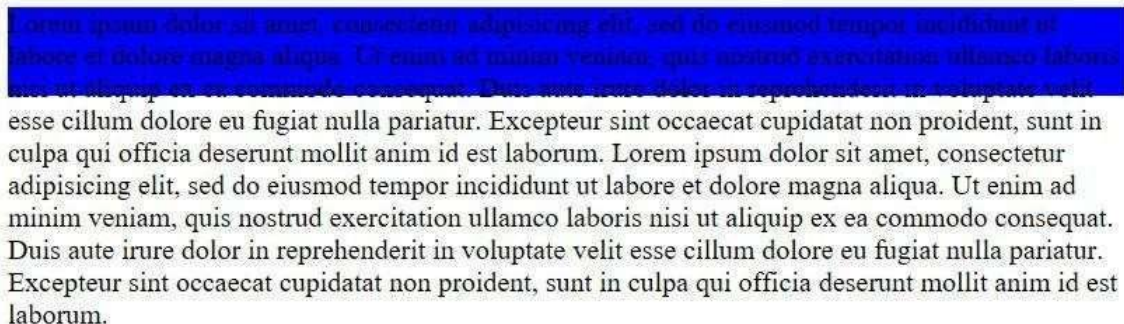
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nostrum, possimus alias quos recusandae hic odio, harum quae amet maiores

# Overflow

Está propiedad permite establecer con el contenido cuando desborda a su contenedor, valores posibles,

## #1 Visible

El contenido se visualiza por más que el contenedor sea más corto o el height no alcance a cubrir todo el contenido.

A screenshot of a text overflow example. It shows a paragraph of Latin text. The first line is highlighted in blue, indicating it is the visible portion of the text that fits within the container's height. The rest of the text is visible below the blue highlight, showing that the content is not truncated but simply extends beyond the initial container's height.

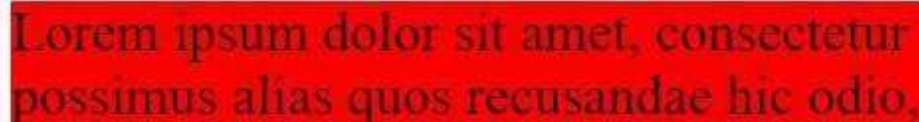
esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Overflow

Si el **height** , fuese o tuviese su valor predeterminado qué es auto, entonces este problema y está propiedad no se aplicarían porque el valor auto de hight **permite que el contenedor fluya** con su contenido por ejemplo

```
div { height:auto;  
      background-color: red; }
```

Se verá en el navegador de la siguiente forma,



# Overflow

## #Hidden,

Este valor, hace que el contenido que se desborda se enconda. Por ejemplo en nuestro editor se verá de la siguiente manera,

```
div {  
  height: 2em;  
  background-color: red;  
  overflow: hidden; }
```

En el navegador se verá de la siguiente forma,



Lorem ipsum dolor sit amet,

# Overflow

## #Overflow,

El valor scroll muestra ambas barras de scroll para poder ver el resto del contenido, sea necesario o no, por ejemplo,

```
div {  
  height: 2em;  
  background-color: red;  
  overflow: scroll; }
```

En el navegador se verá de la siguiente forma,





# Overflow

## #Auto

El valor auto, sólo muestra las barras de scroll si son necesarias,

```
div {  
  height: 2em;  
  background-color: red;  
  overflow: auto; }
```

En el navegador se verá de la siguiente forma,



# Margin

Esta propiedad  **fija un espacio entre elementos contiguos**, se puede trabajar con medidas de longitud

,

absolutas

- cm
- mm
- pt
- pc
- in

También se puede trabajar con medidas relativas,

- px
- %
- em
- ex

# Margin

Un ejemplo en nuestro código, de lado derecho con los cuatro lados, del otro lado la imagen qué se muestra es del uso de margin cómo shorthand

También tenemos el inspector de elementos que nos muestra elementos con **margin predeterminado** por ejemplo,

```
#encabezado {  
  width: 100%;  
  margin-top: 1em;  
  background-color: lightgray;  
}
```

```
body {  
  display: block;  
  margin: ▼ 8px;  
  margin-top: 8px;  
  margin-right: 8px;  
  margin-bottom: 8px;  
  margin-left: 8px;  
}
```

```
<!DOCTYPE html>  
<html>  
<head>  
  <title> CSS</title>  
  <style>  
  
    a {  
      margin-top: 12px;  
      margin-left: 10%;  
      margin-right: -12px;  
      margin-bottom: 10px;  
    }  
  
  </style>  
</head>  
<body>
```

# Margin

En muchos casos queremos quitar el margin a elementos de forma predeterminada, por caso

```
body {font: 90% 'Lato', sans-serif; margin:0;}
```

Otro ejemplo podría ser,

```
h1 { font-size: 3em; margin: 0; }
```

También hay otros ejemplos de shorthand, en este caso **el primer valor es margin-top, el segundo margin left y right, y el tercero margin-bottom**

```
#seccion-principal img {  
  width: 50%;  
  float: left;  
  margin: 1em 1% 2em;  
}
```

# Margin

Si trabajamos con dos valores el primero es **margin-top** y **margin-bottom**, el segundo valor es **margin-left** y **margin-right**,

```
#seccion-principal img {  
width: 50%;  
float: left;  
margin: 1em 1%;  
}
```

Si colocamos cuatro valores será **como las agujas del reloj**,

```
#seccion-principal img {  
width: 50%;  
float: left;  
margin: 1em 1% 2em 2%;  
}
```

# Padding

Esta propiedad indica el espacio entre los bordes de un elemento y su contenido. A **diferencia del margin** , **pocos elementos tienen padding**, por ejemplo las listas tanto ordenadas como desordenadas tienen , si queremos quitarles este espacio no tendremos más que,

```
<!DOCTYPE html>
<html>
<head>
  <title> CSS</title>
  <style>

<style>

ul, ol { margin: 0; padding: 0;}

</style>

</style>
</head>
```

# Padding

Tiene en cuanto a su sintaxis la misma construcción que margin.

En este caso, los cuatro lados son iguales. O si trabajamos con 4 valores distintos, debemos seguir el sentido de las agujas del reloj,

```
#encabezado, #pie {  
width: 100%;  
background-color: lightgray;  
height: 90px ;  
padding: 1%;  
}
```

Por otro lado, es importante saber, que el padding suma al total del ancho de un elemento, por ejemplo si en nuestra estructura ponemos **1% de padding al encabezado pero al main no, por más que ambos de width tengan 100%.**

# Padding

El ejemplo anterior se verá así,



Es decir uno queda más grande que el otro, pues **ahora el #encabezado, no es más 100% sino 100% más 1% de padding de ambos lados.**

De esta forma, entonces, debemos implementar una propiedad que permite hacer que el padding forme parte del width y no que se suma, esta propiedad es **box-sizing**



# box-sizing

Esta propiedad tiene dos valores posibles que explicarán su uso,

- **content-box**

El width total == width + padding + border

- **border-box**

El width total = width(padding + border)

```
* {box-sizing: border-box;}
```

# border-style

Es el estilo del borde tiene los siguientes valores,

- dotted – Borde un punto al lado del otro
- dashed -Borde una línea al lado de la otra
- solid – Borde Sólido
- double – Borde Doble
- groove – Borde 3d
- ridge – Borde 3d
- inset – Borde hacia adentro
- outset – Borde hacia afuera
- none – No hay borde

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>

  <style>
input { border-style: none; }

  </style>
</head>
<body>

<input type="text">

</body>
</html>
```

# border-width

Me permite establecer el **grosor o ancho del borde**, acá podemos trabajar con cualquier medida de longitud de las ya vistas **en otras propiedades como width, height o padding**.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>

  <style>
div { border-width: 1em; border-style: solid; }

  </style>
</head>
<body>

  <div>
    <p> div con borde </p>

  </div>

</body>
</html>
```

# border-color

El color del borde se puede aplicar con cualquier valor de los ya conocidos, **valores hexadecimales, palabras o también cantidad de rgb() o rgba()**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>

  <style>
div { border-width: 1em; border-style: solid; border-color: rgba(0,0,0,0.5); }

  </style>
</head>
<body>

<div>
  <p> div con borde </p>
</div>

</body>
</html>
```

# outline

El outline es un **borde luego del border**, por eso toma el nombre de outline, y está por fuera del elemento, si bien al principio puede parecer una propiedad un tanto extraña.

Cobra sentido porque está **de forma predeterminada en elementos de formularios** tales como el campo de texto que un poco más arriba elaboramos.



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>

  <style>
input { border-style: none; }

  </style>
</head>
<body>

<input type="text">

</body>
</html>
```

# outline

El outline cómo el border posee las mismas propiedades es decir:

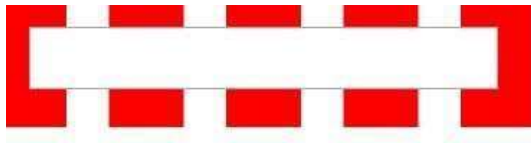
**#1 *outline-width***

**#2 *outline-style***

**#3 *outline-color***

```
<style>  
input { outline-width: 1em; outline-color: red; outline-style: dashed ; }  
</style>
```

En el navegador se verá de la siguiente manera,

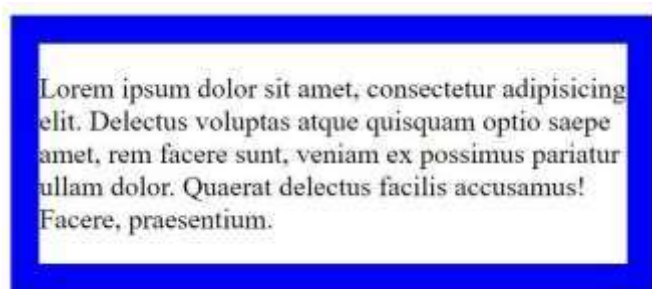


# outline

Luego también ambas propiedades tienen shorthands, que si bien son fáciles de aprender al poseer tantas propiedades **relacionadas (width, style, color)** las posibilidades son amplias, veamos algunos ejemplos,

```
div { width: 50%; border: 1em solid blue;}
```

En el navegador se verá de la siguiente manera,



# outline y border

A tener cuidados, porque si olvidamos poner el style en el shorthand, el resultado será que el borde desaparecerá, ya que el width indicarlo o el color no es obligatorio pero sí el style, por eso la siguiente regla está mal,

```
<style>  
div { width: 50%; border: 1em blue;}  
  
</style>
```

También puedo encarar solo un lado, por ejemplo, si quiero lograr el siguiente resultado:

- Lorem ipsum dolor sit amet, consectetur adipisicing elit. Delectus voluptas atque quisquam optio saepe
- amet, rem facere sunt, veniam ex possimus pariatur ullam dolor. Quaerat delectus facilis accusamus!
- Facere, praesentium.



# outline y border

En el ejemplo anterior desde el editor se verá de la siguiente forma,

```
<style>
div { width: 50%; border-left: 1em dotted red;}

</style>
```

También podemos lograr lo siguiente,

```
<style>
div { width: 50%; border-width: 1em 2%;
      border-color: blue red green orange;
      border-style: solid outset inset;

      }

</style>
```

# outline y border

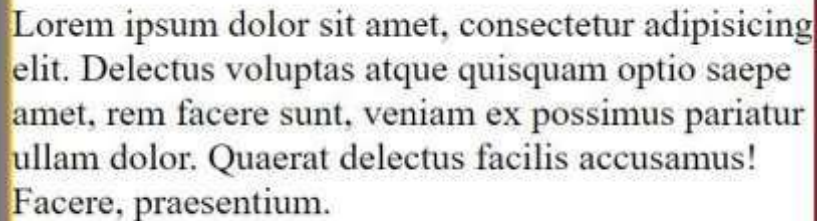
En el navegador lograremos el siguiente resultado,

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Delectus voluptas atque quisquam optio saepe amet, rem facere sunt, veniam ex possimus pariatur ullam dolor. Quaerat delectus facilis accusamus! Facere, praesentium.

# outline y border

Si combinamos **outline** y **border**,  
lograremos el siguiente resultado  
en el navegador,

```
<style>
div { width: 50%;
      border-width: 1em 2%;
      border-color: blue red green orange;
      border-style: solid outset inset;
      outline: 0.4em solid rgba(0,0,0,0.5);
    }
</style>
```



Lorem ipsum dolor sit amet, consectetur adipisicing  
elit. Delectus voluptas atque quisquam optio saepe  
amet, rem facere sunt, veniam ex possimus pariatur  
ullam dolor. Quaerat delectus facilis accusamus!  
Facere, praesentium.