POSICIONAMIENTO Y VISIBILIDAD

Sumario

1 Posicionamiento

https://www.w3schools.com/css/css_positioning.asp

La propiedad position especifica el tipo de método de posicionamiento utilizado para un elemento.

Hay cinco valores de posición diferentes:

- static
- relative
- fixed
- `absolute`
- sticky

Los elementos se colocan utilizando las propiedades top, bottom, left, and right.
Es necesario haber definido primero la posición, ya que funcionan de manera diferente dependiendo de ésta.

1.1 position:static

Los elementos HTML se colocan estáticos por defecto.

Los elementos posicionados estáticamente no se ven afectados por las propiedades top, bottom, left, and right. .

Un elemento con `position: static;` no se coloca de ninguna manera especial.

Siempre se posiciona de acuerdo con el flujo normal de la página:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.static {
  position: static;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: static;</h2>

<p>An element with position: static; is not positioned in any special way; it is
always positioned according to the normal flow of the page:</p>

<div class="static">
  This div element has position: static;
</div>

</body>
</html>
```

1.2 position: relative;

Un elemento con `position: relative;` se coloca en relación con su posición normal.

La configuración de las propiedades superior, derecha, inferior e izquierda de un elemento relativamente posicionado hará que se ajuste fuera de su posición normal. El otro contenido no se ajustará para encajar en el espacio dejado por el elemento.

Aquí está el CSS que se utiliza:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: relative;</h2>

<p>An element with position: relative; is positioned relative to its normal position:</p>

<div class="relative">
This div element has position: relative;
</div>

</body>
</html>
```

1.3 position: fixed

Un elemento con `position: fixed;` está posicionado en relación con la ventana gráfica, lo que significa que siempre permanece en el mismo lugar incluso si la página está desplazada. Las propiedades superior, derecha, inferior e izquierda se utilizan para colocar el elemento.

Un elemento fijo no deja un espacio en la página donde normalmente estaría ubicado.

Aquí está el CSS que se utiliza:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>


<h2>position: fixed;</h2>


<p>An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled:</p>


<div class="fixed">
This div element has position: fixed;
</div>
```

```
</body>
</html>
```

1.4 position:absolute

Un elemento con `position: absolute;` se posiciona en relación con el antepasado posicionado más cercano (en lugar de posicionarse en relación con la ventana gráfica, como en el fijo).

Sin embargo; Si un elemento posicionado absoluto no tiene ancestros posicionados, usa el body del documento y se mueve junto con el desplazamiento de la página.

Nota: Un elemento "posicionado" es uno cuya posición es cualquier cosa excepto `static`.

Aquí está el CSS que se utiliza:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  width: 400px;
  height: 200px;
  border: 3px solid #73AD21;
}

div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 200px;
  height: 100px;
  border: 3px solid #73AD21;
}
```

```
</style>
</head>
<body>

<h2>position: absolute;</h2>

<p>An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed):</p>

<div class="relative">This div element has position: relative;
  <div class="absolute">This div element has position: absolute;</div>
</div>

</body>
</html>
```

1.5 position: sticky;

Un elemento con `position: sticky;` se posiciona en función de la posición de desplazamiento del usuario.

Un elemento sticky alterna entre `relative` y `fixed`, dependiendo de la posición de desplazamiento. Se coloca en una posición relativa hasta que la posición de desplazamiento indicada se alcanza en la ventana gráfica, luego se "pega" en ese punto (como en la posición: fija).

Nota: *Internet Explorer, Edge 15 y versiones anteriores no admiten el posicionamiento fijo. Safari requiere un prefijo -webkit-. También se debe especificar al menos uno de* `top`, `right`, `bottom` *o* `left` *para que funcione.*

```
<!DOCTYPE html>
<html>
<head>
<style>
div.sticky {
```

```
  position: -webkit-sticky;
  position: sticky;
  top: 0;
  padding: 5px;
  background-color: #cae8ca;
  border: 2px solid #4CAF50;
}
</style>
</head>
<body>

<p>Try to <b>scroll</b> inside this frame to understand how sticky positioning works.</p>
<p>Note: IE/Edge 15 and earlier versions do not support sticky position.</p>

<div class="sticky">I am sticky!</div>

<div style="padding-bottom:2000px">
  <p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.</p>
  <p>Scroll back up to remove the stickyness.</p>
  <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
  <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
</div>

</body>
```

```
</html>
```

1.6 Elementos superpuestos

Cuando los elementos están posicionados, pueden superponerse a otros elementos.

La propiedad `z-index` especifica el orden de apilamiento de un elemento (qué elemento debe colocarse delante o detrás de los demás).

Un elemento puede tener un orden de pila positivo o negativo:

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<img src="w3css.gif" width="100" height="140">
<p>Because the image has a z-index of -1, it will be placed behind the text.</p>

</body>
</html>
```

Un elemento con un orden de apilamiento mayor está siempre delante de un elemento con un orden de apilamiento inferior.

Nota: `z-index` solo funciona en elementos posicionados (posición: absoluta, posición: relativa o posición: fija).

Nota: *Si dos elementos posicionados se superponen sin un* `z-index` *especificado, el elemento que se coloca en último lugar en el código HTML se mostrará en la parte superior.*

1.7 Posicionar texto en una imagen

Realiza el último ejercicio de la página https://www.w3schools.com/css/css_positioning.asp y observa como funciona.

2 La propiedad display

La propiedad `display` es una de las propiedades CSS más importantes para controlar el diseño.

La propiedad `display` especifica si / cómo se muestra un elemento.

Cada elemento HTML tiene un valor de visualización predeterminado según el tipo de elemento que sea. El valor de visualización predeterminado para la mayoría de los elementos es `block` o `inline`.

A continuaciín repasamos los elementos en bloque y en línea que ya vimos con anterioridad

2.1 Elementos a nivel de Bloque

Un elemento de nivel de bloque siempre comienza en una nueva línea y ocupa todo el ancho disponible (se extiende hacia la izquierda y hacia la derecha todo lo que pueda).

Ejemplos:

-
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

2.2 Elementos en linea

Un elemento en línea no se inicia en una nueva línea y solo ocupa el ancho necesario.

Ejemplos de elementos en línea:

-
- <a>
- <img>

Anular el valor de visualización predeterminado

Como se mencionó, cada elemento tiene un valor de visualización predeterminado. Sin embargo, puede anular esto.

Cambiar un elemento en línea a un elemento de bloque, o viceversa, puede ser útil para hacer que la página se vea de una manera específica y seguir los estándares web.

Un ejemplo común es hacer `<li>`elementos en línea para menús horizontales:

```
li {
  display: inline;
}
```

Nota: la configuración de la propiedad de visualización de un elemento solo cambia la **forma en que se muestra el elemento** , NO el tipo de elemento que es. Por lo tanto, un elemento en línea `display: block;` no tiene permitido tener otros elementos de bloque dentro de él.

El siguiente ejemplo muestra los elementos como elementos de bloque:

```
span {
  display: block;
}
```

El siguiente ejemplo muestra <a> elementos como elementos de bloque:

```
a {
  display: block;
}
```

Con el valor display: inline-block el elemento es formateado como in-line, pero puedes aplicarle valores de alto y de ancho.

2.3 display: inline-block

La diferencia principal con display:inline es que `display: inline-block` permite establecer una anchura y altura en el elemento.

Además, con `display: inline-block`, los márgenes superior e inferior y el padding son respetados, pero con `display: inline` no lo son.

La principal diferencia con display:block es que `display: inline-block` no añade un salto de línea después del elemento, por lo que el elemento puede sentarse al lado de otros elementos.

El siguiente ejemplo muestra el diferente comportamiento de `display: inline`, `display: inline-block` y `display: block`:

```
<!DOCTYPE html>

<html>
```

```
<head>
<style>
span.a {
  display: inline; /* the default for span */
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

span.b {
  display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

span.c {
  display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
<body>
```

```
<h1>The display Property</h1>


<h2>display: inline</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. <span class="a">Aliquam</span> <span class="a">venenatis</span> gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>


<h2>display: inline-block</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. <span class="b">Aliquam</span> <span class="b">venenatis</span> gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>


<h2>display: block</h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. <span class="c">Aliquam</span> <span class="c">venenatis</span> gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet. </div>


</body>

</html>
```

Puedes ver el resto de los valores de la propiedad display en https://www.w3schools.com/cssref/pr_class_display.asp

2.4 Esconder un elemento - display:none or visibility:hidden?

Se puede ocultar un elemento estableciendo la propiedad display en none. El elemento se ocultará y la página se mostrará como si el elemento no estuviera allí:

```
h1.hidden {
  display: none;
}
```

`visibility:hidden;` También esconde un elemento.

Sin embargo, el elemento seguirá ocupando el mismo espacio que antes. El elemento estará oculto, pero aún afectará el diseño:

```
h1.hidden {
  visibility: hidden;
}
```

Realiza el útimo ejercicio de la página https://www.w3schools.com/css/css_display_visibility.asp para probar su funcionamiento.

2.5 El uso de inline-block para crear vínculos de navegación

Un uso común para `display: inline-block` es para mostrar los elementos de la lista horizontalmente en lugar de verticalmente. El siguiente ejemplo crea enlaces de navegación horizontales:

```
.nav {
  background-color: yellow;
  list-style-type: none;
  text-align: center;
  padding: 0;
  margin: 0;
}

.nav li {
  display: inline-block;
  font-size: 20px;
  padding: 20px;
}
```

3 Visibility

La propiedad `visibility` especifica si un elemento es visible.

```
h2.a {
  visibility: visible;
}

h2.b {
  visibility: hidden;
}
```

Nota: Si ocultamos elementos con la propiedad visibility estos ocuparán espacio en la página. Utiliza la propiedad display para ocultar y eliminar a la vez un elemento de la presentación del documento.

4 Overflow

La propiedad `overflow` controla lo que ocurre con el contenido que es demasiado grande para caber en un área.

La propiedad `overflow` especifica si se recorta el contenido o añadir barras de desplazamiento cuando el contenido de un elemento es demasiado grande para caber en un área especificada.

La propiedad `overflow` tiene los siguientes valores:

- `visible`- Defecto. El desbordamiento no se recorta. Se hace fuera de la caja del elemento
- `hidden` - El desbordamiento se recorta, y el resto del contenido será invisible
- `scroll` - El desbordamiento se recorta, pero se agrega una barra de desplazamiento para ver el resto del contenido
- `auto` - Si se recorta desbordamiento, una barra de desplazamiento, debe añadirse a ver el resto del contenido

Las propiedades `overflow-x`y `overflow-y` especifican si se debe cambiar el desbordamiento del contenido horizontal o verticalmente (o ambos):

`La propiedad overflow-x` especifica qué hacer con los bordes izquierdo / derecho del contenido.
La propiedad `overflow-y` especifica qué hacer con los bordes superior / inferior del contenido.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow-x: hidden;
  overflow-y: scroll;
}
</style>
</head>
```

```
<body>


<h2>CSS Overflow</h2>

<p>You can also change the overflow of content horizontally or vertically.</p>

<p>overflow-x specifies what to do with the left/right edges of the content.<br>

overflow-y specifies what to do with the top/bottom edges of the content.</p>


<div>You can use the overflow property when you want to have better control of the layout. The overflow property specifies what happens if content overflows an element's box.</div>


</body>

</html>
```

5 Otros

Revisa como hacer:

- Estilos de enlaces https://www.w3schools.com/css/css_link.asp
- Barras de navegación https://www.w3schools.com/css/css_navbar.asp