**¿Qué es exactamente un botón?**

os botones son elementos para hacer clic que le proporcionan algún tipo de interacción al usuario. Pueden ayudarlo a navegar por una página, descargar archivos o enviar formularios, sólo por mencionar algunos de sus usos. Sabrás que es un botón porque *se ve como uno*.

Hay tres opciones: los elementos <a>, <input> y <button>. Los tres pueden verse y comportarse exactamente igual, por lo que la elección entre ellos se reduce a la semántica; es decir, **¿para qué quieres usar el botón?**

os elementos de anclaje son para la navegación. Si deseas que las personas vayan a algún lado o descarguen un archivo haciendo clic en tu botón, utiliza las etiquetas <a>.

Por otro lado, los elementos <input> y <button> se usan para los formularios. Representan algún tipo de entrada de datos (como enviar o restablecer el formulario). Úsalos si lo que quieres es agregar botones a algún formulario HTML.

.btn {

display: inline-block;

text-decoration: none;

font-family: Arial, sans-serif;

text-align: center;

font-size: 14px;

line-height: 1.42857143;

padding: 6px 12px;

margin-bottom: 10px;

border: 1px solid transparent;

border-radius: 4px;

white-space: nowrap;

vertical-align: middle;

}

Esto se ve más complicado de lo que parece. Déjame explicarte qué hace cada regla:

* El display: inline-block es crucial. Los elementos de anclaje son inline por defecto, lo cual significa que no pueden aceptar ninguna propiedad del modelo de cajas (como padding o margin). Esta regla cambia eso y nos permite cambiar el elemento a un botón.
* Las siguientes cinco reglas (text-decoration, font-family, text-align, font-size y line-height) son para la estilización básica de la tipografía.
* El padding le da al texto algo de espacio para que el botón no se vea abarrotado.
* La regla margin-bottom creará algo de espacio debajo del botón. No tocará otros elementos, si es que se apila en una pantalla más pequeña.
* Las propiedades border y border-radius crean un borde redondeado y transparente. Ahorita no lo puedes ver debido al valor transparent, pero aparecerá tan pronto como le demos color con las clases contextuales.
* La propiedad white-space: nowrap previene que el texto se rompa en una nueva línea dentro del elemento (lo cual sería un problema con los botones de texto más largos).
* Y para terminar, vertical-align: middle alineará el botón al centro vertical de la línea en la que actualmente se encuentra.

**Escribir el HTML**

Esto es tan sencillo como parece. Sólo dos elementos de anclaje con una clase button-nav:

<a href="#" class="button-nav">Previous</a>

<a href="#" class="button-nav">Next</a>

HTML

Copy

**Agregar los estilos predeterminados**

No hay versiones para la base y el contexto por ahora; ambos botones emplean el mismo diseño. Todavía tendremos que definir un hover state, pero comencemos primero con el estado que no es hover:

.button-nav {

display: inline-block;

font-size: 1rem;

font-family: Arial, sans-serif;

text-decoration: none;

color: #000;

padding: .8rem 1rem;

border: 1px solid #000;

}

CSS

Copy

Una vez más, tenemos que cambiar el modo de visualización a bloque en línea y luego darle un estilo al texto dentro de los anclajes. Después de eso, algo de relleno y un borde. Observa dos cosas: una es que el relleno es más pequeño en la parte superior e inferior, lo que garantiza que los botones se vean alargados. La otra, utilicé rems en lugar de pixeles (excepto para el borde).

Aunque los botones generalmente no cambian mucho entre los viewports, tenemos que diseñarlos sin olvidar la accesibilidad. El uso de rems garantiza que los botones puedan ajustarse si un usuario desea aumentar el tamaño de la fuente base.

**Agregar el Hover State**

Ya tenemos dos botones bonitos, pero vamos a darle más vida con un hover state invertido y una sutil animación de transición:

.button-nav:hover {

color: #fff;

background-color: #000;

transition: background-color .15s ease-in-out;

}

Las primeras dos propiedades te resultarán familiares, pero transition es totalmente nueva. Desplázate sobre los botones y observa cómo el cambio entre los dos estados parece más suave. Puedes eliminar la regla temporalmente y ver la transición predeterminada. Se ven entrecortadas al compararlas, ¿verdad? O cambia .15s por algo más asombroso, como 3s.

En efecto, esta es la propiedad transition (es decir, transición). Si un elemento tiene dos estados (como nuestro botón), esta propiedad puede animar la transición entre ambos. La transición es una abreviatura ("shorthand", en inglés) y toma múltiples valores. En este caso, le dimos tres:

* transition-property: la propiedad que debe aplicarse para que aparezca el efecto de transición (en este caso, background-color),
* transition-duration: la duración de la animación (en este caso, .15s),
* transition-timing-function: el curso de la velocidad del efecto (en este caso, ease-in-out crea una animación con un inicio y cierre lentos).

Es una característica muy buena que de seguro utilizarás mucho.

# Barra de navegación

**El esquema para cada navegación (HTML NAVBAR)**

El primer paso es (claro) el scaffold HTML. La navegación tienen su propia etiqueta semántica en HTML5: el elemento <nav>. Cuando construyas una navbar, empieza con esto:

<nav>

<!-- Here comes the navigation. -->

</nav>

HTML

Copy

Genial, ahora puedes agregar los enlaces de navegación. A continuación, te explicaré dos formas para hacerlo.

Opción A: puedes empezar y agregar una lista desorganizada con etiquetas a (pues técnicamente es una lista de elementos de navegación):

<nav>

<ul>

<li><a href="#">Link 1</a></li>

<li><a href="#">Link 2</a></li>

<li><a href="#">Link 3</a></li>

<li><a href="#">Link 4</a></li>

</ul>

</nav>

HTML

Copy

Nota: En este caso, acuérdate de quitarle el aspecto predeterminado a esta lista (es decir, eliminar las viñetas, mostrarla en línea, etc.)

Opción B: Por el contrario, puedes ahorrarte las molestias que traen estas listas; los elementos de anclaje estarán bien por sí solos:

<nav>

<a href="#">Link 1</a>

<a href="#">Link 2</a>

<a href="#">Link 3</a>

<a href="#">Link 4</a>

</nav>

**Estilos de base**

Como estamos trabajando con el método de la lista, lo primero sería anular cualquier estilo predeterminado (como las viñetas y demás). Pero el reset de Meyer ya se encargó de eso, así que podemos omitirlo. Tampoco tenemos que cambiar la posición del nav, ya que estamos construyendo para dispositivos móviles, y esta configuración de apilación vertical es perfecta para eso.

Genial, esto significa que ya podemos saltar directo al paso dos y diseñar los elementos de navegación. Comencemos con los estilos móviles predeterminados.

**<ul>**

Comencemos con la etiqueta ul:

/\* ==== GENERAL STYLES ==== \*/

\* {

box-sizing: border-box;

}

/\* ==== A SIMPLE TOP NAVIGATION ==== \*/

nav ul {

background-color: #eee;

text-align: center;

}

+1 consejo: Observa cómo el fondo gris ocupa todo el ancho de la ventana. ¿Recuerdas por qué? Porque ul es, por defecto, un elemento en bloque.

Bueno, avancemos un poco más para estilizar los elementos de la lista.

**<li>**

Agrega este bloque debajo de las reglas anteriores:

nav li {

font-family: 'Anton', sans-serif;

font-size: 1.4rem;

line-height: 3rem;

height: 3rem;

border-bottom: 1px solid #ccc;

}

El menú de navegación comienza a tomar forma. Sin embargo, apareció un pequeño error: el código hace referencia a una fuente externa que no hemos vinculado hasta ahora. Arreglémoslo copiando esta línea en la sección head:

<link href="https://fonts.googleapis.com/css?family=Anton" rel="stylesheet">

Sucedieron un par de cosas interesantes en este último conjunto de reglas. Las propiedades line-height y height están configuradas intencionalmente con el mismo valor. Este pequeño truco de CSS centra verticalmente el texto dentro de los elementos. Mientras tanto, se utiliza la propiedad border-bottom para crear una separación visual entre los elementos.

Terminemos los estilos base con etiquetas de anclaje.

**<a>**

Copia este código al final de tu CSS:

nav a {

text-decoration: none;

color: #C74F78;

display: block;

}

El text-decoration: none; le quita a los enlaces su apariencia azul y subrayada, de esta manera podemos declarar un nuevo color en la siguiente línea. Sin embargo, la regla más importante es la display: block;. Gracias a ella, las anclas ahora ocupan todo el espacio dentro de las etiquetas li. Esto amplía el área para hacer clic a todo el ancho y alto del botón, lo que hace que sea más fácil darle clic a un vínculo de navegación.

**Estados especiales**

Todo buen enlace necesita de un hover state:

nav a:hover {

background-color: #fff;

}

Y algunas animaciones de transición bonitas:

nav a:hover {

background-color: #fff;

transition: background-color .3s ease-in-out;

}

Es una práctica común en los sitios de páginas múltiples marcar la pestaña que está actualmente activa en la barra de navegación. Digamos que ahora mismo estamos construyendo la página de inicio. Eso hace que el enlace "Inicio" (o home, en inglés) sea el activo. Añade entonces una clase .active al elemento ancla correspondiente en tu HTML:

<nav>

<ul>

<li><a href="#" class="active">Home</a></li>

<li><a href="#">News</a></li>

<li><a href="#">Blog</a></li>

<li><a href="#">About us</a></li>

<li><a href="#">Contact</a></li>

</ul>

</nav>

Y añade este al final de tu CSS:

nav a.active {

background-color: #c74f78;

color: #eee;

}

Voilá, un enlace activo. Nuestra barra de navegación está casi lista, sólo tenemos que agregar los estilos para la tablet y la computadora de escritorio.