

HTML y CSS (Parte II)

Añadiéndole CSS al proyecto

¿Qué es CSS?

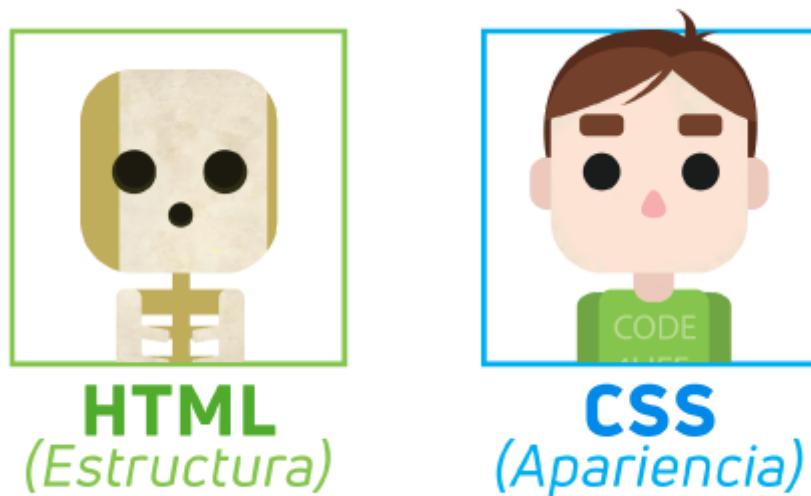


Imagen 1: HTML y CSS

Desde un punto de vista simple, HTML sería el esqueleto del personaje, la estructura. Por otro lado, CSS sirve para darle apariencia al contenido, colores, fuentes, tipografías, etc. Todo el estilo y apariencia visual de la página.

CSS significa "Cascading Style Sheets", que traducido es "hojas de estilo en cascada". En resumen CSS define cómo se mostrarán los elementos HTML en la pantalla, en papel o en otros medios especificados.

Dentro de este capítulo vamos a abrir el proyecto que creamos anteriormente y le vamos a añadir CSS.

Formas de añadir CSS

Existen varias formas de añadir CSS a las páginas:

- En línea en el HTML, añadiéndole el atributo `style` a las etiquetas.
- Añadiéndole la etiqueta `<style>` en el head.
- A través de un archivo de tipo CSS, llamado desde el head. **(RECOMENDADO)**

La mejor práctica es hacerlo través de un archivo externo al HTML de tipo `.css` llamándolo en nuestra etiqueta `<head>`.

Entonces, vamos a crear un nuevo archivo de tipo `.css`, pero, como ya lo vimos en el caso de las imágenes, es importante clasificar los distintos tipos de archivos y no dejarlos todos en la carpeta raíz.

Por lo tanto, dentro de nuestra ya creada carpeta **assets**, crearemos una carpeta llamada **css**. Ahí crearemos el archivo que llamaremos `style.css`.

Ahora, en nuestro archivo `index.html`, dentro de la etiqueta `<head>` vamos a colocar el siguiente código:

```
<link rel="stylesheet" href="assets/css/style.css">
```

La etiqueta es link. El atributo rel especifica la relación entre el documento actual y el documento vinculado, por lo tanto, su valor vendría siendo `stylesheet`. El atributo href, que ya lo hemos visto en otras ocasiones, especifica de dónde viene el documento, o sea, debemos darle la ruta al archivo `.css` que queremos vincular.

¿Nos funcionó?

Haremos una pequeña prueba...

```
body {  
    background: red;  
}
```

Guardaremos tanto nuestro archivo HTML como CSS y recargaremos el navegador.

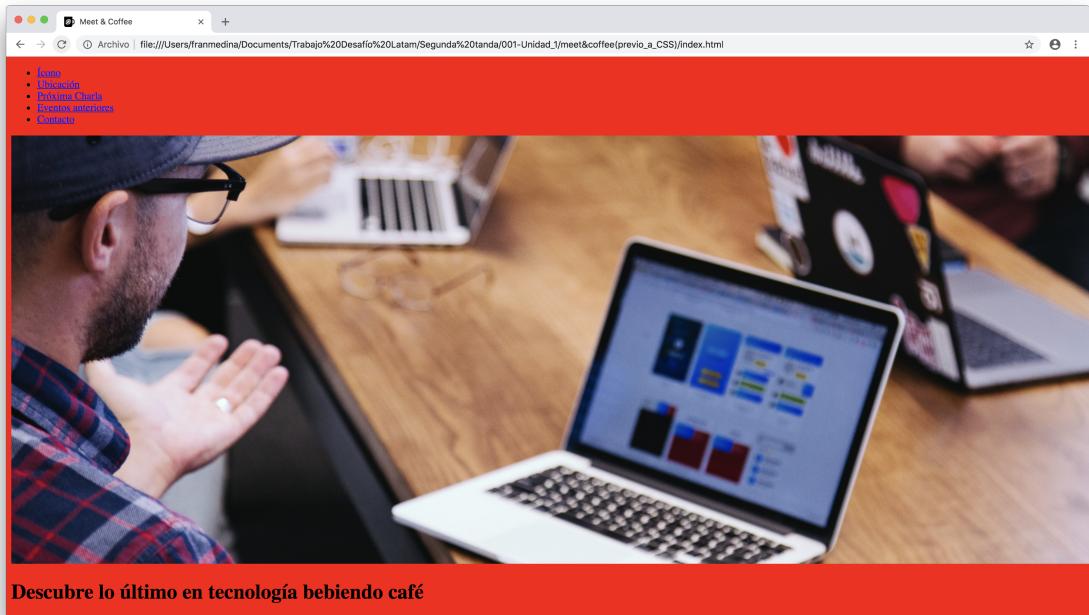


Imagen 2: Vincular HTML con CSS

Bueno, ahí lo tenemos. Nuestro archivo HTML ha sido vinculado con un CSS.

Introducción a CSS

Lo que hicimos en el video anterior fue asignarle una característica de tipo background a la etiqueta body.

Sintáxis

Analicemos un poco la sintáxis que escribimos, que es distinta a la sintáxis que ya conocimos de HTML.

Podemos identificar lo siguiente: un selector de etiqueta (`body`), las llaves curvas (`{ ... }`) que determinan el inicio y cierre del bloque de declaración, una declaración (`background: red;`) que está compuesta por una propiedad (`background`) y su valor (`red`).

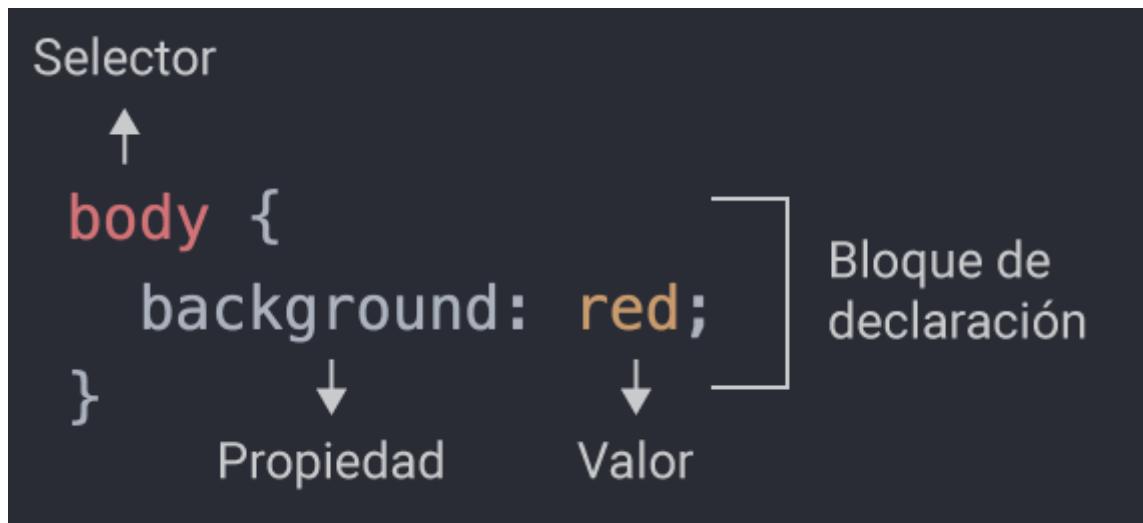


Imagen 3: Sintáxis

El selector indica qué elemento del HTML queremos modificar.

Dentro del bloque de declaración puede haber más de una declaración. Éstas se separan por un punto y coma.

Tipos de selectores

Hay diversos tipos de selectores. De momento vamos a ver los siguientes:

- Selector por *etiqueta*
- Selector por *id*
- Selector por *clase*

El **selector por etiqueta** es el que acabamos de revisar recién. Éste selecciona el elemento según su nombre de etiqueta.

Lo que hicimos fue darle estilo a la etiqueta `<body>`, la etiqueta general del contenido de nuestro HTML.

Ahora haremos una prueba dándole estilos a la etiqueta `<h2>`, la cual hemos escrito 4 veces en nuestro HTML. Le daremos dos declaraciones, la primera de el alineamiento del texto, la segunda del color de la letra.

```
h2 {  
    text-align: center;  
    color: blue;  
}
```

A la propiedad `text-align` le hemos puesto el valor de `center` y a la propiedad de `color` que especifica el color de letra le hemos puesto el valor `blue`, o sea, azul. Guardemos nuestro archivo CSS y recarguemos nuestro navegador.

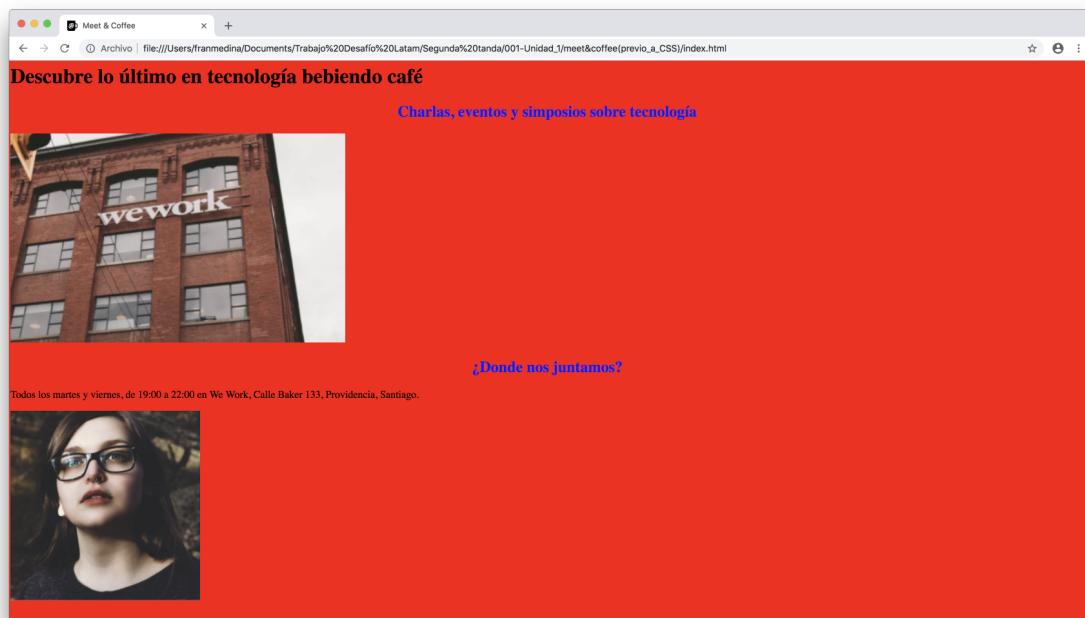


Imagen 4: Propiedad `text-align`

Ha cambiado, todos los textos de tipo `h2` han pasado de estar alineados a la izquierda (que es el valor por defecto) a estar centrados. Además, su letra se ha puesto de color azul, y no negra.

Además del selector por etiqueta existe el **selector por clase**. Las clases sirven para que varias etiquetas tengan características determinadas. El selector de clase selecciona todos los elementos con el mismo valor del atributo de clase.

Por ejemplo, queremos que algunos textos sean de color verde, entonces, vamos a atribuirle a algunas etiquetas el atributo `class` con valor `verde` :

```
<h1 class="verde">Descubre lo último en tecnología bebiendo café</h1>  
<h4 class="verde">Rafaela Valdés</h4>  
<h2 class="verde">Eventos anteriores</h2> .
```

Para seleccionar esa clase en específico en el CSS, debemos escribir un `.` seguido por el valor de la clase:

```
.verde {  
    /* Soy un comentario de CSS, no confundirse con comentarios de HTML. */  
}
```

Le podremos dar valores a las distintas propiedades de las etiquetas con esa clase "verde". Le pondremos que tenga la letra de color verde.

```
.verde {  
    color: green;  
}
```

Ahora, vemos que a todas las etiquetas con clase verde se les cambió el color de letra, incluyendo a la etiqueta `h2`, ya que la clase es más específica que la etiqueta por sí sola.

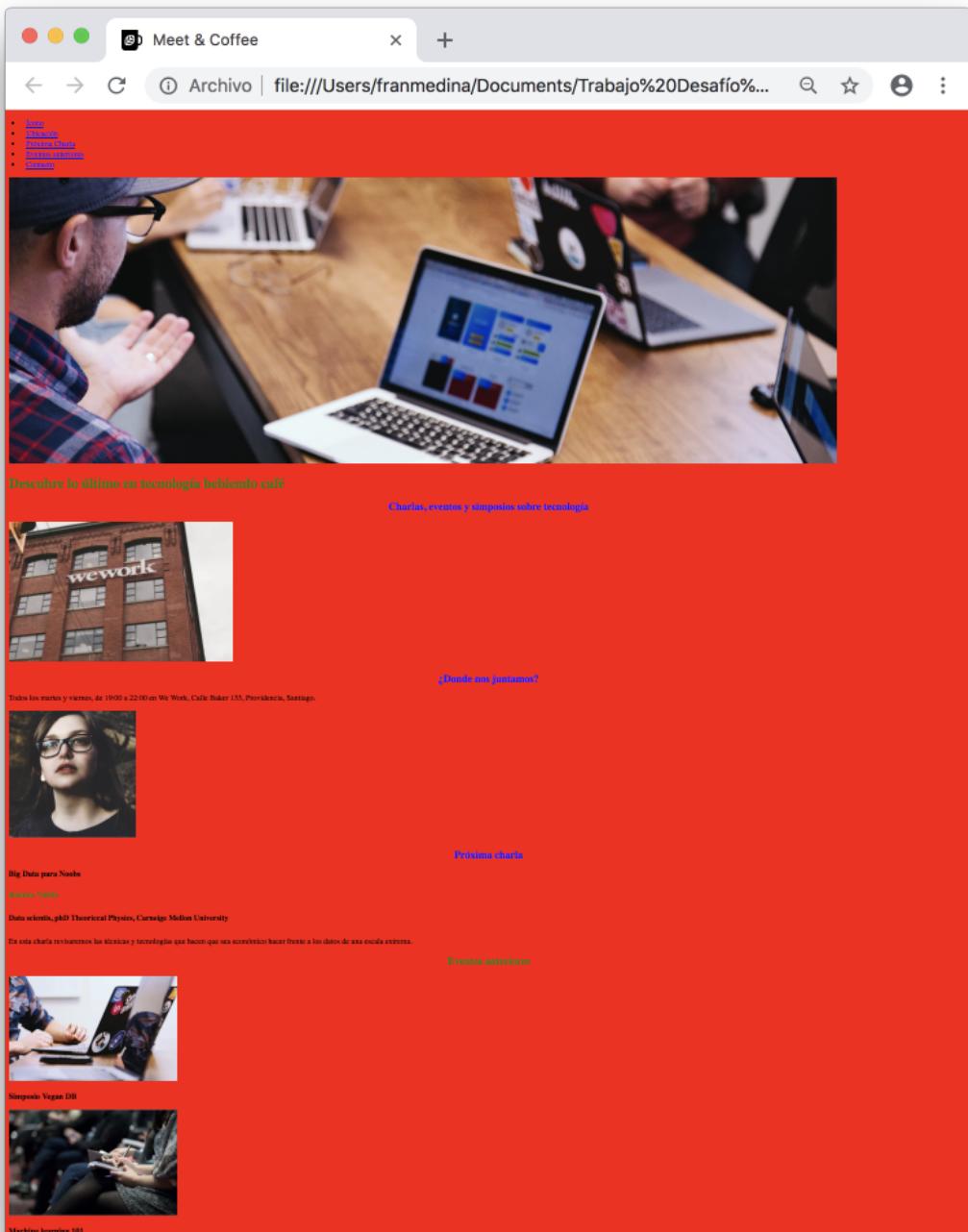


Imagen 5: Etiqueta con clase

Las etiquetas pueden tener más de una clase. Como ejemplo:

- A `h1` vamos a añadirle otra clase.
- Al atributo `class` le vamos a dar el valor `verde centrado`.

En el CSS especificaremos que todas las etiquetas con clase `centrado` tendrán un alineamiento de texto al centro.

```
<h1 class="verde centrado">Descubre lo último en tecnología bebiendo café</h1>
```

```
.centrado {  
    text-align: center;  
}
```

Para añadirle más de una clase a las etiquetas sólo hay que separarlas por espacio.

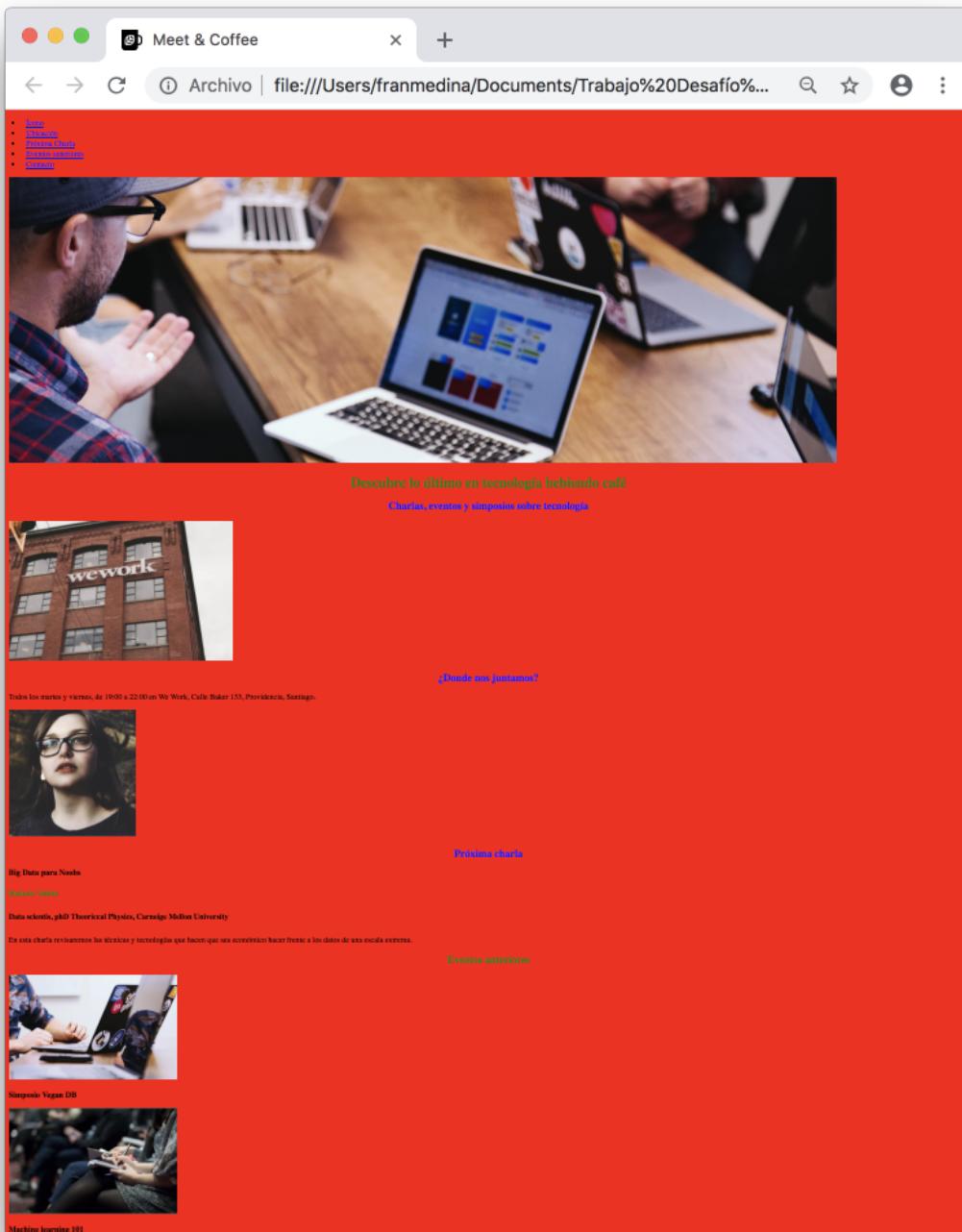


Imagen 6: Etiquetas con más de una clase

Además del selector por etiqueta y del selector por clase, existe el **selector por id**, que utiliza valor del atributo `id` de una etiqueta HTML para seleccionar un elemento específico.

El valor `id` de un elemento debe ser único, no puede repetirse en ninguna otra etiqueta dentro de la página. Además de que cada etiqueta puede tener como máximo un solo id.

Entonces, vamos a añadir un `id` a uno de los **h2**. Comencemos añadiéndoselo al primero, el que está justo por debajo del **h1**. Atributo `id`, valor `subtitulo`.

```
<h2 id="subtitulo">Charlas, eventos y simposios sobre tecnología</h2>
```

Para seleccionar ese id en específico en el CSS, debemos escribir un `#` seguido por el valor del `id`:

```
#subtitulo {  
    /* Soy un comentario de CSS, no confundirse con comentarios de HTML. */  
}
```

Ahora, le podremos dar valores a las distintas propiedades de ese **h2** con `id` "subtitulo". Le pondremos que tenga la letra de color blanco y que esté alienado a la derecha.

```
#subtitulo {  
    text-align: right;  
    color: white;  
}
```

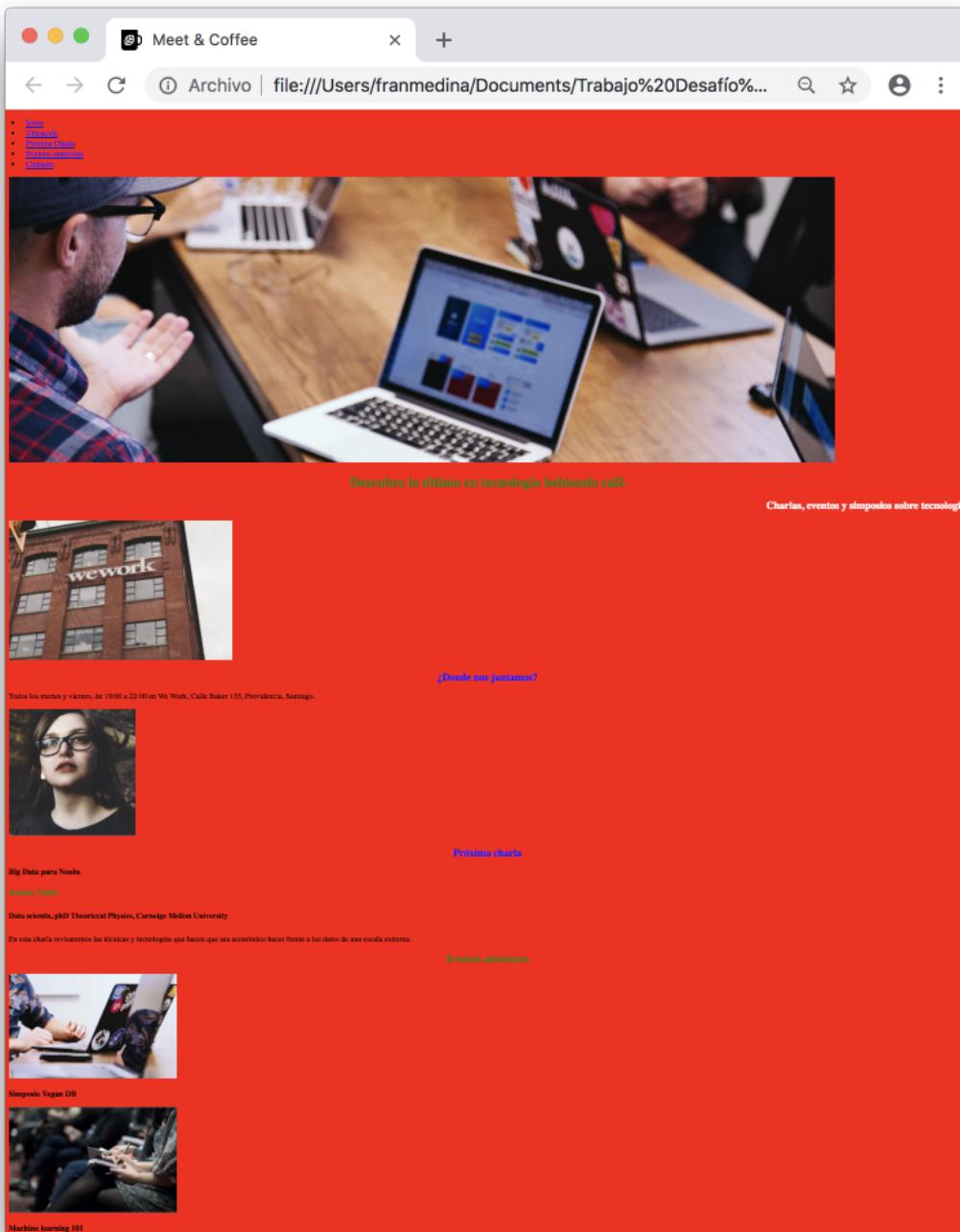


Imagen 7: Agregar propiedades al subtítulo

A lo largo de este módulo, trataremos de utilizar clases por sobre los id. La razón es, básicamente, por cómo se calcula el peso de su especificidad.

La etiqueta es menos específica que la clase, la clase es menos específica que el id.

Pues bien, si llevamos los selectores al mundo real, se podría decir que mi etiqueta sería "humano", mis clases serían mi nombre, mi edad, mi nacionalidad, entre otros y mi id sería mi rut.

Si quieres aprender más sobre los selectores te recomiendo este juego: [CSS Diner](#).

La cascada de CSS

En CSS existen dos reglas importantes.

Regla N°1: La primera es que **la última regla manda**. Por ejemplo, si yo declaro que un background-color es gris y más adelante específico que el color es negro, el resultado será el último que indicamos.

```
body {  
    background-color: grey;  
    padding: 0;  
    margin: 0;  
    background-color: black;  
}
```

Si lo vemos con el inspector de elementos, veremos que existen los dos background-color pero uno está mandando por sobre el otro, es decir **la última regla manda**.

Regla N°2: La segunda regla de oro de CSS es que **lo específico manda por sobre lo general**.

¿Qué quiere decir esto?

Si definimos que los textos dentro de un `div` en específico tengan la letra de color rojo, pero luego en su interior indicamos específicamente que el `h2` sea de color verde, ¿qué es lo que creen que sucederá?

Como dijimos, la regla más específica manda, por lo tanto las letras de la etiqueta serán de color verde. Si lo vemos en el inspector de elementos, podremos comprobar que para el `h2` hay una propiedad que se hereda que es el de letra de color rojo; pero tenemos otra propiedad más específica que indica que el color de letra debe ser verde.

```
div {  
    color: red;  
}  
  
div h2 {  
    color: green;  
}
```

Entonces, cuando estemos trabajando en CSS y no se vean los resultados que buscamos, siempre tenemos que tener en consideración esas dos reglas: **la última es la que manda y la más específica es la que manda**.

Dividiendo nuestra página en secciones

Los ejercicios que hicimos anteriormente fueron sólo por un tema pedagógico para conocer los selectores, porque nuestra página no debería verse así.

Por lo mismo, vamos a volver a cómo teníamos antes el HTML, pero dejándole el link al archivo CSS y vamos a dejar el archivo CSS vacío de momento.

Te dejo el código en la guía:

```
<!DOCTYPE html>
<html>
<head>
  <title>Meet & Coffee</title>
  <meta charset="utf-8">
  <link rel="shortcut icon" type="image/png" href="favicon.png">
  <meta name="author" content="Francisca Medina Concha">
  <meta name="description" content="Comparte tus conocimientos tomando café">
  <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
  <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>
  <ul>
    <li><a href="#">Ícono</a></li>
    <li><a href="#">Ubicación</a></li>
    <li><a href="#">Próxima Charla</a></li>
    <li><a href="#">Eventos anteriores</a></li>
    <li><a href="#">Contacto</a></li>
  </ul>

  
  <h1>Descubre lo último en tecnología bebiendo café</h1>
  <h2>Charlas, eventos y simposios sobre tecnología</h2>

  
  <h2>¿Dónde nos juntamos?</h2>
  <p>Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.</p>

  
  <h2>Próxima charla</h2>
  <h3>Big Data para Noobs</h3>
  <h4>Rafaela Valdés</h4>
  <h5>Data scientist, PhD Theoretical Physics, Carnegie Mellon University</h5>
  <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.</p>

  <h2>Eventos anteriores</h2>
  
  <h3>Simposio Vegan DB</h3>
  
  <h3>Machine learning 101</h3>
  
  <h3>Scrum sin scream</h3>
</body>
</html>
```

Dividiendo nuestra página en secciones

De momento, vamos a volver a trabajar duramente sobre el HTML. En este capítulo vamos a segmentar nuestra página web, con el objetivo de agrupar el contenido, darle sentido y forma a las diferentes secciones que la componen, para, en un siguiente video, darle el estilo CSS.

¿Recuerdas cómo la dividimos?

- Una barra de navegación
- Una sección principal, llamada **Hero section**
- Sección de lugar
- Sección de próxima charla
- Sección de eventos anteriores
- Sección final, de contacto, llamada **footer**

La etiqueta clásica para generar divisiones es la etiqueta `<div>`.

Si vamos a cualquier sitio, por ejemplo [Ebay](#) y analizamos su página principal, podremos ver visualmente que tiene diferentes secciones de información. Si damos click derecho y apretamos **inspeccionar** podemos ver su código HTML. Ahí podremos darnos cuenta que dentro de su `<body>` hay varias etiquetas `<div>` que agrupan la información.

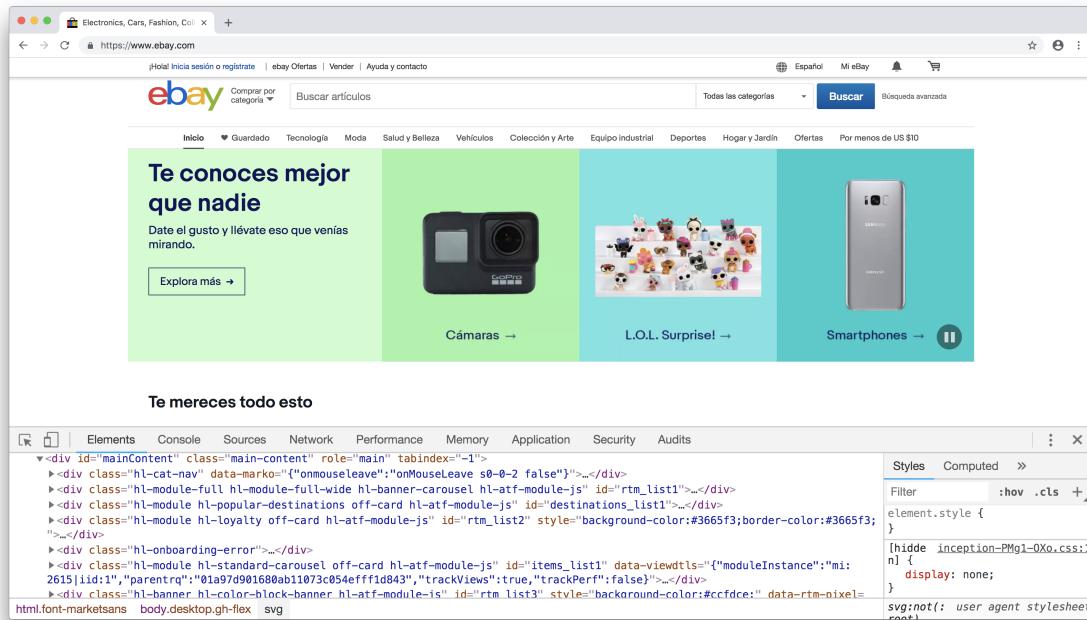


Imagen 8: Etiqueta `<div>`

La etiqueta `div` por sí misma sólo divide el contenido, pero no tiene semántica.

¿A qué se refiere con esto?

La semántica es el estudio de los significados. En el caso del `div`, éste sólo divide contenido, pero no se le atribuye ninguna carga de significado específica.

HTML5 -el estándar de hoy- tiene etiquetas que cumplen la misma función de dividir que el `div`, pero que le atribuyen semántica al contenido.

Nosotros usaremos: `<nav>`, `<header>`, `<section>` y `<footer>`.

Te dejaremos una lectura al final del video, donde podrás conocer mejor las etiquetas semánticas de HTML5. En la lectura podrás saber por qué existen, cuáles son las principales y cómo y cuándo usarlas.

Comencemos analizando sección a sección nuestro HTML. Para ello usaremos el "HTML5 Element Flowchart" de HTML5 Doctor.

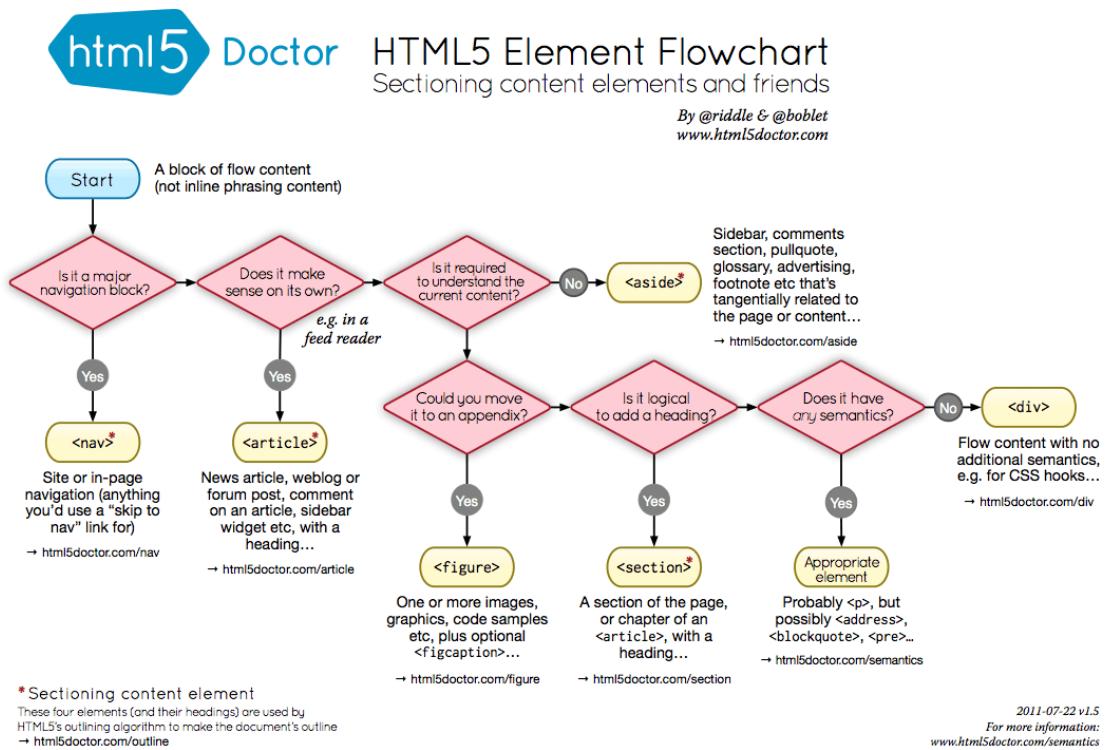


Imagen 9: HTML5

Comencemos por la **primera sección**: Una barra de navegación.

- ¿Es un bloque de navegación?: Sí, por lo tanto la etiqueta que usaremos será `<nav>`.

```
<nav>
  <ul>
    <li><a href="#">Ícono</a></li>
    <li><a href="#">Ubicación</a></li>
    <li><a href="#">Próxima Charla</a></li>
    <li><a href="#">Eventos anteriores</a></li>
    <li><a href="#">Contacto</a></li>
  </ul>
</nav>
```

Sigamos con la **segunda sección**, la sección principal (o Hero section).

- ¿Es un bloque de navegación? No.
- ¿Tiene sentido por sí misma? No, porque es la primera parte de un contenido mayor.
- ¿Se requiere para entender el contenido? Sí.
- ¿Podría moverse a un apéndice (complemento)? No.
- ¿Tiene lógica añadirle un encabezado?: Sí, por lo tanto sería un `<section>`.

Sin embargo, existe una sección más específica que `<section>` para esta división. Si nos fijamos bien, en esta sección está el encabezado principal `<h1>`, además, es lo primero que se ve de la página después del menú de navegación e incluye la información que permite contextualizar respecto al sitio que se está visitando.

En resumidas cuentas, es la introducción a todo el sitio. Por lo tanto, esta sección la podremos etiquetar como `<header>` (**No confundir con la etiqueta `<head>`**).

```
<header>
  
  <h1>Descubre lo último en tecnología bebiendo café</h1>
  <h2>Charlas, eventos y simposios sobre tecnología</h2>
</header>
```

Sigamos con la **tercera sección**, la sección de lugar.

- ¿Es un bloque de navegación? No.
- ¿Tiene sentido por sí misma? No, ya que depende de las otras secciones para su entendimiento.
- ¿Se requiere para entender el contenido? Sí.
- ¿Podría moverse a un apéndice (complemento)? No.
- ¿Tiene lógica añadirle un encabezado?: Sí, por lo tanto sería un `<section>`.

```
<section>
![We work location](assets/img/we-work.jpg)
```

Si nos fijamos bien, las secciones que le siguen también son de tipo `<section>`, ya que en su conjunto se requieren para entender el contenido de la página.

Cuarta sección, sección de próxima charla

```
<section>
![speaker](assets/img/speaker.jpg)
```

Quinta sección, sección de eventos anteriores

```
<section>
<h2>Eventos anteriores</h2>
![scrum-sin-scream](assets/img/scrum-sin-scream.jpg)
```

Ahora nos faltaría la **sección final**, de contacto, a veces llamada **footer**.

Si nos fijamos en la [maqueta final](#) tiene 3 íconos de redes sociales y un texto final que dice "Meet & coffe 2018. Todos los derechos reservados."

Por mientras, no añadiremos esos íconos ya que más adelante aprenderemos cómo. Por lo mismo, sólo pondremos el texto dentro de una etiqueta `<p>` y esa etiqueta `<p>` la pondremos dentro de la etiqueta `<footer>`.

La etiqueta `<footer>` representa un pie de página para el elemento principal, en este caso la página y generalmente contiene información sobre su sección, como quién lo escribió, enlaces a documentos relacionados, datos de derechos de autor y similares.

```
<footer>
  <p>Meet & coffe 2018. Todos los derechos reservados.</p>
</footer>
```

Ahora si guardamos y recargamos el navegador no vemos que haya cambiado nada, pero si lo inspeccionamos, podemos ver que cada contenido está agrupado según su información.

Vinculando el menú con cada sección

Anteriormente conocimos los **id**. Ahora, nos ayudarán a vincular cada ítem del menú de navegación con cada sección de la página.

A cada sección: el `<header>`, las tres `<section>` y el `<footer>` le pondremos un id único y representativo:

```
<header id="inicio">  
  
<section id="ubicacion">  
  
<section id="proxima-charla">  
  
<section id="eventos">  
  
<footer id="contacto">
```

Recuerda no poner mayúsculas, ni tildes, ni espacios en el valor de los `ids` (menos en el valor de las clases, etiquetas, archivos y otros).

Ahora en cada ítem del menú, dentro de la etiqueta `<a>`, en su atributo `href`, pondremos como valor el valor de cada id antepuesto por un `#`.

```
<nav>  
  <ul>  
    <li><a href="#inicio">Ícono</a></li>  
    <li><a href="#ubicacion">Ubicación</a></li>  
    <li><a href="#proxima-charla">Próxima Charla</a></li>  
    <li><a href="#eventos">Eventos anteriores</a></li>  
    <li><a href="#contacto">Contacto</a></li>  
  </ul>  
</nav>
```

Vamos a guardar y recargamos.

Ahora, cada vez que hacemos clic en un ítem del menú nos vincula con cada sección de la página.
¡Intentémoslo!

Resultado

```
<!DOCTYPE html>
<html>
<head>
    <title>Meet & Coffee</title>
    <meta charset="utf-8">
    <link rel="shortcut icon" type="image/png" href="favicon.png">
    <meta name="author" content="Francisca Medina Concha">
    <meta name="description" content="Comparte tus conocimientos tomando café">
    <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
    <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>
    <nav>
        <ul>
            <li><a href="#inicio">Ícono</a></li>
            <li><a href="#ubicacion">Ubicación</a></li>
            <li><a href="#proxima-charla">Próxima Charla</a></li>
            <li><a href="#eventos">Eventos anteriores</a></li>
            <li><a href="#contacto">Contacto</a></li>
        </ul>
    </nav>

    <header id="inicio">
        
        <h1>Descubre lo último en tecnología bebiendo café</h1>
        <h2>Charlas, eventos y simposios sobre tecnología</h2>
    </header>

    <section id="ubicacion">
        
        <h2>¿Dónde nos juntamos?</h2>
        <p>Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.</p>
    </section>

    <section id="proxima-charla">
        
        <h2>Próxima charla</h2>
        <h3>Big Data para Noobs</h3>
        <h4>Rafaela Valdés</h4>
        <h5>Data scientist, PhD Theoretical Physics, Carnegie Mellon University</h5>
        <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.</p>
    </section>

    <section id="eventos">
        <h2>Eventos anteriores</h2>
        
        <h3>Simposio Vegan DB</h3>
        
        <h3>Machine learning 101</h3>
        
        <h3>Scrum sin scream</h3>
    </section>

```

```
<footer id="contacto">
  <p>Meet & coffe 2018. Todos los derechos reservados.</p>
</footer>
</body>
</html>
```

Alineación de texto, color de letra y background

Ahora comenzaremos a añadirle el estilo a nuestra página para que quede igual a la [maqueta final](#).

Alineación de texto y color de letra general

Comencemos por algo básico y conocido. La Alineación de texto y el color de la letra.

Es buena práctica ir de lo más general a lo más específico. Si miramos la [maqueta final](#) y la [guía de estilo](#) podremos darnos cuenta que la mayoría de los textos están centrados y que el color de letra que predomina en la mayoría de los textos es el Rich Black.

- Color Rich Black: #000f08

Por lo tanto, vamos a asignarle a todos los textos ese alineamiento y color de momento.

Para no ir de una etiqueta a otra dándole la propiedad, nos simplificamos la vida indicando la propiedad al selector de body.

```
body {  
    text-align: center;  
    color: #000f08;  
}
```

Como todo el contenido está dentro de la etiqueta `<body>` se aplicará para todos los textos.

Veamos cómo cambia.

Tipos de código de color

Al valor de la propiedad `color` le asignamos un tipo de código de color llamado **hexadecimal**. El código de color hexadecimal está compuesto por un `#` más 6 números y/o letras.

Se especifica con: `#RRGGBB`, donde los enteros hexadecimales RR (rojo), GG (verde) y BB (azul).

En cada uno de sus 6 caracteres se permiten valores que van desde el 0 hasta la f (a partir del número 10 se empiezan a mostrar como letras, desde la `a` hasta la `f`): `0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e` y `f`.

Eso especifica los componentes del color (0 = menor valor | f = mayor valor).

Por ejemplo:

- Blanco: `#ffffff`
- Negro: `#000000`
- Rojo: `#ff0000`
- Verde: `#00ff00`
- Azul: `#0000ff`

Gracias a esto, se pueden obtener hasta 16.777.216 combinaciones diferentes

Lo que habíamos estado utilizando anteriormente por nombre (como `red`, `green`, `blue`) eran los colores predefinidos, que son los colores soportados por todos los navegadores. En total son 140 y puedes conocerlos en el siguiente enlace: [CSS Colors](#).

Alineación de texto, color de letra y color de fondo por sección

Si visualizamos la [maqueta final](#) podremos darnos cuenta que las secciones que tienen colores de fondo (diferente al blanco, que es el que viene por defecto) son el `nav`, la sección de `próxima charla` y el `footer`.

Entonces, le entregaremos a cada sección su color de fondo correspondiente. Según la [guía de estilo](#) podemos identificar los siguientes colores:

- Color Rich Black: `#0000f08`
- Color Sweet Brown: `#a4452c`
- Color Blanco: `#ffffff`

Tanto el `<nav>` como el `<footer>` comparten la característica de tener fondo de color Rich Black y la letra de color blanco, por lo tanto le podremos dar una misma clase para añadirle tales propiedades sin tener que repetirnos: `<nav class="oscuro">` y `<footer id="contacto" class="oscuro">`

```
.oscuro {  
background: #0000f08;  
color: #ffffff;  
}
```



Imagen 10: nav



Imagen 11: Footer

Los links (etiqueta `<a>`) se siguen viendo azules y con el subrayado. Eso lo podremos quitar añadiendo:

```
.oscuro a{  
color: #ffffff;  
text-decoration: none;  
}
```

Esto lo logramos usando el selector descendente. Si en CSS escribimos un `elemento + espacio + otro elemento`, por ejemplo, `.oscuro a`: selecciona todas las etiquetas `<a>` que estén dentro de las etiquetas con clase `oscuro`.



Imagen 12: Selector descendente

Ahora, editaremos el fondo y el color de la sección de próxima charla. Si nos fijamos es la única sección con texto alineado a la izquierda. Entonces, le pondremos una clase para diferenciarla:

```
<section id="proxima-charla" class="sweet-brown">
```

Ahora en CSS añadiremos:

```
.sweet-brown {  
    text-align: left;  
    background: #a4452c;  
    color: #ffffff;  
}
```



Imagen 13: Nueva clase

Imagen de fondo

Si nos fijamos la sección con etiqueta `<header>` tiene el texto alineado a la derecha, la letra de color blanco con una sombra de color Nickel (#707070) y, además, tiene una imagen de fondo (que hasta ahora la hemos puesto como etiqueta ``). Vamos a borrarle esa foto, que llamaremos a través del background y a añadirle una clase a ese header:

```
<header id="inicio" class="hero-section">
  <h1>Descubre lo último en tecnología bebiendo café</h1>
  <h2>Charlas, eventos y simposios sobre tecnología</h2>
</header>
```

```
.hero-section {
  text-align: right;
  background-image: url('../img/bg-hero.png');
  background-repeat: no-repeat;
  background-size: cover;
  color: #ffffff;
  text-shadow: 2px 3px 2px #707070;
}
```

- `text-align: right;` Especificamos que el texto estará alineado a la derecha.
- `background-image: url('../img/bg-hero.png');` Llamamos a la imagen que está dentro de la carpeta **img** que está dentro de la carpeta **assets**. Como nuestro archivo CSS está dentro de la carpeta **css** que está dentro de la carpeta **assets**, es necesario salir de la carpeta **css** para llegar a **assets** y buscar la carpeta **img**. Esto se hace anteponiendo los `../`.
- `background-repeat: no-repeat;` Especifica que la imagen de fondo no se repita, sino quedaría como un mosaico.
- `background-size: cover;` Escala la imagen para que cubra el elemento que la contiene.
- `color: #ffffff;` Hace que el color de letra sea blanco

Y la propiedad `text-shadow: 2px 3px 2px #707070;` Hace que se genere la sombra. Podemos identificar 4 valores:

- El primero es el `h-shadow` que es la posición horizontal de la sombra, o sea, 2px hacia el lado derecho (los valores negativos están aceptados llevando la sombra hacia la izquierda).
- El segundo es el `v-shadow` que es la posición vertical de la sombra, o sea, 2px hacia abajo (los valores negativos están aceptados llevando la sombra hacia arriba).
- El tercero es el `blur-radius` que especifica el desenfoque de la sombra. Mientras más grande será más desenfocado.
- El cuarto es el color, en este caso el color Nickel (`#707070`) especificado por la guía.

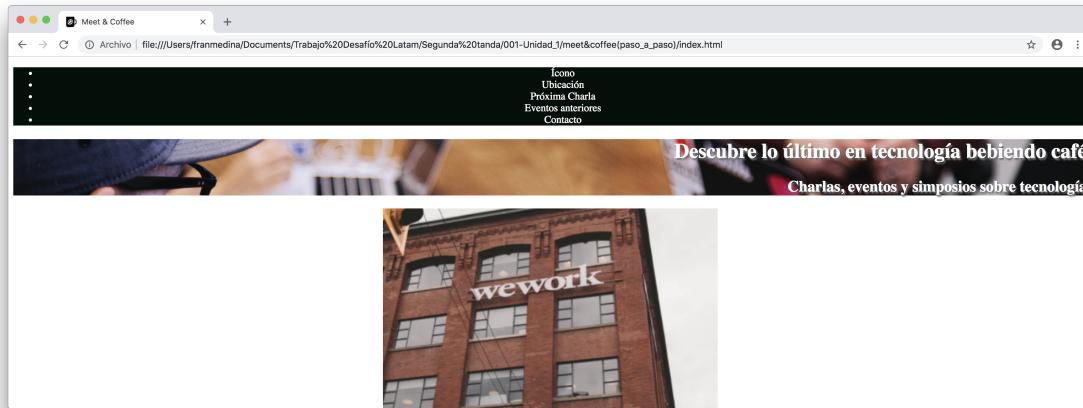


Imagen 14: Sombras

Modelo de cajas

El modelo de cajas es uno los conceptos más importantes detrás de CSS, y responde a la pregunta de cuánto mide realmente cada elemento.



Imagen 15: Modelo de cajas

Propiedades del modelo de cajas

Primero, vamos a analizar el **área de contenido**. El área de contenido es lo que está comprendido dentro del elemento, como textos e imágenes, por ejemplo.

Para editarla, podemos alterar el contenido (ya sea agregándole, editando o quitándole contenido en HTML) o asignarle un **ancho (width)** y/o un **alto (height)** como propiedades en CSS.

Además del área de contenido existe el **padding**, que se utilizan para generar espacio alrededor del contenido, dentro del borde del elemento. El padding es siempre transparente.

El **borde (border)** es lo que contiene y está alrededor del contenido + el padding. Las propiedades de borde de CSS permiten especificar el estilo, el ancho y el color del borde de cada elemento. Aunque, por defecto, en la mayoría de las etiquetas el valor de la propiedad de borde es `none`,

```
border: none;
```

Por su lado, el **margen (margin)** es espacio alrededor de los elementos, fuera de los bordes definidos. El margen es siempre transparente.

Nosotros podemos cambiar los valores de todas estas propiedades utilizando CSS, pero antes de cambiarlas con código directamente, utilizaremos el inspector de elementos para familiarizarnos con estas propiedades y los valores que vienen por defecto en cada etiqueta.

Si inspeccionamos primero la etiqueta `<body>` de nuestra página, veremos lo computado por el navegador. En la pestaña de **Style**, del inspector de elementos, bajaremos hasta encontrar una caja que dice `margin`, `border` y `padding`.

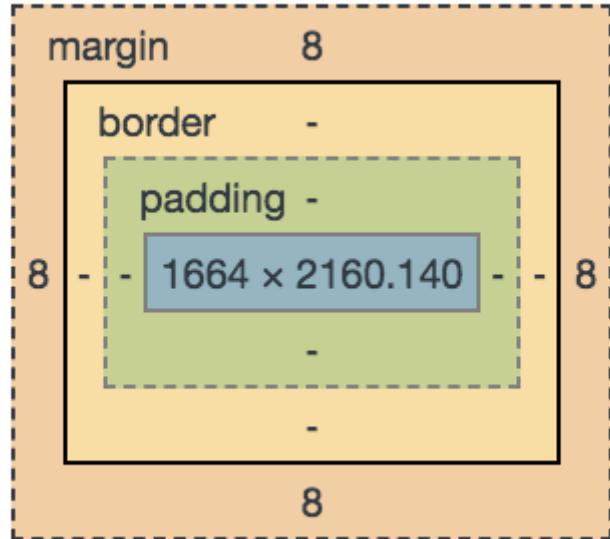


Imagen 16: Propiedades del modelo de cajas - body

Ahí podremos ver cuánto mide el área de contenido de nuestro `body` (ancho x alto), el padding (que no tiene en este caso), el border (que tampoco tiene) y un margin que es de 8px en todas las direcciones.

Si inspeccionamos la etiqueta `<nav>` podremos ver que sólo tiene un area de contenido delimitada por un ancho x alto. Luego, nos vamos a la etiqueta `` y vamos encontrando cosas curiosas.

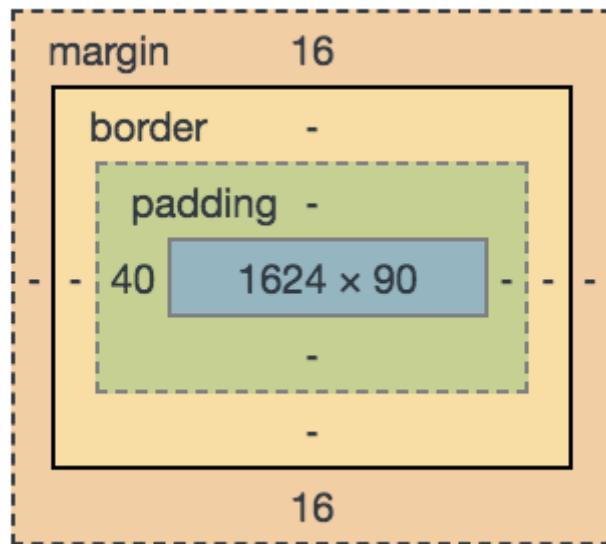


Imagen 17: Propiedades del modelo de cajas - nav

Podemos observar el tamaño de su área de contenido (ancho x alto), un padding sólo a la izquierda de 40px, una carencia de borde y un margin de 16px arriba y abajo.

Luego vamos a inspeccionar la etiqueta `<header>`, que al igual que la etiqueta `<nav>`, sólo tiene un área de contenido delimitada por un ancho x alto. Pero, si nos vamos a la etiqueta `<h1>` y `<h2>`, podremos ver que tienen propiedades de margen arriba y abajo.

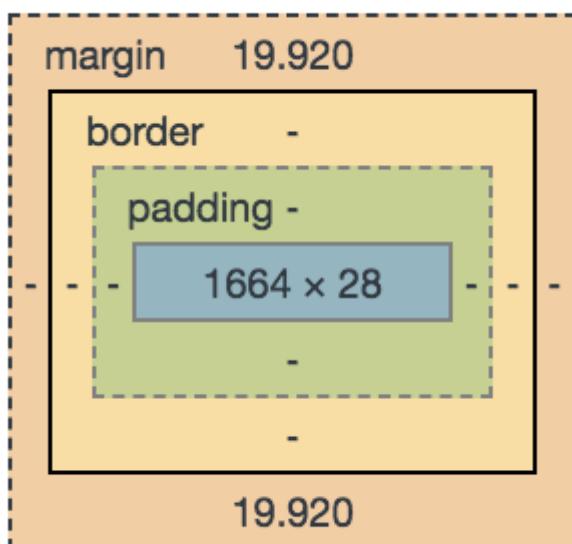
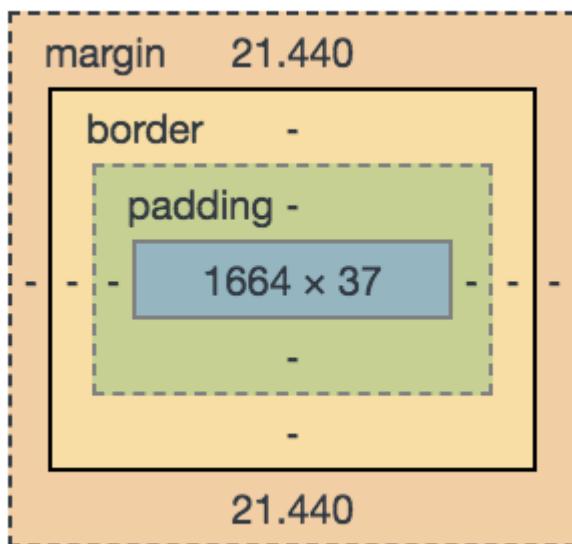


Imagen 18: Propiedades del modelo de cajas - header & nav

Podemos ir inspeccionando una a una las etiquetas, pero lo importante de este ejercicio es darnos cuenta que cada etiqueta tiene propiedades con valores por defecto.

Modificando elementos desde el inspector

Desde el inspector de elementos, podremos modificar las características de cada caja. Si vamos a **padding** o **height**, podremos ir cambiando valores para verlos reflejados de inmediato en la pantalla. Es importante destacar que podemos alterar cualquier propiedad desde el inspector de elementos, pero estos cambios **no quedan guardados en el código**. Incluso, puedes intentarlo con cualquier otro sitio que no te pertenezca, de manera online.

Seleccionemos una sección de nuestra página desde el inspector de elementos y agreguémossle margen.

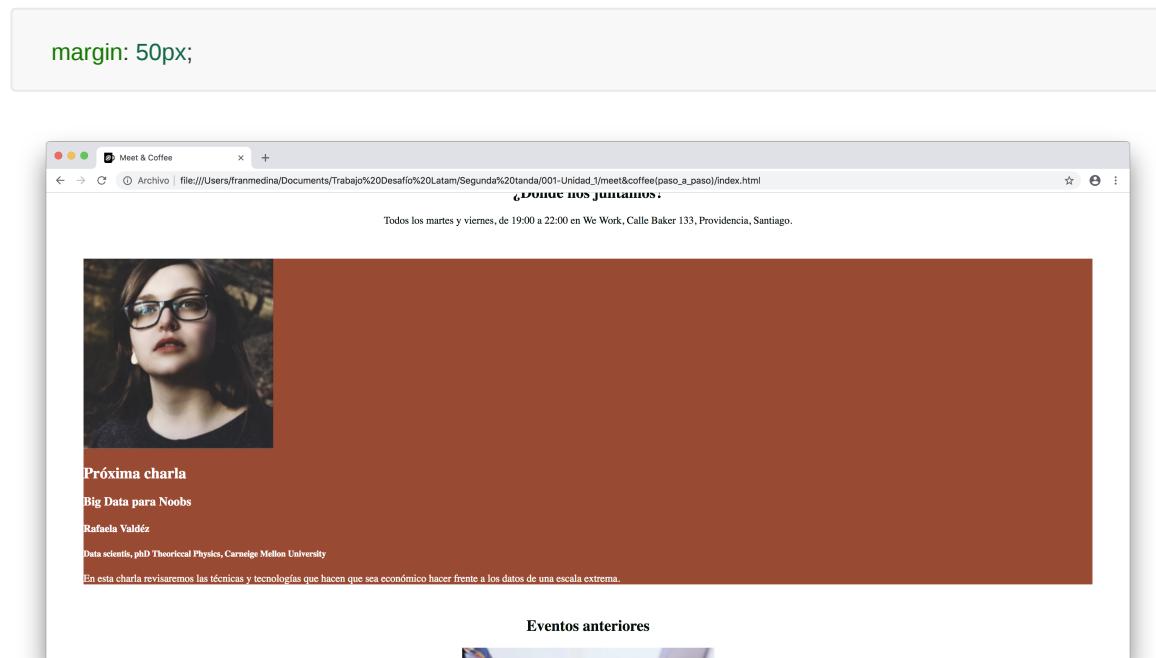


Imagen 19: Agregar margen

Observa lo que ocurre con el elemento.

Ahora cambiemos los bordes. Vamos a utilizar las siguientes propiedades:

```
border-width: 5px;  
border-style: dashed;  
border-color: blue;
```

También podemos especificar las 3 propiedades en una sola línea.

Ejemplo:

```
border: 5px dashed blue;
```

Las dos formas son correctas, pero probablemente te será más cómodo escribirlas en una sola línea.

Observemos los cambios:



Imagen 20: Especificar 3 propiedades en una línea de código

Si quieres saber más sobre la propiedad de borde mira este [link de W3Schools](#)

También podemos modificar el **padding**, recuerda que este es el espacio que existe entre el elemento y el borde.

```
padding: 20px;
```



Imagen 21: Modificar el padding

Tanto en margin como padding se pueden dar valores para `top`, `right`, `bottom` y `left`. Cuando se da un sólo valor por convención se asume que aplica a todos los lados. Cuando tiene dos valores el primero indica `top` y `bottom` y el segundo indica `right` y `left`. Cuando tiene 4 valores aplica la ley del reloj: primero `top`, segundo `right`, tercero `bottom` y cuarto `left`.

También se puede definir como `margin-top`, `margin-right`, `margin-bottom`, `margin-left`, `padding-top`, `padding-right`, `padding-bottom` y `padding-left`.

Hagamos algunas pruebas con el padding:

```
padding: 20px 50px;
```

Bueno, estos ejercicios nos han ayudado a conocer mejor las diferencias entre el margin, border y padding.

Vamos a actualizar la página y ninguno de los cambios que hicimos desde el inspector de elementos se ha guardado.



Imagen 22: Cambios desde el inspector de elementos

Reseteando las propiedades

Observando la [maqueta final](#) podremos notar que el body (contenedor de todo el contenido), no tiene margen. Por lo tanto, le daremos a todo el body un `margin: 0;` (añadiéndoselo al selector que ya habíamos escrito).

```
body {
    text-align: center;
    color: #0000f08;
    margin: 0; /* esta es la propiedad que añadimos */
}
```

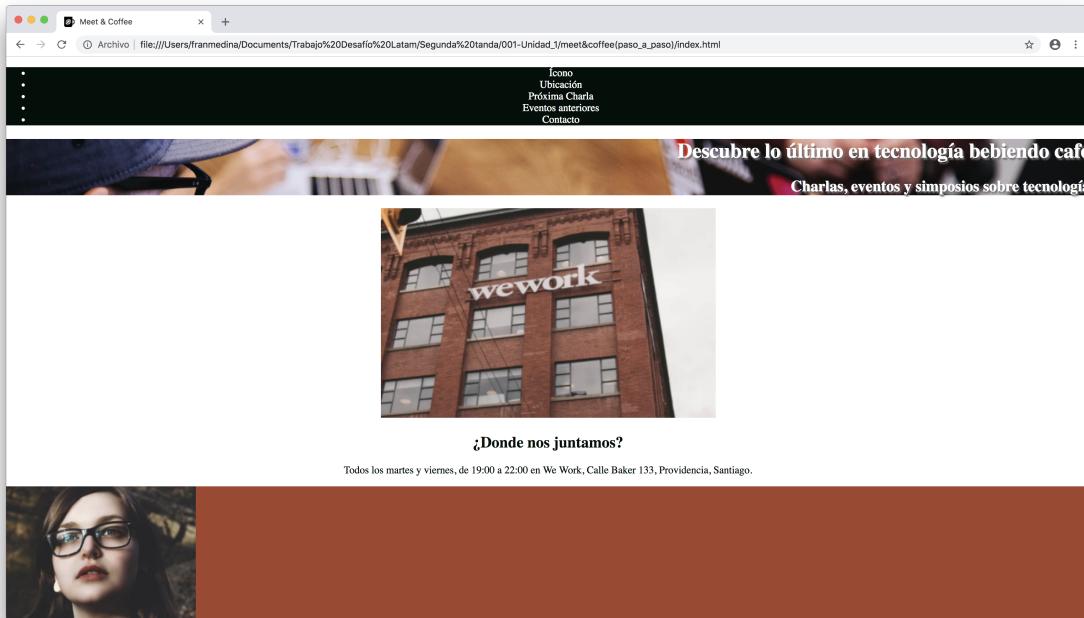


Imagen 23: Resetear las propiedades

Ya podemos notar los cambios.

Ahora resetearemos todas las etiquetas dentro del `<body>` dándole las propiedades de margin, border y padding con el valor `0`. Esto es para que vayamos asignándole a cada sección encabezado y texto sus valores correspondientes para acercarnos más a la maqueta.

Bajo el bloque de declaración del selector de `body` pondremos un nuevo selector para todas las etiquetas. Pero, en vez de estar una a una mencionándolas y separándolas por coma usaremos un selector de tipo descendente.

```
body * {  
    /* Aquí le valor a las daremos propiedades */  
}
```

El `*` como selector en CSS significa todo (es el selector universal que aplica a todas las etiquetas), y que esté seguido de `body` separados por un espacio significa que a todos las etiquetas que están dentro del body le daremos propiedades en el bloque de declaración.

```
body * {  
    margin: 0;  
    border: 0;  
    padding: 0;  
}
```

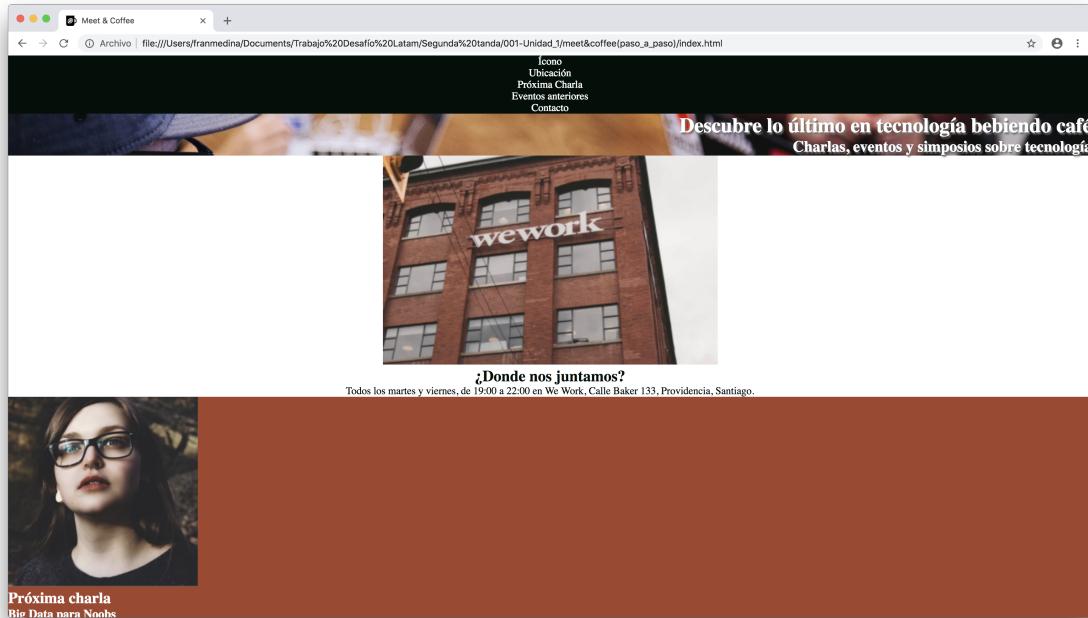


Imagen 24: Etiquetas dentro del body utilizando `*`

Trabajando en el proyecto

Hasta el momento las unidades de medida las hemos trabajado en píxeles, pero existen otras unidades que veremos después. De momento los píxeles nos ayudarán a definir los márgenes entre las secciones y los paddings de cada sección.

Comencemos por el `<header>` hero section. Si nos fijamos en la [maqueta final](#) podemos ver que tiene un padding similar arriba y al lado derecho, además de un gran padding hacia abajo. Por lo tanto añadiremos lo siguiente: `padding: 50px 50px 250px 50px;` al selector de la clase `hero-section`.

```
.hero-section {  
    text-align: right;  
    background-image: url('../img/bg-hero.png');  
    background-repeat: no-repeat;  
    background-size: cover;  
    color: #ffffff;  
    text-shadow: 2px 3px 2px #707070;  
    padding: 50px 50px 250px 50px; /* esta es la propiedad que añadimos */  
}
```

Lo que hicimos fue añadirle 50px arriba al lado derecho y al lado izquierdo y 250px hacia abajo.

Por otro lado, podemos ver que existe una separación entre el `<header>`, `<hero section>` y la sección `<ubicación>`. Pero, relativamente todas las secciones están separadas por un margen similar, por lo tanto las seleccionaremos como grupo para darles a todas la propiedad de `margin-top`.

```
section, footer {  
    margin-top: 75px;  
}
```

El haber usado el selector de tipo elemento le da las propiedades a todas las secciones de la página y al footer.

Ahora, nos damos cuenta que la sección de próxima charla y el footer también tiene padding. A la sección de próxima charla se lo añadiremos usando el selector de clase que ya habíamos usado para esa sección, `sweet-brown`. Y al footer se lo podremos dar creando un selector de etiqueta.

```
.sweet-brown {  
    text-align: left;  
    background: #a4452c;  
    color: #ffffff;  
    padding: 50px; /* esta es la propiedad que añadimos */  
}
```

```
footer {  
    padding: 50px;  
}
```

Vamos a guardar nuestros cambios en el CSS y vamos a recargar el navegador.

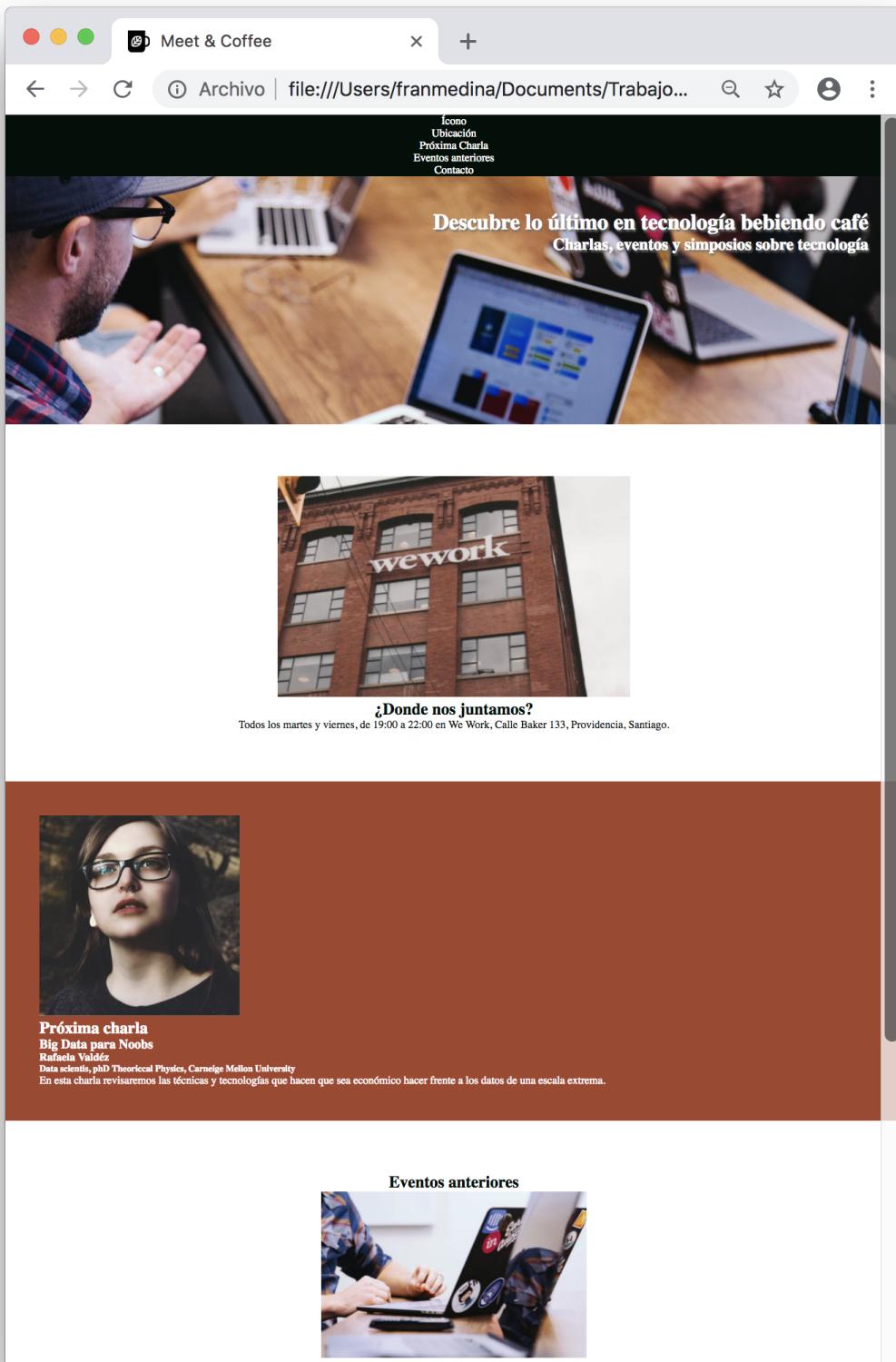


Imagen 25: Añadir selector de clase

Resultado

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Meet & Coffee</title>
  <meta charset="utf-8">
  <link rel="shortcut icon" type="image/png" href="favicon.png">
  <meta name="author" content="Francisca Medina Concha">
  <meta name="description" content="Comparte tus conocimientos tomando café">
  <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
  <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>
  <nav class="oscuro">
    <ul>
      <li><a href="#inicio">Ícono</a></li>
      <li><a href="#ubicacion">Ubicación</a></li>
      <li><a href="#proxima-charla">Próxima Charla</a></li>
      <li><a href="#eventos">Eventos anteriores</a></li>
      <li><a href="#contacto">Contacto</a></li>
    </ul>
  </nav>

  <header id="inicio" class="hero-section">
    <h1>Descubre lo último en tecnología bebiendo café</h1>
    <h2>Charlas, eventos y simposios sobre tecnología</h2>
  </header>

  <section id="ubicacion">
    
    <h2>¿Dónde nos juntamos?</h2>
    <p>Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.</p>
  </section>

  <section id="proxima-charla" class="sweet-brown">
    
    <h2>Próxima charla</h2>
    <h3>Big Data para Noobs</h3>
    <h4>Rafaela Valdés</h4>
    <h5>Data scientist, PhD Theoretical Physics, Carnegie Mellon University</h5>
    <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.</p>
  </section>

  <section id="eventos">
    <h2>Eventos anteriores</h2>
    
    <h3>Simposio Vegan DB</h3>
    
    <h3>Machine learning 101</h3>
    
    <h3>Scrum sin scream</h3>
  </section>
```

```
<footer id="contacto" class="oscuro">
  <p>Meet & coffe 2018. Todos los derechos reservados.</p>
</footer>
</body>
</html>
```

CSS

```
body {  
    text-align: center;  
    color: #000f08;  
    margin: 0;  
}  
  
body * {  
    margin: 0;  
    border: 0;  
    padding: 0;  
}  
  
.oscuro {  
    background: #000f08;  
    color: #ffffff;  
}  
  
.oscuro a{  
    color: #ffffff;  
    text-decoration: none;  
}  
  
.sweet-brown {  
    text-align: left;  
    background: #a4452c;  
    color: #ffffff;  
    padding: 50px; /* esta es la propiedad que añadimos */  
}  
  
.hero-section {  
    text-align: right;  
    background-image: url('../img/bg-hero.png');  
    background-repeat: no-repeat;  
    background-size: cover;  
    color: #ffffff;  
    text-shadow: 2px 3px 2px #707070;  
    padding: 50px 50px 250px 50px; /* esta es la propiedad que añadimos */  
}  
  
section, footer {  
    margin-top: 75px;  
}  
  
footer {  
    padding: 50px;  
}
```

Propiedad Display (inline, block, inline-block)

Antes que sigamos formateando nuestro contenido es necesario que conozcamos la propiedad **display**. La propiedad display es la propiedad de CSS más importante para controlar la disposición de los elementos.

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 en **block** o **inline**.

Elementos block

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

Algunos ejemplos son:

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

Elementos inline

Por su parte, los elementos inline no comienzan en una nueva línea y su ancho ocupa el mínimo espacio posible (sólo lo necesario según su contenido).

Algunos ejemplos son:

- ``
- `<a>`
- ``
- ``

block vs inline

Vamos a inspeccionar la página para ver las diferencias entre cada uno de estos tipos. Comparemos un link `<a>` con un encabezado `<h2>`.

inline-block

Existe un híbrido de valor display entre el block y el inline. El **inline-block**.

La diferencia principal entre el inline-block y el block es que no agrega un salto de línea después del elemento, por lo que el elemento puede ubicarse junto a otros elementos.

La diferencia principal entre el inline-block y el inline es que permite establecer un ancho (width) y una altura (height) al elemento. Además respeta los márgenes y paddings.

Un ejemplo claro son las imágenes, ya que se despliegan unas al lado de las otras pero podemos asociarle un ancho (width) y un alto (height).

Trabajando en el proyecto

En nuestra página podemos encontrar más de algún inline-block. Por ejemplo en el menú, en la sección de próxima charla y en la sección de eventos anteriores.

El menú lo dejaremos para más adelante así que trabajaremos en la sección de próxima charla y en la sección de eventos anteriores.

En la sección de próxima charla podemos identificar la imagen y al lado de ésta el texto. También podemos darnos cuenta que, si bien el texto está alineado a la izquierda, el contenido está centrado.

Lo que haremos será agrupar tanto la imagen como los textos en un `<div>` y luego las etiquetas del texto (`<h2>`, `<h3>`, `<h4>`, `<h5>` y `<p>`) en otro `<div>` más específico.

Y, tanto a ese div específico contenedor de las etiquetas de texto como a la imagen le daremos la propiedad `display: inline-block;`, a través de una clase para cada una.

Escribamos...

HTML

```
<section id="proxima-charla" class="sweet-brown">
  <div class="centrado">
    

    <div class="charla-texto">
      <h2>Próxima charla</h2>
      <h3>Big Data para Noobs</h3>
      <h4>Rafaela Valdés</h4>
      <h5>Data scientis, phD Theoretical Physics, Carnegie Mellon University</h5>
      <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.</p>
    </div>
  </div>
</section>
```

CSS

```
.charla-speaker, .charla-texto {
  display: inline-block;
}
```

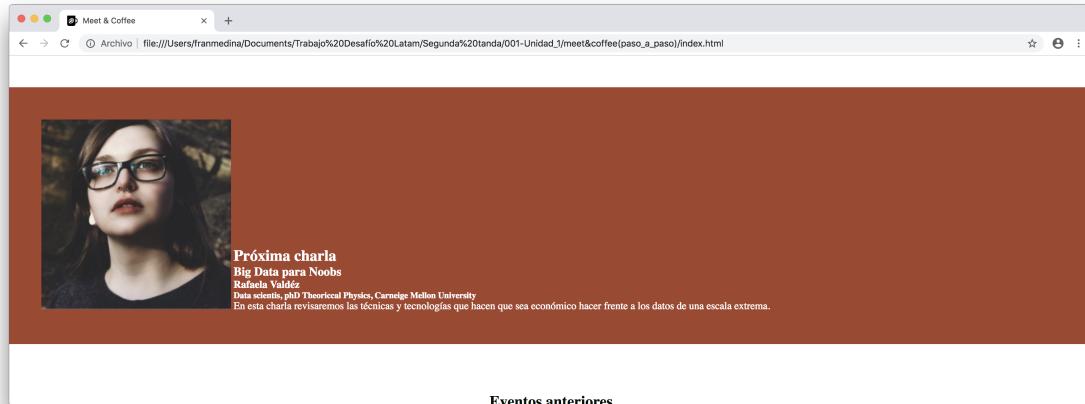


Imagen 26: CSS

En un siguiente video ocuparemos el `<div>` con clase centrado para centrar el contenido.

Bueno, en la sección de eventos anteriores podemos ver los 3 eventos desplegados uno al lado del otro. Por lo tanto, debemos agrupar cada uno de ellos en un divisor propio. Podríamos intuir que cada evento es un artículo por separado ya que tiene sentido por sí mismo, entonces la info de cada evento la englobaremos en un `<article>`. A cada `<article>` le añadiremos una clase para darle la propiedad `display: inline-block`.

HTML

```
<section id="eventos">
  <h2>Eventos anteriores</h2>
  <article class="evento">
    
    <h3>Simposio Vegan DB</h3>
  </article>

  <article class="evento">
    
    <h3>Machine learning 101</h3>
  </article>

  <article class="evento">
    
    <h3>Scrum sin scream</h3>
  </article>
</section>
```

```
.charla-speaker, .charla-texto, .evento {
  display: inline-block;
}
```

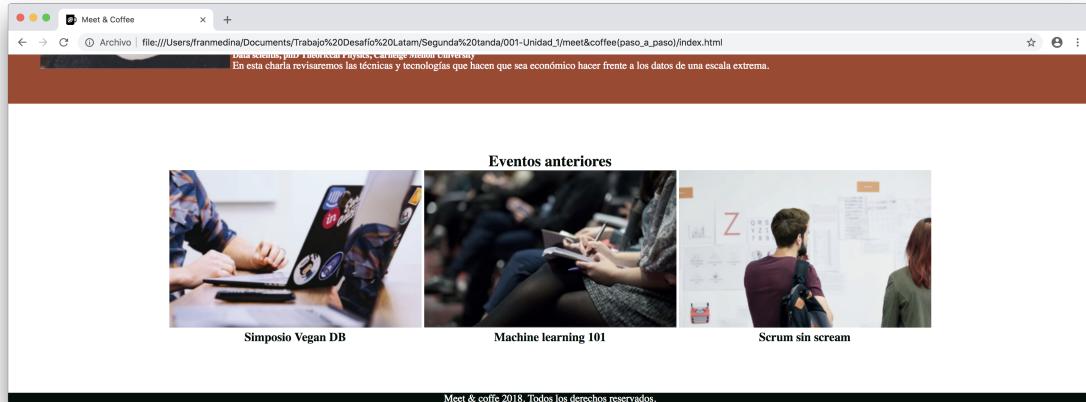


Imagen 27: HTML

Width y Height + Unidades de medida

Width y **height** son las propiedades que se utilizan para establecer el ancho y el alto de un elemento, respectivamente. Las propiedades de anchura y altura no incluyen el padding, bordes o márgenes, sino que afectan directamente al área de contenido del elemento.

Los valores que pueden alcanzar estas propiedades son valores de longitud basadas en las unidades de medida estándar para CSS.

Podemos clasificar las unidades de medida para CSS en dos, **unidades absolutas** y **unidades relativas**.

Unidades absolutas

Las unidades de medida absolutas son unidades que están completamente definidas, o sea, se encuentran definidas en términos concretos y de manera medible. Esto quiere decir que su valor no depende de otro valor de referencia. La que hemos estado utilizando hasta ahora es el pixel, pero también existen los milímetros, centímetros, pulgadas y puntos. Pero, como estamos trabajando con pantallas lo más lógico que trabajemos con los píxeles, que es la unidad mínima de resolución de la pantalla.

Ojo que los píxeles (px) son relativos al dispositivo de visualización, por ende cambian si el dispositivo es de una alta resolución o baja.

Unidades relativas

Las unidades relativas no son valores exactos, sino que se calculan a partir de otro valor de referencia. A pesar de parecer más difíciles de calcular, son las más utilizadas en el diseño de sitios web responsive por su adaptabilidad a los diferentes dispositivos.

- **Porcentaje (%)**: Su valor está calculado en base a su contenedor. Si, por ejemplo, el porcentaje se utiliza para establecer la anchura de un elemento, su referencia es la anchura de su elemento contenedor. Si el elemento no se encuentra dentro de ningún otro elemento, su referencia es la anchura de la página entera.
- **Viewport width (vw) y viewport height (vh)**: Son medidas relativas de acuerdo al viewport o ventana de visualización del navegador.
 - 1vw = 1% del ancho de la ventana de visualización.
 - 100vw = el ancho total de la ventana de visualización.
 - 1vh = 1% de la altura de la ventana de visualización.
 - 100vh = altura de la ventana de visualización.
- **rem**: Las unidades rem dependen directamente del tamaño de fuente del elemento `<html>`. Si el `font-size` del `<html>` es `16px`, 1rem sería igual a 16px en cualquier parte del documento.
- **em**: Las unidades em dependen del `font-size` definido dentro del elemento en el que se encuentra.

Las últimas dos medidas (rem y em) son más utilizadas para los tamaños de fuente, ahora que vamos a trabajar sobre el ancho (width) y el alto (height) nos fijaremos más en los porcentajes (%).

Trabajando en el proyecto

Lo primero que haremos para ejercitarnos es darle un ancho específico al `<div>` contenedor de la próxima charla, ese a que le dimos clase `centrado`.

```
.centrado {  
    width: 75%;  
    margin: 0 auto;  
}
```

Le añadiremos una propiedad `margin: 0 auto` para centrarlo horizontalmente, ya que con eso le damos valor de 0 para arriba y abajo y auto para el lado derecho e izquierdo.

Podemos ver el resultado:

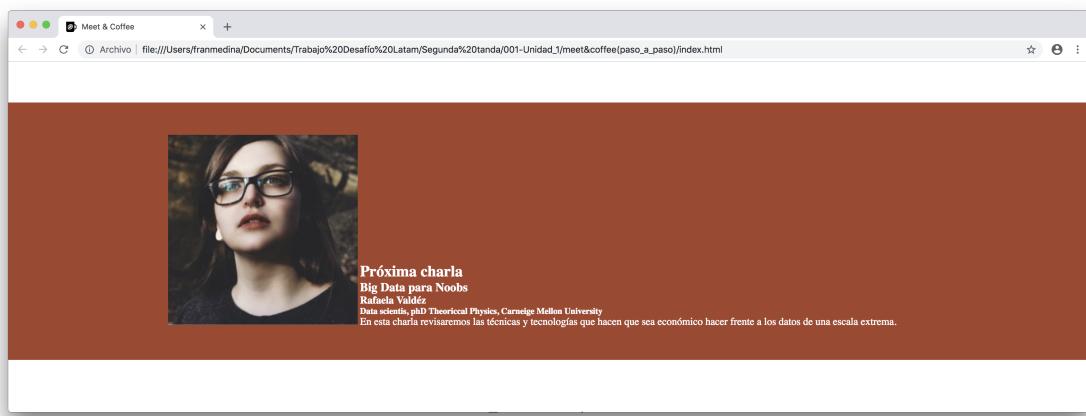


Imagen 28: Añadir propiedad margin

El ancho que le dimos resultó ser el 75% del contenedor `<section id="proxima-charla" class="sweet-brown">` que a su vez tiene el tamaño del 100% de la ventana de visualización del navegador.

Si achicamos la ventana podemos ver cómo el `div` también se achica. Llega un punto en el que los dos elementos en `display: inline-block` no caben uno al lado del otro dentro del `<div class="centrado">`, lo que genera que el segundo debe caer.

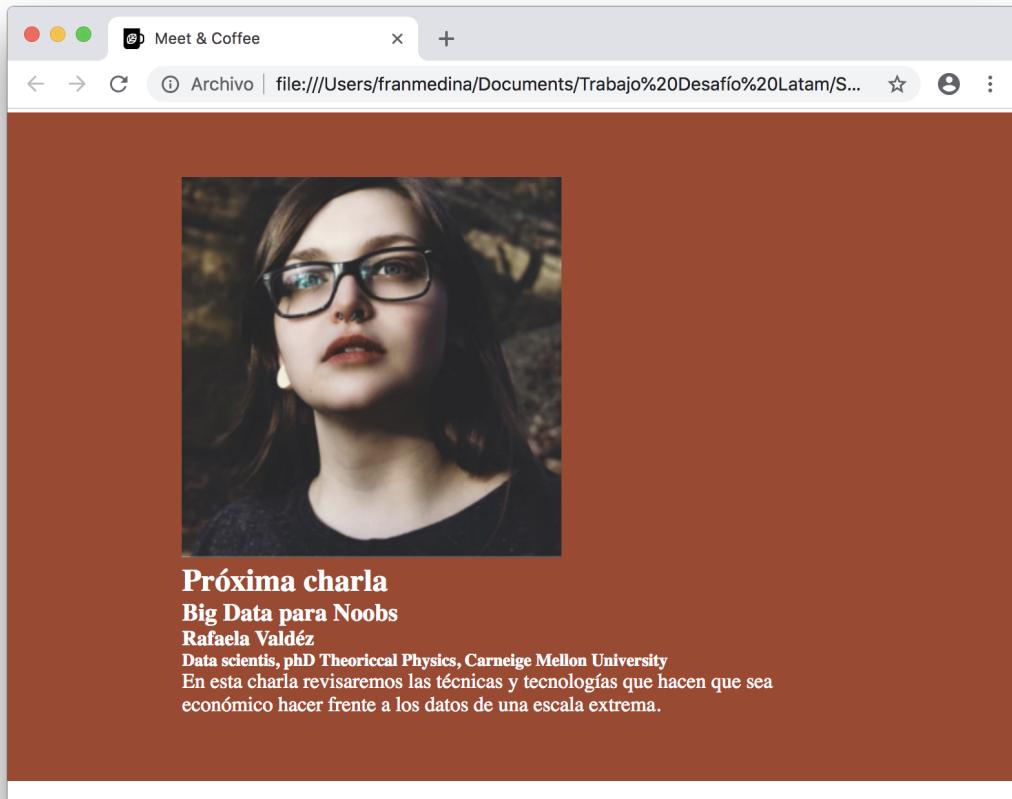


Imagen 29: Cambio de posición de texto

Lo que haremos a continuación será darle una propiedad de ancho a la imagen y al div contenedor del texto, también en porcentajes.

Como a la imagen y al `<div>` ya le habíamos dado la clase `charla-speaker` y `charla-text` respectivamente, crearemos 2 nuevos selectores con esa clase.

```
.charla-speaker{  
    width: 25%;  
}  
  
.charla-texto {  
    width: 72.5%;  
}
```

A la imagen `` le dimos un ancho del 25% del 100% del contenedor, que a su vez es el 75% de su contenedor `<section id="proxima-charla" class="sweet-brown">`. Al div `<div class="charla-texto">` le dimos un ancho del 72.5% del 100% del contenedor, que a su vez es el 75% de su contenedor `<section id="proxima-charla" class="sweet-brown">`.

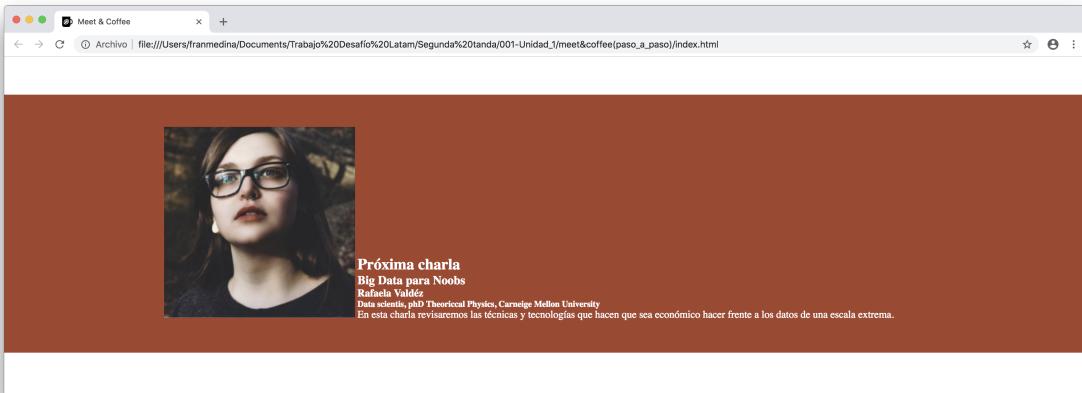


Imagen 30: Cambiar tamaño de imagen

Si agrandamos o achicamos la pantalla se va a seguir viendo en proporción.

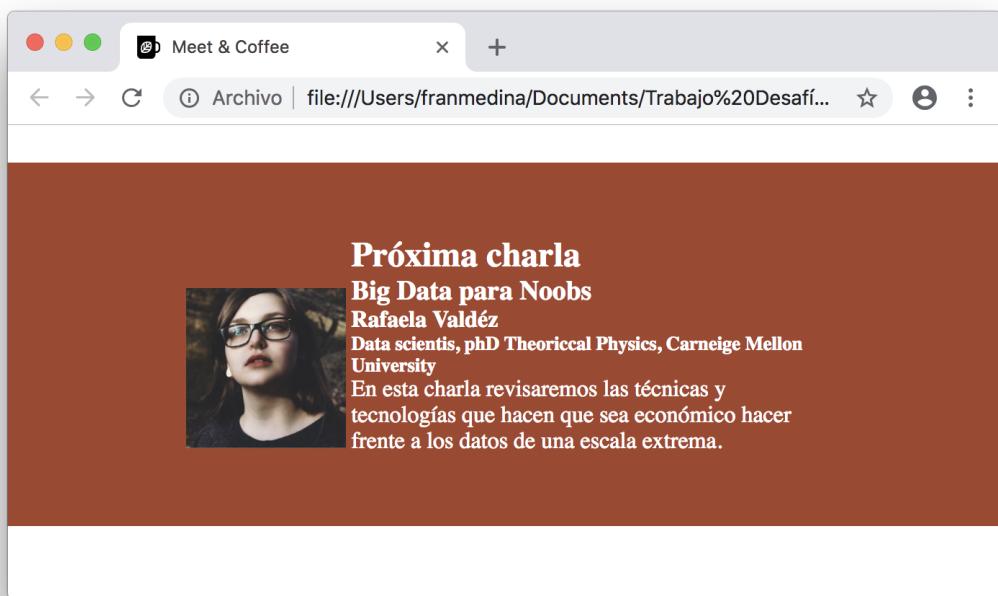


Imagen 31: Agrandar pantalla

Nos faltan dos cosas, la primera es que la imagen en relación al texto esté verticalmente alineada al centro y la segunda es que tenga su margen entre imagen y texto. Para la primera le vamos a añadir la propiedad `vertical-align: middle` a ambas etiquetas.

```
.charla-speaker{  
width: 25%;  
vertical-align: middle; /* esta es la propiedad que añadimos */  
}  
  
.charla-texto {  
width: 72.5%;  
vertical-align: middle; /* esta es la propiedad que añadimos */  
}
```

Y podemos ver el resultado:

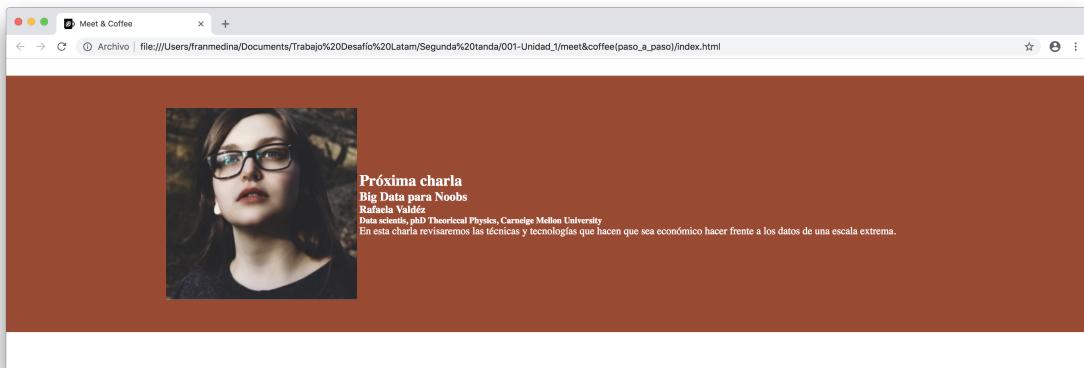


Imagen 32: vertical-align

Para lo segundo vamos a hacer un margen al lado izquierdo del div `<div class="charla-texto">` de 2.5% para que sumados con la `<imagen + margen + div contenedor>` de los textos den 100% ($25 + 2.5 + 72.5 = 100$)

```
.charlaTexto {  
    width: 72.5%;  
    vertical-align: middle;  
    margin-left: 2.5%; /* esta es la propiedad que añadimos */  
}
```

Ahora recargamos y ¿Qué pasó? Cayó el `<div>` de los textos.

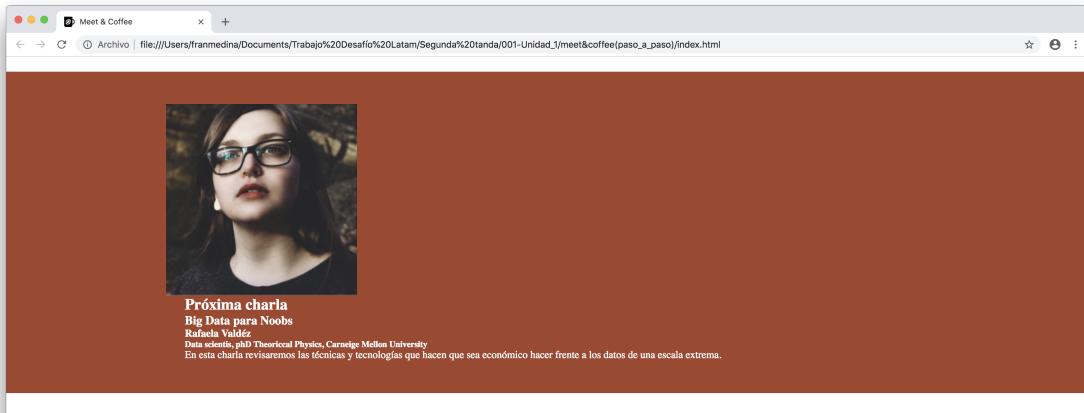


Imagen 33: Marhen izquierdo

¿Por qué? Si la `<imagen + margen + div contenedor>` da exactamente 100%.

Aunque suene obvio, el espacio entre la **imagen** y el **div** en el HTML es el responsable.

Miremos qué sucede si pegamos el **div** a la imagen:

```
<section id="proxima-charla" class="sweet-brown">
  <div class="centrado">
    <div class="charla-
  texto">
    <h2>Próxima charla</h2>
    <h3>Big Data para Noobs</h3>
    <h4>Rafaela Valdés</h4>
    <h5>Data scientis, phD Theorrical Physics, Carneige Mellon University</h5>
    <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer
  frente a los datos de una escala extrema.</p>
  </div>
</div>
</section>
```

Pues no tenemos ese problema...

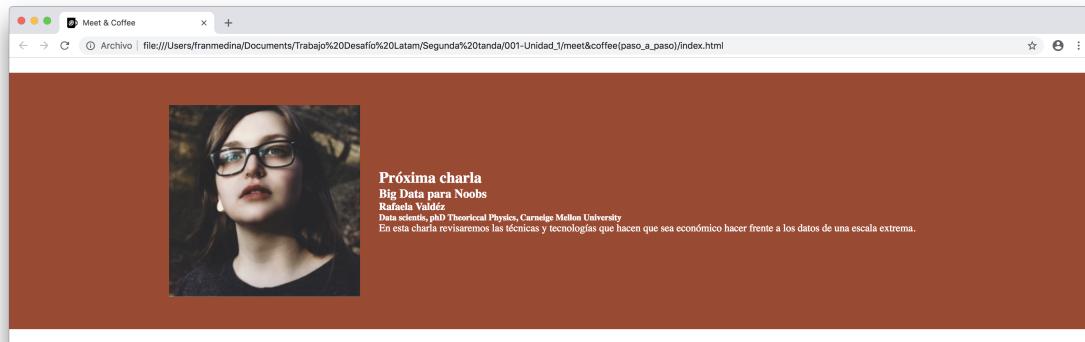


Imagen 34: Espacio entre imagen y div

Pero, no vamos a ir una a una pegando las etiquetas unas con otras para que esto no nos suceda, eso nos haría muy confuso entender nuestro propio código.

Hay diversas formas de solucionar este problema. Vamos a revisar una de ellas, pero puedes encontrar más soluciones en este [link](#).

- Tenemos que añadir la propiedad `white-space: nowrap;` al contenedor de estos dos elementos, o sea al `<section id="proxima-charla" class="sweet-brown">`
- Luego resetear esta misma propiedad en el **div contenedor** de texto para que no le afecte al texto de ésta.

```
.sweet-brown {  
    text-align: left;  
    background: #a4452c;  
    color: #ffffff;  
    padding: 50px;  
    white-space: nowrap; /* esta es la propiedad que añadimos */  
}
```

```
.charla-texto {  
    width: 72.5%;  
    vertical-align: middle;  
    margin-left: 2.5%;  
    white-space: normal; /* esta es la propiedad que añadimos */  
}
```

Y lo tenemos.

Te voy a dejar el desafío que en la sección de eventos anteriores cada `<article class="evento">` tenga un ancho determinado en porcentajes del 30% de su contenedor y que la imagen en su interior sea de 75% de ese 30%.

Tipografías, peso tipográfico y tamaños de fuente

Hasta ahora hemos trabajado con la tipografía y los tamaños por defecto que vienen para los encabezados y los párrafos.

Esta tipografía se puede cambiar.

CSS Web Safe Font

En CSS vienen por defecto las CSS Web Safe Font que son las tipografías que se pueden llamar libremente con CSS sin problemas. En este [link de W3Schools](#) podemos conocer cuáles son. Para llamar a una y aplicarla a nuestra página debemos escribir la propiedad `font-family: "Courier New", Courier, monospace;`

En el ejemplo de W3Schools aparece más de una tipografía en los valores de la propiedad `font-family`. Estos son los fallbacks, o sea el plan b, cuando no funciona la primera tipografía se ocupa la segunda o la tercera opción de la lista.

Al agregar fuentes en tu página web es importante saber que las CSS Web Safe Font te servirán en cualquier navegador, debido a que son estándar y están incluidas en casi todos los sistemas operativos.

Sin embargo, son pocas tipografías y no siempre se adaptarán al diseño que queremos. Según la [guía de estilo](#) la tipografía usada es la 'Open Sans' en sus pesos Extra-Bold, Bold y Regular.

Open Sans Extra-Bold (800)

Open Sans Bold (700)

Open Sans Regular (400)

Imagen 35: Tipografías

Pero, la Open Sans no está dentro de las CSS Web Safe Font.

Google Fonts

Un recurso muy valioso es **Google Fonts**, un repositorio de tipografías proporcionado por Google, donde podrás encontrar muchas tipografías de forma gratuita y libre.

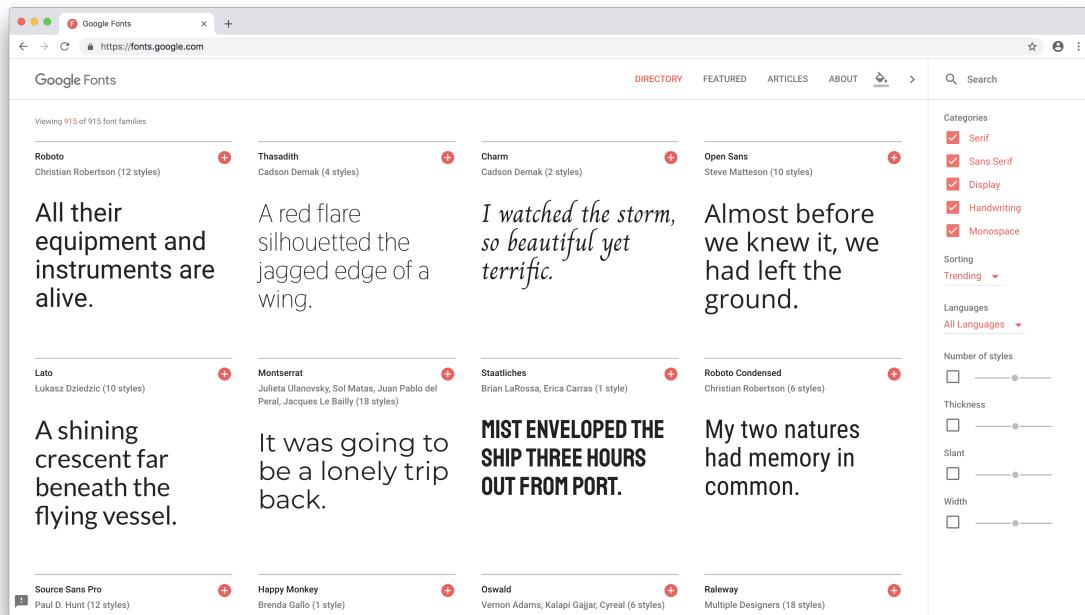


Imagen 36: Google Fonts

Para utilizar Google Fonts debes realizar los siguientes pasos:

- **Paso 1:** Ingresaremos a [Google Fonts](#).
- **Paso 2:** A la derecha de la página existe un buscador, vamos a escribir el nombre de una las tipografías de nuestro sitio.

Comencemos buscando la 'Open Sans'. Una vez encontrada, podemos ver un botón a la derecha que dice **Select this font**, presinémoslo y en la parte inferior derecha aparece un menú el cual nos indica la fuente seleccionada.

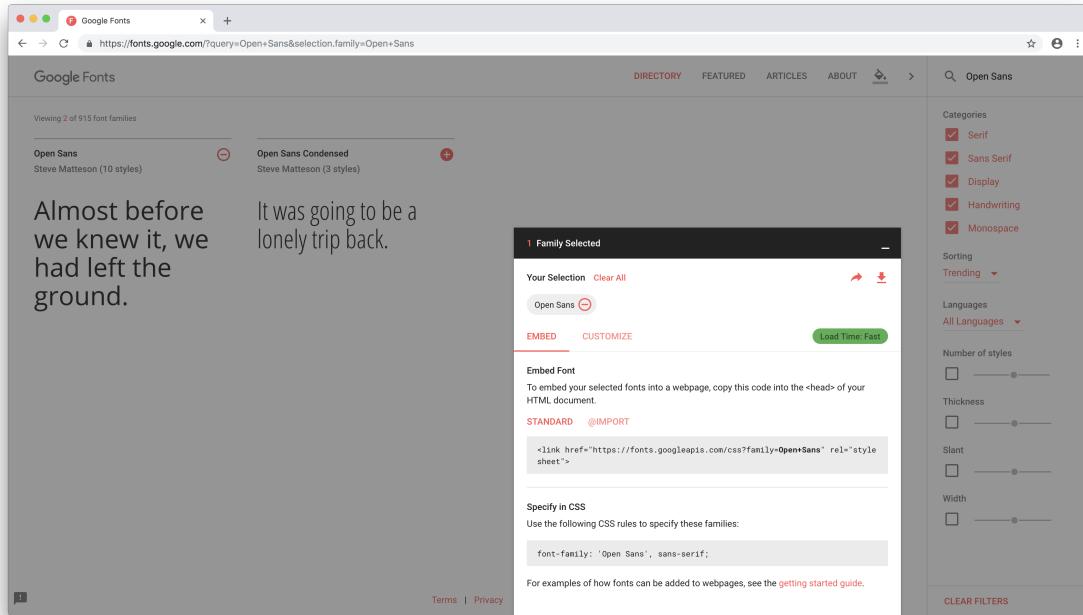


Imagen 37: Utilizar Google Fonts

Con la fuente seleccionada, debemos customizar los estilos que necesitamos descargar. Para ello existe la pestaña llamada **customize** donde marcaremos la opción de los pesos que requerimos: Extra-Bold, Bold y Regular.

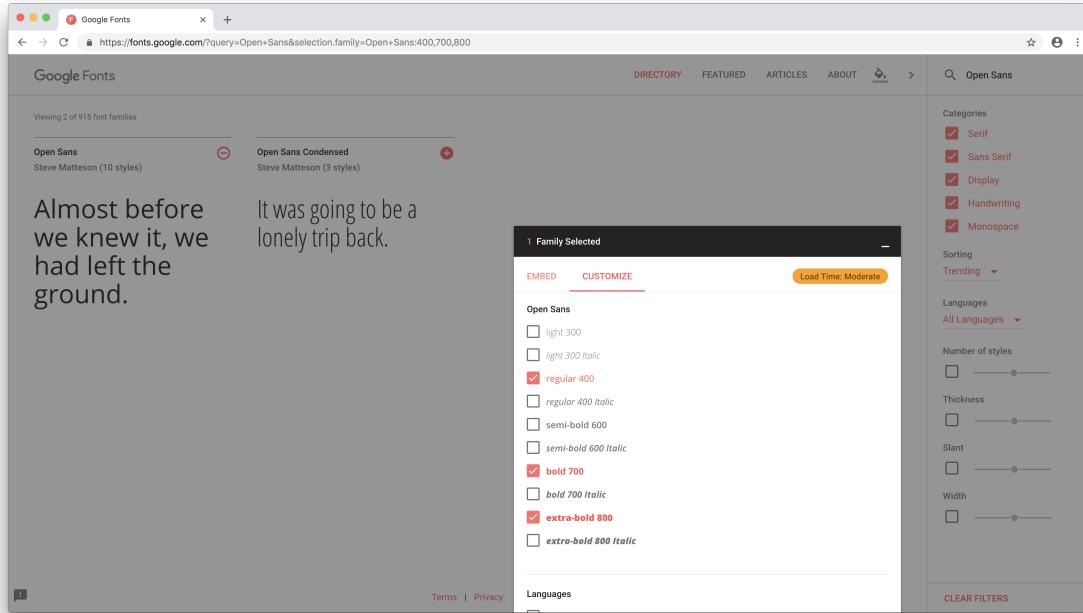


Imagen 38: Customizar estilos

Una vez seleccionado todo lo que necesitamos, presionaremos otra vez la pestaña de **embed** donde aparecerá la URL que necesitamos añadir a nuestro archivo HTML.

- **Paso 3:** Copiaremos el código que entrega Google y lo pegaremos dentro de la etiqueta head de la página, siempre por sobre del llamado a tu CSS propio ¿Por qué? Porque en nuestro CSS vamos a utilizar las tipografías, por lo tanto, las necesitamos llamar antes.

```
<head>
  <title>Meet & Coffee</title>
  <meta charset="utf-8">
  <link rel="shortcut icon" type="image/png" href="favicon.png">
  <meta name="author" content="Francisca Medina Concha">
  <meta name="description" content="Comparte tus conocimientos tomando café">
  <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700,800" rel="stylesheet"> <!--
Esto es lo que hemos añadido -->
  <link rel="stylesheet" href="assets/css/style.css">
</head>
```

- **Paso 4:** Para utilizar la tipografía donde corresponda debemos usar el ejemplo proporcionado por Google Fonts, es decir, usar la propiedad `font-family` con el valor `'Open Sans'`, `sans-serif` en los elementos que corresponda.

Como, en este caso, la tipografía va a ser la misma para todos los textos (párrafos y distintos tipos de encabezados) y sólo varían los valores de los pesos, le vamos a dar la propiedad `font-family: 'Open Sans', sans-serif;` a la etiqueta del `body`.

```
body {
  text-align: center;
  color: #0000f08;
  margin: 0;
  font-family: 'Open Sans', sans-serif; /* esta es la propiedad que añadimos */
}
```

Miremos el resultado:

Meet & Coffee

Archivo | file:///Users/franmedina/Documents/Trabajo%20...

Ícono Ubicación Próxima Charla Eventos anteriores Contacto

Descubre lo último en tecnología bebiendo café
Charlas, eventos y simposios sobre tecnología



¿Donde nos juntamos?

Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.



Próxima charla
Big Data para Noobs
Rafaela Valdés
Data scientist, PhD Theoretical Physics, Carnegie Mellon University
En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala

Imagen 39: Resultado

Muy bien, ha cambiado.

font-weight

Ahora nos queda editar sus pesos.

Podemos identificar en la [maqueta final](#) que la tipografía Open Sans Extra-Bold se aplica sobre el `<h1>`, para ello debemos aplicar la propiedad `font-weight`. En el mismo Google Fonts podemos ver cuál es el peso correspondiente a Extra-Bold, en el caso de esta tipografía es 800.

```
h1 {  
    font-weight: 800;  
}
```

Podemos ver que la mayoría de los `<h2>` tienen peso de Bold, es decir, de `700`, excepto el subtítulo que dice `<h2>Charlas, eventos y simposios sobre tecnología</h2>`. Por lo tanto, lo que haremos será añadirle una clase específica a ese `<h2>`, `<h2 class="subtitulo">Charlas, eventos y simposios sobre tecnología</h2>` y le daremos de `font-weight: 400`. A todos los demás le daremos un `font-weight: 700`.

```
h2 {  
    font-weight: 700;  
}  
  
.subtitulo {  
    font-weight: 400;  
}
```

Además, utilizamos `h4` y `h5` para la sección de próxima charla. Ambos por defecto tienen un `font-weight` de `700` pero en la maqueta son regular. También editaremos eso.

```
.subtitulo, h4, h5 {  
    font-weight: 400;  
}
```

Si jugaste CSS dinner podrás recordar que para darle propiedades a más de un elemento basta con separarlos por coma. Ahí nos estaríamos ahorrando líneas de código.

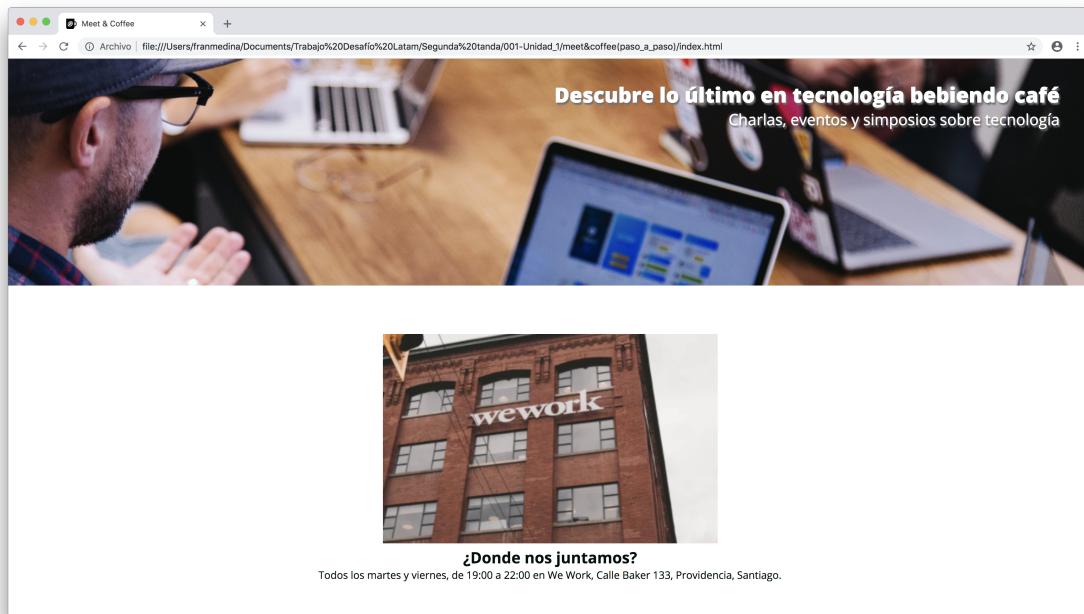


Imagen 40: Ahorrar líneas de código

font-size

Lo que haremos a continuación será añadirle tamaños a las fuentes tipográficas con la propiedad `font-size`. Para ello debemos volver a retomar el tema de las unidades absolutas (pixeles) y unidades relativas (em y rem, en este caso).

Para asignarle tamaños a las fuentes no se recomienda utilizar los pixeles (px). Como estudiamos anteriormente, mencionamos las unidades de medida, los pixeles son de tipo absoluta, por ende rígidas. Si usamos pixeles también estamos ignorando las configuraciones que cada usuario pueda tener en su navegador.

Vamos a comenzar de la base de que los navegadores de manera predeterminada definen un `font-size` de 16px al elemento HTML.

- **em:** La unidad em es escalable y siempre depende de su elemento padre. Por ejemplo, si el elemento body tiene un tamaño de fuente de 16px y un elemento hijo tiene una fuente con tamaño 1.3em, este texto se mostrará de un tamaño un 30% más grande que el del body (20.8px), mientras que si dentro de ese elemento tenemos otro hijo con un `font-size` de 1.3 em, el tamaño de fuente de este objeto sería un 30% más grande que el tamaño de su padre (27.04px). Es recomendable usarla para:
 - Tamaños de fuente
 - Saltos de línea
 - Margen entre párrafos
- **rem:** La unidad rem no es escalable porque no depende del elemento padre, sino del elemento raíz del documento, el elemento HTML. Si el elemento HTML tiene un tamaño de fuente de 16px (como es por defecto), entonces 1rem, sería igual a 16px, y si queremos aplicar un tamaño basado en rem a cualquier elemento de la página, no importará cual sea el tamaño de fuente que tenga asociado ese elemento, ya que 1 rem siempre será igual a 16 pixeles a no ser que se modifique el elemento raíz. Es recomendable usarla para:
 - Elementos del layout que tengan medida fija.

Vamos a ver cuántos em miden los encabezados y los párrafos por defecto. Para ellos vamos a utilizar nuevamente el inspector de elementos.

- h1 (2em)
- h2 (1.5em)
- h3 (1.17em)
- h4 (no dice, se asume que es 1em)
- h5 (0.83em)
- p (no dice, se asume que es 1em)

Entonces, pongámosle las medidas para que queden similar a laqs que están en la [maqueta final](#).

Comencemos por añandiéndole un `font-size` y un `margin-bottom` al `h1`:

```
h1 {  
    font-weight: 800;  
    font-size: 3.75em; /* esta es la propiedad que añadimos */  
    margin-bottom: 0.25em; /* esta es la propiedad que añadimos */  
}
```

Podemos ver que ese `margin-bottom`, pese a tener un "tamaño pequeño" (0.25) igual es relativamente grande. Eso es porque esos 0.25em equivalen al 25% de tamaño del `h1`, que en este caso es el 375% del tamaño del padre (HTML), que por defecto es 16. Por lo tanto el `h1` en pixeles estaría midiendo 60 y su margen inferior, 15.

Sigamos con el `h2`, `h3`, `h4` y `h5`.

```
h2 {  
    font-weight: 700;  
    font-size: 2.5em;  
}  
  
h3 {  
    font-size: 2em;  
}  
  
h4 {  
    font-size: 1.75em;  
}  
  
h5 {  
    font-size: 1.25em;  
}
```

También les asignaremos a los encabezados, dependiendo dónde se encuentren, `margin-top` y `margin-bottom`.

```
#ubicacion h2 {  
    margin: 1em auto;  
}  
  
.sweet-brown h3, .sweet-brown h5 {  
    margin-bottom: 0.5em; /* Aquí se puede ver el ejemplo claro de cómo los em están condicionados por  
el elemento padre */  
}  
  
#eventos h2 {  
    margin-bottom: 1em;  
}  
  
#eventos h3 {  

```

Otra cosa que vamos a pasar a em será el `text-shadow`, como los tamaños de fuente para el `h1` y el `h2` cambiaron se ven muy pequeños, así que escribiremos lo siguiente:

```
.hero-section {  
    text-align: right;  
    background-image: url('../img/bg-hero.png');  
    background-repeat: no-repeat;  
    background-size: cover;  
    color: #ffffff;  
    text-shadow: 0.125em 0.25em 0.125em #707070; /* Esta línea la pasamos a em */  
    padding: 3.125rem 3.125rem 6.25rem 3.125rem;  
}
```

Lo que hicimos fue hacer el tamaño de sombra proporcional al tamaño de letra.

Como aprendimos también sobre los rem, vamos a pasar los márgenes y paddings, que hemos puesto en pixeles en nuestro layout, a rem. Para ello vamos a utilizar esta [calculadora](#).

Resultado

Meet & Coffee

Archivo | file:///Users/frannedina/Documents/Trabajo%20Desafío%20Latam/Segunda%20tanda/001-Unidad_1/meet&coffee(paso_a_paso)/index.html

Icono
Ubicación
Próxima charla
Eventos anteriores
Contacto

Descubre lo último en tecnología bebiendo café

Charlas, eventos y simposios sobre tecnología

¿Donde nos juntamos?

Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.

Próxima charla
Big Data para Noobs

Rafaela Valdés
Data scientist, PhD Theoretical Physics, Carnegie Mellon University

En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.

Eventos anteriores

Simposio Vegan DB

Machine learning 101

Scrum sin scream

Meet & coffee 2018. Todos los derechos reservados.

Imagen 41: Resultado font-size

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Meet & Coffee</title>
  <meta charset="utf-8">
  <link rel="shortcut icon" type="image/png" href="favicon.png">
  <meta name="author" content="Francisca Medina Concha">
  <meta name="description" content="Comparte tus conocimientos tomando café">
  <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700,800" rel="stylesheet">
  <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>
  <nav class="oscuro">
    <ul>
      <li><a href="#inicio">Ícono</a></li>
      <li><a href="#ubicacion">Ubicación</a></li>
      <li><a href="#proxima-charla">Próxima Charla</a></li>
      <li><a href="#eventos">Eventos anteriores</a></li>
      <li><a href="#contacto">Contacto</a></li>
    </ul>
  </nav>

  <header id="inicio" class="hero-section">
    <h1>Descubre lo último en tecnología bebiendo café</h1>
    <h2 class="subtitulo">Charlas, eventos y simposios sobre tecnología</h2>
  </header>

  <section id="ubicacion">
    
    <h2>¿Dónde nos juntamos?</h2>
    <p>Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia, Santiago.</p>
  </section>

  <section id="proxima-charla" class="sweet-brown">
    <div class="centrado">
      
      <div class="charla-texto">
        <h2>Próxima charla</h2>
        <h3>Big Data para Noobs</h3>
        <h4>Rafaela Valdés</h4>
        <h5>Data scientist, PhD Theoretical Physics, Carnegie Mellon University</h5>
        <p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer frente a los datos de una escala extrema.</p>
      </div>
    </div>
  </section>

  <section id="eventos">
    <h2>Eventos anteriores</h2>
    <article class="evento">
      
      <h3>Simposio Vegan DB</h3>
    </article>
  </section>
</body>
```

```
<article class="evento">

<h3>Machine learning 101</h3>
</article>

<article class="evento">

<h3>Scrum sin scream</h3>
</article>
</section>

<footer id="contacto" class="oscuro">
<p>Meet & coffe 2018. Todos los derechos reservados.</p>
</footer>
</body>
</html>
```

CSS

```
body {  
    text-align: center;  
    color: #000f08;  
    margin: 0;  
    font-family: 'Open Sans', sans-serif;  
}  
  
body * {  
    margin: 0;  
    border: 0;  
    padding: 0;  
}  
  
h1 {  
    font-weight: 800;  
    font-size: 3.75em;  
    margin-bottom: 0.25em;  
}  
  
h2 {  
    font-weight: 700;  
    font-size: 2.5em;  
}  
  
h3 {  
    font-size: 2em;  
}  
  
h4 {  
    font-size: 1.75em;  
}  
  
h5 {  
    font-size: 1.25em;  
}  
  
.subtitulo, h4, h5 {  
    font-weight: 400;  
}  
  
#ubicacion h2 {  
    margin: 1em auto;  
}  
  
.sweet-brown h3, .sweet-brown h5 {  
    margin-bottom: 0.5em;  
}  
  
#eventos h2 {  
    margin-bottom: 1em;  
}  
  
#eventos h3 {  
    margin: 1em auto;  
}
```

```
.oscuro {  
background: #0000f08;  
color: #ffffff;  
}  
  
.oscuro a{  
color: #ffffff;  
text-decoration: none;  
}  
  
.sweet-brown {  
text-align: left;  
background: #a4452c;  
color: #ffffff;  
padding: 3.125rem;  
white-space: nowrap;  
}  
  
.hero-section {  
text-align: right;  
background-image: url('../img/bg-hero.png');  
background-repeat: no-repeat;  
background-size: cover;  
color: #ffffff;  
text-shadow: 0.125em 0.25em 0.125em #707070;  
padding: 3.125rem 3.125rem 6.25rem 3.125rem;  
}  
  
section, footer {  
margin-top: 4.5rem;  
}  
  
footer {  
padding: 3.125rem;  
}  
  
.centrado {  
width: 75%;  
margin: 0 auto;  
}  
  
.charla-speaker, .charla-texto, .evento {  
display: inline-block;  
}  
  
.charla-speaker{  
width: 25%;  
vertical-align: middle;  
}  
  
.charla-texto {  
width: 72.5%;  
vertical-align: middle;  
margin-left: 2.5%;  
white-space: normal;  
}
```

```
.evento {  
    width: 30%;  
}
```

```
.evento img {  
    width: 75%;  
}
```

Tipografías desde archivos

Muchas veces nos pasarán un diseño y la tipografía ocupada no estará en [Google Fonts](#). Hay una forma de subir nuestros archivos tipográficos a nuestra web.

Formateando el menú (position)

Llegó la hora de formatear el menú. Podemos ver en la [maqueta final](#) que los elementos del menú:

1. Están alineados a la izquierda (`text-align: left;`)
2. Son desplegados uno al lado del otro (`display: inline-block;`)
3. Tienen márgenes entre ellos (`margin: 0 1em;`).

También podemos ver que la barra tiene ciertos paddings arriba y abajo y hacia ambos lados (`padding: 1.25rem 4rem;`).

Otro punto importante es que los ítems del `ul` no deben tener su viñeta por defecto, eso lo arreglaremos con la propiedad `list-style-type` con valor `none`.

Vamos a escribir entonces:

```
nav {  
    text-align: left;  
    padding: 1.25rem 4rem;  
}  
  
nav ul {  
    list-style-type: none;  
}  
  
nav ul li {  
    display: inline-block;  
    margin: 0 1em;  
}
```

Muy bien, ahora se ve mucho mejor:

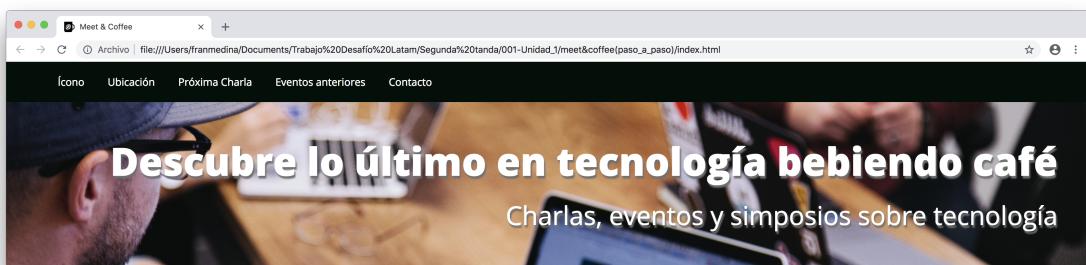


Imagen 42: Formatear menú

Position

Pero nos queda algo importante (además del ícono, que veremos más adelante).

En muchos sitios el menú nos sigue mientras hacemos scroll.

- [En Facebook](#)
- [En LinkedIn](#)
- [En W3Schools](#)

Eso facilita al usuario ir navegando por la página o sitio.

Por lo tanto, le añadiremos esa propiedad a nuestro menú, para que nos siga. La propiedad se llama `position`.

La propiedad de `position` especifica el tipo de método de posicionamiento utilizado para un elemento. Los valores pueden ser: `static`, `relative`, `fixed`, `absolute` o `sticky`.

Por defecto los elementos son `static`, o sea, no se posiciona de ninguna manera especial. Siempre se posiciona de acuerdo con el flujo normal de la página (uno bajo el otro o uno al lado del otro dependiendo de su `display`).

El valor que le vamos a poner al nav será `fixed`. Un elemento con la propiedad `position: fixed;` por defecto se posiciona en la esquina superior izquierda de la ventana de visualización del navegador y no se mueve de ahí aunque hagamos scroll. Hagamos la prueba:

```
nav {  
    text-align: left;  
    padding: 1.25rem 4rem;  
    position: fixed; /* esta es la propiedad que añadimos */  
}
```

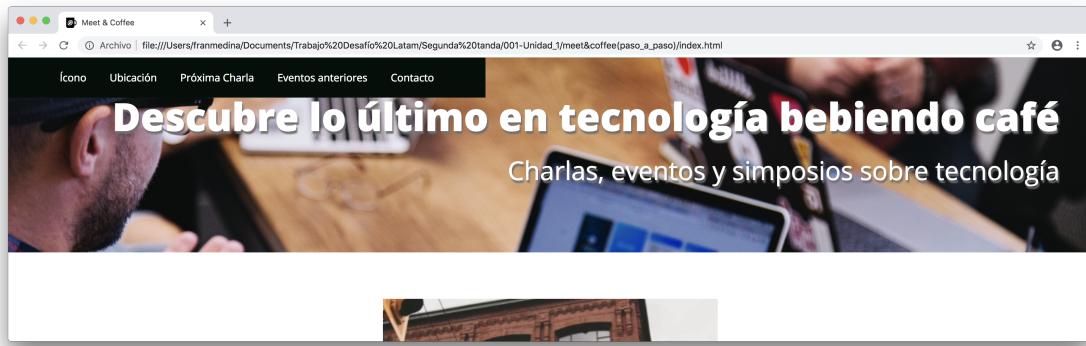


Imagen 43: Position

Nos quedó nuestro menú. Si hacemos scroll éste nos seguirá, pero quedó más pequeño y se sobrepuso al `<header id="inicio" class="hero-section">`.

Eso es porque salió del flujo normal de la página. Por ende, tenemos que hacer 2 cosas.

Primero, asignarle al nav un ancho del 100% del ancho de la pantalla:

```
nav {  
    text-align: left;  
    padding: 1.25rem 4rem;  
    position: fixed;  
    width: 100vw; /* esta es la propiedad que añadimos */  
}
```

Y segundo, darle un padding top mayor a `<header id="inicio" class="hero-section">` para no perder parte de su alto.

```
.hero-section {  
    text-align: right;  
    background-image: url('../img/bg-hero.png');  
    background-repeat: no-repeat;  
    background-size: cover;  
    color: #ffffff;  
    text-shadow: 0.125em 0.25em 0.125em #707070;  
    padding: 3.125rem 3.125rem; /* esto fue lo que modificamos */  
}
```

Lo hemos logrado:

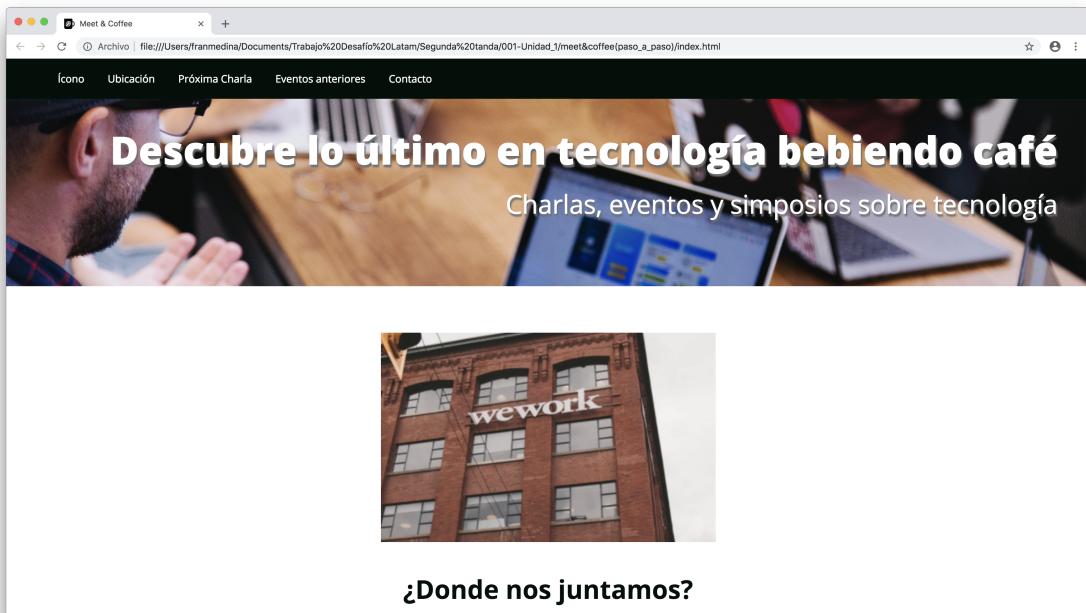


Imagen 44: Resultado nav y padding

Si quieres conocer más sobre la propiedad `position` te dejo este [link de W3Schools](#).

z-index

Muchas veces, con los temas de posición te verás enfrentado a que los elementos se sobreponen. Ahí conocerás la propiedad `z-index`, que define cómo se apilarán los elementos. Te dejo este [link de W3Schools](#) donde podrás conocerla mejor.

SVG y Font Awesome

SVG

Los gráficos vectoriales son muy útiles en muchas circunstancias. Tienen un tamaño de archivo pequeños y son altamente escalables, ya que no se pixelean cuando les haces zoom o amplías a un gran tamaño. SVG es un lenguaje basado en XML para describir imágenes vectoriales. SVG significa Scalable Vector Graphics.

Nosotros, en nuestro menú, deberíamos tener uno, que es un ícono de una taza de café.

Podemos ver en nuestra carpeta `assets/img` tenemos un archivo llamado `coffee-cup.svg`, que descargamos hace ya un tiempo desde la carpeta `assets`.

Una manera rápida de añadir svg a nuestra página es llamarla como imagen ``. Lo malo de esto es que no podremos aprovechar las posibilidades que nos da svg. Como controlar el contenido SVG con CSS, desde nuestro propio `style.css`.

Para poder lograr hacer eso, vamos a colocar el svg en línea, copiando el código que viene en el archivo y reemplazándolo por el texto que dice `Ícono`. Además a su `` contenedor le vamos a dar una clase `icono` para poder darle un tamaño específico a través de css.

```
<nav class="oscuro">
  <ul>
    <li class="icono"><a href="#inicio">
      <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 44 44"><path d="M11.8 26l-2.2 2.3c-.5-2.9-.7-7.5 2.2-11.2V26zm-.4 4.1c2.7.5 7.2.7 10.6-2.3h-8.5l-2.1 2.3zm5.3-16.2c-.8-2-1.7.5-2.4.9v8.5l2.4-2.6v-6.8zm2.5-.3v4.6l4-4.2c-1.1-3-2.5-5-4-.4zM16 25.2h8.1c.4-.8-.7-1.7.9-2.6h-6.6L16 25.2zm28-6.1V25c0 4.3-3.3 7.8-7.4 7.8h-1.8v2.9c0 4.6-3.6 8.3-7.9 8.3h-19C3.6 44 0 40.3 0 35.7V0h34.8v11.4h1.8c4.1 0 7.4 3.5 7.4 7.7zm-16.2.9c.1-2.6-.4-5-.8-6.4-.2-.9-.4-1.4-.4-1.4s-.5-.2-1.4-.4c-1.4-.4-3.6-.9-6.1-.8-.8 0-1.7.1-2.5.3-.8.2-1.6.4-2.4.7-.9.4-1.7.8-2.5 1.5-.6.5-1.1 1-1.6 1.6-4.5 5.4-3.4 12.5-2.6 15.5.2.9.4 1.4.4 1.4s.5.2 1.4.4c2.8.8 9.6 2 14.7-2.7.6-.5 1.1-1.1 1.5-1.7.6-.8 1-1.7 1.4-2.6.3-8.5-1.7.7-2.6.1-1.1.2-1.9.2-2.8zm13.2-.9c0-2.6-2-4.6-4.4-4.6h-1.8v15.1h1.8c2.4 0 4.4-2.1 4.4-4.6v-5.9zM21 20h4.4c0-1.6-1.3-4.4L21 20z"/></svg>
    </a></li>
    <li><a href="#ubicacion">Ubicación</a></li>
    <li><a href="#proxima-charla">Próxima Charla</a> </li>
    <li><a href="#eventos">Eventos anteriores</a></li>
    <li><a href="#contacto">Contacto</a></li>
  </ul>
</nav>
```

```
.icono {
  width: 2rem;
}
```

Si recargamos el navegador no lo vamos a poder ver, pero si inspeccionamos lo podremos encontrar. Resulta que este ícono es negro.

Para cambiar su color a blanco debemos crear un selector que apunte a ese svg en específico y ahí le debemos dar la propiedad `fill:` que es el relleno para los svg con el valor `#ffffff`

```
.icono svg {  
    fill: #ffffff;  
}
```

Y lo tenemos, ahora, falta alinear los `li` verticalmente. Eso ya lo sabemos hacer, con `vertical-align: middle;`.

```
nav ul li {  
    display: inline-block;  
    margin: 0 1em;  
    vertical-align: middle; /* esta es la propiedad que añadimos */  
}
```

Font Awesome

Otra forma de poner íconos a nuestra página es a través de **Font Awesome**. Font Awesome es un conjunto de herramientas de fuentes e íconos basado en CSS. Está diseñado para ser utilizado con elementos en línea (inline). Los elementos `<i>` y `` se usan ampliamente para los iconos.

Vamos a utilizar esta herramienta para poner los íconos restantes que nos faltan en el `<footer id="contacto" class="oscuro">`, los isotipos de GitHub, Twitter y LinkedIn.

Primero nos vamos a meter en fontawesome.com. Le daremos click al ítem del menú que dice **Start**.

Ahí nos saldrán tres sencillos pasos de cómo utilizarlo.

- **Paso 1:**

Coparemos y pegaremos este código

```
<link rel="stylesheet"
      href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
      integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
      crossorigin="anonymous">
```

dentro de la etiqueta `<head>` antes del llamado a nuestro estilo CSS propio.

```
<head>
  <title>Meet & Coffee</title>
  <meta charset="utf-8">
  <link rel="shortcut icon" type="image/png" href="favicon.png">
  <meta name="author" content="Francisca Medina Concha">
  <meta name="description" content="Comparte tus conocimientos tomando café">
  <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700,800" rel="stylesheet">
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
        integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
        crossorigin="anonymous"> <!--Esto fue lo que agregamos-->
  <link rel="stylesheet" href="assets/css/style.css">
</head>
```

- **Paso 2:** Buscaremos el ícono en la [galería de íconos de Font Awesome](#).

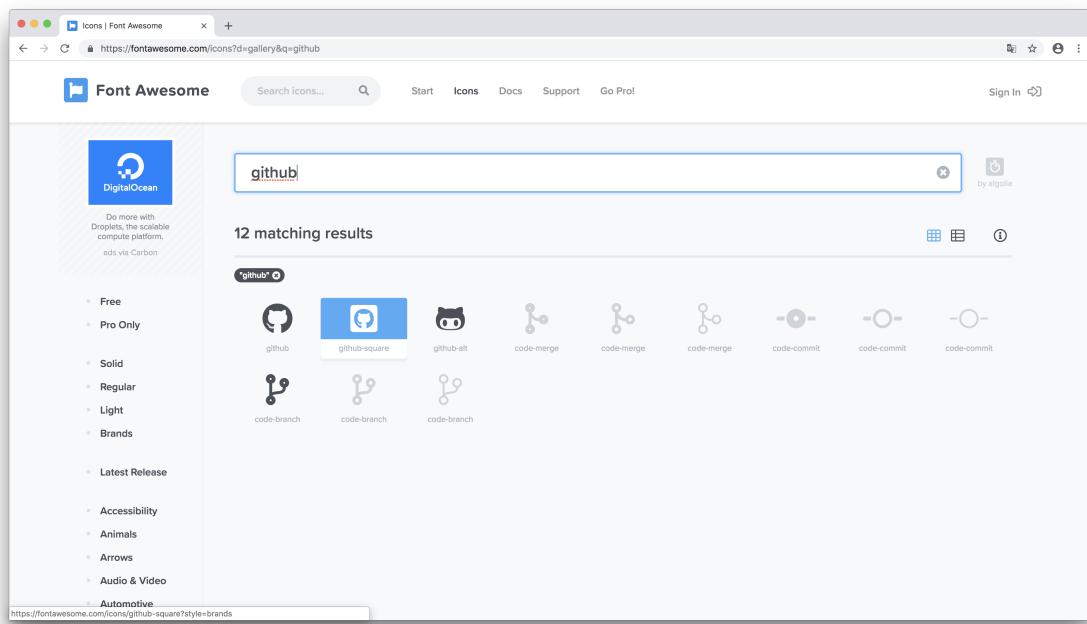


Imagen 45: Font awesome - buscar ícono

- **Paso 3:** Clickearemos el ícono

- **Paso 4:** Copiaremos la etiqueta `<i>` con todas las clases que nos entrega `<i class="fab fa-github-square"></i>`

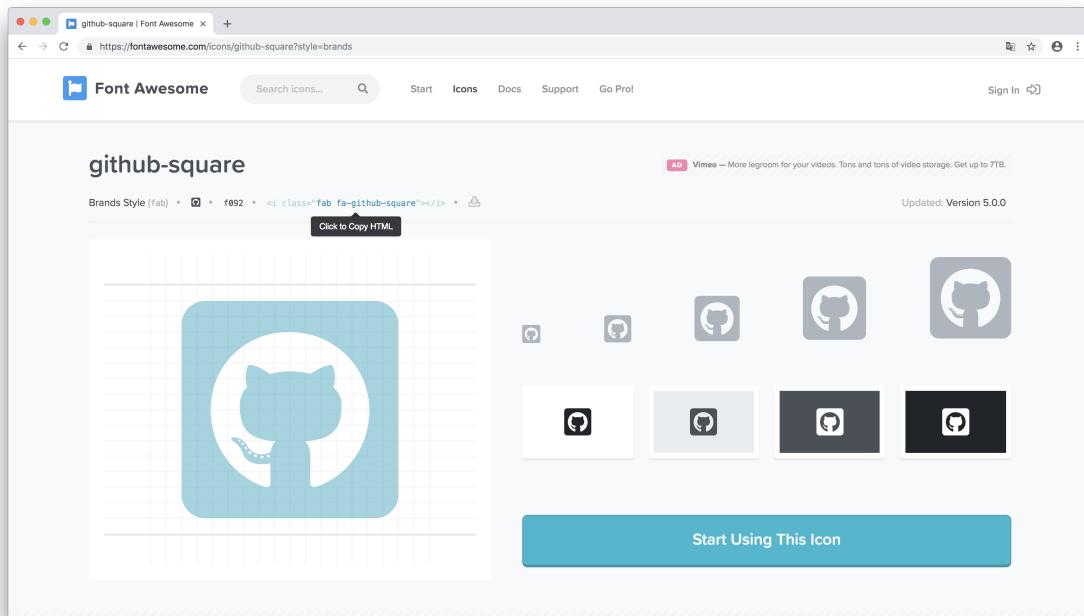


Imagen 46: Font awesome - Copiar etiqueta `<i>`

- **Paso 5:** Lo pegaremos donde corresponda.

```
<footer id="contacto" class="oscuro">
  <i class="fab fa-github-square"></i>
  <p>Meet & coffee 2018. Todos los derechos reservados.</p>
</footer>
```

Vamos a repetir el pase 2, 3, 4 y 5 por cada ícono que necesitemos (GitHub, Twitter y LinkedIn).

```
<footer id="contacto" class="oscuro">
  <i class="fab fa-github-square"></i>
  <i class="fab fa-twitter-square"></i>
  <i class="fab fa-linkedin"></i>
  <p>Meet & coffee 2018. Todos los derechos reservados.</p>
</footer>
```

Y veremos el resultado:

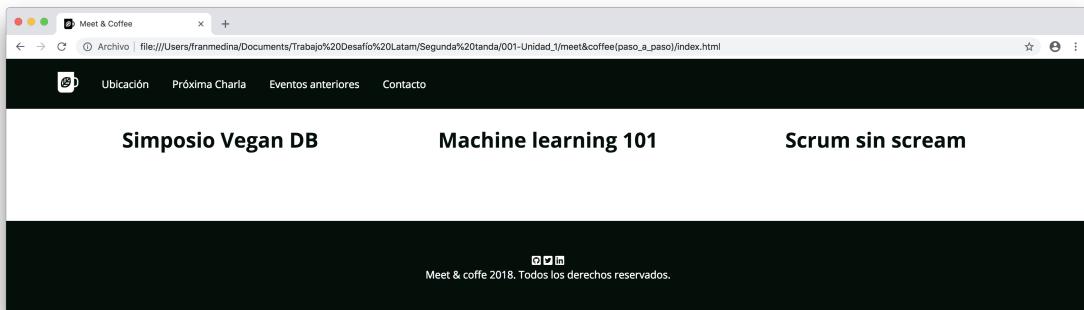


Imagen 47: Resultado Font Awesome

Muy bien, gracias a Font Awesome se han desplegado los íconos. Además, estos están blancos.

Es importante entender que estos íconos funcionan como fuentes, por lo tanto, los cambios que le hagamos a la fuente en el CSS se verán reflejados en estos íconos. Por ejemplo que el color de letra de las secciones con clase `oscuro` sean de color blanco va a aplicar a los íconos de Font Awesome que estén dentro de esas secciones.

Ahora, cada uno de esos íconos nos llevará a la red social específica. Eso lo haremos añadiéndole la etiqueta `<a>`.

```
<footer id="contacto" class="oscuro">
  <a href="https://github.com"><i class="fab fa-github-square"></i></a>
  <a href="https://twitter.com"><i class="fab fa-twitter-square"></i></a>
  <a href="https://www.linkedin.com"><i class="fab fa-linkedin"></i></a>
  <p>Meet & coffee 2018. Todos los derechos reservados.</p>
</footer>
```

Ahora sólo nos falta añadirle tamaño. Si bien podemos añadirle tamaño usando la propiedad `font-size`, la forma más recomendada es hacerlo al estilo Font Awesome. Para eso está la página [Sizing Icons](#) del sitio de Font Awesome donde nos indicarán cómo hacerlos más grandes.

```
<footer id="contacto" class="oscuro">
  <a href="https://github.com"><i class="fab fa-github-square fa-5x"></i></a>
  <a href="https://twitter.com"><i class="fab fa-twitter-square fa-5x"></i></a>
  <a href="https://www.linkedin.com"><i class="fab fa-linkedin fa-5x"></i></a>
  <p>Meet & coffee 2018. Todos los derechos reservados.</p>
</footer>
```

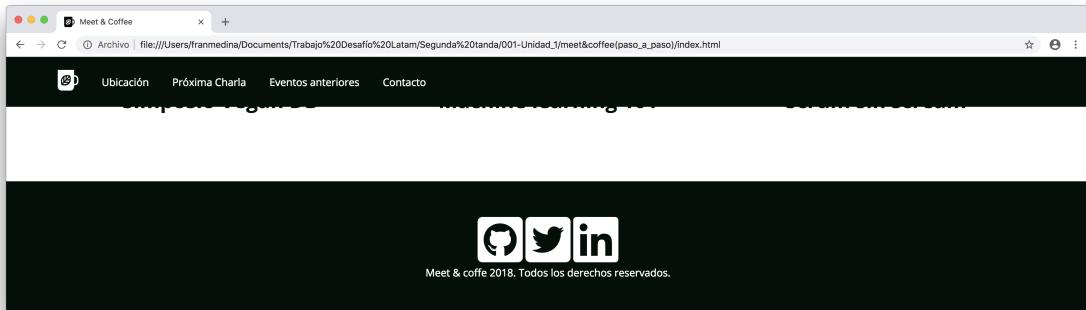


Imagen 48: Utilizar propiedad font-size

Ahora sólo nos toca añadirle la distancia entre íconos y estamos listos con nuestra página index.

```
footer i {  
    margin: 0 0.25em 0.5em 0.25em;  
}
```

Resultado

The screenshot shows a web browser window for a site titled "Meet & Coffee". The header includes a logo, navigation links for "Ubicación", "Próxima Charla", "Eventos anteriores", and "Contacto", and a search bar. The main banner features a man with a beard looking at a laptop screen, with the text "Descubre lo último en tecnología bebiendo café" and "Charlas, eventos y simposios sobre tecnología". Below the banner is a photo of a brick building with "we work" signage. A section titled "¿Donde nos juntamos?" provides meeting details. The "Próxima charla" section highlights "Big Data para Noobs" by Rafaela Valdés, featuring her photo and bio. The "Eventos anteriores" section shows three thumbnail images for past events: "Simposio Vegan DB", "Machine learning 101", and "Scrum sin scream". The footer contains social media icons for GitHub, Twitter, and LinkedIn, and a copyright notice: "Meet & coffee 2018. Todos los derechos reservados."

Imagen 49: Resultado con Font Awesome

HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Meet & Coffee</title>
    <meta charset="utf-8">
    <link rel="shortcut icon" type="image/png" href="favicon.png">
    <meta name="author" content="Francisca Medina Concha">
    <meta name="description" content="Comparte tus conocimientos tomando café">
    <meta name="keywords" content="charlas, eventos, simposios, tecnología, co-work">
    <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700,800" rel="stylesheet">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8IWUE00s/"
crossorigin="anonymous">
    <link rel="stylesheet" href="assets/css/style.css">
</head>
<body>
    <nav class="oscuro">
        <ul>
            <li class="icono"><a href="#inicio">
                <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 44 44"><path d="M11.8 26l-2.2
2.3c-.5-2.9-7.5 2.2-11.2V26zm-4.4 4.1c2.7.5 7.2.7 10.6-2.3h-8.5l-2.1 2.3zm5.3-16.2c-.8-2-1.7-5
2.4.9v8.5l2.4-2.6v-6.8zm2.5-.3v4.6l4-4.2c-1.1-.3-2.5-.5-4.4zM16 25.2h8.1c.4-.8-7-1.7-9-2.6h-6.6L16
25.2zm28-6.1V25c0 4.3-3.3 7.8-7.4 7.8h-1.8v2.9c0 4.6-3.6 8.3-7.9 8.3h-19C3.6 44 0 40.3 0
35.7V0h34.8v11.4h1.8c4.1 0 7.4 3.5 7.4 7.7zm-16.29c1-2.6-4-5-8-6.4-2-9-4-1.4-4-1.4s-5-2
1.4-4c-1.4-4-3.6-9-6.1-8-8 0-1.7-1.2-5.3-8.2-1.6-4-2.4-7-9.4-1.7-8-2.5 1.5-6.5-1.1 1-1.6 1.6-4.5 5.4-
3.4 12.5-2.6 15.5-2.9-4 1.4-4 1.4s-5.2 1.4-4c2.8-8.9 6.2 14.7-2.7-6.5 1.1-1.1 1.5-1.7-6.8 1-1.7 1.4-
2.6-3-8.5-1.7-7-2.6-1-1.2-1.9-2-2.8zm13.2-9c0-2.6-2-4.6-4.4-4.6h-1.8v15.1h1.8c2.4 0 4.4-2.1 4.4-
4.6v-5.9zM21 20h4.4c0-1.6-1-3-4-4.2L21 20z"/></svg>
                </a></li>
            <li><a href="#ubicacion">Ubicación</a></li>
            <li><a href="#proxima-charla">Próxima Charla</a></li>
            <li><a href="#eventos">Eventos anteriores</a></li>
            <li><a href="#contacto">Contacto</a></li>
        </ul>
    </nav>

    <header id="inicio" class="hero-section">
        <h1>Descubre lo último en tecnología bebiendo café</h1>
        <h2 class="subtitulo">Charlas, eventos y simposios sobre tecnología</h2>
    </header>

    <section id="ubicacion">
        
        <h2>¿Dónde nos juntamos?</h2>
        <p>Todos los martes y viernes, de 19:00 a 22:00 en We Work, Calle Baker 133, Providencia,
Santiago.</p>
    </section>

    <section id="proxima-charla" class="sweet-brown">
        <div class="centrado">
            
            <div class="charla-texto">
                <h2>Próxima charla</h2>
                <h3>Big Data para Noobs</h3>
                <h4>Rafaela Valdés</h4>
            </div>
        </div>
    </section>
```

```
<h5>Data scientis, phD Theoriccal Physics, Carneige Mellon University</h5>
<p>En esta charla revisaremos las técnicas y tecnologías que hacen que sea económico hacer
frente a los datos de una escala extrema.</p>
</div>
</div>
</section>

<section id="eventos">
<h2>Eventos anteriores</h2>
<article class="evento">

<h3>Simposio Vegan DB</h3>
</article>

<article class="evento">

<h3>Machine learning 101</h3>
</article>

<article class="evento">

<h3>Scrum sin scream</h3>
</article>
</section>

<footer id="contacto" class="oscuro">
<a href="https://github.com"><i class="fab fa-github-square fa-5x"></i></a>
<a href="https://twitter.com"><i class="fab fa-twitter-square fa-5x"></i></a>
<a href="https://www.linkedin.com"><i class="fab fa-linkedin fa-5x"></i></a>
<p>Meet & coffe 2018. Todos los derechos reservados.</p>
</footer>
</body>
</html>
```

CSS

```
body {  
    text-align: center;  
    color: #000f08;  
    margin: 0;  
    font-family: 'Open Sans', sans-serif;  
}  
  
body * {  
    margin: 0;  
    border: 0;  
    padding: 0;  
}  
  
h1 {  
    font-weight: 800;  
    font-size: 3.75em;  
    margin-bottom: 0.25em;  
}  
  
h2 {  
    font-weight: 700;  
    font-size: 2.5em;  
}  
  
h3 {  
    font-size: 2em;  
}  
  
h4 {  
    font-size: 1.75em;  
}  
  
h5 {  
    font-size: 1.25em;  
}  
  
.subtitulo, h4, h5 {  
    font-weight: 400;  
}  
  
#ubicacion h2 {  
    margin: 1em auto;  
}  
  
.sweet-brown h3, .sweet-brown h5 {  
    margin-bottom: 0.5em;  
}  
  
#eventos h2 {  
    margin-bottom: 1em;  
}  
  
#eventos h3 {  
    margin: 1em auto;  
}
```

```
.oscuro {  
background: #0000f08;  
color: #ffffff;  
}  
  
.oscuro a{  
color: #ffffff;  
text-decoration: none;  
}  
  
.sweet-brown {  
text-align: left;  
background: #a4452c;  
color: #ffffff;  
padding: 3.125rem;  
white-space: nowrap;  
}  
  
.hero-section {  
text-align: right;  
background-image: url('../img/bg-hero.png');  
background-repeat: no-repeat;  
background-size: cover;  
color: #ffffff;  
text-shadow: 0.125em 0.25em 0.125em #707070;  
padding: 6.25rem 3.125rem;  
}  
  
section, footer {  
margin-top: 4.5rem;  
}  
  
footer {  
padding: 3.125rem;  
}  
  
.centrado {  
width: 75%;  
margin: 0 auto;  
}  
  
.charla-speaker, .charla-texto, .evento {  
display: inline-block;  
}  
  
.charla-speaker{  
width: 25%;  
vertical-align: middle;  
}  
  
.charla-texto {  
width: 72.5%;  
vertical-align: middle;  
margin-left: 2.5%;  
white-space: normal;  
}
```

```
.evento {  
    width: 30%;  
}  
  
.evento img {  
    width: 75%;  
}  
  
nav {  
    text-align: left;  
    padding: 1.25rem 4rem;  
    position: fixed;  
    width: 100vw;  
}  
  
nav ul {  
    list-style-type: none;  
}  
  
nav ul li {  
    display: inline-block;  
    margin: 0 1em;  
    vertical-align: middle;  
}  
  
.icono {  
    width: 2rem;  
}  
  
.icono svg {  
    fill: #ffffff;  
}  
  
footer i {  
    margin: 0 0.25em 0.5em 0.25em;  
}
```