Curso de Layouts Responsivos.

Clase 1

***Emmet:*** Herramienta que viene por defecto en la gran mayoría de editores de código. Permite, al revés de escribir línea por línea de código, abreviar y generar los códigos más rápido.

***Signo >:*** Deja claro que dentro de una etiqueta vendrá otra nueva. Siendo la correlación padre > hijo.

***Signo +:*** Deja claro que dentro de una etiqueta hija, vendrán añadidas otras etiquetas siendo la correlación padre>hijo+hermano1+hermano2+hermano3.

***Signo \*:*** Multiplica la cantidad de elementos que tendrá una etiqueta dentro de sí.

***Medida REM:*** REM es una unidad relativa a la propiedad font-size de la etiqueta <html>, por lo tanto si la etiqueta <html> tiene 16px de font-size 1 REM equivale a 16px. REM proviene de “Root ephemeral” que en traducción técnica es la “variable de la raíz”, o sea, es una variable en relación de la propiedad font-size de la etiqueta raiz (HTML). Tiene el mayor nivel de jerarquía y será adaptable a todos los tamaños del navegador.

***Medida EM:*** EM es relativa a la propiedad font-size de la etiqueta madre, por lo tanto, si la etiqueta madre es un font-size de 16px, 1 EM equivale a 16px. EM proviene de “ephemeral” que significa “variable” y es una variable de la propiedad font-size de la etiqueta madre.

***Imagen PNG:***

* Poco compacto
* Se mantiene la calidad de la imagen
* Tamaño elevado: 29.6Kb

***Imagen JPG:***

* Muy compacto
* La calidad de la imagen es inferior
* Tamaño leve: 9.29Kb

***Imagen GIF:***

* La calidad de la imagen es baja
* Uso limitado
* Tamaño depende de la calidad de la imagen: 199Kb

***Imagen SVG:***

* Al acercarse no se pierde tanto la calidad
* Imágenes vectoriales en dos dimensiones
* Tamaño variable: 4.26Kb,
* Algunas imágenes pueden perder color
* El SVG no mantiene detalles de una imagen compleja

***¿Cuándo utilizar SVG?***

* Logotipos
* Íconos
* Ilustraciones sencillas

Clase 2

***Mobile First:*** Concepto en donde la versión móvil de una página web debe ser la correcta.

**Ventajas:**

* Se concentra en lo esencial.
* La información es de más fácil acceso.
* El rendimiento se potencia al máximo.
* Se reduce el código fuente.
* Se hace el uso de HTML 5, dejando de lado parcialmente JavaScript.

**Para saber más:**

Para seguir un concepto de desarrollo, sea mobile-first o sea desktop-first, puedes seguir lo que prefieras porque no existe una regla que indique cual sea primero. Cada uno de ello tiene sus ventajas y desventajas que deben ser evaluadas de acuerdo con las necesidades del proyecto.

Para demostrar, aquí están algunas motivaciones de utilizar el desarrollo mobile-first:

* Gran parte de los accesos y ventas son de dispositivos móviles;
* El diseño es minimalista y simplificado;
* Enfoque en el contenido.

Las motivaciones para utilizar el desarrollo desktop-first:

* La interfaz tiene features más ricas;
* Mayor capacidad de ejecución de las instrucciones;
* El producto es optimizado para desktop (ejemplo: Google Docs).

Así que al desarrollar es importante analizar todos los puntos que influencian en el proyecto. Clientes, features, costo y lenguaje visual son algunos de los puntos que favorecen un abordaje sobre el otro.

***Meta Viewport:*** Habilita al navegador para que reconozca esa visualización de la pagina como si fuera un dispositivo móvil.

***Initial Scale:*** Define cuanto zoom (o como se muestra el contenido) puede ser aplicado en la pagina.

***MaxWidth:***

***BoxSizing:***

***Bloques:*** Usar propiedad margin.

***Textos:*** Usar propiedad padding.

***Variables CSS:*** Se pueden crear usando el selector :root {variables}. El root es una etiqueta html, pero su especificidad es mayor que la del selector html normal. La sintaxis de estas variables son las siguientes: --nombre-variable: valor;

Entonces… Declarar dentro de algún selector (que se logra ser el :root{}) escribiendo el --nombre-de-la-variable: valor y llamar las variables según la sintaxis, utilizando una función var con el nombre de la variable dentro de esa función. Opción correcta porque precisamos declarar variables CSS dentro de algún selector. Y para utilizar esas variables necesitamos seguir la sintaxis var(--nombre-de-la-variable).

***Referenciar variables CSS:*** font-family: **var(--open-sans);**

Clase 3

Emmet signo (.): Permite crear y nombrar las clases del element.

Etiqueta html Figure: El elemento HTML <figure> representa contenido independiente, a menudo con un título. Si bien se relaciona con el flujo principal, su posición es independiente de éste. Por lo general, se trata de una imagen, una ilustración, un diagrama, un fragmento de código, o un esquema al que se hace referencia en el texto principal, pero que se puede mover a otra página o a un apéndice sin que afecte al flujo principal.

Etiqueta html figC: El elemento **HTML <figcaption>** representa un subtítulo o leyenda asociado al contenido del elemento padre [<figure>](https://developer.mozilla.org/es/docs/Web/HTML/Element/figure), pudiendo ser colocado como primer o último hijo. Es importante destacar que el elemento **<figcaption>** es opcional.

Propiedad line-height: La propiedad [CSS](https://developer.mozilla.org/es/docs/Web/CSS) line-height establece la altura de una casilla remarcada por líneas. Comúnmente se usa para establecer la distancia entre líneas de texto. A nivel de elementos de bloque, define la altura mínima de las casillas encuadradas por líneas dentro del elemento. En elementos en linea no reemplazables, especifica la altura que se usa para calcular la altura de la casila encuadrada por líneas.

Valor 100vw: Representa el ancho del elemento en relación del ancho total de la pantalla del dispositivo. vw está específicamente relacionada a la largura de la pantalla (en pixel).

Limitaciones de los dispositivos:

Teniendo en cuenta las pantallas compactas:

Espaciamientos superiores, inferiores y laterales de los elementos de la página. La orientación de las pantallas es vertical, por tanto los elementos deben de mostrarse adecuadamente en ese sentido.

Precision del toque. Algunos dispositivos puede que no tengan la precisión suficiente a la hora de tocar en la pantalla ciertos elementos. Se debe tener en cuenta el tamaño de los elementos dependiendo del dispositivo. Se recomienda dejar el suficiente espacio entre los elementos de la pagina para así dar cierto margen de imprecisión al usuario.

Señal del dispositivo o conectividad a la red. Puede que dependiendo de la señal, ciertos elementos o datos carguen o no de la manera adecuada.

Clase 4

***Background-repeat:***

***Background-position:***

***Etiqueta address:***

***Etiqueta article:***

***Propiedad de video controls:***

Ventajas de escribir un código semántico.

* ***Para que otras personas desarrolladoras puedan entender:*** Debemos recordar que nuestro código tal vez necesite pasar a otras personas y ellas necesitan entender lo que sucede en nuestro código cuando tengan que leerlo.
* ***Para atender a los requisitos de accesibilidad:*** Debemos dejar nuestra página lo más inclusiva y para eso precisamos utilizar todas las etiquetas correctas para demarcar todo el contenido de la página web.
* ***Para herramientas de indexación:*** Hay muchos bots (herramientas automáticas) que verifican todo el código de nuestro proyecto, después hacen una validación para cambiar el posicionamiento de nuestra página en los resultados de motores de búsqueda (ejemplo: Google, Bing, Yahoo).

Clase 5

Mediaqueries: Las media queries permiten crear páginas con estilos diferentes en el mismo archivo CSS por muchas condiciones diferentes, como por ejemplo, la largura de la pantalla de un dispositivo. Ejemplo: @media (min-width: 768px) { “estilo aquí” }.