Práctica U2 – CSS Avanzado

Proyecto CSS

2º DAW – Diseño de Interfaces Web

Rosa Medina / Arturo Bernal
IES San Vicente
Curso 2025 – 2026

Sumario

Objetivo

Transformar una página web estática básica en un portal de gaming moderno, adaptable y accesible. Partirás de un archivo **index.html** disponible en Aules, un archivo **.stylelintrc.json** y la configuración de Vite (**vite.config.js**). Deberás crear un proyecto Vite e instalar stylelint para que verifique tus archivos en el script prestart. Una vez hecho eso, crearemos un estilo CSS aplicando una arquitectura CSS modular y técnicas avanzadas de maquetación.
¡**Importante**! Es obligatorio **usar nesting,** por eso tenéis puesto como reglas que se debe aplicar nesting en el archivo de configuración stylelintrc.

El resultado final deberá verse como las capturas que se adjunta al final del enunciado.

Requisitos de la práctica

Requisitos:

1. Crea un proyecto **Vite** e instala y configura **StyleLint** en el proyecto (utiliza el archivo proporcionado). Si no la tienes todavía, instala también la **extensión** de stylelint para VS Code.
2. **HTML:** No se puede modificar nada del archivo index.html proporcionado.

Arquitectura CSS Modular (@layer e @import)

El archivo CSS que tenemos enlazado en nuestro proyecto ./src/style.css será nuestro archivo principal, en él sólo definiremos las capas (orden) e importaremos los módulos CSS que vamos a crear.

1. Definiremos las capas CSS a utilizar. Las capas que tendremos son (en este orden): base, layout, components, theme y utilities.
2. Creamos los módulos (los 5 archivos CSS que usaremos para dar estilo a nuestro proyecto):
    1. **base.css**: En este archivo definiremos las variables a utilizar en nuestro proyecto, haremos el reseteo básico de las propiedades CSS que pone por defecto el navegador, y le daremos los estilos globales del body

2. **layout.css**: En este archivo le daremos estilos al header, footer, main, aside, así como al page-container y al grid principal que tendrá el body.
3. **components.css**: Estilos para los elementos card, button, nav-links, submenu, hamburguer-menu, donde si nos fijamos se podrían reutilizar en otros proyectos.
4. **theme.css**: Estilos que dependerán de las preferencias de usuario.
5. **utilities.css**: En este archivo tendremos las animaciones

3. Importamos esos módulos en su capa correspondiente.

Requisitos CSS - Maquetación

1. En el **body** (definido en base.css), debemos darle las propiedades para que mediante grid tengamos el header arriba del todo, el footer abajo y el contenido ocupe todo el espacio restante. Para eso, no olvides decirle que el alto mínimo deberá ser la altura del viewport. El objetivo es hacer que si la página tiene poco contenido el footer se mantenga abajo.
2. Nuestro layout principal (main & aside) deberá ser responsive, deberemos ajustar ese tamaño responsive para que no se nos rompa el diseño, pero los diseños que definiremos serán:
    1. **Móvil**: main y aside pasarán a estar en 1 única columna.
    2. **Escritorio**: main-aside serán 2 columnas, por ejemplo 3fr y 1fr.
3. **Fallbacks (@supports)**: Deberemos incluir un bloque para aquellos navegadores que no soporten las propiedades interpolate-size, puesto que al hacer hover sobre las cards se despliegan y si le ponemos la palabra reservada auto a la altura no vamos a ver la animación que queramos darle. Si ves que hay más propiedades que aplicas donde debas usar @supports hazlo.
4. **Menú** y **submenú responsive**: En el HTML tienes la hamburguesa implementada, ahora falta hacerlo funcional, es decir:
    1. Usad la propiedad z-index para desplegar el menú en el móvil, tened en cuenta que debe estar por encima de los cards
    2. submenú: El submenú deberá estar oculto por defecto y mostrarse al hacer hover sobre su elemento padre en el escritorio. En el móvil estará integrado de manera visible en el menú desplegado.
5. **Imágenes del card**: Asegúrate que todas las imágenes de las cards tengan un aspect-ratio 16/9 y tengan un object-fit cover para evitar deformaciones.
6. **Transiciones y filtros**:
    1. Las imágenes de las cards tendrán un filtro grayscale por defecto, al hacer hover ese filtro se quitarán y se le dará un ligero zoom. **Ambos efectos deben usar transition** para ser suaves.

7. **Animaciones (@keyframes)**:
    1. Crea una animación **fade-in** en utilities.css, esta animación se le aplicará a las cards para que aparezcan al cargar la página. Para ello, usa **animation-delay** en cada **nth-child** de tal manera que aparezcan de manera escalonada.
8. **Accesibilidad (Preferencias de usuario, theme.css)**:
    1. **Tema light-dark**: Implementa un modo oscuro completo.
    2. **Movimiento reducido**: Define la duración de tus transiciones y animaciones utilizando variables CSS. Por ejemplo --transition-duration:0,3s. Usa prefers-reduce-motion para cambiar esas variables a un valor casi nulo (Ejemplo 0,01s) y desactivar así los efectos.

Estilos a tener en cuenta en cada archivo

style.css

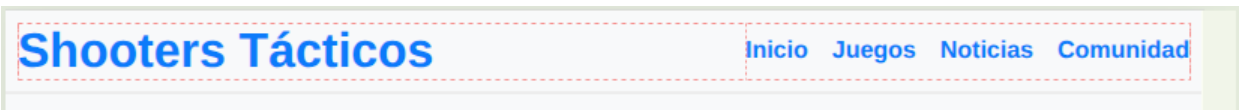
- En este archivo no tendremos ningún estilo, simplemente importamos y definimos las capas

base.css

- Define las variables para los colores, tipografía, bordes y sombras, duración de animaciones y transiciones y propiedades extra como el cover de las imágenes.
- Resetea todos los estilos que se da por defecto el navegador
- Establece el grid que tiene el body
- Asigna los colores a los headings de nuestra web (h1, h2, h3), usa para ello algún selector avanzado de los vistos.
- Da un margen interno por defecto al header, footer, main y aside. Esta especificidad será valorada como 0, de tal manera que si luego se reasigna más abajo/arriba haya sido sobreescrita, para conseguir esto usa los selectores avanzados mencionados arriba.

layout.css

- El header estará alineado horizontalmente al centro y el header-content será un flex donde tengamos los ítems separados. Estos items no podrán actuar en modo multilínea (aunque esto también nos debemos asegurar en el mediaquery). El heading h1, deberá tener un factor de crecimiento de 1, de tal manera que ocupe más espacio que los ítems del menú.



- El page-container (main & aside), será un grid de una columna centrado donde la anchura de este page es el 100% del ancho.

    - Crea un media query de tal forma que si el ancho de la pantalla es superior a 900px, entonces el grid será de 2 columnas (3fr 1fr por ejemplo).
- Tanto el main como el aside será un contenedor, ponedle un nombre a cada uno de ellos.
    - El card-container, será un grid de 1 columna. Estos valores se los estamos dando por defecto, los gestionaremos realmente con los @media y @container.
        - Gestiona este contenedor para que cuando sea como mínimo 500px el grid sea de 2 columnas.
        - Además gestiona este contenedor para que cuando sea como mínimo 800px el grid sea de 3 columnas.
    - El aside, tendrá una altura que se ajustará a su contenido, el heading tendrá un borde inferior, y las listas no tienen el “decorado” por defecto, estarán en modo flex-columnas. Los enlaces de estos elementos de la lista no tendrán decorado. Al hacer hover sobre cada ítem de la lista se aplicará un estilo distinto cambio letra/fondo.
        - Gestiona este contenedor para que cuando el ancho sea como máximo 200px tenga otro estilo, por ejemplo un text-align y margen interno distinto.
- El texto del footer estará alineado en el centro, borde, color de fuente...

components.css

- Vamos a darle a las cards un estilo en este archivo, fondo, borde, sombra, ocultaremos si hay un overflow, y gestionaremos las transiciones de transform, box-shadow y altura que se aplicarán al hacer un hover sobre la misma. Es importante que al poner en el hover que la altura del hover pase a ser auto deberemos asegurarnos la compatibilidad de la propiedad interpolate-size, y hacer que en caso de no soportarlo darle una altura predeterminada. Las imágenes del card estarán con un aspect-ratio 16/9 y deberán ser cover.
- Para aquellas card que TENGAN una imagen tengan un descendiente de la clase .card-content deberán tener y border-top que quieras.
- Para aquellas cards que no tengan la clase featured aplica otro borde distinto.
- Aplícale el estilo que quieras a los botones (button class) haciendo que haya una transición al hacer hover y cambie de color.
- Para el menú de navegación (nav-links), deberá estar sin “decorado” de lista, y en modo flex, debemos quitarle la decoración a los enlaces.
- El menu-toggle inicialmente deberá estar oculto (display:none), al igual que el menú hamburguesa
- Para el hamburguer-menu la dirección de este flex será en modo columna, los elementos span que tiene estarán en modo bloque con una altura determinada y un ancho. Aplica un borde, fondo y haz que todas las transiciones se apliquen en el mismo tiempo usando la función ease-in-out.
- El submenú estará ocultado inicialmente (display: none), en una posición absoluta. Dale un fondo, shadow, márgenes, asegúrate que esté visible (z-index), pon un opacity a 0 y

desplazalo en el ejeY 10px. Recuerda poner una transición para el opacity y el desplazamiento.

    - Pon un fondo diferente a los enlaces del submenú al hacer un hover

- Al elemento que tiene has-submenu, ponle una posición relativa, haciendo que al hacer hover esté en modo bloque (antes estaba a none), tenga un opacity de 1 (antes estaba a 0) y se vaya al ejeY 0 (antes lo pusimos a 10). Las propiedades de antes las dimos en el punto de arriba del submenú.

Parte responsive del menú (móviles)

En mi caso es para pantallas como máximo de 768px, pero ajustad siempre esas dimensiones a vuestras fuentes, tamaños, márgenes, etc. que no aparezca “un roto”.

- El header-content estará centrado
- Para el nav-links deberemos darle una posición fija, al principio le daremos los valores arriba 0, derecha -100%, anchura 250px (por ejemplo), altura 100vh. Dale un color de fondo, sombra y posiciónalo en modo flex-columna. Recuerda aplicar un z-index alto pero no tan alto como la hamburguesa que le daremos a continuación. No olvides gestionar la transición a la derecha con una función ease-in-out.
- Para la hamburger-menu, será un flex con posición fija: arriba 20px, derecha 30px, y un gran z-index.
- Vamos a simular el evento click PERO usando CSS, ya que el menú es un label asociado a un input de tipo checkbox. Para el input que ha sido marcado (checked), el hermano con la clase nav-links posiciónalo a la derecha con valor 0 (inicialmente -100%).
- Ahora animaremos el icono X, para aquel el input que ha sido marcado y vaya seguido de un hermano con la clase hamburger-menu:
    - span que sea primer hijo: rotación de 45deg y una translación 5px 5px.
    - span que sea segundo hijo: opacity de 0
    - span que sea tercer hijo: rotación de -45deg, y una translación de 7px y -6px.
- La clase submenú estará en una posición estática, en modo bloque, con una opacidad de 1. Sin transformaciones, ni shadow (está visible en todo momento) ponle fondo, ancho, márgenes...

theme.css

- Lo primero que haremos será modificar el valor de las variables definidas para cuando el tema sea dark, colores, y fondos. Haciendo que los enlaces al hacer hover cambien de fondo, y los cards que no sean featured tengan un borde diferente.
- Vamos a controlar las animaciones, es decir, debemos cambiar la duración de las transiciones y animaciones (sobreescribimos esas variables), le quitamos la transición y animación a las card, button, hamburguer-menu span, nav-links, submenu y card img.

Importante, nos aseguramos que las cards sean visible si la animación se anula (poned opacity a 1)

- Para cuando el contraste sea aumentado (more), ampliamos los bordes del botón, card y los enlaces del sidebar

utilities.css

- Crea un keyframe llamado fade-in que sea desde una opacidad 0 y translacion en el ejeY 20px a una opacidad 1, y translación en el ejeY de 0.
- Aplica la animación que has creado en la card, inicia dándole una opacidad 0 para que la animación funcione y usa la propiedad animation para cambiar esto.
- Crea una animación escaloanda para un efecto más dinámico, es decir, haz que cada hijo de la card tenga una animación delay de un tiempo distinto y que vayan apareciendo así poco a poco.
- A las imágenes de los cards aplícale un filtro de escapa de grises y usa de la transición para el filtro y escalado (transform).
- Cuando hagamos un hover sobre la imagen quita el filtro de gris dándole valor 0, y haz ese transform dándole un zoom.

Puntuaciones

- style.css → **0,5 puntos**
- base.css → **1,5 puntos**
- layout.css → **2 puntos**
- components.css → **3 puntos**
- theme.css → **1,5 puntos**
- utilities.css → **1,5 puntos**