Informe: Migración a Metodología BEM y Mejores Prácticas de HTML/CSS

Índice

Introducción

Este informe documenta el proceso de migración de un sitio web de presentación profesional hacia la metodología BEM (Block, Element, Modifier). El objetivo principal fue mejorar la estructura del código, haciéndolo más mantenible, escalable y fácil de entender, sin alterar significativamente el aspecto visual del sitio.

El proyecto consta de un sitio web personal para un desarrollador Python que incluye diferentes secciones: presentación principal (hero), sobre mí, proyectos destacados, habilidades y un formulario de contacto.

Análisis del Código Original

Problemas Identificados

El código HTML y CSS original presentaba varios problemas comunes que dificultaban su mantenimiento y escalabilidad:

1. Inconsistencia en la Nomenclatura de Clases

El código utilizaba varios estilos de nomenclatura simultáneamente:

```html
<!-- Mezcla de estilos de nomenclatura -->
<div class="spotlight"></div>        <!-- Nombre simple -->
<section class="hero"></section>     <!-- Nombre simple -->
<img class="sobremi-img">            <!-- kebab-case -->
<section class="sobre_mi"></section> <!-- snake_case -->
<div class="tag-box2"></div>         <!-- Numeración en clases -->
```

Esta inconsistencia crea confusión y dificulta predecir cómo se nombrarán las nuevas clases.

2. Falta de Jerarquía Explícita

Las relaciones entre elementos estaban implícitas por su anidación en HTML, pero no por su nomenclatura:

```html
<section class="hero">
    <div class="content">
        <h1 class="soluciones">Título</h1>
    </div>
</section>
```

Esto hace difícil entender qué elementos pertenecen a qué componentes solamente leyendo el CSS.

3. Selectores Genéricos

Se usaban clases genéricas que podían provocar colisiones:

```css
.section-title { /* Estilo que podría aplicarse en diferentes contextos */ }
.content { /* Nombre demasiado genérico */ }
.content2 { /* Nombre poco descriptivo con numeración */ }
```

4. Estructura CSS Desorganizada

El CSS no seguía una estructura clara por componentes, mezclando estilos de diferentes secciones.

Diagrama de la Estructura Original

```
├── HTML
│   ├── spotlight (clase genérica)
│   ├── bg-wrapper (clase genérica)
│   │   └── image-gradient (clase específica pero sin relación explícita)
│   ├── hero (componente/sección)
│   │   ├── robot-container (componente sin relación explícita)
│   │   └── content (clase genérica)
│   │       ├── soluciones (clase específica pero sin relación explícita)
│   │       └── hero-card (mezcla componente con función)
│   └── sobre_mi (snake_case)
│       └── sobremi-img (kebab-case, inconsistente con padre)
│
└── CSS
    ├── Selectores genéricos (.content, .section-title)
    ├── Propiedades repetidas en múltiples selectores
    └── Media queries generales al final del archivo
```

La Metodología BEM

¿Qué es BEM?

BEM (Block, Element, Modifier) es una metodología de nomenclatura para clases CSS desarrollada por Yandex. Proporciona una convención clara para nombrar clases que facilita el entendimiento de la estructura del código.

Componentes Básicos de BEM

Bloque (Block)

Un componente independiente y reutilizable de la página.

- Ejemplo: `.hero`, `.sobre-mi`, `.proyectos`
- Características: Funciona como una entidad autónoma y tiene significado por sí mismo.

Elemento (Element)

Una parte de un bloque que no tiene significado independiente.

- Sintaxis: `.bloque__elemento`
- Ejemplo: `.hero__title`, `.sobre-mi__imagen`
- Características: Siempre forma parte de un bloque y se separa con doble guion bajo.

Modificador (Modifier)

Una variante o estado de un bloque o elemento.

- Sintaxis: `.bloque--modificador` o `.bloque__elemento--modificador`
- Ejemplo: `.hero__button-container--alt`, `.sobre-mi__titulo--grande`
- Características: Indica una versión diferente del componente y se separa con doble guion medio.

Ventajas de BEM

1. **Claridad estructural**: La relación entre elementos es explícita en el nombre de las clases.
2. **Modularidad**: Los componentes son independientes y reutilizables.
3. **Especificidad controlada**: Se evitan conflictos y cascadas CSS complejas.
4. **Mantenibilidad**: El código es más fácil de entender, modificar y depurar.

Implementación de BEM en el Proyecto

Cambios Principales en el HTML

1. Estructura Base

```html
<body class="page">
    <div class="effect__spotlight"></div>
    <div class="background">
        <img class="background__image" src="./bg.jpg" alt="gradient">
    </div>
```

2. Estructura de Encabezado

```html
<header class="header">
    <nav class="header__nav">
        <a href="#sobre-mi" class="header__link">Sobre mí</a>
        <!-- Más enlaces -->
    </nav>
</header>
```

3. Sección Hero

```html
<section class="hero">
    <div class="hero__robot-container">
        <spline-viewer class="hero__robot-3d" url="..."></spline-viewer>
    </div>
    <div class="hero__content">
        <h1 class="hero__title">Soluciones digitales</h1>
        <p class="hero__card">...</p>
        <div class="hero__buttons">
            <div class="hero__button-container">
                <a href="#proyectos" class="hero__button">Ver Proyectos</a>
            </div>
            <div class="hero__button-container hero__button-container--alt">
                <a href="#contacto" class="hero__button hero__button--contact">Contactar</a>
            </div>
        </div>
    </div>
</section>
```

Cambios Principales en el CSS

1. Organización por Componentes

El CSS se reorganizó para agrupar los estilos por componente:

```css
/* VARIABLES GLOBALES */
:root {
    --primary-color: #6600c5;
    /* Otras variables */
}

/* ESTILOS BASE */
.page {
    font-family: var(--body-font);
    /* Más estilos */
}

/* HEADER */
.header {
    /* Estilos de header */
}
.header__nav {
    /* Estilos de navegación */
}
.header__link {
    /* Estilos de enlaces */
}

/* SECCIÓN HERO */
.hero {
    /* Estilos de sección hero */
}
.hero__content {
    /* Estilos del contenido del hero */
}
/* Y así sucesivamente para cada componente */
```

2. Nomenclatura BEM Consistente

Se implementó una nomenclatura consistente para todos los elementos:

```css
/* Bloques */
.hero { }
.sobre-mi { }
.proyectos { }

/* Elementos */
.hero__title { }
.sobre-mi__imagen { }
.proyectos__grid { }

/* Modificadores */
.hero__button-container--alt { }
.proyecto-card--destacado { }
```

3. Media Queries Organizadas

Las media queries se reorganizaron para mantenerlas cercanas a los componentes que afectan:

```css
/* Media queries específicas por dispositivo */
@media (max-width: 768px) {
    .hero__content {
        /* Ajustes para tabletas */
    }

    .sobre-mi__imagen {
        /* Ajustes para tabletas */
    }
}
```

Diagrama de la Nueva Estructura BEM

```
├── HTML
│   ├── effect__spotlight (elemento dentro del bloque effect)
│   ├── background (bloque)
│   │   └── background__image (elemento del bloque background)
│   ├── header (bloque)
│   │   ├── header__nav (elemento del bloque header)
│   │   └── header__link (elemento del bloque header)
│   ├── hero (bloque)
│   │   ├── hero__robot-container (elemento del bloque hero)
│   │   ├── hero__content (elemento del bloque hero)
│   │   │   ├── hero__title (elemento del bloque hero)
│   │   │   └── hero__card (elemento del bloque hero)
│   │   └── hero__buttons (elemento del bloque hero)
│   │       ├── hero__button-container (elemento del bloque hero)
│   │       └── hero__button-container--alt (modificador del elemento)
│   └── sobre-mi (bloque)
│       ├── sobre-mi__container (elemento del bloque sobre-mi)
│       ├── sobre-mi__titulo (elemento del bloque sobre-mi)
│       └── sobre-mi__imagen (elemento del bloque sobre-mi)
│
└── CSS
    ├── Organizado por componentes/bloques
    │   ├── .page (estilos globales)
    │   ├── .header y sus elementos
    │   ├── .hero y sus elementos
    │   └── .sobre-mi y sus elementos
    └── Media queries específicas por componente
```

Optimización para Dispositivos Móviles

Se implementaron optimizaciones específicas para mejorar la experiencia en dispositivos móviles, con especial atención al iPhone 14 Pro Max.

Problemas Identificados en Dispositivos Móviles

1. **Problemas de tamaño y posicionamiento**: Elementos demasiado grandes o mal posicionados.
2. **Problemas de viewport en Safari**: El 100vh no funciona correctamente en Safari iOS.
3. **Efectos hover persistentes**: En dispositivos táctiles, los efectos hover quedaban "pegados".

Soluciones Implementadas

1. Media Queries Específicas

Se crearon media queries específicas para el tamaño de pantalla del iPhone 14 Pro Max:

```css
@media only screen and (min-device-width: 390px) and (max-device-width: 430px)
       and (min-device-height: 800px) and (max-device-height: 950px)
       and (-webkit-device-pixel-ratio: 3) {
    /* Ajustes específicos */
}
```

2. Corrección del Viewport en Safari

```css
.page {
    min-height: -webkit-fill-available;
}

.hero {
    height: auto;
    min-height: calc(100vh - 60px);
}
```

3. Manejo de Efectos Hover

```css
@media (hover: hover) {
    /* Efectos hover solo para dispositivos que lo soportan */
}

@media (hover: none) {
    /* Usar :active en lugar de :hover para dispositivos táctiles */
}
```

4. Ajustes de Diseño Responsivo

```css
.hero__content {
    margin: 1rem auto;
    text-align: center;
    width: 90%;
}

.hero__buttons {
    flex-direction: column;
    gap: 1.5rem;
}
```

Implementación del Formulario de Contacto

Se diseñó un formulario de contacto siguiendo la metodología BEM y manteniendo la coherencia visual con el resto del sitio, especialmente con la sección hero.

Estructura HTML del Formulario

```html
<section class="contacto" id="contacto">
    <div class="contacto__container">
        <h2 class="contacto__titulo">Contacto</h2>
        <div class="contacto__contenido">
            <form class="contacto__form">
                <div class="contacto__grupo">
                    <input type="text" id="nombre" class="contacto__input" placeholder=" ">
                    <label for="nombre" class="contacto__label">Nombre</label>
                </div>
                <!-- Más campos -->
                <div class="contacto__boton-container">
                    <button type="submit" class="contacto__boton">Enviar mensaje</button>
                </div>
            </form>

            <div class="contacto__info">
                <!-- Información de contacto -->
            </div>
        </div>
    </div>
</section>
```

Características Principales del Formulario

1. **Efecto de label flotante**: Cuando el usuario hace focus en un campo o comienza a escribir, la etiqueta se mueve y cambia de tamaño.
2. **Consistencia visual**: Se utilizaron los mismos gradientes, colores de borde y efectos de hover de la sección hero.
3. **Diseño responsivo**: Adaptado para diferentes tamaños de pantalla.
4. **Accesibilidad**: Etiquetas claramente asociadas con los campos de entrada.

Efectos Visuales

1. **Efecto de brillo en contenedor**: Similar al utilizado en otras secciones.
2. **Botón con gradiente animado**: Coherente con los botones del hero.
3. **Transiciones suaves**: Para mejorar la experiencia de usuario.

Buenas Prácticas y Conceptos Clave

Principios Fundamentales de CSS

1. Especificidad CSS

La especificidad determina qué regla CSS se aplica cuando hay conflictos. BEM ayuda a mantener una especificidad uniforme:

```
css

/* Evita selectores anidados excesivos */
.sobre-mi .titulo { } /* ❌ Aumenta la especificidad inneces
```