TALLERES DE VERANO 2025

SUBFLASH

DISEÑAR WEB COMPONENTS SIN QUE QUIERAN HACERTE VUDÚ

PACO MORENO

TALLERES DE VERANO 2025

SUBFLASH

Francisco Moreno Sánchez-Aguililla

www.fcomoreno.net

Col·legi Oficial Disseny Gràfic* Catalunya

Diseñador de Producto Digital – Colegiado 606
Trabajando desde 1997
14 años como director creativo en **Im3diA comunicación**
4 años como director de diseño en **Marpadal Interactive Media** (Campuseducacion.com)
6 años como director creativo en **BUDA Business Data Software** (net2rent.com)

TALLERES DE VERANO 2025

SUBFLASH

2006 Oviedo

TALLERES DE VERANO 2025

SUBFLASH

2014 Albacete

TALLERES DE VERANO 2025

SUBFLASH

@LaFlores y LaRita

TALLERES DE VERANO 2025
SUBFLASH

¿Qué son **Web Components**?

- Los **Web Components**, o componentes web, son un conjunto de tecnologías web que permiten crear **elementos HTML personalizados, reutilizables y encapsulados**.
- Estos **elementos pueden ser utilizados en cualquier página web y aplicación, sin importar el framework o librería que se utilice**.
- Básicamente, **permiten dividir la interfaz de usuario en componentes modulares y reutilizables**, mejorando la organización, mantenibilidad y reutilización del código.

¿Cómo son los **Web Components**?

- **Custom Elements:** Permiten definir tus propios elementos HTML, como **<mi-componente></mi-componente>,** con su propia funcionalidad y estilo.
- **Shadow DOM:** Proporciona un alcance aislado para los estilos y el marcado de un componente, evitando conflictos con otros elementos de la página.
- **HTML Templates:** Permiten definir fragmentos de código HTML que no se renderizan inicialmente en la página, pero que pueden ser clonados y utilizados por los componentes.
- **Reusables y Modulares:** Facilitan la organización y reutilización de código JavaScript, permitiendo importar y exportar componentes entre diferentes archivos.

TALLERES DE VERANO 2025
SUBFLASH

¿Cómo se personaliza el diseño de los **Web Components**?

- Para personalizar el diseño de los Web Components, se suelen definir **variables CSS** (**Custom Properties**) que pueden ser modificadas desde fuera del objeto para cambiar su apariencia, ya que los componentes están encapsulados.

```
@customElement("b-button")
export class BButton extends LitElement {
  static override styles = css`
    /* Variables used only on this component
    --b-buttom-default-bg-color: #ffffff;
    --b-buttom-default-font-color: #333333;
    --b-button-default-border-color: #c1c1c1;
    --b-button-default-disabled-color: #cacaca;
    --b-buttom-outline-bg-color: transparent;
    --b-buttom-outline-font-color: var(--b-primary-bg-color, #0066ff);
    --b-buttom-outline-border-color: transparent;
    --b-buttom-outline-disabled-color: transparent;
    --b-button-border-radius: 5px;
    */
```

TALLERES DE VERANO 2025
SUBFLASH

¿Qué es un **Sistema de Diseño**?

- Un **Sistema de Diseño** (o Design System en inglés) es un conjunto de **elementos reutilizables** y directrices que se utilizan para crear productos digitales de manera coherente y eficiente. Incluye desde la **identidad visual de la marca hasta patrones de diseño** y código, facilitando la colaboración entre equipos de diseño y desarrollo.

- Un **Sistema de Diseño** es una herramienta esencial para cualquier organización que busque crear productos digitales consistentes, eficientes y escalables.

- Los diseñadores de producto, construimos los sistemas de diseño basándonos en la metodología **Atomic Design**, de **Brad Frost**, se basa en la idea de que **las interfaces de usuario pueden dividirse en componentes** cada vez más complejos, al igual que la materia se compone de átomos, moléculas y organismos.

TALLERES DE VERANO 2025
SUBFLASH

¿Qué es un **Componente de Diseño**?

- Un **componente de diseño** es un **elemento reutilizable en un sistema de diseño,** como un botón, un icono o un campo de formulario, que se puede usar repetidamente en diferentes partes de un diseño, manteniendo la coherencia visual y funcional.

- **El uso de componentes en diseño obedece a la** metodología **atomic design** que facilita la creación, reutilización y mantenimiento de componentes de UI consistentes y escalables.

TALLERES DE VERANO 2025
SUBFLASH

Características de un **Componente**

- **Reutilización:** Los componentes permiten a los diseñadores **evitar recrear el mismo elemento una y otra vez,** lo que ahorra tiempo y esfuerzo, además de garantizar que el diseño sea consistente en todo el proyecto.

- **Consistencia:** Al actualizar un componente maestro, **todas las instancias de ese componente se actualizan automáticamente**, lo que ayuda a mantener la coherencia visual y funcional en todo el sistema de diseño.

- **Modularidad:** Los componentes pueden ser considerados como **bloques de construcción que se pueden combinar y reutilizar** para crear interfaces de usuario más complejas.

¿Cómo se personaliza el diseño de los **Componentes** de un sistema de diseño?

- **Creación de estilos de estado:** Se definen estilos para **diferentes estados** de un componente, como hover, active, disabled, etc. Estos estilos se aplican a los componentes **para crear variaciones** visuales según la interacción del usuario.

- **Definición de variables o Tokens:** Se establecen variables para propiedades como colores, tipografía, espaciado, sombras, etc. Estas variables se utilizan en los componentes, permitiendo cambiar su valor globalmente y afectando a todos los lugares donde se utilizan.

- En un buen sistema diseño-desarrollo, **los Tokens se corresponden con las Custom Properties**

Componente de Diseño

Tokens
(Figma/Penpot...)

Web Component

Custom Properties
(CSS)

Tokens de diseño: el puente entre UI y código

Los **tokens de diseño** son valores de diseño empaquetados en variables. Son la unidad mínima de estilo que define cómo se ve y se siente un producto digital:
colores, tipografías, tamaños, espaciados, radios, sombras...

*En vez de que un botón en Figma tenga **"#FF5722"** y otro **"16px de padding**", lo que tenemos es:*

```
--color-primary: #FF5722;
--spacing-md: 16px;
```

De esta forma, cuando un desarrollador implementa un Web Component, puede usar exactamente esos mismos valores.
Beneficio: si cambia un token, cambia en todo el sistema sin romper la coherencia.

Ejemplos clave de tokens

Color:

```
--color-primary → #FF5722
--color-secondary → #009688
```

Tipografía:

```
--font-family-base → "Inter, sans-serif"
--font-size-sm → 0.875rem
```

Espaciado:

```
--spacing-sm → 8px
--spacing-lg → 32px
```

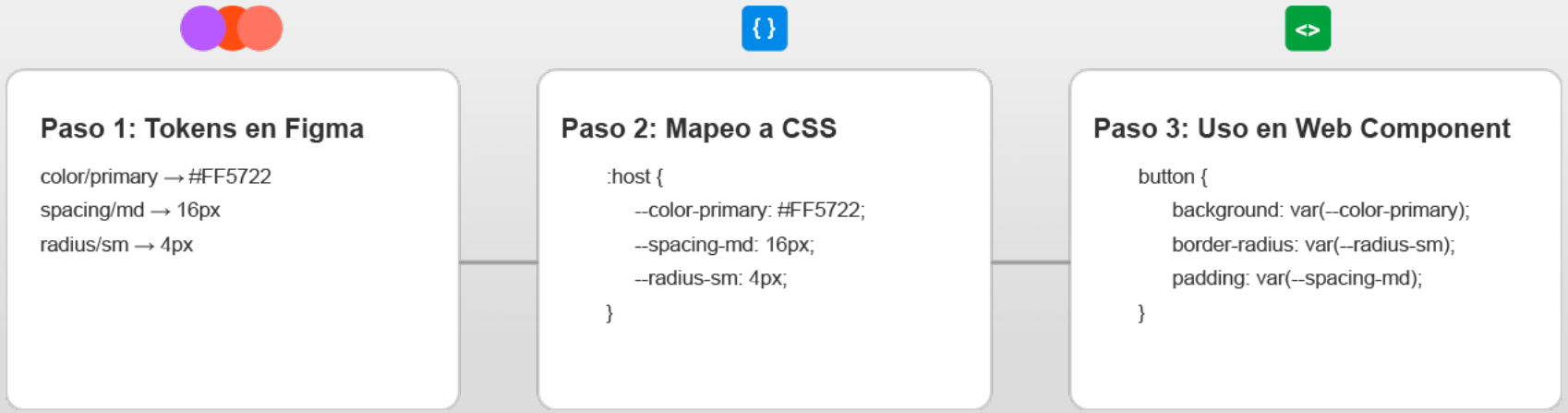
Bordes y radios:

```
--radius-sm → 4px
--radius-pill → 9999px
```

Mapeo Figma ↔ Web Components

En Figma podemos nombrar tokens con una convención clara (por ejemplo, **`color/primary`** o **`spacing/sm`**) usando estilos y variables.

Luego, en desarrollo, se transforman en ***custom properties*** CSS con la misma lógica.



TALLERES DE VERANO 2025

SUBFLASH

Vamos a Figma...

TALLERES DE VERANO 2025
SUBFLASH

Del prototipo a la implementación

1 Preparar el prototipo en Figma

- **Componentes finales**: Asegúrate de que todos los elementos usan tokens en vez de valores fijos.
- **Estados y variaciones**: Define variantes (*hover*, *active*, *disabled*) para que el desarrollador no tenga que imaginar cómo se ven.
- **Estructura lógica**: Nombra capas y componentes con una convención que coincida con la del código.
- **Uso de Auto Layout**: Facilita entender márgenes, paddings y alineaciones.

💡 Tip: Los desarrolladores odian recibir un diseño que solo funciona “a ojo”.
Auto Layout + tokens = cero ambigüedades

Del prototipo a la implementación

2 Documentar dentro del mismo archivo

- Usa descripciones en Figma para cada componente, indicando:
  - Uso recomendado
  - Estados disponibles
  - Qué tokens se usan

- Añade notas directamente en el prototipo con `Shift + N` para detalles específicos.

Del prototipo a la implementación

3 Exportar especificaciones para desarrollo

Opciones recomendadas:

Figma Inspect: El desarrollador puede seleccionar cualquier elemento y ver:

- Tokens aplicados
- Medidas exactas
- Tipografía y colores
- Propiedades CSS listas para copiar

Plugins:

Design Tokens (para exportar JSON con todos los tokens)
Tokens Studio (para sincronizar con repositorios de código)
Anima Figma to Code (para generar HTML/CSS básico como referencia) Veámoslo en Figma

Del prototipo a la implementación

4 Ciclo de retroalimentación

4.1. Diseño → Desarrollo

- El diseñador prepara el prototipo con **tokens** y **variaciones** ya definidos.
- Se entrega junto con la documentación y exportación de tokens (JSON o variables CSS).
- El desarrollador implementa los Web Components usando esas variables como base.

Del prototipo a la implementación

4 Ciclo de retroalimentación

4.2. Desarrollo → Diseño

- El desarrollador detecta:
  - **Problemas técnicos** (p. ej., un radio de 12px que rompe en un layout responsive).
  - **Limitaciones de rendimiento** (un efecto de sombra demasiado pesado).
  - **Casos no contemplados** (un estado de error que no estaba diseñado).

- Informa de vuelta al diseñador, preferiblemente en el mismo archivo de Figma o en un canal acordado (Slack, Jira...).

💡 **Tip:** Usar comentarios en Figma en lugar de mensajes sueltos en chat.
Así la observación queda pegada al elemento.

Del prototipo a la implementación

4 Ciclo de retroalimentación

4.3. Iteración rápida

- El diseñador ajusta el token o la variante en Figma.
- El cambio se sincroniza y llega al código de forma automática si usan una integración (*Tokens Studio, GitHub Actions, etc.*).
- Si no hay integración automática, se vuelve a exportar el JSON/variables y se actualiza en el repositorio.

Del prototipo a la implementación

4 Ciclo de retroalimentación

4.4. Beneficios del bucle

- **Coherencia visual mantenida:** Todos los ajustes pasan por tokens, así que los cambios se aplican globalmente.
- **Menos errores de implementación:** No hay interpretaciones diferentes de un mismo diseño.
- **Velocidad de reacción:** Cambiar un token en Figma puede reflejarse en el producto en minutos.
- **Menos deuda técnica y de diseño:** No se acumulan “apaños” que luego nadie recuerda.

TALLERES DE VERANO 2025

SUBFLASH

¡GRACIAS!