Unidad 4: Incorporando SVG a la web

Atributos imprescindibles en un SVG externo

`<svg xmlns="http://www.w3.org/2000/svg">…</svg>`
Atributo imprescindible para que el archivo SVG cargue como un archivo externo. Si no dispone de este atributo, puede visualizarse como una imagen *rota* (perdida) en el navegador.

`<svg xmlns:xlink="http://www.w3.org/1999/xlink">…</svg>`
Atributo adicional al anterior que deberá incorporarse a la etiqueta **`<svg>`** si este SVG hace referencias a otros elementos a través del atributo **`xlink:href`** (por ejemplo, en la etiqueta **`<use>`** o en la etiqueta **`<image>`**).

Formas de cargar un SVG en la web

`<svg>…</svg>`
SVG dentro del archivo HTML o *inline*. Permite manipulación con CSS y JavaScript, pero no permite el *cacheo* (memorizar el archivo por parte del navegador), por lo que si es un elemento reutilizable (como un icono) se estará descargando en cada página.

`<img src="archivo.svg" alt="…" />`
SVG como imagen externa de contenido. No permite su manipulación por CSS o JS, no tiene cambios de estado o eventos (como **`:hover`**). Método aceptable para imágenes relativamente estáticas, como logos.

`background-image: url(archivo.svg);`
SVG como imagen externa de fondo. Tiene las mismas limitaciones que con la etiqueta **`<img>`**.

`<object type="image/svg+xml" data="logo.svg"></object>`
SVG como imagen externa a través de la etiqueta **`<object>`**. Permite estilos con CSS, eventos e interacción con JavaScript, siempre y cuando tanto la CSS como el JS se incorporen desde el propio archivo SVG.

Se puede incluir una CSS externa en una imagen SVG que se llame desde esta etiqueta, pero para esto se debe poner esta línea al principio del código del archivo, fuera de la etiqueta **`<svg>`**:
`<?xml-stylesheet type="text/css" href="estilos.css"?>`

`<use xlink:href="sprites.svg#icono-menu" />`
Podemos cargar el código SVG a través de un *sprite* (*mapa* o recopilación de diferentes imágenes en un solo archivo) externo. Para hacer esto, crearemos un archivo SVG con los *sprites* definidos cada uno de ellos en un **`<symbol>`** con su **`id`** correspondiente. Para emplearlos, utilizaremos la etiqueta **`<use>`** dentro de una etiqueta **`<svg>`** en nuestro HTML, cuyo **`xlink:href`** haga referencia al nombre (y la ruta) del archivo, y acabe en almohadilla o *hash* **`#`** + el **`id`** del **`<symbol>`** a usar.

Este sistema tiene la ventaja de que el SVG es manipulable por CSS y JS (al ser *inline* la etiqueta **`<svg>`** que lo llama) pero toda la complejidad del dibujado queda en un archivo externo que se puede *cachear* sin problemas.

Este método se soporta bien en todos los navegadores actuales, pero es relativamente reciente; por lo que si nos interesa mejorar el soporte en versiones antiguas de Microsoft Edge, Safari o Internet Explorer, se puede incluir este *polyfill* (archivo JS) para arreglarlo: https://github.com/jonathantneal/svg4everybody

SVG responsive

`<svg viewBox="…"></svg>`
Si omitimos **`width`** y **`height`** de un SVG, tenderá a ocupar todo el espacio disponible, tomando el **`viewBox`** como referencia de proporciones.

`<style>@media screen and (min-width: 400px){…}</style>`
Podemos incorporar *media queries* dentro de la etiqueta **`<style>`** de un SVG que se cargue de forma externa. Estas *media queries* harán referencia al espacio de visualización del propio SVG, en vez de la página que lo llama. Es decir, un SVG externo crea su propio *viewport*, como si fuera un **`<iframe>`**.

`vector-effect="non-scaling-stroke"`
Esta propiedad fuerza a un elemento SVG a mantener el **`stroke`** al mismo grosor, independientemente de que el SVG se aumente o disminuya de tamaño. Si la escribimos como propiedad CSS (en vez de atributo), sería:
`vector-effect: non-scaling-stroke;`

SVG accesible

Un SVG será más fácilmente interpretado por lectores de pantalla si es *inline*, es decir, si está dentro del propio HTML.

Aún así, para SVG relativamente simples como logos o iconos, si se cargan de forma externa, podemos servirnos del atributo alt (si se carga con la etiqueta **`<img>`**). Si se

cargan de forma externa a través de la etiqueta **<use>**, podemos incorporar un **<title>** a la etiqueta **<svg>** que la llama.

Además, un SVG *inline* puede *reforzarse* (asegurarse de que sea correctamente leída por todo tipo de lectores de pantalla) añadiendo a la etiqueta **<svg>** el atributo **role="img"**.

<title>…</title>
Permite dar un título al SVG para que sea identificado rápidamente por lectores de pantalla.

Podemos reforzar el uso de esta etiqueta (mejorando el soporte en los diversos lectores de pantalla existentes) con los *aria roles*.

Si identificamos el **<title>** con un **id**, podemos incorporar a la etiqueta **<svg>** el atributo **aria-labelledby="**idDelTitulo**"** y conseguir mejor compatibilidad.

<desc>…</desc>
Permite dar una descripción más extensa de los contenidos del SVG.

Podríamos reforzar el papel de la etiqueta **<desc>** con el atributo **aria-describedby="idDeLaDesc"** en la etiqueta **<svg>**, pero actualmente está mejor soportado **aria-labelledby**, por lo que podemos combinar los id de **<title>** y **<desc>** en el atributo **aria-labelledby="**idDelTitulo idDeLaDesc**"**.