

Enlaces con estilo

Una buena práctica que afecta positivamente a la accesibilidad y usabilidad de nuestras páginas es la de crear diferencias de estilo mediante la CSS en los enlaces en función del documento al que apunten (procurando siempre mantener la **consistencia que permita identificar los enlaces**). Mediante esta práctica, los usuarios con acceso a los navegadores gráficos podrán reconocer si el documento al que van a acceder es, por ejemplo, un pdf o si el destino se encuentra fuera de nuestro portal.

Problemas

- Tendremos que identificar cada tipo de enlace mediante una clase que permita aplicarle el estilo.
- En muchas ocasiones las personas que introducen los enlaces son los responsables del contenido y no tienen acceso a marcar el código.

Solución

Mediante una sencilla función escrita en javascript, e introducida de forma no intrusiva, podremos identificar el tipo de documento en función del contenido del atributo "title" (si está presente) y/o de la extensión del archivo destino.

nota: es una buena práctica el que los editores/responsables de contenido adopten un protocolo común de identificar el tipo de destino del enlace en el atributo "title" del mismo, pues la mayor parte de las herramientas de edición permiten su introducción.

Adicionalmente, se le ha añadido al script la funcionalidad de incluir un texto en el atributo "title" de aquellos enlaces que no lo posean y que apunten a un documento reconocible por el propio script.

El script

```
// declaramos un array con el fragmento de texto introducido por el
// editor/responsable del contenido y que describe el tipo de destino del enlace.
var tFormat = ["enlace externo", "documento en formato pdf", "documento en formato
word"];

// declaramos una tipología de enlaces
// (declarado en el mismo orden que el anterior)
var Format = ["externo", "pdf", "doc"];

// declaramos las clases para que la hoja de estilos formatee cada tipo de enlace
// (declarado en el mismo orden que los anteriores)
var classFormat = ["externo", "pdf", "doc"];

function enlaces() {
// Seleccionamos todos los enlaces del documento
```

```
var links = document.getElementsByTagName("a");
for (var i = 0; i < links.length; i++) {
    // Comprobamos que no son anclas
    if(links[i].href.indexOf(".") > -1) {
        // Comprobamos si tienen atributo "title"
        if(links[i].getAttribute("title")) {
            // Obtenemos el atributo "title" en minúscula
            var title = links[i].getAttribute("title").toLowerCase();
            // Hacemos un bucle que comprueba si en el "title"
            // se encuentra uno de los textos indicados en el
            // array tFormat. Si lo encuentra, le añade la
            // clase declarada en la misma posición que el texto
            // en el array classFormat
            for(var f = 0; f < tFormat.length; f++)
                if(title.indexOf(tFormat[f]) > -1)
                    links[i].className = classFormat[f] + " " + links[i].className;
        }
        // Dado que no siempre todos los enlaces tienen atributo "title"
        // comprobaremos, por si acaso, el atributo "href" obteniendo
        // la extensión del documento destino y comparándolo con las declaradas
        // en el array Format.
        var inicio = links[i].href.length-4;
        var fin = links[i].href.length;
        var wFormat = links[i].href.substring(inicio,fin).toLowerCase();
        for(var f = 0; f < Format.length; f++)
            if(wFormat.indexOf(Format[f]) > -1) {
                links[i].className = classFormat[f] + " " + links[i].className;
                // Si no tiene el atributo "title" se lo añade
                if(!links[i].getAttribute("title"))
                    links[i].setAttribute("title", tFormat[f])
            }
    }
}
window.onload = enlaces;
```
