Hasta hace unos años, cada idioma utilizaba su propio juego de caracteres, con denominación de estándar ISO. En nuestro ámbito lingüístico empleábamos una versión de caracteres latinos en la que se incluían las tildes, la cedilla, la eñe… Los países escandinavos y bálticos usaban otra versión del juego latino. Los sajones, sin problemas de tildes, podían utilizar prácticamente cualquiera de los juegos de alfabetos occidentales. Y rusos, árabes, hindúes, chinos, japoneses, etcétera, usaban sus propios juegos de caracteres.

El objetivo de cada `charset` era aligerar la carga de las tipografías y acelerar la localización de cada letra para cada idioma.

Cada carácter se corresponde en el `charset` con una identidad alfanumérica, que es la que utilizan los sistemas informáticos para comunicarse. Esta identificación podía existir simultáneamente en varios juegos de caracteres para grafías distintas.

Con la adopción de los `charset` UTF (Unicode Transformation Format) se elimina este problema. El juego es completo y se amplía, además, con pictogramas diversos: flechas, símbolos matemáticos, corchetes y llaves de todo tipo, símbolos de monedas, glifos, etcétera. Cada letra ocupa un espacio y puede escribirse de derecha a izquierda y viceversa, en la versión de 8 bytes (UTF-8). Los idiomas japonés, chino y coreano, en los que cada espacio tipográfico se compone de la superposición de varios caracteres distintos, ocupa 16 bytes (UTF-16).

UTF, que se corresponde con la norma ISO 10646, debería ser el estándar para todas las fórmulas de escritura web y en nuestro ámbito lingüístico deberíamos utilizar únicamente UTF-8. Sin embargo, todavía se usan los ISO anteriores causando, cuando menos, una terrible desazón en el periodista que se tropieza con los problemas de provocan.

Por ejemplo, puedes escribir la palabra “extrañísimo” y cuando buscas el código fuente creado en el gestor de contenidos, te encuentras `extraÃ±Ãsimo`. No se te ocurra corregirlo, porque está bien. Te muestra una traslación directa a/o desde UTF, pero se mostrará correctamente en pantalla.

Si tienes que cambiar o crear un atributo, hazlo sin alterar el texto que hayas redactado para mostrar en pantalla. Cuando hayas termina-

do, regresa a la visión del editor WYSIWYG, para comprobar que el funcionamiento es correcto antes de guardarlo.

También puedes tropezarte con otras manipulaciones de texto que puede realizar el editor de forma automática. Por ejemplo, `extra&ntilde;&iacute;simo`, en formato HTML. Será igualmente correcto.[13]

En nuestra producción habitual podemos encontrarnos con esta extraña forma de convertir letras en formato HTML cuando escribimos cualquier vocal con tilde, la eñe (ñ), la cedilla (ç), el punto intermedio (·), las comillas simples y dobles tipográficas, las letras voladas (ª y º), los grados (°), los símbolos monetarios o de negocio (€, ©, ®), las rayas (—), las comillas angulares (« y »), los interrogantes (¿ y ?) y las exclamaciones (¡ y !), o los puntos suspensivos (…).

Pero también sucede con los corchetes angulares (< y >), que son los que se emplean para distinguir las etiquetas y elementos en HTML y que siempre utilizaremos para etiquetar nuestro contenido dentro del código fuente. De lo contrario, aquello que contuvieran sería interpretado como texto.

Resumiendo, si utilizamos un editor WYSIWYG para escribir y vemos correctamente nuestro texto en el front-end pero se muestra alterado cuando entramos en el código fuente para modificar las etiquetas, no nos preocupemos y realicemos la modificación pertinente sin intentar corregirlo, porque el comportamiento es correcto ya que se debe a un ajuste de `charset`.

Cuando escribamos etiquetas y atributos en el código fuente, no olvidemos que las letras conflictivas, como las citadas anteriormente, están prohibidas.[14] Usemos letras sin tildes y palabras sin espacios ni guiones.

13 Siempre te puede servir de ayuda el conversor online del consorcio que gestiona Internet, W3C. Permite trasladar un texto a distintos juegos de caracteres y formatos escapados: http://people.w3.org/rishida/tools/conversion/

14 Prohibidas con la excepción de los atributos que se pueden mostrar en el front-end. Por ejemplo, el atributo `alt` en las imágenes sirve para mostrar una explicación de la

(pasa a la página siguiente)

La dificultad estriba en que para escribir una cursiva se emplean habitualmente dos elementos distintos (`i` y `em`). Se usan estos porque son los que, por defecto, utilizan los navegadores para mostrar el texto como itálicas. Sin embargo, los editores WYSIWYG sólo tienen un botón para ello. Y lo mismo sucede con las negritas (`b` y `strong`).

Queremos escribir algo en cursiva y el editor sólo nos presenta el botón de la *a tumbada*. Ignoramos cuál de las dos etiquetas `i` o `em` codificará finalmente.

Y la diferencia no sólo es importante. Es fundamental. Porque las etiquetas `i` y `b` no aportan valor semántico al contenido, pero `em` y `strong`, sí. Y mucho.

Como el editor sólo escribe una de estas dos etiquetas alternativas, la solución pasa por acceder al código fuente y editarlo manualmente. La idea es sustituir la `i` por `em` cuando se necesite. Y la `b` por `strong` cuando toque.

Aún así, podría darse la situación de que, una vez hecho el cambio y guardado en la base de datos, el dibujado de la página vuelve a utilizar el elemento que habías modificado. Puede ser que el gestor de contenidos no admita estas etiquetas y la programación del editor las corrija al guardar el texto o que la programación del front-end sustituya de forma automática etiquetas que considera antiguas, como `i` o `b`.

'i' y 'b', ahora sí, ahora no…

Las pruebas realizadas tanto con el WYSIWYG Xinha como con Drupal 7 para escribir este volumen mostraron que el editor siempre utilizaba `em` y `strong` para las cursivas y las negritas, respectivamente (Figura 5). Probablemente la mayoría de los editores y CMS actuales apliquen la misma solución.

Esta opción se debe a que en la versión anterior a HTML5 —XHTML 1.1— las etiquetas `i` y `b` se consideraban obsoletas, prefiriéndose sus alternativas naturales. Si se necesitaba redactar un texto que convencionalmente se exhibía en cursiva pero sin valor semántico, se debía utilizar un truco de estilos CSS:

```
<span style="font-style:italic;">cursiva</span>
```

Como todos los gestores de contenidos y editores visuales proceden de las versiones anteriores de HTML, siguen escribiendo únicamente los elementos énfasis (`em`) e importante (fuerte, `strong`). Pero son cosas distintas y tienen usos diferentes.

HTML5 ha recuperado los elementos despreciados en la versión anterior y ha dotado a la itálica (`i`) de expresividad para mostrar las convenciones visuales en los textos escritos en la mayoría de los idiomas. Otro tanto ha hecho con la negrita (`b`), aunque ésta no tiene ningún valor semántico.

Como periodistas que elaboramos textos exquisitamente elaborados, necesitamos recuperar `i` y `b` y toda su expresividad. Pero se hace muy difícil.

Si no logras corregir los elementos que escribe tu editor WYSIWYG o muestra tu CMS, estarás transmitiendo permanentemente mensajes semánticos que no se corresponderán a lo que deseas decir. Será un pecado de exceso.

Con todo, mi recomendación es que te fijes bien en lo que quieres decir. Especialmente, respecto al uso de cursivas. Algunos tipos de contenido deben etiquetarse con otros elementos HTML5 que en pantalla producen itálicas.

Las cursivas y todos sus elementos

Creo que el mejor punto de partida para aprender a utilizar las etiquetas correspondientes a las cursivas es olvidarse del concepto que se transmite y pensar únicamente en que convencionalmente aquello que marcaremos se muestra siempre en cursiva.

Una vez hemos definido que algo se marcará, será más fácil determinar qué tipo de etiqueta le corresponde.

Empezaremos por el elemento itálica (`i`). Cuando se escribe se está indicando que aquella palabra o expresión, en el idioma en el que está escrita y en el contexto del contenido, se muestra siempre en versión

documento —si se ha cambiado el archivo web— y el elemento `time` de la página.

Por cierto, si el CMS que usas no genera automáticamente las fechas de creación y modificación, y/o no las puedes utilizar para crear las etiquetas `time` aunque sea manualmente, tírale de las orejas a tu programador para que lo solucione de inmediato. Crear contenidos para una página que no se indexa adecuadamente es tirar el dinero.

El porqué y el cuándo de la datación

Nosotros lo entendemos, porque somos periodistas. Pero el resto de los mortales, especialmente directivos de empresa, de marketing y un larguísimo etcétera de gente que tiene la mala costumbre de mandar en sitios web, no. Y aunque lo expliques una y mil veces, no te harán caso.

Pero hay que insistir: una noticia caduca en sí misma. Una noticia corporativa caduca en sí misma. Una historia en un medio informativo caduca en sí misma.

La noticia, por sí misma, es una radiografía que captura la instantánea de un momento determinado y la plasma junto a la fecha. Ergo, la situación descrita al día siguiente puede haberse modificado sustancialmente y nadie puede reclamar por ello. Si en una noticia anuncias que "desde hoy y hasta el martes próximo se realizará una captación de personal", su data pone límite al reclutamiento. Y nadie debería reclamar expirado el proceso.

Los directivos a los que aludía creen que cuando un acontecimiento anunciado ha pasado hay que eliminar el contenido. Consideran que historias viejas no interesan, pueden perjudicar o proporcionar una imagen descuidada del sitio web, si no se actualizan.

Pues que contraten a más periodistas para mantener actualizado el apartado de noticias. Pero las noticias no se borran. Nunca.

Un argumento más: las noticias antiguas están posicionadas en los buscadores. Cuanto más viejas, mejor posición. Es ley de vida. Y eliminarlas podría perjudicar a todo el sitio web.

Insiste como yo insisto, aunque pierdas.

Por lo tanto, el primer criterio para datar contenido es el que hace referencia a las noticias. Repasemos:

- Noticias y entradas de blog
  La fecha fija la instantánea de un momento histórico, aporta caducidad automática a las historias, exime de responsabilidad expirado el límite del acontecimiento narrado y facilita la ordenación cronológica del bagaje de la organización. Es indispensable.
- Ofertas y precios especiales
  Cualquier tipo de oferta comercial que suponga una alteración temporal del precio habitual de las cosas debe tener una fecha límite. Incluso si se trata de una liquidación, en la que la oferta desaparece en el momento en el que se agotan los productos. Siempre hay que indicar la fecha de inicio y la de final para evitar que alguien, dentro de unos años, reclame el producto al importe del chollo caducado. No está de más echarle un vistazo a la legislación pertinente en materia de ofertas y *outlet,* que puede fijar restricciones tanto de fechas como de procedimientos de anuncio de la ganga, por ejemplo en Catalunya.
- Algunos avisos legales
  En mercados regulados (sanitario, financiero). Las condiciones de contratación de las tiendas on line, las de reclamación, los acuerdos temporales con proveedores que proporcionen una pátina de calidad al sitio web… El usuario debe saber cuándo empiezan y cuándo acaban las condiciones pactadas.
- Proyectos temporales
  Uniones temporales de empresas, asociación de organizaciones para el desarrollo de proyectos, declaraciones de intenciones, manifiestos, adhesión corporativa a movimientos socioeconómicos externos… Vale la pena saber cuándo se produce la unión. Si se rompe, basta una actualización o una noticia que lo deje claro.
- Páginas de usabilidad susceptibles de caer en el olvido
  Por ejemplo, los FAQ enormes. Un sistema de preguntas frecuentes es más eficiente si hay más preguntas distintas que dan acceso a

```
▶
            <mn>4</mn>
          </mrow>
            <mo>=</mo>
            <mn>0</mn>
        </mrow>
      </math>
    </figure>
    <p>Y eso es todo.</p>
</article>
```

En el ejemplo anterior, dentro de `article` se ha escrito un título y un par de párrafos que se ilustran con una fórmula matemática. Dentro de `figure` se crea un elemento `math` que declara, con una llamada a la URL, el modelo de código que debe mostrar y un `figcaption` (elemento de titular del `figure`). El elemento `math` crea factores de cálculo (`mrow`), cifras (`mn`), operadores (`mo`), incógnitas (`mi`) y construye potencias (`msup`). Todos los elementos se tendrán que escribir directamente en el modo código fuente del editor WYSIWYG.

Para el ejemplo anterior, el navegador muestra "$x^2 + 4x + 4 = 0$".

Se puede complicar más con raíces cuadradas, divisores con distintos niveles, símbolos logarítmicos, estadísticos… de todo.

Cálculos sencillitos

Los navegadores más modernos que interpretan HTML5 adecuadamente son capaces de realizar cálculos no demasiados complejos que el usuario ve en pantalla y cuyos resultados no residen en el servidor ni se guardan en caché. En otras palabras, el resultado sólo lo ve el usuario que emplea el formulario y le sirve en ese momento. No lo puede guardar.

Los cálculos se efectúan mediante la API de JavaScript propia de HTML5.

¿Para qué nos sirve? Para crear pequeños *gadgets* que resulten útiles para los visitantes de la página web: conversores de todo tipo, cálculos de relaciones entre entidades, etcétera.

Por ejemplo escribiremos una calculadora del Índice de Masa Corporal (IMC) y la acompañaremos de una lista con una valoración de los resultados obtenidos. La explicación del cálculo es sencilla: es la división del peso de la persona entre el cuadrado de su estatura.

La escritura de la calculadora la haremos dentro de un `figure` y utilizando el modo código fuente en el editor WYSIWYG. Dentro del `figure` crearemos un formulario que contiene la programación JavaScript, los campos de entrada de los valores del usuario y un campo de salida:

```
<figure>
  <form onsubmit="return false"
oninput="imc.value =p.value / (a.value * a.value)">
    <legend>Calculadora del Índice de Masa Corporal
</legend>
    <input name="p" type="number" placeholder="Peso
(kg)" title="Peso (kg)">
    <input name="a" type="number" placeholder="Altura
(m)" title="Altura (m)">
    <output name="imc"></output>
  </form>
  <dl>
    <dt>Infrapeso</dt>
    <dd>Menos de 16</dd>
    <dt>Delgadez</dt>
    <dd>De 16 a 18,5</dd>
    <dt>Normal</dt>
    <dd>De 18,5 a 25</dd>
    <dt>Sobrepeso</dt>
    <dd>De 25 a 30</dd>
    <dt>Obeso</dt>
    <dd>Mayor de 30</dd>
  </dl>
  <figcaption>Traslade su IMC a la tabla inferior para
ver su estado nutricional. Los valores de la tabla se
aplican a adultos de 20 a 60 años.</figcaption>
</figure>
```

Tu programador debe haber fijado con JavaScript el nombre del atributo `mediagroup`, para que lo puedas utilizar. En el ejemplo anterior `nombre_del_grupo_audiovisual` será establecido mediante JavaScript con un par de líneas de programación en el encabezamiento de la página.

Una vez fijado el valor, se puede usar identificándolo en el atributo `mediagroup`, que convierte en un todo cualquier elemento de vídeo o audio que lo tengan en común. De manera que cuando el usuario active el botón de reproducción del vídeo, automáticamente se reproducirá el del audio, sustituyendo al sonido original que ocultaremos con `muted`.

Nuestro trabajo quedará más elegante y no se oirá al cámara dar instrucciones: "Más a la derecha, ¡sonreíd!".

El estado de los atributos

En el momento de redactar este capítulo, todos los atributos que he citado forman parte del estándar tecnológico y está prevista su implementación en todos los dispositivos y navegadores HTML5. Aún así, el atributo `preload` no existe en Internet Explorer.

`poster`, que puede ayudar a diseñar el espacio en página para el vídeo, muestra la imagen escogida, pero no siempre traslada sus dimensiones al objeto vídeo.

Las particularidades de uso de los dispositivos móviles, por pura lógica, eliminan la interpretación de los atributos `autoplay` (que arruinaría a los usuarios de bonos de conexión rápida) y `controls`, que se muestran siempre, aunque algunas versiones de Android pueden ser una excepción.

Los vídeos se muestran a toda pantalla activando el botón de ampliación obtenido mediante el atributo `controls`. Pero también con dos excepciones: Internet Explorer y Opera no enseñan el botón.

Estos dos navegadores tampoco implementan completamente el elemento `track`, que funciona a medias en el resto, cuando existe. Generalmente requiere el refuerzo de JavaScript para comportarse correctamente, lo que limita notablemente su accesibilidad.

La semántica de los elementos video y audio

Todo aquello que se muestra dentro de las etiquetas de los elementos `video` o `audio` forma un todo. El conjunto es una pieza audiovisual que transmite una única variable semántica. Pero no se acompaña de un valor informativo para la variable, de modo que no facilita demasiado las cosas.

Ninguna máquina será capaz de localizar el vídeo o el fragmento de sonido con los únicos datos del paquete de elementos y atributos audiovisuales.

Un primer paso para facilitar su identificación es incorporar el atributo `title` en el elemento, describiendo a modo de titular su contenido.

Pero eso no es suficiente. Piensa que un minuto de discurso en audio o vídeo es poco más de medio folio a doble espacio. Y multiplica por los minutos de tu pieza.

Es mucho mejor y más necesaria una extensa descripción de qué incluye el vídeo. Y si se trata de una entrevista, la transcripción literal del diálogo. Deberían ser piezas informativas en código HTML estándar en la misma página.

La pieza informativa principal es el vídeo o el fragmento de audio. Y la transcripción o explicación de detalles, circunstancias y principales conclusiones es la pieza secundaria. No al revés.

De este modo el contenido adquirirá valor semántico para las máquinas.

Además, aunque es lo más infrecuente, todos los vídeos deberían incluir un `track` de tipo `descriptions` o subtítulos que faciliten la accesibilidad de la pieza informativa. Es de esperar que en un futuro no muy lejano estos archivos de `track` sean legibles por las máquinas y se evite tener que escribir un texto complementario. Pero de momento no es así.

La ubicación de los vídeos y los fragmentos de audio en la página web

Los elementos `video` o `audio`, por definición, forman parte de la ilustración del contenido principal: un título y un cuerpo de texto. Y

```
▶
    <a
      href="http://maps.google.es/maps?f=q&amp;source
=embed&amp;hl=es&amp;geocode=&amp;q=UAB+-+VILA+UNIVERS
IT%C3%80RIA,+Cerdanyola+del+Vall%C3%A8s&amp;aq=1&amp;
oq=uab&amp;sll=41.692248,1.745868&amp;sspn=3.408404,3.
565063&amp;ie=UTF8&amp;hq=UAB+-+VILA+UNIVERSIT%C3%80R
IA,&amp;hnear=Cerdanyola+del+Vall%C3%A8s,+Barcelona,+
Catalu%C3%B1a&amp;t=m&amp;ll=41.494922,2.119257&amp;s
pn=0.011713,0.049003"
      title="Mapa con la ubicación dela Vila
Universitària de la UAB"
      target="_blank"
      rel="external"
      itemprop="name">
        Vila Universitària de la UAB
      </a>
      <span
        style="display:none;"
        itemprop="geo"
        itemscope
        itemtype="http://schema.org/GeoCoordinates">
          Latitud: 41° 29' 42" N. Longitud: 2° 7' 9" E
          <meta itemprop="latitude"
content="41.494922">
          <meta itemprop="longitude"
content="2.119257">
      </span>
  </figcaption>
</figure>
```

No te alteres, que lo repasamos paso a paso.

Para facilitar la comprensión del código, los atributos que he retocado más están separados en líneas tabuladas.

Primero convertimos el `figure` en el contenedor del snippet, utilizando el formato *Place* (lugar). Pegamos a continuación el elemento `iframe` —únicamente— que nos ha entregado Google Maps.

Fíjate que he retirado los atributos que no se corresponden con HTML5. Más adelante ya verás cuáles son.

Después creamos un `figcaption` en el que escribiremos el nombre del lugar dentro de un enlace: "Vila Universitària de la UAB". Copiamos la URL de la nota al margen (`small`) y la usamos como atributo `href` del enlace. Escribimos un atributo `title` y seleccionamos un `target` de apertura en nueva página usando el atributo `rel` para contenido externo.

Todo esto es para que se abra en una página nueva, manteniendo la nuestra abierta. Si usas el código original de Google, cualquier usuario que activara el enlace perdería tu página y pasaría directamente a Google Maps.

En el enlace, finalmente, usamos el atributo `itemprop="name"`, para definir el contenido como el nombre del lugar.

Nos faltan las coordenadas geográficas para completar el snippet enriquecido correspondiente a la ubicación del mapa. Como escribir la latitud y la longitud junto a la toponimia de la villa universitaria resultaría extraño, escribimos dentro del enlace un elemento `span` que, mediante un atributo de estilo, evitaremos que se muestre en pantalla.

Además del atributo de estilo escribimos el atributo `itemprop` correspondiente al `itemtype` lugar con valor `geo`. Iniciamos una nueva estructura de datos con el elemento *booleano* `itemscope` y usamos un nuevo `itemtype` para las coordenadas geográficas.

Dentro del `span` escribimos el texto *legible* por humanos, aunque podríamos prescindir de él y nos bastaría escribir el código `meta`. La latitud y la longitud la escribimos en su estándar sexagesimal,[59] utilizando los datos del enlace de Google convertido con cualquier herramienta geográfica que encontremos por Internet (por ejemplo, [Andrew]). Un navegador GPS de calidad o Google Earth pueden servirnos.

59 Es decir, los valores decimales separados entre comas que aparecían en una de las variables de la URL.

```
  </ul>
  <p>Tiempo de preparación: <time itemprop="prepTime"
datetime="PT15M">15 minutos</time></p>
  <p itemprop="recipeInstructions">Se pica fino, sin
triturar, la cebolla troceada, el pimiento, la zanahoria
y el ajo. Se sala y se sofríe con aceite en una sartén.
<br>
Entretanto, se limpian los champiñones y se vacían
cuidadosamente con una cuchara tras retirar la tija.
Los champiñones vacíos se sumergen en un bol con agua y
limón. Los restos limpios de los champiñones se pasan
por la picadora.<br>
Con el sofrito pochado es el momento de añadir el tomate
rayado y los fragmentos de champiñón picados. Se añade
pimienta y orégano. Cuando esté cocinado, se retira del
fuego y se le añade el atún seco, mezclándolo bien.<br>
En una sartén con una cucharada de aceite se doran los
champiñones vacíos. Después, se pasan a una fuente de
horno y se rellenan cuidadosamente con la mezcla.
Posteriormente se cubren de queso rallado y perejil,
para concluir gratinándolos en el horno.</p>
</figure>
```

Acontecimiento programado en la agenda

```
<figure itemprop="event"
itemscope itemtype="http://schema.org/Event">
  <figcaption itemprop="name">The Rolling Stones en
Boston</figcaption>
  <p itemprop="location" itemscope
itemtype="http://schema.org/PostalAddress">TD Garden</p>
  <p><span itemprop="streetAddress">100 Legends Way
</span><br>
<span itemprop="addressLocality">Boston</span>, <span
itemprop="addressRegion">MA</span> <span
```

```
▶
itemprop="postalCode">02114</span></p>
  <p><time itemprop="startDate" datetime="2013-06-
12T20:00">12 de Junio de 2013, a las 20 horas</time><br>
<a href="ticketmaster.com/foofighters/may20-2011"
itemprop="offers">Entradas a la venta</a></p>
</figure>
```

Lugar sin geolocalización ni tarjeta de visita

```
<figure itemscope itemtype="http://schema.org/Place">
  <figcaption itemprop="name">Vila Universitària
</figcaption>
  <p itemprop="address" itemscope
itemtype="http://schema.org/PostalAddress">
  <span itemprop="streetAddress">Casa de Vila de Puig
</span><br>
  <span itemprop="addressLocality">08193</span> - <span
itemprop="addressLocality">Cerdanyola del Vallès</span>,
<span itemprop="addressRegion">Barcelona</span>
  </p>
</figure>
```

Medicamento

```
<figure itemscope itemtype="http://schema.org/Drug">
  <figcaption itemprop="name">Frenadol (comprimidos
efervescentes)</figcaption>
  <p>Fabricado por <span itemprop="manufacturer"
itemscope
itemtype="http://schema.org/Organization"><span
itemprop="name">Johnson & Johnson, SA</span></span>.<br>
Clasificado como <span itemprop="drugClass" itemscope
itemtype="http://schema.org/DrugClass"><span
                                                    ▶
```