Editorial de la
Universidad Tecnológica Nacional

[Tutorial]

Como incluir código HTML por medio de JAVASCRIPT

edUTecNe

Editorial de la Universidad Tecnológica Nacional – edUTecNe
http://www.edutecne.utn.edu.ar
edutecne@utn.edu.ar

[Tutorial]
Conocimientos previos requeridos:
desarrollo de páginas web con HTML; elementos de JavaScript.

Incluir código HTML por medio de JAVASCRIPT

Introducción

Una página web no es otra cosa que un archivo digital que contiene una serie de instrucciones interpretables por un programa (software) de navegación (browser).

Este programa – inserto en un artefacto electrónico (PC, Tablet, Smartphone) – finalmente genera en una pantalla textos, imágenes, fórmulas (y eventualmente sonidos en reproductores de audio).
Las instrucciones de ese archivo son un conjunto de códigos que siguen diversas reglas de sintaxis (se habla de "lenguajes") que se han ido desarrollando y normalizando a lo largo de los años.

El lenguaje que sirve de estructura básica para las páginas web fue y sigue siendo el conocido como **HTML**. No obstante, la experiencia demostró que era posible enriquecerlo insertando otras instrucciones no previstas por el HTML y que produjeran efectos innovadores, por ejemplo teclas en pantalla que con un clic del cursor hicieran aparecer ventanas o cambiaran el color de una parte del texto (JavaScript) o viñetas (banners) con imágenes que van cambiando en el tiempo (Flash).

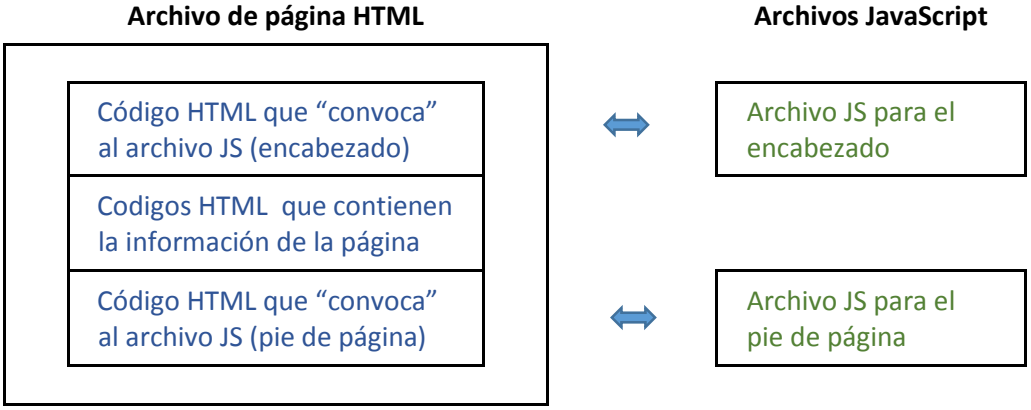
El lenguaje JavaScript evolucionó aportando un amplio espectro de posibilidades aceptadas prácticamente en forma universal. En lo que sigue solo nos detendremos en una de las más simples pero muy útil *cuando se trata de portales que incluyen varias páginas relacionadas*: la capacidad de incluir en un archivo externo un grupo de instrucciones entendibles por los browsers como si fueran códigos HTML.

Ejemplo: *se pretende que todas las páginas finalicen con un "pie de página" con datos de la empresa dueña del portal (su logo, nombre, dirección, forma de contactarla).*
Todos estos datos se incluyen en un archivo JavaScript, por ejemplo ***pie_de_pagina.js*** *(la extensión* ***.js*** *es la característica de un archivo JavaScript). Obviamente los datos se codifican de acuerdo a las reglas del lenguaje JS.*
Cada página HTML "convoca" al archivo externo JS como se haría con cualquier hipervínculo.
El browser lo "lee" como si fuera parte de las instrucciones HTML y hace aparecer su contenido en pantalla.

Ventaja: *Si la empresa cambia alguno de sus datos (por ejemplo la dirección) basta modificarlo en el archivo* ***pie_de_página.js*** *y todas la páginas quedan actualizadas.*

Se reitera: este procedimiento es muy práctico para textos/imágenes (y otros objetos) que se repiten en varias páginas. Los archivos JS externos pueden ser varios (por ejemplo un archivo JS para el encabezado y otro archivo JS distinto para el pie de página). El contenido de estos archivos puede ser tan sofisticado como lo permita el lenguaje HTML. Cada archivo JS puede ser "convocado" en cualquier lugar de la página de acuerdo a la necesidad de su diseño.

El siguiente cuadro muestra el esquema de una de las páginas HTML y los archivos JS asociados.



Detalles del diseño de páginas HTML asociadas a archivos JS

Ejemplo de un archivo JS con la información que iría al pie de cada una de las páginas:

1.- se define un texto (puede incluir imágenes) que aparecerá al final de todas las páginas (pie de página único).
2.- se codifica en HTML
3.- se lo transforma en código JavaScript [incorporado a un archivo que – por ejemplo – se llame "pie_de_pagina.js"]
4.- se "convoca" al archivo .js en cada una de las distintas páginas en la posición que corresponda (en este caso al final) en una forma parecida al link de hipertexto.

Tal como se vio previamente, la ventaja de este procedimiento es que cuando se hagan cambios (en este caso en el pie de página) se retocará solo el archivo "pie_de_pagina.js" y automáticamente se modificarán todas las páginas que lo "convocan".

Caso Nº 1 *(muy elemental)*

En varias páginas web se quiere hacer aparecer el siguiente texto como nota al pie:

[Copyright]: Las obras publicadas en este sitio son de libre acceso para fines académicos.

El código HTML que corresponde será el siguiente:

```
<p>[Copyright]: Las obras publicadas en este sitio son de libre acceso para fines académicos.</p>
```

El código JavaScript que se incluirá en el archivo que llamaremos pie_de_pagina.js será:

```
document.write("<p>");
document.write("[Copyright]:Las obras publicadas en este sitio son de libre acceso para fines académicos.");
document.write("<\/p>");
```

Interpretación del código JavaScript:

document.write : este comando le indica al navegador que lo siguiente [por ejemplo <p>] debe interpretarlo como si estuviera escrito en formato HTML.

Para evitar confusiones, el contenido se incluye entre paréntesis y se termina con ;

("….");

Cuando HTML aplica el símbolo / , en JS se agrega antes el símbolo \

("<\/p>");

Link ("convocatoria") al archivo JavaScript en la página web:

En cada una de las páginas web se incluye este comando HTML que "convoca" al archivo JavaScript, parecido a lo que se haría en un caso de hipertexto:

```
<script src="pie_de_pagina.js" type="text/javascript"></script>
```

Nota importante: en el ejemplo citado se da por supuesto que tanto estas páginas web como el archivo JavaScript se encuentran en el mismo URL; esto no es obligatorio ya que el archivo JavaScript podría estar en cualquier otra dirección web (por ejemplo en una subcarpeta):

/copyright/pie_de_pagina.js).

En este caso, la "convocatoria" debe tener esto en cuenta:

```
<script src="./copyright/pie_de_pagina.js" type="text/javascript"></script>
```

El símbolo ./ simplemente indica que la subcarpeta "copyright" está inmediatamente después del URL en el que se encuentran las páginas web.

Nota: cuando las distintas páginas que "convocan" al archivo JS se encuentran en varias carpetas, conviene definir el link utilizando toda la identificación del archivo JavaScript:

```
<script src="http://[URL del sitio]/copyright/pie_de_pagina.js" type="text/javascript"></script>
```

Caso Nº 2: no se tocan las páginas web *pero se modifica el archivo JavaScript*

El diseñador del portal considera que el texto incluido en el primer ejemplo es demasiado simple y decide modificarlo por el siguiente:

©[Copyright]
Las obras publicadas en este sitio son de libre acceso para fines académicos y como un medio de difundir la producción cultural y el conocimiento generados por autores universitarios o auspiciados por las universidades, pero que estos y la Editorial se reservan el derecho de autoría a todos los fines que correspondan.

El código HTML será el siguiente:

```
<hr style="margin: 30pt 0pt 0pt;">
<p style="margin: 10pt 100pt 0pt; text-align: justify; line-height: normal;">
<i>
©[Copyright]
</i></p>
<p style="margin: 0pt 100pt 10pt; text-align: justify; line-height: normal;">
<i>
Las obras publicadas en este sitio son de libre acceso para fines académicos y como un medio de difundir la producción cultural y el conocimiento generados por autores universitarios o auspiciados por las universidades, pero que estos y la Editorial se reservan el derecho de autoría a todos los fines que correspondan.
</i></p>
<hr>
```

El nuevo código JavaScript resultante será:

```
document.write("<hr style=\"margin: 30pt 0pt 0pt;\">");
document.write("<p style=\"margin: 10pt 100pt 0pt; text-align: justify; line-height: normal;\">");
document.write("<i>");
document.write("©[Copyright]");
document.write("<\/i><\/p>");
document.write("<p style=\"margin: 0pt 100pt 10pt; text-align: justify; line-height: normal;\">");
document.write("<i>");
document.write("Las obras publicadas en este sitio son de libre acceso para fines académicos y como un medio de difundir la");
document.write("producción cultural y el conocimiento generados por autores universitarios o auspiciados por las universidades, pero que estos y la Editorial se reservan el derecho de autoría a todos los fines que correspondan.");
document.write("<\/i><\/p>");
document.write("<hr>");
```

Este código nuevo ahora se incluirá en el archivo **pie_de_pagina.js** (el mismo nombre).

Las páginas web no necesitarían modificarse porque ya estaban "convocando" a este archivo, que ahora contiene nueva información.

En otras palabras: solo modificando el archivo JavaScript se modifica la presentación de todas las páginas que lo "convocan" (nuevo "pie de página").

Como se construye un archivo JavaScript a partir de sus códigos

En el ejemplo elemental de la primera página los códigos JavaScript eran los siguientes:

```
document.write("<p>");
document.write("[Copyright]:Las obras publicadas en este sitio son de libre acceso para fines académicos.");
document.write("<\/p>");
```

Para construir el archivo *pie_de_pagina.js* se crea un documento con formato TXT (texto puro) que contenga los códigos JavaScript. Esto se hace con cualquier procesador de textos (por ejemplo el Bloc de Notas de Windows) y se "Guarda como" con la extensión [.js] en lugar de [.txt] .

Una alternativa es guardarlo directamente con la extensión [.txt] y posteriormente modificar la extensión por [.js]. En este caso el sistema suele alertar que el archivo podría deteriorarse: no hay que hacerle caso.

Como se pasa de HTML a JS

Si bien esto se puede hacer manualmente respetando la sintaxis de JavaScript (como si fuera una traducción), existen programas (conversores) en línea que facilitan la tarea. Basta escribir los códigos HTML (o pegarlos) y el programa se encarga de hacer la traducción a JavaScript.
Un ejemplo de este tipo de herramienta se encuentra en:

http://www.accessify.com/tools-and-wizards/developer-tools/html-javascript-convertor/
opción: *Output as series of document.write statements*

Comparación de HTML y su conversión a JavaScript

La conversión en línea vuelve casi innecesaria la traducción manual. No obstante y a fin de tener una idea más completa de las características de JavaScrip se presentan algunos ejemplos de la sintaxis de *document.write*.

HTML

```
</i></p>
```

JS

```
document.write("<\/i><\/p>");
```

Interpretación:

1.- Como ya se indicó previamente, la información de document.write se ubica entre paréntesis y comillas [("……")], finalizando cada "párrafo" con [;].

2.- el símbolo HTML [/] se traduce en JS como [\/] , por ejemplo [<\/i><\/p>]

HTML

```
<table style="width: 90%;" border="0" cellpadding="2" cellspacing="0">
```

JS

```
document.write("<table style=\"width: 90%;\" border=\"0\" cellpadding=\"2\" cellspacing=\"0\">");
```

Interpretación:

1.- después del signo igual (por ejemplo en HTML border="0"] los parámetros en JavaScript se inician y se cierran con el signo [\ ] , en este último caso antes de las comillas finales [border=\"0\"].

HTML

```
<img style="width: 150px; height: 95px;" alt="logo"
src="http://www.editorial.edu.ar/img/logo_con_marco.jpg">
```

JS

```
document.write("<img style=\"width: 150px; height: 95px;\" alt=\"logo\" ");
document.write("src=\"http:\/\/www.editorial.edu.ar\/img\/logo_con_marco.jpg\">");
```

Interpretación:

1.- el comando HTML que se inicia y termina con <…> está escrito en dos líneas, lo que es legal en este lenguaje.

2.- JavaScript lo interpreta como si fueran dos líneas diferentes:

document.write("……línea 01 …….");

document.write("……línea 02 …….");

3.- Pero el navegador lo traduce como un único comando HTML que empieza con < y termina con > .

Ejemplo concreto de un portal que utiliza este procedimiento para el encabezamiento y el pie de sus páginas:

http://www.edutecne.utn.edu.ar/inicio.html

http://www.edutecne.utn.edu.ar/indices/menu_publicaciones_papel.html

http://www.edutecne.utn.edu.ar/indices/menu_publicaciones_digitales.html

http://www.edutecne.utn.edu.ar/indices/monografias.html

http://www.edutecne.utn.edu.ar/indices/tutoriales.html

[continúan otras páginas]

BIBLIOGRAFIA

Para conocer en detalle el lenguaje JS conviene recurrir al sitio de la **w3schools**, uno de los mejores que se encuentran en la red.

http://www.w3schools.com/js/

En particular sobre el tema de este tutorial (acciones que JS puede producir sobre páginas HTML), se encuentra información detallada en:

http://www.w3schools.com/js/js_htmldom_document.asp