APRENDERAPROGRAMAR.COM

COMENTARIOS CONDICIONALES CSS. PROBLEMAS DE COMPATIBILIDAD ENTRE NAVEGADORES. STRICT MODE. FILTROS (CU01055D)

Sección: Cursos
Categoría: Tutorial básico del programador web: CSS desde cero

Fecha revisión: 2029

Resumen: Entrega nº55 del Tutorial básico: “CSS desde cero”.

Autor: César Krall

COMENTARIOS CONDICIONALES CSS

Con el paso de los años y la evolución de los lenguajes y tecnologías empleados en desarrollos web hay un factor que ha permanecido como un problema: las diferentes formas de interpretar o mostrar el código de los distintos navegadores. Una técnica creada por Microsoft basada en condicionales se ha venido usando para tratar de corregir problemas específicos de ciertos navegadores.

Las personas que trabajan en desarrollos web están acostumbrados a tener que lidiar con este tipo de problemas. No todos los navegadores responden igual. Algunos navegadores (al menos en sus versiones más modernas) cumplen o tratan de cumplir en todo lo posible el estándar definido por la W3C, organización encargada de definir el estándar CSS. Otros navegadores cumplen la mayor parte del estándar pero no responden bien con algunas propiedades o características. Además algunos navegadores incorporan características particulares propias, que no se encuentran dentro del estándar.

Cuando se está realizando un desarrollo web, es frecuente realizar pruebas en distintos navegadores y en distintas versiones. No se pueden probar todos los navegadores y todas las versiones porque sería imposible, pero los desarrolladores web suelen tener un conjunto de navegadores y versiones para test. Uno de los navegadores que históricamente ha venido dando más problemas es el Internet Explorer de Microsoft, aunque hay que decir que en las últimas versiones se han realizado mejoras y cada vez tiende a cumplir mejor los estándares.

Para permitir a los programadores y diseñadores solucionar los problemas o diferencias específicas de Internet Explorer, Microsoft creó un mecanismo de “filtro” que permite definir reglas específicas para navegadores concretos.

En primer lugar veamos en qué puede consistir un problema de distintas interpretación entre navegadores. Para ello hemos escrito este código HTML y el CSS en un archivo con el nombre adecuado y hemos realizado un test para comprobar la visualización en dos navegadores distintos:

```
<html>
<head>
<title>Portal web - aprenderaprogramar.com</title> <meta charset="utf-8">
<link rel="stylesheet" type="text/css" href="CU01055D_A.css">

</head>
<body>
<div id="txtCabecera"><h1>aprenderaprogramar.com</h1></div>
</body>
</html>
```

```
/* Curso CSS estilos aprenderaprogramar.com*/
*{font-family: arial;}
#txtCabecera {width: 500px; margin: 0 auto; padding: 10px; border: solid 2px; text-align: center;
text-shadow: 2px 2px 12px red, -6px -6px 12px blue ;}
```

Esta prueba en concreto la hemos hecho con el navegador Firefox (versión actual) y con un navegador Internet Explorer de hace unos 12 años (esta es una versión antigua, pero aquí no queremos entrar a discutir sobre las versiones o sobre los navegadores, simplemente queremos poner un ejemplo de cómo con el mismo código se pueden obtener visualizaciones distintas y entender que esto puede ser un problema para los programadores y diseñadores web. Podríamos tener problemas de distintas visualizaciones también entre navegadores modernos). Al comprobar el resultado de visualización nos hemos encontrado con estas diferencias principales:

a) Un navegador muestra el efecto text-shadow mientras que el otro no.
b) Un navegador alinea la caja en el centro de la pantalla mientras que el otro lo alinea a la izquierda de la pantalla.
c) Un navegador muestra un mayor espacio superior de relleno sobre el texto dentro de la caja mientras que el otro muestra menos espacio de relleno.

Para analizar un problema como el de text-shadow un paso lógico es preguntarse: ¿soportan estos navegadores la propiedad text-shadow? Es decir, ¿están preparados para interpretarla correctamente? Da la impresión de que el segundo navegador no. De todas formas, para comprobarlo realizamos la búsqueda “text-shadow mozilla developer” en internet y comprobamos lo siguiente:

Desktop Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	2.0.158.0	3.5 (1.9.1)	10	9.5	1.1 (100)

Aquí nos indica para la propiedad text-shadow la compatibilidad de los navegadores y nos dice que esta propiedad es admitida en Chrome desde la versión 2.0.158.0, en Firefox desde la versión 3.5, en

Internet Explorer desde la versión 10, en Opera desde la versión 9.5 y en Safari desde la versión 1.1. Esto nos confirma lo que sospechábamos: el navegador que no muestra el efecto text-shadow no lo hace porque no está preparado para interpretar esta propiedad, debido a su antiguedad.

Es conveniente realizar comprobaciones básicas de este tipo antes de dar por hecho que "la culpa es del navegador", ya que a veces la culpa no es del navegador, sino del programador o diseñador que ha creado un código inadecuado.

Una vez conocemos el problema podemos elegir entre dos opciones: corregirlo para que la visualización sea correcta en el navegador donde hemos detectado el problema, o ignorarlo. Si lo ignoramos la página web no se verá correctamente en ese navegador, pero como hemos comentado no se pueden comprobar ni corregir los problemas de visualización en todos los navegadores. Hay navegadores muy antiguos en los que no se suelen chequear ni corregir problemas de visualización porque se consideran obsoletos y que hay muy poca gente que pueda estar usándolos todavía. No tiene sentido aspirar a que nuestros desarrollos web se vean correctamente en todos los navegadores porque esto nos llevaría semanas o meses de trabajo y tendría poco interés práctico: lo normal es centrarse en que la página se vea bien en los principales navegadores en versiones relativamente modernas (hasta qué versión comprobar es algo que cada desarrollador o equipo de desarrollo tiene que decidir).

VARIAS FORMAS DE SOLUCIONAR UN PROBLEMA

Con frecuencia existen varias formas de solucionar un problema. Vamos a referirnos, a modo de ejemplo, al problema de que la caja no se muestre centrada en uno de los navegadores que hemos probado. A modo de comprobación, realizamos la búsqueda "margin css mozilla developer" en internet y nos vamos a la página de documentación de Mozilla correspondiente. En la parte denominada "Browser compatibility" aparece que para Internet Explorer el Basic support fue introducido en la versión 3.0 mientras que para auto value indica que la versión en que se introdujo fue 6.0 (strict mode). Esto nos da una pista sobre una forma de solucionar el problema: la indicación de strict mode quiere decir que Internet Explorer no realiza el centrado con margin: 0 auto; excepto si en la cabecera del documento HTML se hace una declaración de que el documento está definido en HTML strict. Para corregir el problema escribiríamos esta línea como primera línea del documento HTML:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

Una vez hecho esto comprobaríamos que la caja ya aparece centrada. Quizás esta solución pueda resultarnos adecuada. O quizás no, porque no queramos realizar este tipo de declaración.

También es posible que no nos demos cuenta de que el problema se podría resolver de esta manera.

En caso de que no nos resulte una solución adecuada o que se nos pase por alto la posibilidad de usar esta solución, normalmente dispondremos de varias alternativas para lograr un mismo objetivo. Existen en internet muchas webs donde se habla de técnicas CSS para resolver problemas o conseguir un objetivo de distintas maneras. No tiene sentido aspirar a conocerlo todo ni a saber todos los "trucos" o alternativas posibles. Sí será conveniente que nos vayamos construyendo nuestro pequeño repertorio de trucos o formas de solución y que nos acostumbremos a buscar en internet formas de solucionar problemas.

FILTROS CSS

Se habla de "filtro CSS" en alusión a alguna instrucción o mecanismo que permita conocer qué navegador, versión o tipo de dispositivo (tablet, smartphone, etc.) es en el que se va a visualizar la página web y aplicar unas reglas específicas para ese caso.

Lo ideal sería no tener que usar filtros, sino que unas solas reglas nos resolvieran todos los problemas, pero en la práctica a veces no queda más remedio que recurrir a diversas estrategias para lograr mejorar la visualización en navegadores o dispositivos que presentan problemas de visualización y no queremos dejar atrás. No obstante, ten presente una cosa: cuantos menos filtros, trucos o especificidades utilices mejor, el código será más sencillo y por tanto más fácil de mantener y de corregir.

Existen diversos tipos de filtros y diversas técnicas. Vamos a comentar ahora una de ellas, que se ha venido usando para resolver problemas de visualización específicos del navegador Internet Explorer de Microsoft, denominada técnica de "comentarios condicionales para Internet Explorer".

La forma de aplicar estos filtros se basa en indicar con un condicional unas reglas específicas a cargar y aplicar exclusivamente para ese navegador (o conjunto de navegadores). Las reglas se escriben dentro de las etiquetas <head> … </head> del documento HTML y estos ejemplos facilitan su comprensión:

Ejemplo	Significado
<!--[if IE 7]> <link rel="stylesheet" type="text/css" href="ie7.css"> <![endif]-->	Si el navegador es Internet Explorer 7, concretamente para este navegador aplicar las reglas definidas en el archivo indicado. Se puede especificar IE 7, IE 8, IE 9, IE 10, IE 12, IE 15, IE 20, IE 30… es decir, cualquier versión que queramos.
<!--[if IE]> <link rel="stylesheet" type="text/css" href="ie.css"> <![endif]-->	Si el navegador es Internet Explorer en cualquiera de sus versiones, concretamente para este navegador aplicar las reglas definidas en el archivo indicado.
<!--[if gte IE 15]> <link rel="stylesheet" type="text/css" href="ie15h.css"> <![endif]-->	Si el navegador es Internet Explorer 15 o superior, concretamente para esta versión o sus superiores aplicar las reglas definidas en el archivo indicado. gte = greater than or equal to
<!--[if gt IE 15]> <link rel="stylesheet" type="text/css" href="ie15u.css"> <![endif]-->	Si el navegador es Internet Explorer versión superior a la 15, concretamente para estas versiones o sus superiores aplicar las reglas definidas en el archivo indicado. gt = greater than
<!--[if lte IE 15]> <!--[if lt IE 15]>	Mismo razonamiento pero para versiones inferiores o iguales, o inferiores. lt = lower than, inferior. lte = lower than or equal to, inferior o igual a.
<!-- [if (gt IE 6)&(lt IE 10)]> <link rel="stylesheet" type="text/css" href="ieR.css"> <![endif]-->	Operador lógico "and", si el navegador cumple el primer requisito y el segundo al mismo tiempo, concretamente para esas versiones aplicar las reglas definidas en el archivo indicado.
<!-- [if (IE 19)|(IE 20)]> <link rel="stylesheet" type="text/css" href="ieY.css"> <![endif]-->	Operador lógico "or", si el navegador cumple el primer requisito, o el segundo, o ambos, concretamente para esas versiones aplicar las reglas definidas en el archivo indicado.

Nota: no deben existir espacios sobrantes dentro de los tags porque en ese caso puede no interpretarse el código.

Ahora veamos cómo sería la aplicación concreta para resolver el problema que hemos descrito anteriormente.En el código HTML añadiríamos el fragmento condicional de la siguiente manera:

```
<html>
<head>
<title>Portal web - aprenderaprogramar.com</title> <meta charset="utf-8">
<link rel="stylesheet" type="text/css" href="CU01055D_A.css">
<!--[if(IE 7)]>
<link rel="stylesheet" type="text/css" href="ie7.css">
<![endif]-->
</head>
<body>
<div id="txtCabecera"><h1>aprenderaprogramar.com</h1></div>
</body>
</html>
```

Fíjate que aquí no hemos incluido la declaración de DOCTYPE que indicamos antes. Vamos a suponer que no queremos incluir esta declaración (o que no nos hemos dado cuenta de que podemos usar esto para resolver el problema de centrado de la caja).

Ahora en un archivo css independiente del principal, al que vamos a denominar ie7.css, incluiremos estas reglas:

```
/* Curso CSS estilos aprenderaprogramar.com*/
body  {text-align:center;}
#txtCabecera { background-color: #FA8258; }
h1 { padding: 15px 5px 5px 5px;}
```

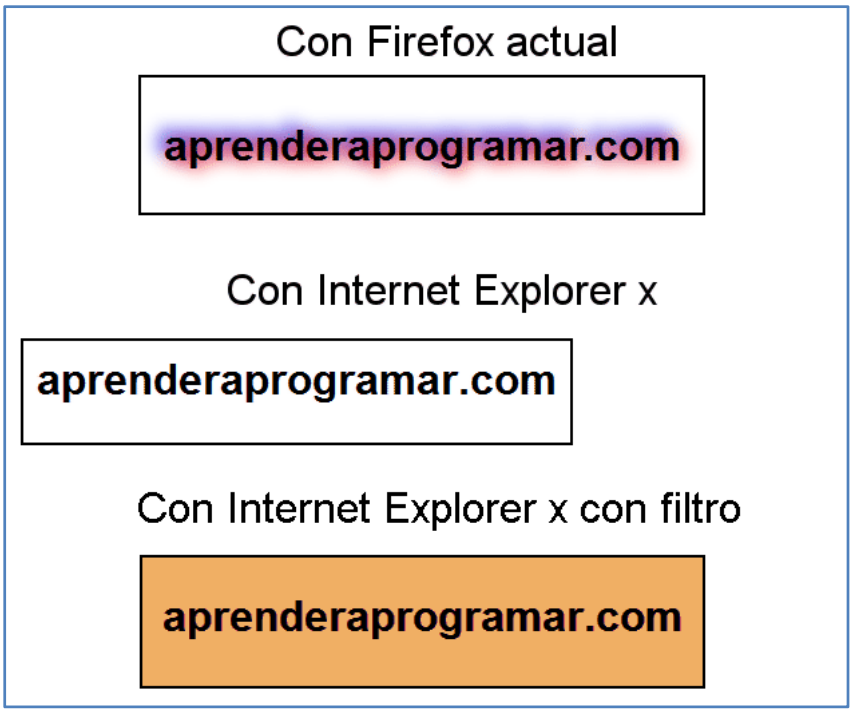
Ahora tenemos 3 archivos. El archivo html del documento html, el archivo CSS principal (en nuestro caso lo hemos llamado CU01055D_A.css) y el archivo CSS específico para los navegadores que cumplen el condicional indicado (en nuestro caso el condicional es que el navegador sea Internet Explorer7 y el archivo lo hemos denominado ie7.css). Las tres reglas definidas en el archivo ie7.css se aplicarán sólo cuando el navegador donde se vaya a mostrar la página web sea el que hemos indicado en el condicional. Las reglas que aplicamos son:

a) body {text-align:center;} , alineamos al centro los elementos dentro de body. Esta regla posiblemente no la usaríamos en un desarrollo web real porque en general no nos va a interesar alinear todos los elementos dentro de body al centro (si lo hacemos, tendríamos que establecer alineaciones específicas para el resto de elementos cuando no quisiéramos que estén centrados). Pero para este ejemplo concreto nos sirve.
b) #txtCabecera {background-color: #FA8258;} , a falta del efecto text-shadow aplicamos como sustituto un color de fondo. No logramos el mismo efecto que con text-shadow, pero supondremos que es la alternativa que elegimos entre las varias posibles. Otra alternativa sencilla sería usar una imagen de fondo. Podríamos asemejar más el aspecto al que genera text-shadow, pero no vamos a entretenernos en ello.

c) h1 { padding: 15px 5px 5px 5px;}, ampliamos el relleno para conseguir una visualización lo más parecida a lo que nosotros deseamos.

Ten en cuenta que hablamos de "visualización parecida", similar, etc. ya que no podemos o no queremos detenernos a buscar "la precisión absoluta" porque esto no tiene sentido.

El resultado ahora obtenido lo comprobamos a continuación. Gracias al filtro hemos centrado la caja, adecuado su relleno y aunque no hemos igualado el efecto de sobra al menos hemos puesto un color de fondo:



MÁS FILTROS CSS

Hemos comentado una técnica de filtrado concreta, pero hay otras técnicas. Algunas de esas otras técnicas las veremos más adelante dentro del curso. Ten en cuenta que las técnicas de filtrado van evolucionando con el tiempo: algunas técnicas dejan de ser útiles o válidas mientras que surgen otras nuevas. En este curso no nos interesa conocer las diferentes técnicas de filtrado para los distintos navegadores, sino comprender la problemática existente en torno a las diferencias entre navegadores y el concepto de filtrado como técnica que puede ayudar a solucionar problemas específicos.

EJERCICIO

Busca en alguna página web o blog de internet una técnica de filtrado, aplícala y explica su utilidad. Para comprobar si tu código y respuestas son correctas puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU01056D

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente: http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=75&Itemid=203