Canvas

Canvas.

Autor: Yuliana Melissa Vera Jaramillo

*Universidad Tecnologica de Pereira, Pereira, Colombia*

Correo: yuliana.vera@utp.edu.co

***Resumen*— “<canvas> es un elemento HTML el cual puede ser usado para dibujar gráficos usando scripts (normalmente JavaScript). Este puede, por ejemplo, ser usado para dibujar gráficos, realizar composición de fotos o simples (y no tan simples) animaciones”.**

***Palabras clave—* HTML, gráficos, dibujos, animaciones, scripts.**

***Abstract*— <canvas> is an HTML element which can be used to draw graphics using scripts (usually JavaScript). This can, for example, be used to draw graphics, make composition of photos or simple (and not so simple) animations.**

***Keys Words* —** **HTML, graphics, drawings, animations, scripts.**

INTRODUCCIÓN

“Canvas (o lienzo, traducido al español), es un elemento añadido a HTML5 mediante el cual, se puede dibujar usando scripts (habitualmente JavaScript).

Fué introducido por Apple para Mac OS X Dashboard y después implementado en Safari y Google Chrome. No está soportado en navegadores antiguos, pero si funciona en la mayoría de versiones más recientes de los navegadores.

Canvas como tal, es solo un contendor de gráficos, un lienzo como su nombre indica, ya que la «magia» la haremos con JavaScript.

El verdadero potencial de Canvas reside en la capacidad para actualizar contenidos en tiempo real, lo cual unido a la posibilidad de responder a eventos de usuario, proporciona un abanico de posibilidades para crear herramientas o juegos”.

1. CONTENIDO

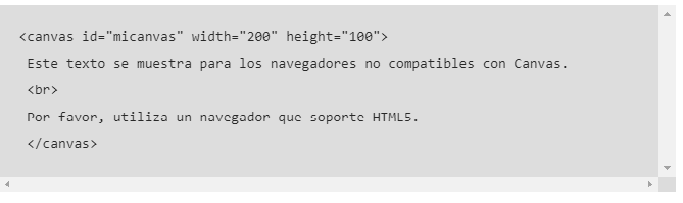
Canvas (lienzo) es un elemento HTML que permite la creación de gráficos y animaciones de forma dinámica por medio de scripts. Sus aplicaciones son practicamente iniminaginables: crear juegos, interfaces, editores gráficos o efectos dinámicos, aplicaciones 3D…. Sólo la imaginación pone límites a Canvas. En este artículo, profundizamos en las posibilidades de Canvas y las ejemplificamos con un sencillo gráfico.

A modo anecdótico, sólo señalar Canvas fue creado por Apple, pero han liberado la propiedad intelectual para englobarlo dentro de los estándares de HTML. Para el usuario, todo son ventajas. No requiere ningún plugin adicional, sólo una un navegador que soporte HTML5 y hoy en día todos los navegadores importantes (Safari, Chrome, Firefox, Opera e Internet Explorer) soportan Canvas, desde hace ya unas cuantas versiones.

Realizar un dibujo con el API de Canvas

El ejemplo consistirá en dibujar dos rectángulos de colores diferentes y, para ello, utilizaremos un par de funciones del API con JavaScript. Primero, reservaremos el espacio del lienzo Canvas dentro de la página, utilizando la etiqueta HTML canvas, y después, dibujaremos los rectángulos.

Colocando el elemento HTML Canvas



Atributos de este Canvas

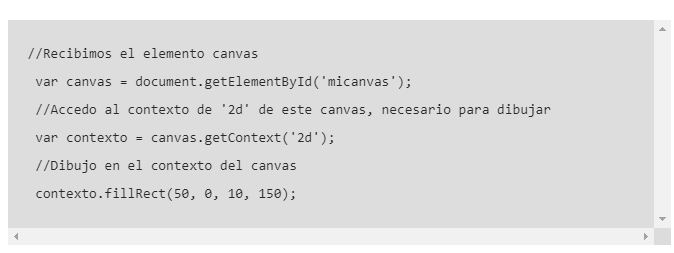
Atributo id: Le asignamos un nombre único para referirnos a éste desde JavSscript.

Atributos width y height: Indicamos el alto y ancho del área del canvas.

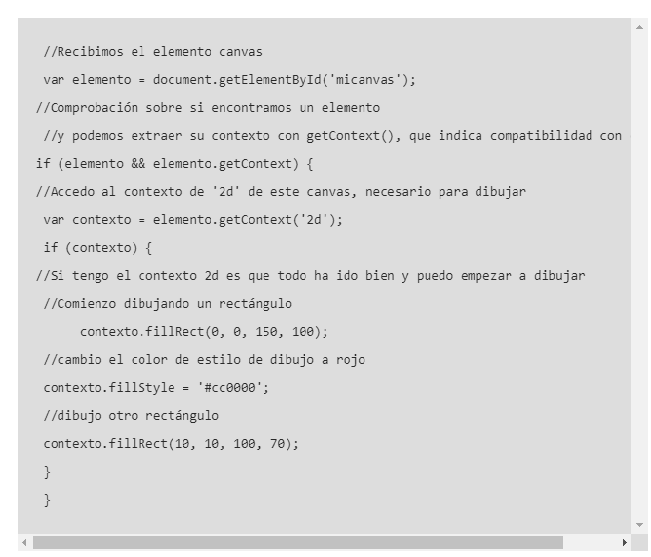
Por supuesto, podemos colocar de manera opcional otros, como style, para indicar atributos de hojas de estilo y definir su aspecto.

Dibujando el Canvas con JavaScript

Dibujaremos un par de formas sencillas, aunque antes comprobaremos sin el navegador es compatible con Canvas.



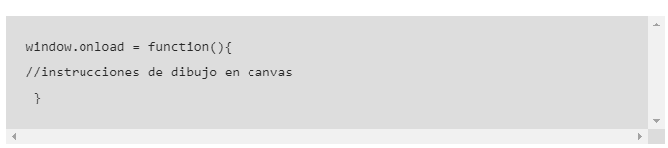
Con getElementById() obtenemos el elemento de la página que se pasa como parámetro, que es el Canvas. Después accedemos al contexto 2D del canvas y, por último, ejecutamos tantos métodos como deseemos sobre el contexto del Canvas para dibujar elementos en el lienzo.



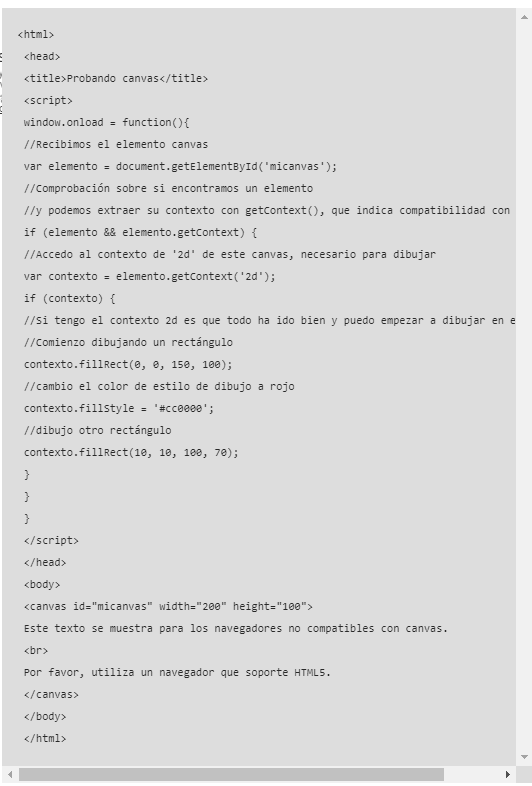
Con el evento onload del body, ejecutaremos las acciones cuando la página esté cargada por completo:



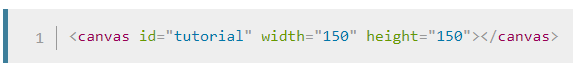
Aunque antes deberemos crear la función funcionDeDibujo() o el recurso para asignar el evento directamente desde un script Javascript:



A continuación, el código completo que resume lo expuesto en el artículo:



La sección de<canvas> elementos



A primera vista, se <canvas>parece al <img>elemento, con la única diferencia clara de que no tiene los atributos srcy alt. De hecho, el <canvas>elemento tiene solo dos atributos, widthy height. Ambos son opcionales y también se pueden configurar usando las propiedades DOM . Cuando se especifican no y atributos, el lienzo inicialmente tendrá 300 píxeles de ancho y 150 píxeles de alto. El elemento puede ser dimensionado arbitrariamente por CSS , pero durante la representación, la imagen se ajusta para ajustarse a su tamaño de diseño: si el tamaño de CSS no respeta la proporción del lienzo inicial, aparecerá distorsionado.width height

El idatributo no es específico del <canvas>elemento, pero es uno de los atributos HTML globales que se pueden aplicar a cualquier elemento HTML (como classpor ejemplo). Siempre es una buena idea proporcionar una, idya que esto hace que sea mucho más fácil identificarla en un script.

El <canvas>elemento puede ser de estilo al igual que cualquier imagen normal ( margin, border, background...). Estas reglas, sin embargo, no afectan el dibujo real en el lienzo.

El <canvas>elemento difiere de una <img>etiqueta en que, como para <video>, <audio>o <picture>elementos, es fácil definir algún contenido alternativo, que se mostrará en los navegadores más antiguos que no lo admiten, como las versiones de Internet Explorer anteriores a la versión 9 o los navegadores de texto. Siempre debe proporcionar contenido alternativo para que se muestren en esos navegadores.

Proporcionar contenido alternativo es muy sencillo: solo inserte el contenido alternativo dentro del <canvas>elemento. Los navegadores que no son compatibles <canvas>ignorarán el contenedor y mostrarán el contenido de respaldo en su interior. Los navegadores que sí admiten <canvas>ignorarán el contenido dentro del contenedor y simplemente representarán el lienzo normalmente.

Por ejemplo, podríamos proporcionar una descripción de texto del contenido del lienzo o proporcionar una imagen estática del contenido renderizado dinámicamente. Esto puede verse algo como esto:



Por ejemplo, decirle al usuario que use un navegador diferente que admita lienzo no ayuda a los usuarios que no pueden leer el lienzo. Proporcionar un texto de reserva o sub DOM útil ayuda a que el lienzo sea más accesible.

</canvas>Etiqueta requerida Sección

Como consecuencia de la forma en que se proporciona el respaldo, a diferencia del <img>elemento, el <canvas>elemento requiere la etiqueta de cierre ( </canvas>). Si esta etiqueta no está presente, el resto del documento se consideraría contenido de respaldo y no se mostraría.

Si el contenido alternativo no es necesario, un simple <canvas id="foo" ...></canvas>es totalmente compatible con todos los navegadores que admiten lienzos.

La sección de contexto de representación

El <canvas>elemento crea una superficie de dibujo de tamaño fijo que expone uno o más contextos de representación , que se utilizan para crear y manipular el contenido mostrado. En este tutorial, nos centramos en el contexto de renderización 2D. Otros contextos pueden proporcionar diferentes tipos de representación; por ejemplo, WebGL utiliza un contexto 3D basado en OpenGL ES .

El lienzo está inicialmente en blanco. Para mostrar algo, una secuencia de comandos primero debe acceder al contexto de representación y dibujar en él. El <canvas>elemento tiene un método llamado getContext(), que se utiliza para obtener el contexto de representación y sus funciones de dibujo. getContext()Toma un parámetro, el tipo de contexto. Para gráficos en 2D, como los que se tratan en este tutorial, especifique "2d"para obtener un CanvasRenderingContext2D.



La primera línea en el script recupera el nodo en el DOM que representa el <canvas>elemento llamando al document.getElementById()método. Una vez que tenga el nodo del elemento, puede acceder al contexto de dibujo utilizando su getContext()método.

Comprobando la sección de soporte

El contenido alternativo se muestra en los navegadores que no son compatibles <canvas>. Los scripts también pueden verificar el soporte mediante programación simplemente comprobando la presencia del getContext()método. Nuestro fragmento de código de arriba se convierte en algo como esto:



Una sección de plantilla de esqueleto

Aquí hay una plantilla minimalista:

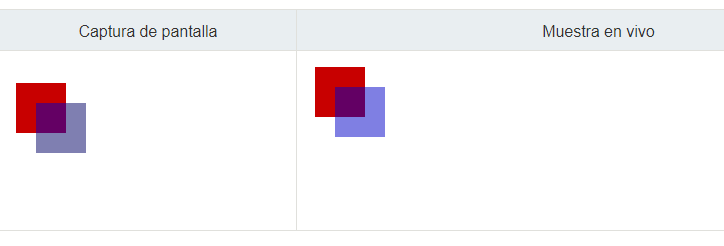


El script incluye una función llamada draw(), que se ejecuta una vez que la página termina de cargarse; Esto se hace escuchando el loadevento en el documento. Esta función, o una similar, también podrían ser llamadas usando window.setTimeout(), window.setInterval()o cualquier otro controlador de eventos, siempre y cuando la página se ha cargado en primer lugar.

Para comenzar, echemos un vistazo a un ejemplo simple que dibuja dos rectángulos que se cruzan, uno de los cuales tiene transparencia alfa.



Este ejemplo se ve así:



1. CONCLUSIONES

El elemento HTML canvas (<canvas>) se puede utilizar para dibujar gráficos a través de secuencias de comandos (por lo general JavaScript ). Por ejemplo, puede usarse para dibujar gráficos, hacer composiciones de fotos o incluso realizar animaciones.

REFERENCIAS

1. <https://www.adictosaltrabajo.com/2017/05/18/introduccion-a-canvas/>
2. <https://www.arsys.es/blog/programacion/diseno-web/que-es-canvas/>
3. <https://developer.mozilla.org/es/docs/Web/Guide/HTML/Canvas_tutorial>
4. <https://developer.mozilla.org/es/docs/Web/HTML/Elemento/canvas>