Canvas

Autor: Juan David Arce Martinez
Facultad de ingenierías, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: juandavid.arce@utp.edu.co

***Resumen*— Canvas se encuentra ligado a HTML, CSS y a JavaScript, Canvas (lienzo) es un elemento HTML que permite la creación de gráficos y animaciones de forma dinámica por medio de scripts. Sus aplicaciones son prácticamente inimaginables: crear juegos, interfaces, editores gráficos o efectos dinámicos, aplicaciones 3D…. Sólo la imaginación le puede poner límites a Canvas. La importancia de Canvas ha ido creciendo cada vez más hasta llegar a lo que es ahora, es una herramienta muy útil en el campo de la tecnología y aún más en el de la informática.**

Palabras clave__ Canvas, CSS, HTML, Informática, scripts, Tecnología,

***Abstract*— Canvas is linked to HTML, CSS and JavaScript, Canvas (canvas) is an HTML element that allows the creation of graphics and animations dynamically by means of scripts. Its applications are practically unimaginable: create games, interfaces, graphic editors or dynamic effects, 3D applications.... Only imagination can put limits on Canvas. The importance of Canvas has been growing more and more until it reaches what it is now, it is a very useful tool in the field of technology and even more in that of information technology.**

***Key Word* — Canvas, CSS, HTML, Computing, scripts, Technology,**

I. 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» se hace 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.

II. CONTENIDO

Sistemas operativos

[1].Canvas está soportado para los siguientes sistemas operativos:

-Windows 7 y más nuevos Mac OSX 10.6 y más nuevos
-Linux – chromeOS
-Android
-IOS

El elemento HTML <canvas> se utiliza para dibujar gráficos en una página web.

Dibujar un rectángulo

A diferencia de SVG, <canvas> solo soporta una forma primitiva: rectángulos. Todas las otras formas deben ser creadas por la combinación de uno o más trazos, listas de puntos conectados por líneas. Afortunadamente, tenemos una variedad de funciones para dibujar trazos que hacen posible componer formas muy complejas.

Primero veamos el rectángulo. Estas son tres funciones que se pueden usar en el canvas para dibujarlos:

-fillRect(x, y, width, height)
Dibuja un rectángulo relleno.

-strokeRect(x, y, width, height)
Dibuja el contorno de un rectángulo.

-clearRect(x, y, width, height)
Borra un área rectangular especificada, dejándola totalmente transparente.

Cada una de estas tres funciones toma los mismos parámetros. X e Y especifican la posición del canvas (en relación con el origen) desde la esquina superior izquierda del rectángulo. También especifica los parámetros de anchura y altura que proporcionan el tamaño del rectángulo.

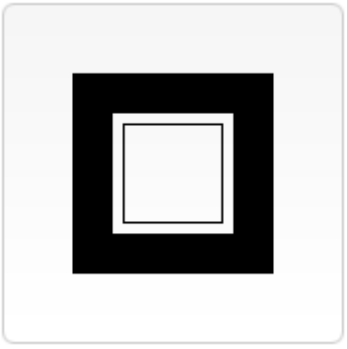
Fecha de Recepción: (Letra Times New Roman de 8 puntos)
Fecha de Aceptación: Dejar en blanco

Ejemplo de forma rectangular

```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext) {
    var ctx = canvas.getContext('2d');

    ctx.fillRect(25,25,100,100);
    ctx.clearRect(45,45,60,60);
    ctx.strokeRect(50,50,50,50);
  }
}
```

El anterior código nos daría como resultado:



La función fillRect() dibuja un cuadrado grande negro de 100 píxeles en cada lado. La función clearRect() luego borra un cuadrado de 60x60 píxeles del centro, y luego strokeRect() es llamado para crear un contorno rectangular de 50x50 píxeles dentro del cuadrado borrado.

Líneas

Para dibujar líneas rectas usa el método lineTo().

lineTo(x, y)

Dibuja una línea desde la posición actual del dibujo a la posición especificada por x e y. Este método toma dos argumentos x e y, los cuales son las coordenadas del punto final de la línea. El punto de inicio es dependiente de los trazos previamente dibujados, donde el punto final del trazo anterior es el punto inicial para el siguiente, etc. El punto de inicio también puede ser cambiado usando el método moveTo().

El siguiente ejemplo dibuja dos triángulos, uno rellenado y el otro contorneado.

```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext){
    var ctx = canvas.getContext('2d');

    // Triángulo rellenado
    ctx.beginPath();
    ctx.moveTo(25,25);
    ctx.lineTo(105,25);
    ctx.lineTo(25,105);
    ctx.fill();

    // Triángulo contorneado
    ctx.beginPath();
    ctx.moveTo(125,125);
    ctx.lineTo(125,45);
    ctx.lineTo(45,125);
    ctx.closePath();
    ctx.stroke();
  }
}
```

[2].Esto comienza llamando a beginPath() para empezar una nueva forma. Entonces usamos el método moveTo() para mover el punto de inicio a la posición deseada. Debajo de esto dos líneas son dibujadas lo cual pinta dos lados del triángulo.

Curvas Bezier curvas cuadráticas

El siguiente tipo de trazos disponibles son las curvas Bézier, en sus dos variantes, cúbicas y cuadráticas. Son usadas generalmente para dibujar complejas formas orgánicas.

quadraticCurveTo(cp1x, cp1y, x, y)
Dibuja una curva cuadrática de Bézier desde la posición actual de la pluma hasta el punto final especificado por x e y, utilizando el punto de control especificado por cp1x y cp1y.

bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)

[3].Dibuja una curva cúbica de Bézier desde la posición actual de la pluma hasta el punto final especificado por x e y, utilizando los puntos de control especificados por (cp1x, cp1y) y (cp2x, cp2y).

La diferencia entre estos puede describirse mejor utilizando la imagen de la derecha. Una curva cuadrática de Bézier tiene un punto inicial y un punto final (puntos azules) y un solo punto de control (indicado por el punto rojo), mientras que una curva cúbica de Bézier utiliza dos puntos de control.

Los parámetros x e y de ambos métodos son las coordenadas del punto final. cp1x y cp1y son las coordenadas del primer punto de control, y cp2x y cp2y son las coordenadas del segundo punto de control.

El uso de curvas cuadráticas y cúbicas Bézier puede ser bastante difícil, ya que a diferencia del software de dibujo vectorial como Adobe Illustrator, no tenemos respuesta visual directa en cuanto a lo que estamos haciendo. Esto hace bastante difícil dibujar formas complejas

```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext){
    var ctx = canvas.getContext('2d');

    // Quadratric curves example
    ctx.beginPath();
    ctx.moveTo(75,40);
    ctx.bezierCurveTo(75,37,70,25,50,25);
    ctx.bezierCurveTo(20,25,20,62.5,20,62.5);
    ctx.bezierCurveTo(20,80,40,102,75,120);
    ctx.bezierCurveTo(110,102,130,80,130,62.5);
    ctx.bezierCurveTo(130,62.5,130,25,100,25);
    ctx.bezierCurveTo(85,25,75,37,75,40);
    ctx.fill();
  }
}
```

El código anterior nos mostraría en pantalla:

[4].Para finalizar en el siguiente ejemplo vamos a ver cómo podemos dibujar un círculo sobre un Canvas de HTML5. La realidad es que dentro de la especificación del Canvas de HTML5 no existe ningún método que nos dibuje un círculo directamente, si no, que nos ofrece un método que dibuja arcos. Y por definición un círculo no deja de ser un arco de 360º.

Así tenemos que conocer en profundidad la función .arc(), con la cual vamos a dibujar nuestro círculo sobre el Canvas. La sintaxis de este método sería:

```
void arc(in float x, in float y, in float radius, in float startAngle, in float endAngle, in boolean anticlockwise Optional );
```

Echándole un vistazo por encima lo que vemos es que podemos dibujar un arco en unas coordenadas (x,y), cuyo radio será el del valor del radius y tendrá un angulo de inicio y otro de fin.

Nos vamos a detener en estos dos valores y es que los valores de los ángulos hay que expresarlos en radianes. Así que, tendremos:

180 grados = π radianes
Por lo tanto, si queremos trabajar con grados, tendremos que utilizar la siguiente ecuación:
(π/180)*grados

Es decir, si queremos trazar el arco en la dirección de las agujas del reloj, le damos un valor de false, si queremos al revés que la dirección de las agujas, pues le damos true.

Para pintar nuestro primer arco, vamos a crear uno que vaya del 0 a 270 grados.

```
ctx.arc(100,70,50,0,(Math.PI/180)*270,true);
```

O que sea medio círculo, del 0 al 180:

```
ctx.arc(230,70,50,0,(Math.PI/180)*180,true);
```

ya solo nos queda saber dos cosas. La primera es cómo indicar el color del relleno del círculo. Esto lo hacemos con la propiedad. fillStyle y siempre ejecutando el método. fill() Ya que si no, no se ejecutara el pintado del relleno.

```
1. ctx.arc(360,70,50,0,(Math.PI/180)*360,true);
2. ctx.fillStyle="#f99";
3. ctx.fill();
```

1. Las notas de pie de página deberán estar en la página donde se citan. Letra Times New Roman de 8 puntos

Y si queremos poner un borde al círculo tenemos las propiedades. strokeStyle y .lineWidth. En la primera le damos el color al borde mediante un formato RGB y la segunda, en pixels, indicamos el ancho del borde.

```
1. ctx.arc(360,70,50,0,(Math.PI/180)*3
   60,true);
2. ctx.strokeStyle = "#f00";
3. ctx.lineWidth = 10;
4. ctx.stroke();
```

Lo último que haremos, será ejecutar la función. stroke() que sin ella no se volcará nada de contenido al Canvas. Así ya tenemos dibujado nuestro círculo sobre un Canvas de HTML5

III. CONCLUSIONES

Canvas es un elemento HTML que delimita un lienzo para dibujar gráficos mediante JavaScript haciendo uso de un objeto de contexto que los crea sobre la marcha. Su API contiene métodos que permiten crear trazados en contextos 2D y 3D. La representación de los gráficos utiliza el modelo “fire and forget”, los gráficos no se guardan, una vez generados no podremos obtener su forma y posición. Para la creación de una escena, el desarrollador debe repintar el gráfico completamente cada vez que se quiera modificar

Desde la llegada de HTML5 existen estándares para manipular gráficos en la web. El uso de Canvas y SVG (Scalable Vector Graphics) son la mejor opción para crear gráficos rápidos y ligeros en una página web. Son muchas las ventajas que nos proporcionan este tipo de estándares, por ejemplo, evitan la instalación de complementos, permiten un buen posicionamiento o se visualizan de forma correcta en dispositivos móviles

REFERENCIAS

Referencias de publicaciones periódicas:

[1] Wikipedia. (17 dic 2017.).[Online].Canvas Available: https://es.wikipedia.org/wiki/Canvas

[2] Oscar Campos (20 de junio de 2011). «Introducción al elemento canvas de HTML5»

[3] Jose Mª Baquero García. (17 dic 2017.).[Online].¿Que es Canvas ? Available: https://www.arsys.es/blog/programacion/diseno-web/que-es-canvas/

[4] Md web docs. (2019.).[Online]. Crea el lienzo (canvas) y dibuja en él Available: https://developer.mozilla.org/es/docs/Games/Workflows/Famoso_juego_2D_usando_JavaScript_puro/Create_the_Canvas_and_draw_on_it