

w3.unpo<code>todo

PUBLICIDAD:



&NBSP;

Canvas - la chuleta



General

Método	JavaScript	Descripción	Defecto
width	<code>canvas.width</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la anchura del canvas	300
height	<code>canvas.height</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la altura del canvas	150
getContext()	<code>canvas.getContext("2d");</code>	Devuelve un objeto que proporciona todos los métodos y propiedades para dibujar en el canvas.	
toDataURL()	<code>canvas.toDataURL(tipo);</code>	Convierte el contenido del canvas en una imagen - data:uri. El parámetro entre paréntesis indica el tipo de imagen	image/png


Colores, estilos, y sombras

Método	JavaScript	Descripción	Defecto
fillStyle	<code>context.fillStyle = color gradiente patrón;</code>	Determina o devuelve el color, gradiente o patrón del relleno.	 negro
strokeStyle	<code>context.strokeStyle = color gradiente patrón;</code>	Determina o devuelve el color, gradiente o patrón de la línea.	negro
lineWidth	<code>context.lineWidth=numero;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la grosor de la línea.	1

shadowColor	<code>context.shadowColor=color;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el color utilizado para las sombras.	#000000; transparente
shadowBlur	<code>context.shadowBlur=number;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el nivel de desenfoque de las sombras.	0
shadowOffsetX	<code>context.shadowOffsetX=number;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la distancia horizontal entre la sombra y la forma que la genera. El valor tiene que ser > 0 para que la sombra tenga efecto.	0
shadowOffsetY	<code>context.shadowOffsetY=number;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la distancia horizontal entre la sombra y la forma que la genera. El valor tiene que ser > 0 para que la sombra tenga efecto.	0

Método	JavaScript	Descripción	Defecto
createLinearGradient()	<code>context.createLinearGradient(x0,y0,x1,y1);</code>	Crea un gradiente lineal para utilizar en el <canvas> x0,y0 son las coordenadas del punto donde empieza el gradiente. x1,y1 son las coordenadas del punto donde acaba el gradiente.	
createPattern()	<code>context.createPattern(img,"repeat repeat-x repeat-y no-repeat");</code>	Repite una imagen en la dirección especificada.	
createRadialGradient()	<code>context.createRadialGradient(x0,y0,r0,x1,y1,r1);</code>	Crea un gradiente radial para utilizar en el <canvas> x e y son las coordenadas del centro de los círculos r es el radio de los círculos.	
addColorStop()	<code>gradient.addColorStop(stop,color);</code>	Especifica los colores y la posición donde para el gradiente.	

Líneas


Propiedad	JavaScript	Descripción	Defecto
lineWidth	<code>context.lineWidth=numero;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la grosor de la línea.	1
lineCap	<code>context.lineCap="butt round square";</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el aspecto de las puntas de una línea.	butt
lineJoin	<code>context.lineJoin="bevel round miter";</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el aspecto de las juntas entre líneas. Posibles valores: bevel (biselado), round (redondeado), miter (en angulo)	mitter
miterLimit	<code>context.miterLimit=numero;</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el aspecto de las juntas en miter. Puede tomar valores entre 1 (punta roma, aspecto biselado) y 5 (punta en angulo). 	10

Rectángulos

Método	JavaScript	Descripción	Defecto
rect()	<code>context.rect(x,y,anchura,altura);</code>	Define un rectángulo desde un punto (x,y).	

fillRect()	<code>context.fillRect(x,y,anchura,altura);</code>	Define y rellena un rectángulo desde un punto (x,y).	
strokeRect()	<code>context.strokeRect(x,y,anchura,altura);</code>	Define y dibuja un rectángulo desde un punto (x,y).	
clearRect();	<code>context.clearRect(x,y,width,height);</code>	Borra los píxeles especificados dentro de un rectángulo dado.	

Trazados

Método	JavaScript	Descripción	Defecto
fill()	<code>context.fill();</code>	Rellena una forma geométrica.	black
stroke()	<code>context.stroke();</code>	Dibuja una línea ya definida.	
beginPath()	<code>context.beginPath();</code>	Inicia un nuevo trazado	
moveTo()	<code>context.moveTo(x,y);</code>	Mueve el "lápiz" a un punto en el canvas, especificado por sus coordenadas "x" e "y". NO DIBUJA ninguna línea.	x=0; y=0;
closePath()	<code>context.closePath();</code>	Cierra una línea poligonal o una línea curva	
lineTo()	<code>context.lineTo(x,y);</code>	Define una línea desde un punto especificado anteriormente hasta otro punto especificado por sus coordenadas "x" e "y". Mueve el "lápiz" a este punto.	
clip()	<code>context.clip()</code>	Recorta una región con la forma y tamaño del trazado dibujado previamente en el canvas. Cualquier cosa dibujada después, será visible solo dentro de la región de recorte (<i>clipping region</i>).	
quadraticCurveTo()	<code>context.quadraticCurveTo(cx,cy,x,y);</code>	Define una curva cuadráticas de Bézier. cx,cy = coordenadas punto de anclaje (control point) x,y = coordenadas punto final (el punto de partida siendo determinado previamente).	
bezierCurveTo()	<code>context.bezierCurveTo(cx1,cy1,cx2,cy2,x,y);</code>	Define una curva de Bézier. cx1,cy1,cx2,cy2= coordenadas puntos de anclaje (control points) x,y = coordenadas punto final (el punto de partida siendo determinado previamente).	
arc()	<code>context.arc(x, y, r, sA, eA, aC)</code>	Define un segmento circular. x y = coordenadas centro r = radio sA = ángulo de partida en radianes eA = ángulo final en radianes aC = sentido contra reloj (anti-Clockwise) true/false 360° = 2π radianes. radianes = (Math.Pi / 180) * grados;	
arcTo()	<code>context.arcTo(x1,y1,x2,y2,r);</code>	Crea un arco de círculo entre dos tangentes x1, y1 y x2, y2	
ellipse()	<code>context.ellipse(X, Y, rX, rY, ar, ap, af, cR);</code>	Dibuja una elipse X y Y son las coordenadas del centro, rX y rY representan el radio en x y el radio en y, ar representa el ángulo de rotación del eje horizontal  en radianes), ap es el ángulo de partida (en radianes), af es el ángulo final (en radianes), y cR en el sentido del reloj (false) o en sentido contrario (true)	

isPointInPath()	<code>context.isPointInPath(x,y);</code>	Detecta si un punto cuyas coordenadas son x e y se encuentra en un trazado dado	
------------------------	--	---	--

Transformations

Método	JavaScript	Descripción	Defecto
scale()	<code>context.scale(h,v);</code>	Reduce o amplía a escala el dibujo actual. h = horizontal; v = vertical Valores que pueden tomar los parametros del método: 1=100%, 0.5=50%, 2=200%, etc...	1
rotate()	<code>context.rotate(ángulo);</code>	Gira los trazados posteriores un ángulo dado (en radianes). El punto alrededor del cual gira coincide con el origen del canvas (0,0)	
translate()	<code>context.translate(x,y);</code>	Mueve el origen (0,0) del <canvas> en un punto dado (x,y).	
transform()	<code>context.transform(a,b,c,d,e,f);</code>	Cambia los trazados posteriores, cambiando la matriz de estos.	
setTransform()	<code>context.setTransform(a,b,c,d,e,f);</code>	Reinicia el canvas a los valores iniciales, antes de proceder a cambiar los trazados posteriores.	

Text

Propiedad	JavaScript	Descripción	Defecto
font	<code>context.font = "font-style font-variant font-weight font-size font-family";</code> <code>context.font = "italic small-caps bold 12px arial";</code>		10px sans-serif
textAlign	<code>context.textAlign="center end left right start";</code>		start
textBaseline	<code>context.textBaseline = "alphabetic top hanging middle ideographic bottom";</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) la alineación vertical del texto.	alphabetic

Método	JavaScript	Descripción	Defecto
fillText()	<code>context.fillText(text,x,y,maxWidth);</code>	Dibuja texto relleno con un color, gradiente o patrón previamente definido. maxWidth es opcional. No se admite en Safari.	#000;
strokeText()	<code>context.strokeText(text,x,y,maxWidth);</code>	Dibuja texto bordeado con un color, gradiente o patrón previamente definido. maxWidth es opcional.	black
measureText()	<code>context.measureText(text).width;</code>	Devuelve un objeto que contiene la anchura del texto especificado entre paréntesis.	



Image Drawing

Método	JavaScript	Descripción	Defecto
drawImage()	<code>context.drawImage(img,x,y);</code>	Dibuja una imagen en el canvas desde el punto (x,y)	

drawImage()	<code>context.drawImage(img,x,y,w,h);</code>	Dibuja una imagen en el canvas desde el punto (x , y), donde w y h son el ancho y el alto de la imagen, respectivamente.	
drawImage()	<code>context.drawImage(img,sx,sy,sw,sh,x,y,w,h);</code>	Recorta la imagen empezando desde un punto (sx , sy), sw y sh siendo el ancho y el alto de la zona recortada. Dibuja esta imagen en el canvas desde el punto (x , y), w y h siendo el ancho y el alto de la imagen resultante.	

Pixel Manipulation

Propiedad	JavaScript	Descripción	Defecto
width	<code>imgData.width</code>	Devuelve el ancho del objeto ImageData, en píxeles	
height	<code>imgData.height</code>	Devuelve la altura del objeto ImageData, en píxeles	
data	<code>imgData.data</code>	Devuelve un objeto conteniendo todos los datos del objeto ImageData.	

Método	JavaScript	Descripción	Defecto
createImageData()	<code>context.createImageData(ancho, alto);</code>	Crea un nuevo objeto ImageData en blanco. Toma dos argumentos: la anchura y la altura del objeto creado	
createImageData()	<code>context.createImageData(imgData);</code>	Crea un nuevo objeto ImageData con las mismas dimensiones que el objeto especificado por el argumento imgData.	
getImageData()	<code>context.getImageData (x, y, ancho, alto);</code>	Devuelve un objeto ImageData que copia los datos de los píxeles del rectángulo especificado.	
putImageData()	<code>context.putImageData(imgData, x, y, <u>dirtyX</u>, <u>dirtyY</u>, <u>dirtyWidth</u>, <u>dirtyHeight</u>);</code>	Pone los datos de la imagen (de un objeto ImageData especificado) de nuevo en el canvas	

Compositing

Propiedad	JavaScript	Descripción	Defecto
globalAlpha	<code>context.globalAlpha = numero</code>	Determina (<i>sets</i>) o devuelve (<i>returns</i>) el valor alfa o la transparencia actual del dibujo.	1.0
globalCompositeOperation	<code>context.globalCompositeOperation = "source-in"</code>	Define la apariencia de nuevos trazados, y como estos afectan o están afectados por los trazados ya existentes en el <canvas> . posibles valores: <i>source-over</i> , <i>source-in</i> , <i>source-out</i> , <i>source-atop</i> , <i>destination-over</i> , <i>destination-atop</i> , <i>destination-in</i> , <i>destination-out</i> , <i>lighter</i> , <i>darker</i> , <i>copy</i> , <i>xor</i>	source-over

Other

Método	JavaScript	Descripción	Defecto
save()	<code>context.save();</code>	Guarda el estado actual del canvas.	

<code>restore()</code>	<code>context.restore()</code>	Recupera el estado previamente guardado del canvas.	
<code>getContext()</code>	<code>canvas.getContext("2d");</code>	Devuelve un objeto (<i>context</i>) que proporciona todos los métodos y propiedades para dibujar en el canvas.	
<code>toDataURL()</code>	<code>canvas.toDataURL()</code>	Convierte el contenido del canvas en una imagen data:uri. El parámetro entre paréntesis indica el tipo de imagen.	"image/png"

Artículos relacionados

- [Canvas - la chuleta](#)
- [Canvas - una introducción](#)

Enlaces útiles

- [Vea la chuleta](#) de canvas.
- [Más información acerca del soporte de canvas en los navegadores](#)

PUBLICIDAD:



Morricone, Zimmer & Williams

Concierto: «The music of Morricone, Zimmer & Will
Marbella Arena. 14 Agosto 22:00h

Excelentia

