Desarrollo de aplicaciones multiplataforma

Elaborado por: José Antonio Sánchez

Módulo 3

Programación HTML5

Innova4B

SESIÓN 4

Innova4B

7. Dataset

HTML5 permite incorporar atributos de datos personalizados, custom data (data-*), en todos los elementos HTML

Prefijar los atributos personalizados con data- asegura que van a ser completamente ignorados por el agente de usuario. Para el navegador y el usuario final no existe esta información

Cada elemento HTML puede tener un número indefinido de atributos de datos personalizados, con cualquier valor

7. Dataset

Hasta ahora, la manera de lograr un comportamiento similar era incluir estos datos como clases CSS:

```
<input class="spaceship shields-5 lives-3 energy-75">
```

Una vez definidos estos valores en class era necesario realizar un trabajo extra para extraer su nombre y su valor (convertir energy-75 en energy = 75)

7. Dataset

Gracias a los atributos dataset esto ya no es necesario, y se crean de la siguiente forma:

Nombre del atributo: debe ser de al menos un carácter de largo y debe tener el prefijo data-. No debe contener letras mayúsculas

Valor del atributo: el valor del atributo puede ser cualquier cadena

```
<ul id="vegetable-seeds">
  <li data-spacing="10cm" data-sowing-time="March to June">Carrots</li>
  <li data-spacing="30cm" data-sowing-time="February to March">Celery</li>
  <li data-spacing="3cm" data-sowing-time="March to September">Radishes</li>
</ul>
```

Innova4B

7. Dataset

Con estos datos almacenados podemos crear una experiencia de usuario más rica y atractiva. Por ejemplo, al hacer clic en un "vegetable", una nueva capa se abre en el explorador que muestra la separación de semillas e instrucciones de siembra

Podemos ahorrar llamadas AJAX al tener los datos directamente en el HTML

7.1. Utilización de los data attributes

Algunos casos en los que PUEDEN ser utilizados:

- Para almacenar la altura inicial o la opacidad de un elemento que pudiera ser necesaria en los cálculos de animación JavaScript posterior
- Para almacenar los parámetros para una película de Flash que se carga a través de JavaScript
- Para almacenar los datos estadísticos de una web
- Para almacenar los datos acerca de la salud, munición o vida de un elemento en un juego JavaScript
- Para poder añadir subtítulos a un <video>
- ...

7.1. Utilización de los data attributes

Algunos casos en los que NO DEBEN ser utilizados:

- Si hay un atributo o elemento existente que es más adecuado: por ejemplo un “input required”
- No están pensados para ser usados públicamente, es decir, el software externo no debe interactuar con ellos (ni siquiera el navegador)
- La presencia/ausencia de un atributo de datos personalizado no se deben utilizar como una referencia para los estilos de CSS, esto se deberían marcar de una manera más accesible

Innova4B

7.1. Data attributes y JavaScript

Si quisiéramos recuperar o actualizar estos “data attributes” podríamos hacerlo a través de JavaScript utilizando los métodos getAttribute y setAttribute

```
<div id="strawberry-plant" data-fruit="12"></div>


<script>
	// "Getting" data-attributes using getAttribute
	var plant = document.getElementById("strawberry-plant");
	var fruitCount = plant.getAttribute("data-fruit"); // fruitCount = "12"

	// "Setting" data-attributes using setAttribute
	plant.setAttribute("data-fruit","7"); // Pesky birds
</script>
```

Innova4B

7.1. Data attributes y JavaScript

Este método funcionará en los navegadores modernos, pero no es la manera en la que los data attributes deben ser utilizados

Para lograr lo mismo debe accederse a la propiedad dataset de un elemento

```
<div id="sunflower" data-leaves="47" data-plant-height="2.4m"></div>

<script>
    var plant = document.getElementById("sunflower");
    var leaves = plant.dataset.leaves; // leaves = 47;

    var tallness = plant.dataset.plantHeight;
    plant.dataset.plantHeight = "3.6m";
</script>
```

Innova4B

7.1. Data attributes y JavaScript

Si en algún momento un atributo data- específico ya no es necesario, es posible eliminarlo por completo del elemento DOM

```
plant.dataset.leaves = null;
```

Los data attributes personalizados son una buena manera de simplificar el almacenamiento de datos de la aplicación en las páginas web

7.2. Data attributes y jQuery

jQuery permite acceder a los data attributes de una forma más eficiente en memoria que el método anterior (eso dicen), a través de la función “data()”

```
<section id="content" data-role="page" data-page-num="42">
    <!-- Imagine a bunch of page-type content here... -->
</section>

<script type="text/javascript">
    console.log($('content').data('role')); // Expect string "page"
    console.log($('content').data('pageNum'));    // Expect 42, an integer...!
    $('#content').data('newAttribute',34); // New attributte is hidden
</script>
```

Cómo utilizar data(): http://api.jquery.com/data/

Innova4B

7.3. Ejercicio práctico

A partir del siguiente HTML, realizar los siguiente JavaScript y jQuery:

```
<ul>
   <li class="user" data-name="Arkaitz Garro" data-city="Donostia"
      data-lang="es" data-food="Txuleta">Arkaitz Garro</li>
   <li class="user" data-name="John Doe" data-city="Boston"
      data-lang="en" data-food="Bacon">John Doe</li>
   <li class="user" data-name="Divya Resig" data-city="Tokyo"
      data-lang="jp" data-food="Sushi" data-delete="true">Divya Resig</li>
</ul>
```

- Obtener cada uno de los atributos data- de los elementos de la lista, y mostrarlos por consola.
- Modificar el idioma es por es_ES.
- Eliminar los elementos de la lista cuyo atributo data-delete sea true

Innova4B

8. Multimedia

Hasta hace no mucho tiempo, la tecnología Flash era el dominador indiscutible en el campo multimedia de la web

Gracias a esta tecnología es relativamente sencillo transmitir audio y vídeo a través de la red, y realizar animaciones que de otra manera sería imposible

Incluso se diseñaban páginas completamente en Flash

Prácticamente todos los navegadores tienen incorporado un plugin para la reproducción de archivos flash

Entonces, ¿por qué una necesidad de cambio?

8. Multimedia

Para incluir un elemento multimedia en un documento, se hacía uso del elemento <object>

Debido a la incompatibilidad entre navegadores, se hacía también necesario el uso del elemento <embed> y duplicar una serie de parámetros:

```
<object width="425" height="344">
   <param name="movie"
      value="http://www.youtube.com/v/9sEI1AUFJKw&hl=en_GB&fs=1&"></param>
   <param name="allowFullScreen" value="true"></param>
   <param name="allowscriptaccess" value="always"></param>
   <embed src="http://www.youtube.com/v/9sEI1AUFJKw&hl=en_GB&fs=1&"
      type="application/x-shockwave-flash" allowscriptaccess="always"
      allowfullscreen="true" width="425" height="344"></embed>
</object>
```

Innova4B

8. Multimedia

Tenemos varios inconvenientes:

- El navegador tiene que transmitir el vídeo a un plugin instalado en el navegador, y el usuario deberá tener instalada la versión correcta: ¿puede instalar? ¿tiene permisos para hacerlo? Deberá instalar el plugin antes de ver el vídeo... :(
- Los plugins pueden causar que el navegador o el sistema se comporte de manera inestable, algunos navegadores desactivan los plugins desactualizados (por inseguros) y usuarios sin conocimientos técnicos pueden percibir esa inseguridad... :(
- Y si mi dispositivo no dispone de plugin para ese contenido... :(

Innova4B

8.1. Video

Una de las mayores ventajas de HTML5 son los elementos multimedia <video> y <audio>

Están totalmente integrados en la web, no es necesario depender de software de terceros

Además, el elemento <video> puede personalizarse a través de estilos CSS: se puede cambiar su tamaño, animarlo con transiciones CSS...

Podemos manipularlos con total libertad, ya que pertenecen al estándar. Ya no se ejecutan en una caja negra, tenemos disponible un API:

http://www.w3schools.com/tags/ref_av_dom.asp

Innova4B

8.1. Video

<video></video>

Para hacer funcionar el vídeo en HTML, es suficiente con incluir la etiqueta:

```
<video src="movie.webm"> </video>
```

Sin embargo, lo único que se muestra es el primer fotograma de la película

No hemos dicho al navegador que inicie el vídeo, ni le hemos mostrado al usuario ningún tipo de control para reproducir o pausar el vídeo

Innova4B

8.1. Video

autoplay

Podemos indicar al navegador que reproduzca el vídeo de manera automática una vez se haya cargado la página

```
<video src="movie.webm" autoplay>
    <!-- Your fallback content here -->
</video>
```

No es una buena práctica, a muchos usuarios les parecerá una práctica muy intrusiva, sobre todo los usuarios de dispositivos móviles

Innova4B

8.1. Video

controls

Proporcionar controles es mejor que reproducir el vídeo de manera automática en la mayoría de los casos

```
<video src="movie.webm" controls>
<!-- Your fallback content here -->
</video>
```

Los navegadores muestran la interfaz de los controles de manera diferente, ya que la especificación no indica qué aspecto deben tener

Pero todos ellos muestran: reproducir/pausa, una barra de progreso y un control de volumen

Innova4B

8.1. Video

poster

Indica la imagen que el navegador debe mostrar mientras el vídeo se está descargando, o hasta que el usuario reproduce el vídeo

Si no se indica este atributo, el navegador muestra el primer fotograma del vídeo, que puede no ser representativo del vídeo que se va a reproducir

8.1. Video

muted

Permite que el elemento multimedia se reproduzca inicialmente sin sonido, lo que requiere una acción por parte del usuario para recuperar el volumen

loop

Indica que el vídeo se reproduce de nuevo una vez que ha finalizado su reproducción

8.1. Video

dimensiones

Los atributos height y width indican al navegador el tamaño del vídeo en pixels

Si no se indican estas medidas, el navegador utiliza las medidas definidas en el vídeo de origen, si están disponibles

Si no lo están, utiliza las medidas definidas en el fotograma poster, si están disponibles

Si ninguna de estas medidas está disponible, el ancho por defecto es de 300 pixels

8.1. Video

dimensiones

Si se especifica una de las dos medidas, el navegador ajusta la medida de la dimensión no proporcionada, conservando la proporción del vídeo

Si se especifican las dos medidas, pero no coinciden con la proporción del vídeo original, el vídeo no se deforma a estas nuevas dimensiones

8.1. Video

preload

Es posible indicar al navegador que comience la descarga del vídeo antes de que el usuario inicie su reproducción.

```
<video src="movie.webm" controls preload>
   <!-- Your fallback content here -->
</video>
```

Existen tres valores definidos para preload, pero preload es un consejo, no un comando

Innova4B

8.1. Video

preload

El navegador tomará una decisión en función del dispositivo, las condiciones de la red y otros factores:

- preload=auto: se sugiere al navegador que comience la descarga
- preload=none: se sugiere al navegador que no comience la descarga hasta que lo indique el usuario
- preload=metadata: se sugiere al navegador que cargue los metadatos (dimensiones, fotogramas, duración...), pero nada más

Si no indicamos uno en concreto, es el propio navegador el que decide qué hacer

Innova4B

8.1. Video

src

El atributo src indica la localización del recurso, que el navegador debe reproducir si el navegador soporta el codec o formato específico

Utilizar un único atributo src es únicamente útil y viable en entornos totalmente controlados: disponibilidad asegurada y codec controlado

Sin embargo, como no todos los navegadores pueden reproducir los mismos formatos, en entornos de producción debemos especificar más de una fuente de vídeo

8.2. Y ahora los codecs...

En los primeros borradores de HTML5, se indicaba que se debía de ofrecer soporte para al menos dos codecs multimedia: Ogg Vorbis para audio y Ogg Theora para vídeo

Sin embargo, estos requisitos fueron eliminados después de que Apple y Nokia se opusieran, de modo que la especificación no recomendase ningún codec en concreto

Esto ha creado una situación de fragmentación, con diferentes navegadores optando por diferentes formatos, basándose en sus ideologías o convicciones comerciales

Innova4B

8.2. Y ahora los codecs...

Actualmente, hay dos codecs principales: el nuevo formato WebM, construido sobre el formato VP8 que Google compró y ofrece de manera libre, y el formato MP4, que contiene el codec propietario H.264

	WEBM	MP4	OGV
Opera	Sí	No	Sí
Firefox	Sí	Sí	Sí
Chrome	Sí	No	Sí
IE9+	No	Sí	No
Safari	No	Sí	No

WebM funciona en IE9+ y Safari sólo si el usuario ha instalado los codec de manera manual.

La mejor solución en estos momentos es ofrecer tanto el formato libre WebM, como el propietario H.264

Innova4B

8.2. Y ahora los codecs...

<source>

Para poder ofrecer ambos formatos, primeramente debemos codificarlos por separado

Existen diversas herramientas y servicios on-line para realizar esta tarea

Uno de los software más conocidos sea Miro Video Converter que permite convertir los vídeos a muchos formatos y optimizar para diferentes tipos de dispositivos como iPhone, Android, PS2, etc.

Innova4B

8.2. Y ahora los codecs...

<source>

Después, es necesario indicar todas las localizaciones de estos formatos, para que sea el navegador el que decida que formato reproducir

Para ello, no podemos especificarlos todos dentro del atributo scr, por lo que tendremos que hacerlo de manera separada

```
<video controls>
   <source src="leverage-a-synergy.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
   <source src="leverage-a-synergy.webm" type='video/webm; codecs="vp8, vorbis"'>
   <p>Your browser doesn’t support video.
      Please download the video in <a href="leverage-a-synergy.webm">webM</a>
      or <a href="leverage-a-synergy.mp4">MP4</a> format. </p>
</video>
```

Innova4B

8.2. Y ahora los codecs...

media queries

Los ficheros de vídeo tienden a ser pesados, y enviar un vídeo en alta calidad a un dispositivo con un tamaño de pantalla reducido es algo totalmente ineficiente

HTML5 permite utilizar el atributo media en el elemento <source>, ofreciendo la misma funcionalidad que los Media Queries en CSS3. Por lo tanto, podemos consultar al navegador por el ancho de la pantalla, la relación de aspecto, colores, etc

```
<video controls>
    <source src="hi-res.mp4" media="(min-device-width: 800px)">
    <source src="lo-res.mp4">
</video>
```

Innova4B

8.3. El API multimedia

Los elementos multimedia <video> y <audio> ofrecen un API muy completo y fácil de utilizar

Los eventos y métodos de los elementos de audio y vídeo son exactamente los mismos, su única diferencia se da en los atributos

Atributos	Métodos	Eventos
error state	load()	loadstart
error	canPlayType(type)	progress
network state	play()	suspend
src	pause()	abort
currentSrc	addTrack(label, kind, language)	error
networkState		emptied
preload		stalled
buffered		play
ready state		pause

Innova4B

8.3. El API multimedia

Atributos	Métodos	Eventos
readyState		loadedmetadata
seeking		loadeddata
controls		waiting
controls		playing
volume		canplay
muted		canplaythrough
tracks		seeking
tracks		seeked
playback state		timeupdate
currentTime		ended
startTime		ratechange

Innova4B

8.3. El API multimedia

Atributos	Métodos	Eventos
muted		
paused		
defaultPlaybackRate		
playbackRate		
played		
seekable		
ended		
autoplay		
loop		
width [video only]		
height [video only]		
videoWidth [video only]		
videoHeight [video only]		
poster [video only]		

Innova4B

8.3. El API multimedia

Con este nuevo API, tenemos el control completo sobre los elementos multimedia

http://www.w3schools.com/tags/ref_av_dom.asp

Por ejemplo:

```
video.addEventListener('canplay', function(e) {
    this.volume = 0.4;
    this.currentTime = 10;
    this.play();
}, false);
```

Innova4B

8.4. Video Fullscreen

HTML5 establece un único elemento full-screen, que está pensado para imágenes, vídeo y juegos que utilizan el elemento canvas

Una vez que un elemento pasa a pantalla completa, aparece un mensaje de forma temporal para informar al usuario, así no genera confusión

Las principales propiedades, métodos y estilos son:

- element.requestFullScreen(): hace que un elemento individual pase a pantalla completa: document.getElementById("myvideo").requestFullScreen()
- document.cancelFullScreen(): sale del modo pantalla completa y vuelve a la vista del documento
- document.fullScreen: devuelve true si el navegador está en pantalla completa
- :full-screen: se trata de una pseudo-clase CSS que se aplica a un elemento cuando está en modo pantalla completa.

8.4. Video Fullscreen

Además, podemos modificar los estilos del elemento utilizando CSS:

```
#myelement {
   width: 500px;
}
#myelement:full-screen {
   width: 100%;
}
#myelement:full-screen img {
   width: 100%;
}
```

Innova4B

8.5. Audio

El elemento multimedia <audio> es muy similar en cuanto a funcionalidad al elemento video, de hecho el API es el mismo

La principal diferencia existe al indicar el atributo controls:

- Si lo especificamos, el elemento se mostrará en la página juntamente con los controles
- Si no lo hacemos, el audio se reproducirá, pero no existirá ningún elemento visual en el documento. El elemento existirá en el DOM y tendremos acceso completo a su API desde JavaScript

Para hacer funcionar el audio en HTML:

```
<audio src="audio.mp3">
</audio>
```

Innova4B

8.5. Audio

Pero con audio seguimos con los problemas de codecs...

	MP3	MP4	WAV	OGG
Opera	No	No	Sí	Sí
Firefox	No	No	Sí	Sí
Chrome	Sí	Sí	Sí	Sí
IE9+	Sí	Sí	No	No
Safari	Sí	Sí	Sí	No

La mejor solución en estos momentos es ofrecer tanto el formato libre OGG, como el propietario MP3:

```
<audio controls>
    <source src="audio.ogg" type="audio/ogg">
    <source src="audio.mp3" type="audio/mpeg">
</audio>
```

Innova4B

8.6. Ejercicio práctico

Crear un reproductor de vídeo que cumpla las siguientes características:

- Reproducir los vídeos independientemente del codec soportado por el navegador.
- Incluir controles de reproducción, pausa, parar, avanzar y retroceder 10 segundos, inicio y fin.
- Control de volumen y paso a pantalla completa.
- Un indicador de progreso de la reproducción.

Añadir una lista de reproducción que permita seleccionar un nuevo vídeo, y éste se reproduzca sin recargar la página.

http://www.arkaitzgarro.com/html5/capitulo-18.html#ej06

Innova4B

9. Canvas

Proporciona un API para dibujar líneas, formas, imágenes, texto, etc en 2D, sobre un “lienzo”

Este API ya está siendo utilizado de manera exhaustiva, en la creación de fondos interactivos, elementos de navegación, herramientas de dibujo, juegos o emuladores

Es uno de los elementos que cuenta con una de las mayores especificaciones dentro de HTML5

Innova4B

9. Canvas

Elementos básicos

```
<canvas id="tutorial" width="150" height="150"></canvas>
```

Los dos atributos width y height, son opcionales y pueden establecerse mediante las propiedades DOM

Si no se especifican, el tamaño inicial será de 300px por 150px de alto

Este elemento pueden modificarse utilizando CSS, pero las reglas afectarán al elemento, no a lo dibujado en el lienzo

Innova4B

9. Canvas

Elementos básicos

Todos los navegadores son compatibles con el elemento canvas, la diferencia entre ellos radica en qué funcionalidades del API han implementado

La manera de dibujar en el lienzo es a través de JavaScript

Lo primero es obtener el contexto de dibujado, que se utilizará para crear y manipular el contenido mostrado

9. Canvas

Elementos básicos

En el contexto de representación 2D, el canvas está inicialmente en blanco. Es necesario el acceso al DOM para usar las funciones de dibujo

```
var canvas = document.getElementById('tutorial');
var ctx = canvas.getContext('2d');
```

También existe un contexto para “3D” llamado WebGL

```
var gl = canvas.getContext("experimental-webgl");
```

Actualmente es un API de 2D pero proporcionar “shaders” para realizar un efecto de tres dimensiones

Innova4B

9. Canvas

```
<html>
 <head>
  <script type="application/javascript">
   window.onload = function() {
    var canvas = document.getElementById("canvas");
    if (canvas.getContext) {
     var ctx = canvas.getContext("2d");
     ctx.fillStyle = "rgb(200,0,0)";
     ctx.fillRect (10, 10, 55, 50);
     ctx.fillStyle = "rgba(0, 0, 200, 0.5)";
     ctx.fillRect (30, 30, 55, 50);
    }
   };
  </script>
 </head>
 <body><canvas id="canvas" width="150" height="150"></canvas></body>
</html>
```

Innova4B

9.1. Dibujar formas

Dentro del elemento canvas tenemos la “cuadrícula” o espacio de coordenadas

El punto de origen se coloca en la esquina superior izquierda, punto(0,0), y todos los elementos se colocan con relación a este origen

Normalmente, una unidad en la cuadrícula corresponde a un px en el lienzo

Solamente se admite una forma primitiva de dibujo: los rectángulos, el resto de las formas deberán crearse mediante la combinación de una o más funciones

9.1. Dibujar formas

Existen tres funciones que dibujan un rectángulo en el lienzo:

- fillRect(x,y,width,height): dibuja un rectángulo relleno
- strokeRect(x,y,width,height): dibuja un contorno rectangular
- clearRect(x,y,width,height): borra el área especificada y hace que sea totalmente transparente

Cada una de estas funciones tiene los mismos parámetros: x e y especifican las coordenadas en el lienzo, width es la anchura y height la altura

Las tres funciones de rectángulo dibujan inmediatamente en el lienzo

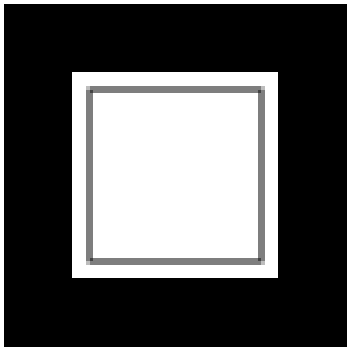
9.1. Dibujar formas

```
var canvas = document.getElementById('tutorial');
var ctx = canvas.getContext('2d');
ctx.fillRect(25,25,100,100);
ctx.clearRect(45,45,60,60);
ctx.strokeRect(50,50,50,50);
```

Innova4B

9.1. Dibujar formas

```
var canvas = document.getElementById('tutorial');
var ctx = canvas.getContext('2d');
ctx.fillRect(25,25,100,100);
ctx.clearRect(45,45,60,60);
ctx.strokeRect(50,50,50,50);
```



Innova4B

9.2. Trazar rutas

Las rutas son utilizadas para dibujar formas: líneas, curvas, polígonos, etc

Las rutas se almacenan como una lista de subrutas (líneas, arcos, etc.) que, en conjunto, forman una figura. Cada vez que se llama al método "**beginPath**", la lista se pone a cero y podemos empezar a dibujar nuevas formas

El paso final sería llamar al método "**closePath**": este método intenta cerrar la forma trazando una línea recta desde el punto actual hasta el inicial

9.2. Trazar rutas

```
var canvas  = document.getElementById('tutorial');
var context = canvas.getContext('2d');

context.beginPath();
//... path drawing operations
context.closePath();
```

El siguiente paso es dibujar la forma como tal utilizando las funciones de dibujado de líneas y arcos

Innova4B

9.2. Trazar rutas

moveTo

La función moveTo, no dibuja nada. Es utilizada para colocar el punto de partida en otro lugar o para dibujar rutas inconexas

```
ctx.beginPath();
ctx.moveTo(25,25); // Punto inicial para dibujar
ctx.lineTo(105,25);
ctx.lineTo(25,105);
ctx.closePath();
```

Innova4B

9.2. Trazar rutas

lineTo

Este método toma dos argumentos x e y, que son las coordenadas del punto final de la línea. El punto de partida depende de las rutas anteriores

```
ctx.beginPath();
ctx.moveTo(25,25); // Posición inicial
ctx.lineTo(105,25); // Linea hasta la posición 105, 25
ctx.lineTo(25,105); // Línea hasta la posición 25, 105
ctx.closePath(); // Cerrar la figura
ctx.fill();
```

Innova4B

9.2. Trazar rutas

lineTo

```
ctx.beginPath();
ctx.moveTo(125,125);
ctx.lineTo(125,45);
ctx.lineTo(45,125);
ctx.closePath();
ctx.stroke();
```

“stroke” se utiliza para dibujar una forma con contorno

“fill” se utiliza para pintar una forma sólida

Innova4B

9.2. Trazar rutas

arc

Para dibujar arcos o círculos se utiliza el método arc (la especificación también describe el método arcTo)

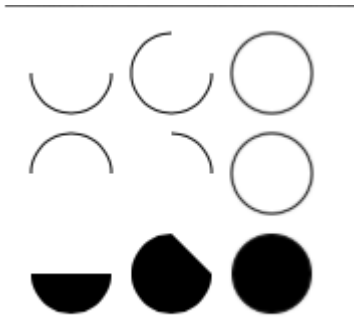
```
ctx.arc(x,y,radius,startAngle,endAngle, anticlockwise);
```

Este método toma cinco parámetros: x e y, el radio, startAngle y endAngle (puntos de inicio y final del arco en radianes) y anticlockwise (booleano)

9.2. Trazar rutas

arc

```
for(var i=0;i<4;i++){
   for(var j=0;j<3;j++){
      ctx.beginPath();
      var x            = 25+j*50;      // coordenada x
      var y            = 25+i*50;      // coordenada y
      var radius       = 20;           // radio del arco
      var startAngle   = 0;            // punto inicial del círculo
      var endAngle     = Math.PI+(Math.PI*j)/2; // punto final
      var anticlockwise  = i%2==0 ? false : true;
      ctx.arc(x,y,radius,startAngle,endAngle, anticlockwise);
      if (i>1) ctx.fill();
      else ctx.stroke();
   }
}
```



Innova4B

9.2. Trazar rutas

quadraticCurveTo()

http://www.w3schools.com/tags/canvas_quadraticcurveto.asp

bezierCurveTo()

http://www.w3schools.com/tags/canvas_beziercurveto.asp

arc, bezier y quadratic utilizan radianes, pero si preferimos trabajar en grados, es necesario convertirlo a radianes:

```
var radians = degrees * Math.PI / 180;
```

Innova4B

9.3. Colores

Si queremos aplicar colores a una forma, hay dos características importantes que podemos utilizar: fillStyle y strokeStyle.

```
fillStyle = color;      // Relleno de la figura
strokeStyle = color; //Contorno de la figura
```

El color puede ser una cadena que representa un valor de color CSS, un objeto degradado o un objeto modelo

```
ctx.fillStyle = "orange";
ctx.fillStyle = "#FFA500";
ctx.fillStyle = "rgb(255,165,0)";
ctx.fillStyle = "rgba(255,165,0,1)";
```

9.3. Colores

Ejemplo de fillStyle

```
function draw() {
    for (var i=0;i<6;i++){
        for (var j=0;j<6;j++){
            ctx.fillStyle = 'rgb(' + Math.floor(255-42.5*i) + ','
                + Math.floor(255-42.5*j) + ',0)';
            ctx.fillRect(j*25,i*25,25,25);
        }
    }
}
```

Innova4B

9.3. Colores

Ejemplo de strokeStyle

```
function draw() {
    for (var i=0;i<6;i++){
        for (var j=0;j<6;j++){
            ctx.strokeStyle = 'rgb(0,' + Math.floor(255-42.5*i)
                + ',' +  ath.floor(255-42.5*j) + ')';
            ctx.beginPath();
            ctx.arc(12.5+j*25,12.5+i*25,10,0,Math.PI*2,true);
            ctx.stroke();
        }
    }
}
```

Innova4B

9.4. Degradados y patrones

Los degradados funcionan de una manera similar a los definidos en CSS3, donde se especifican el inicio y los pasos de color para el degradado.

Los patrones, permiten definir una imagen como origen y especificar el patrón de repetido, de nuevo de manera similar a como se realizaba con la propiedad background-image de CSS

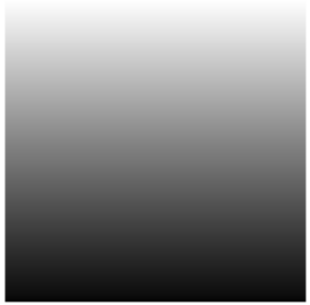
Lo que hace interesante al método createPattern es que como origen podemos utilizar una imagen, un canvas o un elemento de vídeo

9.4. Degradados y patrones

Los argumentos de createLinearGradient son el punto de inicio del degradado (x1 e y1) y el punto final del degradado (x2 e y2)

```
var canvas = document.getElementById('tutorial');
var ctx = canvas.getContext('2d');
var gradient = ctx.createLinearGradient(0, 0, 0, canvas.height);

gradient.addColorStop(0, '#fff');
gradient.addColorStop(1, '#000');
ctx.fillStyle = gradient;
ctx.fillRect(0, 0, canvas.width, canvas.height);
```



Innova4B

9.4. Degradados y patrones

Los degradados radiales son muy similares, con la excepción que definimos el radio después de cada coordenada:

```
gradient = ctx.createRadialGradient(canvas.width/2,
                                    canvas.height/2,
                                    0,
                                    canvas.width/2,
                                    canvas.height/2,
                                    150);
```

El primer punto del degradado se define en el centro del canvas, con radio cero y el siguiente punto se define con un radio de 150px pero su origen es el mismo, lo que produce un degradado circular

9.5. Transparencias

Podemos dibujar formas semitransparentes mediante la propiedad globalAlpha

globalAlpha = transparency value

```
ctx.globalAlpha = 0.2;
// Dibujar círculos semitransparentes
for (var i=0;i<7;i++){
        ctx.beginPath();
        ctx.arc(75,75,10+10*i,0,Math.PI*2,true);
        ctx.fill();
}
```

Innova4B

9.6. Transformaciones

Podemos definir algunas transformaciones como rotación, escalado, transformación y traslación (similares a las conocidas de CSS3)

ctx.translate(x, y);

Traslada el centro de coordenadas desde su posición por defecto (0, 0) a la posición indicada

Innova4B

9.6. Transformaciones

ctx.rotate(angle);

Inicia la rotación desde su posición por defecto (0,0)

Si se rota el canvas desde esta posición, el contenido podría desaparecer por los límites del lienzo, por lo que puede necesario definir un nuevo origen para la rotación

Sólo precisa tomar el ángulo de rotación que se aplicará al marco. Este parámetro es una rotación dextrógira medida en radianes

9.6. Transformaciones

ctx.scale(x, y);

Aumentar o disminuir las unidades del tamaño de nuestro marco

x e y, definen el factor de escala en la dirección horizontal y vertical respectivamente

Los valores menores que 1.0 reducen el tamaño de la unidad y los valores mayores que 1.0 aumentan el tamaño de la unidad

Por defecto, una unidad en el área de trabajo equivale exactamente a un píxel