

Contenido

1	SVG	3
1.1	Círculo	3
1.2	Rectángulo	3
1.3	ELIPSE	4
1.4	LINEA	4
1.5	POLÍGONO	4
1.6	POLILÍNEAS	5
1.7	DIBUJAR trazo	5
1.8	TEXTO	5
2	Animaciones	5
2.1	La regla @keyframes	5
3	Transformaciones	9
3.1	Transformaciones 3D	11
4	Las imágenes	13
4.1	Tipos	13
4.2	Aplicaciones	14
4.3	Formatos	14
4.4	Ventajas y desventajas	14
4.5	Conceptos sobre imágenes	15
4.5.1	Profundidad de color	15
4.5.2	Tamaño de una imagen de pixeles en la memoria:	17
4.5.3	Compresión	17
4.5.4	La rejilla de pixeles	18
4.5.5	Anti-alias	18
5	Formatos de archivos de imagen.	19
5.1	TIFF (Tagged Image File Format)	19
5.2	GIF (Graphic Interchange Format, Formato de intercambio de gráficos)	20
5.3	PNG (Gráficos de Red Portátiles)	20
5.4	JPG	21
5.5	SVG	21
6	Optimización de imágenes para la web	22
6.1	Recomendaciones de optimización	22
6.2	Herramientas de optimización	23
7	Conceptos básicos de vídeo digital	23

8	Formatos de archivos de video	24
8.1	AVI (Audio Video Interleaved = Audio y Video Intercalado)	24
8.2	MPEG (Moving Pictures Expert Group = Grupo de Expertos de Películas)	24
8.3	MOV (http://www.apple.com/es/quicktime/)	24
8.4	WMV (http://www.microsoft.com/windows/windowsmedia/es/)	25
8.5	FLV (http://www.adobe.com)	25
8.6	Otros formatos	25
9	¿Qué es el streaming?	26
10	Conceptos básicos del sonido digital	27
10.1	Frecuencia.	27
10.2	Tasa de muestreo (sample rate).	27
10.3	Resolución (bit resolution)	27
10.4	Velocidad de transmisión (bitrate)	27
10.5	Códec.	27
10.6	Decibelio.	28
11	Formatos de audio	28
11.1	El formato MP3 (MPEG 1 Layer 3)	28
11.2	Ogg	28
11.3	Real Audio	29
11.4	Windows Media Audio (wma)	29
11.5	Wav o wave (WAVEform audio file format):	29
11.6	Free Lossless Audio Codec (FLAC)	30
11.7	Formato MIDI	30
12	Optimización de archivos de video y audio	30
13	Insertar elementos multimedia en la web	31
13.1	Insertar audio en una web	31
13.2	Consideraciones para el uso de audio en un sitio web	34
13.3	Insertar vídeo en una web	34

1 SVG

SVG son las siglas de Scalable Vector Graphics. SVG define gráficos basados en vectores en formato XML.

Código SVG:

- Una imagen SVG comienza con un elemento <svg>
- Los atributos de ancho y alto del elemento <svg> definen el ancho y alto de la imagen SVG
- La etiqueta de cierre </svg> cierra la imagen SVG
- Como SVG está escrito en XML, todos los elementos deben estar bien cerrados

1.1 Círculo

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1>My first SVG</h1>
```

```
<svg width="100" height="100">
```

```
<circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />
```

Sorry, your browser does not support inline SVG.

```
</svg>
```

```
</body>
```

```
</html>
```

- Los atributos cx y cy definen las coordenadas xey del centro del círculo. Si se omiten cx y cy, el centro del círculo se establece en (0,0)
- El atributo r define el radio del círculo

1.2 Rectángulo

```
<rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
```

- Los atributos de ancho y alto del elemento <rect> definen la altura y el ancho del rectángulo
- El atributo de estilo se usa para definir las propiedades de CSS para el rectángulo
- La propiedad de relleno CSS define el color de relleno del rectángulo
- La propiedad CSS stroke-width define el ancho del borde del rectángulo
- La propiedad de trazo CSS define el color del borde del rectángulo

```
<rect x="50" y="20" width="150" height="150"
```

```
style="fill:blue;stroke:pink;stroke-width:5;fill-opacity:0.1;stroke-opacity:0.9" />
```

- El atributo x define la posición izquierda del rectángulo (ej. X = "50" coloca el rectángulo 50 px desde el margen izquierdo)
- El atributo y define la posición superior del rectángulo (por ejemplo, y = "20" coloca el rectángulo 20 px desde el margen superior)
- La propiedad CSS stroke-opacity define la opacidad del color de trazo (rango legal: 0 a 1)

Si sólo ponemos opacity afecta a los dos elementos, borde y relleno.

```
<rect x="50" y="20" rx="20" ry="20" width="150" height="150" style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />
```

- Los atributos rx y the ry redondean las esquinas del rectángulo

1.3 ELIPSE

```
<ellipse cx="200" cy="80" rx="100" ry="50" style="fill:yellow;stroke:purple;stroke-width:2" />
```

- El atributo cx define la coordenada x del centro de la elipse
- El atributo cy define la coordenada y del centro de la elipse
- El atributo rx define el radio horizontal
- El atributo ry define el radio vertical

1.4 LINEA

```
<line x1="0" y1="0" x2="200" y2="200" style="stroke:rgb(255,0,0);stroke-width:2" />
```

- El atributo x1 define el inicio de la línea en el eje x
- El atributo y1 define el inicio de la línea en el eje y
- El atributo x2 define el final de la línea en el eje x
- El atributo y2 define el final de la línea en el eje y

1.5 POLÍGONO

Crea un polígono con tres lados

```
<polygon points="200,10 250,190 160,210" style="fill:lime;stroke:purple;stroke-width:1" />
```

El atributo de puntos define las coordenadas x e y para cada esquina del polígono

Crea un polígono con cuatro lados:

```
<polygon points="220,10 300,210 170,250 123,234" style="fill:lime;stroke:purple;stroke-width:1" />
```

Crea una estrella

```
<polygon points="100,10 40,198 190,78 10,78 160,198" style="fill:lime;stroke:purple;stroke-width:5;fill-rule:nonzero;" />
```

cambiando la propiedad de regla de relleno a "evenodd" no se rellena completamente el color de la estrella

1.6 POLILÍNEAS

El elemento <polyline> se usa para crear cualquier forma que consista solo en líneas rectas:

```
<polyline points="20,20 40,25 60,40 80,120 120,140 200,180" style="fill:none;stroke:black;stroke-width:3" />
```

1.7 DIBUJAR trazo

El elemento <ruta> se usa para definir una ruta. Path - <ruta de acceso>

Los siguientes comandos están disponibles para datos de ruta:

- M = moveto
- L = lineto
- H = línea horizontal a
- V = línea vertical a
- C = curvatura
- S = curva suave
- Q = curva cuadrática de Bézier
- T = curva Bézier cuadrada suave
- A = arco elíptico
- Z = closepath

Todos los comandos anteriores también se pueden expresar con letras minúsculas. Las letras mayúsculas significan absolutamente posicionadas, los casos en minúsculas significan relativamente posicionados.

```
<path d="M150 0 L75 200 L225 200 Z" />
```

1.8 TEXTO

```
<text x="0" y="15" fill="red" transform="rotate(30 20,40)">I love SVG</text>
```

Escribe el texto y lo rota

2 Animaciones

Una animación permite que un elemento cambie gradualmente de un estilo a otro. Puede cambiar tantas propiedades CSS que desee, tantas veces como desee.

Para usar animación CSS3, primero debe especificar algunos fotogramas clave para la animación. Los fotogramas clave mantienen qué estilos tendrá el elemento en determinados momentos.

2.1 La regla @keyframes

Cuando especifica estilos CSS dentro de la @keyframes regla, la animación cambiará gradualmente del estilo actual al estilo nuevo en determinados momentos.

Para que una animación funcione, debe vincular la animación a un elemento.

```
<!DOCTYPE html>

<html>

<head>

<style>

div {
    width: 100px;
    height: 100px;
    background-color: red;
    -webkit-animation-name: example; /* Safari 4.0 - 8.0 */
    -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
    animation-name: example;
    animation-duration: 10s;
}

/* Safari 4.0 - 8.0 */
@-webkit-keyframes example {
    from {background-color: red;}
    to {background-color: yellow;}
}

/* Standard syntax */
@keyframes example {
    from {background-color: red;}
    to {background-color: yellow;}
}

</style>

</head>

<body>

<div></div>
```

```
</body>
</html>
```

El siguiente ejemplo cambiará el color de fondo del elemento <div> cuando la animación esté completa al 25%, completa al 50% y nuevamente cuando la animación esté completa al 100%:

```
@keyframes example {
  0% {background-color: red;}
  25% {background-color: yellow;}
  50% {background-color: blue;}
  100% {background-color: green;}
}
```

Cambia tanto el color como la posición

```
@keyframes example {
  0% {background-color:red; left:0px; top:0px;}
  25% {background-color:yellow; left:200px; top:0px;}
  50% {background-color:blue; left:200px; top:200px;}
  75% {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
```

animation-delay

Retrasa el comienzo de la animación 2 segundos. Los valores negativos también están permitidos. Si usa valores negativos, la animación comenzará como si ya hubiera estado reproduciéndose durante N segundos.

```
animation-delay: 2s;
```

animation-iteration-count

Establecer cuántas veces debe ejecutarse una animación, usa el valor "infinito" para que la animación continúe para siempre:

```
animation-iteration-count: 3;
```

animation-direction

La propiedad de dirección de animación puede tener los siguientes valores:

- normal- La animación se reproduce como normal (adelante). Esto es predeterminado

- reverse - La animación se juega en dirección inversa (hacia atrás)
- alternate- La animación se reproduce hacia adelante primero, luego hacia atrás
- alternate-reverse - La animación se reproduce hacia atrás primero, y luego hacia adelante

animation-direction: reverse;

animation-timing-function

La animation-timing-function propiedad especifica la curva de velocidad de la animación.

La propiedad animation-timing-function puede tener los siguientes valores:

- ease - Especifica una animación con un inicio lento, luego rápido, luego termina lentamente (esto es por defecto)
- linear - Especifica una animación con la misma velocidad de principio a fin
- ease-in - Especifica una animación con un inicio lento
- ease-out - Especifica una animación con un final lento
- ease-in-out - Especifica una animación con un inicio y un final lentos
- cubic-bezier(n,n,n,n) - Le permite definir sus propios valores en una función cúbica-bezier

animation-fill-mode

La animation-fill-mode propiedad especifica un estilo para el elemento de destino cuando la animación no se está reproduciendo (antes de que comience, después de que termine o ambos).

La propiedad animation-fill-mode puede tener los siguientes valores:

- none- Valor por defecto. La animación no aplicará ningún estilo al elemento antes o después de ejecutarlo
- forwards - El elemento retendrá los valores de estilo establecidos por el último fotograma clave (depende de la dirección de animación y del recuento de iteración de animación)
- backwards - El elemento obtendrá los valores de estilo que establece el primer fotograma clave (depende de la dirección de la animación) y lo conservará durante el período de demora de animación
- both - La animación seguirá las reglas tanto para adelante como para atrás, extendiendo las propiedades de animación en ambas direcciones

El siguiente ejemplo permite que el elemento <div> conserve los valores de estilo del último fotograma clave cuando finaliza la animación:

animation-fill-mode: forwards;

Ejemplo final

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```



```

<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    position: relative;
    /* animation: myfirst 5s linear 2s infinite alternate;*/
    animation-name: example;
    animation-duration: 5s;
    animation-timing-function: linear;
    animation-delay: 2s;
    animation-iteration-count: infinite;
    animation-direction: alternate;
}
/* Standard syntax */
@keyframes example {
    0% {background-color:red; left:0px; top:0px;}
    25% {background-color:yellow; left:200px; top:0px;}
    50% {background-color:blue; left:200px; top:200px;}
    75% {background-color:green; left:0px; top:200px;}
    100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

<div></div>
</body>
</html>

```

3 Transformaciones

El **translate()** método mueve un elemento desde su posición actual (de acuerdo con los parámetros dados para el eje X y el eje Y)

```
<!DOCTYPE html>

<html>

<head>

<style>

div {
    width: 300px;
    height: 100px;
    background-color: yellow;
    border: 1px solid black;
    -ms-transform: translate(50px,100px); /* IE 9 */
    -webkit-transform: translate(50px,100px); /* Safari */
    transform: translate(50px,100px); /* Standard syntax */
}

</style>

</head>

<body>

<h1>The translate() Method</h1>

<p>The translate() method moves an element from its current position:</p>

<div>

This div element is moved 50 pixels to the right, and 100 pixels down from its current position.

</div>

</body>

</html>
```

El **rotate()** método gira un elemento en sentido horario o antihorario según un grado dado.

```
div {
```

```

-ms-transform: rotate(20deg); /* IE 9 */
-webkit-transform: rotate(20deg); /* Safari */
transform: rotate(20deg);
}

```

El **scale()** método aumenta o disminuye el tamaño de un elemento (de acuerdo con los parámetros dados para el ancho y la altura).

El siguiente ejemplo aumenta el elemento <div> para ser dos veces su ancho original y tres veces su altura original:

```

div {
    -ms-transform: scale(2, 3); /* IE 9 */
    -webkit-transform: scale(2, 3); /* Safari */
    transform: scale(2, 3);
}

```

El **skewX()** método sesga un elemento a lo largo del eje X en el ángulo dado.

El siguiente ejemplo sesga el elemento <div> 20 grados a lo largo del eje X:

```

div {
    -ms-transform: skewX(20deg); /* IE 9 */
    -webkit-transform: skewX(20deg); /* Safari */
    transform: skewX(20deg);
}

```

El **skewY()** método sesga un elemento a lo largo del eje Y en el ángulo dado.

El **skew(x,y)** método sesga un elemento a lo largo de los ejes X e Y por los ángulos dados. Si el segundo parámetro no está especificado, tiene un valor cero.

El **matrix()** método combina todos los métodos de transformación 2D en uno.

El método **matrix ()** toma seis parámetros, que contienen funciones matemáticas, que le permiten rotar, escalar, mover (traducir) y sesgar elementos.

Los parámetros son los siguientes: **matrix (scaleX (), skewY (), skewX (), scaleY (), translateX (), translateY ())**

```

div {
    -ms-transform: matrix(1, -0.3, 0, 1, 0, 0); /* IE 9 */
    -webkit-transform: matrix(1, -0.3, 0, 1, 0, 0); /* Safari */
    transform: matrix(1, -0.3, 0, 1, 0, 0);
}

```

3.1 Transformaciones 3D

El **rotateX()** método gira un elemento alrededor de su eje X en un grado dado:

div {

 -webkit-transform: rotateX(150deg); /* Safari */

 transform: rotateX(150deg);

}

El **rotateY()** método gira un elemento alrededor de su eje Y en un grado dado:

El **rotateZ()** método gira un elemento alrededor de su eje Z en un grado dado:

4 Las imágenes

La elección y el diseño de las imágenes es una de las decisiones más importantes en el diseño web, deben ser de calidad, deben pesar poco, el conjunto debe ser homogéneo, la información que aportan debe ser accesible, deben ser responsivas, agradables, funcionales, entendibles, deben estar en consonancia con los contenidos de la web, no debemos vulnerar los derechos de autor. etc. Etc. Etc.

Pero...¿de dónde sacamos las imágenes?. Pues tenemos varias posibilidades, podemos fabricarlas nosotros, podemos copiarlas, podemos copiarlas y adaptarlas, podemos utilizar la iconografía de un framework y también podemos comprarlas.

Una de las principales decisiones a la hora de incluir imágenes en una web es elegir el formato correcto para cada tipo de imagen de manera que se logre una correcta relación entre la calidad visual de la misma y su peso en bytes. Básicamente, existen distintos tipos de imágenes: las que son mapas de bits, las vectoriales y las imágenes animadas. Estas últimas las trataremos más adelante en la sección dedicada a las animaciones. Esta sección se centra en los otros dos grupos de imágenes (vectoriales y mapas de bits).



Píxeles

Los píxeles son la unidad mínima de las imágenes de **mapas de bits**, que también son llamadas imágenes **raster** o **bitmaps**.

Un mapa de bits es una matriz cartesiana (bidimensional) de píxeles, con coordenadas verticales y horizontales que determinan la posición de un pixel en la imagen.

Vectores

Los vectores son la **descripción geométrica** (matemática) de una imagen.

Por ejemplo, para describir todos los puntos del perímetro de un círculo sólo es necesaria su fórmula ($x^2 + y^2 = R$). Modificando la variable R, se obtienen círculos de todos los radios posibles.

Un ejemplo ilustrativo de la diferencia entre imágenes de píxeles y de vectores es el siguiente: Hay dos maneras de enviar una tarta de cumpleaños: Enviarla ya cocinada en una caja, o enviar la receta, de manera que el destinatario la pueda cocinar siguiendo los pasos contenidos en ella

Cuando yo envío un archivo de mapa de píxeles estoy enviando **la tarta entera**. Cuando envío un archivo de vectores, estoy enviando **la receta**, pues los vectores siempre se pueden "cocinar" para obtener un mapa de píxeles.

4.1 Tipos

Un pixel puede requerir mayor o menor cantidad de memoria para ser almacenado, y de acuerdo a este valor (llamado la *profundidad de un pixel*) la imagen podrá desplegar una mayor o menor cantidad de colores.

Existen diferentes tipos de vectores o, lo que es igual, diferentes métodos matemáticos de describir una imagen. Por ejemplo, una curva es un primitivo importante de la información vectorial.

4.2 Aplicaciones

Utilizados en software de captura, retoque o composición de imágenes reales (video o imagen fija)

- Photoshop
- AfterEffects
- Premiere

Son utilizados generalmente en los programas de dibujo técnico, o modelamiento tridimensional.

- Illustrator, Freehand, CorelDraw
- Flash
- 3D Studio, InfiniD, Maya...

La conversión de una imagen de pixeles a una imagen vector se llama *vectorización*.

La conversión de una imagen de vectores a una imagen de pixeles se llama render

4.3 Formatos

Los formatos de imágenes de pixeles no dependen tanto de la aplicación con la que fueron creados. Casi todas las aplicaciones que procesan imágenes pixeles pueden leer diversos formatos.

BMP (windows), PSD (photoshop), JPEG, GIF, TIF, TGA.

Los formatos de vectores están muy ligados al tipo de software que se utiliza para crearlos o interpretarlos, y aunque existen maneras de convertir de un formato a otro, siempre existe pérdida de información en dicha conversión.

AI (illustrator), CDR (coreldraw), DXF (autocad)

Existen formatos que almacenan información vector y de pixeles en un sólo archivo. Algunos formatos que pueden contener información mezclada son: PICT (macintosh), WMF (windows), EPS (encapsulated postscript, empleado para impresión en papel) y PDF (formato portátil de adobe)

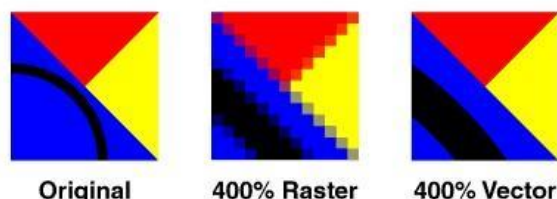
4.4 Ventajas y desventajas

Los pixeles requieren menos operaciones del procesador para ser decodificados.

Requieren mayor cantidad de operaciones del procesador para ser decodificados y desplegados en la pantalla, ya que siempre se convierten finalmente en una imagen de pixeles a través de un proceso de render

Generalmente, por almacenar cada punto de la imagen, ocupan mayor espacio en memoria, y requieren un tiempo mayor de transferencia a través de las redes.

Almacenan en pocos bytes información compleja, de manera que se transfieren rápidamente a través de las redes.



Tienen una resolución fija, determinada por la cantidad de píxeles que se hayan almacenado en el archivo. Cualquier operación de reducción o ampliación de la cantidad de píxeles, redundará en una pérdida de información o aliasing

Son independientes de la resolución, es decir, con la descripción geométrica almacenada se pueden generar imágenes de diversos tamaños de píxeles, tan sólo ampliando la escala del vector.

Son buenos para almacenar texturas complejas.

No son buenos para almacenar texturas, sino más bien áreas de color plano.

Tienen una resolución fija, determinada por la cantidad de píxeles que se hayan almacenado en el archivo. Cualquier operación de reducción o ampliación de la cantidad de píxeles, redundará en una pérdida de información o aliasing

Son independientes de la resolución, es decir, con la descripción geométrica almacenada se pueden generar imágenes de diversos tamaños de píxeles, tan sólo ampliando la escala del vector.

Son buenos para almacenar texturas complejas.

No son buenos para almacenar texturas, sino más bien áreas de color plano.

Tienen una resolución fija, determinada por la cantidad de píxeles que se hayan almacenado en el archivo. Cualquier operación de reducción o ampliación de la cantidad de píxeles, redundará en una pérdida de información o aliasing

Son independientes de la resolución, es decir, con la descripción geométrica almacenada se pueden generar imágenes de diversos tamaños de píxeles, tan sólo ampliando la escala del vector.

Son buenos para almacenar texturas complejas.

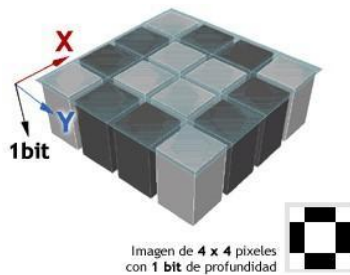
No son buenos para almacenar texturas, sino más bien áreas de color plano.

4.5 Conceptos sobre imágenes

Algunos conceptos importantes son los siguientes:

4.5.1 Profundidad de color


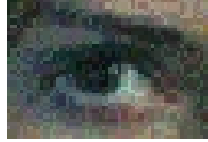


Si las coordenadas del píxel determinan su posición en la imagen, la profundidad es la cantidad de memoria requerida para almacenar su color.



La profundidad de un pixel **no se debe confundir** con la posición de ese pixel en un eje Z imaginario (considerando los ejes X y Y como su posición en el plano). Esta "profundidad" sólo representa cantidad de información, no posición espacial.


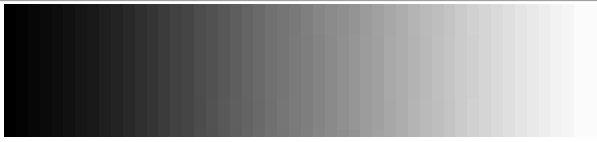
La unidad mínima de almacenamiento en la memoria de un computador es 1 bit, el cual puede tomar solamente dos valores: **1** ó **0**. Por ello, los computadores, en lugar de usar el sistema decimal de numeración que utilizamos en la vida cotidiana, utilizan el sistema binario.

Esto quiere decir que para calcular la cantidad de colores que puede contener una imagen de píxeles, debemos elevar el número **2** a la cantidad de bits utilizados para almacenar el color en un pixel. Los ejemplos de esta fórmula se encuentran aquí abajo.

Imagen	Profundidad	Cantidad de colores	Formatos más utilizados
	1 bit	$2^1 = 2$ colores	GIF, BMP
	8 bits (1 Byte)	$2^8 = 256$ colores	GIF, BMP
	16 bits (2 bytes)	$2^{16} = 65536$ colores	BMP, TGA, TIF, PSD, PICT
	24 bits (1 byte Rojos 1 byte Azules 1 byte Verdes)	$2^{24} = 16'777.216$ colores	BMP, TGA, TIF, PSD, PICT, JPG


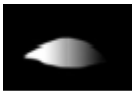
Las imágenes de 8 bits son un caso especial, puesto que su color se define por índices (color "indexado") o números almacenados en una tabla de color.

La diferencia entre una imagen de 16 y 24 bits sólo es notoria en colores suavemente degradados. En la imagen de 16 bits se ven mucho más las bandas de color, debido a la falta de colores para representar un degradado continuo.

16 bits	
24 bits	

Imágenes de 32 bits

Existen además imágenes con profundidad de pixel de 32 bits. Los 8 bits (1 byte) adicionales de profundidad sobre las imágenes de 24 bits, le permiten almacenar la transparencia de la imagen. Este byte adicional es generalmente llamado **máscara** o **canal alfa**, y almacena, en una imagen de 256 niveles de grises, diferentes valores de transparencia.

	Imagen: 24 bits (3 bytes de color)	$2^{24} = 16'777.216$ colores
	Máscara: 8 bits (1 byte de transparencia)	$2^8 = 256$ niveles de transparencia
	Imagen resultante sobre fondo verde	$2^{32} = \text{color} + \text{transparencia}$

Normalmente, un pixel blanco en la máscara hace que el pixel correspondiente en la imagen se muestre completamente opaco (no deja ver el fondo) y un pixel negro en la máscara hace al pixel de la imagen completamente transparente (deja ver el fondo). Los grises logran transparencias intermedias.

4.5.2 Tamaño de una imagen de pixeles en la memoria:

Se puede calcular el tamaño de cualquier archivo de imagen de pixeles multiplicando la cantidad de pixeles horizontales por la cantidad de pixeles verticales, y luego multiplicar ese producto por la profundidad, así:

Tamaño en pixeles	Profundidad de pixel	Tamaño del archivo			
		bits	bytes	Kbytes	Mbytes
640 x 480	x 1 bit	= 307.200	= 38.400	= 37.5	= 0.036
640 x 480	x 8 bits	= 2'457.600	= 307.200	= 300	= 0.292
640 x 480	x 24 bits	= 7'372.800	= 921.600	= 900	= 0.878
640 x 480	x 32 bits	= 9'830.400	= 1'228.800	= 1200	= 1.171

4.5.3 Compresión

La compresión es un término importantísimo en el almacenamiento y transferencia digital de la información. Compresión es cualquier tipo de proceso (o *algoritmo*) que reduzca la cantidad de información contenida en un archivo, ya sea perdiendo o no parte de la información original.

De hecho, convertir una imagen de 24 bits (un archivo TGA, por ejemplo) a una imagen de 8 bits (un GIF, p.ej.) es un proceso de compresión, puesto que estoy reduciendo la cantidad de memoria que necesito para representar cada pixel de la imagen original.

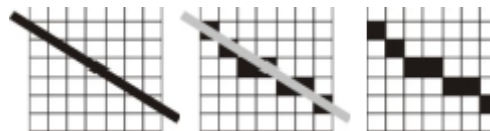
Una compresión sin pérdidas devuelve la imagen descomprimida exactamente igual a la original. Por el contrario, la compresión con pérdidas acepta alguna degradación en la imagen de cara a una mayor compresión.

Aliasing es la palabra acuñada por la teoría de la información para denotar la pérdida de datos después de un proceso o transferencia de información. En la imagen digital el aliasing redundante en la pérdida de calidad, en imágenes digitales de poca profundidad de pixel, genera algunas veces el efecto de *escalera* o *línea dentada*, que se ve sobre todo en las líneas curvas u oblicuas.

4.5.4 La rejilla de pixeles

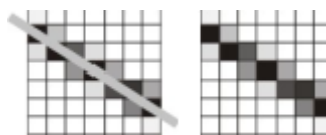
En la imagen digital, por lo menos en los sistemas más utilizados hoy en día, la imagen se divide en una rejilla *discreta* (discontinua) de pixeles, de resolución limitada. Si quisiéramos representar una línea continua con una imagen de 7 x 7 pixeles, ocurriría lo siguiente:

En la imagen de la izquierda, tenemos la línea original. Un pixel sólo puede ser negro o blanco (considerando 1 solo bit de . En este caso, no se puede representar un pixel cubierto parcialmente por la línea. Debemos, pues, eliminar parte de la información original, para obtener el resultado de la imagen de la derecha. El resultado: una imagen con un aliasing muy alto.

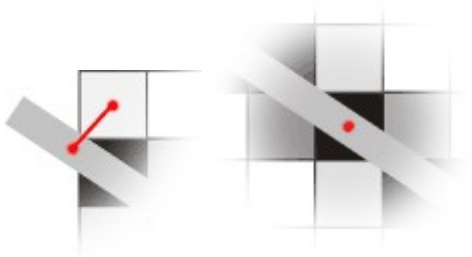


4.5.5 Anti-alias

Supongamos que tenemos una mayor profundidad de pixel para representar la misma línea, digamos 8 bits (256 grises, en este caso). ¿Cómo se puede recuperar la información perdida?



Inventando un *algoritmo de antialias*, es decir, un proceso que permita recuperar algo de lo perdido, hice que el porcentaje de negro de cada pixel dependiera de qué tan cerca está el centro de la línea del centro del pixel.



Así, al primer pixel de la esquina superior izquierda, cuyo centro está muy lejos del centro de la línea, le asigno un color con porcentaje de negro muy bajo (digamos 5%). Y al pixel central, cuyo centro coincide con el centro de la línea, le asigno un negro total (100%).

Este algoritmo hecho puede no reflejar lo que hace un verdadero algoritmo de anti-alias, pero arroja un resultado similar a los algoritmos de los softwares actuales, que se manifiesta como un degradado de colores entre el color del objeto y el color del fondo.

El aliasing ocurre entonces cuando se intenta representar una información **continua** a través de un medio **discreto**. En el caso de las gráficas digitales este paso ocurre cuando convertimos una información en una imagen de proceso llamado **render**, o en cualquier proceso de digitalización, cuando, por ejemplo, se escanea, se captura una imagen a través de una cámara de video o se mueve el ratón para dibujar una línea.

5 Formatos de archivos de imagen.

5.1 TIFF (Tagged Image File Format)



TIFF es, en principio, un formato muy flexible con o sin pérdida. Los detalles del algoritmo de almacenamiento de la imagen se incluyen como parte del fichero.

En la práctica, TIFF se usa casi exclusivamente como formato de almacenamiento de imágenes sin pérdidas y sin ninguna compresión. Consecuentemente, los archivos en este formato suelen ser muy grandes. Algunas veces se usa un algoritmo de compresión sin pérdidas llamado LZW, pero no siempre.

El formato TIF (Formato de archivo de imágenes con etiquetas) es un formato de archivo de gráficos de mapa de bits (una trama). Fue desarrollado en 1987 por Aldus (ahora pertenece a Adobe). Las últimas especificaciones (Revisión 6.0) se publicaron en 1992.

Además, el formato TIF permite que se utilicen varios espacios de color: RGB (rojo, verde, azul) CMYK (cian, magenta, amarillo, negro) etc.

El principio del formato TIF consiste en definir etiquetas (de ahí el nombre Formato de archivo de imágenes con etiquetas) que describen las características de la imagen. Las etiquetas permiten almacenar información acerca de las dimensiones de la imagen, la cantidad de colores utilizados, el tipo de compresión, etc. Por lo tanto, una descripción de imagen que utiliza etiquetas simplifica la programación del software permitiendo guardar información en formato TIFF. Por otro lado, la cantidad de opciones es tan amplia que muchos editores de imágenes que admiten el formato TIFF no las integran todas.

Otra ventaja de este formato que, por su importancia, merece ser reseñada, es su implantación. Y es que los ficheros TIFF **pueden ser manipulados por prácticamente cualquier aplicación de edición fotográfica o diseño** del mercado. Y esto no es todo. Además, pueden almacenarse con capas y guardarse una y otra vez sin deteriorarse.

Actualmente tenemos a nuestra disposición otros formatos de imagen con características más atractivas que TIFF. Los ficheros almacenados en este último formato **son pesados**, sobre todo si contienen varias capas, y su compresión no es muy eficiente. Hay opciones, como **PSD**, por ejemplo, que ofrecen una **calidad equivalente** y una **compresión mucho mayor que TIFF**.

Además, las técnicas de compresión que se aplican actualmente a los archivos JPEG se han depurado mucho. De hecho, un JPEG puede ofrecer un **acabado cercano al de un fichero comprimido sin pérdida de calidad**, pero **ofreciendo un «peso» mucho menor**. Aunque, si guardamos muchas veces el mismo archivo, la pérdida de calidad será cada vez más perceptible.

5.2 GIF (Graphic Interchange Format, Formato de intercambio de gráficos)



GIF crea una tabla de 256 colores a partir de una de 16 millones. Es un formato de archivos de gráficos de mapa de bits (una trama) desarrollado por Compuserve.

Si la imagen tiene menos de 256 colores, GIF puede almacenar la imagen sin pérdidas. Cuando la imagen contiene muchos colores, el software que crea el archivo GIF usa algún algoritmo para aproximar los colores de la imagen con una paleta limitada de 256 colores disponibles. Un buen algoritmo de este tipo tratará de encontrar un conjunto óptimo de 256 colores. Algunas veces, GIF usa el color más cercano para representar cada píxel, y algunas veces usa un "error de difusión" para ajustar los colores de los píxeles vecinos y así corregir el error producido en cada píxel.

GIF produce compresión de dos formas. Primero, reduce el número de colores de la imagen a 256 y por tanto, reduce el número de bits necesario por píxel. Después, reemplaza áreas de color uniforme usando código de secuencias: en lugar de almacenar "blanco, blanco, blanco, blanco, blanco" almacena "5 blanco"

Por tanto, GIF es una compresión de imágenes sin pérdida sólo para imágenes de 256 colores o menos. Sin embargo, para una imagen de 16 millones de colores GIF puede "perder" el 99.998% de los colores.

Dado que la paleta tiene un número de colores limitado (no limitado en cuanto a colores diferentes), las imágenes que se obtenían con este formato por lo general eran muy pequeñas. Sin embargo, dado que el algoritmo de compresión estaba patentado, todos los editores de software que usaban imágenes GIF debían pagarle regalías a Unisys, la compañía propietaria de los derechos. Esta es una de las razones por las que el formato se está volviendo cada vez más popular, en perjuicio del formato GIF.

Su principal utilidad hoy en día sigue siendo el despliegue de imágenes animadas para páginas web, al ser el único formato soportado por multitud de navegadores que permita dicho efecto. Cabe destacar que la animación de este tipo de imágenes solo se puede visualizar en cierto

tipo de aplicaciones y programas como presentaciones power point o páginas web, pero en hojas de cálculo o documentos de texto las imágenes gif pierden su animación.

Las redes sociales han provocado una nueva edad de oro en este formato, que había perdido terreno, frente a otros de alta resolución para las fotografías. Las redes sociales que permiten las animaciones han hecho que el gif animado vuelva a ser un formato muy utilizado por su sencillez de edición y poco peso frente a los vídeos.

5.3 PNG (Gráficos de Red Portátiles)

Fue desarrollado en 1995 como una alternativa gratuita al formato GIF, que es un formato patentado cuyos derechos pertenecen a Unisys (propietario del algoritmo de compresión LZW), a quien todos los editores de software que usan este tipo de formato deben pagar regalías. Por lo tanto, PGN es un acrónimo recursivo de *PNG No es GIF*.

El formato PNG permite almacenar imágenes en blanco y negro (una profundidad de color de 16 bits por píxel) y en *color real* (una profundidad de color de 48 bits por píxel), así como también imágenes indexadas, utilizando una paleta de 256 colores.

Además, soporta la transparencia de canal alfa, es decir, la posibilidad de definir 256 niveles de transparencia, mientras que el formato GIF permite que se defina como transparente sólo un color de la paleta. También posee una función de entrelazado que permite mostrar la imagen de forma gradual.

La compresión que ofrece este formato es (*compresión sin pérdida*) de 5 a 25% mejor que la compresión GIF.

Por último, el PNG almacena información gama de la imagen, que posibilita una corrección de gama y permite que sea independiente del dispositivo de visualización. Los mecanismos de corrección de errores también están almacenados en el archivo para garantizar la integridad.

5.4 JPG



JPG es el método de compresión más adecuado para fotografías e imágenes de tonos continuos similares que contiene muchos colores. Permite obtener unos ratios de compresión muy altos manteniendo a su vez una calidad en la imagen muy elevada.

JPG analiza las imágenes y elimina la información que no es apreciable. JPG almacena imágenes de 16 millones de colores. Otro aspecto importante es que el método JPG permite distintos niveles de compresión. En niveles de compresión de imágenes moderado, es muy difícil discernir las diferencias de la imagen original. Programas de tratamiento de imágenes avanzados como Paint Shop Pro o Photoshop permiten ver la calidad de la imagen y el tamaño del fichero como una función de nivel de compresión, de esa forma, se puede elegir convenientemente la calidad y el tamaño del fichero deseado.

5.5 SVG

Es un formato vectorial poco conocido pero muy útil para su uso online por su flexibilidad y por la capacidad de ofrecer gráficos con calidad.



El formato SVG es para muchos un total desconocido. Cuando queremos colocar algún gráfico en la web, en la mayoría de las ocasiones optamos por JPG o alguna vez por un PNG o GIF, en caso de necesitar transparencia o animación. Lo que muchos no saben es que se pueden usar archivos vectoriales para su uso en navegadores.

SVG es vectorial, lo que supone tener todas las ventajas de cualquier formato vectorial. Es escalable, pesa poco y permite una definición mayor a tamaños reducidos, mucho mayor que los archivos bitmap. El formato es igual al que se utiliza con cualquier programa vectorial como Corel Draw o Adobe Illustrator.

Así como el formato Flash, que también era vectorial, propiedad de Macromedia, es decir de Adobe, el formato SVG es un formato abierto, estándar, compatible y basado en XML. Aunque las primeras versiones no se podían ver en los diferentes navegadores, hoy ya es un estándar que funciona sin problemas en todos los navegadores. SVG se convirtió en una recomendación en septiembre de 2001 con lo que en estos momentos ya es admitido por todos. A ser un formato basado en XML necesitamos cierto control de código para hacer que un archivo SVG funcione adecuadamente.

6 Optimización de imágenes para la web

Al crear un sitio web es muy recomendable que los archivos que contienen las imágenes ocupen el menor número posible de bytes para agilizar su descarga y visualización por Internet. Esto garantizará el acceso de aquellos clientes que utilicen conexiones con anchos de banda modestos.

El tamaño de un archivo gráfico viene determinado, entre otros, por los siguientes factores:

- Dimensiones de la imagen.
- Profundidad o paleta de colores.
- Resolución.
- Formato de archivo (JPG, GIF, PNG).

6.1 Recomendaciones de optimización

Conviene definir una resolución de imagen no superior a 96 ppp. No interesa optar por valores mayores, ya que aumenta considerablemente el peso del archivo a descargar y el usuario no lo aprecia.

En ocasiones puede interesar **reducir el número de colores de la paleta** porque ello supone disminuir el tamaño del archivo.

Conviene utilizar un programa de tratamiento de imágenes para **definir las dimensiones concretas** de una imagen antes de insertarla en una página web.

Es recomendable guardar los originales de las imágenes favoritas en formato BMP, TIFF ó JPEG sin comprimir. A partir de ellas se puede crear una copia en formato GIF (PNG) o JPEG con las dimensiones, resolución y paletas optimizadas para publicarlas en la web.

Las imágenes GIF son más adecuadas para dibujos, gráficos y logotipos. Son aquellas donde predominan los colores sólidos y una paleta con un número reducido de colores.

Las imágenes JPEG se adaptan mejor a fotografías e imágenes con degradados complejos. Admiten color de 24 bits gracias a su compresión ofrecen una imagen más brillante que ocupa menos espacio.

La correcta combinación de aspectos relacionados con la resolución, el color, el número de bits y el formato, es muy importante para la optimización de una imagen. Puesto que en la web se deben cumplir, por encima de todo, dos principios: que las imágenes se vean bien y que pesen poco, para que no tarden mucho en cargarse.

6.2 Herramientas de optimización

Hay herramientas específicas y relacionadas con actividades de optimización y reducción del tamaño, por ejemplo:

- **ImageOptimización:** con esta herramienta *on line* se puede cargar una imagen de mapa de bits (GIF, JPG o PNG) y optimizarla para su uso en la Web, sin sacrificar calidad.
- **DoSize:** permite cambiar el tamaño de una imagen y adaptarlo a las necesidades del diseñador.

7 Conceptos básicos de vídeo digital

Dimensiones.

Es el tamaño del video (ancho x alto) expresado en píxeles cuando se visualiza al 100%, sin agrandar ni reducir.

Los reproductores pueden mostrar un video a pantalla completa o con una ampliación del 200%, 300%, etc. En estos casos el video pierde calidad de imagen y esta pérdida depende del formato de archivo. Un video AVI puede tener cualquier ancho y alto mientras que los estándares de VideoCD son 352 x 288 y de DVD 720 x 576.

Codec.

Acrónimo de "codificación/decodificación". Un códec es un algoritmo especial que reduce el número de bytes que ocupa un archivo de video. Los archivos codificados con un códec específico requieren el mismo códec para ser decodificados y reproducidos. Algunos de los códec más utilizados para el formato AVI son: DivX, XviD, CinePak, Intel Indeo 5, DV, etc.

Bitrate (Tasa de bits)

Conocido también como el flujo de datos es la tasa de bits o datos que son procesados por unidad de tiempo. Por tal motivo, de la misma forma que sucede con el tamaño de imagen, **cuanto mayor sea el flujo de datos, mayor será la calidad del material.**

Fotogramas por segundo.

Un video resulta de la exposición imágenes o fotogramas uno detrás de otro. Un parámetro de la calidad del video es el número de fotogramas por segundo que muestra durante su reproducción. Este valor oscila entre 15 y 30. Por ejemplo los vídeos en DVD en Europa exhiben 25 fotogramas por segundo (25 fps).

Fotogramas Clave.

Cuando se aplica un códec de compresión a un video, se suele producir cierta pérdida de la información de sus fotogramas. Algunos fotogramas (los fotogramas clave) se almacenan completamente en el archivo comprimido, mientras que el resto sólo se guardan parcialmente.

En la descompresión, estos fotogramas intermedios se reconstruyen a partir de los fotogramas clave.

Proporción o ratio de aspecto.

Es la proporción entre la anchura y altura de un video. Cuando se reproduce un video se suele mantener por defecto esta proporción para evitar deformación de las imágenes. Por este motivo cuando se elige la visualización a pantalla completa, aparecen franjas negras arriba y abajo. Es habitual una relación 4:3 para los videos domésticos (352x288 píxeles, por ejemplo) mientras que en DVD se suele trabajar con ratios de 16:9.

8 Formatos de archivos de video

Los videos digitales se pueden guardar en archivos de distintos formatos. Cada uno se corresponde con una extensión específica del archivo que lo contiene. Existen muchos tipos de formatos de video. Aquí se citan algunos de los más utilizados. Asimismo, cada tipo de archivo admite en cada momento un códec de compresión distinto.

Estos formatos son en realidad formatos contenedores, es decir, un tipo de formato de archivo que almacena información de vídeo, audio, subtítulos, capítulos, meta-datos e información de sincronización siguiendo un formato preestablecido en su especificación

8.1 AVI (Audio Video Interleaved = Audio y Video Intercalado)

- Es el formato estándar para almacenar video digital.
- Cuando se captura video desde una cámara digital al ordenador, se suele almacenar en este formato con el códec DV (Digital Video).
- El archivo AVI puede contener video con una calidad excelente. Sin embargo, el peso del archivo resulta siempre muy elevado.
- Admite distintos códecs de compresión como CinePak, Intel Indeo 5, DV, etc. Los códecs con más capacidad de compresión y una calidad aceptable son DivX y XviD.
- El formato AVI puede ser visualizado con la mayoría de los reproductores: Windows Media, QuickTime, etc. siempre y cuando se encuentren instalados en el equipo los adecuados códecs para cada tipo de reproductor.
- Es ideal para guardar videos originales que han sido capturados de la cámara digital (codificados con DV).

- No es recomendable publicarlos en Internet en este formato por su enorme peso.
- Los códecs CinePak, Intel Indeo, DV, etc. no ofrecen una gran compresión. Los códecs DivX y XviD por el contrario consiguen una óptima compresión, aunque se suelen destinar sobre todo a la codificación de películas de larga duración.

8.2 MPEG (Moving Pictures Expert Group = Grupo de Expertos de Películas)

- Es un formato estándar para la compresión de video digital.
- Son archivos de extensión *.MPG ó *.MPEG.
- Admite distintos tipos de códecs de compresión: MPEG-1 (calidad CD), MPEG-2 (calidad DVD), MPEG-3 (orientado al audio MP3) y MPEG-4 (más orientado a la web).
- Se reproducen con Windows Media Player y QuickTime.
- También se le llama MP4

8.3 MOV (<http://www.apple.com/es/quicktime/>)

- Es el formato de video y audio desarrollado por Apple.
- Utiliza un códec propio que evoluciona en versiones con bastante rapidez.
- Este tipo de archivos también pueden tener extensión *.QT
- Se recomienda utilizar el reproductor de QuickTime. Existe una versión gratuita del mismo que se puede descargar de Internet.
- Es ideal para publicar videos en Internet por su razonable calidad/peso.
- Admite streaming.

8.4 WMV (<http://www.microsoft.com/windows/windowsmedia/es/>)

- Ha sido desarrollado por Microsoft.
- Utiliza el códec MPEG-4 para la compresión de video.
- También puede tener extensión *.ASF
- Sólo se puede visualizar con una versión actualizada de Windows Media 7 o superior.
- Esta aplicación viene integrada dentro de Windows.
- Es ideal para publicar videos en Internet por razonable calidad/peso.
- Admite streaming.

8.5 FLV (<http://www.adobe.com>)

- Es un formato que utiliza el reproductor Adobe Flash para visualizar vídeo en Internet.
- Utiliza el códec Sorenson Spark y el códec On2 VP6. Ambos permiten una alta calidad visual con bitrates reducidos.
- Son archivos de extensión *.FLV, SWF, Y .F4V
- Se pueden reproducir desde distintos reproductores locales: MPlayer, VLC media player, Riva, Xine, et.
- Opción recomendada para la web por su accesibilidad. Al visualizarse a través del reproductor de Flash es accesible desde la mayoría de los sistemas operativos y navegadores web.
- Los repositorios de vídeo más conocidos en Internet utilizan este formato para la difusión de vídeos: YouTube, Google Video, iFilm, etc.

- Permite configurar distintos parámetros del vídeo para conseguir una aceptable calidad/peso.
- Admite streaming.

8.6 Otros formatos

- **OGG Y .OGV:** *ogv* es el correspondiente contenedor Open Source de la Fundación Xiph.Org. Apropiado para contener el formato Theora .
- **MKV** (Matroska): es un formato Open Source que puede contener casi cualquier tipo de formato de vídeo. Muy usado originalmente para comprimir películas que se han de compartir por Internet.
- **WEBM** (WebM): es un contenedor de vídeo Open Source desarrollado por Google, muy dirigido para usarse con HTML5. Está compuesto por el códec VP8 y el códec de audio Vorbis (*ogg*) dentro de un contenedor multimedia Matroska.

9 ¿Qué es el streaming?

En la navegación por Internet es necesario descargar previamente el archivo (página HTML, imagen JPG, audio MP3, etc.) desde el servidor remoto al cliente local para luego visualizarlo en la pantalla de este último.

La tecnología de streaming se utiliza para optimizar la descarga y reproducción de archivos de audio y video que suelen tener un cierto peso.

El streaming funciona de la siguiente forma:

- **Conexión con el servidor.** El reproductor cliente conecta con el servidor remoto y éste comienza a enviarle el archivo.
- **Buffer.** El cliente comienza a recibir el fichero y construye un buffer o almacén donde empieza a guardarlo.
- **Inicio de la reproducción.** Cuando el buffer se ha llenado con una pequeña fracción inicial del archivo original, el reproductor cliente comienza a mostrarlo mientras continúa en segundo plano con el resto de la descarga.
- **Caídas de la velocidad de conexión.** Si la conexión experimenta ligeros descensos de velocidad durante la reproducción, el cliente podría seguir mostrando el contenido consumiendo la información almacenada en el buffer. Si llega a consumir todo el buffer se detendría hasta que se volviera a llenar.

El streaming puede ser de dos tipos dependiendo de la tecnología instalada en el servidor:

- **Descarga progresiva.** Se produce en servidores web que disponen de Internet Information Server (IIS), Apache, Tomcat, etc. El archivo de vídeo o audio solicitado por el cliente es liberado por el servidor como cualquier otro archivo utilizando el protocolo HTTP. Sin embargo, si el archivo ha sido especialmente empaquetado para streaming, al ser leído por el reproductor cliente, se iniciará en streaming en cuanto se llene el buffer.
- **Transmisión por secuencias.** Se produce en servidores multimedia que disponen de un software especial para gestionar más óptimamente el streaming de audio y

vídeo: Windows Media Server, Flash Communication Server, etc. La utilización de un servidor multimedia ofrece múltiples ventajas frente al servidor web.

Las más destacadas son:

- a. Mayor rapidez en la visualización de este tipo de contenidos.
- b. La comunicación entre servidor/cliente se puede realizar por protocolos alternativos al HTTP. Tiene el inconveniente del bloqueo impuesto por Firewalls, pero tiene la ventaja de una mayor rapidez.
- c. Mejor gestión del procesador y ancho de banda de la máquina del servidor ante peticiones simultáneas de varios clientes del mismo archivo de audio o vídeo.
- d. Control predefinido sobre la descarga que pueden realizar los clientes: autenticada, filtrada por IP, sin almacenarla en la caché del cliente, etc.
- e. Mayor garantía de una reproducción ininterrumpida gracias al establecimiento de una conexión de control inteligente entre servidor y cliente.
- f. Posibilidad de distribución de transmisiones de audio y vídeo en directo.

10 Conceptos básicos del sonido digital

10.1 Frecuencia.

Es el número de vibraciones por segundo que da origen al sonido analógico. El espectro de un sonido se caracteriza por su rango de frecuencias. Ésta se mide en Hertzios (Hz). El oído humano capta sólo aquellos sonidos comprendidos en el rango de frecuencias 20 Hz y 20.000 Hz.

10.2 Tasa de muestreo (sample rate).

Un audio digital es una secuencia de ceros y unos que se obtiene del muestreo de la señal analógica. La **tasa de muestreo o sample rate** define cada cuánto tiempo se tomará el valor de la señal analógica para generar el audio digital. Esta tasa se mide en Hertzios (Hz).

Por ejemplo: 44100 Hz. nos indica que en un segundo se tomaron 44.100 muestras de la señal analógica de audio para crear el audio digital correspondiente. Un audio tendrá más calidad cuanto mayor sea su tasa de muestreo. Algunas frecuencias estándares son 44100 Hz., 22050 Hz., y 11025 Hz.

10.3 Resolución (bit resolution)

Es el número de bits utilizados para almacenar cada muestra de la señal analógica. Una resolución de 8-bits proporciona 256 (2^8) niveles de amplitud, mientras que una resolución de 16-bits alcanza 65536 (2^{16}). Un audio digital tendrá más calidad cuanto mayor sea su resolución.

Ejemplo: El audio de calidad CD suele ser un sonido de 44.100 Hz – 16 bits – estereo.

10.4 Velocidad de transmisión (bitrate)

El bitrate define la cantidad de espacio físico (en bits) que ocupa un segundo de duración de ese audio.

Por ejemplo, 3 minutos de audio MP3 a 128kBit/sg, ocupa 2,81 Mb de espacio físico ($3\text{min} \times 60 \text{ seg/min} \times 128 \text{ kBit/seg} = 23040 \text{ kBits} \rightarrow 23040 \text{ kBits} \times 1024 \text{ bits/Kbit} : 8 \text{ bits/bytes} : 1024 \text{ bytes/Kbytes} : 1024 \text{ Kbytes/Mbytes} = 2,81 \text{ MBytes ó Mb}$). Por ejemplo en los audios en formato MP3 se suele trabajar con bitrates de 128 kbps (kilobits por segundo). El audio tendrá más calidad cuanto mayor sea su bitrate y el archivo que lo contiene tendrá mayor peso. Esta magnitud se utiliza sobre todo en el formato MP3 de audio más destinado a la descarga por Internet.

10.5 Códec.

Acronimo de "codificación/decodificación". Un códec **es un algoritmo especial que reduce el número de bytes que ocupa un archivo de audio**. Los archivos codificados con un códec específico requieren el mismo códec para ser decodificados y reproducidos. El códec más utilizado en audio es el MP3.

10.6 Decibelio.

Unidad de medida del volumen o intensidad de un sonido. El silencio o ausencia de sonido se cuantifica como 0 dB y el umbral del dolor para el oído humano se sitúa en torno a los 130,140 dB.

11 Formatos de audio

11.1 El formato MP3 (MPEG 1 Layer 3)

Fue creado por el Instituto Fraunhofer, y por su extraordinario grado de compresión y alta calidad está prácticamente monopolizando el mundo del audio digital.

- Es ideal para publicar audios en la web. Se puede escuchar desde la mayoría de los reproductores.
- La transformación de WAV a MP3 o la publicación directa de una grabación en formato MP3 es un proceso fácil y al alcance de los principales editores de audio.
- Tiene un enorme nivel de compresión respecto al WAV. En igualdad del resto de condiciones reduciría el tamaño del archivo de un fragmento musical con un factor entre 1/10 y 1/12.
- Presentan una mínima pérdida de calidad.

Éste es un formato de compresión de audio digital patentado que usa un algoritmo **con pérdida** para conseguir un menor tamaño de archivo. Es un formato de audio común usado para música tanto en ordenadores como en reproductores de audio portátil.

Más concretamente, *mp3* fue desarrollado por el *Moving Picture Experts Group* (MPEG) para formar parte del estándar MPEG-I y del posterior y más extendido MPEG-2.

Un **mp3** puede comprimirse usando una mayor o menor **tasa de bits por segundo**, resultando directamente en su mayor o menor calidad de audio final, así como en el tamaño del archivo resultante.

Es importante destacar que, en lo que respecta al desarrollo web y al uso del audio para ser interpretado por los diferentes navegadores, **mp3 no es único y universal**, es decir, hay más formatos iguales o mejores y no todos los navegadores lo soportan. Por lo tanto, hay otras posibilidades de formatos que hay que conocer, sobre todo para, como luego veremos, permitir que la reproducción de audio sea lo más universal posible entre todos los navegadores actuales (al margen de sus batallas empresariales).

Los siguientes formatos no son los únicos que existen, pero sí son los más conocidos en Internet, y los más usados en la distribución de audio en sitios web.

11.2 Ogg

Es un formato **válido para audio y vídeo**, desarrollado por la Fundación Xiph.org.

Su principal ventaja con respecto a mp3 es que es un formato libre de patentes y abierto diseñado para dar un alto grado de eficiencia en el streaming y la compresión de archivos. De manera incorrecta, muchas veces los archivos ogg se les llama Vorbis ya que éste fue el primer códec desarrollado para leer este formato. Sin embargo, hay muchos otros códec además de Vorbis que interpretan ogg. Actualmente, Firefox y Chrome interpretan ogg.

- Muestra un grado de compresión similar al MP3 pero según los expertos en música la calidad de reproducción es ligeramente superior.
- No todos los reproductores multimedia son capaces de leer por defecto este formato. En algunos casos es necesario instalar los códec o filtros oportunos.
- El formato OGG puede contener audio y vídeo.

11.3 Real Audio

Sin duda, este formato es el rey del streaming. La gran mayoría de las páginas que usan streaming utilizan el formato Real Audio para difundir sus contenidos. RealNetworks es la compañía que ha creado Real Audio y, por supuesto, también ha creado un reproductor (player en inglés) para su formato, el RealPlayer (del que existe una versión gratuita y otra comercial).

11.4 Windows Media Audio (wma)

Es un formato desarrollado por Microsoft. Ofrece gran calidad de sonido en un tamaño reducido. Y como ventaja para las compañías y los músicos, el formato de Microsoft incluye un sistema de gestión de los derechos de autor para evitar la piratería y la creación de copias no autorizadas de las canciones.

Por las características de calidad y tamaño wma resulta muy adecuado para la reproducción de archivos en streaming. El reproductor que Microsoft ha desarrollado para escuchar su formato de audio es el Windows Media Player que se encuentra disponible en su web. La desventaja de wma es que, al ser de Microsoft, el navegador IExplorer lo soporta, pero no se puede asegurar en el resto.

11.5 Wav o wave (WAVEform audio file format):

Es un formato de audio que se suele utilizar sin compresión de datos. Este formato es muy conocido en entornos Windows, pero no se suele usar en Internet por ocupar mucho espacio al ser usado sin compresión.

- El formato WAV (Wave Form Audio File) es un archivo que desarrolló originalmente Microsoft para guardar audio. Los archivos tienen extensión *.wav
- Es ideal para guardar audios originales a partir de los cuales se puede comprimir y guardar en distintos tamaños de muestreo para publicar en la web.
- Es un formato de excelente calidad de audio.
- Sin embargo, produce archivos de un peso enorme. Una canción extraída de un CD (16 bytes, 44100 Hz y estéreo) puede ocupar entre 20 y 30 Mb.

Los archivos WAV se pueden guardar con distintos tipos de compresión. Las más utilizadas son la compresión PCM y la compresión ADPCM. No obstante, incluso definiendo un sistema de compresión, con un audio de cierta duración se genera un archivo excesivamente pesado.

El formato WAV se suele utilizar para fragmentos muy cortos (no superiores a 3-4 segundos), normalmente en calidad mono y con una compresión Microsoft ADPCM 4 bits.

11.6 Free Lossless Audio Codec (FLAC)

Es un códec de audio que permite que el audio digital sea comprimido sin pérdidas de tal manera que el tamaño del archivo de audio se reduce sin que se pierda ningún tipo de información. El audio digital comprimido por el algoritmo de FLAC típicamente se puede reducir de 50 a 60% de su tamaño original y se descomprime en una copia idéntica de los datos de audio originales. La desventaja es que el archivo ocupa mucho más espacio del que se obtendría al aplicar un algoritmo de compresión con pérdida.

FLAC es un formato abierto con licencia libre de regalías y una implementación de referencia la cual es software libre. FLAC cuenta con soporte para etiquetado de metadatos, inclusión de la portada del álbum, y la búsqueda rápida.

11.7 Formato MIDI

El formato MIDI (Musical Instrument Digital Interface = Interface Digital para Instrumentos Digitales) en realidad no resulta de un proceso de digitalización de un sonido analógico. Un archivo de extensión *.mid almacena secuencias de dispositivos MIDI (sintetizadores) donde se recoge qué instrumento interviene, en qué forma lo hace y cuándo.

Este formato es interpretado por los principales reproductores del mercado: Windows Media Player, QuickTime, etc.

Los archivos MIDI se pueden editar y manipular mediante programas especiales y distintos de los empleados para editar formatos WAV, MP3, etc. El manejo de estos programas suele conllevar ciertos conocimientos musicales.

Los archivos MIDI permiten audios de cierta duración con un reducido peso. Esto es debido a que no guardan el sonido sino la información o partitura necesaria para que el ordenador la componga y reproduzca a través de la tarjeta de sonido.

Se suelen utilizar en sonidos de fondo de páginas HTML o para escuchar composiciones musicales de carácter instrumental.

El formato MIDI no permite la riqueza de matices sonoros que otros formatos ni la grabación a partir de eventos sonoros analógicos.

12 Optimización de archivos de video y audio

Para optimizar el peso del archivo de video será necesario editarlo para establecer alguno o algunos de los siguientes parámetros:

En el Audio:

- **Velocidad de transmisión** (bitrate). Configurar bitrates más bajos: 128 Kbps, 96 Kbps, 64 Kbps, etc.
- **Calidad estéreo/mono.** La reducción a calidad “mono” reduce considerablemente el peso del archivo. Por otro lado, la calidad de reproducción “mono” para la mayoría de audios y de público es apenas perceptible.
- **Factor de compresión.** El formato WAV admite distintos factores de compresión: PCM y ADPCM.
- **Resolución.** Establecer resoluciones más pequeñas: 32-bits, 16-bits, 8-bits, 4-bits, etc.
- **Tasa de muestreo.** Definir valores inferiores: 44100 Hz., 22050 Hz., 11025 Hz, etc.
- **Duración.** En ocasiones se puede utilizar un fragmento más corto que reproducido en bucle cubre el tiempo suficiente de acompañamiento musical. A éstos se les llama loops.
- **Formato.** Es preferible utilizar el formato MP3 u OGG en lugar del WAV por su potente factor de compresión y su aceptable calidad de audio.

En el Video:

- El **códec de compresión** de video utilizado: MPEG-1, MPEG-2, MPEG-4, Intel Indeo, Cinepak, DivX, etc.
- **Método de BitRate.** Utilizar un bitrate variable VBR puede optimizar la calidad del video y repercutir en el peso final del archivo frente a un bitrate constante CBR.
- **Velocidad de transmisión (bitrate).** Configurar bitrates más bajos: 1000 Kbps, 768 kbps, 360 Kbps, etc.
- **Dimensiones.** Cuanto más pequeña sea la altura y anchura en píxeles de los fotogramas de un video, menos tamaño ocupará su archivo.
- **Velocidad de fotogramas.** Se puede reducir el número de fotogramas por segundo que mostrará el video: 30, 24, 20, 16, etc.
- **Fotogramas Clave.** Durante la compresión también se puede indicar cada cuánto se guardará un fotograma completo (fotograma clave): 24, 48, 96, 128, etc. Cuanto mayor sea esta cadencia más bajo será el peso del archivo resultante.

Otros elementos que inciden en la optimización:

- **Duración.** Cuanto más corto es un video, menos peso ocupa su archivo. En ocasiones puede resultar interesante fraccionar un archivo de video en sus escenas para facilitar su descarga.
- **Formato de archivo.** Los archivos *.WMV, *.MOV, *.RM y *.FLV son los más adecuados para publicar un video en Internet por su adecuada relación calidad/peso y porque admiten streaming. Los archivos *.AVI con códecs de compresión baja son ideales para guardar los videos originales. Los archivos *.AVI con códecs DivX-XviD son apropiados para videos de películas de cierta duración. Los archivos *.MPG con códec MPEG-1 se utilizan para crear Video-CDs. Los archivos *.MPG con códec MPEG-2 se utilizan como fuente para montar un DVD.

13 Insertar elementos multimedia en la web

13.1 Insertar audio en una web

En HTML5 la inclusión de sonidos en páginas web intenta paliar muchos de los Inconvenientes de las etiquetas <bgsound> o <embed>. De alguna manera, HTML5 ofrece una alternativa más sencilla y eficaz para que el desarrollador pueda insertar audio. Sin embargo, la realidad es que cada navegador implementa de HTML5 lo que le quiere, por lo que tampoco con esta opción se consigue una solución universal.

La propuesta de HTML5 utiliza la etiqueta <audio>. Esta etiqueta no existe para versiones de HTML4. Evidentemente, eso obliga a que los navegadores que la interpretan deban implementar HTML5 en general (y esta etiqueta en particular).

Los atributos de esta etiqueta son:

- **Autoplay** especifica que el audio debería empezar automáticamente tan pronto como esté listo para reproducirse (el único valor de este atributo es autoplay).
- **Controls** indica que los controles de reproducción serán visibles (el único valor posible de este atributo es controls).
- **Loop** indica que el audio se repetirá automáticamente cuando termine (el único valor posible de este atributo es loop).
- **Preload**, especifica si el audio debería ser cargado cuando la página se carga o no, es decir, se empieza a cargar cuando el usuario lo quiera reproducir (el valor auto indica que se cargará todo el archivo cuando se cargue la página, el valor meta indica que solo se cargarán los metadatos del archivo y none que no habrá precarga).
- **src** especifica la URL de archivo de audio.

Además de estos atributos, <audio> soporta también atributos globales de HTML5 y eventos. Un ejemplo del uso de esta etiqueta para obtener un reproductor para un audio que se inicia solo sería:

```
<audio src="audios/Aretha.mp3" controls="controls" autoplay></audio>
```


Una mejora para hacer más versátil esta opción es usar la etiqueta <source> dentro de <audio>. De esta manera, el navegador intenta cargar la primera línea <source>. Si falla o no es soportada esa reproducción, entonces pasa a la siguiente

El siguiente ejemplo empieza definiendo el tipo de documento (<!DOCTYPE html>) que es: navegadores como IE Explorer lo necesitan para interpretar bien el contenido.

```
<!DOCTYPE html>

<html>

<head>

</head>

<body>

    <audio controls="controls" preload="auto">

        <source src="music.mp3" type="audio/mpeg"/>

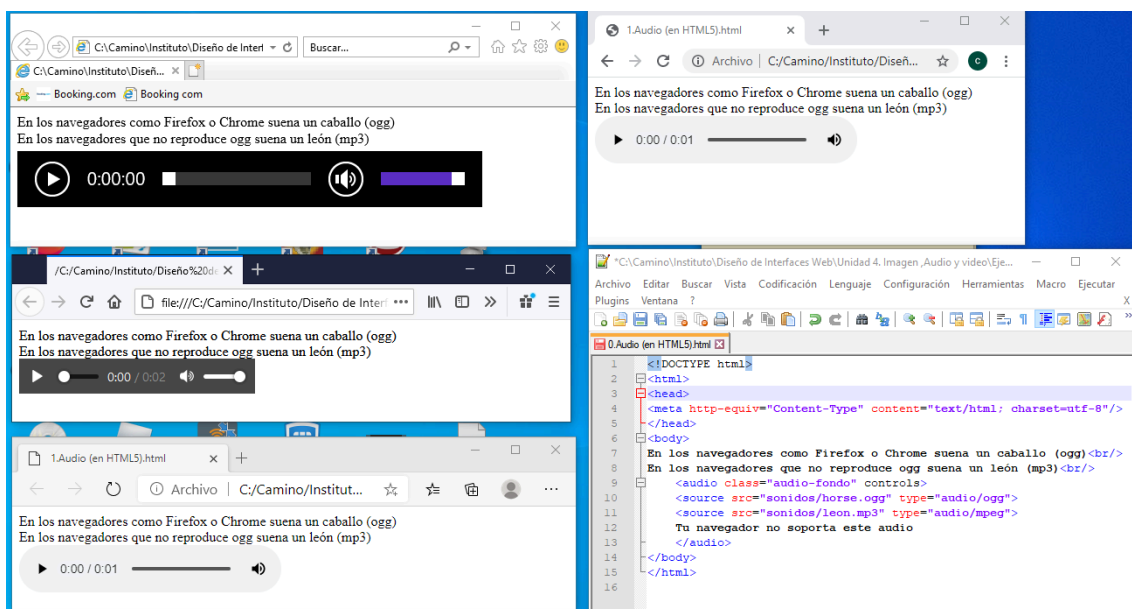
        <source src="music.ogg" type="audio/ogg"/> El navegador no soporta el audio

    </audio>

</body>

</html>
```

A continuación, se muestra un ejemplo de cómo se muestra en distintos navegadores:



La Figura muestra el resultado en Google Chrome.

A la etiqueta <audio> también se le puede aplicar estilos CSS. Por ejemplo, el siguiente código define un estilo para hacer que una etiqueta audio que use la clase audio-fondo aparezca con un tamaño determinado.

```
<style> .audio-fondo { width: 160px; height: 36px; } </style>
```

Los dispositivos móviles actuales integran navegadores que soportan HTML5 por lo que es cuestión de poco tiempo que todos los usuarios tengan navegadores HTML5 y no sea necesario buscar alternativas con <object> o JavaScript para detectar que audio puede reproducir un complemento determinado de un navegador concreto. Aunque será inevitable que en la etiqueta <audio> se den opciones variadas respecto al tipo de archivo (mp3, ogg).

Por último, como opción de diseño avanzada, se puede usar JavaScript y HTML5 para personalizar el reproductor. El siguiente código utiliza botones para reproducir, en vez de un player. Esta opción da mucho juego en el diseño del reproductor al poder cambiar los botones por imágenes acordes con el estilo del sitio.

```
<!DOCTYPE html>

<html>

<head> </head>

<body>

  <audio id="player" src="Aretha.mp3"></audio>

  <div>

    <button onclick="document.getElementById('player').play()">Play</button>

    <button onclick="document.getElementById('player').pause()">Pausa</button>

    <button onclick="document.getElementById('player').volume+=0.1"> + Volumen
  </button>

    <button onclick="document.getElementById('player').volume-=0.1"> - Volumen
  </button>

  </div>

</body>

</html>
```

En JavaScript existen librerías, como audio.js <http://kolber.github.com/audiojs/>, que permite reproducir audio en todos los navegadores como con la etiqueta <audio> aunque no soporten HTML5.

También tenemos la posibilidad de implementar reproductores de audio de librerías de jQuery, como, por ejemplo: <http://audiocogs.org/codecs/alaac/>

13.2 Consideraciones para el uso de audio en un sitio web

En general, el audio en sitios web se puede usar de dos maneras:

- Como contenido y por tanto tiene un lugar en un sitio web: Por ejemplo, un sitio web de una cadena de radio necesita insertar sonido para que los usuarios puedan acceder a programas de días anteriores.
- Como elemento decorativo, por ejemplo, música de fondo.

El primer uso es inevitable y está asociado con el objetivo del sitio web, por lo que debe implementarse para que pueda ser interpretado por la mayor cantidad de navegadores.

Sin embargo, el segundo uso es meramente ornamental, no imprescindible, con el que se puede conseguir un efecto positivo (por ejemplo, llamar la atención del usuario) o negativo, contrario al que se espera de él: Un ejemplo clásico del audio como elemento decorativo negativo es cuando se usa como fondo.

Actualmente su uso está "pasado de moda", los diseños actuales no suelen usar música de fondo, ya que en muchos casos son una molestia para los usuarios (por ejemplo, la música aparece cuando el volumen de los altavoces está en los más alto). En cualquier caso, si aparece sonido de fondo, siempre debe existir la posibilidad de que el usuario lo apague (y/o inicie) cuando quiera, por lo que deberían aparecer los controles del reproductor cuando se muestre.

En conclusión, los audios son archivos con un tamaño considerable. Por lo tanto, **su uso se debe restringir para cuando es imprescindible**, y siempre optimizando su reproducción (jugando, por ejemplo, con *preload* de la etiqueta <audio>).

13.3 Insertar vídeo en una web

Para aprender a insertar vídeo en páginas web se empezará por las posibilidades de HTML5 donde hay una etiqueta <video> que permite embeber archivos de vídeo de forma nativa sin necesidad de complementos (plugins) adicionales.

La etiqueta <video> es muy parecido a <audio>: dispone de los atributos autoplay, loop y preload, con la misma sintaxis y semántica que en <audio>. También se puede especificar la fuente de un archivo bien usando el atributo src o bien usando la etiqueta <source>. Así mismo, se puede utilizar los controles que ofrece el navegador de forma nativa utilizando el atributo controls o bien puedes ofrecer tus propios controles en JavaScript

Sin embargo, a diferencia del audio, el vídeo necesita un tamaño para reproducirse, tal y como ocurre con las imágenes. Para ello se pueden usar los atributos height (alto) y width (ancho) con más sentido que en <audio>.

Un ejemplo de uso de esta etiqueta con un archivo mp4 es el siguiente:

```
<video poster="images/portada.png" src="videos/recetapollo.mp4" controls width="360" height="240"></video>
```

Además, de los atributos vistos, la etiqueta <video> también tiene un atributo **poster** en el que se indica una imagen que se usará como portada antes de que el vídeo empiece a reproducirse.

Para hacer la reproducción adecuada para la mayor cantidad de navegadores posible, se puede utilizar la etiqueta <source> igual que se hizo en <audio>.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> </head>
```

```
<body>
```

```
  <video controls="" width="360" height="240">
```

```
<source src="videos/recetapollo.mp4" type="video/mp4">
```

```
<source src="videos/recetapollo.ogv" type="video/ogg">
```

```
<source src="videos/recetapollo.webm" type="video/webm">
```

Tu navegador no soporta este vídeo

```
</video>
```

```
</body>
```

```
</html>
```