# Introducción a HTML

Roberto Casado-Vara [1]

[1] University of Salamanca, Plaza de los Caídos s/n – 37002 – Salamanca, Spain
rober@usal.es

**Resumen:** HyperText Markup Language (HTML) es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con vínculos o enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas y con inserciones multimedia (gráficos, sonido, etc.). Este lenguaje fue desarrollado por Tim Berners-Lee durante los años 90 y ha proliferado con el crecimiento explosivo de la Web. Durante este tiempo, el lenguaje HTML se desarrollo de diferentes maneras, pero la Web en sí misma depende de que to-dos los desarrolladores compartan las mismas convenciones HTML, lo que ha motivado un trabajo colectivo en la especificación de HTML a lo largo de un gran periodo de tiempo. En este capítulo se introduce el lenguaje y se presentan ejemplos básicos.

**Palabras clave:** Usabilidad Web

**Abstract.** HyperText Markup Language (HTML) is a very simple language for describing hypertext, i.e. text presented in a structured and pleasant way, with links (hyperlinks) leading to other documents or related information sources and multimedia insertions (graphics, sound, etc.). This language was developed by Tim Berners-Lee during the 90s and has proliferated with the explosive growth of the Web. During this time, the HTML language developed in different ways, but the Web itself depends on everyone sharing the same HTML conventions, which has led to collective work on HTML specification over a long period of time. This chapter introduces the language and presents basic examples.

**Keywords:** Web Usability



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 1 Introducción a HTML

## 1.1 Versión HTML

Para publicar información y distribuirla globalmente, se necesita un lenguaje entendido universalmente, una especie de lengua franca de publicación que todas las computadoras y dispositivos móviles puedan comprender potencialmente.
**HTML** (*HyperText Markup Language*) es un lenguaje muy sencillo que permite describir hipertexto, es decir, *texto presentado de forma estructurada* y agradable, con vínculos o enlaces (*hyperlinks*) que conducen a otros documentos o fuentes de información relacionadas y con inserciones multimedia (gráficos, sonido, etc.) [1-2].
El lenguaje HTML da a los autores las herramientas para, realizar una gran variedad de tareas, como por ejemplo:

- Publicar documentos en línea con encabezados, textos, tablas, listas, fotos, etc.
- Obtener información en línea a través de vínculos de hipertexto, haciendo clic con el botón de un ratón.

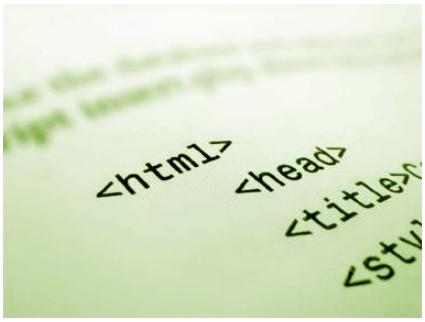



Figura 6 - HTML

- Diseñar formularios para realizar transacciones con servicios remotos, para buscar información, hacer reservas, pedir productos, etc.
- Incluir hojas de cálculo, videoclips, sonidos, y otras aplicaciones directamente en sus documentos.

Este lenguaje fue desarrollado por Tim Berners-Lee durante los años 90, este lenguaje ha proliferado con el crecimiento explosivo de la Web. Durante este tiempo, el lenguaje HTML se desarrollo de diferentes maneras, pero la Web en sí misma depende de que todos los desarrolladores compartan las mismas convenciones HTML, lo que ha motivado un trabajo colectivo en la especificación de HTML a lo largo de un gran periodo de tiempo.
El lenguaje HTML debe funcionar en diferentes plataformas y navegadores, ha sido desarrollado con la premisa de que cualquier tipo de dispositivo debería ser capaz de usar información de la Web (ordenadores, teléfonos móviles, PDA´s, etc), para ello ha sido necesario un gran consenso entre los interlocutores de la industria cada vez que aparecía una nueva versión de HTML, para que así cada nueva versión sea el estándar que sea implementado por los diferentes navegadores Web.
La versión actual de HTML es la 4.01, que fue definida el 24 de diciembre de 1999, por lo que ya es un poco antigua, pero actualmente sigue siendo la más utilizada. Dentro de la última versión de HTML, se distinguen además tres versiones:

- *HTML 4.01 Strict*, en él se utilizan los elementos especificados en HTML 4.01 y no se aceptan otros elementos obsoletos (de versiones anteriores).
- *HTML 4.01 Transitional*, que permite el uso de elementos de diversas versiones de HTML, lo cual en principio, no se considera muy recomendable.
- *HTML 4.01 Frameset*, que soporta *frames*, algo que tampoco se considera muy recomendable actualmente.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

La versión HTML 4.01 implementa una serie de nuevas funciones y mecanismos respecto a versiones anteriores que iremos aprendiendo a lo largo de este módulo:

- **Internacionalización.** Ofrece mayor soporte para diversos lenguajes humanos dentro de un documento, lo que permite un indexado más efectivo por parte de los motores de búsqueda, tipografía de más calidad, mejor conversión de texto a voz, mejor separación de palabras, etc.
- **Accesibilidad.** Esta versión se ha diseñado para hacer las páginas más accesibles a aquellos usuarios con limitaciones físicas, incluyendo características nuevas en formularios, imágenes, tablas, hojas de estilo, etc.
- **Tablas.** Se proporciona un nuevo modelo de construcción de tablas que permite un mayor control sobre la estructura y la presentación de la mismas.
- **Documentos compuestos.** Esta versión implementa un mecanismo estándar para incluir objetos genéricos y aplicaciones completas dentro de documentos HTML.
- **Hojas de estilo.** La inclusión de hojas de estilo simplifican el código HTML y liberan en gran medida al lenguaje HTML de responsabilidades propias de presentación, lo que proporciona además un mayor control sobre el diseño final de documento (fuentes, colores, alineaciones, etc.).
- **Ejecución de scripts.** Gracias a los scripts, los programadores pueden crear páginas Web más dinámicas, además la inclusión de un código de scripts en un documento HTML es independiente del propio lenguaje de script.
- **Impresión.** Existe la posibilidad de facilitar a los usuarios la impresión de los documentos HTML, sin limitarse al documento actual.

Como se ha comentado, la versión actual de HTML esta bastante obsoleta por lo que han ido surgiendo otras versiones nuevas, que sin embargo no lo han conseguido desbancar totalmente, estamos hablando del lenguaje xHTML. Éste nació principalmente para solucionar problemas de compatibilidad con dispositivos móviles o similares que no tenían la potencia suficiente para interpretar correctamente las páginas HTML [3-5].

Las diferencias entre ambos lenguajes no son muy grandes, ya que principalmente están destinados a conseguir por parte de xHTML conformidad con XML. Entre las diferencias principales cabe destacar:

- Los elementos vacíos deben cerrarse siempre.
- Los elementos no vacíos deben cerrarse siempre.
- Los elementos anidados deben tener un correcto orden de apertura cierre.
- Los valores de los atributos deben ir siempre encerrados entre comillas (simples o dobles).
- Los nombres de elementos y atributos deben ir en minúsculas.
- No esta permitida la minimización de atributos.
- Los atributos desaprobados en HTML 4.01 no forman parte de xHTML.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Aunque habitualmente se suele utilizar el término URL (*Universal Resource Locator*) para identificar documentos concretos de HTML en la red redes, resulta mucho más correcto hablar de URI (*Universal Resource Identifier*), ya que es un termino que engloba al primero y permite identificar tanto el recurso, como el fragmento determinado dentro del contenido del recurso. Veamos un ejemplo para entenderlo mejor.

| **Protocolo** | **nombre de dominio** | **Directorio** | **archivo** | **solicitud** | **fragmento** |
|---|---|---|---|---|---|
| http:// | bisite.usal.es | /website/ | index.php | ?q=1 | #formacion |

Cuando empezamos a desarrollar nos podemos encontrar con un problema muy típico en lo que a desarrollos Web se refiere, este es que un misma página en un navegador sea visualice de una forma (correcta o no), pero en otro navegador diferente se vea de otra forma totalmente diferente, incluso que en dos versiones de un mismo navegador se vea diferente. Este problema suele ser debido principalmente a dos razones:

- Por un lado, puede suceder que no estemos implementando correctamente nuestras páginas de acuerdo al estándar HTML 4.01. Este problema tiene un solución simple, se basa en validar nuestras páginas mediante un validador de código, el más utilizado se encuentra en el siguiente enlace: http://validator.w3.org/.
- Por otro lado, también puede suceder que sean los propios navegadores los que no implementen correctamente el estándar, no existe una solución precisa a este problema, aunque realizar código que valide con el estándar suele ser la más habitual.

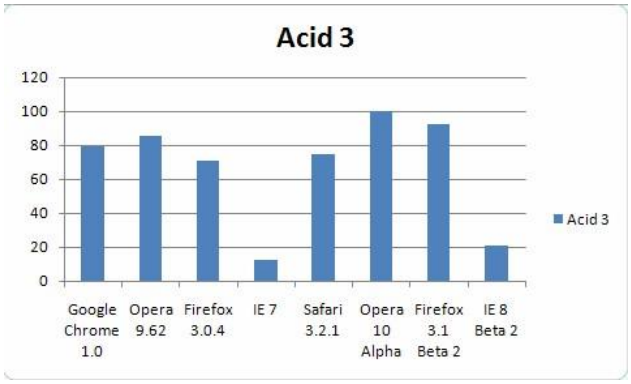



**Figura 7 - Resultados Test Acid 3 para diversos navegadores**

A lo largo de este capitulo, tendremos como objetivo principal aprender a programar documentos en lenguaje HTML de forma correcta y que validen los estándares propuestos. El siguiente objetivo, además de aprender a programar en lenguaje HTML, será estructurar correctamente nuestros documentos, para que así sea mucho más simple aplicar el diseño a los documentos creados. Aunque mediante HTML podremos modificar el diseño del documento, esto no es recomendable y no nos centraremos en ello, nosotros aprenderemos a modificar el estilo de los documentos en el segundo apartado de este módulo, mediante hojas de estilo.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc


**Figura 8 – Validador de HTML**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 1.2 Elementos básicos del lenguaje

Un documento HTML esta compuesto por elementos que representan estructuras o comportamientos deseados, como por ejemplo, párrafos, vínculos, listas, etc. Una declaración de un elemento consta generalmente de tres partes: etiqueta inicial, contenido y etiqueta final.

```
<nombre-elemento>
[ … contenido … ]
</nombre-elemento>
```

```
<p>
        Hola mundo
</p>
```

Dentro de esta regla general existen algunas variantes que se permitan, aunque no es recomendable utilizarlas, estas son:

1. Algunos tipos de elementos permiten a los autores omitir las etiquetas finales (por ejemplo elemento (por ejemplo P y LI).
2. Otros tipos de elementos permiten omitir las herramientas iniciales (HEAD y BODY).
3. Existen otros elementos, como por ejemplo BR que no tiene contenido y su único papel es determinar un salto de línea.

Cada elemento a su vez puede tener una serie de atributos asociados, todas estos atributos (parejas atributo/valor) se declaran en la etiqueta inicial, después del nombre del elemento. El valor del atributo aunque puede ir sin ningún tipo de comillas, pero se recomienda utilizar siempre comillas (simples o dobles).

```
<nombre-elemento nombre-atributo=”valor”>
[ … contenido … ]
</nombre-elemento>
```

```
<p id=”miPrimerParrafo">
        Hola mundo
</p>
```

### 1.2.1 Caracteres individuales o especiales

Dentro del contenido del elemento pueden aparecer diferentes caracteres especiales, que pueden incluirse dentro de un elemento HTML, como por ejemplo el símbolo &, o el punto y como (;), para expresar estos caracteres de forma que siempre se visualicen correctamente desde los navegadores se recomienda utilizar referencias a estos caracteres, existen dos tipos de referencias:

- Referencias numéricas (decimal o hexadecimal).
- Referencias a entidades de caracteres.

| Carácter | Referencia Numérica | Referencia a entidades | Descripción |
|---|---|---|---|
| á é í ó ú | | á é í ó ú | Vocales en minúsculas con tilde |
| Á É Í Ó Ú | | Á É Í Ó Ú | Vocales en mayúsculas con tilde |
| Ñ, ñ | | Ñ ñ | Letra eñe |
| “ | &#34; | &quot; | Comillas dobles |
| # | &#35; | | Almohadilla |
| $ | &#36; | | Dólar |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | | | |
|---|---|---|---|
| % | &#37; | | Tanto por ciento |
| & | &#38; | &amp; | Ampersand |
| ‘ | &#39; | | Comillas simples |
| < | &#60; | &lt; | Menor que |
| > | &#61; | | Igual |
| = | &#62; | &gt; | Mayor que |
| ¿ | &#191; | &iquest; | Interrogación |
| @ | &#64; | | Arroba |
| © | &#169; | &copy; | Copyright |
| ® | &#174; | &reg; | Registrado |

### 1.2.2 Colores

Un valor de color puede ser o bien un número hexadecimal (anteponiendo un signo #) según la descripción propuesta por el modelo RGB o uno de los siguientes dieciséis nombre de colores:

Black = "#000000" (Negro)
Silver = "#C0C0C0" (Plateado)
Gray = "#808080" (Gris)
White = "#FFFFFF" (Blanco)
Maroon = "#800000" (Marrón)
Red = "#FF0000" (Rojo)
Purple = "#800080" (Púrpura)
Fuchsia = "#FF00FF" (Fucsia)
Green = "#008000" (Verde)
Lime = "#00FF00" (Verde lima)
Olive = "#808000" (Verde oliva)
Yellow = "#FFFF00" (Amarillo)
Navy = "#000080" (Azul marino)
Blue = "#0000FF" (Azul)
Teal = "#008080" (Azul verdoso)
Aqua = "#00FFFF" (Celeste)

**Figura 9 - Códigos de color**

La descripción del modelo RGB hace referencia a la composición de color en términos de intensidad de colores primarios con que se forma: rojo, verde y azul. La combinación de ellos da lugar a diferentes colores, como por ejemplo el color negro es RGB(0, 0, 0), el verde es RGB (0, 255, 0), el marrón es RGB(200, 100, 0), etc. Decimos que en HTML se puede utilizar un código hexadecimal (precedido de “#”), ya que un color concreto en el modelo RGB se puede representar utilizando al notación hexadecimal, según la formula:

**Formula general:**

R/16 = x + y/16

G/16 = x' + y'/16

B/16 = x'' + y''/1

**Fórmula para RGb (80, 6, 143):**

```
80/16 = 5 + 0/16
6/16 = 0 + 6/16
143/16 = 8 + 15/16;
```

**Resultado: 500023**

Aunque los colores pueden añadirse cantidades significativas de información a los documentos y hacerlos más legibles, hay que tener en cuenta las siguientes puntualizaciones:



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- El uso de elementos y atributos HTML para especificar colores esta desaprobado. Se aconseja utilizar hojas de estilo.
- No utilizar combinaciones de colores que puedan causar problemas a personas con dificultades para distinguir colores.
- Los colores especificados tienen apariencias diferentes en plataformas diferentes (Windows, Macs, paneles LCD).

### 1.3 Estructura básica de un documento HTML

La estructura de un documento escrito en lenguaje claramente definida, consta siempre de estas tres partes:

1. Una línea que contiene información sobre la versión de HTML.

Una sección de cabecera del documento delimitada por el elemento HEAD

2. Un cuerpo, en el que esta el contenido que posteriormente se verá desde el navegador. Esta sección se delimita mediante el elemento BODY.

A continuación veremos un pequeño ejemplo para entender mejor esta estructura:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
http://www.w3.org/TR/html4/strict.dtd">
<html>
        <head>
                <title>Experto en comercio electrónico</ti-
tle>
        </head>
        <body>
                Bienvenido a mi primara página en lenguaje
```

Como vemos en el ejemplo, la primera línea de todo documento en lenguaje HTML tiene que contener la versión del documento HTML utilizado, como hemos anteriormente existen tres versiones diferentes, cada una con una identificación diferente, veamos:

- HTML 4.01 estricto que incluye todos los elementos que no han sido desaprobados.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
```

- HTML 4.01 Transitional, incluye todos los elementos de la versión anterior más los elementos desaprobados.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
        "http://www.w3.org/TR/html4/loose.dtd">
```

- HTML 4.01 Frameset, incluye toda la versión anterior más los elementos de construcción e marcos.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
      "http://www.w3.org/TR/html4/frameset.dtd">
```

También podemos ver en el ejemplo anterior, el resto del documento está contenido dentro del elemento HTML, este a su vez contiene el encabezado del documento (HEAD) y el cuerpo (BODY).

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 1.3.1 El encabezado (elemento HEAD)

La cabecera en un documento HTML contiene información sobre el documento actual (título, keywords, autor, descripción, etc.). El contenido de esta sección, no se muestra a los usuarios cuando acceden al documento, pero resultada de vital importancia para la indexación en los motores de búsqueda, como por ejemplo Google o Yahoo.
En esta sección se incluye el elemento TITLE, elemento donde se especifica el título del documento. Se recomienda utilizar títulos que describan el contexto del documento HTML, es decir, por ejemplo el título "Aspectos relevantes" no proporciona mucha información acerca del contexto, pero en cambio con el título "HTML – Aspectos relevantes" se sitúa mejor a los usuarios en el tema del documento.
Los metadatos también se incluye en el encabezado del documento, un metadato contiene información sobre el contenido del propio documento, la estructura de un metadato es la siguiente:
`<meta name="[propiedad]" content="[contenido]">`
Existe gran cantidad variedad de metadatos, en la siguiente tabla se presentan algunos de ellos.

| **Metadatos** | **Descripción** |
|---|---|
| keyword | Cada documento debe tener palabras clave, que sirvan para describir el documento. |
| description | Descripción del documento HTML |
| author | Autor del documento |
| Content-Type | Tipo de contenido de la declaración y codificación de caracteres |
| language | Lenguaje del documento. Para España y en es español: es_es |
| robots | Proporciona instrucciones a los rastreadores, acerca de cómo indexar |
| copyright | Copyright si lo hubiera. |
| Content-Language | Lenguaje del documento HTML |
| geo.placement | Posición geográfica de la organización |
| geo.position | Posición geográfica en coordenadas de la organización |

Los metadatos suelen ser utilizados por los motores de búsqueda para mejorar la calidad de los resultados de la búsqueda, cuando se proporciona varios elementos META con información en varios idioma, el atributo lang sirve como filtro para mostrar las preferencias del idioma del usuario.

```
<!-- Para hablantes de inglés británico -->
        <meta name="keywords" lang="en" content="car, rent">
<!-- Para hablantes de español -->
        <meta name="keywords" lang="es" content="coche, alqui-
        ler">
```

Por ultimo, señalar también en el encabezado vínculos a otras páginas y códigos en otros lenguajes (scripts), ambos los veremos en secciones posteriores de este módulo.

### 1.3.2 El cuerpo (elemento BODY)

El cuerpo del documento incluye el contenido del documento, es decir, la información que se le presenta al usuario final. Hay que tener en cuenta, que existe diferentes tipos de agentes de usuario


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

(navegadores) y en cada uno se presenta el contenido de una forma diferente, es decir, para navegadores visuales, se presentan texto, imágenes, colores, etc. pero para navegadores de voz, el mismo contenido podría ser pronunciado. Debido a este motivo cualquier tipo de referencia a la presentación (diseño) del documento debe realizarse mediante hojas de estilos y nunca debe incrustarse directamente en el documento HTML.
El cuerpo del documento se define mediante el elemento BODY, que irá siempre a continuación del elemento de encabezado.

```
<html>
	<head></head>
	<body>
		[…]
		[CONTENIDO DEL DOCUMENTO HTML]
		[…]
	</body>
</html>
```

Los elementos que forman parte de la sección de contenido del documento en lenguaje HTML se identifican mediante los atributos id y class:

- El atributo id, es un identificador único para cada elemento y tiene las siguientes funciones:
  - Selector para hojas de estilo.
  - Vinculo de destino para vínculos de hipertexto.
  - Medio de referenciar un elemento desde un script.
  - Nombre de un elemento object previamente declaro.
  - Para procesos generales por parte de agentes de usuario.

El elemento id, comparte el mismo espacio de nombre que el atributo name, por lo que se recomienda que estos dos atributos sean idénticos para el mismo elemento.

- El atributo class, se puede asignar a más de un elemento, creando grupos de elementos que pertenecen a un mismo grupo class, tiene las siguientes papeles dentro de HTML:
  - Selector para hojas de estilo (de todos los elementos integrantes de la clase).
  - Para procesos generales por parte de agentes de usuario.

Veamos un ejemplo para entender mejor el funcionamiento de los atributos id y class. Imaginemos un documento HTML que contiene dos párrafos, el primer párrafo será rojo, en mayúsculas y en negrilla, mientras que el segundo será gris y también estará en negrilla. Por un lado, ambos párrafos pertenecerán a la clase que transforma la tipografía en negrilla (clase “negrilla”, en el ejemplo), pero sólo el primero pertenecerá a la que transforma el texto en mayúsculas (clase “mayusculas”, en el ejemplo). Por otro lado, cada párrafo tendrá un identificador diferente, al identificador del primer párrafo se le asignará el color de la fuente rojo y mientras que al del segundo párrafo se le asignará el color de fuente gris. Es decir, tendremos dos elementos de párrafo, con identificadores diferentes, ambos pertenecen a la clase que define el texto como negrilla, pero sólo el primero pertenecerá a la clase que transforma el texto en mayúsculas. El código HTML asociado a este ejemplo se puede ver en la figura siguiente [6-8].

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
<html>
<head>
   <title>Ejemplo atributos id y class
   en html</title>
   <link rel="stylesheet"
   type="text/css" href="css/estilo.css"
   />
</head>
<body>
   <p id="parrafoRojo" class="negrilla
   mayusculas">
      Lorem ipsum dolor sit amet, con-
      sectetur adipiscing elit. Nulla
      eget eros neque. Cras volutpat fa-
      cilisis.
   </p>
   <p id="parrafoGris" class="negrilla">
      Lorem ipsum dolor sit amet, con-
      sectetur adipiscing elit. Aenean
      lacinia magna quis.
   </p>
</boyd>
</html>
```

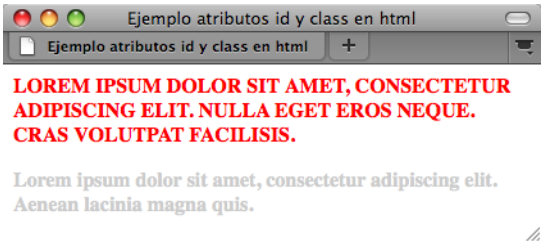



**Figura 10 - Ejemplo HTML: id y class**

Dentro de los elementos que puede utilizarse dentro del cuerpo de un documento HTML (etiqueta BODY), podemos diferenciar dos tipos diferentes de elementos:

- **Elementos a nivel de línea.** Pueden contener datos y/o otros elementos de línea. Este elemento se especifica en el lenguaje HTML mediante el elemento SPAN.
- **Elementos a nivel de bloque.** Pueden contener otros elementos de bloque y/o elementos de líneas. En HTML se utiliza el elemento DIV para definir bloques de elementos.

Mediante los elementos DIV y SPAN junto con los atributos id y class, el lenguaje HTML ofrece un mecanismo genérico para definir una estructura en el documento, aunque no imponen ningún estilo de presentación de contenido, tan solo especifican si el contenido es en línea o en bloque.

El último elemento estructural dentro de un documento HTML, son los encabezados, elementos H1, H2, H3, H4, H5 y H6. Un encabezado describe brevemente el tema sección que se introduce. Existen seis niveles de encabezados en HTML, siendo H1 el más importante y el H6 el menos importante, los navegadores normalmente representan los encabezados más importantes con un tipo de fuente más grande que los menos importantes, aunque la presentación es totalmente configurable mediante hojas de estilos.

```
<html>
<head>
       <title>Ejemplo encabezados</title>
</head>
<body>
       <div id="seccionPrincipal">
              <h1>T&iacute;tulo de la secci&oacute;n</h1>
              <p>
                     Lorem ipsum dolor sit amet, consectetur adipiscing elit.
                     Aenean lacinia magna quis.
              </p>
              <h2>Subt&iacute;tulo de sección</h2>
              <p>
```



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
                        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
                        Donec faucibus adipiscing erat sit amet accumsan. Cras
                        adipiscing sagittis bibendum. Donec eu risus id mi semper.
            </p>
                  <h2>Subt&iacute;tulo de sección 22</h2>
                        <h3>Subtítulo de sección 2</h3>
                        <p>
                              Lorem ipsum dolor sit amet, consectetur adipiscing
                              elit. Donec faucibus adipiscing erat sit amet ac-
                              cumsan.
                        </p>
            </div>
</body>
</html>
```

En el ejemplo anterior se define un bloque mediante el elemento DIV, este bloque contiene un encabezado principal (H1) y también contiene un secundario (H2), que contiene a su vez un encabezado H3.
Se suele considerar un error saltarse niveles en el encabezado, ya que dificulta la visión global de la estructura del documento HTML, es decir, esta permitido <H1> <H2> <H3>, pero no esta permitido <H1> <H3> <H3>

## 1.4 Texto estructurado

En esta sección vamos a detallar los elementos del lenguaje utilizados para estructurar el texto de la página, sin tener en cuenta otros elementos relacionados con la presentación del mismo (alineación, fuentes, hojas de estilo, etc.).

### 1.4.1 Elementos de frase

Los elementos de frase añaden información estructural al fragmentos de texto, tal y como se aprecia en la siguiente tabla.

| **Elemento** | **Función** |
|---|---|
| EM | Énfasis |
| STRONG | Énfasis mayor |
| CITE | Contiene una cita o referencia |
| DFN | Indica que es aquí donde se define el término encerrado |
| ABBR | Indica una forma abreviada (HTML, URI, etc.) |
| ACRONYM | Indica un acrónimo (radar, WAC, etc). |

Los elementos EM y STRONG se usan para indicar énfasis, la presentación de ambos dependerá del agente de usuario (navegador), en general, se presenta el texto EM en cursiva o itálica y el elemento STRONG en negrita. No se deben utilizar estos elementos para alterar la presentación del texto.
Por su parte, los elementos ABBR y ACRONYM permiten a identificar claramente la aparición de abreviaturas y acrónimos. El contenido de los elementos ABBR y ACRONYM especifican la expresión abreviada, tal y como aparece en el texto, mientras que se utiliza el atributo tittle de estos elementos para proporcionar la forma expandida de la expresión.

```
<p>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
        Las oficinas centrales del <ACRONYM title=”Federal Bureau of Investiga-
        tion”>F.B.I.</ACRONYM> estan situadas en Washington.
</p>
```

### 1.4.2 Citas

Las citas se representan mediante los elementos BLOCKQUOTE, para citas largas (bloques) y mediante Q para citas cortas (contenido de línea), se utiliza el atributo cite para designar el enlace a la página que designa el documento o mensaje original.

Los agentes de usuario representan el elemento BLOCKQUOTE como un bloque con sangría y el contenido del elemento Q mediante marcas de citación a su alrededor (comillas dobles), por lo que no se deben colocar marcas de cita alrededor.

```
<blockquote cite="http://es.wikipedia.org/wiki/El_Padrino">
              Algún, día, y puede que ese día no llegue nunca, iré a pedirte un favor.
</blockquote>
<p>
              Como una vez dijo el Don: <q>Algún, día, y puede que ese día no llegue
              nunca, iré a pedirte un favor.</q>
</p>
```

### 1.4.3 Líneas y párrafos

La forma en que HTML define un párrafo se realiza mediante el elemento del lenguaje P, en cambio, la presentación visual de los mismos no es tan simple, hay que tener varios aspectos, tanto de estilo como técnicos:

- Tratamiento del espacio en blanco.
- Saltos de línea y cambio automático de línea.
- Justificación.
- División de palabras.
- Convenciones del idioma escrito y direccionalidad del texto.
- Formateo de los párrafos con respecto al contenido circundante.

Todas estas cuestiones las trataremos a continuación a excepción de la alineación y los objetos flotantes que los veremos más adelante, durante el apartado de CSS.

Como hemos mencionado el elemento P representa a un párrafo, este es único y no puede contener a otros elementos en bloque (incluyendo otros párrafos). Para indicar separaciones de línea (retorno de carro) se utiliza el elemento BR, este elemento finaliza forzosamente la línea actual de texto. Se desaconseja a los autores el uso de elemento P vacíos, ya que los navegadores no deberían tener en cuenta los elementos P vacíos.

El modo en que los párrafos se re representan visualmente depende del agente de usuario. Normalmente los párrafos se representan alineados a la izquierda y con un espacio en blanco antes y después del mismo. Convencionalmente, los agentes de usuario de HTML visuales ajustan automáticamente las líneas de texto a la anchura disponible entre los márgenes. Los algoritmos de ajuste automático de línea dependen del tipo de escritura a que se esté dando formato (escrituras occidentales, hebreas, etc.).


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 1.4.4 Listas

El lenguaje HTML ofrece varios mecanismos para especificar listas de información, diferenciándose tres tipos diferentes:

- **Información no ordenada.** Se crea mediante el elemento UL, se utilizan para crear listas de ítems sin una secuencia determinada.
- **Información ordenada.** Mediante el elemento OL, se utilizan principalmente para presentar información para la cuál es importante el orden de sus elementos.
- **Lista de definiciones.** Mediante el elemento DL, consisten en una serie de parejas término/definición.

Las listas ordenadas y no ordenadas se crean de la misma manera, se utiliza el elemento OL o UL para definir el inicio de la lista y LI para especificar cada objeto (ítem) de la lista, tal y como se aprecia en el siguiente ejemplo.

```
<p>Lista no ordenada:</p>
<ul>
       <li>valor 1</li>
       <li>valor 2</li>
</ul>

<p>Lista ordenada:</p>
<ol>
       <li>Primer valor</li>
       <li>Segundo valor</li>
</ol>
```

Lista no ordenada:

- valor 1
- valor 2

Lista ordenada:

1. Primer valor
2. Segundo valor

Por su parte, las listas de definiciones (elemento DL) sólo se difieren de las otras listas, en que los objetos de la lista consisten en dos pares: un término y una descripción. El término se expresa mediante el elemento DT y está restringido a contenidos en línea, la descripción se especifica mediante el elemento DD que contiene elementos de bloque.

```
<p>Lista de definiciones:</p>
<dl>
   <dt>SEM</dt>
   <dd>Trata todo lo relacionado con la promoción y aparición en los buscadores</dd>
   <dt>SEO</dt>
   <dd>Optimizar las páginas web para que aparezcan en los primeros resultados de
búsqueda de los buscadores de Internet.</dd>
</dl>
```

Lista ordenada:

SEM
: Trata todo lo relacionado con la promoción y aparición en los buscadores

SEO
: Optimizar las páginas web para que aparezcan en los primeros resultados de búsqueda de los buscadores de Internet.

En las listas ordenadas se puede inicializar indicando mediante atributo value del elemento OL, la numeración de los objetos subsiguientes de la lista continúa a parte del nuevo valor.

```
<ol>
  <li value="3"> objeto de lista número 30.</li>
  <li value="4"> objeto de lista número 40. </li>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
  <li> lista número 5. </li>
</ol>
```

Tanto para las listas ordenadas, como para las listas no ordenadas se puede utilizar también el atributo type que define las opciones de representación para los agentes de usuario visuales.

| **Tipo de listado** | **Valor** | **Definición** |
|---|---|---|
| **Lista no ordenada UL** | disc | Círculo relleno |
| | circle | Círculo vacío |
| | square | Cuadrado vacío |
| **Lista ordenada OL** | 1 | Números arábigos |
| | a | Alfabética en minúsculas |
| | A | Alfabética en mayúsculas |
| | i | Números romanos en minúsculas |
| | I | Número romanos en mayúsculas |

Para finalizar este apartado de listas en HTML, tenemos que señalar que estos tres tipos de listado se pueden anidar, pudiéndose utilizar al mismo tiempo diferentes tipos de listas anidadas, tal y como se muestra en el siguiente ejemplo.

```
<dl>
        <dt><strong>Ingredientes:</strong></dt>
        <dd>
                <ul>
                        <li>100 g de harina</li>
                        <li>10 g de azúcar</li>
                        <li>1 taza de agua</li>
                        <li>2 huevos</li>
                        <li>sal, pimienta</li>
                </ul>
        </dd>
        <dt><strong>Pasos a seguir:</strong></dt>
        <dd>
                <ol>
                        <li>Mezcle los ingredientes secos íntima-
                        mente.</li>
                        <li>Vierta los ingredientes líquidos.</li>
                        <li>Remueva durante 10 minutos.</li>
                        <li>Hornear durante una hora a 300 grados.</li>
                </ol>
        </dd>
        <dt><strong>Notas:</strong></dt>
        <dd>Puede añadir uvas para mejorar la receta.</dd>
</dl>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 1.5 Tablas

El modelo de tablas de HTML permite a los programadores organizar el resto del contenido (texto, imágenes, texto preformateado, imágenes, vínculos, formularios, campos de formularios, otras tablas anidadas, etc.), pero aunque HTML permita utilizar tablas para organizar el contenido de nuestra página Web, actualmente esta es una técnica totalmente desaconsejable, ya que usualmente se utilizan para esta tarea los elementos de bloque y de línea (DIV y SPAN, respectivamente), dejando la funcionalidad de las tablas únicamente para mostrar datos debidamente organizados a los usuarios de nuestros documentos HTML.
Las tablas se designa mediante el elemento TABLE, las filas mediante el elemento TR y por último para especificar las celdas de cada fila se utiliza el elemento TD para las celdas que contienen datos y el elemento TH para las celdas sean encabezados. Resulta importante diferenciar las celdas de datos de las que son encabezados para que así los agentes de usuario NO visuales puedan diferenciar más rápidamente la información de los encabezados y así puedan mostrar la información de manera más precisa al usuario. A continuación, vemos un ejemplo simple, donde la primera fila contiene dos celdas de encabezado y la segunda dos celdas de datos:

```
<table>
	<tr>
		<th>Celda 1.1 </th>
		<th>Celda 1.2 </th>
	</tr>
	<tr>
		<td>Celda 2.1 </td>
		<td>Celda 2.2 </td>
	</tr>
</table>
```

Las filas de una tabla se dividen en secciones (encabezado, pie y cuerpo), lo que permite asociar una información estructural a la tabla, que posteriormente es utilizada por los agentes de usuario para enfatizar esta estructura. Para especificar esta secciones se utilizan los elementos:

- Encabezado de la tabla, que se especifica mediante el elemento THEAD.
- El contenido de la tabla, es decir, dónde van los datos que se le quieren mostrar al usuario, determinado por el elemento TBODY
- Pie de la tabla, el elemento utilizado para determinar esta sección TFOOT.

```
<table>
	<thead>
		<tr>
			<th>Nombre</th>
			<th>Apellidos</th>
			<th>Tel&eacute;fono</th>
		</tr>
	</thead>
	<tfoot>
		<tr>
			<td colspan="3">N&uacute;mero de usuarios: 2</td>
			</tr>
	</tfoot>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
        <tbody>
                <tr>
                        <td>Fernando</td>
                        <td>Llorente Garc&iacute;a</td>
                        <td>666666666</td>
                </tr>
                <tr>
                        <td>Lu&iacute;s</td>
                        <td>M&aacute;rquez Dom&iacute;nguez</td>
                        <td>999999999</td>
                </tr>
        </tbody>
</table>
```

De la misma manera, que se permite definir una estructura para las filas de una tabla también se permite agrupar de columnas y así crear nuevas divisiones estructurales que permitan un tratamiento mejor por parte de los agentes de usuario, para esta tarea se utilizan los elementos COLGROUP y COL. El funcionamiento de estos atributos es simple:

- Utilizamos COLGROUP cuando se desea especificar explícitamente un grupo de columnas, el número de columnas se especifica mediante el atributo span.
- Utilizamos el elemento COL, si queremos agrupar columnas especificaciones de atributos para columnas, es decir no agrupa estructuralmente grupos de columnas. Puede aparecer dentro o fuera de un grupo explícito de columnas (COLGROUP).

```
<table>
        <colgroup span="4">
                <col span="3" />
                <col width="2" />
        </colgroup>
        <colgroup span="5" width="15" />
        <thead>
                 <TR>
                         <TD>
                [...]
</table>
```

El modelo de tablas de HTML ha sido diseñado de modo que los agentes de usuario puedan representar las tablas incrementalmente (es decir, a medida que reciben el contenido de los datos) y no cuando hayan recibido toda la tabla completamente. Para ello es necesario que los programadores o autores proporcionen a los navegadores la información del número de columnas y su anchura mediante los elementos COLGROUP y COL.

Como ya hemos comentado el elemento TABLE contiene a todos los demás elementos que forman la tabla, además de estos, el elemento puede contener una serie de atributos que utilizan los agentes de usuario utilizan para representar mejor la tabla:

- El atributo summary, utilizado para especificar el resumen del contenido y la estructura de la misma, principalmente utilizado para el uso de los agentes visuales.
- El atributo title, para representar el título de la tabla.
- El atributo cellspacing y cellpadding, que indican el espacio entre celdas y el margen de la celda, respectivamente.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- El atributo width, indica el ancho de la tabla.
- El atributo border que determina la anchura del borde de la tabla.

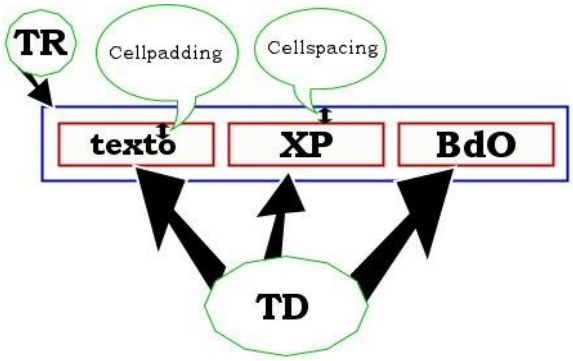



**Figura 11 - Código HTML: TR y TD, cellpading y cellspacing**

Existe otro elemento de presentación de las tablas del que todavía no hemos hablado, este es el elemento CAPTION, utilizado para describir las tablas. Este elemento debería contener un resumen de la naturaleza de la tabla y sólo se permite su uso justo a continuación del elemento TABLE y se les presenta únicamente a los usuarios visuales, Por lo que los autores, deberían esforzarse en proporcionar información adicional resumiendo el propósito y la estructura de la tabla utilizando el atributo summary del elemento TABLE.
Por último, tenemos a hablar de las celdas que pueden abarcar más de una fila o más de una columna. El número de filas o de columnas abarcado por una celda se establece con los atributos rowspan y colspan de los elementos TH y TD. Veamos a continuación una serie de ejemplos:

```
<table>
<caption>
       Ejemplo atributo colspan
</caption>
       <tr>
              <th></th>
              <th></th>
              <th></th>
       <tr>
              <td></td>
              <td colspan="2"></td>
       </tr>
       <tr>
              <td      colspan="2"></td>
              <td></td>
       </tr>
</table>
```

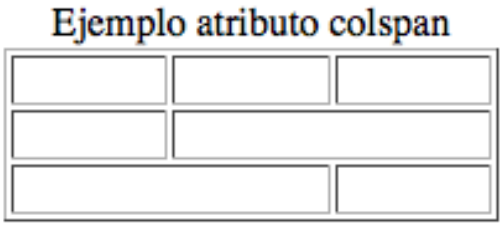



**Figura 12 - Atributo colspan**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
<table>
<caption>
       Ejemplo atributo rowspan
</caption>
       <tr>
              <th rowspan=”2”></th>
              <th></th>
              <th></th>
       <tr>
              <td></td>
              <td rowspan="2"></td>
       </tr>
       <tr>
              <td></td>
              <td></td>
       </tr>
</table>
```

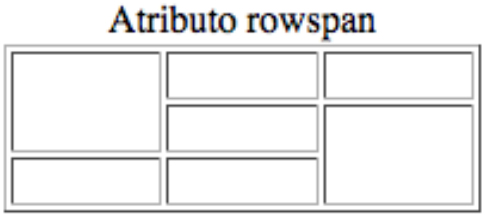



**Figura 13 - Atributo rowspan**

## 1.6 Vínculos

HTML ofrece muchas de las posibilidades de publicación convencionales para la creación de texto enriquecidos y documentos estructurados, pero lo que lo separa de la mayoría otros lenguajes son sus características para crear hipertextos y documentos enriquecidos. En esta sección presentaremos los vínculos (hipervínculo o enlace), la estructura básica del hipertexto. Un vínculo es una conexión desde un recurso Web a otro. Aunque es un concepto simple, el vínculo ha sido una de las principales fuerzas que ha hecho posible el éxito de la Web [9-11].

Un vínculo tiene dos extremos (llamados en inglés *anchors*, anchas), y una dirección. El vínculo comienza en el ancla de origen y apunta al ancla de destino, que puede ser cualquier recurso de la Web, el comportamiento por defecto asociado a un vínculo es la obtención de este recurso. Veamos el siguiente ejemplo, que utiliza el elemento A para especificar los vínculos:

```
<p>
              El enunciado al ejercicio lo podr&aacute;s encontrar en el <a
              href=”tema7.html”>tema 7</a> del libro de pr&aacute;cticas. Este <a
              href=”graficoTemperatura.png”>gr&aacute;fico</a> te puede ayudar en la
              resolución del mismo.
</p>
```

En el ejemplo anterior, tenemos dos vínculos, el primero dirigido a otra página HTML, mientras que el segundo dirigido a una imagen en forma PNG. Como se puede observar el atributo href de cada origen del vínculo especifica la dirección del destino por medio de una URI (dirección Web). Estos destinos se definen mediante una URI, que ya vimos al inicio del documento, veamos diferentes ejemplos:

- Un URI absoluto: `http://bisite.usal.es/misArchivos/imágenes/pagina.html#sec1`
- Un URI relativo: `./misArchivos/imágenes/pagina.html#sec1`
- Cuando el vínculo está definido en el mismo documento: `#sec1`


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

El destino del vínculo puede ser un documento HTML completo, o una fragmento de este. Para que un vínculo apunte a un fragmento de un documento, en vez de al documento completo, se fija como condición necesario la unicidades de los nombres de los fragmentos de documentos HTML, es decir, no se les puede dar el mismo nombre a dos fragmentos de código diferentes, mediante el elemento id o name. Los destinos de los vínculos a fragmentos de un documento HTML se pueden realizar de dos formas:

- Especificando el fragmento mediante el elemento A y utilizando el atributo name para hacer la referencia.
- Mediante cualquier otro elemento, utilizando para ello el atributo id de cada elemento.

Así, por ejemplo, un autor podría crear una tabla de contenidos cuyas entradas estuvieran vinculadas con los encabezados de cada sección del mismo documento.

```
<h1 id=”index”>tabla de contenidos</h1>
    <ul>
       <li><a href=”#seccion1”> Sección
       1</a></li>
       <li><a href=”#seccion2”> Sección
       2</a></li>
    </ul>
<h1 id=”seccion1”>Sección 1.</h1>
   […]
<a href=”#index”>Volver al indice</a>
<h2 id=”seccion1-1”>Sección 1.1.</h1>
   […]
<a href=”#index”>Volver al indice</a>
<h1 id=”seccion2”>Sección 2.</h1>
   […]
```

```
<h1><a name=”index”>tabla de conteni-
dos</a></h1>
    <ul>
       <li><a    href=”#seccion1”>Sección
       1</a></li>
       <li><a    href=”#seccion2”>Sección
       2</a></li>
    </ul>
<h1 id=”seccion1”>Sección 1.</h1>
  […]
<a href=”#index”>Volver al indice</a>
<h2><a          name=”seccion1-1”>Sección
1.1</a></h1>
  […]
<a href=”#index”>Volver al indice</a>
<h1><a            name=”seccion2”>Sección
2.</a></h1>
  […]
```

Por otro lado, totalmente diferente, vamos a hablar del atributo title de los elemento A, este elemento se utiliza para añadir información sobre la naturaleza del vínculo. Esta información puede ser pronunciada por un agente de usuario, representada como una indicación visual (“tool-tip”), etc.

```
<a target="_blank"
href="http://www.usal.es" title="Página
de la Universidad de Salamanca">
   <img src="LogoUsalCabecera.png"
   alt="Universidad de Salamanca">
</a>
```


**Figura 14 - Atributo title**

Además del elemento A que ya hemos visto, existe otro elemento destinado a especificar vínculos, este es el elemento LINK, ambos se utilizan para definir vínculos. Pero la diferencia existente entre ambos es que el elemento LINK sólo puede utilizar en el encabezado del documento (elemento HEAD), mientras que el elemento A sólo puede utilizarse en el cuerpo del documento (elemento BODY).

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

Los vínculos especificados con LINK no son representados visualmente en el navegador de los usuarios (ningún elemento contenido por HEAD se presenta). El elemento LINK, no tiene contenido, se utiliza para especificar relaciones entre documentos HTML que son utilizados por los agentes de usuario.
Los atributos rel y rev del elemento LINK juegan papeles complementarios: el atributo rel especifica un vínculo directo y el atributo rev especifica un vínculo inverso. Este tipo de vínculos resulta muy útil para los buscadores de Internet, ya que permiten estructurar los diferentes recursos autoreferenciados.

```
<link rel="index" href="index.html" />
```

Para indicar la página índice de la página actual.

```
<link rev="index" href="pagina1.html" />
```

Para indicar *desde* el index, que la página actual es índice de la página a la que te refieres.
También podemos indicar otra serie de información como por ejemplo:

| | |
|---|---|
| Diferentes versiones de un documento | `<LINK title="El manual en neerlandés" type="text/html" rel="alternate" hreflang="nl" href="http://algunsitio.com/manual/neerlandes.html">` |
| Edición de impresión | `<LINK media="print" title="El manual en postscript" type="application/postscript" rel="alternate" href="http://algunsitio.com/manual/postscript.ps">` |
| Portada del documento | `<LINK rel="Start" title="La primera página del manual" type="text/html" href="http://algunsitio.com/manual/portada.html">` |

El elemento LINK también es utilizado para vincular hojas de estilo externas al documento, el atributo type, especifica el lenguaje de la hoja de estilo, y el atributo media especifica el medio o medios de representación deseados.

```
<link href="css/portada.css" rel="stylesheet" type="text/css" media="screen" />
```

Los diferentes elementos media se utilizan para ahorrar tiempo obteniendo sólo aquellas hojas de estilo que se apliquen al dispositivo actual. Veamos los diferentes tipos de medios:

- screen. Para pantallas no paginadas de computadora.
- tty. Para medios que utilicen una cuadrícula de caracteres de ancho fijo, como teletipos, terminales y dispositivos portátiles con posibilidades limitadas de representación.
- tv. Para dispositivos tipo televisión (baja resolución, en color, desplazamiento limitado).
- projection. Para proyectores.
- handheld. Para dispositivos de mano (pantalla pequeña, monocromos, gráficos por mapas de bits, ancho de banda limitado).
- print. Para material paginado, opaco, y para documentos que se ven en una pantalla en modo de presentación preliminar a la impresión.
- braille. Para dispositivos táctiles braille.
- aural. Para sintetizadores de voz.
- all. Apropiado para todos los dispositivos.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 1.7 Objetos, Imágenes y Aplicaciones

Las características multimedia de HTML permiten a los autores incluir en sus páginas diversos recursos multimedia como imágenes, aplicaciones (programas que se descargan automáticamente), videoclips, y otros documentos HTML en sus páginas.
Las versiones anteriores de HTML permitían a los autores incluir imágenes (por medio del elemento IMG) y aplicaciones (por medio de APPLET). Estos elementos tienen varias limitaciones:

- No sirven para resolver el problema más general de cómo incluir tipos de medios nuevos y futuros.
- El elemento APPLET sólo funciona con aplicaciones basadas en el lenguaje de programación Java. Este elemento esta desaprobado.
- Plantean problemas de accesibilidad.

Para solucionar estos problemas, HTML 4 introduce un nuevo elemento, este es el elemento OBJECT, que ofrece una solución universal para la inclusión de objetos genéricos, permitiendo especificar todos aquellos datos que necesitan los agentes de usuario para la representación del objeto: código fuente, valores iniciales, etc. A partir del nacimiento de este nuevo objeto, se empiezan a utilizar el término "objeto" para hablar de recursos que se quieren utilizar en un documento Web (applets, plug-ins, sonido, imágenes, etc.).

| **Tipo de inclusión** | **Elemento específico** | **Elemento genérico** |
|---|---|---|
| Imagen | IMG | OBJECT |
| Aplicación | APPLET (desaprobado) | OBJECT |
| Otro documento HTML | IFRAME | OBJECT |

Como vemos en la tabla anterior, cada tipo de inclusión tiene una solución específica y una general, pero el elemento OBJECT siempre servirá como método de inclusión de tipos de medios futuros.
Para incluir imágenes, se suele utilizar el elemento IMG, este elemento no tiene contenido, los navegadores únicamente se reemplazan la línea de código por la imagen que designa su atributo src, también se suele utilizar el atributo alt que contiene el texto alternativo cuando la imagen no puede ser mostrada y el atributo longdesc para utilizar una descripción más completa de la imagen.

```
<img src="./vacaciones/familia.png" alt="Una foto de mi familia
en el lago.">
```

Como hemos comentado previamente, esta imagen también se podría representar de forma genérica mediante el elemento OBJECT.

```
<OBJECT data="http://www.algunsitio.com/Gente/Juan/vacaciones/fa-
milia.png" type="image/png">Una foto de mi familia en el
lago.</OBJECT>
```

### 1.7.1 Elemento OBJECT

La mayoría de agentes de usuario tienen mecanismos para representar tipos de datos comunes como textos, imágenes, colores, fuentes, etc. El elemento OBJECT permite a los autores controlar si los datos deberían ser representados externamente por algún programa, debido a que el agente de usuario no lo pueda representar de forma nativa.

```
<OBJECT data="superficie.gif" type="image/gif">
```

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
        Gráfico de la superficie de los continentes. Puede consultar la <a
        href="descrip.html#superficie"> descripción en modo texto del grá-
        fico</a>.
    </OBJECT>
```

El elemento OBJECT, contiene una serie de atributos que resulta interesante que veamos:

- **classid.** Este atributo puede utilizarse para especificar la localización de la implementación de un objeto mediante un URI (dirección Web). Puede utilizarse junto con el atributo data o como alternativa al mismo.
- **data.** Este atributo puede utilizarse para especificar la localización de los datos del objeto.
- **type.** Especifica el tipo de contenido de los datos especificados por data.
- **archive.** Que especifica una serie de recursos relevantes para el recurso separados por espacios
- **Standby.** Este atributo especifica un mensaje que puede presentar un agente de usuario mientras carga la implementación y los datos del objeto.

Para representar correctamente un objeto puede ser necesario proporcionar información al agente de usuario, esta tarea se realiza mediante el elemento PARAM, que tiene la siguiente estructura:
`<PARAM name=”<valor>” value=”<valor>” valuetype=”<tipo_contenido>” />`
En el caso más general, puede ser necesario que el autor tenga que especificar tres tipos de información diferente, aunque no siempre es necesario representar esta información, los datos de los que hablamos son:

- La implementación del objeto incluido.
- Los datos que deben representarse
- Otros valores que necesite el objeto para su ejecución.

Los elementos OBJECT se representar de acuerdo a las siguientes reglas de precedencia:

- El agente de usuario debe intentar en primer lugar representar el objeto. No debería representar los contenidos del elemento, pero debe examinarlos por si el elemento contiene hijos directos que sean elementos PARAM.
- Si por cualquier razón el agente de usuario no es capaz de representar el objeto, debe intentar representar su contenido.

De estas dos reglas, se extrae una característica significativa del diseño del elemento OBJECT que es la posibilidad para especificar representaciones alternativas del objeto; cada declaración OBJECT anidada puede especificar tipos de contenidos alternativos.

```
<!-- Primero, intentarlo con la aplicación Python -->
<OBJECT title="La Tierra vista desde el espacio"  clas-
sid="http://www.observer.mars/LaTierra.py">
     <!-- Si no, intentarlo con el vídeo MPEG -->
     <OBJECT data="LaTierra.mpeg" type="application/mpeg">
            <!-- Si no, intentarlo con la imagen GIF -->
            <OBJECT data="LaTierra.gif" type="image/gif">
                   <!-- Si no, representar el texto -->
                   <STRONG>La Tierra</STRONG> vista desde el espa-
cio.
            </OBJECT>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

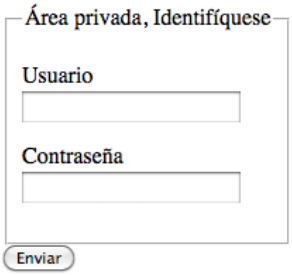
```
        </OBJECT>
    </OBJECT>
```

En el ejemplo anterior, la declaración externa especifica una aplicación que no necesita datos ni valores iniciales. La segunda declaración especifica una animación MPEG, y como no define la localización de ninguna implementación que procese MPEG, depende de que el agente de usuario represente la animación. También establece el atributo type para que los agentes de usuarios que no puedan procesas el formato de video MPEG no intenten descargarlo. La tercera declaración especifica la localización de un imagen en formato GIF y ya por último sino ninguno de los anteriores objetos precedentes se ha podido visualizar se muestra un texto alternativo.

## 1.8 Formularios

Un formulario HTML es una sección de un documento que contiene contenido normal, código, elementos especiales llamados controles. Los usuarios normalmente "rellenan" un formulario modificando sus controles (introduciendo texto, seleccionando objetos, etc.), antes de enviar el formulario a un agente que lo procese.

```
<form id="formAcceso" name="formAcceso" method="post" ac-
tion="validacion">
          <fieldset>
          <legend>Área privada, Identifíquese</legend>
          <p>
                <label for="usuario">Usuario</label><br />
                <input id="usuario" name="usuario" type="text"
                tabindex="1" maxlength="10" />
          </p>
          <p>
                <label for="password">Contraseña</label><br />
                <input id="password" name="password"
                type="password" tabindex="1" />
          </p>
          </fieldset>
          <input type="submit" id="enviar" name="enviar"
          value="Enviar" />
</form>
```




Los usuarios interaccionan con los formularios a través de los llamados controles, el nombre del control viene dado por el atributo name, que será el nombre que utilicemos para referirnos al contenido de un determinado control para el tratamiento de la información posterior al envío del mismo formulario por parte del usuario. Todo control de un formulario debe estar dentro del elemento FORM, actúa como contenedor de controles, es decir delimita el formulario dentro de un documento HTML, este elemento FORM tiene al menos dos atributos que tenemos que conocer:

- action. Especifica cual va a ser el agente que procese el formulario. Este valor se especifica mediante una URI.
- method. Especifica que método http se usará para enviar los datos del formulario, estos valores son:
    - Valor "get", que es el valor por defecto. En este caso los diferentes datos del formulario se agregan al URI especificado por el atributo action, utilizando como signo separador con el resto de la dirección Web una interrogación (?) y separando los diferentes valores de los controles mediante el elemento ampersand (&).

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
http://www.google.com/search?client=safari&rls=en&q=test&ie=UTF-
8&oe=UTF-8
```

- Valor “post”. El conjunto de datos del formulario se incluye en el cuerpo del formulario para que se envíe al agente procesador.

Es decir, sin entrar en detalles, la principal diferentes entre ambos métodos es que el método GET, los parámetros se pueden ver en la barra de dirección y mediante el método POST no son visibles, al menos a simple vista.

Existen diferentes tipos de controles, cada uno utilizado para una función diferente, en las siguientes secciones veremos los principales controles que proporciona HTML.

### 1.8.1 Botones

Los autores, pueden crear tres tipos de botones:

- Botón de envío (submit buttons). Cuando se activa el botón, se envía el formulario. Un formulario puede contener más de un botón de envío
- Botón de reinicialización (reset buttons). Cuando se activa, se reinicializan todos los controles del formulario.

Los autores crean botones con el elemento BUTTON o el elemento INPUT. Veamos como:

- Mediante el elemento INPUT. Para indicar que el botón será de envío el elemento TYPE tiene el atributo submit y para indicar que es un botón de reinicialización, el atributo TYPE tendrá el valor reset.

```
<input type="submit" id="enviar" name="enviar" value="Enviar" />
```


```
<input type="reset" id="enviar" name="enviar" value="Borrar" />
```


- Mediante el elemento BUTTON. Funcionan de forma similar a los botones enviados mediante el elemento INPUT, tal y como vemos a continuación.

```
<button type="submit" name="enviar" id="enviar">En-
viar</button>
```


```
<button type="reset" name="reset" id="re-
set">Borrar</button>
```


Como vemos el procedimiento es más o menos similar, la diferencia entre ambos es que el elemento BUTTON ofrecen posibilidades más ricas de representación ya que puede contener diferentes elementos, como por ejemplo imágenes.

### 1.8.2 Casillas de verificación y radiobotones

Las casillas de verificación (checkbox) y los radiobotones (radiobutton) son interruptores de encendido/apagado, que pueden ser conmutados por el usuario. Ambos están “marcados” cuando se establece el atributo checked del elemento del control. Varias casillas de verificación de un formulario pueden compartir el mismo nombre.

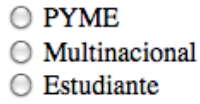
La diferencia entre ambos, es que los radiobotones son mutuamente exclusivos, cuando uno en un grupo esta “marcado”, el resto no lo estarán. Mientras que se pueden marcar varias casillas de verificación de un mismo grupo.

El elemento utilizado para especificar que un control será una casilla de verificación o un radiobotón es el elemento INPUT, para ello, una vez más se utiliza el atributo type. El valor radio de


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

este atributo corresponderá a los radiobotones y el valor checkbox, corresponderá a las casillas de verificación.

```
<input name="tipoTrabajo" value="1" type="radio"/> PYME
<input name="tipoTrabajo" value="2" type="radio"/> Multinacional
<input name="tipoTrabajo" value="3" type="radio"/> Estudiante
```




```
<input name="mercadosEmpresa" value="1" type="checkbox">Local
<input name="mercadosEmpresa" value="2" type="checkbox">Regional
<input name="mercadosEmpresa" value="2" type="checkbox">Nacional
<input name="mercadosEmpresa" value="2" type="checkbox">Unión Europea
<input name="mercadosEmpresa" value="2" type="checkbox">Internacional
```


### 1.8.3 Menus

Los menús ofrecen al usuario la posibilidad de elegir un valor o varios de entre sus elementos. Para especificar este tipo de control se utiliza el elemento SELECT junto con los elementos OPTGROUP y OPTION.

El elemento SELECT es el encargado de crear el menú. Cada opción ofrecida por el menú se representa por el elemento OPTION, mientras que el elemento OPTGROUP se utiliza para formar grupos de opciones.

```
<select name="pais">
       <optgroup label="Europa">
            <option value="1" selected>Espa&ntilde;a</option>
            <option value="2">Francia</option>
            <option value="3">Italia</option>
       </optgroup>
       <optgroup label="Asia">
            <option value="4">Japon</option>
            <option value="5">China</option>
       </optgroup>
</select>
```


Si nos fijamos en el ejemplo, vemos que la primera opción, la correspondiente a España, vemos que el elemento incluye el atributo selected, este atributo se utiliza para preseleccionar una determina opción.

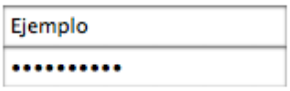
ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Otros atributos del elemento SELECT interesantes son size, que indica las filas que tendrá el atributo, en el ejemplo anterior tiene una fila y también la combinación de este elemento junto con el atributo multiple de SELECT, lo que nos permite seleccionar varios elementos del menú.

### 1.8.4 Entrada de texto

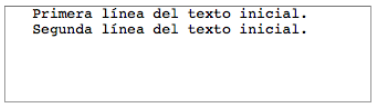
Este control se utiliza para permitir a los usuarios introducir texto. Existen dos elementos que permiten introducir texto, el elemento INPUT y el elemento TEXTAREA, la diferencia entre ambos es que el elemento TEXTAREA crea un control que permite introducir varias líneas.
Si utilizamos el elemento INPUT, tendremos que asignar al atributo type de este el valor text para una entrada de texto normal o el valor password, cuando queramos que los caracteres que se introduzcan se oculten a medida que se introducen.

```
<input type="text" value="Ejemplo" name="texto" />
<input type="password" value="Contraseña"
name="password" />
```




Como hemos comentado, también podíamos crear un control de texto mediante el elemento TEXTAREA, a continuación mostramos un ejemplo de uso.

```
<textarea name="eltexto" rows="20"
cols="80">
Primera línea del texto inicial.
Segunda línea del texto inicial.
</textarea>
```




### 1.8.5 Selección de ficheros

Este tipo de control, permite al usuario elegir ficheros de modo que sus contenidos pueden ser enviados con un formulario. Para crearlos, una vez más, se utiliza el elemento INPUT, el valor para type debe ser file. Veamos un ejemplo:

```
<input type="file" name="archivo"
/>
```


### 1.8.6 Controles ocultos

Los autores pueden crear controles que no se muestran pero cuyos valores se envían con el formulario. Este tipo de control se utiliza normalmente para almacenar información de intercambio entre el cliente/servidor. El elemento que se utiliza para crear este tipo de elementos es INPUT, fijando el valor para el atributo type a hidden.

```
<input type=”hidden” name=”oculto” value=”información” />
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 1.9 Marcos

Los marcos permiten a los autores presentar documentos con vistas múltiples, que pueden ser ventanas, o subventanas independientes. Las vistas múltiples ofrecen a los autores una manera de mantener cierta información visible mientras otras vistas se desplazan o se sustituyen.
En el ejemplo siguiente se muestra un ejemplo de uso de marcos, estos se crean mediante el elemento FRAMESET, concretamente veremos como se crean dos marcos (izquierda y derecha) donde a su vez el primer marco (izquierda) esta dividido en otros dos marcos (arriba y abajo).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
       "http://www.w3.org/TR/html4/frameset.dtd">
         <html>
                  <head>
                           <title>Un documento simple con marcos</title>
                  </head>
                  <frameset cols="20%, 80%">
                           <frameset rows="100, 200">
                                    <frame src="contenido_html_marco1.html">
                                    <frame src="contenido_html_marco2.html">
                           </frameset>
                           <frame src="contenido_html_marco3.html">
                           <noframes>
                                    <p>Su navegador no soporta frames</p>
                           </noframes>
                  </frameset>
         </html>
```

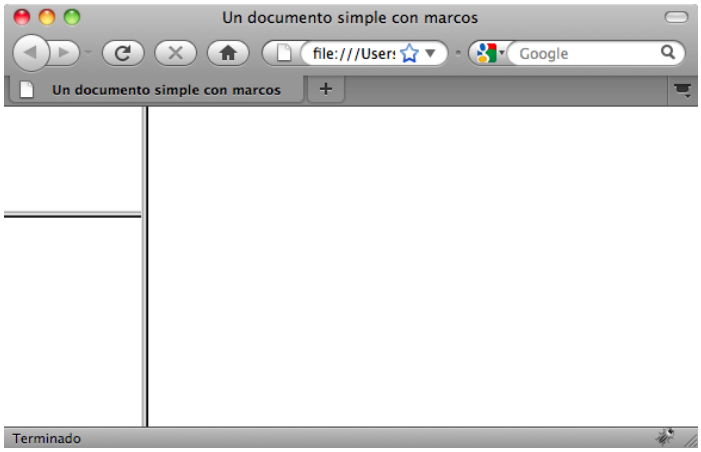



**Figura 15 - Código HTML: Frameset**

Como hemos visto en el ejemplo anterior, los documentos HTML con marcos tienen una estructura diferente a la de los documentos HTML sin marcos. Un documento normal tiene una sección HEAD y a continuación una sección BODY, en cambio, un documento con marcos tiene una sección HEAD y a continuación una sección FRAMESET. Esta sección FRAMESET es donde se especifica la disposición de las vistas en la ventana principal del agente de usuario. Además vemos en el ejemplo que los grupos de marcos también pueden anidarse [12-15].
Con el elemento FRAMESET especifica la organización de la ventana principal en un número determinado de subespacios rectangulares, teniendo en cuenta que los éstos se crean de izquierda a derecha y de arriba abajo, para ello se disponen de los atributos cols (para definir subespacios horizontales) y rows (para definir subespacios verticales). Hay que tener en cuenta que se pueden anidar tantos elementos FRAMESET como se desen. Veamos una serie de ejemplos:

| | |
|---|---|
| División en dos subespacios verticales | `<FRAMESET rows=”50%, 50%”>` |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| | […]<br></FRAMESET> |
| División en tres columnas | <FRAMESET cols=”1*, 250, 3*”><br>[…]<br></FRAMESET> |
| Una cuadrícula de 2x3 | <FRAMESET rows=”30%, 70%” cols=”33%, 34%, 33%”><br>[…]<br></FRAMESET> |

El elemento FRAME se utiliza para definir los contenidos y la apariencia de un marco dado, para ello este elemento dispone de los siguientes atributos:

- El atributo src define el contenido inicial del marco.
- noresize es utilizado para que el marco no se pueda redimensionar
- scrolling, define si el marco tendrá barras de desplazamiento utilizando para ello los valores auto, yes o no.
- frameborder define el border del marco, puede ser 1 (con bordes) y 0 (sin bordes).
- marginwidth y marginheight para especificar los márgenes del marco.

### 1.9.1 Especificación de información sobre el marco destino

Al asignar un nombre a un marco por medio del atributo name, los autores pueden referirse a él como el “destino” de los vínculos definidos por otros elementos. Se pueden establecer el atributo target para los elementos que creen vínculos (A, LINK). Veamos un ejemplo que lo ilustre:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
        <head>
                <title>Un documento con marcos</title>
        </head>
        <frameset rows="50%,50%">
                <frame name="fijo" src="inicial_fijo.html">
                <frame name="dinamico" src="inicial_dinamico.html">
        </frameset>
</html>
```

Después, en inicial_dinamico.html, hacemos un vínculo al marco llamado “dinámico”.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
        <head>
                <title>Un documento con vínculos con destinos específi-
                cos</title>
        </head>
        <body>
                ...comienzo del documento...

                        <p>Puede avanzar ahora a la <a href="diapo2.html"
                        target="dinamico">diapositiva 2.</a>

                ...más documento...
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
                                    <p>Lo está haciendo muy bien. Vaya ahora a la <a
                                    href="diapo3.html" target="dinamico">diapositiva
                                    3.</a>

                        </body>
                </html>
```

Al activar cualquiera de los vínculos se abre un nuevo documento en el marco llamado “dinámico”, mientras que el otro marco “fijo”, mantiene sus contenidos iniciales.
Los autores pueden compartir datos entre varios marcos, incluyendo estos datos a través de un elemento OBJECT dentro del elemento HEAD, que contenta el atributo id. Cualquier documento que sea el contenido de un marco del documento puede hacer referencia a este identificador.

### 1.9.2 Marcos en línea

El elemento IFRAME permite a los autores insertar un marco dentro de un bloque de texto, de forma similar a como se insertaba un objeto mediante el elemento OBJECT.

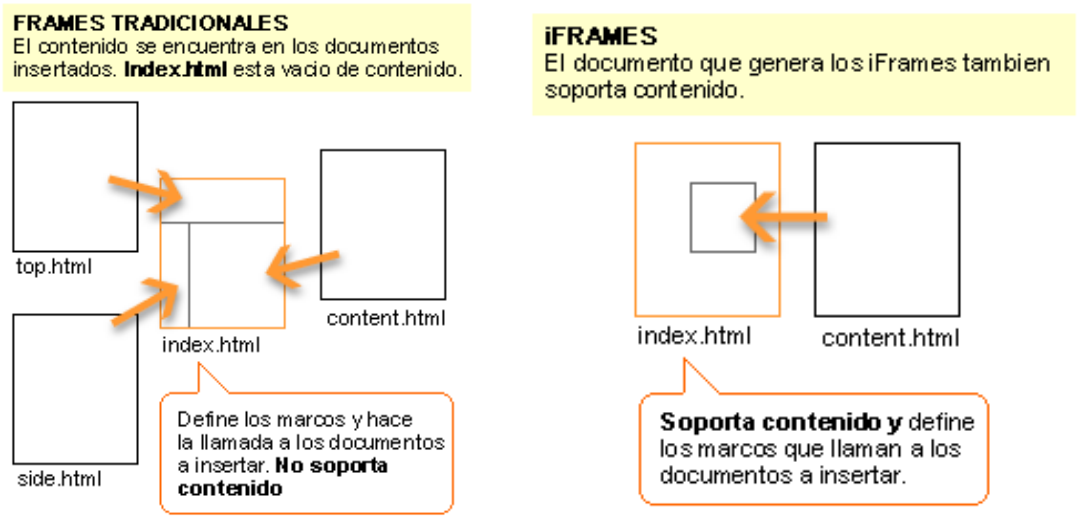



**Figura 16 - Frames tradicionales vs. iFrames**

El documento a insertar en línea se designa mediante el atributo src de este elemento IFRAME. Veamos el siguiente ejemplo.

```
<iframe id="miNuevoFrame" name="miNuevoFrame" src="incrustado.php" framebor-
der="0" framespacing="0" scrolling="auto" border="0” />
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 2 Hojas de estilo y CSS

## 2.1 Versión CSS

El lenguaje HTML está limitado a la hora de aplicar estilo a un documento puesto que fue diseñados para definir la estructura de un documentos Web, sin tener en cuenta el aspecto gráfico del mismo.
Inicialmente, los desarrolladores Web utilizaban tablas para la maquetación del contenido de la página, por ejemplo, o modificaban determinados atributos de las etiquetas HTML que definían color, colocación dentro de la página, etc. Esta técnica es considerada como una mala práctica de diseño puesto que la visualización se veía afectada por las plataformas o navegadores utilizados, además de que se mezclaba el contenido de la Web expresado con código HTML y el estilo del documento, con la consiguiente dificultad en el mantenimiento de las páginas Web.
**CSS** (*Cascading Style Sheets*) es un lenguaje que permite definir, mediante una sintaxis especial la presentación de un documento HTML o bien de un sitio Web manteniendo la consistencia de estilo en todas las páginas del sitio. Además es posible aplicar estilo a una etiqueta HTML en concreto, pudiendo definir varios estilos diferentes para una sola etiqueta.
CSS permite a los diseñadores:

- Podemos definir interlineado e identación de un párrafo
- Seleccionar fuente, tamaño o color del texto.
- Maquetación del contenido de la página con mayor precisión.
- Colocación de imágenes
- Se pueden utilizar distintas unidades como píxeles (px), pulgadas (in), puntos (pt), centímetros (cm) o porcentajes (%).

El objetivo básico de CSS es la separación del contenido de un documento Web (escrito en un lenguaje de marcado) de su presentación (expresado en CSS). De esta forma se logra una mejora en la accesibilidad del contenido, mayor flexibilidad y ofrece una mayor nivel de control sobre distintas características de presentación del documento.
Este lenguaje surge como una propuesta del W3C para mejorar la capacidades de la presentación Web. Desde el W3C se propusieron nueve lenguajes de estilo distintos de los cuales se seleccionaron dos que darían origen al CSS: *Cascading HTML Style Sheets (CHSS) y Stream-based Style Sheet Proposal (SSP).* El primero fue propuesto por *Hakon Wium Lie* en Octubre del 1994 y presenta ciertas similitudes con el CSS actual. Por su parte *Bert Bos* estaba trabajando en un navegador llamado *Argo* que utilizaba su propio lenguaje de estilo, el SSP. Ambos autores trabajaron junto para desarrollar el estándar CSS (la H se elimina porque estas hojas de estilo se podría aplicar a otros leguajes de marcado además de HTML). A pesar de que existan otro lenguajes de estilo como DSSSL y FOSI, CSS permitía que el estilo de un documentos estuviera incluido por múltiples hojas de estilo. Una hoja de estilo puede heredar de otro, permitiendo una mezcla de preferencias de estilo controlados igualmente por el diseñador del sitio y el usuario.
El desarrollo del estándar CSS fue impulsado desde el W3C a través de un grupo de trabajo al que pertenecían *Hakon y Bert*, junto con otros miembros como *Thomas Reardon* de Microsoft. A finales de 1996, CSS esta lista para hacerse oficial y la recomendación del CSS1 fue publicada en Diciembre.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

El grupo de trabajo de CSS intentaron resolver temas que no se trataban con CSS1, dando lugar a la creación del CSS2 en Noviembre de 1997 y publicada por el W3C en 1998. El desarrollo de CSS3 comenzó en 1998, y todavía se encuentra bajo desarrollo.
La versión de CSS manejada en el manual es la 2.1, que fue publicada como una recomendación del W3C en Septiembre del 2009. Los errores en las especificación del CSS2 han sido corregidos mediante la publicación de varias erratas. Muchos de estos temas serán tratadas en la especificación del CSS3. En la especificación de la CSS 2.1 se tratar de resolver dichas cuestiones:

- Manteniendo la compatibilidad con aquellas porciones de CSS2 que han sido aceptadas e implementadas ampliamente.
- Incorporando todas las erratas publicadas sobre CSS2.
- Cuando las implementaciones difieren en gran medida de la especificación CSS2, modificar la especificación de acuerdo con las prácticas aceptadas generalmente.
- Eliminar las características del CSS2 que, por no haber sido implementadas, han sido rechazadas por la comunidad CSS. CSS 2.1 tiene como objetivo reflejar que las características CSS son razonablemente implementadas para los lenguajes HTML y XML en general (en lugar de que sean solo para un lenguaje XML en particular o solo para HTML).
- Eliminar características de CSS2 que serán obsoletas para CSS3, impulsando la adopción de las características propuestas para CSS en su lugar.
- Añadiendo un pequeño número de nuevos valores de propiedad cuando las experiencia en implementación muestren que se necesitan para implementar CSS2.

A lo largo de este capitulo, tendremos como objetivo principal aprender a desarrollar hojas de estilo que se podrán aplicar a los documentos HTML que desean presentar en Internet. Así haremos una revisión de las principales etiquetas de CSS2 para la creación de documentos de estilo que definan colores, fuentes, localización de los elementos y sentar la bases para el desarrollo de hojas de estilo más complejas en las que se defina herencia entre elementos, etc.

## 2.2 Elementos básicos del lenguaje

Una hoja de estilo en CSS está formada por un conjunto de reglas que tienen dos clases principales, un selector (elemento HTML, nombre de una clase,…) y una declaración (propiedad:valor). La declaración tendrá a su ver dos partes, el nombre de la propiedad y un valor.

```
h1  { color: red }
```

La regla anterior definirá que el texto contenido dentro de las etiquetas <h1></h1> del documento HTML sobre el que se aplica el estilo, se representará en color rojo. Esta se puede considerar como una hoja de estilo en sí mismo.
La especificación de HTML 4 define que las hojas de estilo se pueden especificar bien dentro del documento HTML o a través de una hoja de estilo externo. Para incluirlo dentro del documento utilizamos el elemento STYLE:

```
<STYLE type="text/css">
h1 { color: red }
</STYLE>
```

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Para una mayor flexibilidad, se recomienda que se utilice la opción de hojas de estilo externas de modo que no sea necesario modificar el documento HTML directamente y pueda ser reutilizada para otros documentos. Para incluir el enlace a las hojas de estilo externas se utiliza el elemento LINK, en el que se especificar el tipo de enlace (*stylesheet*), la situación del fichero y el tipo de hoja de estilos (*text/css*):

```
<LINK rel="stylesheet" href="estilo.css" type="text/css">
```

### 2.2.1 Premisas

A la hora de definir un documento CSS debemos considerar las siguientes premisas:

- Las sintaxis CSS no se ve afectada por las mayúsculas en el rango ASCII ( es decir [a-z] es equivalente a [A-Z], excepto para apartados que no estén bajo control de CSS, como por ejemplo los valores de los atributos HTML "id" y "class" de los nombres de fuentes.
- En CSS los identificadores (incluyendo nombres de elementos, clases e IDs en los selectores solo puede contentes caracteres [a-zA-Z0-9] y caracteres ISO 10646, además del guión y guió bajo; no pueden empedar por un dígito o un guión seguido por un dígito. También pueden contener caracteres de escape y cualquier carácter ISO 10646 como un código numérico.
- En CSS 2.1, el carácter \ indica tres tipos de escape:
  - Dentro de una cadena la barra seguida de un carácter de nueva línea es ignorada
  - Cancela el significado de los caracteres especiales CSS.
  - La barra \ permite al autor hacer referencia a caracteres que no pueden incluir fácilmente en un documento. En este caso la barra está seguida por un número y un espacio, por ejemplo "\26 B" o por seis dígitos hexadecimales "\000026B".

## 2.3 Estructura básica de un documento CSS

En una hoja de estilo CSS se pueden distinguir dos tipos de elementos: las reglas que comienzan con '@' y los conjuntos de reglas.

Los primeras son reglas del tipo *@import* o *@page*, es decir que comienzan con el carácter '@' seguido de un identificador. Además finalizan con un ';'. Según la especificación de CSS 2.1 es ilegal incluir una reglas de este tipo dentro de un bloque o después de una línea que sea distinta a reglas del tipo *@charset* o *@import*. Por ejemplo los siguientes bloques de código CSS son ilegales:

```
@import "subs.css";
h1 { color: blue }
@import "list.css";
```

```
@import "subs.css";
@media print {
    @import "print-main.css";
     body { font-size: 10pt }
}
h1 {color: blue }
```

Los bloques se encuentran entre llaves '{}'. Dentro puede haber cualquier tipo de carácter excepto paréntesis, corchetes y llaves que deben aparecer en pares. La comilla simple y doble también aparecen en pares y los caracteres que aparecen entre ellas son considerados como una cadena.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Un conjunto de reglas es un selector seguido de un bloque de declaración que se encuentra limitado por dos llaves '{}' y está formado por 0 o más declaraciones separadas por ';'. Un mismo bloque de declaración puede hacer referencia a varios selectores distintos separados por ','.

```
h1, h2  { color: red }
```

Una declaración está compuesta por un nombre de propiedad, seguida de ':', un espacio y seguida de un valor. Las declaraciones que afectan a un mismo selector se pueden organización un mismo conjunto. El siguiente código es equivalente.

```
h1 { font-weight: bold }
h1 { font-size: 12px }
h1 { line-height: 14px }
h1 { font-family: Helvetica }
h1 { font-variant: normal }
h1 { font-style: normal }
```

```
h1 {
  font-weight: bold;
  font-size: 12px;
  line-height: 14px;
  font-family: Helvetica;
  font-variant: normal;
  font-style: normal
}
```

Es posible que toda las declaraciones aparezcan en la misma línea manteniendo la separación con ';'.

Finalmente los comentarios empiezan con los caracteres "/*" y finaliza con los caracteres "*\".

### 2.3.1 Valores

En CSS se utilizan números enteros o reales. Los números reales y enteros se especifican en notación decimal. Un entero consiste en uno o más dígitos del 0 al 9. Un real puede ser como un entero o bien 0 o más dígitos seguido de un punto y 0 o más dígitos. Tanto los enteros como reales pueden ir precedidos de `-´ o `x´ para indicar el signo.

**Longitudes**

Hacen referencia a medidas horizontales y verticales. Su formado es un número (entero o real) seguido de un identificador de unidad (px, em,…). Hay dos tipos de unidades de longitud las relativas y las absolutas. Las primeras especifican una longitud relativa a otra propiedad de longitud.

Las unidades relativas son:

- **em**: el 'font-size' de una fuente relevante
- **ex**: the 'x-height' de la fuente relevante
- **px**: pixels, relativo a la resolución del dispositivo de visualización.

```
h1 { margin: 0.5em }
h1 { margin: 1ex }
p { font-size: 12px }
```

Por su parte las unidades de longitud absolutas son solo útiles si las propiedades físicas del medio de salida son conocidas:

- **in:** pulgadas (inches) – 1 pulgada son 2.54 cm
- **cm:** centímetros
- **mm:** milímetros
- **pt:** puntos – los puntos de CSS 2.1 son igual a 1/72 de un pulgada

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **pc:** picas – 1 pica es igual a 12 puntos.

```
h1 { margin: 0.5in }
h2 { line-height: 3cm }
h3 { word-spacing: 4mm }
h4 { font-size: 12pt }
h4 { font-size: 1pc }
```

El formato de un valor de porcentaje es un número seguido inmediatamente por un '%'. Los valores de porcentaje son siempre relativos a otros valores, por ejemplo de una longitud.

```
p { font-size: 10px }
p { line-height: 120% } /* 120% of 'font-size' */
```

Los elementos hijos normalmente heredan los valores de propiedades de sus padres, pero en el ejemplo anterior los hijos del elemento heredarán un valor de 12px para line-height no el porcentaje 120%.

**URI/URL**

Para la especificación de valores de URI, se utiliza el valor de propiedad "url()". Dentro se especificará el valor de la URI, que opcionalmente podrá ir entre comillas simples o dobles. Las rutas podrán ser direcciones de Internet o bien rutas relativas o absolutas donde está localizado el recurso al que se hace referencia.

```
body { background: url("http://www.example.com/pinkish.png") }
```

**Colores**

En lo que respecta a los valores para expresar colores se utiliza una palabra clave o una especificación numérica RGB. La lista de palabras clave para colores con el correspondiente código hexadecimal son:

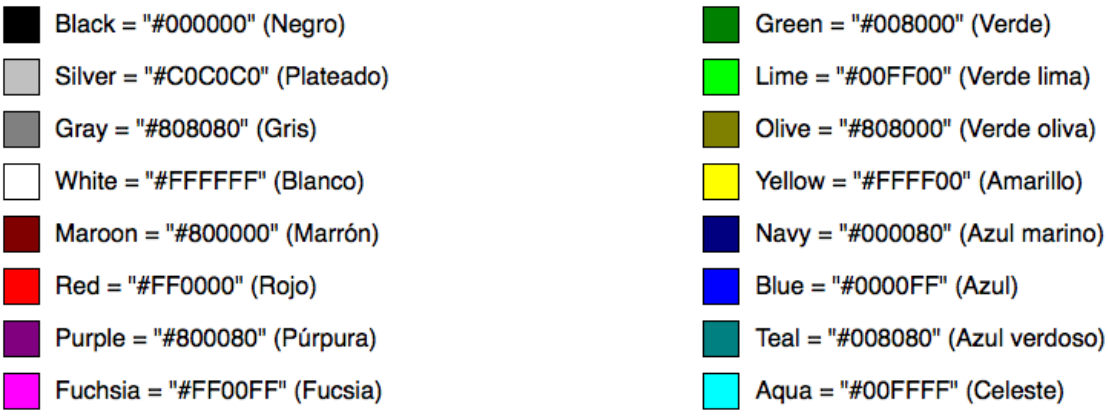



**Figura 17 - Nombre y códigos hexadecimal colores**

Además de estas palabras clave, los usuarios pueden especificar palabras clave que corresponden a los colores utilizados por ciertos objetos en el entorno de usuario.

```
body {color: black; background: white }
h1 { color: maroon }
h2 { color: olive }
```

El modelo de colores RFB se utiliza para las especificaciones numéricas de color.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
em { color: #f00 }
em { color: #ff0000 }
em { color: rgb(255,0,0) }
em { color: rgb(100%, 0%, 0%) }
```

Los ejemplos anteriores se corresponden con el mismo color.
El formato de un valor RGB en la notación funcional es “rgb()” y entre paréntesis una lista separada por comas de tres valores numéricos (tanto tres enteros como tres valores de porcentaje), de modo que 100% se corresponde con el valor 255.

**Cadenas de caracteres**

Se pueden escribir tanto con comillas dobles como simples. Para poder incluir una comilla doble entre dos comillas dobles es necesario utilizar el carácter de escape ‘\’. Los mismo ocurro para las comillas simples.

```
"this is a ’string’"
"this is a \"string\""
’this is a "string"’
’this is a \’string\’’
```

Para incluir una nueva línea en una cadena es necesario incluir un carácter de nueva línea tal que “\A” o “\00000a”. Se puede dividir una cadena entre varias líneas, pero es necesario incluir un carácter de escape ‘\’ al final de la línea.

### 2.3.2 Selectores

Los selectores identifica que reglas de estilos se aplican a los elementos en el árbol del documento. Estos selectores pueden ser tanto simples nombre de elementos como patrones contextuales.
La tabla siguiente contiene los selectores descritos en la especificación CSS 2.1:

| | |
|---|---|
| * | Aplica a cualquier elemento. |
| A | Aplica a cualquier elemento A. |
| A B | Aplica a cualquier elemento B que es un descendiente del elemento A. |
| A > B | Aplica a cualquier elemento B que sea hijo del elemento A. |
| A:first-child | Aplica a cualquier elemento A cuando A es el primer hijo de su padre |
| A:link<br>A:visited | Aplica al elemento A si A hace referencia a un hipervínculo que no ha sido visitado todavía o ya ha sido visitado |
| A:active<br>A:hover<br>A:focus | Aplica al elemento A durante ciertas acciones de usuario |
| A:lang(l) | Aplica a un elemento de tipo A si está expresado en el lenguaje l. |
| A+B | Aplica a cualquier elemento B inmediatamente precedido por un elemento A hermano. |
| A[atr] | Aplica a cualquier elemento A con el conjunto de atributos “att” |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| A[atr=”valor”] | Aplica a cualquier elemento A cuyo atributo “atr” tiene valor “valor” |
| A[atr~=”valor”] | Aplica a cualquier elemento A cuyo atributo “atr” tiene como valor una lista de valores separados por espacios, de los cuales uno es igual a “valor” |
| A[lang\|="en"] | Aplica a cualquier elemento A cuyo atributo “lang” tiene una lista de valores separados por guiones que comienza (por la izquierda) con “en” |
| DIV.clase | Específico para HTML, se aplicaría a un elemento DIV cuya clase sea “clase” |
| A#id | Aplica a cualquier elemento A con un ID igual a “id” |

**Selector de tipo**

Los *selectores de tipo* coinciden con el nombre de un elemento de tipo de un lenguaje de etiquetado como HTML. Se aplicará a cada instancia de este tipo de elemento en el árbol de documento.

```
h1 { font-family: sans-serif }
```

La regla anterior se aplicaría a todos los elementos H1 del documento HTML.

**Selector descendente**

Un *selector descendente* expresa una relación dentro de un patrón. Está formado por dos más selectores separados por un espacio en blanco de modo que “A B” se aplicará cuando un elemento B es descendiente, está contenido, en un elemento A.

```
h1 { color: red }
em { color: red }
h1 em { color: blue }
```

En este caso el texto contenido dentro de un tag EM que a su vez está contenido en un tag H1 tendrá color azul.

**Selectores hijo**

Por su parte, los selectores hijo hacen referencia a un elemento que es hijo de otro elemento, formado por dos o más selectores separados por “>”.

```
body > P { line-height: 1.3 }
```

Esta regla se aplica a todos los elementos <P> que son hijos de <BODY>.

**Selectores hermano**

Estos selectores tienen las sintaxis E1 + E2, cuando E1 y E2 comparten el mismo padre en el árbol de documento y E1 precede inmediatamente a E2.

```
h1 + h2 { margin-top: -5mm }
```

Con la regla anterior se reduce el espacio vertical de separación entre un elemento H1 y uno H2 que le sigue inmediatamente.

**Selectores atributo**

Se pueden considerar cuatro tipos de este selector:

| | |
|---|---|
| [atr] | Se aplica a todos los elementos con atributo “atr”, cualquiera que sea el valor de este atributo |
| [atr=val] | Se aplica en aquellos elementos cuyo atributo “atr” tiene valor “val” |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| [atr~=val] | Representa un elemento con atributo “att” cuyo valor es una lista de palabras separadas por espacios en blanco, de los que uno es exactamente “val” |
| [atr\|=val] | Representa un elemento con atributo “atr”, cuyo valor es exactamente “val” o comienza con “val” inmediatamente seguida por un guión. |

En el caso concreto del lenguaje HTML

```
h1[title] { color: blue; }
```

Hace referencia al atributo “title” de un elemento H1.

```
*[lang|="en"] { color : red }
```

La regla anterior hace referencia a valores del atributo “lang” que empieza con “en” incluyendo “en”, “en-US” y “en-cockney”.

**Selectores de clase**

En el lenguaje HTML, se utiliza ‘.’ para representar el atributo *class.*

```
*.todos { color: green }
.todos { color: green }
```

Ambas instrucciones se utiliza para asignar estilo a los elementos de *class = todos*.

**Selectores ID**

El atributo ID en un lenguaje como HTML permite a los autores asignar un identificador a una instancia de un elemento en el árbol del documento. Los selectores ID de CSS hacen referencia a una instancia de elemento basada en su identificador. Un selector ID de CSS contiene un “#” inmediatamente seguido del valor ID, que puede ser un identificador.

```
h1#titulo { text-align: center }
```

La regla anterior hace referencia a un elemento H1 cuyo atributo ID tiene el valor “titulo”.

## 2.4 Propiedades y herencia

La asignación de valores específicos a cada propiedad se puede basar en los siguientes mecanismos (en orden de precedencia):

- Cascada
- Valor por defecto de la propiedad
- Herencia de propiedades

En CSS existe la estructura “!import” que sigue a cualquier declaración, haciendo que esta preceda a cualquier declaración normal.

**Herencia de propiedades**

Algunos valores de propiedades son heredados por los hijos de un determinado elemento. Cada propiedad define si puede o no ser heredada.

```
<H1>The headline <EM>is</EM> important!</H1>
```

En el ejemplo anterior el elemento EM heredará el color del elemento padre que es H1. En el caso siguiente tenemos que la propiedad “font-size” del elemento H! Tomará valor de 13pt (130% de 10pt que es el valor del padre”.

```
body { font-size: 10pt }
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
h1 { font-size: 130% }
```

Cada propiedad puede tener un elemento específico de herencia, de modo que la propiedad toma el mismo valor que la propiedad del elemento padre.

```
body {
       color: black !important;
       background: white !important;
}
* {
       color: inherit !important;
       background: transparent !important;
}
```

Con el bloque anterior tenemos que todos los elementos contenidos en el BODY tendrán el mismo valor que el definido para este y el fondo será transparente.

## 2.5 Modelo de caja

Este modelo describe las cajas rectangulares generadas para los elementos en los árboles de documento y diseñados de acuerdo con el modelo de formateo visual.
Cada caja tiene un área de contenido (ej., texto, imagen,…) y áreas opcionales de bordes, márgenes y relleno, cuyo tamaño se definen con propiedades CSS. Un diagrama representativo puede ser el siguiente:

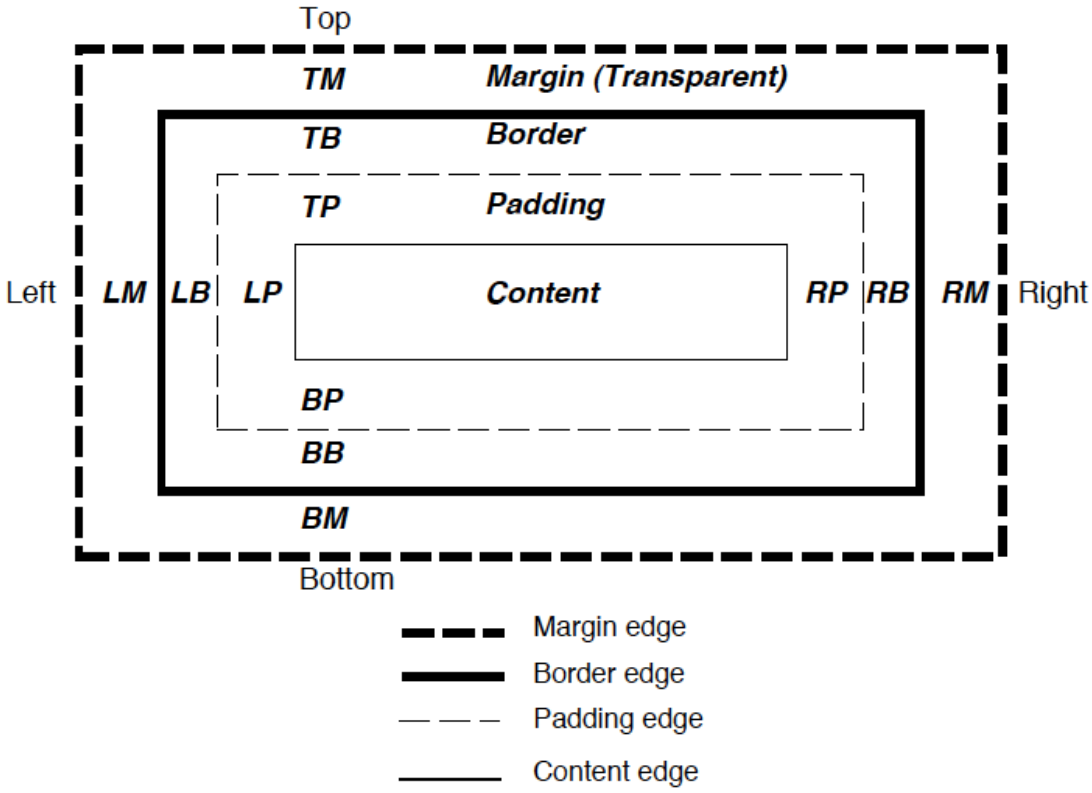



**Figura 18 - Modelo de caja**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

El margen, borde y relleno se puede dividir en segmentos “top”, “right”, “bottom” y “left” (superior, derecha, inferior e izquierda). El perímetro de cada una de las cuatro áreas (contenido, relleno, borde y margen) se denomina “edge” y cada una de las cajas tiene cuatro.

- **content edge:** rodea el rectángulo dado por “width” (anchura) y “height” (altura) de la caja.
- **padding edge:** el perímetro de relleno envuelve el relleno de la caja. Si el relleno tiene una anchura 0, the perímetro de relleno es el mismo que el perímetro del contenido.
- **border edge:** el perímetro del borde rodea el borde de la caja. Si el borde es de anchura 0, el perímetro del border es el mismo que el perímetro del relleno.
- **margin edge:** el perímetro del margen rodea el margen de la caja. Si el margen tiene una anchura 0, el perímetro del margen es el mismo que el perímetro del borde.

El estilo de fondo de las áreas de contenido, relleno y borde de una caja se especifica con las propiedad “background” del elemento generado. Los fondos de margen son siempre transparentes.

```
UL {
       background: yellow;
       margin: 12px 12px 12px 12px;
       padding: 3px 3px 3px 3px;
       /* No borders set */
}
LI {
       color: white; /* text color is white */
       background: blue; /* Content, padding will be blue */
       margin: 12px 12px 12px 12px;
       padding: 12px 0px 12px 12px; /* Note 0px padding right */
       list-style: none /* no glyphs before a list item */
       /* No borders set */
}
LI.withborder {
       border-style: dashed;
       border-width: medium; /* sets border width on all sides */
       border-color: lime;
}
```

Con las reglas del ejemplo anterior se definen las propiedades asociadas a las elementos HTML utilizadas para definir listas. El primer bloque se corresponden con propiedades generales de la lista definida con el elemento UL indicando que el fondo es de color amarillo, el margen total es de 12px (top right bottom left) y el relleno de 3px.

Para cada línea de la lista que estará definido por el tag LI, indica que el color del texto será blanco, el fondo del la “caja” que lo rodea será azul, el margen será de 12 px y el relleno de 12 px salvo en el extremo derecho que no define tamaño. La propiedad “list-style” indica que no se incluyan símbolos como guiones o puntos en los elementos de la lista.

El último bloque define propiedades para un elemento LI de clase “withborder”. Se indica que este elemento tendrá un borde con línea discontinua; anchura del borde, media y el color del borde, verde lima.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc


**Figura 19 - Resultado del estilo**

### 2.5.1 Propiedades de margen

El ancho del margen se expresa bien con una anchura fija o bien como porcentaje calculado respecto a la anchura de la caja generada.

| | |
|---|---|
| *margin-top*<br>*margin-bottom*<br>*margin-right*<br>*margin-left* | Valor: <anchura-margen> \| inherit<br>Valor inicial: 0<br>Aplica a: todos los elementos excepto aquellos con tipos de tabla distintos a *table-caption, table* e *inline-table.*<br>Heredado: no<br>Porcentajes: hace referencia a la anchura del bloque contenedor. |
| *margin* | Valor: <anchura>{1, 4} \| inherit<br>Valor inicial: 0<br>Aplica a: todos los elementos excepto aquellos con tipos de tabla distintos a *table-caption, table* e *inline-table.*<br>Heredado: no<br>Porcentajes: hace referencia a la anchura del bloque contenedor. |

Distintas formas de expresar el tamaño del margen

```
body { margin: 2em } /* todos los márgenes tiene una anchura de 2em */
body { margin: 1em 2em } /* margen superior e inferior (top y bottom) = 1em,
derecho  e izquierdo (right & left) = 2em */
body { margin: 1em 2em 3em } /* top=1em, right=2em, bottom=3em, left=2em */
```

### 2.5.2 Propiedades de relleno

El ancho de relleno se expresa bien con una anchura fija o bien como porcentaje calculado respecto a la anchura de caja considerada.

| | |
|---|---|
| *padding-top*<br>*padding-bottom*<br>*padding-right*<br>*padding-left* | Valor: <anchura-relleno> \| inherit<br>Valor inicial: 0<br>Aplica a: todos los elementos excepto *table-row-group, table-header-group, table-footer-group, table-row, table-column-group* y *table-column*<br>Heredado: no<br>Porcentajes: hace referencia a la anchura del bloque contenedor. |
| *padding* | Valor: <anchura>{1, 4} \| inherit<br>Valor inicial: 0 |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| | Aplica a: todos los elementos excepto *table-row-group, table-header-group, table-footer-group, table-row, table-column-group* y *table-column.*<br>Heredado: no<br>Porcentajes: hace referencia a la anchura del bloque contenedor. |

### 2.5.3 Propiedades de borde

El ancho del borde se expresa bien con un valor numérico o bien con:

- *thin:* un borde delgado.
- *medium:* un borde medio.
- *thick*: grueso

| | |
|---|---|
| *border-top-width*<br>*border-right-width*<br>*border-bottom-width*<br>*border-left-width* | Valor: <anchura-borde> \| inherit<br>Valor inicial: *medium*<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica<br>Si toma valor 0 significa que el estilo del borde es “none” o “hidden” |
| *border-width* | Valor: <anchura-borde> \| inherit<br>Valor inicial: *medium*<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica<br>Si toma valor 0 significa que el estilo del borde es “none” o “hidden” |

Un ejemplo para la especificación del tamaño del borde:

```
h1 { border-width: thin } /* thin thin thin thin */
h1 { border-width: thin thick } /* thin thick thin thick */
h1 { border-width: thin thick medium } /* thin thick medium thick */
```

El color del borde se expresa bien un valor de color (predefinido o con la expresión rgb()) o toma valor *transparent* para indicar que el borde es transparente

| | |
|---|---|
| *border-top-color*<br>*border-right-color*<br>*border-bottom-color*<br>*border-left-color* | Valor: <color>\| transparent \| inherit<br>Valor inicial: valor de la propiedad *color*<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica |
| *border-color* | Valor: [<color> \| transparent] {1,4} \| inherit<br>Valor inicial: valor de la propiedad *color*<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica<br>Si toma valor 0 significa que el estilo del borde es “none” o “hidden” |

La propiedades del estilo del borde especifican el estilo de la línea que dibuja el borde. Los valores posibles que pueden tomar estas propiedades son:

| | |
|---|---|
| *none* | Sin borde |
| *hidden* | Borde oculto |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| *dotted* | Línea punteada |
| *dashed* | Línea discontinua |
| *solid* | Línea continua |
| *double* | Dos líneas continuas |
| *groove* | Línea como un surco |
| *ridge* | Línea con relieve |
| *inset* | La caja parece que está en un surco |
| *outset* | La caja parece que está en relieve |

Las propiedades del estilo de borde son:

| | |
|---|---|
| *border-top-style*<br>*border-right-style*<br>*border-bottom-style*<br>*border-left-style* | Valor: <estilo-borde>\| inherit<br>Valor inicial: ninguno<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica |
| *border-style* | Valor: <estilo-borde>{1,4} \| inherit<br>Valor inicial: valor de la propiedad *color*<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica<br>Si toma valor 0 significa que el estilo del borde es “none” o “hidden” |

Es posible combinar las propiedades de anchura, estilo y color del borde en un solo tipo de propiedades:

| | |
|---|---|
| *border-top*<br>*border-right*<br>*border-bottom*<br>*border-left* | Valor: [<border-width> <border-style> <border-color>] \|inherit<br>Valor inicial: depende de las propiedades<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica |
| *border* | Valor: [<border-width> <border-style> <border-color>] \|inherit<br>Valor inicial: valor de las propiedades individuales<br>Aplica a: todos los elementos<br>Heredado: no<br>Porcentajes: No se aplica<br>Si toma valor 0 significa que el estilo del borde es “none” o “hidden” |

Por ejemplo para el elemento P:

```
p {
        border-top: solid red;
        border-right: solid red;
        border-bottom: solid red;
        border-left: solid red
}
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 2.6 Modelo de formateo visual

En CSS 2.1, muchas de las posiciones de caja y tamaños se calculan con respecto a los bordes de un caja rectangular llamado *bloque contenedor*. En general, las cajas actúan como bloques contenedores de las cajas descendentes. Para cada una de las cajas se da la posición con respecto a su bloque contenedor, aunque no tiene por qué estar limitada por su bloque contenedor; puede estar por encima.

### 2.6.1 Elementos de bloque

Los elementos de bloque son aquellos elementos del documento fuente que se formatean visualmente como bloques (ej. párrafos). Distintos valores de la propiedad *display* define un elemento de bloque: *block, list-item, run-in* y *table*.

#### 2.6.1.1 Bloques anónimos

Supongamos que tenemos la siguiente hoja de estilos

```
body { display: inline }
p { display: block }
```

utilizada en el siguiente documento HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HEAD>
<TITLE>Texto anónimo interrumpido por un bloque</TITLE>
</HEAD>
<BODY>
Este es un texto anónimo antes de P.
<P>Este es el contenido de P.</P>
Este es el texto anónimo después de P.
</BODY>
```

En este caso el elemento BODY contiene texto, seguido de un elemento bloque representado por el elemento P y seguido de otra porción de texto. Tendríamos entonces un bloque anónimo entorno a BODY, conteniendo una bloque anónimo alrededor del primer texto, el bloque P, y otro bloque anónimo alrededor del segundo texto.
Las propiedades de las cajas anónimas se heredan del bloque no anónimo que las contiene. Las propiedades no heredados tienen su valor inicial. Por ejemplo, si se asigna un borde al elemento BODY del ejemplo anterior el borde se dibujaría entrono a los dos texto contenidos en el cuerpo.

### 2.6.2 Elementos en línea

Los elementos en línea son aquellos elemento del documento que no constituyen nuevos bloques de contenido; el contenido se distribuye en líneas (ej.: piezas de texto destacadas en un párrafo, imágenes en línea, etc.). Distintos valores de la propiedad *display* definen un elemento en línea: *inline, inline-table, inline-block* y *run-in.*

#### 2.6.2.1 Bloques en línea

En un documento HTML que contiene la siguiente línea:

```
<p>Algún texto <em>destacado</em> impreso</p>
```

el elemento P genera una caja bloque, con varias cajas en línea dentro. La caja para “destacado” es una caja en línea generada por un elemento en línea EM, pero las otras cajas “Algún texto” e “impreso” son cajas en línea generadas por un elemento bloque P.
Estas cajas en línea anónimas heredan las propiedades heredables de su bloque padre. Las propiedades no heredadas tienen su valor inicial. En el ejemplo anterior el color de las cajas en línea anónimas son heredadas de P, pero el fondo es transparente.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 2.6.3 Cajas “Run-in”

Este tipo de cajas se comportan de la siguiente forma:

1. Si la caja *run-in* contiene una caja bloque, la caja *run-in* llega a ser una caja bloque.
2. Si una caja de bloque hermano (que no flota y no está colocada absolutamente) sigue a las caja *run-in,* ésta llega a ser la primera caja en línea de la caja bloque.
3. En los demás casos, las caja *run-in* se convierte en una caja bloque.

Consideremos el siguiente ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
	<HEAD>
		<TITLE>Un ejemplo de caja run-in</TITLE>
		<STYLE type="text/css">
		H3 { display: run-in }
		</STYLE>
	</HEAD>
	<BODY>
		<H3>Un encabezado run-in</H3>
		<P>Y un párrafo de texto que
		lo sigue.
	</BODY>
</HTML>
```

y el resultado formateado sería:

**A run-in heading.** And a paragraph of text that follows it.

### *2.6.4* La propiedad *display*

| | |
|---|---|
| *display* | Valores: *inline* \| *block* \| *list-item* \| *run-in* \| *inline-block* \| *table* \| *inline-table* \| *table-row-group* \| *table-header-group* \| *table-footer-group* \| *table-row* \| *table-column-group* \| *table-column* \| *table-cell* \| *table-caption* \| *none* \| *inherit*<br>Valor Inicial: *inline*<br>Se aplica a todos los elemento<br>No es heredada |

Los valores que puede esta propiedad tiene los siguientes significados

| | |
|---|---|
| *block* | El elemento actúa como una caja bloque |
| *inline-block* | El elemento con esta propiedad se representa como una caja bloque, que fluye como una caja en línea única, similar a un elemento reemplazado. El interior de un bloque en línea se formatea como una caja bloque, y el elemento en sí mismo es formateado como un elemento en lína reemplazado |
| *inline* | Este valor provoca que un elemento genere uno o más cajas en línea |
| *list-item* | Este valor hace que un elemento genere una caja bloque principal y una caja de elementos de lista en línea (ej. LI en HTML). |
| *none* | Este valor hace que un elemento no aparezca en la estructura de formateo. |
| *run-in* | Este valor crea cajas bloque o en línea, dependiendo del contexto. |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 2.6.5 Esquemas de posicionamiento

En CSS 2.1, un caja puede ser colocada de acuerdo a tres esquemas de posicionamiento:

1. *Flujo normal*: En CSS 2.1 incluye el formateo de bloque para cajas bloque, formateo en línea de cajas en línea, posicionamiento relativo de cajas bloque o en línea, y posicionamiento de cajas *run-in.*
2. *Flotante:* En el modelo flotante, una caja es colocado de acuerdo con el flujo normal, entonces es movida a la derecha o izquierda tanto como sea posible.
3. *Posicionamiento absoluto:* En el modelo de posicionamiento absoluto, una caja es eliminada por completo del flujo normal y asignada a una posición con respecto a un bloque contenedor.

### 2.6.6 La propiedad "posición"

| | |
|---|---|
| *position* | Valores: *static* \| *relative* \| *absolute* \| *fixed* \| *inherit*<br>Valor Inicial: *static*<br>Se aplica a todos los elemento<br>No es heredada |

Los valores que puede esta propiedad tiene los siguientes significados

| | |
|---|---|
| *static* | La caja es una caja normal colocada de acuerdo con el flujo normal. Las propiedades top, right, bottom y left no son aplicables. |
| *relative* | La posición de la caja se calcula de acuerdo al flujo normal. Entonces la caja es compensada de forma relativa a su posición normal. |
| *absolute* | La posición de la caja se especifica con las propiedades top, right, bottom y left. |
| *fixed* | La posición de la caja se coloca de acuerdo con el modelo absoluto, pero además, la caja se fija con respecto a alguna referencia. |

### *2.6.7* Propiedades de caja: *top, right, bottom, left*

Un elemento es posicionado si su propiedad *position* tiene un valor distinto de *static*. Los elementos posicionados generan cajas posiciones, colocadas de acuerdo a cuatro propiedades:

| | |
|---|---|
| *top* | Valores: <longitud> \| <porcentaje> \| *auto* \| *inherit*<br>Valor Inicial: *auto*<br>Se aplica a los elementos posicionados<br>No es heredada<br>Los porcentajes se refieren a la altura del bloque contenedor<br>Esta propiedad especifica cómo se coloca de modo absoluta el borde superior de la caja respecto del borde superior del bloque contenedor. Para cajas posicionadas relativamente, se realiza con especto a los bordes superiores de la caja en sí. |
| *right* | Valores: <longitud> \| <porcentaje> \| *auto* \| *inherit*<br>Valor Inicial: *auto*<br>Se aplica a los elementos posicionados<br>No es heredada<br>Los porcentajes se refieren a la anchura del bloque contenedor |

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| | Esta propiedad especifica cómo se coloca el borde derecho de la caja a la izquierda del borde derecho del bloque contenedor. Para cajas colocadas relativamente, el desplazamiento se realiza respecto del borde derecho de la caja en sí. |
| *bottom* | Valores: <longitud> \| <porcentaje> \| *auto* \| *inherit*<br>Valor Inicial: *auto*<br>Se aplica a los elementos posicionados<br>No es heredada<br>Los porcentajes se refieren a la altura del bloque contenedor<br>Esta propiedad especifica cómo se coloca el borde inferior de la caja por encima del borde inferior del bloque contenedor. Para cajas colocadas relativamente, el desplazamiento se realiza respecto del borde inferior de la caja en sí. |
| *left* | Valores: <longitud> \| <porcentaje> \| *auto* \| *inherit*<br>Valor Inicial: *auto*<br>Se aplica a los elementos posicionados<br>No es heredada<br>Los porcentajes se refieren a la anchura del bloque contenedor<br>Esta propiedad especifica cómo se coloca el borde izquierdo de la caja a la derecha del borde izquierdo del bloque contenedor. Para cajas colocadas relativamente, el desplazamiento se realiza respecto del borde izquierdo de la caja en sí. |

### 2.6.8 Flujo normal

#### 2.6.8.1 Contexto de formateo en línea

Un ejemplo de este tipo de formateo sería el siguiente bloque HTML:

```
<P>Varias <EM>palabras destacadas</EM> aparecen
<STRONG>en esta</STRONG> frase.</P>
```

En este caso el elemento P genera una caja bloque que contiene cinco cajas en línea, tres de las cuales son anónimas.

- Anónimo: “Varias”
- EM: “palabras destacadas”
- Anónimo: “aparecen”
- STRONG: “en estas”
- Anónimo: “frase”

En este ejemplo, la caja generada por el elemento P establece el bloque contenedor para las cajas de línea. Si el bloque contenedor es suficientemente ancho, todas las cajas en línea se podrán incluir en una sola línea.

```
Varias palabras destacadas aparecen en esta frase.
```

Si no, las cajas en línea se dividirán y serán distribuidas entre varias líneas.

```
Varias palabras destacadas aparecen
en esta frase.
```

Si consideramos la siguiente bloque HTML:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
<HEAD>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
<TITLE>Ejemplo de flujo en línea en varias líneas</TITLE>
<STYLE type="text/css">
       EM {
       padding: 2px;
       margin: 1em;
       border-width: medium;
       border-style: dashed;
       line-height: 2.4em;
       }
</STYLE>
</HEAD>
<BODY>
<P>Varias <EM>palabras destacadas</EM> aparecen aquí.</P>
</BODY>
</HTML>
```

Dependiendo de la anchura de P, la caja se distribuiría así:

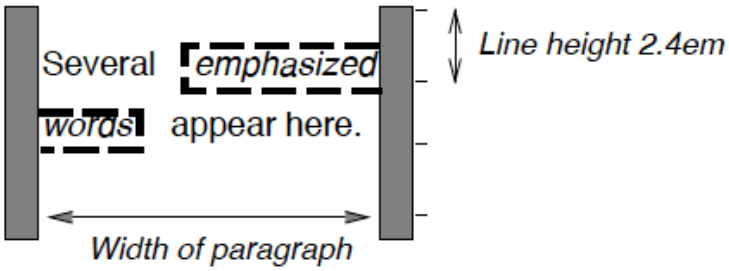



**Figura 20 - Texto formateado en flujo normal**

### 2.6.8.2 Posicionamiento relativo

Si consideramos el siguiente ejemplo:

```
div.a8 { position: relative; direction: ltr; left: -1em; right: auto }
div.a8 { position: relative; direction: ltr; left: auto; right: 1em }
div.a8 { position: relative; direction: ltr; left: -1em; right: 5em }
```

las propiedades *top* y *bottom* mueven de forma relativa los elemento arriba o abajo, respectivamente, sin cambiar su tamaño. Si ambas propiedades son *auto*, sus valores serán 0. Si uno de ellos es *auto*, su valor es el negativo del otro. Si ninguno de ellos es *auto* entonces la propiedad *bottom* es ignorada.

### 2.6.9 Flotante

Un flotante es una caja que es movida a la izquierda o derecha en su actual línea. Las característica más interesante de un flotante (o caja flotante) es que el contenido puedo flotar a lo largo de su lado.

Si consideremos el ejemplo inferior, el bloque contenedor es demasiado estrecho para contener el texto cercano al flotante, entonces el texto es movido por debajo de los flotantes de modo que es alineado en la caja de acuerdo a la propiedad *text-align*.

```
p { width: 10em; border: solid aqua; }
span { float: left; width: 5em; height: 5em; border: solid blue; }
...
<p>
<span> </span>
Supercalifragilisticexpialidocious
</p>
```

Se mostraría gráficamente como:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

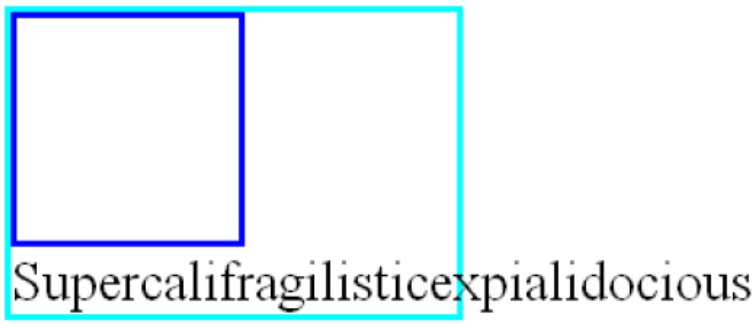



**Figura 21 - Ejemplo flotante**

### 2.6.10 Propiedad *float*

| | |
|---|---|
| *float* | Valores: *left* \| *right* \| *none* \| *inherit*<br>Valor Inicial: *none*<br>Se aplica a todos los elemento<br>No es heredada<br>Esta propiedad especifica si una caja debería flotar en la izquierda, derecho o en ninguna. Se aplica en aquellas cajas que no son colocadas de forma absoluta |

Los valores que puede esta propiedad tiene los siguientes significados:

| | |
|---|---|
| *left* | El elemento genera una caja bloque que flota en la izquierda. El contenido flota en el lado derecho de la caja, comenzando en la parte superior. |
| *right* | Parecido a *left* excepto que la caja flota en la derecha y el contenido flota en el lado izquierdo de la caja, comenzando en la parte superior. |
| *none* | La caja no flota. |

### 2.6.11 Posicionamiento absoluto

En el modelo de posicionamiento absoluto, la caja se coloca respecto a cu bloque contenedor.

#### 2.6.11.1 Posicionamiento fijo.

Es una subcategoría del posicionamiento absoluto. La única diferencia es que para un caja colocada de manera fija, el bloque contenedor está fijado por la Vista.
Por ejemplo si quisiéramos lograr una disposición de capas como la inferior:


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

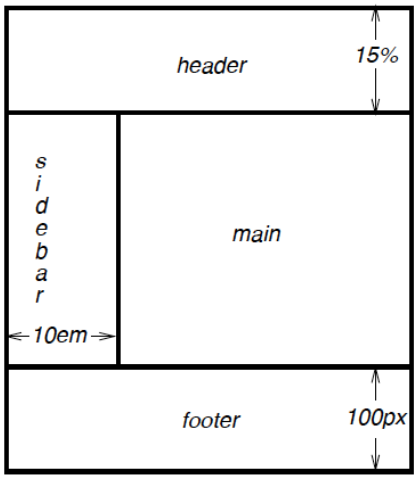



**Figura 22 - Colocación por posicionamiento absoluto**

lo lograríamos con el siguiente bloque HTML y reglas de estilo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
<HEAD>
<TITLE>A frame document with CSS 2.1</TITLE>
<STYLE type="text/css" media="screen">
       BODY { height: 8.5in } /* Required for percentage heights below */

       #header {
       position: fixed;
       width: 100%;
       height: 15%;
       top: 0;
       right: 0;
       bottom: auto;
       left: 0;
       }

       #sidebar {
       position: fixed;
       width: 10em;
       height: auto;
       top: 15%;
       right: auto;
       bottom: 100px;
       left: 0;
       }

       #main {
       position: fixed;
       width: auto;
       height: auto;
       top: 15%;
       right: 0;
       bottom: 100px;
       left: 10em;
       }

       #footer {
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
        position: fixed;
        width: 100%;
        height: 100px;
        top: auto;
        right: 0;
        bottom: 0;
        left: 0;
        }
</STYLE>
</HEAD>
<BODY>
<DIV id="header"> ... </DIV>
<DIV id="sidebar"> ... </DIV>
<DIV id="main"> ... </DIV>
<DIV id="footer"> ... </DIV>
</BODY>
</HTML>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 3 Lenguajes de scripts

## 3.1 Javascript

El lenguaje JavaScript es un lenguaje interpretado (el ordenador va leyendo cada instrucción del programa, la traduce y la ejecuta) y los programas escritos con estos lenguajes son conocidos como scripts o guiones.
Con JavaScript no pueden construirse programas independientes, sólo pueden escribirse scripts que funcionarán en el entorno de una página Web, interpretado por un explorador y hay que tener en cuenta que no todos los exploradores integran en la misma forma los guiones JavaScript.
La primera versión de JavaScript se debe a Netscape, que lo introdujo con la versión 2.0 de su explorador, posteriormente han ido surgiendo nuevas versiones habiendo sido estandarizado por la European Computer Manufacturers Asociation (**ECMA**).
El uso de JavaScript se justifica en que con un buen programa editor podemos obtener una página para publicar en la red, pero esa página Web consistiría en: texto, imágenes e hipervínculos, con los atributos como colores, tipos de letra y poco más sobre los que el autor puede actuar. Pero ¿y si quisiéramos poner un menú desplegable?, ¿y si queremos que el visitante pueda mover una imagen por la pantalla? .¿y si necesitamos validar el texto entrado por el usuario?.... En resumen si queremos ir mas allá de la simple presentación de un documento en pantalla y queremos controlar al explorador no hay mas remedio que utilizar un programa. ¿Por qué en JavaScript? muy simple: es soportado por todos los exploradores, es sencillo y es el que está siendo contemplado por los organismos de normalización [16-18].
En JavaScript, lo normal es que la ejecución se realice de forma automática cuando el navegador carga una página, o cuando el usuario pasa el ratón por una imagen, o cuando pulsa el botón de un formulario, etc. Estos cambios provocan los llamados **eventos** que son recibidos por el navegador que reaccionará en la forma adecuada: si haces click en un hiperenlace se genera un evento y el navegador abre una nueva página. Esos eventos son los que se aprovechan para que se ejecuten las instrucciones que nosotros escribimos en JavaScript. A cada evento se le puede asociar una función para que haga algo predeterminado por nosotros. Por ejemplo cuando el navegador carga una página se produce un evento que puede aprovecharse para hacer que se abra otra ventana (las conocidas como ventanas popup tan usadas para mensajes publicitarios), o cuando pasamos el ratón por una enlace se produce otro evento que puede aprovecharse para llamar a una función que modifique el color en que se muestra el enlace, o cuando el usuario pulsa una tecla. Los eventos tienen la naturaleza de objetos, o sea, poseen métodos y propiedades. Así cuando se produce un evento podemos saber quien lo dispara, en que posición de la pantalla se ha disparado y otras propiedades dependientes de cada evento en concreto.
Cuando el navegador empieza a leer el script para ejecutarlo lo hace en orden secuencial, o sea, empieza con la primera instrucción, sigue por la segunda y así hasta llegar al final. Esto es lo que se conoce como ejecución secuencial o lineal. Pero a veces es necesario saltarse instrucciones, por ejemplo, cuando se construye una página a la que sólo pueden entrar determinadas personas, será necesaria una función que pida el nombre de quien desee ver la página, si es una persona autorizada muestra la página y si no lo es no la muestra. El programa no ha seguido un flujo lineal, unas veces ejecutará la parte de mostrar la página y otras no. Otra situación bastante común: se busca que un programa recorra todas las imágenes de tu página y vaya cambiando su contenido, no se escribirá el mismo código una y otra vez, lo ideal sería escribir las instrucciones y poderlas repetir.. Cualquier lenguaje de programación tiene solucionado este asunto mediante las llamadas sentencias de control del flujo de programa. Son sentencias que permiten que se ejecuten condicionalmente algunos pasos (condicionales) o repetir una serie de instrucciones una y otra vez (bucles).

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

### 3.1.1 Elementos del lenguaje

#### 3.1.1.1 Tipos de datos

JavaScript maneja cuatro tipos de datos:

- **Numéricos**: Los datos numéricos sirven para manejar cualquier número real. También son datos numéricos aquellos con formato: 2.0E2, 0xF0, 012. El primero corresponde a la llamada notación científica: $2.0*10^2$, mientras que el segundo ejemplo corresponde a la notación hexadecimal y el tercero a la octal.
- **Cadena**: Los datos de cadena son los usados para cadenas alfanuméricas como por ejemplo “DSCE", “Comercio y negocio electrónico”. Estos datos están encerrados entre comillas, que es la forma de indicar es una cadena de caracteres y no un nombre o descriptor de algún elemento del programa.

  Si tenemos la siguiente cadena “JavaScript es un lenguaje de “script” o lenguaje interpretado”, la interpretación de esta cadena daría un error. Esto es debido a que hay dos cadenas y un nombre en medio: "JavaScript es un lenguaje de", script y "o lenguaje interpretado". JavaScript intentará encontrar una variable con el nombre *script*. Para poder hacer esto correctamente se usa el carácter de escape: \, lo que sigue a esta barra invertida es un carácter especial que debe interpretarse como un carácter normal, prueba ahora esto:

```
"JavaScript es un lenguaje de \"script\" o lenguaje interpretado"
```

  Existen mas caracteres especiales como tabulaciones, cambios de línea de escritura, borrado izquierda, la propia barra invertida.

```
"Esto usa una \t tabulación"
"Esto usa un \n cambio de línea"
"Esto usa un \r retorno de carro
"Esto es la barra invertida \\ "
```

- **Booleano:** es un tipo de dato ni numérico ni cadena, que puede tomar dos valores posibles *true* o *false*. Estos datos son los utilizados cuando en los programas se tengan que tomar decisiones, o sea, comprobar algo y actuar en consecuencia.
- **Punteros:** se emplea a menudo en los scripts para la captura de eventos, son direcciones de memoria, usadas para asignar funciones. Si en una variable se guarda el nombre de una función esa variable se convierte en otro nombre para esa función.

#### 3.1.1.2 Variables

Los valores numéricos no siempre pueden utilizarse de manera directa, sino que habitualmente se almacenan en una posición de memoria con nombre, esto es lo que se llama una **variable**. En otros lenguajes de programación existen diferentes tipos de variables en función del tipo de datos que pueden guardar: variables para cadenas, para números enteros, para números reales, etc. JavaScript es muy permisivo en este aspecto de manera que una variable puede guardar cualquier tipo de dato y además pueden crearse en cualquier parte del programa. Esto último es cómodo pero peligroso pues puede llevar a programas difíciles de depurar y de modificar, es conveniente llevar un cierto control sobre las variables que vas usando en cada función y declararlas al principio de la misma.

Los nombres de las variables pueden ser cualquier combinación de letras y dígitos, mas el guión bajo, siempre que el primer carácter no sea un dígito y que no coincida con una palabra reservada


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

del lenguaje, es decir, palabras con un significado especial para el intérprete como *close, open, write*... Es aconsejable usar nombres autoexplicativos, es una forma de documentar los programas. Por ejemplo una variable para guardar una dirección de un icono puede llamarse **direc_icono**. Por último, JavaScript diferencia entre mayúsculas y minúsculas, así *Edad* y *edad* serían dos variables distintas.
Otro aspecto a tener en cuenta a la hora de usar las variables es su ámbito, es decir, qué funciones tienen acceso a ellas. Si se crea una variable dentro de una función sólo será conocida dentro de esa función, se trata de *variables locales*. Si se necesita que varias funciones tengan acceso a una determinada variable ésta debe crearse como **variable global**, esto se hace creándola fuera de todas las funciones. Por ejemplo en el siguiente script tenemos variables globales y locales:

```
<script language="Javascript">
var navegador_version = 0;
function verNavegador()
{
var version;
version = document.appVersion;
return version; }
</script>
```

En este ejemplo *navegador_version* es una variable global mientras que *version* es local a la función *verNavegador()*. Las variables están creadas con la palabra clave **var**, el uso de esta palabra es opcional, sólo es obligatorio si una variable local tienen el mismo nombre que una global. Otro detalle a tener en cuenta es que al mismo tiempo que creamos la variable podemos darle un valor, si no lo hacemos la variable contendrá el valor **null**.

#### 3.1.1.3 Objetos

Javascript no posee todas las características de los lenguajes orientados a objetos como Java o C++, pero si es capaz de manejar objetos e incluso crearlos. De hecho si un programa en este lenguaje es capaz de interactuar con el explorador es gracias a esta capacidad. Javascript posee algunos objetos predefinidos u objetos intrínsecos como son: **Array, Boolean, Date, Function, Global, Math, Number, Object, RegExp** y **String**. Además el programador puede crear objetos nuevos, con sus propios métodos y propiedades, adaptados a las necesidades concretas de su aplicación.
Crear un objeto nuevo es tan simple como definir cuales serán sus propiedades y sus métodos, teniendo en cuenta que cualquier objeto que definamos ya posee heredados los métodos y propiedades del objeto predefinido **object**. En el ejemplo siguiente creamos un objeto pagina que aclarará la forma de crear objetos propios:

```
function pagina (titulo, color, fondo)
{
     this.titulo = titulo;
     this.color = color;
     this.imgfondo = fondo;
     this.length = 3;
}
var miPagina = new pagina("Mi página", "Blue", "cruces.gif");
var nuevapag = new pagina("2a Página", "White", "");
```

Este objeto se crea con el operador **new** pasándole como argumentos las propiedades declaradas para este objeto: *titulo, color, imgfondo*. La palabra clave **this** se usa para referirnos al propio objeto que estamos definiendo. Aún mas podemos crear propiedades nuevas sólo para la variable **miPagina**, pero estas propiedades no afectarán al objeto pagina en sí. Por ejemplo:

```
miPagina.descripcion = "Este es un ejemplo"; alert (miPagina.descripcion);
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

da como resultado la frase Este es un ejemplo, pero si ahora escribiéramos:

```
alert(nuevaPag.descripcion);
```

obtendríamos **undefined** pues *descripcion* solo es una propiedad de la variable ***miPagina*** no del objeto **pagina**. Para ampliar un objeto usamos la propiedad **prototype**:

```
pagina.prototype.descripcion = "Objeto definido por mi"; alert(nueva-
Pag.descripcion);
alert(miPagina.descripcion);
```

Ahora hemos añadido una propiedad al objeto **pagina** y esta propiedad se transmite a las variables de tipo **pagina** (mas correctamente instancias de pagina), por tanto en ambos casos obtendríamos la frase *Objeto definido por mí*, que es el valor dado a la nueva propiedad.

#### 3.1.1.4 Arrays

Los arrays no son más que estructuras para almacenar listas de valores. A diferencia de otros lenguajes, en JavaScript los arrays no son un tipo de variable sino que fueron implementados por Netscape como un objeto, lo cual les da una potencia enorme.
Un array puede verse como una lista con nombre donde cada valor se identifica por su número de orden en el array (la lista), su índice, y el número total de espacios para asignar valores en el array es su longitud. Si en un array tenemos anotados 5 números, el primero de todos será el de índice 0 y el último tendrá como índice el 4, siendo 5 la longitud del array.

```
....
semana = new Array(7);
miLista = new Array(1,5,9);
nombres= new Array('Juan', 'Luis', 'María');
vacio = new Array(); interesante = new Array(4);
...
```

El array semana se ha creado con longitud 7, o sea, permite 7 elementos. El array **miLista** se ha creado con longitud 3 y se ha dado valor a cada elemento al igual que el array **nombres**. Por último el array vacio se ha creado sin longitud.

```
...
semana[0] = 'Lunes';
semana[1] = 'Martes';
semana[2] = 'Miércoles';
semana[3] = 'Jueves' ;
semana[4] = 'Viernes';
semana[5] = 'Sábado';
semana[6] = 'Domingo;

vacio[5] = 'ultimo';

interesante['Jose'] = 10;
interesante['Pilar'] = 5;
interesante['Antonio'] = 8;
....
```

En este último segmento de código se han rellenado los arrays: **semana** se ha rellenado con los nombres de los días de la semana. En el array **vacio**, se ha dado valor al elemento 5 de modo este array tendra longitud 6 con los cinco primeros elementos con valor null y el sexto (vacio[5]) con


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

valor 'último'. Por último al asignar valores en el array i**nteresante** utilizamos como índices cadenas de caracteres puesto que no es necesario que sean numéricos.
Cada elemento de un array puede ser un valor de cualquier tipo: números, cadenas ... u otro array, en este último caso tendremos arrays multidimensionales, o sea, donde cada elemento tiene varios índices o coordenadas para localizarlos. Por ejemplo un array bidimensional sería un array donde cada elemento es a su vez otro array con lo que cada elemento tendrá dos índices uno para el primer array y otro para el segundo. El array bidimensional se puede considerar como una tabla ordenada en filas y columnas, cada elemento del array es una celda de esa tabla.

#### 3.1.1.5 Funciones

Las funciones son bloques de instrucciones de programa con nombre y que pueden ejecutarse sin mas que llamarlas desde alguna parte de otra función o desde la página HTML, bien sea directamente o mediante eventos.
Habitualmente una función se crea para ejecutar una acción muy concreta. JavaScript posee una serie de funciones predefinidas o **funciones globales** pero el programador puede crear las suyas propias. Para crear una función, tan sólo es necesario indicárselo al intérprete mediante la plabra clave **function** seguida del nombre de la función y, encerrados entre paréntesis, las variables que simbolizan los valores con los que deba trabajar la función, los *argumentos*. Los paréntesis deben escribirse aunque no haya argumentos. Para los nombres de funciones seguimos las mismas reglas que para las variables: carateres, dígitos y guión bajo, debiendo comenzar por un carácter o el guión bajo.
Es recomendable dar a las funciones un nombre representativo de la operación que realice. Opcionalmente la función puede finalizar con la palabra clave **return** seguida de un valor, este valor será el que devuelva la función al programa que la llame.
Por ejemplo:

```
function sumar(a,b)
{
var suma; suma = a + b;
return suma;
}
```

Creamos así la función llamada **sumar**, que utiliza dos argumentos y lo que hace es sumar estos argumentos. Por último se devuelve el resultado de la operación, mediante la palabra clave **return** seguida del valor que debe devolver. Ahora en el siguiente código de programa usamos la función recién definida:

```
var operacion;
operacion = sumar(4,5);
```

En este código llamamos a la función con los argumentos 4 y 5 y almacenamos el resultado en la variable **operacion**. Hasta aquí el comportamiento de las funciones JavaScript es similar a cualquier otro lenguaje, pero en JavaScript las funciones también son objetos.

```
var multiplicar = new Function("x", "y", "return x * y")
```

En el ejemplo anterior **multiplicar** no es una variable cualquiera sino una instancia del objeto **Function** y puede usarse como la propia función. Esta característica permite asignar directamente funciones a los eventos de las páginas web y así simplificar su programación.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

### 3.1.2 Operadores

#### 3.1.2.1 Operadores Aritméticos

| | |
|---|---|
| + (Suma) | Se trata de un operador usado para sumar dos valores numéricos o para concatenar cadenas entre sí o números y cadenas.<br>`var var1 = 10, var2= "Buenos", var3 = " días", var4 = 31;`<br>`document.write(var1+var4) /* resultado 41 */`<br>`document.write(var2+var3) /* resultado: Buenos días */`<br>`document.write(var1+var3) /* resultando: 10 días */` |
| - (Resta) | Operador usado para restar valores numéricos. Puede actuar sobre un único operando numérico cambiándole de signo.<br>`var num1 = 10, num2 = 8, res = 0;`<br>`res = num1 - num2;  /*res contiene 2 */`<br>`res = -res /* ahora res contiene -2*/` |
| * (Producto) | Realiza las operación aritmética de multiplicar dos valores<br>`var op1 = 50, op2= 4, mul;`<br>`mul = op1 * op2     /*mul contendrá 200 */` |
| / (Cociente) | Realiza las operación aritmética de dividir dos valores<br>`var op1 = 50, op2= 4, div;`<br>`div = op1/op2  /*div contendrá 12.5 */` |
| % (Resto) | También llamado operador módulo calcula el resto de una división.<br>`var op1 = 50, op2= 4, resto;`<br>`resto = op1 % op2;  /*resto contiene 2 */` |
| ++ (Incremento)<br>-- (Decremento) | Estos operadores se usan para incrementar o decrementar en 1 el valor de una variable. Si el operador se antepone a la variable la operación de incremento o decremento es prioritaria sobre cualquier otra.<br>`var op1=5, op2 = 5, res;`<br>`res = ++op1; /*res adquiere el valor 6 y luego op1 el 6*/`<br>`res = op1++; /*res adquiere el valor 5 y luego op2 el 6*/` |
| Operadores Compuestos | Los operadores +, -, *, / pueden asociarse con el operador de asignación (=) para cambiar el valor de una variable numérica por incrementándolo, decrementándolo, multiplicándolo o dividiéndolo por un valor. El operador += puede usarse igualmente con variables de cadena.<br>`var num = 20, cad = "buena";`<br>`num += 5; /*num adquiere el valor 25 (20 + 5) */`<br>`cad += 's'; /*cad adquiere el valor 'buenas' */`<br>`num *= 10; /*num adquiere el valor 250 (25*10) */` |

#### 3.1.2.2 Operadores Binarios

El ordenador, internamente, trata cualquier tipo de datos como una cadena binaria ( ceros y unos). Así los números se representan en sistema binario de numeración mientras que los caracteres se convierten a código ASCII, que son números que se almacenan codificados en binario. JavaScript ofrece los operadores típicos para trabajar con estas cadenas a nivel de bit (cada uno de los ceros o unos de las cadenas binarias.

| | |
|---|---|
| ~ (Complementación) | Complementa una cadena binaria convirtiendo los 1 en 0 y los 0 en 1. Por ejemplo el número 38 escrito en sistema binario es 00100110 si le aplicamos este operador se convierte en 11011001, o sea el -39 |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| | (JavaScript usa codificación en complemento a 2 para los números negativos). |
| << (Desplazamiento izquierda) | Desplaza los bits a la izquierda los lugares que se le indique rellenando con ceros por la derecha y desechando los bits de mayor peso, esto equivale a multiplicar por potencias de 2. Por ejemplo si al 00011010 (26) lo desplazamos 2 a la izquierda tendremos el 01101000 (104).<br>`var num = 26, res;`<br>`res = num << 2;  /* num contendrá 104 */` |
| >> (Desplazamiento derecha) | Desplaza los bits a la derecha los lugares que se le indique rellenando con ceros por la izquierda y desechando los bits de menor peso, esto equivale a una división entera por potencia de 2. Por ejemplo si al 00011010 (26) lo desplazamos 2 a la derecha tendremos el 00000110 (6).<br>`var num = 26, res;`<br>`res = num << 2;  /* num contendrá 104 */` |
| & (AND lógico binario) | Realiza un AND lógico bit a bit entre dos valores. El AND lógico da como resultado 1 sólo si ambos bits son 1. Por ejemplo<br>01101101 (109)<br>AND 00100110 (38)<br>resultado: 00100100 (36)<br><br>`var op1 = 109, op2 = 38, res;`<br>`res = op1 & op2;` |
| \| (OR lógico binario) | Realiza un OR lógico bit a bit entre dos valores. El OR lógico da como resultado 0 sólo si ambos bits son 0. Por ejemplo<br>00111010 (58)<br>OR 01010010 (82)<br>resultado: 01111010 (122)<br>En el ejemplo podemos ver la sintaxis del operador<br>`var op1 = 58, op2 = 82, res;`<br>`res = op1 \| op2;  /*res contiene 122 */` |
| ^ (XOR lógico binario) | Realiza un XOR lógico bit a bit entre dos valores. El XOR lógico da como resultado 1 si uno sólo de los bits es 1. Por ejemplo<br>00111010 (58)<br>XOR 01010010 (82)<br>resultado: 00101000 (40)<br>En el ejemplo podemos ver la sintaxis del operador<br>`var op1 = 109, op2 = 38, res;`<br>`res = op1 ^ op2;  /*res contiene 40*/` |

#### 3.1.2.3 Operadores Lógicos

Los operadores lógicos se utilizan para realizar comparaciones entre valores, numéricos o no, dando como resultado un valor **booleanos** (**true, false**). La operación lógica negación invierte el operando, si es **true** lo hace **false** y viceversa. Si se comparan números con cadenas, JavaScript intenta convertir internamente los datos. En los operadores relacionales (>, <, >=, <=) intenta convertir los datos en tipo número. Para los operadores de igualdad (== !=) intenta convertir los tipos de datos a cadena, número y booleano. Los operadores de identidad (===, !==) no realizan conversión de tipo.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| > (Mayor que) | Compara dos valores y devuelve true si el primero es mayor que el segundo. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = 10, comp;`<br>`comp = hoy > ayer  /* comp adquiere el valor false*/` |
| < (Menor que) | Compara dos valores y devuelve true si el primero es mayor que el segundo. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = 10, comp;`<br>`comp = hoy < ayer  /* comp adquiere el valor false*/` |
| >= (Mayor o igual) | Compara dos valores y devuelve true si el primero es mayor o es igual que el segundo. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = 4, comp;`<br>`comp = hoy >= ayer  /* comp adquiere el valor true*/` |
| <= (Menor o igual) | Compara dos valores y devuelve true si el primero es menor o es igual que el segundo. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = 4, comp;`<br>`comp = hoy <= ayer  /* comp adquiere el valor true*/` |
| == (Iguales) | Compara dos valores y devuelve *true* si ambos son iguales. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = 4, comp;`<br>`comp = hoy == ayer  /* comp adquiere el valor true*/` |
| === (Idénticos) | Similar a == pero también compara el tipo de datos de los operandos. Compara dos valores y devuelve *true* si el primero es mayor o es igual que el segundo. Compara tanto números como cadenas.<br>`var hoy = 4; ayer = '4', comp;`<br>`comp = hoy == ayer; /* comp adquiere el valor true*/`<br>`comp = hoy === ayer /* comp adquiere el valor false*/` |
| != (No iguales)<br>!== (No idénticos) | Invierten el sentido de las comparaciones iguales == e idénticos === respectivamente. |
| && (AND lógico) | Este operador se utiliza para concatenar comparaciones, es decir, para comprobar varias condiciones. El resultado sólo será true si todas las comparaciones lo son.<br>`var op1 = 2, op2 = 50, op3 = 25, comp;`<br>`comp = (op1 > op2) && (op1 < op3); /*comp adquiere el valor false */`<br><br>comp es *false* por que op1 no es mayor que op2 aunque sea mayor que op3 |
| \|\| (OR lógico) | Como el anterior, sirve para realizar comparaciones compuestas y sólo devolverá *false* cuando todas las comparaciones los sean. Es decir hasta que una comparación sea *true* para que devuelva el valor *true*.<br>`var op1 = 2, op2 = 50, op3 = 25, comp;`<br>`comp = (op1 > op2) && (op1 < op3); /*comp adquiere el valor true */`<br><br>comp es true por que op1 es menor que op3, (op1 < op3 es por tanto *true*) |

#### 3.1.2.4 Operadores Varios

| | |
|---|---|
| **delete** | Se usa para borrar propiedades de un objeto o elementos de un array. Devuelve *true* si la operación se realizó con éxito.<br>`var lista = new Array(1,4,7,9,10);`<br>`delete(lista,0);` |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| | El elemento lista[1] contiene ahora *undefined.* |
| **new** | Se utiliza para crear instancias de un objeto<br>`var hoy = new Date("10 /30/2000")` |
| **typeof** | Devuelve el tipo de dato al que pertenece una variable o expresión. Los tipos devueltos son *number*, *string*, *boolean*, *object*, *function* y *undefined.*<br>`hoy = 1.2345; tipo = typeof(hoy);`<br>La variable tipo contendrá **number.** |

### 3.1.2.5 Funciones Globales

A sí como JavaScript proporciona objetos predefinidos, también posee una serie de funciones predefinidas. Se trata de las funciones: **eval, isNan, Number, String, parseInt, parseFloat, escape, unescape.**

| | |
|---|---|
| **eval** | Se usa para evaluar una cadena con código JavaScript sin referirse a un objeto concreto. La sintaxis de *eval* es: *eval(expr)* donde *expr* es la cadena a evaluar. |
| **isNaN(arg)** | Determina si el argumento es un valor NaN (not a number) |
| **parseint(str, base])** | Convierte una cadena de caracteres en un valor numérico. La función lleva como argumento la cadena a convertir y opcionalmente puede llevar un segundo argumento para indicar la base de numeración en que está escrita la cadena. Si se omite se supone que la cadena representa un número en base 10. La cadena sólo podrá contener caracteres válidos para el sistema de numeración indicado: dígitos (0..9 para la base 10, 0 1 para números binarios, 0..7 para sistema octal, 0..9, A..F para sistema hexadecimal) y signo (+, -). Si encuentra algún carácter no válido sólo interpreta desde el principio de la cadena hasta el carácter no válido. Si comienza por un carácter ilegal devuelve **NaN**.<br>Ejemplo:<br>`...`<br>`var minum1 = "14";`<br>`document.write(parseInt(minum1));`<br>`....`<br><br>Escribirá 14. En el siguiente ejemplo transforma un número binario a decimal:<br>`...`<br>`var minum1 = "11001"; document.write(parseInt(minum1,2));`<br>`....`<br>Ahora escribirá 25, el equivalente decimal al binario 11001. |
| **parseFloat(str)** | Convierten la cadena que se le pasa como argumento a un valor numérico de tipo flotante. Los caracteres válidos de la cadena son los mismos que en parseInt mas el punto decimal y el exponente (E). No admite un segundo argumento. Por lo demás funciona exactamente igual que **parseInt**.<br>`...`<br>`var minum1 = "14.5E2";`<br>`document.write(parseInt(minum1));`<br>`....`<br>Escribirá el número 1450, 14.5 por 10 elevado a 2. |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| **Number(objArg)**<br>**String(objArg)** | Permiten convertir el objeto pasado como argumento a un número o a una cadena. Por ejemplo:<br>`...`<br>`var hoy = new Date();`<br>`hoy.getDate();`<br>`document.write(string(hoy)); ....`<br>Escribirá en pantalla la cadena "Sun Sep 3 20:40:05 UTC+0200 2000" si la fecha del día es domingo 3 de Septiembre y la hora es las 20:40:05. |
| **escape(cadarg)** | Codifica la cadena del argumento substituyendo todos los caracteres no ASCII por su código en el formato %xx. Por ejemplo:<br>`....`<br>`var cadena = "Buenos días"; document.write(escape(cadena));`<br>`...`<br>Produce la frase "Buenos d%EDas", pues la í (i acentuada) es el código hexadecimal ED de ese carácter. |
| **unescape(cadarg)** | Es inversa a la anterior, de manera que si la cadena contiene códigos del tipo %xx son convertidos al correspondiente carácter ASCII extendido.<br>`....`<br>`var cadena = "Buenos d%EDas"; document.write(escape(cadena));`<br>`.....`<br>Ahora se escribirá "Buenos días", se ha substituido %ED por su equivalente í (i acentuada). |

### 3.1.2.6 Expresiones Regulares

Las expresiones regulares constituyen un mecanismo bastante potente para realizar manipulaciones de cadenas de texto. El proceso para el que se usan estas expresiones, es el de buscar y/o sustituir una subcadena de texto dentro de otra cadena. En principio esto puede hacerse usando los métodos del objeto **string**, pero el problema surge cuando no tenemos una subcadena fija y concreta sino que queremos buscar un texto que responda a un cierto esquema, como por ejemplo: buscar aquellas palabras que comienzan con http: y finalizan con una \, o buscar palabras que contengan una serie de números consecutivos, etc.; es en estos casos cuando las expresiones regulares muestran toda su potencia. La subcadena que buscamos en el texto es lo que se llama un patrón y se construye encerrando entre dos barras inclinadas ( / ) una serie de caracteres normales y símbolos especiales llamados comodines o metacaracteres, (algo parecido a la orden dir *.bat usada en el DOS cuando queríamos listar los ficheros con extensión bat). Este patrón es una descripción del texto que se está buscando y JavaScript encontrará las subcadenas que concuerdan con ese patrón o definición. Las expresiones regulares se usan con el objeto **Regular Expresion** y también dentro de los métodos **String.match, String.replace, String.search** y **String.split**.
En la tabla que sigue se muestran los caracteres comodín usados para crear los patrones y su significado, junto a un pequeño ejemplo de su utilización.

| | Significado | Ejemplo | Resultado |
|---|---|---|---|
| \ | Marca de carácter especial | /\$ftp/ | Busca la palabra $ftp |
| ^ | Comienzo de una línea | /^-/ | Líneas que comienzan por - |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | | | |
|---|---|---|---|
| **$** | Final de una línea | /s$/ | Líneas que terminan por s |
| **.** | Cualquier carácter (menos salto de línea) | /\b.\b/ | Palabras de una sola letra |
| \| | Indica opciones | /(L\|l\|f\|)ocal/ | Busca Local, local, focal |
| **()** | Agrupar caracteres | /(vocal)/ | Busca vocal |
| **[]** | Conjunto de caracteres opcionales | /escrib[aoe]/ | Vale escriba, escribo, escribe |

La tabla que sigue describe los modificadores que se pueden usar con los caracteres que forman el patrón. Cada modificador actúa sobre el carácter o el paréntesis inmediatamente anterior.

| | **Descripción** | **Ejemplo** | **Resultado** |
|---|---|---|---|
| ***** | Repetir 0 o mas veces | /l*234/ | Valen 234, 1234, 11234... |
| **+** | Repetir 1 o mas veces | /a*mar/ | Valen amar, aamar, aaamar... |
| **?** | 1 o 0 veces | /a?mar/ | Valen amar, mar. |
| **{n}** | Exactamente n veces | /p{2}sado/ | Vale ppsado |
| **{n,}** | Al menos n veces | /(m){2}ala/ | Vale mmala, mmmala.... |
| **{m,n}** | entre m y n veces | /tal{1,3}a/ | Vale tala, talla, tallla |

Los siguientes son caracteres especiales o metacaracteres para indicar caracteres de texto no imprimibles, como puedan ser el fin de línea o un tabulador, o grupos predefinidos de caracteres (alfabéticos, numéricos, etc.).

| | **Descripción** | **Ejemplo** | **Resultado** |
|---|---|---|---|
| **\b** | Principio o final de palabra | /\bver\b/ | Encuentra ver en "ver de", pero no en "verde" |
| **\B** | Frontera entre no-palabras | /\Bver\B/ | Empareja ver con "Valverde" pero no con "verde" |
| **\d** | Un dígito | /[A-Z]\d/ | No falla en "A4" |
| **\D** | Alfabético (no dígito) | /[A-Z]\D/ | Fallaría en "A4" |
| **\O** | Carácter nulo | \t | Caracter ASCII 9 (tabulador) |
| **\f** | Salto de página | \n | Salto de línea |
| **\w** | Cualquier alfanumérico, [a-zA-Z0-9_ ] | /\w+/ | Encuentra frase en "frase.", pero no el . (punto). |
| **\W** | Opuesto a \w ([^a-zA-Z0-9_ ]) | /\W/ | Hallaría sólo el punto (.) |
| **\s** | Carácter tipo espacio (como tab) | /\sSi\s/ | Encuentra Si en "Digo Si ", pero no en "Digo Sientate" |
| **\S** | Opuesto a \s | \cX | Carácter de control X |
| **\c9** | El tabulador | \oNN | Carácter octal NN |
| **\xhh** | El hexadecimal hh | /\x41/ | Encuentra la A (ASCII Hex41) en "letra A" |
| **\b** | Principio o final de palabra | /\bver\b/ | Encuentra ver en "ver de", pero no en "verde" |
| **\B** | Frontera entre no-palabras | /\Bver\B/ | Empareja ver con "Valverde" pero |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | | | |
|---|---|---|---|
| | | | no con "verde" |
| **\d** | Un dígito | /[A-Z]\d/ | No falla en "A4" |

### 3.1.3 Gramática

#### 3.1.3.1 Condicionales

El orden en que se ejecutan las instrucciones de un programa es, por defecto, secuencial: ejecución instrucción tras instrucción. Así un programa se escribirá como una sucesión de instrucciones o sentencias, utilizando un punto y coma para indicar el final de la instrucción. Pueden agruparse una serie de sentencias en un bloque encerrándolas entre llaves. A veces es necesario alterar este orden para ello se utilizan las instrucciones de control: condicionales, selección y bucles.
Una sentencia condicional es una instrucción en la que se hace una comparación y según el resultado verdadero o falso (*true* o *false*) de la misma el programa seguirá ejecutando una u otra instrucciones. La condicional más simple que podemos escribir es aquella que ejecuta u omite una serie de sentencias dependiendo de si la comprobación da verdadero o falso. La sintaxis de esta sentencia es.
**if** (condición)
{bloque a ejecutar si la condición es cierta}
**else**
{bloque a ejecutar si la condición es false}

Si omitimos la parte del **else** tendremos una condicional simple. Esta sintaxis en algunos casos puede simplificarse utilizando la siguiente forma:
(condición) ?{bloque si cierta} : {bloque si falsa}
En el siguiente ejemplo evitamos realizar una división por cero

```
.......
if (div == 0)
     alert('No se puede dividir por 0');
else
     coc = num / div;
.........
```

Otro ejemplo usando la segunda forma:

```
............
cad = (num >= 0) ? ' + ' : ' - ';
........
```

En este ejemplo **cad** tomará el valor + si num es positivo o cero y el - si es negativo. Las sentencias **if** pueden anidarse, es decir, dentro de una sentencia if pueden meterse mas sentencias **if**. Las condiciones pueden ser sencillas como en estos ejemplos o pueden enlazarse usando los operadores **&&** y || (AND y OR lógicos). Veamos un ejemplo en el que comprobamos si un número está comprendido entre 1 y 5:

```
if ((num>=1) y (num < 5)
{
     lista[indice] = 'Muy bajo';
     bajos++;
}
indice++;
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

En este ejemplo si **num** está entre 1 y 5 (excluido) se anota en una lista la palabra 'Muy bajo' y se incrementa la variable **bajos**. Como vemos no se ha usado la parte de **else** y como se deben ejecutar mas de una sentencia las hemos encerrado entre llaves. Si **num** no cumple la condición el programa se salta este bloque. En cualquier caso la siguiente instrucción que se ejecute tras el condicional será la que incrementa el valor de **indice**.

#### 3.1.3.2 Selección Múltiple

Pueden existir casos en los que el programa deba tener mas de dos alternativas, por ejemplo si queremos un programa que presente un título en un idioma de cuatro posibles. Esto puede solucionarse mediante varios if anidados, siguiendo el ejemplo tenemos que elegir entre idiomas: castellano, ingles, francés y alemán.

```
if (leng == "castellano")
     pagCast();
else
     if (leng == "ingles")
          pagIng();
     else
          if (leng == "frances")
               pagFran();
          else
               if (leng == "alemán")
                    pagAlemt();
               else
                    error('idioma no presente');
```

Como vemos resulta un código bastante complejo. Para estos casos disponemos de la sentencia switch...case...default, de selección múltiple. El ejemplo anterior quedaría:

```
... ... ...
switch (idioma) {
     case 'castellano' :
          pagCast();
          break;
     case 'ingles' :
          pagIng();
          break;
     case 'frances' :
          pagFran();
          break;
     case 'alemán' :
          pagAlem();
          break;
     default :
          error ('Idioma no presente');
}
```

Durante la ejecución se compara la variable **idioma** con cada uno de los posibles valores y cuando coincidan ejecuta el código correspondiente. La instrucción **break** pone fin al bloque y hace que el programa salte a la instrucción siguiente a la sentencia switch(), si se omite el programa continuaría con la siguiente comparación. La sección del **default** es opcional, su finalidad es ejecutar algún código cuando ninguna de las condiciones se cumpla.

#### 3.1.3.3 Bucles

A veces es necesario repetir un mismo conjunto de sentencias varias veces. Por ejemplo para borrar todos los elementos de un array simplemente debemos hacer delete en cada uno de ellos,

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

es una sentencia que se repetirá tantas veces como largo sea el array. En esencia la ejecución de un bucle consiste en ejecutar repetidas veces una misma parte del programa (cuerpo del bucle) hasta que se cumpla una determinada condición, en cuyo caso se acaba la repetición y el programa continúa con su flujo normal. Existen varias sentencias de bucles:

- **Sentencia while:** En esta estructura el programa primero comprueba la condición: si es cierta pasa a ejecutar el cuerpo del bucle, y si es falsa pasa a la instrucción siguiente a la sentencia while. Como siempre un ejemplo lo aclarará todo:

```
var lista = new Array(10);
var ind=0;
while (ind < 10)
{
     lista[ind] = '0';
     ind++;
}
```

  En este ejemplo mientras que el valor almacenado en **ind** sea menor que 10 (la longitud del **array**) irá almacenando en cada elemento del array lista un 0 e incrementando el valor de **ind**. Cuando este valor sea 10 el programa no entrará en el cuerpo del bucle. Si no se incrementara el valor de i**nd** el bucle no acabaría nunca, el programa quedaría ejecutando indefinidamente el cuerpo del bucle.
- **Sentencia do...while:** Se trata de un bucle en el que la condición se comprueba tras la primera iteración, es decir que el cuerpo del bucle se ejecuta al menos una vez. El ejemplo anterior quedaría como sigue

```
var lista = new Array(10);
var ind=0;
do
     lista[ind] = '0';
     ind++;
while (ind < 10)
```

  Como vemos aquí no son imprescindibles las llaves para encerar el cuerpo del bucle. No está contemplada en el standard ECMA 1.5.
- **Sentencia for:** Esta sentencia utiliza una variable de control a modo de contador para controlar la repetición del cuerpo del bucle. La sentencia da un valor inicial a este contador y en cada iteración lo modifica según le indiquemos y comprueba la condición, si se cumple ejecuta el cuerpo del bucle, si no lo salta y continúa por la siguiente sentencia. Vemos el ejemplo anterior usando esta sentencia:

```
var lista = new Array(10);
var ind;
for (ind=0; ind < 10; ind++)
{
     lista[ind] = '0';
}
```

  Como vemos el cuerpo del bucle no incrementa la variable **ind**, esto se indica en la cabecera de la sentencia.
- **Sentencia for ... in:** Se trata de una variante de la sentencia **for** utilizada para iterar o recorrer todos los elementos de un objeto o de un array. Usa una variable de control que en cada


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

iteración toma el valor del elemento del objeto recorrido. Por ejemplo si pruebas este código podrás ver todos los elementos del objeto document

```
var item; for (item in document)
document.write(item+'<br>');
```

Con una matriz la variable de control toma el valor de los índices de la matriz, no su contenido.

#### 3.1.3.4 Ruptura de Bucles

En ocasiones puede ser necesario interrumpir la repetición de un bucle o forzar una iteración del mismo, esto puede lograrse mediante las sentencias **break** y **continue**. Son sentencias aplicables a cualquiera de las estructuras de bucle en JavaScript.

- **break**: La sentencia break interrumpe la iteración actual y envía al programa a la instrucción que sigue al bucle.

```
var lista = new Array ('a','b','c','z','x','f');
var item ;
for (item in lista)
{
     if (lista[item] == "z")
               continue;
     document.write(lista[item]+'<br>');
}
```

Este ejemplo escribiría el contenido del array lista hasta encontrar una letra **z**.

- **continue**: La sentencia **continue** interrumpe la iteración actual y envía al programa a la comprobación de la condición, si esta es cierta continúa con la siguiente iteración.

```
var lista = new Array ('a','b','c','z','x','f');
var item ;
for (item in lista)
}
    if (lista[item] == "z")
         continue;
    document.write(lista[item]+'<br>');
{
```

Este ejemplo escribiría el contenido del array saltándose la letra **z**.

### 3.1.4 Objetos

#### 3.1.4.1 Arrays

Como objetos que son, los **arrays** poseen sus **propiedades** y **métodos** predefinidos, que son ampliables por el usuario. Es necesario hacer notar que estos métodos y propiedades son los definidos para el JavaScript 3.0 de Microsoft. Netscape añade mas métodos en su versión, pero los aquí definidos son comunes a ambos navegadores.

**Propiedades:**

- **length:** Como su nombre indica esta propiedad nos devuelve la longitud del array, es decir, el número de elementos que puede almacenar. Su uso es muy simple:

```
var lista = new Array(50);
tamagno = lista.length;  /*tamagno almacenaría el valor 50 */
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **prototype:** Esta es una propiedad muy potente en el sentido que nos permite agregar al objeto Array las propiedades y métodos que queramos.

```
Array.protoype.descriptor = null;
dias = new Array ('lunes', 'Martes', 'Miercoles', 'Jueves',
'Viernes');
dias.descriptor = "Dias laborables de la semana";
```

En este ejemplo hemos creado una nueva propiedad para el objeto array, la propiedad descriptor que podría utilizarse para darle un título a la matriz.

**Métodos**

- **concat(objArray):** Une el objeto Array con el array que se le pasa como argumento y devuelve el resultado en un nuevo array, sin modificar los arrays que se concatenan.
- **join():** Convierte los elementos de un array en una cadena separados por el carácter que se le indique. El separador por defecto es la coma.

```
a= new Array("Hola","Buenos","días");
document.write(a.join() +" <br>");
document.write(a.join(", ") +" <br>");
document.write(a.join(" + ") +" <br>") ;
```

La salida de este programa sería
Hola,Buenos,Días
Hola, Buenos, Días
Hola+Buenos+Días

- **reverse():** Invierte el orden de los elementos de un Array en el propio array, sin crear uno nuevo.
- **slice(ini, fin):** Extrae parte de un Array devolviéndolo en un nuevo objeto Array.

```
lista = new Array('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h');
sublista = lista.slice(2,6);
alert(sublista.join());
```

En el ejemplo sublista contendrá los elementos desde el índice 2 al 5 ambos inclusive, o sea, 'c', 'd', 'e', 'f'. Si se omite el segundo argumento se extrae hasta el último elemento del array y si es negativo se entiende como contando desde el final.

- **sort(rutord):** Ordena alfabéticamente los elementos de un objeto Array. Opcionalmente podemos pasar como argumento una función para determinar el orden, esta función posee dos argumentos y devolverá un valor negativo si el primer argumento es menor que el segundo, cero si son iguales y un valor positivo si el primer argumento es mayor que el segundo. En castellano esto es necesario si queremos que la ñ y vocales acentuadas figuren en su lugar.

#### 3.1.4.2 Booleanos

Las variables booleanas o lógicas son las que sólo pueden tomar dos valores: **true**, verdadero, y **false**, falso. Este tipo de variables está implementado en JavaScript como un objeto.

**Métodos**

- **toString:** Si el valor es **false** devuelve la cadena "false" y si es **true** devuelve la cadena "true".
- **valueOf**: Devuelve el valor booleano (**true** o **false**)

**Propiedades**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **constructor:** Heredada del objeto genérico **Object,** devuelve la referencia al constructor: function Boolean() { [native code] }
- **prototype:** Es una propiedad utilizada para asignar nuevos métodos o propiedades, heredado del objeto genérico **Object**. Por ejemplo podemos traducir los valores **true o false.**

```
function valor () { return this.valueOf()?'cierto':'falso' }
Boolean.prototype.valor = valor;
var item = new Boolean(false);
document.write(item.valor());
```

Con este ejemplo logramos que **true** se devuelva como la cadena "cierto" y **false** como la cadena "falso".

#### 3.1.4.3 Objeto Function

Permite la creación de **funciones**, ya sean con nombre o anónimas. La creación de una función puede realizarse por el método tradicional y común a la mayoría de lenguajes de programación:

```
function sumar(a, b) { return a+b; }
```

O bien mediante el conocido operador **new**:

```
sumar = new Function ("a", "b", "return a+b");
```

En cualquier caso la función se usará de igual forma:

```
document.write( sumar(90, 100) );
```

**Métodos**

Los heredados del objeto Object

**Propiedades**

- **arguments:** Se trata de un array que contiene los argumentos pasados a la función. Esta propiedad permite el uso de funciones con un número variable de argumentos.
- **caller:** Contiene una referencia a la función que llamó a la actual.
- **Constructor**: Heredada de la clase **Object.**

#### 3.1.4.4 Objeto Number

Es el objeto destinado al manejo de datos y constantes numéricas. Realmente no es habitual crear objetos de este tipo por cuanto JavaScript los crea automáticamente cuando es necesario. No obstante la sintaxis para su creación es la habitual para cualquier objeto:
miuúmero = new Number(valorinicial)
El valor inicial es optativo, si no se usa el objeto se crea con valor null.

**Métodos**

Los heredados del objeto **Object**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

**Propiedades**

Además de las heredadas del objeto **Object** Number posee las siguientes propiedades:

- **MAX_VALUE**: Indica el valor máximo utilizable por JavaScript, actualmente 1.79E+308.
- **MIN_VALUE**: Indica el valor mínimo utilizable por JavaScript, actualmente 2.22E-308.
- **NaN**: Una constante usada para indicar que una expresión ha devuelto un valor no numérico. NaN no puede compararse usando los operadores lógicos habituales, para ver si un valor es igual a NaN se debe usar la función incorporada **isNaN**
- **NEGATIVE_INFINITY**: Una constante para indicar infinito positivo, es decir, un valor superior al MAX_VALUE.
- **POSITIVE_INFINITY**: Una constante para indicar infinito negativo, es decir, un valor superior al MAX_VALUE con signo negativo.

#### 3.1.4.5 Objeto Object

Es un objeto llamado **Object** del que derivan todos los objetos de JavaScript, los predefinidos y los definidos por el usuario. Esto significa que los objetos usados en JavaScript heredan las propiedades y métodos de **Object**.

**Métodos**

- **toString**: Devuelve una cadena dependiendo del objeto en que se use

| **Objeto** | **Cadena devuelta por el método** |
|---|---|
| **Array** | Los elementos del array separados por coma |
| **Boolean** | Si el valor es false devuelve "false" si no devuelve "true" |
| **Function** | La cadena "function nombre_de_función(argumentos){ [código]}" |
| **Number** | Representación textual del número |
| **String** | El valor de la cadena |
| **Default** | "[object nombre_del_objeto]" |

- **valueOf:** Devuelve el valor del objeto dependiendo del objeto en que se use

| **Objeto** | **Valor que devuelve el método** |
|---|---|
| **Array** | Una cadena formada por los elementos separados por coma |
| **Boolean** | El valor booleano (true o false) |
| **Date** | La fecha como el número de milisegundos desde el 1/1/1970, 00:00 |
| **Function** | La propia función |
| **Number** | El valor numérico |
| **String** | La cadena |

**Propiedades**

- **constructor:** Esta propiedad contiene una referencia a la función que crea las instancias del objeto en particular. Por ejemplo:


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
x = new String("Hola"); //En este caso s.constructor contendrá
function String()
{
[native code]
}
```

- **prototype:** Es una propiedad utilizada para asignar nuevos métodos o propiedades a un objeto, elementos estos que serán heredados por las diferentes instancias de ese objeto.

  Ejemplo:

```
Array.prototype.nombTipo = "matriz"; lista = new Array(9);
document.write(lista.nombTipo); //Escribirá la palabra matriz
que es el nombTipo //que hemos dado para el objeto Array
```

#### 3.1.4.6 Objeto Regular Expression

JavaScript usa este objeto para trabajar con patrones de expresiones regulares, estos patrones se crean como cualquier otro objeto mediante su inicialización directo o bien mediante el constructor **new RegExp()**, como podemos ver en el ejemplo:

```
......
var mipatron = /^[aeiou]/gi
var mipatron2 = new RegExp("^[aeiou]", "gi")
........
```

Ambas formas conducen al mismo patrón, en este ejemplo define palabras que comienzan con una vocal. El patrón puede llevar modificadores o flags para matizar la forma de buscar, en el ejemplo se usan dos: g i. Estos modificadores tienen los siguientes significados:

| **flags** | **Significado** |
|---|---|
| **g** | Explorar la cadena completa |
| **i** | No distinguir mayúsculas de minúsculas |
| **m** | Permite usar varios ^y $ en el patrón |
| **s** | Incluye el salto de línea en el comodín punto . |
| **x** | Ignora los espacios en el patrón |

Estos patrones poseen en total tres métodos **exec(), test(), compile()** además de los métodos ya citados del objeto String que también usan patrones como son: **match(), replace(), search() y split()**. La única propiedad que funciona en estos objetos es la **source** que refleja el contenido del patrón. En el patrón pueden aparecer caracteres o grupos de caracteres encerrados entre paréntesis, posteriormente podemos usar un índice para referirnos individualmente al contenido de esos paréntesis.

Por ejemplo

```
var cadexp = "234879x089h9y7";
var patron = /([a-z])(\d)/ig;
document.write(cadexp+'<br> ');
cadexp = cadexp.replace(patron, "$2-");
document.write(cadexp)
```

Como ves donde antes existía un dígito seguido de una letra ahora hay un dígito seguido de un guión. Las coincidencias con el primer paréntesis del patrón están en $1 y con el segundo en $2. La primera coincidencia hallada es x0, luego $1 contiene **x** y $2 contiene **0**, sustituyo lo hallado con -$2, o sea, quito $1 y pongo un guión y me quedará -0 en lugar de x0. Como se ha usado el flag g (global) esta operación se realiza en toda la cadena.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 3.1.4.7 Objeto String

El objeto **String** se usa para manipular cadenas de caracteres. En JavaScript todo texto encerrado entre comillas, dobles o simples, se interpreta como una cadena, así '45' no es el número cuarenta y cinco sino la cadena formada por los caracteres 4 y 5. El objeto **String** permite realizar operaciones con cadenas como concatenar o dividir cadenas, buscar texto, extraer parte de un texto, etc.. La operación de crear una variable de este tipo se lleva a cabo como es habitual con el operador **new** pudiéndole pasar un argumento para inicializar la variable. Al usar un método o referirnos a una propiedad podemos usar el nombre de la variable o una constante de cadena así el ejemplo

```
var mitexto = "Esta es una cadena"; var pos = mitexto.indexOf("una")
```

puede también escribirse en la siguiente forma:

```
var pos = "Esta es una cadena". indexOf("una");
```

**Propiedades**

- **length**: devuelve la longitud de la cadena.
- **prototype**: permite agregar métodos y propiedades al objeto

**Métodos**

| | |
|---|---|
| **anchor** | Este método crea, a partir de un objeto String, una cadena conteniendo un elemento HTML ANCHOR, con el atributo NAME igual a la cadena que se le pase |
| **big** | Este método devuelve una cadena consistente en el objeto String rodeado con las etiquetas <BIG> </BIG> del lenguaje HTML |
| **blink** | Este método devuelve una cadena consistente en el String rodeado con las etiquetas <blink></blink> del lenguaje HTML |
| **bold** | Este método devuelve una cadena consistente en el String rodeado con las etiquetas <B> </B>, negrita, del lenguaje HTML |
| **charAt** | Este método aplicado a una cadena devuelve el carácter que se encuentra en la posición dada por el atributo |
| **charCodeAt** | Este método aplicado a una cadena devuelve el código Unicode del carácter que se encuentra en la posición dada por el atributo |
| **concat** | Este método aplicado a una cadena le agrega la cadena pasada como atributo, que será una variable o constante literal, cualquier otro tipo es convertido a cadena |
| **fixed** | Este método devuelve una cadena consistente en el objeto String rodeado con las etiquetas <TT> </TT>, espaciado fijo o teletype, del lenguaje HTML |
| **fontcolor** | Este método crea, a partir de un objeto String, una cadena conteniendo un elemento FONT del lenguaje HTML con el atributo COLOR igual a la cadena que se le pase |
| **fontsize** | Este método crea, a partir de un objeto String, una cadena conteniendo un elemento FONT del lenguaje HTML con el atributo SIZE igual al valor entero que se le pase |
| **fromChar-Code** | Este es un método global del objeto String que crea una cadena a partir de los códigos Unicode que se le pasen como parámetros |


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| **indexOf** | Este método devuelve la primera posición dentro del objeto String donde comienza la subcadena pasada como argumento. Admite un segundo argumento opcional que indica desde que posición debe realizar la búsqueda, si se omite comienza a buscar por el primer carácter de la izquierda |
| **italics** | Este método devuelve una cadena consistente en el String rodeado con las etiquetas <I></I>, cursivas, del lenguaje HTML |
| **lastIndexOf** | Este método devuelve la primera posición dentro del objeto String donde comienza la subcadena pasada como argumento, pero realizando la búsqueda de derecha a izquierda |
| **link** | Este método crea, a partir de un objeto String, una cadena conteniendo un elemento ANCHOR del lenguaje HTML, con el atributo HREF igual a la cadena que se le pase |
| **match** | Este es uno de los más potentes métodos para buscar subcadenas y realizar sustituciones dentro de cadenas de texto. Permite usar patrones de búsqueda construidos con comodines y texto, lo que se denominan expresiones regulares |
| **replace** | Con este método todas las cadenas que concuerden con el primer argumento son reemplazadas por la cadena especificada en el segundo argumento, que puede contener elementos del patrón mediante los símbolos $1 a $9. El resultado devuelto es la cadena con las sustituciones realizadas |
| **search** | Este método realiza una búsqueda en la cadena y devuelve el índice donde se produce la primera concordancia con el patrón o -1 si no existe ninguna |
| **slice** | Este método devuelve la porción de cadena comprendida entre las posiciones dadas por los argumentos inicio y ultimo, o el final de la cadena si se omite este segundo argumento. Si ultimo es negativo, se interpreta como número de posiciones contadas desde el final de la cadena. Si los argumentos no son números enteros, por ejemplo cadenas, se convierten a números enteros |
| **small** | Este método devuelve una cadena consistente en el objeto String rodeado con las etiquetas <SMALL> </SMALL>, reducir tamaño, del lenguaje HTML |
| **split** | Devuelve un array conteniendo las porciones en que queda separada la cadena por el separador indicado mediante el argumento, que será una expresión regular o una cadena literal |
| **strike** | Este método devuelve una cadena consistente en el String rodeado con las etiquetas <STRIKE> </STRIKE>, tachado, del lenguaje HTML |
| **sub** | Este método devuelve una cadena consistente en el objeto String rodeado con las etiquetas <SUB> </SUB>, subíndice, del lenguaje HTML |
| **substr** | Devuelve una subcadena extraída del objeto string comenzando por la posición dada por el primer argumento y con un número de caracteres dado por el segundo argumento |
| **substring** | Devuelve una subcadena del objeto string que comienza en la posición dada por el menor de los argumentos y finaliza en la posición dada por el otro argumento. Si se omite este último argumento la subcadena extraída va desde la posición indicada por el único argumento hasta el final de la cadena. Si los argumentos son literales se convierten a enteros |
| **sup** | Este método devuelve una cadena consistente en el objeto String rodeado con las etiquetas <SUP> </SUP>, superíndice, del lenguaje HTML |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| **toLowerCase** | Devuelve una cadena igual a la original pero con todos los caracteres en minúsculas. No afecta a caracteres no alfabéticos, o sea, a los números, letras acentuadas y caracteres especiales como la Ñ |
| **toUpperCase** | Devuelve una cadena igual a la original pero con todos los caracteres en mayúsculas. No afecta a caracteres no alfabéticos, o sea, a los números, letras acentuadas y caracteres especiales como la Ñ |

#### 3.1.4.8 Objeto Date

El objeto Date contiene un valor que representa fecha y hora de un instante dado. Para crear una instancia de este objeto usamos alguna de las siguientes sintaxis:

```
var fecha= new Date()
var fecha= new date(número)
var fecha= new date(cadena)
var fecha= new date(año, mes, día[, hora[, minutos[, seg[,ms]]]])
```

Los argumentos encerrados entre corchetes son opcionales. En la primera forma la variable **fecha** contendrá la fecha del día actual. La segunda opción almacena en fecha la fecha dada por el argumento como el número de milisegundos transcurridos desde la media noche del 1 de Enero de 1970. El tercer tipo se usa cuando la fecha se pasa en forma de cadena. Por último la fecha puede crearse pasándole como argumento los números de año, mes, día, hora y opcionalmente, hora, minuto, segundo y milisegundo. Los años posteriores a 1970 puede escribirse con dos dígitos, pero es aconsejable usar siempre cuatro dígitos por aquello de los efectos 2000.

```
var hoy = new date()     /*fecha del día en hoy */
var evento = new Date("November 10 1990");
var otro = new Date("10 Nov 1990");
var otro = new Date("10/02/2000"); //Oct, 2, 2000
var instante = new Date(1990, 11, 10, 20,00);
```

Estas son tres posibles formas de declarar objetos de tipo fecha. Las dos últimas almacenan el mismo día, pero en la última además se guarda la hora. Donde se usen cadenas para indicar una fecha podemos añadir al final las siglas GMT (o UTC) para indicar que la hora se refiere a hora del meridiano Greenwich, si no se toma como hora local, o sea, según la zona horaria configurada en el ordenador donde se ejecute el script [19-21].

#### 3.1.4.9 Objeto Math

Es el objeto que usa JavaScript para dotar al lenguaje de funciones matemáticas avanzadas y de constantes predefinidas, como el número PI.

**Propiedades**

Son las habituales constantes como el número e, PI y algunos otros valores habituales en cálculos matemáticos.

| | |
|---|---|
| **E** | Constante de Euler, la base para los logaritmos naturales |
| **LN10** | Logaritmo natural de 10 |
| **LOG10E** | Logaritmo en base 10 de E |
| **SQRT1_2** | Raíz cuadrada de 0.5 o sea la inversa de la raíz de 2 |
| **LN2** | Logaritmo natural de 2 |
| **LOG2E** | Logaritmo en base 2 de E |
| **PI** | El número pi |
| **SQRT2** | Raíz cuadrada de 2 |

**Métodos**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| **abs** | Devuelve el valor absoluto, o sea, sin signo, del argumento. Si el argumento fuera no entero será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **acos** | Es una función trigonométrica que sirve para calcular el ángulo cuyo coseno es el valor dado por el argumento, el arccos(). Este argumento deberá ser una expresión numérica o transformable en numérica, comprendida entre -1 y +1 y el ángulo devuelto viene dado en radianes |
| **asin** | Es una función trigonométrica que sirve para calcular el ángulo cuyo seno es el valor dado por el argumento, es decir, el llamado arcosen. Este argumento deberá ser una expresión numérica, o transformable en numérica, comprendida entre -1 y +1 y el ángulo devuelto viene dado en radianes |
| **atan** | Es una función trigonométrica que sirve para calcular el ángulo cuya tangente es el valor dado por el argumento, o sea el arctg(). Este argumento deberá ser una expresión numérica o transformable en numérica, sin límites, y el ángulo devuelto viene dado en radianes |
| **atan2** | Es una función trigonométrica que sirve para calcular el ángulo cuya tangente es el cociente de sus argumentos, en otras palabras devuelve el ángulo desde el origen de coordenadas hasta un punto cuyas coordenadas son los argumentos de la función. Los argumentos deberán ser numéricos o transformables en numéricos, y el ángulo devuelto viene dado en radianes |
| **ceil** | Devuelve el valor del argumento redondeado por exceso, es decir el menor número entero mayor o igual al argumento. Si el argumento fuera no numérico será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **cos** | Es una función trigonométrica que sirve para calcular el coseno del ángulo pasado como argumento en radianes |
| **exp** | Devuelve el valor del número e (constante de Euler, aproximadamente 2,178) elevada al exponente dado por el argumento. Si el argumento fuera no entero será convertido a numérico siguiendo las reglas de las funciones **parseInt()** o **parseFloat()** |
| **floor** | Devuelve el valor del argumento redondeado por defecto, es decir, el mayor número entero menor o igual al argumento. Si el argumento fuera no numérico será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **log** | Devuelve el logaritmo natural o neperiano, o sea, en base al número e, del argumento. Si el argumento fuera no numérico será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()**. Si el argumento fuera un valor negativo esta función devuelve **NaN** |
| **max** | Devuelve el mayor de los dos números o expresiones numéricas pasadas como argumentos. Si alguno de los argumentos fuera no numérico será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **min** | Devuelve el menor de los dos números o expresiones numéricas pasadas como argumentos. Si alguno de los argumentos fuera no numérico será convertido a numéricos siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **pow** | Calcula la potencia de un número, dado por el primer argumento, elevado al exponente dado por el segundo. Si alguno de los argumentos fuera no numérico será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |

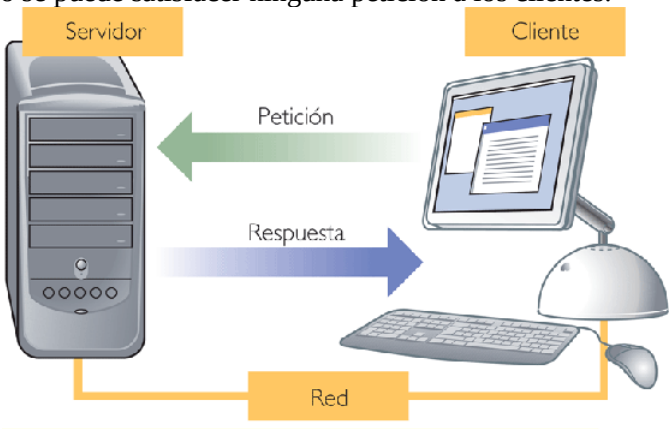
ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

| | |
|---|---|
| **random** | Calcula un número aleatorio, realmente seudo-aleatorio, comprendido entre 0 y 1 ambos inclusive. Cada vez que se carga el intérprete de JavaScript se genera una semilla base para el cálculo. No lleva argumentos |
| **round** | Devuelve el valor entero mas próximo al número pasado como argumento, es decir, redondea. Si la parte decimal del argumento es 0.5 o mayor devuelve el primer entero por encima del argumento (redondeo por exceso) en caso contrario devuelve el entero anterior al argumento (redondeo por defecto). Si el argumento fuera no entero será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat()** |
| **sin** | Es una función trigonométrica que sirve para calcular el seno del ángulo pasado como argumento en radianes. Este argumento deberá ser una expresión numérica o transformable en numérica. |
| **sqrt** | Devuelve la raíz cuadrada del valor pasado como argumento. Si el argumento fuera no entero será convertido a numérico siguiendo las reglas de la función **parseInt()** o **parseFloat().** Si el argumento fuera negativo o cualquier otro valor no numérico devolverá **NaN** |
| **tan** | Es una función trigonométrica que sirve para calcular la tangente del ángulo pasado como argumento en radianes. Este argumento deberá ser una expresión numérica o transformable en numérica. |

### 3.2 PHP: Hipertext preprocessor

Internet, generalmente, funciona gracias a una arquitectura denominada cliente servidor que consta de dos elementos básicos: un cliente y un servidor. El cliente es una aplicación que se conecta al servidor y que utiliza los servicios que éste ofrece, mientras que el servidor es la aplicación que provee de servicios a otras aplicaciones cliente, la comunicación entre el cliente y el servidor se suele realizar mediante una red de ordenadores. Tanto el cliente puede hacer peticiones a diversos servidores, como el servidor puede atender a diversos clientes.
Esta arquitectura tiene diversas ventajas a nivel organizativo como es la centralización de la información, la facilidad de mantenimiento, la escalabilidad y la separación de responsabilidades o tareas. Las principales desventajas son la congestión del tráfico que se produce en el servidor, la alta carga de tareas a la que esta sometido y la falta de robustez ante fallos, ya que si falla el elemento servidor no se puede satisfacer ninguna petición a los clientes.




La separación entre cliente y servidor, es una separación de tipo lógica, ya que ni el servidor es necesario que se ejecute en una sola máquina, ni que sea un único programa. En el contexto en el que nos encontramos, el cliente es un agente de usuario, es decir, un navegador Web del tipo que


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

sea (visual, texto, etc.) y el servidor suele ser un conjunción de varios programas (servidor de aplicaciones, servidor de correo electrónico, servidor de bases de datos, etc.).
Siguiendo con esta distinción entre cliente y servidor, en Internet, también podemos diferenciar dos tipos de lenguajes:

- **Lenguajes del lado del cliente.** Este tipo de lenguajes son aquellos que se interpretan únicamente por el agente de usuario, los servidores en este caso únicamente devuelve el contenido del fichero que el cliente haya solicitado y es el propio cliente el que se encarga de interpretar el código que en él se le presenta.
- **Lenguajes de servidor.** Estos lenguajes, en cambio, requieren un procesamiento previo al envío hacia el cliente, este procesamiento se realiza en el servidor y tiene como resultado un documento escrito en algún tipo de lenguaje que el cliente (navegador) pudiera comprender e interpretar.

Si nos fijamos en los lenguajes que hemos estudiado hasta ahora HTML, CSS y Javascript, todos ellos son lenguajes de lado del cliente, es decir, el servidor envía al navegador Web el documento tal y como lo ha escrito el programador y es el propio navegador el que interpreta su contenido.


En cambio, el lenguaje PHP (acrónimo de "PHP: Hipertext Preprocessor") es un lenguaje interpretado de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Veamos un ejemplo, el siguiente código a simple vista nos puede parecer bastante familiar, tiene la estructura típica de cualquier documento HTML, pero como podemos apreciar también tiene incrustado un código en lenguaje PHP.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
	<head>
		<title>Ejemplo php</title>
	</head>
	<body>
		<?php
			echo "Hola mundo!";
		?>
	</body>
</html>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

El código anterior se procesaría, de forma previa al envío por parte del servidor y se le enviaría al cliente en un lenguaje que pueda interpretar, en este caso lenguaje HTML, tal y como se aprecia en el siguiente código

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
	<head>
		<title>Ejemplo php</title>
	</head>
	<body>
	Hola mundo
	</body>
</html>
```

### 3.2.1 Ventajas

- Es un lenguaje multiplataforma, soportado en sistemas UNIX, Windows 32 bits, QNX, Mac, OS/2, etc.
- Soporta casi todos los motores de base de datos actuales, aunque destacan MySQL y PostgreSQL.
- Extensibilidad de sus funcionalidades mediante el uso de módulos (llamados exts o extensiones).
- Existe una amplia comunidad de desarrolladores que emplean PHP y que ayudan en su desarrollo, testeo y documentación.
- Buena documentación en su página oficial (www.php.net) y a través de la amplia comunidad de desarrolladores.
- Es software abierto, con todas las ventajas que esto supone. Especialmente respecto a los costes de licencia.
- La biblioteca nativa de funciones incluida por defecto es realmente amplia. Además existen proyectos tales como PEAR (PHP Extension and Application Repository) y PECL (PHP Extension Community Library) con muchas más bibliotecas que extienden las funcionalidades básicas.
- El desarrollo de aplicaciones suele ser mucho más rápido que con otras plataformas tales como Java, C, C++, etc.

### 3.2.2 Inconvenientes

- Las aplicaciones en PHP suelen tener problemas de escalabilidad. Aunque se está trabajando en ello, es difícil que PHP alcance el nivel de escalabilidad disponible plataformas tales como los servidores de aplicaciones.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- A pesar de que puede funcionar en distintas plataformas, PHP tiene como entorno principal a la combinación de Linux, Apache y MySQL. Esto hace que bajo otros entornos su gestión y configuración no sea siempre sencilla.
- La migración de un servicio desarrollado sobre PHP a otra plataforma es complicada, teniendo en cuenta que se trata de un lenguaje embebido en la capa de presentación (HTML), no estándar.
- PHP carece, por el momento, de interfaz gráfica, herramientas de monitorización o de características visuales para un sencillo mantenimiento.

### 3.2.3 Referencia del lenguaje

#### 3.2.3.1 Sintaxis básica

Como hemos comentado previamente, el lenguaje PHP se incrusta en las páginas HTML, para ello existen diversos métodos, aunque el más ortodoxo y aconsejable es utilizar las etiquetas <?php y ?>, tal y como se muestra a continuación:

```
<body>
      Este código es HTML
      <?php echo ("este código es PHP y se ejecuta en el servidor"); ?>
      <?php echo ("sin punto y coma al final") ?>
      Este código es HTML
</body>
```

Como vemos las diferentes instrucciones se separan, al igual que en otros lenguajes mediante punto y coma (;), la etiqueta de cierre (?>) también implica cierre de sentencia, por lo que no es obligatorio el punto y coma. Por último en el ejemplo previo, se utiliza la función echo de PHP, que simplemente devuelve el valor de la cadena que se le suministra como parámetro. Siguiendo este mismo esquema se pueden crear estructuras mucho más avanzadas, tal y como vemos a continuación:

```
<body>
      <?php
            $i = 1;
            while ($i <= 5)
            {
      ?>
            Número: <?= $i ?>
      ¿>
            }
      ?>
</body>
```

Los comentarios del código se realizan tal y como se muestra en el siguiente ejemplo:

```
<?php
```

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
      echo “Esto realmente funciona”;
      /* voy a sumar dos números */ $i = 1 + 2
      //Este es un comentario de línea tipo c
      #Este es otro comentario de línea tipo shell
¿>
```

#### 3.2.3.2 Tipos de datos

PHP es un lenguaje dinámicamente tipado (no confundir con lenguaje débilmente tipado), esto quiere decir, que el tipo de la variable normalmente no lo indica el programador, sino que se decide en tiempo de ejecución dependiente del contexto en el que se utilice esa variable. Para obligar que una variable se convierta a un tipo concreto se puede utilizar la función settype().
PHP soporta los siguientes tipos primitivos:

- Cuatro tipos escales:
  - boolean
  - integer
  - float
  - string
- Dos tipos compuestos:
  - array
  - object
- Dos tipos especiales:
  - Resource
  - NULL

##### 3.2.3.2.1 Tipos escalares

Los tipos escalares expresan un valor de verdad, pueden ser TRUE o FALSE, este tipo de dato no es sensible a mayúsculas/minúsculas:

```
<?php
      $cierto = True //Asigna TRUE a la variable $cierto
?>
```

Los enteros son números del conjunto {… -2, -1, 0, 1 , 2, …} se pueden especificar utilizando alguna de las siguientes sintaxis:

```
      $a = 1234; #número decimal
      $a = 0123; #número octal
      $a = 0x12; #número hexadecimal
```

Por su parte, el tipo float, también llamado double en lenguaje PHP, puede ser utilizando mediante las siguientes sintaxis:

```
$a = 1234;
$a = 1.2e3;
$c = 7E-10
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

El tipo string se corresponde con una serie de caracteres, aunque existen diferentes posibilidades de representación, normalmente se utilizan las comillas dobles (“), tal y como se muestra a continuación:

```
$a = “esto es una cadena simple”
      $a = “esto es una cadena” . “ y esto es otra cadena,” . “se
      concatenan string mediante el operador punto (.)”
```

Al igual que en otros lenguajes, se utiliza el carácter barra invertida (\) para especificar caracteres especiales, en la siguiente tabla podemos ver algunos ejemplos:

| Secuencia | Significado |
|---|---|
| \n | Fin de línea (retorno de carro) |
| \t | Tabulador |
| \\ | Barra invertida |
| \” | Comilla doble |

3.2.3.2.2 Tipos compuestos

3.2.3.2.2.1 Arrays

Los arrays en PHP son mapas ordenados, un mapa es un tipo que asocia un valor a una clave, esta clave puede ser tanto un entero, como un string. Para definir arrays se utiliza el constructor del lenguaje array(), los diferentes elementos se separan por comas y tienen una estructura key => value. También se permite fijar únicamente los valores de array, sin clave, en este caso PHP asumirá como claves el conjunto de números enteros empezando por el 0. Para acceder a los elementos del array se utilizan las llaves, encerrando entre ellas la clave, es decir: [key]. A continuación se muestran algunos ejemplos de sintaxis.

```
$a = array (“clave” => “valor”, 12 => true);
      echo $a[“clave”]; //valor
      echo $a[12];      // 1, TRUE se evalúa como entero (1) y FALSE
como 0
$b = array (“arrayinterno” => array (6 => 5, 13 => 9, “a” => 42);
      echo $b[“arrayInterno”][6];         //5
      echo $b[“arrayInterno”][“a”]        //42
$c = array (5 => 43, 32, 56, “b” => 12);
      echo $c[6];       //32
```

También podemos crear y modificar arrays mediante la sintaxis de corchetes, para entender el funcionamiento de esta sintaxis hay que tener en cuenta lo siguiente:

- Si un array no existe, se crea.
- Si existe, se añaden los nuevos elementos al final del array existente.

```
$d = array(5 => 1, 12 => 2);
$d[] = 56;
echo $d[13]; //56
```

Existen otras funciones interesantes que podemos utilizar cuando trabajamos con arrays, estas son:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- print_r(): Muestra el contenido del array.
- unset(): Elimina un elemento concreto del array que se le pasa como parámetro o el array completo sino le pasamos un elemento específico.
- array_values(): Reordena las claves de un array empezando por 0 y terminado por n-1, donde n es el número de elementos del array.

3.2.3.2.2.2 Objetos

Para crear objetos de PHP, se instancia su clase correspondiente, tal y como cabría esperar:

```
<?php
	class Automovil
	{
		//Atributos
		private $fabricante;
		public  $color;
		protected $nPlazas;

		//Métodos
		function arrancar()
		{
			echo "Arrancando.";
		}
		function frenar()
		{
			echo "Frenando.";
		}
		function acelerar()
		{
			echo "Acelerando.";
		}

}
	$astra = new Automovil; //Instancio el objeto astra de la
	clase automovil
	$astra->color = “Negro” //Asigno el color negro al objeto
	astra
	$astra->arrancar();    //Llamo al método arrancar del ob-
	jeto astra
?>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Como se puede observar en el ejemplo anterior, las clases se especifican utilizando la palabra reservada class, los métodos deben llevar delante la visibilidad (public, private y protected) y para los métodos se utiliza la palabra reservada function.
Para instanciar un objeto de una determinada clase se utiliza la palabra new y para acceder a los atributos o llamar a los métodos se utiliza el operador ->.
Veremos de forma más extensa la implementación que PHP hace de este paradigma en el punto correspondiente de esta documentación.

#### 3.2.3.2.3 Tipos especiales

Existen dos tipos de tipos de datos especiales:

- El tipo de dato resource, utilizado para hacer referencia a tipos externos, tales como descriptores de ficheros, conexiones a bases de datos, imágenes, etc.
- El tipo de dato NULL, que representa el valor nulo o vacío, una variable se considera que es NULL cuando se le ha asignado NULL explícitamente, cuando todavía no tiene ningún valor o cuando se le ha aplicado la función unset().

#### 3.2.3.2.4 Conversiones

Como ya comentamos al principio de este apartado, no es necesario determinar el tipo de la variable en la definición, es el propio PHP el que determina el tipo de la variable por el contexto en tiempo de ejecución. Es decir, si asigno la cadena de caracteres "Universidad de Salamanca" a la variable $Universidad, esta se convertirá automática en una variable de tipo string, pero si posteriormente a esta misma variable $Universidad le asigno el valor 10 se convertirá en un entero. A este sistema utilizado por PHP se le denomina tipado dinámico, veamos un ejemplo para comprenderlo mejor.

```
<?php
      $a = "0";         // $a es string (ASCII 48)
      $a += 2;          // $a es ahora un integer (2)
      $a = $a + 1.3;          // $a es ahora un float (3.3)
      $a = 5 + "10 cadena";   // $a es ahora un integer (15)
?>
```

Hay ocasiones en las que puede ser necesario conocer el tipo de datos que se almacena en una determinada variable. Para ello, se pueden utilizar las siguientes funciones:

- is_int (var), devuelve TRUE si var es de tipo entero.
- is float(var), devuelve TRUE si var es de tipo float.
- is_string(var), devuelve TRUE si var es de tipo string.
- is_array(var), devuelve TRUE si var es de tipo array.
- is_object (var), devuelve TRUE si var es de tipo object

Si queremos realizar un casting, es decir, transformar el contenido de la variable de un tipo determinado a otro tipo especificado por nosotros se utiliza el tipo deseado encerrado entre paréntesis, los diferentes casting entre variables posibles son los siguientes:

- (int), (integer) – Casting a integer.
- (bool), (boolean) – Casting a boolean

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- (float), (double), (real) – Casting a float
- (string) – Casting a string
- (array) – Casting a array
- (object) – Casting a object
- (unset) – Casting a NULL.

#### 3.2.3.3 Variables y constantes

##### 3.2.3.3.1 Variables

Como ya hemos visto en los ejemplos anteriores, las variables en PHP se representan por el signo dólar ($), seguido del nombre de la variable, teniendo en cuenta que PHP distingue entre mayúsculas y minúsculas. El nombre de las variables puede empezar por una letra (mayúscula o minúscula) o por una barra baja (_), seguida de números o letras o barras bajas.

```
<?php
      $a = “Alice”;
      $B = “Bob”;
      echo "$a, $B"; // salida "Alice, Bob"
?>
```

El nombre $this es un hombre de variable especial y está reservado.
PHP permite el uso de variables de variables, para entender este concepto veamos un ejemplo que lo ilustre:

```
$coche = “OPEL”;  //Asignamos la cadena “OPEL” a $coche
$$coche = ”astra”;      //Asignamos la cadena “astra” a $$coche, es decir $OPEL
echo $OPEL;       //Resultado: astra
```

PHP también provee de ciertas variables globales predefinidas que se denominan Superglobales, se trata de variables que son globales de forma automáticamente. Algunas de las más importantes son las siguientes:

- $_SERVER, permite que los scripts accedan a un conjunto muy extenso de información que proporciona el servidor Web sobre el que se está ejecutando el script.
- $_GET, se utiliza para recoger variables suministradas mediante el método de invocación GET.
- $_POST, se utiliza para recoger variables suministradas mediante método de invocación POST.

  Si tenemos un formulario que tiene un campo cuyo atributo name es “nombreUsuario”, podríamos acceder a él mediante las variables $_GET y $_POST, en función de si este formulario lo enviamos mediante el método de invocación GET o POST, respectivamente.

  ```
  $_POST[“nombreUsuario”];
  $_GET[“nombreUsuario”];
  ```

- $_COOKIE, se utiliza para manejar cookies procedentes del navegador del cliente. Cuando desde un *script* se quiera establecer el valor de una nueva *cookie* o cambiar el de


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

una existente, se deberá hacer uso de la función de PHP *setcookie*, que se comenta brevemente a continuación:

```
setcookie(nombre, valor, caducidad, camino, dominio, segura)
```

- $_REQUEST, se utiliza para acceder a las variables de la petición HTTP. Contiene la concatenación de la información presente en las variables $_GET, $_POST y $_COOKIE.
- $_FILES, se utiliza para manejar la subida de ficheros.
- $_SESSION, se utiliza para manejar sesiones, es decir, se dispone de un mecanismo denominado "sesión" que permite la gestión de variables que mantiene su valor durante toda la visita o secuencia de accesos de un cliente a un servidor. Mediante esta técnica, una secuencia de accesos por parte de un cliente a un *script* PHP (o a un conjunto de ellos) podrá ser tratada como un todo. Podremos manejar variables para cada nueva sesión abierta por los clientes de nuestra aplicación Web. Para utilizar las variables de sesión, previamente es necesario inicializar sesión mediante session_start(), de la misma manera al terminar de trabajar con sesiones es necesario destruir la sesión mediante session_destroy();

#### 3.2.3.3.1.1 Ámbito de las variables

Una variable *local* es aquella que se define dentro de una función. Su ámbito se restringe a la función en la que se define y, por lo tanto, no es visible desde fuera de la misma. Una variable *global* es la que se define fuera de las funciones y, por lo tanto, puede ser accedida desde cualquier función, pero para ello éstas deben estar declaradas en la función utilizando la palabra reservada *global*. Otra forma de acceder a variables globales se realiza mediante el *array* $GLOBALS que almacena todas las variables globales que hayan sido definidas a lo largo del código PHP.

### 3.2.3.3.2 Constantes

Las constantes se pueden definir como un identificador que se le asigna a un valor, que no cambia durante la ejecución del script. El nombre de las constantes tiene el mismo formato que el de las variables, pero por convención siempre están en mayúsculas, en este caso el nombre no va precedido por el signo dólar ($). Se pueden definir constantes con dos tipos diferentes de sintaxis.

```
define ("PI", "3.1416"); //Mediante la funcion  define
const PI = "3.1416";   //Mediante la palabra cons
```

Existen un conjunto de constantes predefinidas llamadas "constantes mágicas" (*magic constants*). El nombre de estas variables esta sufijado y prefijado por dos caracteres de subrayado (__). El valor de estas constantes depende del contexto en el que se usen. Algunos ejemplos de este tipo de variables se presentan a continuación:

| **Nombre** | **Descripción** |
|---|---|
| **__LINE__** | Número de la línea actual. |
| **__FILE__** | Ruta completa y nombre del fichero actual. |
| **__DIR__** | Ruta completa del directorio en el que se encuentra el fichero actual. |
| **__FUNCTION__** | Nombre de la función actual. |
| **__CLASS__** | Nombre de la clase actual. |
| **__METHOD__** | Nombre del método actual. |

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

#### 3.2.3.4 Operadores

Los operadores se utilizan para transformar uno o más valores en otros nuevos. Existen tres tipos de operadores:

- Unarios. Que actúan sobre una única variable.
- Binarios. Que actúan sobre dos variables.
- Ternarios. Que es el operador "¿ : .", que se utiliza para seleccionar una expresión entre dos expresiones en función de una condición.

```
$a == true ? 1 . 2;
```

##### 3.2.3.4.1 Operadores binarios

Los operadores binarios se utilizan para realizar las operaciones aritméticas básicas, todos ellos los tenemos en la siguiente tabla:

| **Nombre** | **Definición** | **Ejemplo** |
|---|---|---|
| Negación | Número negativo de $a | `$c = -$a;` |
| Suma | Suma de $a y $b | `$c = $a + $b;` |
| Resta | Resta de $a y $b | `$c = $a - $b;` |
| Multiplicación | Multiplicación de $a y $b | `$c = $a * $b;` |
| División | División de $a entre $b | `$c = $a / $b;` |
| Módulo | Módulo de la división entre $a y $b | `$c = $a % $b` |

Al utilizar estos operadores, hay que tener en cuenta la preferencia de operaciones aritméticas y la existencia de paréntesis en las operaciones para evaluar correctamente el orden en que se ejecutarán cada una de las operaciones de le expresión.
PHP también soporta los operadores de incremento y decremento, para sumar una unidad al valor inicial de la variable, mediante el operador ++, o para restarle esa unidad mediante el operador --.
La característica particular es que el incremento se puede realizar de forma previa (Pre-incremento/Pre-decremento) a utilizar el valor, o de forma posterior (Post-incremento/Post-Decremento), para ello habrá que colocar los operadores ++ o –- delante o detrás de la variable, en función de lo que se requiera en el programa., veamos un ejemplo.

```
$a = 1;
echo $a++;  // Devolverá 1
echo $a           // Devolverá 2
```

```
$a = 1
echo ++$a;  //Devolverá 2
      echo $a           //De-
volverá 2
```

##### 3.2.3.4.2 Operador de asignación

El operador de asignación es el signo igual (=), este operador, no se encarga de comparar si dos elementos son iguales, sino que se encarga de asignar a una variable el contenido de otra variable o de una expresión, es decir:

```
$a = 4;     //Se asigna el valor 4 a la variable $a
$a = $b + $c      //Se asigna a la variable $a el resultado de $b + $c
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

El operador asignación se puede combinar con los operadores binarios aritméticos para de la siguiente manera:

```
$a = $a + 2;      //Esta expresión es igual que la siguiente
      $a += 2     //En ambos casos el valor de la variable $a se incre-
                  mentara en 2 //unidades
```

3.2.3.4.3 Operadores de comparación

Este tipo de operadores, se utilizan para comparar dos valores. Los operadores utilizados se pueden apreciar en la siguiente tabla:

| **Nombre** | **Operador** | **Ejemplo** |
|---|---|---|
| Igual | TRUE si $a es igual a $b | `$a == $b` |
| Idéntico | TRUE si $a es igual a $b y tienen el mismo tipo | `$a === $b` |
| No igual (Distinto) | TRUE si $a es distinto a $b | `$a != $b` |
| No idéntico | TRUE si $a es distinto a $b y no tienen el mismo tipo | `$a <> $b` |
| Menor que | TRUE si $a es menor que $b | `$a < $b` |
| Mayor que | TRUE si $a es mayor que $b | `$a > $b` |
| Menor o igual que | TRUE si $a es menor o igual que $b | `$a <= $b` |
| Mayor o igual que | TRUE si $a es mayor o igual que $b | `$a >= $b` |

Existen varias particularidades a tener en cuenta:

- Si comparamos un tipo integer con un tipo string, el segundo se convierte automáticamente a integer.
- Si comparamos NULL con string, NULL se convierte automática a una cadena vacía.
- Si comparamos dos array, sólo es posible la comparación cuando las claves del primero también sean las claves del segundo, sino son incomparables.

3.2.3.4.4 Operadores lógicos

Los operadores lógicos que proporciona PHP se muestra en la siguiente tabla:

| **Nombre** | **Operador** | **Ejemplo** |
|---|---|---|
| And - && | TRUE si $a y $b son TRUE | `$a and $b`<br>`$a && $b` |
| Or - \|\| | TRUE si $a o $b son TRUE | `$a or $b`<br>`$a \|\| $b` |
| Not | TRUE si $a no es TRUE | `$a != $b` |

PHP distingue entre los operadores -and y &&- y -or y ||- debido a que tanto && y || tienen mayor prioridad que and y or.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

#### 3.2.3.4.5 Operador de cadenas

El único operador para cadenas es el operador concatenación que es el operador punto (.), este operador devuelve la concatenación de dos cadenas. También puede ser utilizado en combinación con el operador de asignación para agregar una cadena al final de la primera.

```
$a = “Experto” . “DSCE”     //$a es igual a ExpertoDSCE
$a += 2009/10          //$a es igual a ExpertoDSCE2009/10
```

#### 3.2.3.5 Estructuras de control

Cualquier programa esta compuesto por una conjunto de instrucciones o sentencias, cada una de estas sentencias se ejecutan de forma secuencial, independientes de que estas secuencias sean asignaciones, operaciones, etc.
Existe un tipo de sentencias que se utilizan para alterar este orden básico de la ejecución normal de un programa, estas son las sentencias de control. Dentro de ellas podemos distinguir dos tipos básicos, las sentencias condicionales que permiten la ejecución (o no) de bloques de código en función de una condición y las sentencias de repetición o bucles que permiten la ejecución de un bloque de código durante un número determinado de veces.

##### 3.2.3.5.1 Sentencias condicionales

Dentro de las sentencias condicionales podemos distinguir dos tipos, la instrucción if y la instrucción switch.

- La sentencia if permite ejecutar un bloque de código si y sólo si se cumple una determinada condición.

  Junto a la sentencia if, se puede incluir la sentencia else, que hará que se ejecute otro bloque de código sino se cumple la condición de la sentencia if.
  De la misma manera, también existe la sentencia elseif, la cual ejecuta su bloque de código correspondiente cuando no se cumple la sentencia if y además se cumple su propia condición.

```
<?php
      if ($a == $b)
      {
            echo “$a y $b son iguales”;
      }
      elseif ($a > $b)
      {
            echo “$a es mayor que $b”;
      }
      else
      {
            echo “$a es menor que $b”;
      }
?>
```

- La sentencia switch esta formada por una variable de entrada y una serie de bloques de código, dependiendo del valor de la variable de entrada se ejecutará un determinado bloque de código.

  ```
  <?php
  ```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
        switch ($rueda)
        {
              case “delanteraIzquierda”:
              case “delateraDerecha”:
                    echo “Eje delantero del coche”;
                    break;
        case “traseraIzquierda”:
        case “traseraDerecha:
              echo “Eje trasero del coche”;
              break;
        default:    //Cualquier otra rueda
              echo “Rueda de repuesto”;
        }
  ?>
```

3.2.3.5.2 Bucles

Los bucles permiten ejecutar un bloque de código un número (en principio) determinado de veces, existen los siguientes bucles:

- Bucle while, el bucle while (mientras) ejecuta un bloque de código mientras se cumpla una condición que se comprueba de forma previa a la ejecución del bloque.

```
<?php
      $i = 0;
      while ($i <= 5)
      {
            echo $i;
            $i++;
      }
?>
```

- Bucle do/while, es muy similar al bucle anterior, con la excepción que la condición se comprueba después de cada ejecución del código correspondiente.

```
<?php
      $i = 0;
      do
      {
            echo $i;
            $i++;
      }
      while ($i <= 5);
?>
```

- Bucle for, este tipo de bucles es más completo que los anteriores, ya que además de una sentencia condicional contiene una inicialización y una actualización.

```
<?php
      for ($i = 0; $i <= 5; $i++)
      {
            echo $i;
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
        }
?>
```

- Bucle foreach, permite recorrer un array elemento a elemento, cada elemento se almacena en una variable que será accesible únicamente dentro del bucle.

```
<?php
        $numeros = array(“uno”, “dos”, “tres”);

        foreach ($numero as $numeros)
        {
                echo $numero;
        }

        foreach (list ($clave, $valor) as $numeros)
        {
                echo $clave . “: “ . $valor;
        }
?>
```

Dentro de los bucles también es necesario que tengamos en cuenta las sentencias break y continue:

- La sentencia break, hace que se detenga la ejecución del bucle, aunque se siga cumpliendo la condición de salida.
- La sentencia continue por su parte hace que se detenga la iteración actual del bucle y se continúe con la siguiente o se pase al siguiente elemento en el caso del bucle foreach.

#### 3.2.3.5.3 Inclusión de código desde archivos externos

PHP permite ejecutar código PHP existente en otros archivos. Existen dos sentencias que actúan como funciones y que permiten realizar esta inclusión: *require*() e *include*().
Tanto la función require() como include() incluyen y evalúan el fichero indicado como parámetro; permite insertar y ejecutar un archivo en el programa que se esté cargando. Es adecuado para la inclusión de constantes o funciones que no se van a modificar, además de cualquier código de presentación que se considera repetitivo (encabezados y pies de página, menús, etc.), por ejemplo:

```
require("constantes.php");
include("funciones.php");
```

require() e include() son idénticas, excepto en el comportamiento que presentan ante un error al no encontrar el fichero indiciado: include() produce un Warning y require() produce un Error Fatal.
Es preciso destacar que se debe tener cuidado para no incluir dos o más veces un mismo fichero si en él se definen funciones, ya que se presentaría un error de redefinición de una misma función. Para evitar este problema están las sentencias include_once() y require_once(), que sólo incluyen el fichero una vez durante la evaluación de todo el script.

```
require_once("constantes.php");
incluye_once("funciones.php");
```

#### 3.2.3.6 Funciones

Una función se puede definir con una secuencia de instrucciones que permiten procesar un conjunto de variables para obtener un resultado. Las funciones se utilizan cuando un mismo bloque


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

de código va a ser usado en diferentes momentos de la ejecución del script, este bloque se agrupa en una función.
La definición de una función en PHP tiene siempre la misma estructura, la firma de la función esta compuesta por la palabra reservada function, el nombre de la función y un conjunto de variables que se le suministran como parámetros, están separados por comas y entre paréntesis. El siguiente elemento será el bloque de código que se desea agrupar y por último la función también puede contener aunque no es obligatorio un valor de retorno, este valor no es necesario que este al final de la función, será parte de la secuencia de instrucciones.

```
<?php
	function miFuncion($arg_1, $arg_2, … $arg_n)
	{
		[Bloque de código PHP]

		return $valor_retorno;
	}
?>
```

Para utilizar una función es necesario declararla previamente y llamarla mediante su identificador de función, agregando los parámetros que sean necesarios tal y como indique su firma. Veamos un ejemplo de una función que se encarga de sumar dos números:

```
<?php
	//Declaro la función
	function sumaNumeros ($a, $b)
	{
		$c = $a + $b;

		return $c;
	}

	//Realizo la llamada a la función
	$valorDevuelto = sumaNumeros (4, 5);
	echo $valorDevuelto;    //El resultado será 9
?>
```

Existen dos formas distintas de realizar el paso de parámetros a una función:

- **Paso de parámetros por valor.** Éste es el método por defecto utilizado en PHP. Al parámetro de la función se le asigna una copia del valor con el que es invocado. La función no puede modificar el valor de la variable que se le suministra, los ejemplos anteriores utilizan paso de parámetros por valor.
- **Paso de parámetros por referencia.** El parámetro contiene una referencia a la variable con la que fue invocado. La función puede modificar el valor de la variable y los cambios que se produzcan serán visibles fuera de la función. Para indicar en PHP el paso de una variable por referencia se utiliza el carácter & en la declaración que se haga de la misma en la función, veamos un ejemplo

  ```
  <?php
  ```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
            function aNadeCadena(&$cadena)
            {
                  $string .= “ , a continuación.”;
            }

            $str = “Primeramente”;

            aNadeCadena($str);

            echo $str;  // Resultado: “Primeramente , acontinuación”
?>
```

#### 3.2.3.7 Objetos y clases

El paradigma orientado a objetos permite encapsular la información junto con el comportamiento asociado a dicha información, es decir, se define clases que describen de forma genérica el estado y el comportamiento de un conjunto de objetos que las instancian. Una clase esta compuesta por propiedades y por métodos:

- Las propiedades son las variables que definen el estado de los objetos.
- Los métodos definen el comportamiento de los objetos.

Por ejemplo, si tenemos la clase Automóvil, esta podría tener los atributos fabricante, color, número de plazas, etc. y los métodos arrancar, frenar, acelerar, etc. Los diferentes objetos se crearían a partir de esta clase automóvil y cada uno tendría sus propios valores para los atributos.
El modelo de objetos de PHP fue totalmente reimplementado a partir de la versión 5 para incluir nuevas características y mejorar las existentes. A lo largo de los apartados siguientes, estudiaremos algunas de las características más importantes.

##### 3.2.3.7.1 Conceptos básicos

Una clase se define mediante la palabra class, seguida del nombre. Una clase contiene sus propias constantes, variables (propiedades) y funciones (métodos).

```
<?php
      class PrimeraClase
            {
                  //Propiedaes
            public $var = “valor por defecto”;

            //Métodos
            public function metodo()
                  {
                  echo “Soy un metodo muy simple”;
            }
            public function getVar()
                  {
                  return $this->var;
            }
      }
?>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

La variable $this, o mejor dicho, la pseudo-variable $this hace referencia al propio objeto, permitiendo acceder a las propiedades y a los métodos del mismo.
Para crear una instancia de una clase se debe crear nuevo objeto y asignarlo a una variable. Para realizar este proceso se utiliza la palabra new seguido del nombre de la clase que se desea instanciar. Para el acceso a las diferentes propiedades y métodos de un objeto se utiliza la notación guión-flecha (->).

```
<?php
      $miObjeto = new PrimeraClase();
      $miObjeto->getVar();
¿>
```

El modelo de objetos de PHP también admite la herencia entre clases mediante la palabra reservada extends. Cuando una clase (hija o subclase) hereda de otra clase (padre o superclase), los métodos y propiedades de la clase padre también están disponibles para la clase hija. PHP únicamente admite herencia simple, es decir, cada clase puede heredar únicamente de una única clase.
Los métodos de la clase padre, pueden ser modificados (sobrescritos) en la clase hija, para adaptarlos al contexto en los que se utilizará la subclase. Para evitar que un método pueda ser sobrescrito se utiliza la palabra reservada final. Por otro lado, para acceder a los métodos de la clase padre, desde la clase hija se utiliza parent::

```
<?php
      class SubClase extends MiClase
            {
            //Métodos
            public function getVar()
                  {
                  return “Hijo: ” . parent::getVar();
            }
      }
?>
```

#### 3.2.3.7.2 Propiedades y constantes

Las clases tienen un tipo especial de variables denominadas propiedades o atributos, las propiedades se definen acompañadas de un operador de visibilidad (ver apartado siguiente) y puede asignárseles un valor inicial en la declaración de las mismas, aunque también se le puede asignar un valor por defecto en el constructor (ver apartados siguientes).
Las variables también se las pueden definir como estática, mediante la palabra static, este tipo se denomina “propiedades de clase”.
Mientras que a las variables se accede mediante el operador ->, a las propiedades de clase se accede mediante la notación ::
De la misma forma que se permite la definición de propiedades variables, también se pueden definir propiedades constantes, estás nunca cambian de valor y no comienzan por $. Para el acceso a estas propiedades se utiliza el operador ::

#### 3.2.3.7.3 Visibilidad

La visibilidad de una propiedad o de un método, es decir, el contexto desde los que son accesible se puede definir mediante los siguientes prefijos en la declaración:

- public: Se puede acceder desde cualquier contesto.
- protected: Se puede acceder por desde la clase y la subclase.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- private: Los miembros de este tipo sólo puede ser accedidos desde la clase que los declara.

#### 3.2.3.7.4 Constructores y destructores

Un constructor es un método que se invoca justo en el momento en el que se instancia un objeto de una clase y normalmente se utiliza para precargar el estado del objeto. Mientras que, un destructor se invoca cuando se eliminan las referencias a un objeto, se elimina explícitamente o el mismo finaliza su ejecución y se suele utiliza para liberar recursos que pudiera estar utilizando el objeto.
Existen dos métodos especiales en las clases de PHP5. Su signatura es la siguiente:

- void __construct ($arg_1, $arg_2, … , $arg_n)
- void __destruct ( void )

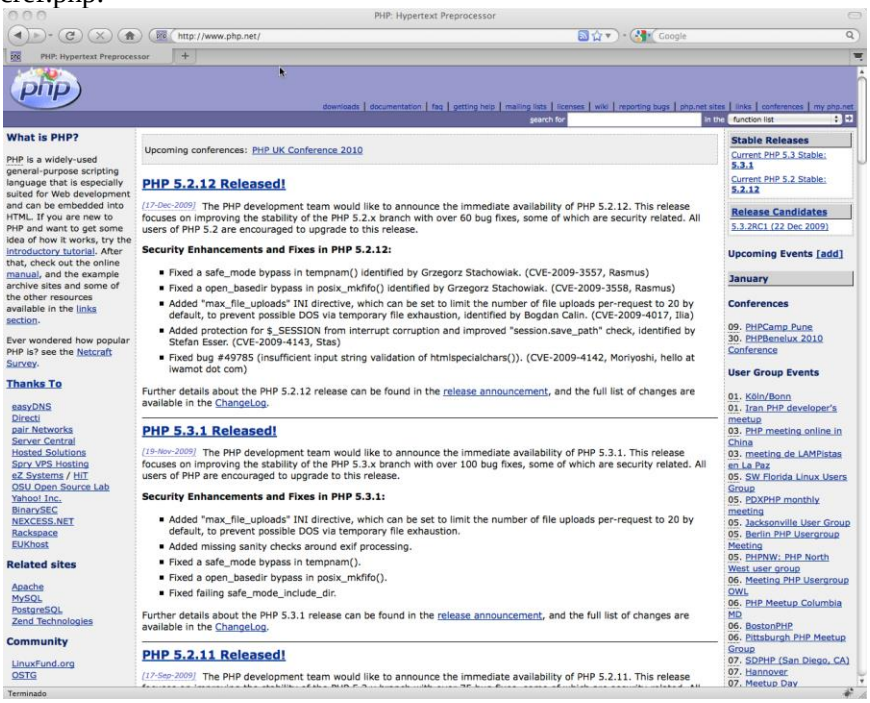
### 3.2.4 Referencias a funciones

PHP provee a los programadores de un elevado número de librerías, bibliotecas o extensiones del lenguaje, éstas se pueden englobar en las siguientes categorías:

- Funciones que afectan al comportamiento de PHP.
- Funciones para la manipulación de formatos de audio.
- Funciones para el manejo de servicios de autentificación.
- Funciones para el manejo de fechas.
- Funciones específicas para el uso de PHP por líneas de ordenes.
- Funciones para la compresión y encapsulación de archivos.
- Funciones para el procesado de pago mediante tarjetas de crédito.
- Funciones criptográficas.
- Funciones para el manejo de bases de datos.
- Funciones para el manejo de archivos.
- Funciones para el manejo de lenguaje humano y codificación de caracteres.
- Funciones para la generación y procesamiento de imágenes
- Función para el manejo de servicios de correo electrónico.
- Funciones matemáticas.
- Funciones MIME
- Funciones para el control de procesos.
- Funciones para manejar motores de búsqueda.
- Funciones especificas de servidores (Apache, IIS, etc.)
- Funciones para el procesamiento de texto.
- Funciones para el manejo de tipos y variables.
- Funciones para servicios Web.
- Funciones específicas para la plataforma Windows
- Funciones para la manipulación de XML.
- Otras funciones.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

En este manual no vamos a adentrarnos en la explicación de la gran cantidad de funciones que proporciona PHP, pero no obstante, todas ellas pueden ser consultas en la página oficial de PHP, http://www.php.net (utilizando el buscador) o directamente en el enlace: http://www.php.net/manual/en/funcref.php.




### 3.2.5 Frameworks

PHP ha evolucionado mucho en los últimos años, ha pasado de ser un lenguaje completamente caótico donde se podía hacer cualquier cosa mezclando mucho código distinto, a ser un lenguaje con pleno soporte para objetos. Además gracias al nacimiento de PHP5 y sobretodo al influjo de *Ruby on Rails* (http://rubyonrails.org/) empezaron a surgir los Framework en PHP.

Un Framework conceptualmente es una estructura previamente definida para resolver de una forma elegante y rápida problemas que a priori son complejos. Proporciona una serie de funcionalidades previamente implementadas que liberan al diseñador de tareas de bajo nivel y así poder dedicarle un mayor tiempo al diseño correcto de los aspectos propios del sistema concreto. En el sentido de reusabilidad se puede decir que son como librerías de software, pero los Framework van un paso más adelante, ya que habitualmente son componentes estructurales de la herramienta que se desarrolla.

Se obtiene diferentes ventajas de la utilización de un Framework:

- Desarrollo rápido de aplicaciones.
- Reutilización de componentes software.
- El uso y la programación de componentes software que siguen una política de diseño uniforme.

Hoy en día realizar cualquier proyecto grande sin un Framework es un suicidio, ya que además de que reducen de forma considerable los tiempos de desarrollo permiten tener un gran control sobre la aplicación y mejorar la seguridad de la misma.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

PHP dispone de una gran cantidad de Frameworks disponibles cada uno con sus características particulares. A continuación, se presentan algunos de los más importantes junto a sus características generales.

| | PHP4 | PHP5 | MVC | ORM | Template | Validation | Ajax | Modules |
|---|---|---|---|---|---|---|---|---|
| **Codelgniter** | X | X | X | | X | X | | |
| **CakePHP** | X | X | X | X | | X | X | X |
| **Yii** | | X | X | X | X | X | X | X |
| **Zend** | | X | X | X | | X | X | X |
| **Symfony** | | X | X | X | | X | X | X |
| **PHP-DevShell** | | X | | | X | | X | X |
| **Prado** | | X | X | X | X | X | X | X |
| **Akelos** | X | X | X | X | X | X | X | X |
| **ZooP** | X | X | X | | X | X | X | |
| **QPHP** | X | X | X | | X | X | X | X |

Una explicación de las características de cada Framework se muestra a continuación:

- PHP4 y PHP5, indica si el Framework puede ser utilizado con PHP4 o PHP5, respectivamente.
- MVC, indica si el Framework tiene soporte para el patrón Modelo-Vista-Controlador.
- ORM, indica si el Framework realiza mapeo objeto-relacional
- Template, indica si el Framework permite el uso de plantillas.
- Validation, indica si el Framework implementa algún compente de validación de campos de formulario.
- Ajax, indica si el Framework tiene soporte para el lenguaje Ajax.
- Modules, indica si el Framework proporciona módulos preimplementados (RSS, PDF, etc.)

### 3.2.6 Ejemplos

#### 3.2.6.1 Conversiones en PHP

```
<html>
     <body>
          <?php
               $cadena = "1234";
               $numero = intval($cadena);
               echo "El valor numérico es " . $numero . "<BR>";

               $cadena = "1010";
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
                // Se convierte de binario a decimal
                $numero = intval($cadena, 2);
                echo "El valor numérico es " . $numero . "<BR>";

                $cadena = "97.76";
                $real = doubleval($cadena);
                echo "El valor numérico es " . $real . "<BR>";

                $numero = 1234;
                $cadena = strval($numero);
                echo "El valor de cadena es " . $cadena . "<BR>";
          ?>
     </body>
</html>
```

**Figura 23 - Bloque de código: Conversión de valores**

### 3.2.6.2 Ejemplos con variables

```
<html>
     <body>
          <?php
                function asignarUno ()
                {
                     global $numero; // $numero es una variable global
                     $numero = 1;
                }
                function asignarDos ()
                {
                     $GLOBALS["numero"] = 2; // $numero es variable global

                }

                $numero = 7;
                echo "El valor de numero es " .$numero. "<BR>";
                asignarUno();
                echo "El valor de numero es " .$numero. "<BR>";
                asignarDos();
                echo "El valor de numero es " .$numero. "<BR>";
          ?>
     </body>
</html>
```

**Figura 24 - Bloque de código: Ámbito de variables**

```
<html>
     <body>
          <?php
                function contar ()
                {
                     static $veces = 0;

                     $veces++;
                     echo "Se ha ejecutado contar " .$veces. " veces <BR>";
                }

                for ($i = 0; $i < 80; $i++)
                     contar();
          ?>
     </body>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
</html>
```

**Figura 25 - Bloque de código: Elemento static**

### 3.2.6.3 Estructuras de control

```
<html>
     <head>
          <title>Ejemplo de if con PHP</title>
     </head>
     <body>
          <?php
               $a = 8;
               $b = 3;
               if ($a < $b)
               {
                    echo "a es menor que b";
               }
               else
               {
                    echo "a no es menor que b";
               }
          ?>
     </body>
</html>
```

**Figura 26 - Bloque de código: Sentencia if-else**

```
<html>
     <head>
          <title>Ejemplo de switch con PHP</title>
     </head>
     <body>
          <?php
               $posicion = "arriba";
               switch($posicion)
               {
                    case "arriba":
                         echo "La variable contiene";
                         echo " el valor arriba";
                         break;
                    case "abajo":
                         echo "La variable contiene";
                         echo " el valor abajo";
                         break;
                    default:
                         echo "La variable contiene otro valor";
                         echo " distinto de arriba y abajo";
               }
          ?>
     </body>
</html>
```

**Figura 27 - Bloque de código: Sentencia switch I**

```
<?php
switch ($i):
    case 0:
        echo "i igual a 0";
        break;
    case 1:
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
        echo "i igual a 1";
        break;
    case 2:
        echo "i igual 2";
        break;
    default:
        echo "i no es igual a 0, 1 o 2";
endswitch;
?>
```

**Figura 28 - Bloque de código: Setencia switch II**

```
<html>
     <body>
          <?php
               $alumno["Nombre"] = "Fernando";
               $alumno["Apellidos"] = "Díaz Gómez";
               $alumno["Edad"] = 32;
               foreach ($alumno as $elemento)
               {
                    echo " $elemento";
                    echo “<BR>”;
               }
          ?>
     </body>
</html>
```

**Figura 29 - Bloque de código: Bucle foreach**

```
<html>
     <body>
          <?php
               $alumno["Nombre"] = "Fernando";
               $alumno["Apellidos"] = "Díaz Gómez";
               $alumno["Edad"] = 32;
               while ($elemento = each($alumno))
               {
                    echo " $elemento[0]: $elemento[1]";
                    echo "<BR>";
               }
          ?>
     </body>
</html>
```

**Figura 30 - Bloque de código: Sentencia each**

### 3.2.6.4 Funciones

```
<html>
     <body>
          <?php
               function intercambiar (&$var1, &$var2) {
                    $aux = $var1;
                    $var1 = $var2;
                    $var2 = $aux;
               }
               $n1 = 7;
               $n2 = 16;

               echo "Antes de la llamada a la función: " .$n1. ", " .$n2. "<BR>";
               intercambiar($n1, $n2);
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
                echo “Después de la llamada a la función: " .$n1. ", " .$n2. "<BR>";
        ?>
    </body>
```

**Figura 31 - Bloque de código: Paso de parámetros a funciones I**

```
<?php
function hacerCafe($tipo = "cappuccino")
{    return "Haciendo una copa de $tipo.\n";
}
echo hacerCafe();            // Haciendo una copa de cappuccino.
echo hacerCafe(null);        // Haciendo una copa de .
echo hacerCafe("espresso");  // Haciendo una copa de espresso.
?>
```

**Figura 32 - Bloque de código: Paso de parámetros a funciones II**

### 3.2.6.5 Manejo de cadenas

```
<html>
    <head>
         <title>Ejemplo de strings con PHP</title>
    </head>
    <body>
      <?php
             $suma = 2 + 3;
             echo "La suma de 2 y 3 es: {$suma}";       // La suma de 2 y 3 es: 5
             echo 'La suma de 2 y 3 es: {$suma}';       // La suma de 2 y 3 es:
{$suma}
       ?>
    </body>
</html>
```

**Figura 33 - Bloque de código: Ejemplo con cadenas I**

```
$vector[40] = 77;
$vector[10] = 45.67;
$vector[1] = "miércoles";

print(count($vector)); // imprime 3

/*Sin embargo, lo anterior no impide crear un array similar a uno clásico, que además
se puede recorrer de una forma cásica también:*/

$dias[0] = “lunes”;
$dias[1] = “martes”;
$dias[2] = “miércoles”;
$dias[3] = “jueves”;
$dias[4] = “viernes”;
$dias[5] = “sabado”;
$dias[6] = “domingo”;

for ($i=1; $i<7; $i++){
       echo “Dia $i es: ”.$dias[$i].”<br>”;
}
```

**Figura 34 - Bloque de código: Ejemplo con cadenas II**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

#### 3.2.6.6 Clases y objetos

```
<?php
class MiClase {
    // declaración de una constante
    const PI = 3.14;

    // declaración de una propiedad
    public $var = 'propiedad';

    // declaración de un método
    public function muestraVar() {
        echo $this->var;
    }
}
?>
       $instancia = new MiClase();
```

**Figura 35 - Bloque de código: Instanciación de objetos**

```
<?php
       class SubClase extends MiClase {
           // Redefine un método de la clase padre.
           function muestraVar() {
               echo "Extendiendo una clase\n";
               parent::muestraVar();
           }
       }

       $extended = new SubClase();
       $extended->muestraVar();
?>
```

**Figura 36 - Bloque de código: Herencia**

```
<?php
       class Propiedades {
              public $var = 'Hola';
              public static $varStatic = 'Adios';

              public getVar() {
                     return $this->var;
              }

              public getVarStatic() {
                     return self::$varStatic;
              }
       }

       $objeto = new Propiedades();
       // Ambos mostrarían el contenido de var
       echo $objeto->var;
       echo $objeto->getVar();

       // Ambos mostrarían el contenido de varStatic
       echo $objeto->getVarStatic();
       echo $objeto::$varStatic;
?>
```

**Figura 37 - Bloque de código: Atributos de clases**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```php
<?php
class MiClase {
    public $public = 'Public';
    protected $protected = 'Protected';
    private $private = 'Private';

    function imprime() {
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}
$obj = new MiClase();
echo $obj->public; // Funciona
echo $obj->protected; // Error Fatal
echo $obj->private; // Error Fatal
$obj->imprime(); // Funcionan todos los echo

// MiClase2 es una subclase de MiClase
class MiClase2 extends MiClase {
    function imprime() {
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}
$obj2 = new MiClase2();
echo $obj2->public; // Funciona
echo $obj2->private; // No está definido
echo $obj2->protected; // Error Fatal
$obj2->imprime(); // Funcionan los echo Public y Protected. Private no está definido
?>
```

**Figura 38 - Bloque de código: Visiblidad**

```php
<?php

// Declaración de la interfaz.
interface iPlantilla {
    public function setVariable($name, $var);
    public function getVariable($name);
}

// Implementación de la interfaz.
class Plantilla implements iPlantilla {
    private $vars = array();

    public function setVariable($name, $var) {
        $this->vars[$name] = $var;
    }

    public function getVariable($name) {
        return $this->vars[$name];
    }
}
?>
```

**Figura 39 - Bloque de código: Declarando interfaces**

```php
<?php
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
abstract class ClaseAbstracta {
    // Este método debe ser implementado en las subclases.
    abstract protected function getValor();
    // Método normal.
    public function imprimir() {
        print $this->getValor() . "\n";
    }
} class ClaseConcreta extends ClaseAbstracta {
    protected function getValor() {
        return "ClaseConcreta";
    }
}
?>
```

**Figura 40 - Bloque de código: Declarando clases abstractas**

### 3.2.6.7 Manejo de sesiones

```
<?php
     session_start();
?>

<html>
     <head>
          <title>Adivina</title>
     </head>
     <body>
          <?php
               $numero = (int) $_REQUEST['numero'];
               if (empty($secreto) || $_REQUEST['reinicio'] == "Reiniciar") {
                    echo "Intente adivinar el número secreto.<BR><BR>";
                    $_SESSION['secreto'] = rand(0, 9999);
                    $_SESSION['intentos'] = 0;
               } else if (empty($numero)) {
                    echo "Intente adivinar el número secreto.<BR><BR>";
               } else if ($_SESSION['secreto'] < $numero) {
                    $_SESSION['intentos']++;
                    echo "El número secreto es menor que " . $numero . "<BR>";
                    echo "Lleva usted " . $_SESSION['intentos'] . " intentos!<BR>";
               } else if ($_SESSION['secreto'] > $numero) {
                    $_SESSION['intentos']++;
                    echo "El número secreto es mayor que " . $numero . "<BR>";
                    echo "Lleva usted ". $_SESSION['intentos'] . " intentos!<BR>”;
               } else {
                    echo "<BR>";
                    echo "ENHORABUENA, el número secreto era " . $numero . "!<BR>";
                    echo "<BR>";
                    session_destroy();
                    exit();
               }
          ?>
               <FORM METHOD="POST">
               Introduzca un número entero entre 0 y 9999<BR>
               Número: <INPUT NAME="numero" TYPE="text"></TEXTAREA>
               <INPUT NAME="envio" TYPE="submit" VALUE="Probar">
               <INPUT NAME="reinicio" TYPE="submit" VALUE="Reiniciar">
               </FORM>
     </body>
</html>
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

**Figura 41 - Bloque de código: Manejo de sesiones I**

```
<?
session_start(); //Siempre se debe poner por si ya hay sesion

	// Se comprueba si se han introducido los datos de login
	if(isset($_POST['login'])){
		$_SESSION['login'] = $_POST['login'];
		header("Location:login.php"); //limpiar el post redirigiendo

	// Se comprueba si se quiere salir
	}else if(isset($_GET['salir'])){

		//Destruccion de una sesion
		$_SESSION = array();
		session_destroy();
		header("Location:login.php"); //limpiar el get redirigiendo

	}
	// No se ha pulsado ni salir ni entrar
	else{
		?>
		<html>
		<head>MiniLogin</head>
		<body>
		<h1>MiniLogin</h1>
		<!-- Un enlace a otro sitio donde se ve lo que hay en sesion -->
		<a href="usasesion.php">[Ir a otro sitio]</a><br><br>
		<?
		// Se puede estar ya identificados
		if(isset($_SESSION['login'])){
			echo "hola ".$_SESSION['login'];
			?>
			<br>
			<a href="?salir=si">Salir</a>
			<?
		}else{
		// Si no se ha identificado lo solicitamos con formulario
			?>
			<form method="post" acion="login.php">
			Nombre: <input name="login" type="text"><br>
			<input type="submit" value="Entrar">
			</form>
			<?
		}
		?>
		</body>
		</html>
		<?
	}
?>
```

**Figura 42 - Bloque de código: Ejemplo con cadenas II**

```
<?
session_start();
?>
<html>
<head>Usa sesion</head>
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
	<body>
		<h1>Usa sesion</h1>
		<?
		if(isset($_SESSION['login'])){
			echo "Hola ".$_SESSION['login']."<br>";

		}else{
			echo "No te has identificado<br>";
		}
		?>
		<a href="login.php">Volver</a>
	</body>
</html>
```

**Figura 43 - Bloque de código: Ejemplo con cadenas III**

### 3.2.6.8 Variables superglobales $_GET, $_POST, $_COOKIE y $_FILES

```
<html>
     <head>
          <title>Búsqueda</title>
     </head>
     <body>
          <?
               $patron = $_GET['patron'];
               if (empty($patron)) {
          ?>
               <FORM METHOD="GET">
               Introduzca el patrón de búsqueda:<BR>
               <INPUT NAME="patron" TYPE="text">
               <INPUT TYPE="submit" VALUE="Buscar">
               </FORM>
          <?
               } else {
                    // En este punto se realizaría la búsqueda...
                    echo "Resultados de la búsqueda del patrón: $patron<BR>...";
                    // A continuación se presentarían los resultados...
               }
          ?>
     </body>
</html>
```

**Figura 44 - Bloque de código: uso de $_GET**

```
<html>
     <head>
          <title>Opinión</title>
     </head>
     <body>
          <?
               $texto = $_POST['texto'];
               if (empty($texto)) {
          ?>
               <FORM METHOD="POST">
               <H2>Nos interesa mucho su opinión!</H2>
               <TEXTAREA NAME="texto" ROWS="10" COLS="30"></TEXTAREA>
               <INPUT TYPE="submit" VALUE="Enviar">
               </FORM>
          <?
               } else {
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
                    // En este punto $texto se guardaría adecuadamente...
                    echo "<H2>Todas las opiniones y sugerencias son bienveni-
das.</H2>";
                    echo "<H2>Gracias!.</H2>";
                    echo "Usted opina:<HR><PRE>$texto<PRE><HR>";
               }
          ?>
     </body>
</html>
```

**Figura 45 - Bloque de código: uso de $_POST**

```
<?
     $contador = $_COOKIE['contador'];
     $contador++;
     setcookie("contador", $contador, time()+60);
?>

<!-- cookie.php -->
<html>
     <head>
          <title>Contador</title>
     </head>
     <body>
          <H1><?echo $contador; ?></H1>
     </body>
</html>
```

**Figura 46 - Bloque de código: uso de $_COOKIE**

```
<html>
     <head>
          <title>Entrega</title>
     </head>
     <body>
          <?
               $fichero = $_FILES['fichero'];
               if (empty($fichero)) {
          ?>
               <FORM ENCTYPE="multipart/form-data" METHOD="POST">
               <H2>Escoja el archivo que desea enviar:</H2>
               <INPUT NAME="fichero" TYPE="file"><BR>
               <INPUT TYPE="submit" VALUE="Enviar">
               </FORM>
          <?
               } else {
                    echo "Los datos relativos al archivo suministrado son:<HR>";
                    echo "Nombre original " . $fichero['name'] . "<HR>";
                    echo "Tipo de archivo " . $fichero['type'] . "<HR>";
                    echo "Tamaño del fichero " . $fichero['size'] . "<HR>";
                    echo "Nombre temporal " . $fichero['tmp_name'] . "<HR>";
                    if (!empty($fichero['error']))
                         echo "Error ocurrido " . $fichero['error'] . "<HR>";
               }
          ?>
     </body>
</html>
```

**Figura 47 - Bloque de código: uso de $_FILES**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 3.2.6.9 Gestión de bases de datos

```
<?
     $desc = mysql_connect("localhost", "root", "");
     if (!$desc)
          die("No se puede conectar con el servidor");
     mysql_select_db("libros", $desc);
     $resultado = mysql_query("SELECT * FROM libros", $desc);
     while ($obj = mysql_fetch_object($resultado))
          echo "ISBN: $obj->isbn AUTOR: $obj->autor <BR>";
     mysql_free_result($resultado);
     mysql_close($desc);
?>
```

**Figura 48 - Bloque de código: Conexión a una base de datos**

```
<HTML>
     <BODY>
          <FORM METHOD="GET" NAME="formulario" ACTION="conssq.php">
               <B>ISBN  </B><INPUT NAME="FISBN"><BR>
               <B>AUTOR  </B><INPUT NAME="FAUTOR"><BR>
               <B>TITULO  </B><INPUT NAME="FTITULO"><BR>
               <B>EDITORIAL  </B><INPUT NAME="FEDITORIAL"><BR>
               <B>AÑO  </B><INPUT NAME="FANO"><BR>
               <INPUT TYPE="submit" VALUE="Enviar">  
               <INPUT TYPE="reset" VALUE="Borrar">
          </FORM>
     </BODY>
</HTML>
```

**Figura 49 - Bloque de código: Búsqueda en una base de datos (Formulario)**

```
<!-- conssq.php -->
<HTML>
     <BODY BGCOLOR=”#ffffff”>
          <H1>Consulta de libros en una base de datos</H1>
          <?php
               function construye_query($isbn, $autor, $titulo, $editorial, $ano)
               {
                    if ($isbn != "")
                         $res_qry = "isbn =" .$isbn;
                    else
                    {
                         if ($autor != "")
                              $res_qry = 'autor like "%'.$autor.'%"';
                        if ($titulo != ""){
                              if (strlen($res_qry) > 0)
                                   $res_qry .=" and ";
                              $res_qry .="titulo like \"%$titulo%\"";
                          }
                         if ($editorial != ""){
                              if (strlen($res_qry) > 0)
                                   $res_qry .=" and ";
                              $res_qry .="editorial like \"%$editorial%\"";
                    }
                         if ($ano != ""){
                              if (strlen($res_qry) > 0)
                                   $res_qry .=" and ";
                              $res_qry .="ano like \"%$ano%\"";
                    }
```

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

```
                    }
                    return "select * from libros where $res_qry;";
               }

               $conexion = mysql_connect(“localhost”, “root”, “”);
               if (!$conexion)
                    die(“No se puede conectar a la base de datos”);
               $bd = mysql_select_db("libros", $conexion);
               $query = con-
struye_query($_GET['FISBN'],$_GET['FAUTOR'],$_GET['FTITULO'],$_GET['FEDITORIAL'],$_GE
T['FANO']);
               $resultado = mysql_query($query, $conexion);
               echo "QUERY: " . $query;
              if($resultado){
                     echo "<TABLE
BORDER=1><TR><TD>ISBN</TD><TD>AUTOR</TD><TD>TITULO</TD></TR>";

                     while ($entrada = mysql_fetch_object($resultado))
                     {
                          echo "<TR><TD><A HREF='zoom.php?FISBN=$entrada-
>isbn'>$entrada->isbn</A></TD>";
                          echo "<TD> $entrada->autor</TD>";
                          echo "<TD> $entrada->titulo</TD>";
                     }

                     echo "</TABLE>";
                          mysql_free_result($resultado);
               }
               else
                    echo "No se ha encontrado ningún registro<BR>";

               mysql_close($conexion);
          ?>
     </BODY>
</HTML>
```

**Figura 50 - Bloque de código: Búsqueda en una base de datos (Búsqueda)**

```
<!-- zoom.php -->
<HTML>
     <HEAD>
          <TITLE>Resultado de consulta</TITLE>
     </HEAD>
     <BODY BGCOLOR="#ffffff">
          <H1>Consulta de libros en una base de datos</H1>
          <?php
               $conexion = mysql_connect("localhost", "root", "");
               if (!$conexion)
                    die("No se puede conectar a la base de datos");
               $bd = mysql_select_db("libros", $conexion);
               $query = "select * from libros where isbn = ".$_GET['FISBN'].";";
               $resultado = mysql_query($query, $conexion);
              if($resultado){
                     echo "<FORM METHOD=GET ACTION=\"update.php?FISBN=$entrada->isbn\">";
                     echo "<TABLE>";
                     while ($entrada = mysql_fetch_object($resultado))
                     {
                          echo "<TR><TD>ISBN</TD><TD><INPUT ";
                          echo "NAME=FISBN TYPE=TEXT SIZE=10 VALUE='$entrada-
>isbn'></TD></TR>";
```


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

```
                         echo "<TR><TD>AUTOR</TD><TD><INPUT ";
                         echo "NAME=FAUTOR TYPE=TEXT SIZE=30 VALUE='$entrada-
>autor'></TD></TR>";
                         echo "<TR><TD>TITULO</TD><TD><INPUT ";
                         echo "NAME=FTITULO TYPE=TEXT SIZE=40 VALUE='$entrada-
>titulo'></TD></TR>";
                         echo "<TR><TD>EDITORIAL</TD><TD><INPUT ";
                         echo "NAME=FEDITORIAL TYPE=TEXT SIZE=30 VALUE='$entrada-
>editorial'></TD></TR>";
                         echo "<TR><TD>AÑO DE EDICIÓN</TD><TD><INPUT ";
                         echo "NAME=FANO TYPE=TEXT SIZE=8 VALUE='$entrada-
>ano'></TD></TR>";
                    }
                    echo "</TABLE>";
                    echo '<INPUT TYPE="submit" VALUE="Enviar">  ';
                    echo '<INPUT TYPE="reset" VALUE="Borrar">';
                    echo "</FORM>";
                    mysql_free_result($resultado);
               }
               else
                    echo "No se ha encontrado ningún registro<BR>";


               mysql_close($conexion);
          ?>
     </BODY>
</HTML>
```

**Figura 51 - Bloque de código: Consulta a una base de datos**

```
<HTML>
     <HEAD>
          <TITLE>Actualización de un registro</TITLE>
     </HEAD>
     <BODY BGCOLOR="#ffffff">
          <H1>Consulta de libros en una base de datos</H1>
          <?php
               $conexion = mysql_connect("localhost", "root", "");
               if (!$conexion)
                    die("No se puede conectar a la base de datos");
               $bd = mysql_select_db("libros", $conexion);
               $query = "update libros set autor = \"".$_GET['FAUTOR']."\", titulo =
\"".$_GET['FTITULO']."\",
               editorial = \"".$_GET['FEDITORIAL']."\", ano = \"".$_GET['FANO']."\"
where isbn = \"".$_GET['FISBN']."\";";
               echo $query;
               $resultado = mysql_query($query, $conexion);
               if ($resultado)
                    echo "Actualización realizada correctamente para ISBN
".$_GET['FISBN'];
               else
                    echo "Se ha producido un error al actualizar el ISBN
".$_GET['FISBN'];
               mysql_close($conexion);
          ?>
     </BODY>
</HTML>
```

**Figura 52 - Bloque de código: Actualizar un registro de la base de datos**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

# 4 Web dinámica. Ejemplos

## 4.1 Introducción

Una de las divisiones que podemos realizar entre todos los tipos de páginas Web existentes podría ser entre estáticas y dinámicas.
Una página Web estática presenta las siguientes características:

- Ausencia de movimiento y funcionalidades.
- Absoluta opacidad a los deseos o búsquedas del visitante a la página.
- Realizadas en xHTML o HTML.
- Para cambiar los contenidos de la página, es imprescindible acceder al servidor donde está alojada la página.
- El usuario no tiene ninguna posibilidad de seleccionar, ordenar o modificar los contenidos o el diseño de la página a su gusto.
- El proceso de actualización es lento, tedioso y esencialmente manual.
- No se pueden utilizar funcionalidades tales como bases de datos, foros, etc.

Por el contrario, una página Web dinámica tiene las siguientes características:

- Gran número de posibilidades en su diseño y desarrollo.
- Cuenta con un gran número de soluciones prediseñadas de libre disposición.
- Permite un gran número de funcionalidades tales como bases de datos, foros, contenido dinámico, etc.
- El visitante puede alterar el diseño, contenidos o presentación de la página a su gusto.
- En su realización se utilizan diversos lenguajes y técnicas de programación.
- Pueden realizarse íntegramente con software de libre distribución.
- El proceso de actualización es sumamente sencillo, sin necesidad de entrar en el servidor.
- Existe una amplia comunidad de programadores que brinda apoyo desinteresado.

En definitiva, el concepto de página Web dinámica se ha impuesto en el mundo del diseño y de la empresa en Internet. Las páginas Web dinámicas permiten interactuar con el visitante y le ofrecen un amplio abanico de posibilidades.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 4.2 Los sistemas gestores de contenido

Vamos a analizar los diferentes CMS (*Content Manager System*), o en español los Sistemas Gestores de Contenido. Lo primero que hay que hacer es dar una definición relativamente formal de lo que se entiende por CMS, un CMS consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. El sistema permite manejar de manera independiente el contenido por una parte y el diseño por la otra. Consiste en una serie de programas en un servidor Web, y opcionalmente, una serie de programas cliente que permiten acceder a esos programas del servidor [22-25].
Se le da el nombre de Gestor de contenidos porque permite la gestión de los contenidos, de forma uniforme, accesible, y cómoda, consiguiendo un portal Web dinámico, con actualizaciones periódicas, y sobre el que pueden trabajar una o más personas, cada una con una funcionalidad determinada. Además, es importante destacar que la gestión de los contenidos se hace teniendo en cuenta todo el ciclo de vida de los contenidos, incluyendo creación, mantenimiento y eliminación.
Desde el punto de vista del cliente, se trata de un sitio Web dinámico, con una apariencia de interfaz uniforme, con un diseño centrado en el usuario, y que permite llevar a cabo fácilmente tareas para las que se ha diseñado. Los sistemas gestores de contenido suelen tener un parte pública donde se presentan los diferentes contenidos y una parte privada donde se administran dichos contenidos.
A la hora de analizar los diferentes sistemas de contenido se puede hacer una clasificación bien diferenciada, en cuanto a la organización de contenidos y filosofía seguida en dicha organización:

- Sistemas de tipo Wiki.
- Sistemas Weblog.
- Sistemas de Foros.
- Sistemas de tipo portal, para noticias.
- Plataformas de teleformación
- Portales integrales.
- Se realizará un análisis detallado en los apartados siguientes.

Para concluir, un Sistema Gestor de Contenidos se utiliza para la gestión de un sitio Web, y por pequeño que sea éste, su organización se realizada de una forma más adecuada y ordenada. Además permite tener una apariencia y navegación uniforme en todo el sitio, realizando las tareas de actualización y gestión del contenido de una manera más simple.

### 4.2.1 Clasificación de CMS

#### 4.2.1.1 Sistemas WIKI

Un Wiki (del hawaiano *wiki wiki*, que significa rápido) es una filosofía de sitio Web en donde se acepta que los usuarios creen, editen, borren o modifiquen el contenido de una página Web de forma interactiva, fácil y rápida. Esto hace que un sistema Wiki sea una herramienta efectiva para la escritura colaborativa. Es decir, cuando alguien edita una página Wiki, sus cambios aparecen inmediatamente en la Web, sin pasar por ningún tipo de revisión previa.
Por otra parte, también se puede llamar Wiki a una colección de páginas hipertexto, que pueden ser visitadas y editadas por cualquier persona, sin ningún tipo de revisión final.
Los orígenes de los Wikis están en la comunidad de patrones de diseño. El primer WikiWikiWeb (http://c2.com/) fue creado por Ward Conninghan quien inventó y dio nombre al concepto wiki

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

en 1995. En 2001 se fundó el proyecto más importante para el que se utiliza un wiki, es la enciclopedia libre y gratuita wikipedia.
La principal ventaja es la posibilidad de crear y mejorar las páginas de forma instantánea, donde se otorga gran libertad al usuario por medio de una interfaz muy simple. Además hay que tener en cuenta la rapidez con la que se actualizan los diferentes contenidos.
El principal inconveniente es lo que se conoce como vandalismo, que consiste en hacer ediciones, generalmente por desconocidos, que borran contenido, o agregan contenido ofensivo o inapropiado, o simplemente incumplen las normas del Wiki.
Algunos de los sistemas Wiki actuales son:

- UseModWiki. el más antiguo. Enlace: http://www.usemod.com/.
- MediaWiki. utilizado en los proyectos de Wikimedia. Enlace: http://www.wikimedia.org/.
- PhpWiki. Basado en UseMod. Enlace: http://phpwiki.sourceforge.net/phpwiki/.
- TikiWiki. CMS completo, con un wiki muy desarrollado. Enlace: http://sobreatico.net/tiki-index.php?page=TikiWiki.
- DokuWiki. Wiki muy completo desarrollado en php, que no utiliza base de datos. Enlace: http://freshmeat.net/projects/dokuwiki/.
- WikkaWiki. Enlace: http://wikkawiki.org/HomePage.


**Figura 53 - Ejemplo de Wiki: Wikipedia**

### 4.2.1.2 Sistemas Weblogs

Un Weblog, bitácora, lista de sucesos, o simplemente blog, es un sitio Web periódicamente actualizado que recopila cronológicamente textos o artículos de uno o varios autores donde el más reciente aparece el primero. Conservando el autor la libertad de publicar lo que crea pertinente.
Algunas de las características más importantes son las siguientes:


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Enlaces.** Algunos elementos comunes en todos los blogs es una lista de enlaces a otros blogs, un archivo de anotaciones anteriores, enlaces permanentes, etc.
- **Sindicación.** Los blogs se publican en una gran multiplicidad de formatos. Aparte de HTML suelen incluir algún medio para sindicarlos. Generalmente se usa RSS (*Really Simple Syndication),* aunque a partir del año 2004 se ha popularizado también el Atom.
- **Permiten comentarios.** La realimentación es una faceta importante en estos sistemas.
- **Ubicación permanente.** La información publicada adquiere una ubicación permanente, es decir, cada entrada del blog tiene un URL que se utiliza a lo largo de todo el ciclo de esa entrada., que se le suele denominar permalink.

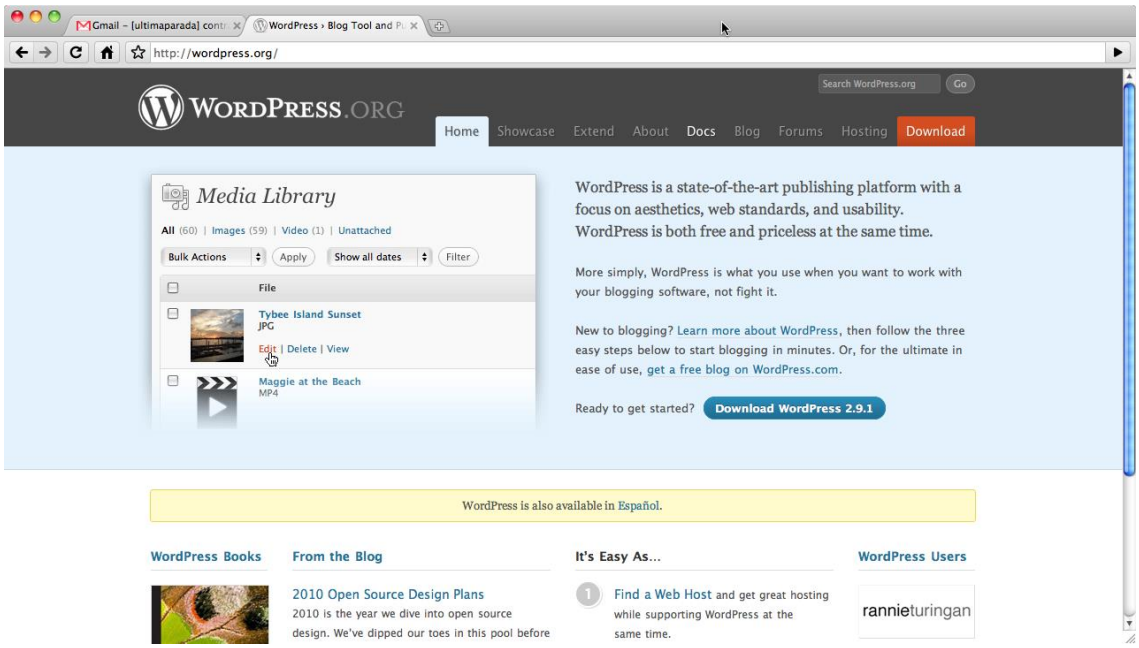



**Figura 54 - Ejemplo de blog: Wordpress**

#### 4.2.1.3 Sistema foro

Los foros en Internet, o foros de mensajes, de opinión o de discusión son por definición una aplicación Web que da soporte a discusiones en línea. Son sitios que dan la posibilidad a los usuarios a discutir, o simplemente a compartir información sobre la temática del sitio, con lo que se llega a formar una comunidad entorno a un interés común.

Los foros son los descendientes modernos de los sistemas de noticias BBS (*Bulletin Borrad System*) y Usenet, muy populares en la década de los ochenta.

Si realizamos una comparación con los Wiki, la diferencia principal reside en la imposibilidad de modificar los aportes de otros miembros a menos que se tenga unos privilegios especiales de administración. La comparación con los WebBlogs arroja la diferencia básica que en un foro existen gran cantidad de usuarios con discusiones anidadas, pero siendo el sistema de comentarios bastante homogéneo.

Existen gran cantidad de Sistemas de foros, algunos de los más importantes que existen actualmente son:

- Invision Power Borrad. http://www.invisionpower.com.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- SMF. http://www.simplemachines.org/.
- phpBB. http://www.phpbb.com/.
- vBulletin. http://www.vbulletin.com/.

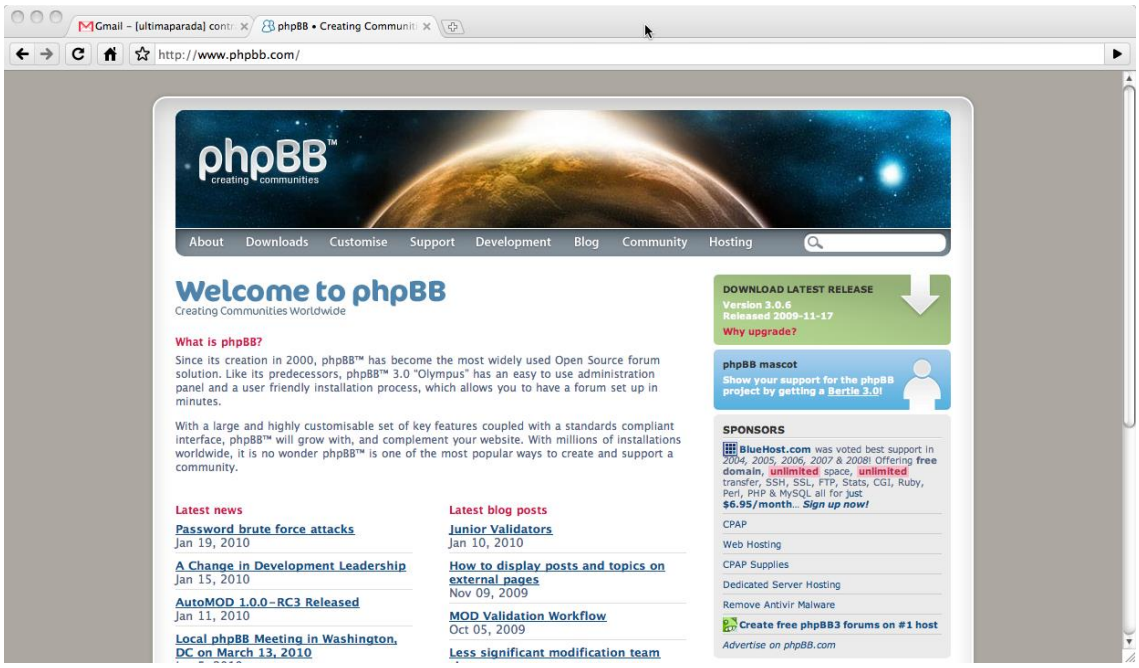



**Figura 55 - Ejemplo de roro: phpBB**

#### 4.2.1.4 Plataformas de teleformación

Una plataforma de teleformación, o un sistema de gestión de aprendizaje en red, es una herramienta informática y telemática organizada en función de unos objetivos formativos de forma integral y de unos principios de intervención psicopedagógica y organizativos, de manera que se cumplen los siguientes criterios básicos.

- Posibilita el acceso remoto tanto a profesores como a alumnos en cualquier momento desde cualquier lugar con conexión a Internet o a redes con protocolo TCP/IP.
- Utiliza un navegador. Permite a los usuarios acceder a la información a través de navegadores estándares (como Nestscape, Internet Explorer, Opera,..), utilizando el protocolo de comunicación http.
- El acceso es independiente de la plataforma o del ordenador personal de cada usuario. Es decir utilizan estándares de manera que la información puede ser visualizada y tratada en las mismas condiciones, con las mismas funciones y con el mismo aspecto en cualquier ordenador.
- Tiene estructura servidor/cliente. Es decir permite retirar y depositar información.
- El acceso es restringido y selectivo.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Incluye como elemento básico una interfaz gráfica común, con un único punto de acceso, de manera que en ella se integran los diferentes elementos multimedia que constituyen los cursos: texto, gráficos, vídeo, sonidos, animaciones, etc.
- Utiliza páginas elaboradas con un estándar aceptado por el protocolo http: HTML o XML.
- Realiza la presentación de la información en formato multimedia. Los formatos HTML o XML permiten presentar la información, además de en hipertexto, pueden utilizarse gráficos, animaciones, audio y vídeo (tanto mediante la transferencia de ficheros como en tiempo real).
- Permite al usuario acceder a recursos y a cualquier información disponible en Internet. Bien a través de enlaces y las herramientas de navegación que le proporciona el navegador en Internet, bien a través del propio entorno de la plataforma.
- Permite la actualización y la edición de la información con los medios propios que han de ser sencillos o con los medios estándares de que disponga el usuario. Tanto de las páginas web como de los documentos depositados.
- Permite estructurar la información y los espacios en formato hipertextual. De esta manera la información se puede organizar, estructurada a través de enlaces y asociaciones de tipo conceptual y funcional, de forma que queden diferenciados distintos espacios y que esto sea percibible por los usuarios.
- Permita establecer diferentes niveles de usuarios con distintos privilegios de acceso. Debe contemplar al menos: el administrador, que se encarga del mantenimiento del servidor, y de administrar espacios, claves y privilegios; el coordinador o responsable de curso, es el perfil del profesor que diseña, y se responsabiliza del desarrollo del curso, de la coordinación docente y organizativa del curso en la plataforma; los profesores tutores, encargados de la atención de los alumnos, de la elaboración de materiales y de la responsabilización docente de las materias; y los alumnos.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

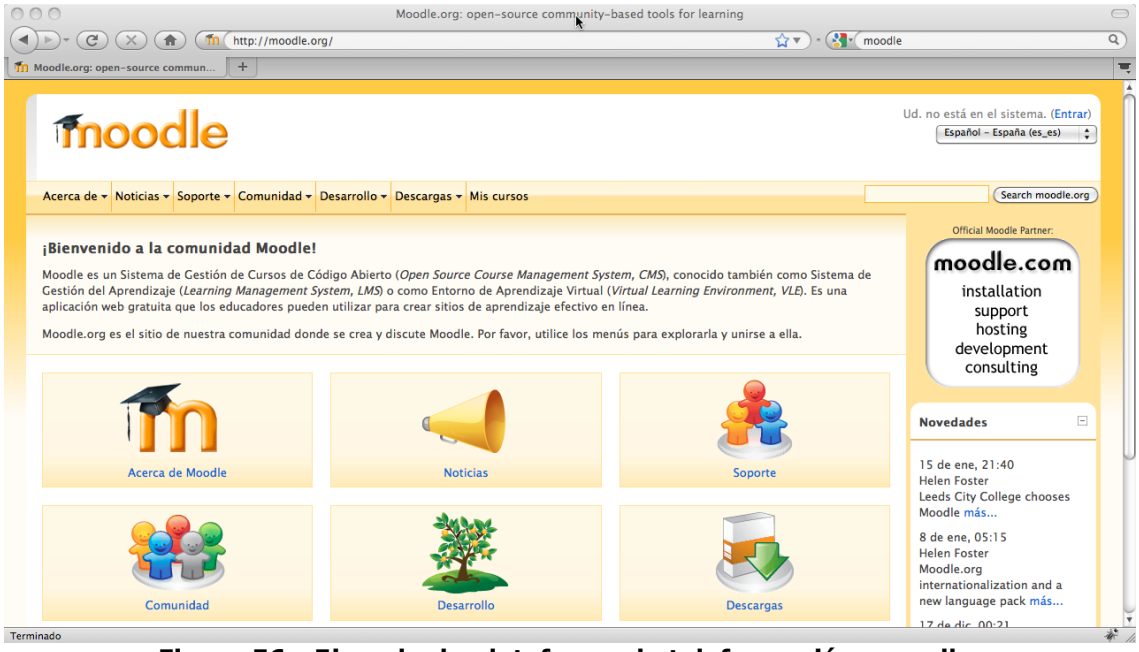



**Figura 56 - Ejemplo de plataforma de teleformación: moodle**

#### 4.2.1.5 Sistema de tipo portal

Estos sistemas incluyen la mayoría de las funcionalidades y características principales de los Sistemas Gestores de Contenidos analizados previamente, pero además incluyen otras características como pueden ser gestión de la información estática y documentación, encuestas, mensajería, etc. El objetivo principal es proporcionar al Webmaster una herramienta para la creación de comunidades basadas en un portal, y que este sistema Web sea totalmente automatizado.
Otra de las características más importantes de este tipo de sistemas es la gran cantidad de idiomas soportados, siendo estos superiores a veinte. También se puede personalizar el estilo gráfico utilizando una seria de plantillas, de una manera trivial.
El mayor problema que tienen estos sistemas viene dado por la escasa seguridad que ofrecen tanto a administradores como a usuarios, ya que son muy vulnerables a ataques externos.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

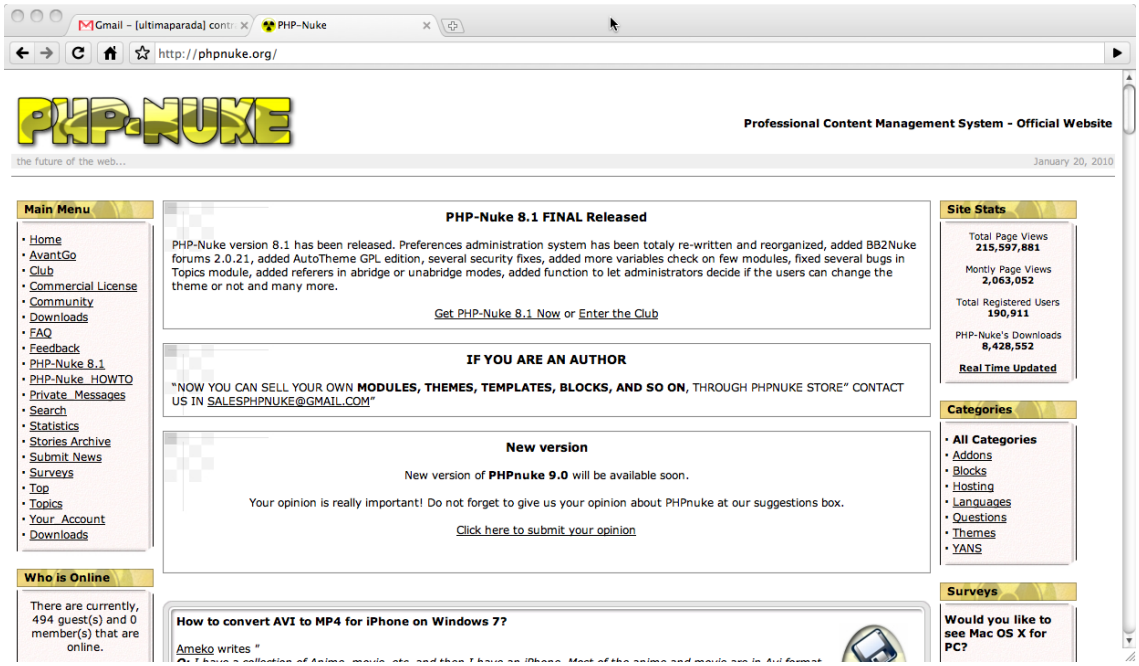



**Figura 57 - Ejemplo de sistema de tipo portal: phpnuke**

#### 4.2.1.6 Sistemas generales

Permiten gestionar información de cualquier tipo, y son generalmente ampliables y configurables con nuevas funcionalidades, mediante un simple proceso de instalación.
Sitios que combinan varias características para crear una comunidad en línea. Básicamente se combinan, un foro y un blog conjuntamente, un apartado para noticias y algunas veces apartados para artículos que no entran dentro del apartado noticias, conocidos habitualmente como información estática.
Una de las características básicas es la modularidad de estos sistemas y la posibilidad de añadir nuevas funcionalidades de una manera rápida. Quizás la única deficiencia sea la rigidez que suelen tener estos sistemas. Aunque esta deficiencia esta invirtiéndose en las últimas fechas.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

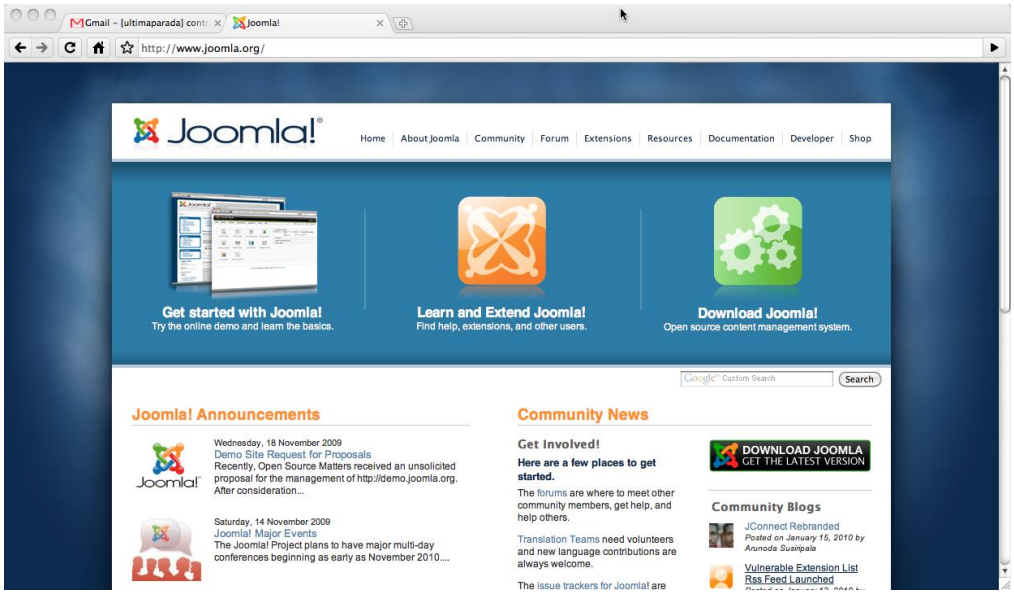



**Figura 58 - Ejemplo de Sist. General: Joomla**

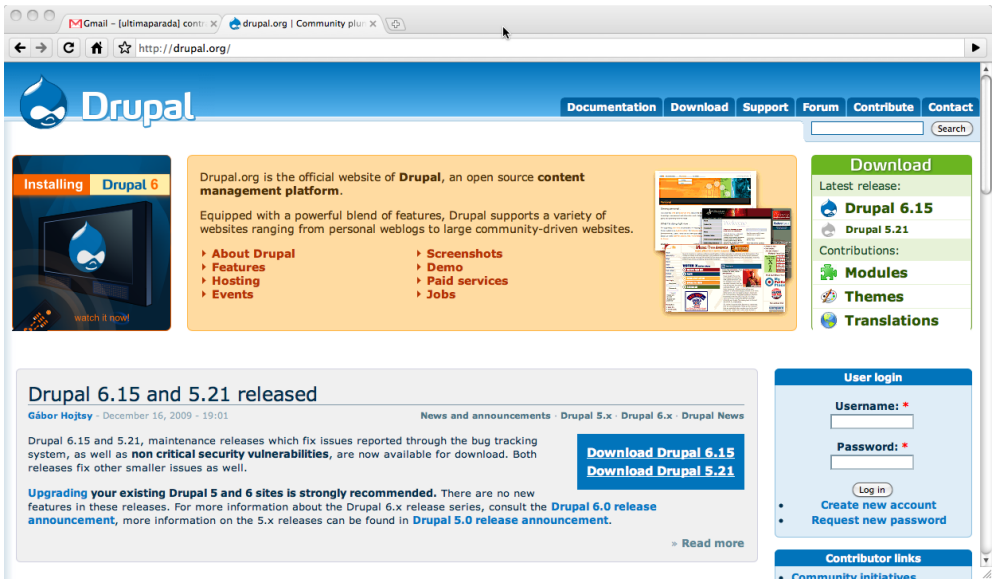



**Figura 59 - Ejemplo de Sit. general: Drupal**



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 4.3 Drupal

### 4.3.1 Introducción

Drupal es un sistema de código abierto, con licencia GNU/GPL, escrito en PHP para la gestión contenido de forma modular y muy configurable. Destaca por la calidad de su código y el de las páginas generadas, el respecto de los estándares Web y un especial énfasis en la usabilidad y consistencia de todo el sistema.
El diseño de Drupal es especialmente idóneo para construir y gestionar comunidades en Internet. No obstante, su flexibilidad y adaptabilidad, así como la gran cantidad de módulos adicionales disponibles, hace que sea adecuado para realizar muchos tipos diferentes de sitios Web.

- Es totalmente configurable, de tal manera que el administrador de un sitio puede activar o desactivar diferentes características y establecer configuraciones que cambian el aspecto y funcionalidad del sitio.
- Tiene un sistema de privilegios que hacen posible crear diferentes tipos de usuario y que estos puedan ver y hacer cosas diferentes en el sitio.
- Está diseñado para ser fácilmente extensible a través de módulos (bloques de código que proveen extra funcionalidad o mejoras). Algunos módulos vienen con todas las instalaciones de Drupal (módulos del núcleo), mientras que otras pueden ser instaladas y descargadas individualmente del sitio Web de Drupal (módulos contribuidos).
- El aspecto básico de un sitio Drupal puede ser cambiado usando diferentes temas. Al igual que con los módulos, existen temas del núcleo y temas contribuidos.

En la actualidad, diferentes entes públicos y privados que utilizan este gestor de contenido por su potencia, usabilidad, sencillez de uso, escalabilidad y rendimiento. Entre ellos podemos encontrar la propia Universidad de Salamanca, tal y como podemos observar en la siguiente figura.


**Figura 60 - La página web de la Universidad de Salamanca esta construida con Drupal**

### 4.3.2 Características de Drupal

#### 4.3.2.1 Características generales

- **Ayuda on-line.** Un robusto sistema de ayuda online y páginas de ayuda para los módulos del núcleo, tanto para usuarios como para administradores.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Búsqueda.** Todo el contenido en Drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento.
- **Código abierto.** El código fuente de Drupal está disponible bajo los términos de la licencia GNU/GPL, es posible extender o adaptar Drupal según las necesidades.
- **Módulos.** La comunidad de Drupal ha contribuido con muchos módulos que proporcionan diferentes funcionalidades como página de categorías, autenticación mediante jabber, mensajes privados, bookmarks, etc.
- **Personalización.** El núcleo de Drupal implementa un robusto entorno de personalización. Tanto el contenido como la presentación pueden ser individualizados de acuerdo las preferencias definidas por el usuario.
- **URLs amigables.** Drupal usa el mod_rewrite de Apache para crear URLs que son manejables por los usuarios y los motores de búsqueda.

4.3.2.2 Gestión de usuarios

- **Autenticación de usuarios.** Los usuarios se pueden registrar e iniciar sesión de forma local o utilizando un sistema de autenticación externo como Jabber, Blogger, LiveJournal o otro sitio Drupal. Para su uso en una intranet, Drupal se puede integrar con un servidor LDAP.
- **Permisos basados en roles.** Los administradores de Drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un 'rol' y agrupar los usuarios por roles.

4.3.2.3 Gestión de contenido

- **Control de versiones.** El sistema de control de versiones de Drupal permite seguir y auditar totalmente las sucesivas actualizaciones del contenido: qué se ha cambiado, la hora y la fecha, quién lo ha cambiado, etc. También permite mantener comentarios sobre los sucesivos cambios o deshacer los cambios recuperando una versión anterior.
- **Enlaces permanentes (*Permalinks*).** Todo el contenido creado en Drupal tiene un enlace permanente asociado a él, para que pueda ser enlazado externamente sin temor de que el enlace falle en el futuro.
- **Objetos de Contenido (*Nodes*).** El contenido creado en Drupal es, funcionalmente, un objeto (Nodo). Esto permite un tratamiento uniforme de la información, como una misma cola de moderación para envíos de diferentes tipos, promocionar cualquiera de estos objetos a la página principal o permitir comentarios -o no- sobre cada objeto.
- **Plantillas (*Templates*).** El sistema de temas de Drupal separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio web. Se pueden crear plantillas con HTML y/o con PHP.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Sindicación del contenido.** Drupal exporta el contenido en formato RDF/RSS para ser utilizado por otros sitios web.

4.3.2.4 Plataforma

- **Independencia de la base de datos.** Aunque la mayor parte de las instalaciones de Drupal utilizan MySQL, existen otras opciones diferentes. Drupal incorpora una capa de abstracción de base de datos que actualmente está implementada y mantenida para MySQL y PostgresSQL, aunque permite incorporar fácilmente soporte para otras bases de datos.
- **Multiplataforma.** Drupal ha sido diseñado desde el principio para ser multiplataforma. Puede funcionar con Apache o Microsoft IIS como servidor Web y en sistemas como Linux, BSD, Solaris, Windows y Mac OS X. Por otro lado, al estar implementado en PHP, es totalmente portable.
- **Múltiples idiomas y Localización.** Drupal está pensado para una audiencia internacional y proporciona opciones para crear un portal multilingüe.

4.3.2.5 Administración y análisis

- **Administración via Web.** La administración y configuración del sistema se puede realizar enteramente con un navegador y no precisa de ningún software adicional.
- **Análisis, Seguimiento y Estadísticas.** Drupal puede mostrar en las páginas web de administración informes sobre referrals (enlaces entrantes), popularidad del contenido, o de cómo los usuarios navegan por el sitio.
- **Registros e Informes.** Toda la actividad y los sucesos del sistema son capturados en un 'registro de eventos', que puede ser visualizado por un administrador.

4.3.2.6 Características de comunidad

- **Comentarios enlazados.** Drupal proporciona un potente modelo de comentarios enlazados que posibilita seguir y participar fácilmente en la discusión sobre el comentario publicado. Los comentarios son jerárquicos, como en un grupo de noticias o un foro.
- **Encuestas.** Drupal incluye un módulo que permite a los administradores y/o usuarios crear encuestas on-line totalmente configurables.
- **Foros de discusión.** Drupal incorpora foros de discusión para crear sitios comunitarios vivos y dinámicos.
- **Libro Colaborativo.** Esta característica es única de Drupal y permite crear un proyecto o "libro" a ser escrito y que otros usuarios contribuyan contenido. El contenido se organiza en páginas cómodamente navegables.

4.3.2.7 Rendimiento y escacalabilidad

- **Control de congestión.** Drupal incorpora un mecanismo de control de congestión que permite habilitar y deshabilitar determinados módulos o bloques dependiendo de la carga del servidor. Este mecanismo es totalmente configurable y ajustable.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Sistema de Cache.** El mecanismo de cache elimina consultas a la base de datos incrementando el rendimiento y reduciendo la carga del servidor.

### 4.3.3 Conceptos básicos

Drupal es un sistema de gestión de contenido para la web. El texto y los enlaces entre el contenido son almacenados en una base de datos, y las páginas se construyen dinámicamente para ser presentadas al usuario en respuesta a una petición web mediante un agente de usuario.
El contenido en drupal se almacena como **nodos**. Un nodo es un objeto de contenido genérico, que se puede corresponder con una página, un artículo, una entrada de blog, etc. A algunos tipos de nodos, dependiendo de la configuración, los usuarios pueden añadir comentarios (los comentarios no son considerados como nodos). Los nodos o los comentarios pueden necesitar ser aprobados por un moderador antes de ser publicados.
La presentación y el diseño de las páginas son gestionadas mediante **Temas** intercambiables. La presentación estándar de una página de drupal, consiste en tres columnas. La columna del centro es la "Columna de contenido". La más típica muestra resúmenes de los nodos publicados más recientemente por orden de fecha. Si hace clic en uno de los resúmenes, el contenido completo del nodo es mostrado en la columna central.
Las columnas izquierda y derecha se llaman normalmente "barras laterales". Las barras laterales pueden mostrar **bloques** o información relacionada. Los bloques a menudo contienen enlaces para navegar hacia otros nodos. Por ejemplo, puede haber bloques mostrando los últimos artículos enviados o los mas populares.
Para nuevas instalaciones, se muestran el bloque de inicio de sesión y el bloque de navegación que contiene un menú de acciones que se pueden llevar a cabo. Diferentes menús pueden ser mostrados en el bloque de navegación, dependiendo de lo que esté haciendo y de qué privilegios o permisos (**roles**) tenga el usuario.
Los bloques también pueden aparecer o no dependiendo de en qué parte del sitio Web esté y de qué acciones está realizando, y también de los privilegios que tenga. Por ejemplo, el bloque de inicio de sesión no aparece si la sesión ya ha sido iniciada, o el de "artículos más recientes" puede no mostrarse si no hay artículos disponibles. El administrador puede habilitar/deshabilitar distintos bloques.
Los nodos se pueden clasificar en categorías, también llamadas **taxonomías**. Los foros son un ejemplo de nodos de contenido organizados por categoría. Las categorías pueden ser jerárquicas lo que significa que una categoría "padre" contiene múltiples categorías "hijos".
A menudo, nuevas funcionalidades son implementadas en Drupal como **módulos**. Una vez que un administrador añade un archivo o carpeta de un módulo en el subdirectorio modules, la opción de utilizar el módulo aparece en la sección Administrar -> Módulos. Si el administrador activa el módulo, las funcionalidades asociadas con él se activan y aparecen en la sección correspondiente de Drupal. Un módulo puede definir un nuevos tipos de nodos, crear nuevas opciones en el menú o proporcionar nuevos bloques que estarán disponibles para mostrar en las barras laterales.

### 4.3.4 Instalación de Drupal

#### 4.3.4.1 Requisitos del sistema

Para la instalación del sistema gestor de contenidos Drupal, básicamente existen tres requisitos fundamentales:

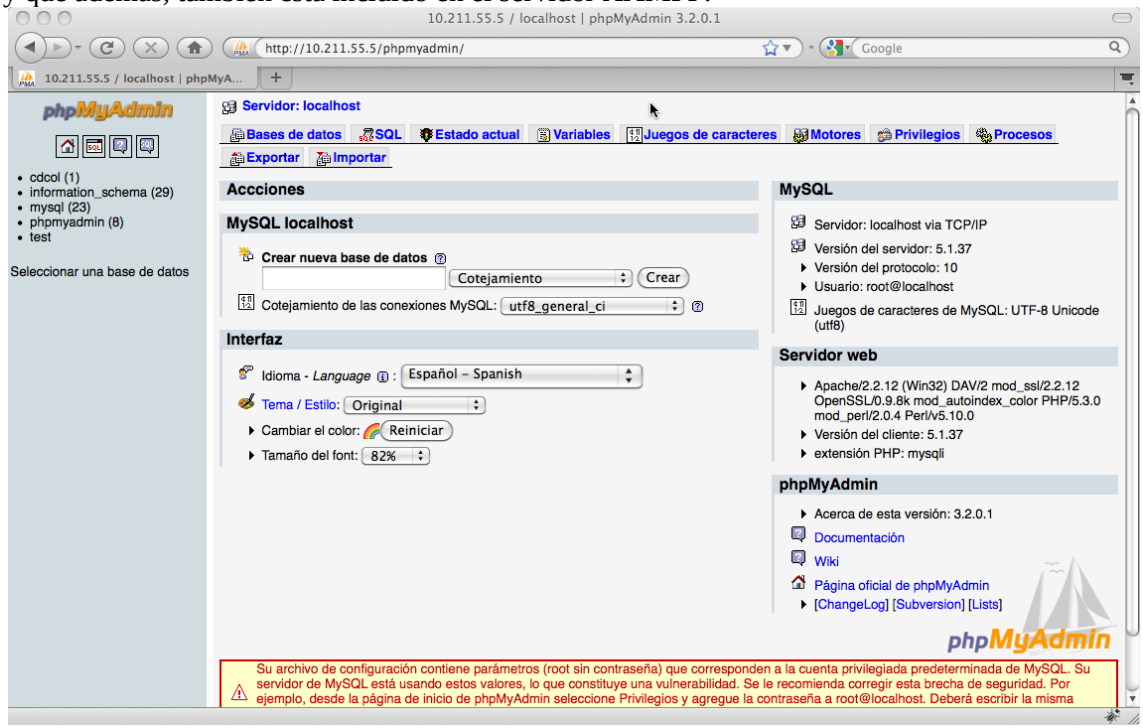

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

1. **Un servidor Web.** Se recomienda el uso de Apache, se ha comprobado que funciona perfectamente con las versiones 1.3.x y 2.0.x. Aunque también se ha comprobado la compatibilidad con el servidor MS-ISS de Microsoft.
2. **PHP.** Para el correcto funcionamiento de Drupal se requiere que el servidor Web sea capaz de interpretar scripts de PHP. Aunque suele funcionar sin problemas con la versión 4.x, una gran cantidad de módulos necesitan una versión mínima de 5.0.x de PHP.
3. **Un servidor de bases de datos que soporte PHP.** Se recomienda MySQL, pero también son posibles otros sistemas de gestores de bases de datos como por ejemplo PostgreSQL.

El servidor XAMPP (http://www.apachefriends.org/en/xampp.html) es un servidor independiente de plataforma y de código abierto, con el que podremos disponer de forma libre y fácil estos tres requisitos fundamentales para la instalación de Drupal.

#### 4.3.4.2 Proceso de instalación

Una vez que tenemos en nuestro máquina un servidor Web (Apache, http://www.apache.org/) y un servidor de bases de datos, como por ejemplo MySQL (http://www.mysql.com/) ya podemos proceder con la instalación de Drupal.
El primer paso de la instalación es la creación de una base de datos que posteriormente utilizará el sistema gestor de contenidos, para ello, podemos utilizar phpmyadmin (http://www.phpmyadmin.net/) que nos facilita la gestión de las bases de datos de nuestro sistema y que además, también esta incluido en el servidor XAMPP.




El proceso de creación de la base de datos es simple, tendremos que crear una base de datos y por cuestiones de seguridad un nuevo usuario de MySQL para el acceso a la base de datos que hemos creado.

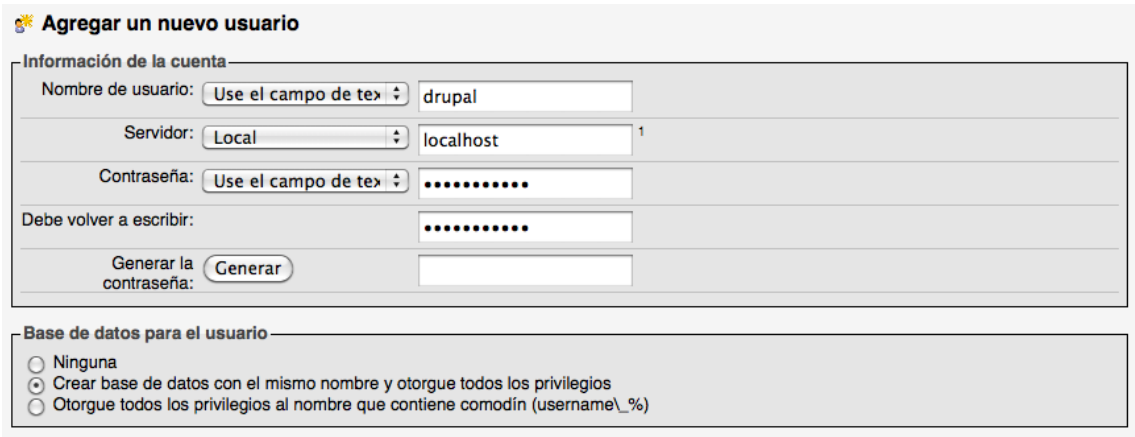
ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Para ello, nos dirigimos a la pestaña Privilegios y dentro de esa opción seleccionamos “Agregar un nuevo usuario”.


Para la creación de un nuevo usuario, se nos pide la siguiente información:

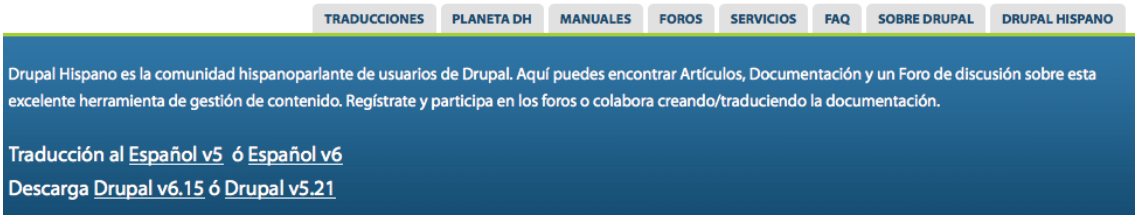
- **Nombre de usuario.** Es el nombre de usuario que deseamos, por ejemplo, *drupal*.
- **Servidor.** Habitualmente *localhost*
- **Contraseña.** Es la contraseña para el usuario. Por ejemplo, *drupal_pass.*

En esta misma ventana, tendremos que marcar la opción “Crear una base de datos con el mismo nombre y otorgue todos los privilegios”, no marcar ninguna casilla de la sección de Privilegios Globales.




Una vez tenemos creado el usuario y la base de datos, el siguiente paso es descargarnos la última versión de Drupal, en nuestro caso vamos a instalar la versión 6.0.x, concretamente la versión 6.0.15, aunque también podríamos instalar otras versiones antiguas como 5.0.x ó la nueva versión 7.0.x que aún esta en desarrollo. Para ello, nos dirigimos a la comunidad hispana de Drupal (http://drupal.org.es/) y descargamos la versión deseada.

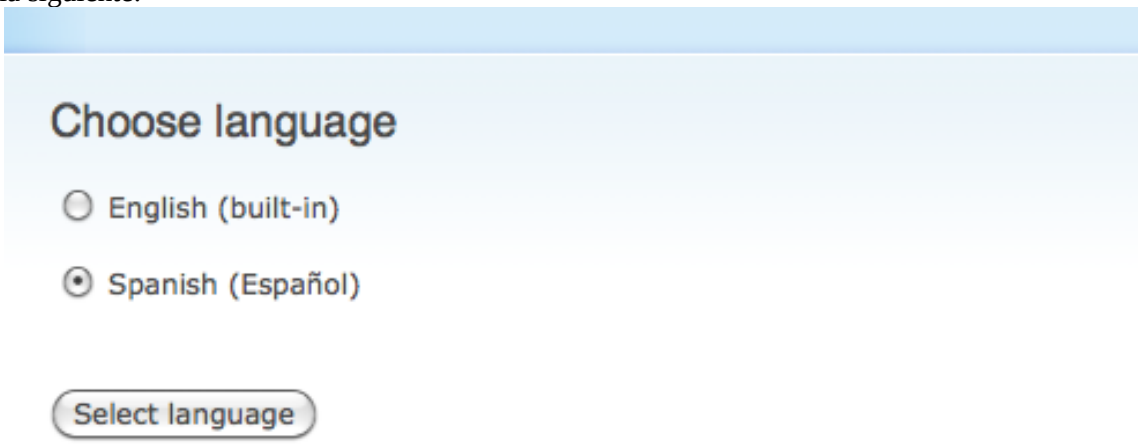
De la misma manera, también descargamos la traducción a español, para que así el proceso de instalación sea más sencillo. Una vez descargadas ambas versiones las descomprimimos y copiamos los archivos de traducción dentro del directorio que se ha descomprimido con el sistema base de drupal. Renombramos esta carpeta con el nombre *drupal*.


Copiamos la carpeta con el sistema base de drupal, en la que hemos descomprimido la traducción, al directorio principal de Apache, que en nuestro caso como utilizamos XAMPP es *xampp\htdocs*. Para comprobar que hemos realizado este paso correctamente este paso, accedemos mediante el


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

navegador a la dirección http://localhost/drupal, con lo que nos aparecerá una ventana similar a la siguiente:

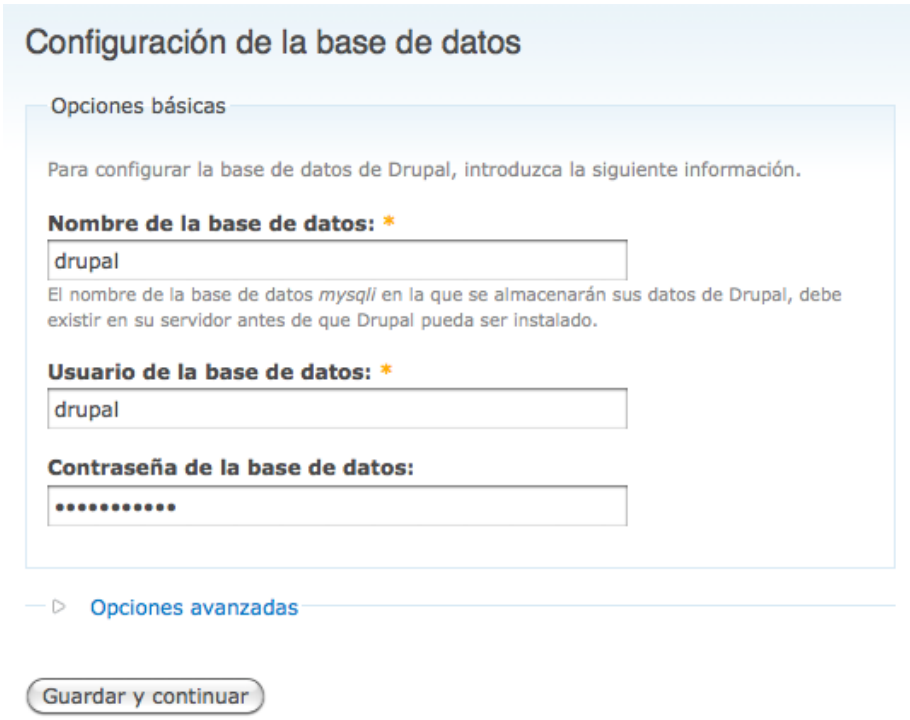



Una vez copiados los archivos, como vemos en la imagen anterior, el primer paso de la instalación es seleccionar el lenguaje en el que se va a realizar dicha instalación, elegir "Spanish (Español)" para continuar.
El siguiente paso Drupal, nos obligará a duplicar el archivo ./sites/default/**default.settings.php** en uno nuevo denominado ./sites/default/**settings.php** y darle a este archivo permisos de lectura y escritura. Una vez realizado el cambio, pulsamos sobre la opción "prueba de nuevo" y si no hemos cometido ningún error, podremos acceder al siguiente paso del proceso de instalación, el de configuración de la base de datos.


Para el proceso de configuración de la base de datos introduciremos los datos configuramos en MySQL, es decir:

- Usuario: drupal
- Contraseña: drupal_pass
- Base de datos: drupal

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

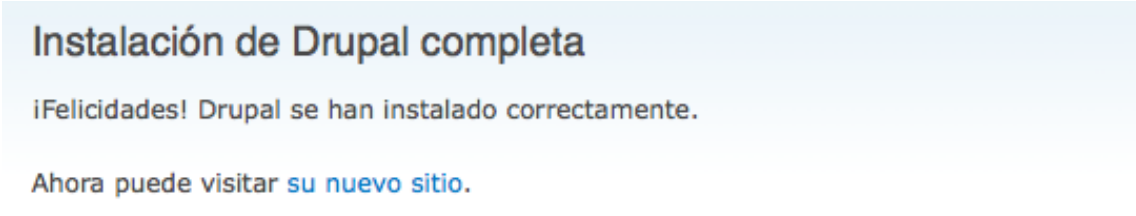



En principio, si tenemos la configuración por defecto de MySQL no es necesario cambiar ninguna de las opciones avanzadas preconfiguradas por drupal. Para comprobar que los datos introducidos son correctos, drupal ejecuta una consulta de prueba, si esta es satisfactoria, el siguiente paso, que se realiza de forma automática, es crear toda las tablas necesarias para el correcto funcionamiento del sistema, así como la inclusión de todos los registros necesarios.
A continuación, tendremos que configurar los datos generales de nuestro portal particular construido con drupal como información del sitio, cuenta de administración y configuración del servidor.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc


Una vez finalizado este paso, si todo es correcto, se nos indicará que Drupal se ha instalado correctamente.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc




Para ver el sitio que acabamos de instalar tan solo es necesario pulsar sobre "su nuevo sitio" o acceder directamente a http://localhost/drupal/.


### 4.3.5 Administración de básica

Drupal es una plataforma extremadamente flexible que permite muchas opciones para cambiar el aspecto de su sitio, cómo los usuarios interaccionan con él, o los tipos de contenido que se pueden mostrar. Aunque existen muchas opciones de configuración, Drupal funciona perfectamente una vez que se ha instalado y no es necesaria mucha configuración inicial.
A medida que el sitio Web evoluciona y la demanda crece, es fácil ir adaptando el aspecto o la funcionalidad a las necesidades. Primero ajustando las opciones de configuración, instalando nuevos módulos o temas visuales y, en último término, ya que Drupal es un producto que se entrega con una licencia GNU/GPL, se puede modificar el código adaptándolo hasta el infinito.
El lugar para comenzar la configuración es siguiendo el menú administrar, lo que le llevará a las opciones básicas del sitio. Debe haber iniciado sesión y tener el correspondiente permiso para acceder a esta página

#### 4.3.5.1 Administración de contenido

Desde la sección de administración se contenido, se permite las siguientes opciones:

- Ver, editar, moderar y eliminar los diferentes contenidos del portal.
- Publicar los contenidos mediante RSS.
- Definir diferentes tipos de contenido.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Categorizar los diferentes tipos de contenidos del portal.

Existen varios tipos de contenidos que se pueden publicar usando Drupal. Muchos de estos están organizados en lo que llamamos nodos. El contenido es agregado o actualizado por medio de formularios en páginas Web. Entonces, para agregar un artículo, se abre el formulario, ingresa el texto en este (título, contenido, etc.) y presiona un botón para enviar el formulario.
El contenido en los sitios Web Drupal usualmente está organizado usando categorías por medio de un sistema llamado "taxonomía". Una taxonomía tiene diferentes "términos" que son usados como categorías para los artículos. Cuando se está agregando un artículo, encontrará una lista desplegable de temas. Seleccionando uno, se define donde categorizar el artículo.
El sitio se puede configurar con una "cola de envíos". Esto quiere decir que los artículos enviados son marcados para su revisión. Cuando el administrador del sitio pueda ver su envío, tomará la decisión acerca de si cumple los requisitos para su publicación o no.
Los comentarios le permiten a un usuario interactuar con los contenidos de un sitio para responder a un artículo, ofreciendo sus propias ideas, adiciones o críticas.
Cuando abra un artículo, busque los vínculos relacionados con comentarios en la parte inferior del artículo. Si no ha ingresado al sitio, puede aparecer "ingrese o regístrese para publicar comentarios". Al ingresar al sitio, debe aparecer algo como "Agregar nuevo comentario". Siga el vínculo y estará listo para hacer comentarios.
El sistema de comentarios en Drupal es basado en hilos. Esto quiere decir que usted puede comentar acerca de un artículo, o responder a un comentario existente. Si responde a un comentario, su comentario comenzará con una sangría para indicar que es parte de una discusión

#### 4.3.5.2 Construcción del sitio.

A través de esta sección, tendremos acceso a la configuración de elementos estructurales de Drupal.

- **Bloques.** Los bloques son cajas de contenido representadas en un área o región de una página Web. A través de esta sección se puede asignar un bloque a una región (Lateral izquierdo, Lateral derecho, Contenido, Encabezado y Pie de página).
- **Menús.** Los menús definen la estructura de navegación por el portal Web. El menú principal se configura a través de la opción "Enlaces Primarios", pero también, se pueden definir otros menús secundarios que no definen una estructura general del sitio, pero facilitan la navegación en áreas concretos del portal.
- **Módulos.** Los módulos son componentes que extienden la funcionalidad del núcleo de drupal. Existe una gran cantidad de módulos preinstalados, pero se pueden añadir de forma rápida otros nuevos adaptados a las necesidades concretas de nuestro portal.

  Los nuevos módulos se pueden descargar desde http://drupal.org/project/Modules, para instalarlos, basta descomprimir el fichero descargado en el directorio modules. Para activar el módulo será necesario hacerlo explícitamente desde esta subsección del área de administración.
- **Temas.** Existen muchos temas disponibles para su descarga en el sitio Web de drupal, con los que podrá comenzar. La instalación de un nuevo tema es un proceso muy simple:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

1. Es necesario descargar el paquete correspondiente desde http://drupal.org/project/Themes verificando que la versión del tema es compatible con la versión de drupal que hemos instalado.
2. Coloque el contenido del paquete en el directorio themes de drupal.
3. Dirigirse a la interfaz de administración de drupal y comprobar que se ha detectado automáticamente el nuevo tema.
4. Configurar las preferencias del tema y seleccionar el tema que se desea que se muestre por defecto.

2. **Traducir interfaz.** A través de esta subsección podremos añadir nuevos lenguajes a nuestra sistema base.

#### 4.3.5.3 Configuración del sitio

A través de esta sección se puede acceder a la configuración general de nuestro portal:

- Fecha y Hora.
- Formatos de entrada
- Idiomas
- Información del sitio
- Información de errores.
- Mantenimiento del sitio.
- Registros y alertas
- Rendimiento del sitio
- Sistema de archivos.
- Activar la opción de URLs límpios.
- Tema de la administraciónn

#### 4.3.5.4 Administración de usuario

En esta sección tenemos a nuestra disposición la gestión de usuarios de drupal, la cual no ofrece las siguientes opciones:

- Configurar opciones generales de usuarios.
- Gestionar roles
- Gestionar usuario.
- Gestionar reglas de acceso.

### 4.3.6 Módulos aplicados al comercio electrónico

Al sistema gestor de contenidos Drupal se le pueden instalar diferentes módulos para conseguir transformarlo completamente en un portal de comercio electrónico. Concretamente el módulo más utilizado para conseguirlo se denomina e-commerce (http://drupal.org/project/ecommerce), que es totalmente en gratuito en código abierto, flexible y con una gran cantidad de características, como:


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Tanto el carrito de compra, como el aspecto son configurables desde un punto de vista de interfaz.
- Permite la configuración de impuestos, descuentos y diferentes tasas adicionales.
- Permite la gestión del inventario.
- Muestra diferentes informes y resúmenes de venta.
- Diferentes plugins de pago y envio (PayPal, Authorize.net, etc.).
- Historial de compras para clientes.
- Etc.

Actualmente, el único problema que tiene este módulo es que sólo existe una versión estable para Drupal 5.0.x, la versión para Drupal 6.0.x esta en desarrollo.

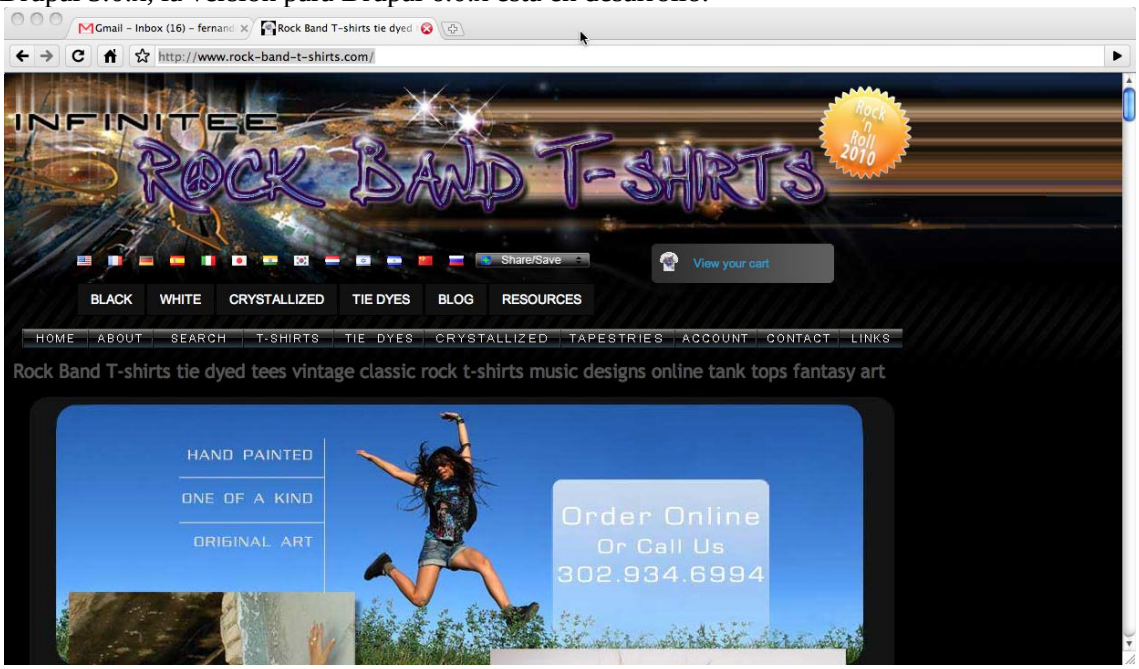



**Figura 61 - Comercio electrónico construido a partir de Drupal**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 5 Introducción a la edición de páginas Web con Dreamweaver

Dreamweaver es un software fácil para crear páginas web. Sus funciones de edición visual permiten agregar rápidamente diseño y funcionalidad a las páginas, sin la necesidad de programar manualmente el código HTML.
Se puede crear tablas, editar marcos, trabajar con capas, insertar comportamientos JavaScript, etc., de una forma muy sencilla y visual.
Además incluye un software de cliente FTP completo, permitiendo entre otras cosas trabajar con mapas visuales de los sitios web, actualizando el sitio web en el servidor sin salir del programa.

## 5.1 Primera página con Dreamweaver

- Abrir Dreamweaver
- Selecciona la Categoría Página básica, HTML

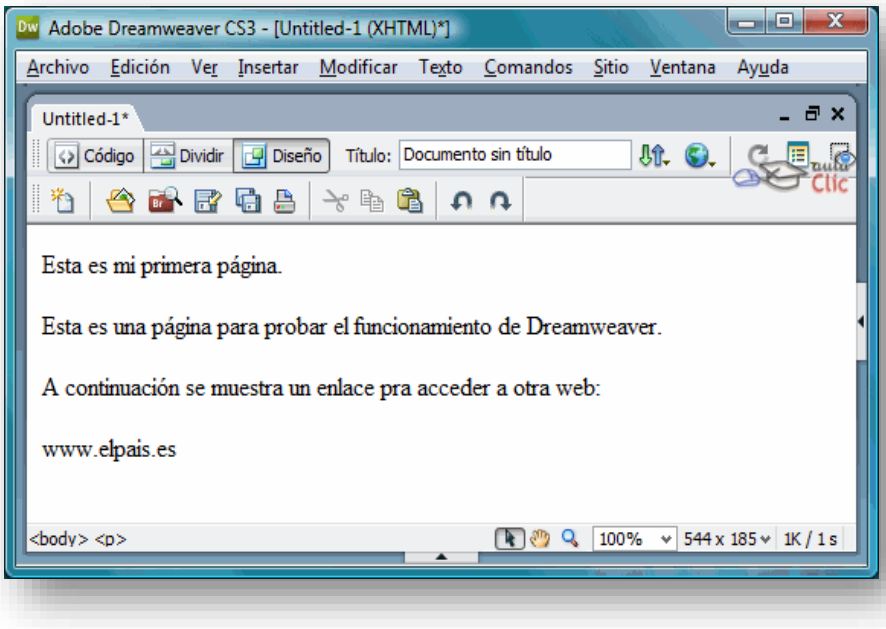



**Figura 62 - Pantalla edición Dreamweaver**

- Modificar el título y el color de fondo del documento.
  - Elegir menú Modificar y la opción Propiedades de la página


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

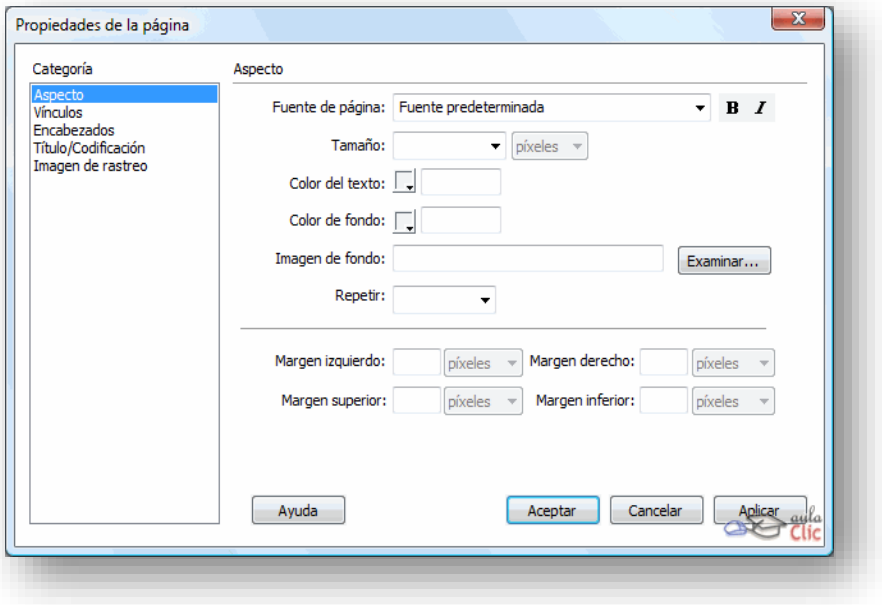



**Figura 63 - Pantalla propiedades de la página**

- Modificar el título y el color de fondo del documento.
  - Elegir menú Modificar y la opción Propiedades de la página
  - Cambia el Título por "**Mi primera página"**.
  - En Color de Fondo escribe **#CCCCFF**, de este modo el fondo del documento pasará a ser de color azul.
  - Pulsa sobre el botón Aceptar.
- Insertar una imagen. Para ello debe encontrarse en nuestro disco duro, en un cd-rom o en un disquete.
- Muchas de las imágenes que vemos en Internet se pueden copiar al disco duro. Al hacer clic con el botón derecho del ratón sobre la imagen que queremos traer de Internet, se abre una ventana con una opción similar a Guardar imagen como... (dependiendo del navegador que utilices) que permite grabar dicha imagen en un fichero de nuestro disco duro.
- Menú Insertar, a la opción Imagen

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

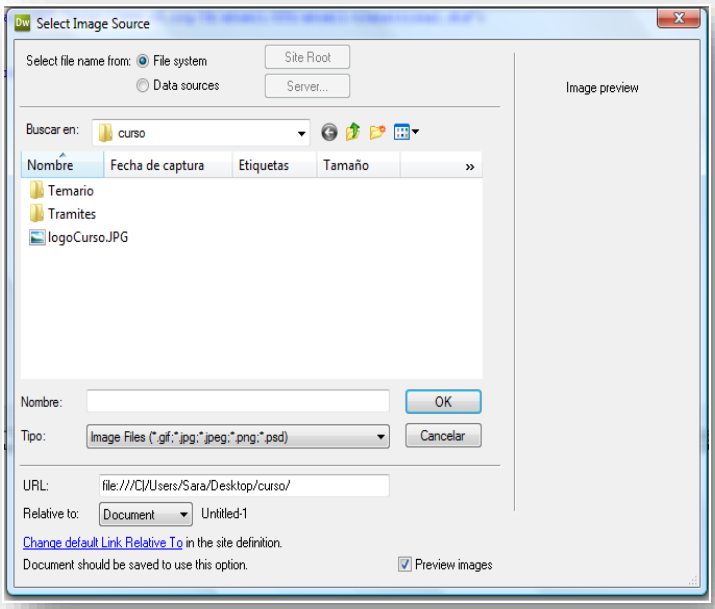



**Figura 64 - Pantalla de inserción de imagenes**

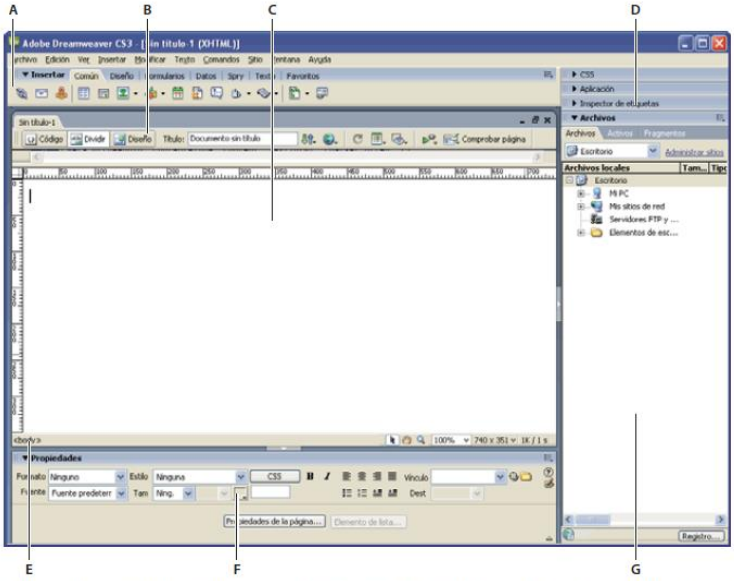
## 5.2 El entorno de aplicación

Al ejecutar Dreamweaver nos encontramos con la siguiente pantalla.


**Figura 65 - Pantalla de bienvenida**

La ventana de bienvenida permite abrir un documento reciente o crear un documento nuevo. Desde la pantalla de bienvenida, también se puede profundizar en los conocimientos sobre Dreamweaver mediante una visita guiada o un tutorial del producto.
Desde esta ventana pasaríamos a la ventana principal del programa


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc




*A. Barra Insertar B. Barra de herramientas Documento C. Ventana de documento D. Grupos de paneles E. Selector de etiquetas F. Inspector de propiedades G. Panel Archivos*

**Figura 66 - Ventana principal**

Las secciones que se pueden distinguir en esta ventana son:

- **Barra Insertar:** Contiene botones para la inserción de diversos tipos de objeto, como imágenes, tablas en un documento. Cada objeto es un fragmento de código HTML que le permite establecer diversos atributos al insertarlo. Por ejemplo, puede insertar una tabla haciendo clic en el botón Tabla de la barra Insertar. Si lo prefiere, puede insertar objetos utilizando el menú Insertar en lugar de la barra Insertar.
- **Barra de herramientas Documento** Contiene botones que proporcionan opciones para diferentes vistas de la ventana de documento (como la vista Diseño y la vista Código), diversas opciones de visualización y algunas operaciones comunes como la obtención de una vista previa en un navegador.
- **Barra de herramientas Estándar** (No semuestra en el diseño de espacio de trabajo predeterminado.)Contiene botones para las operaciones más habituales de los menú sArchivo y Edición:Nuevo,Abrir,Guardar,Guardar todo,Cortar,Copiar, Pegar, Deshacer y Rehacer. Para mostrar la barra de herramientas Estándar, seleccione Ver > Barras de herramientas > Estándar.
- **Barra de herramientas Codificación** (Sólo se muestra en la vista Código.) Contiene botones que le permiten realizar numerosas operaciones de codificación estándar.
- **Ventana de documento** Muestra el documento actual mientras lo está creando y editando.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Inspector de propiedades** Le permite ver y cambiar diversas propiedades del objeto o texto seleccionado. Cada tipo de objeto tiene diferentes propiedades. El inspector de propiedades no está ampliado de forma predeterminada en el diseño del espacio de trabajo del codificador.
- **Selector de etiquetas** Situado en la barra de estado de la parte inferior de la ventana de documento. Muestra la jerarquía de etiquetas que rodea a la selección actual.Haga clic en cualquier etiqueta de la jerarquía para seleccionar la etiqueta y todo su contenido.
- **Grupos de paneles** Conjuntos de paneles relacionados agrupados bajo un encabezado común. Para ampliar un grupo de paneles, haga clic en la flecha de ampliación situada a la izquierda del nombre del grupo; para desacoplar un grupo de paneles, arrastre el punto de sujeción situado en el borde izquierdo de la barra de título del grupo.
- **Panel Archivos** Le permite administrar los archivos y las carpetas, tanto si forman parte de un sitio de Dreamweaver como si se encuentran en un servidor remoto. El panel Archivos también proporciona acceso a todos los archivos del disco local, como ocurre en el Explorador de Windows (Windows) o en el Finder (Macintosh).

Además en Dreamweaver se distinguen tres vistas:

- **Vista Diseño:** Un entorno para el diseño visual de la página, la edición visual y el desarrollo rápido de aplicaciones. En esta vista, Dreamweaver muestra una representación visual del documento completamente editable, similar a la que aparecería en un navegador. Puedes configurar la vista Diseño para que muestre el contenido dinámico mientras trabaja en el documento.

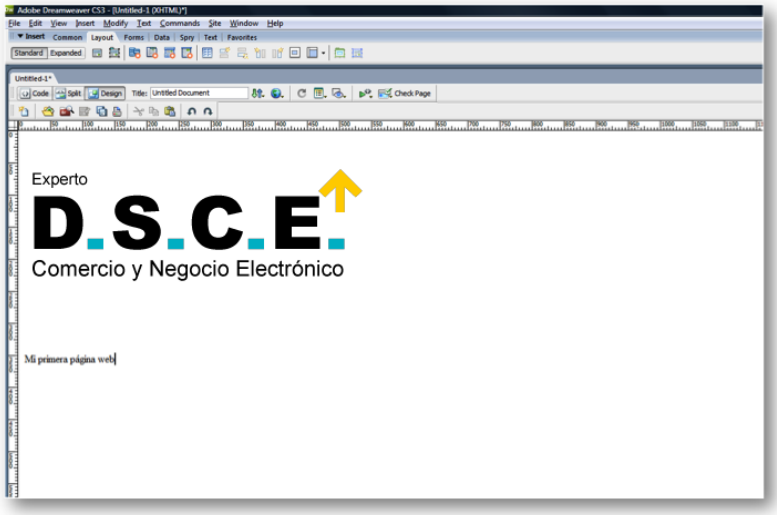



**Figura 67 - Vista Diseño**

- **Vista Código:** Un entorno de codificación manual para escribir y editar código HTML, JavaScript, código de lenguaje de servidor, como por ejemplo PHP o ColdFusion Markup Language (CFML), y otros tipos de código.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

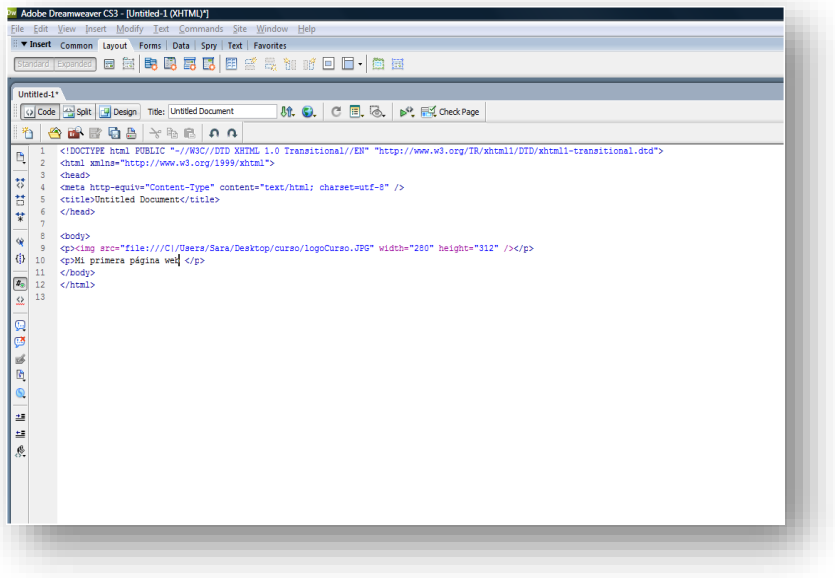



**Figura 68 - Vista Código**

- **Vistas Código y Diseño:** Permite ver las dos vistas, Código y Diseño, para el mismo documento en una sola ventana.

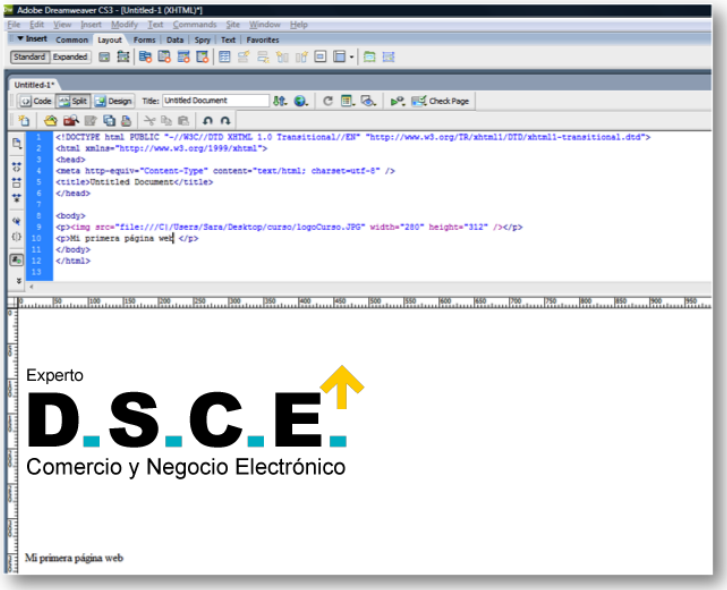



**Figura 69 - Vista Código y Diseño**

La barra de herramientas Documento tiene la forma


*A. Mostrar vista de código B. Mostrar vistas de código y diseño C. Mostrar vista de diseño D. Título del documento E. Administración de archivos F. Vista previa/Depurar en explorador G. Actualizar vista de diseño H. Ver opciones I. Ayudas visuales J. Validar formato K. Comprobar compatibilidad con navegadores*

**Figura 70 - Herramientas Documento**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Por su parte la barra de Insertar tiene las siguientes opciones:

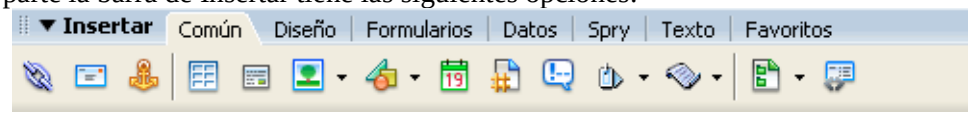



**Figura 71 - Barra Insertar**

- **Categoría Común** Permite crear e insertar los objetos que se utilizan con más frecuencia, como las imágenes y las tablas.
- **Categoría Diseño** Permite insertar tablas, etiquetas div, marcos y widgets de Spry.
- **Categoría Formularios** Contiene botones que permiten crear formularios e insertar elementos de formulario, incluidos widgets de validación de Spry.
- **Categoría Datos** Permite insertar objetos de datos de Spry y otros elementos dinámicos, como juegos de registros, regiones repetidas y grabar formularios de inserción y actualización.
- **Categoría Spry** Contiene botones para crear páginas de Spry, incluidos objetos de datos y widgets de Spry.
- **Categoría Texto** Permite insertar diversas etiquetas de formato de texto y listas, como b, em, p, h1 y ul.
- **Categoría Favoritos** Permite agrupar y organizar los botones de la barra Insertar que utiliza conmás frecuencia en un lugar común.
- **Categorías de código de servidor** Sólo están disponibles para las páginas que emplean un lenguaje de servidor determinado, como ASP, ASP.NET, CFML Basic, CFML Flow, CFML Advanced, JSP y PHP. Cada una de estas categorías contiene objetos de código de servidor que pueden insertarse en la vista Código

También tenemos la barra de herramientas de codificación que permite realizar operaciones de codificación estándar, como ampliar y contraer las selecciones de código, resaltar código no válido.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

**Figura 72 - Barra de herramientas de codificación**

Por su parte la barra de inspector de propiedades permite examinar y editar las propiedades más comunes del elemento de página seleccionado actualmente, como texto o un objeto insertado.

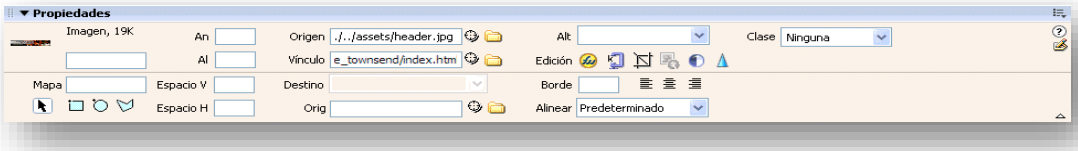



**Figura 73 - Barra de propiedades**

Por último, a través del menú Ayuda puedes ir a varias opciones, veamos algunas de ellas:

- Acceder al cuadro de diálogo de ayuda muy similar al de Windows donde puedes buscar por temas, por índice o por contenido, si seleccionas la opción Ayuda de Dreamweaver o simplemente pulsando F1.
- Acceder a tutoriales en internet (en la opción Recursos de ayusa en línea).
- Abrir el panel Referencia en el que encontrarás la sintaxis y descripción de las etiquetas HTML.
- Acceder al Centro de servicio técnico de Dreamweaver en la web.
- Acceder al Foros en línea de Adobe.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 5.3 Configuración de un sitio local

Un sitio web es un conjunto de archivos y carpetas, relacionados entre sí, con un diseño similar o un objetivo común. Es necesario diseñar y planificar el sitio web antes de crear las páginas que va a contener.
La forma habitual de crear un sitio consiste en crear una carpeta en el disco local. Los documentos HTML normalmente se crean dentro de dicha carpeta, mientras que para contener las imágenes, las animaciones, las hojas de estilo, etc., se deben crear nuevas carpetas dentro de ésta, con el objetivo de tener una mejor organización de los archivos a la hora de trabajar. Esto es lo que se conoce como sitio local.
Después se podrán copiar los archivos en un servidor web, en el denominado sitio remoto, lo que equivale a publicar el sitio, de modo que la gente podrá verlo en Internet.
La organización de los archivos en un sitio permite administrar y compartir archivos, mantener los vínculos de forma automática, utilizar FTP para cargar el sitio local en el servidor, etc.
Es conveniente que la página inicial del sitio tenga el nombre *index.htm* o *index.html*, ya que los navegadores buscan una página con ese nombre cuando se intenta acceder a una URL genérica
Una vez creadas las carpetas que formarán un sitio local, ya es posible definir el sitio en Dreamweaver.
Para ello hay que dirigirse al menú *Sitio*, a la opción *Administrar sitios...*.

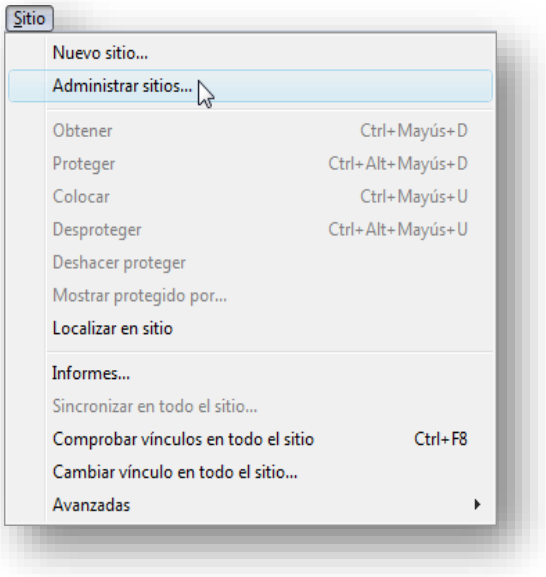



**Figura 74 - Menú *Sitio***

Es necesario recordar que a través del panel *Archivos*, pestaña *Archivos*, se puede acceder a cada uno de los sitios creados y a la opción *Administrar sitio*.
En el caso de haber seleccionado la opción Administrar sitios, aparece una ventana que contiene la lista de sitios locales definidos con anterioridad. Pueden existir varios sitios locales en un mismo ordenador.
Tanto si se elige la opción *Nuevo...*, como si se elige la opción *Editar...*, se mostrará la misma ventana en la que definir las características del sitio. Las características del sitio se agrupan en diferentes categorías que aparecen en la parte izquierda. Para visualizar las características de una categoría basta con seleccionarla de la lista haciendo clic en ella.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

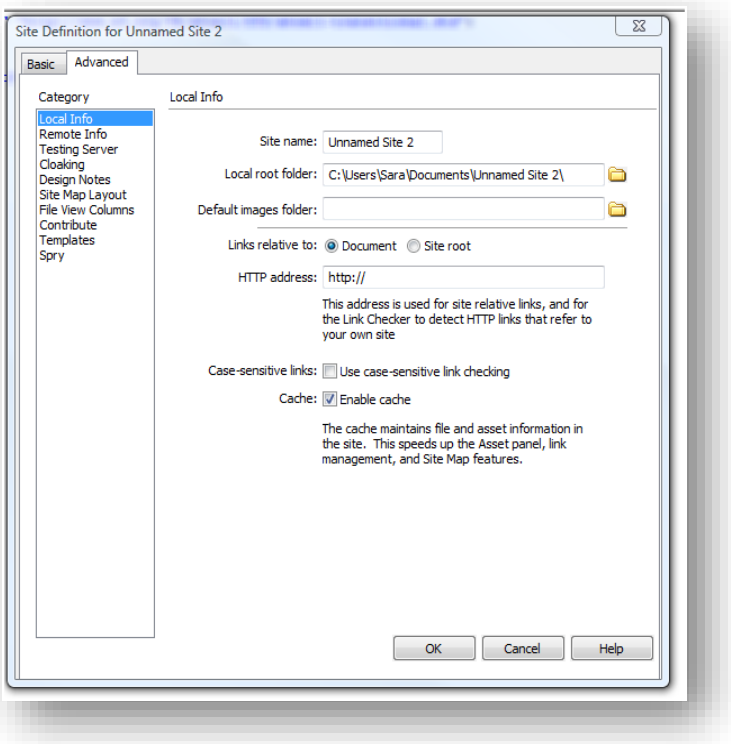



**Figura 75 - Pantalla Configuración de un sitio local**

Para la categoría *Datos locales* se debe definir el Nombre del sitio y la Carpeta raíz local, que es en la que se encuentra el sitio dentro del disco duro local.
Después, si se desea, a través de la categoría *Mapa de diseño del sitio* puede definirse la página principal del sitio, de la que colgarán el resto de documentos HTML dentro del sitio, si en la carpeta raíz del sitio existe una página con el nombre index.htm, Dreamweaver la cogerá por defecto.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

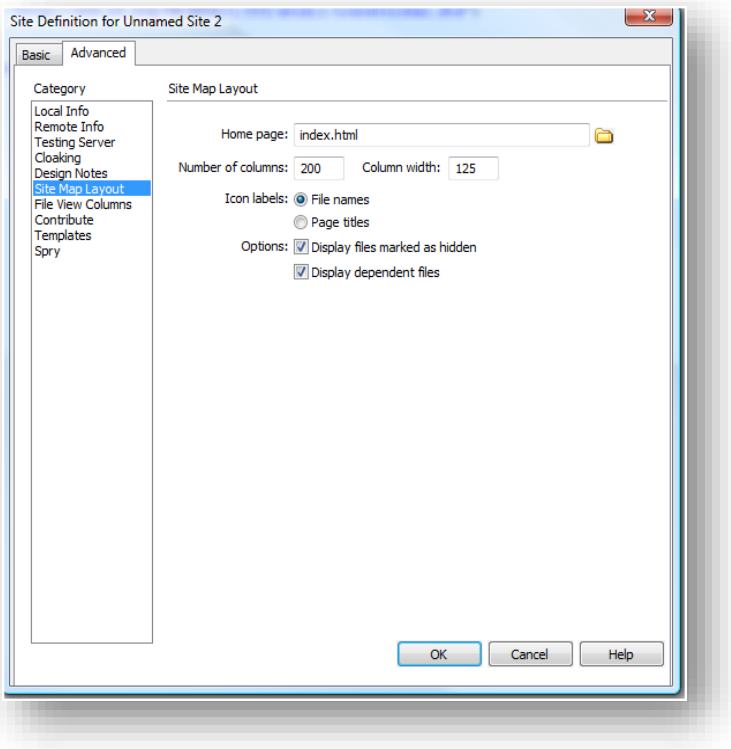



**Figura 76 - Categoría *Mapa de diseño del sitio***

Es posible visualizar un sitio en el panel Archivos o en una ventana. El panel Archivos se puede abrir a través del menú Ventana, opción Archivos. También pulsando F8.
Para cambiar de una vista a otra hay que pulsar sobre el botón que aparece en la parte superior del panel y de la ventana. Al mismo tiempo, es posible visualizar el sitio local, el sitio remoto, el servidor remoto de prueba o el mapa del sitio.
En esta imagen se visualizan el mapa del sitio y el sitio local.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

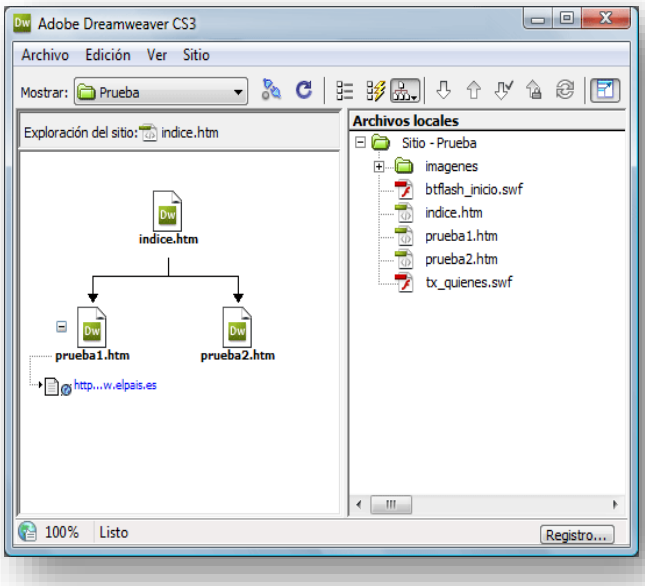



**Figura 77 - Mapa del sitio**

En este caso los documentos prueba1.htm y prueba2.htm cuelgan del documento indice.htm, ya que éste ha sido definido como página principal del sitio y contiene vínculos hacia los otros dos.
Si los archivos se mueven de una carpeta a otra, o se cambian de nombre, desde fuera de Dreamweaver, las páginas no se mostrarán correctamente: no aparecerán imágenes, no funcionarán enlaces, etc.
En cambio, si estas modificaciones se hacen desde dentro de Dreamweaver, a través del sitio, es posible actualizar automáticamente las páginas para que si hacen referencia a un objeto que ha cambiado, después no se produzcan problemas al visualizarlas.
Al modificar algún objeto que es referenciado por algún otro documento, se muestra una ventana similar a esta, que indica los documentos que hacen referencia a dicho objeto, y que pueden ser actualizados para que no se produzcan los problemas antes mencionados. Simplemente hay que pulsar sobre el botón Actualizar.

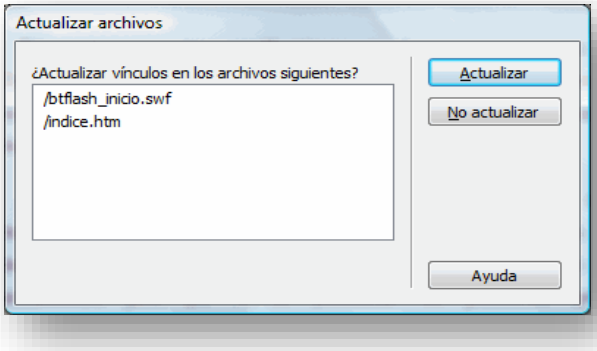



**Figura 78 - Ventana de actualización de archivos**

Cuando se quiera visualizar el sitio en Internet (al terminar el curso o para hacer pruebas) deberás subir los archivos al sitio que hayas contratado o conseguido gratuitamente.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Algunos sitios gratuitos sólo permiten que se suban archivos vía Web, por lo que será necesario subir uno a uno cada archivo y creando las carpetas. Este proceso puede hacerse muy largo y hay que tener mucho cuidado en mantener la estructura tal y como se encuentra en la carpeta del disco duro. Sino los enlaces no funcionarán correctamente y es posible que algunas imágenes no se muestren.

## 5.4 Propiedades del documento

Puede definirse el formato de cada una de las páginas a través del cuadro de diálogo *Propiedades de la página*.

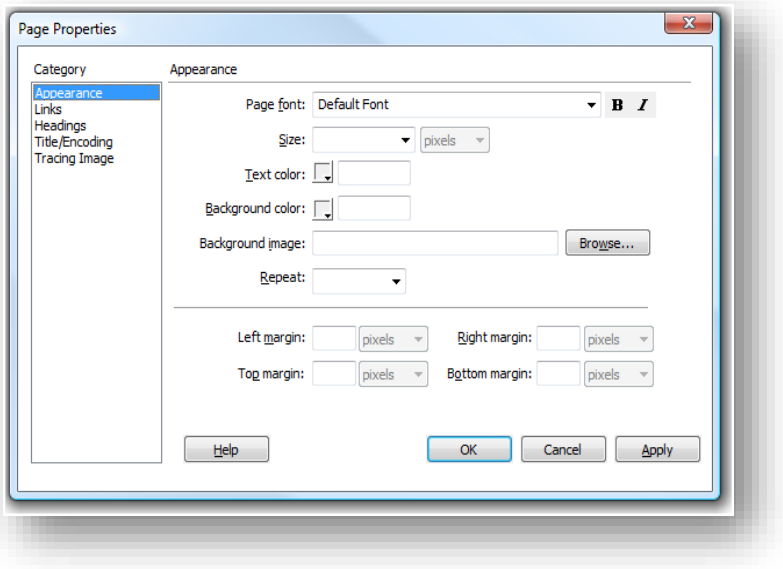



**Figura 79 - Pantalla *Propiedades de Página***

Este cuadro se puede abrir de tres modos diferentes:

- Pulsar la combinación de teclas Ctrl+J.
- Hacer clic sobre el menú Modificar y elegir la opción Propiedades de la página.
- Hacer clic con el botón derecho del ratón sobre el fondo de la página. Aparecerá al final del menú contextual la opción Propiedades de la página. Se abrirá el cuadro de diálogo que ves.

**Categoría Aspecto**

- **Imagen de fondo**: permite especificar una imagen de fondo para el documento. Dicha imagen se muestra en mosaico. Es importante al elegir una imagen de fondo tener en cuenta que según los colores de la imagen será necesario establecer unos u otros colores para el texto, así como que no es conveniente tener un gif animado como fondo.
- **Color de fondo**: permite especificar un color de fondo para el documento, pero dicho color solo se mostrará en el caso de no haber establecido ninguna imagen de fondo.
- **Tamaño**: permite definir el tamaño de la letra.
- **Color del texto**: es el color de la fuente.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Márgenes**: permiten establecer márgenes en el documento. Los márgenes izquierdo y superior solo funcionan en Microsoft Internet Explorer, mientras que el ancho y el alto de márgen solo funcionan para Netscape Navigator. Ninguno de estos márgenes aparece en la ventana de documento de Dreamweaver, solo se mostrarán en los navegadores.

**Categoría Título/Codificación**

- **Título**: es el título del documento, que aparecerá en la barra de título del navegador y de la ventana de documento de Dreamweaver

**Categoría Vínculos**

- **Color de vínculo**: es el color de los vínculos, que ayuda al usuario a distinguir entre el texto normal y los vínculos que sirven de enlace a otras páginas.
- **Vínculos visitados**: es el color de los vínculos visitados, que permite distinguir al usuario si unos vínculos ya han sido visitados o no.
- **Vínculos activos**: es el color de los vínculos activos.
- **Estilo subrayado**: por defecto, cuando tenemos un texto con un vínculo asociado, el texto aparece subrayado, con esta opción podemos elegir otro tipo de estilo por ejemplo para que no aparezca subrayado.

**Categoría Imagen de rastreo**

- **Imagen de rastreo**: permite establecer una imagen como fondo del documento, pero que sólo se mostrará en la ventana de documento de Dreamweaver, y nunca en un navegador. Dicha imagen se utiliza como plantilla gráfica sobre la que crear el documento.
- **Transparencia**: permite establecer la opacidad de la imagen de rastreo.

Si las páginas tienen muchos elementos (imágenes, tablas, etc.) y son muy grandes, posiblemente tarden mucho tiempo en mostrarse totalmente en el navegador. Esto no es nada recomendable, ya que los usuarios pueden perder la paciencia, y no visitar más nuestra página.
Siempre hay que intentar que el tiempo de descarga en el navegador no sea muy elevado.bDreamweaver permite calcular automáticamente el tiempo de descarga de las páginas. Para ello hay que dirigirse al menú *Edición,* a la opción *Preferencias.* En esta nueva ventana lo primero que hay que hacer es seleccionar una Categoría. En este caso nos interesa la de *Barra de estado*.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

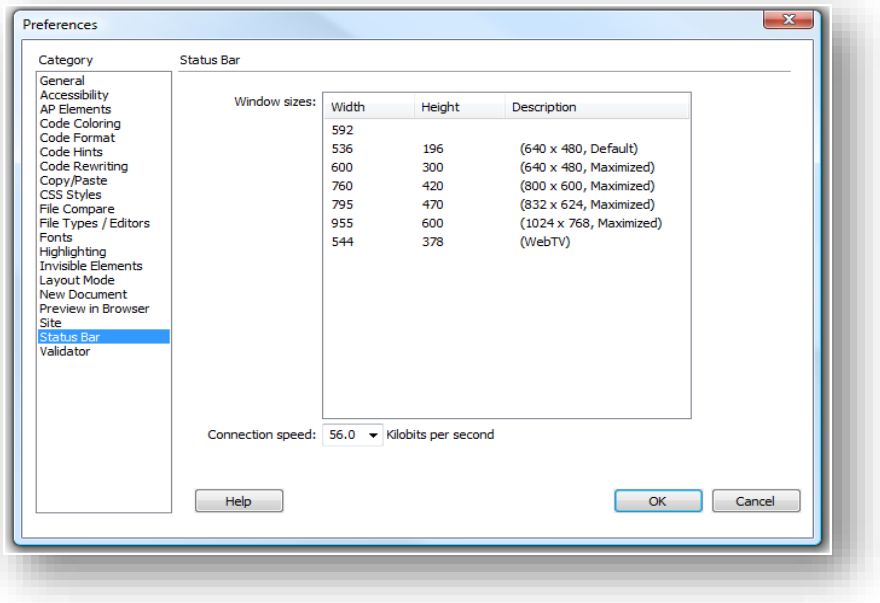



**Figura 80 - Ventana de preferencias**

En ella habrá que establecer una Velocidad de conexión. En España 56,0 es la velocidad de los módems normales, pero podemos elegir otra diferente, por ejemplo si pensamos que la mayoría de nuestros posibles clientes tienen conexiones rápidas.
Por ejemplo, en la página aparecen estos datos entre el tamaño de la ventana de documento y el panel de propiedades, representados por 1K/1 s.

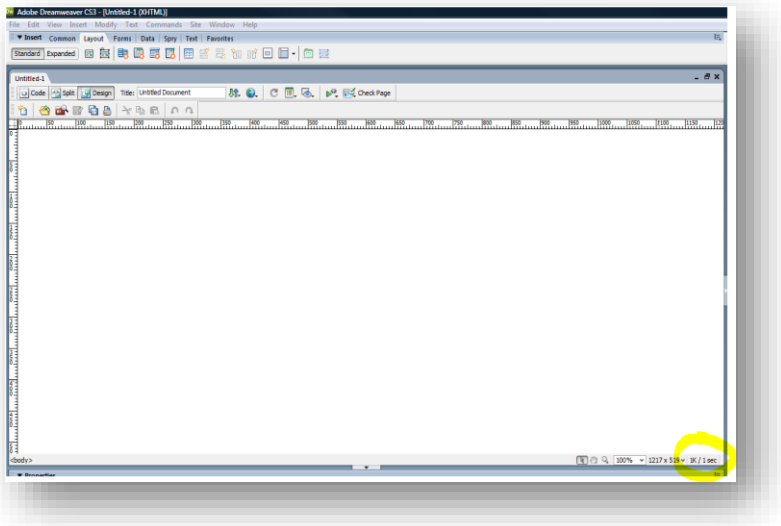


**Figura 81 - Velocidad de conexión**



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 5.5 Elementos HTML en Dreamweaver

### 5.5.1 Texto

Las características del texto seleccionado pueden ser definidas a través del menú Texto, y a través del inspector de propiedades.

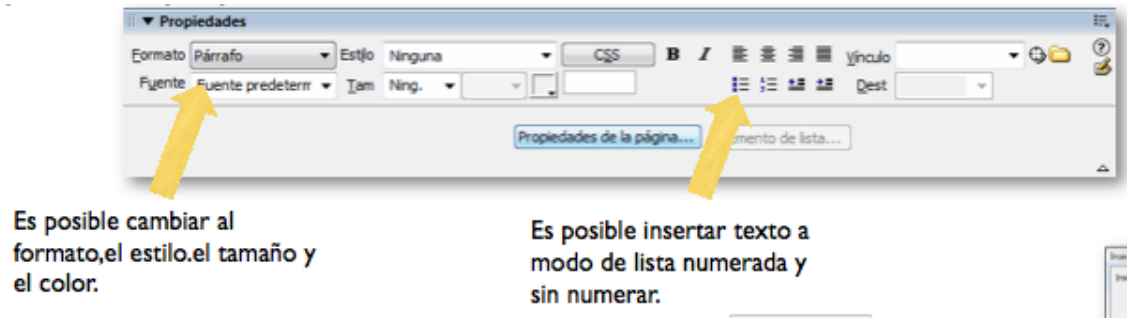



**Figura 82 - Menú *Texto***

En el menú *Insertar →HTML* verás el desplegable de la imagen de la derecha para introducir caracteres especiales

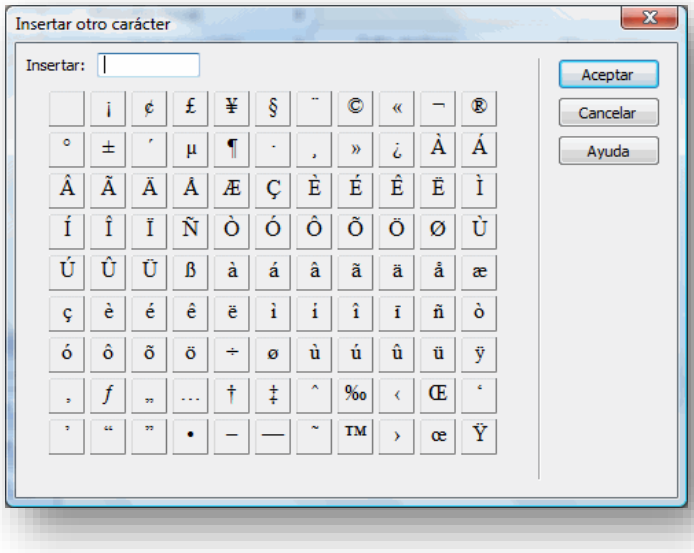



**Figura 83 - Insertar símbolo**

Con Dreamweaver, las características que apliquemos a un texto crearán automáticamente estilos CSS (*Cascading Style Sheets)* que se incrustarán en el documento actual y se aplicarán únicamente sobre él.

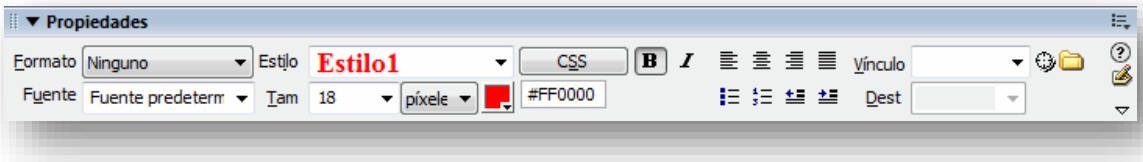



**Figura 84 - Definición de estilo**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Para crear un Estilo CSS personalizado:

1. En el documento, se selecciona el texto al que se desea aplicar características concretas.
2. En el inspector de propiedades se modifican todas las propiedades de formato de texto, se establecen los atributos de la fuente y del párrafo que queramos.
3. Automáticamente Dreamweaver creará un nuevo estilo con el nombre Estilo1 o Estilo2 o Estilo3,... según los nombres de los estilos ya creados.
4. Aparecerá el nombre Estilo1 en el cuadro Estilo del panel Propiedades.
5. Para darle el nombre que queramos al estilo, desplegamos la lista de estilos y seleccionamos la opción Cambiar nombre...
6. Cada nuevo estilo que se crea, se añade a la lista de estilos, y se puede acceder también de una manera rápida a través del menú Texto, opción Estilos CSS.
7. En el Panel CSS se puede verificar que se ha añadido automáticamente el nuevo estilo.

### 5.5.2 Hiperenlace

Enlace, que al ser pulsado lleva de una página o archivo a otra página o archivo. Hay varios tipos

- Referencia absoluta: http://www.misitio.com/pagina/pagina1.html
- Referencia relativa: /pagina/pagina1.html
- Anclaje: /pagina/pagina1.html#miancla. Conduce a un punto dentro de un documento, ya sea dentro del actual o de otro diferente.

Para crear un hipervínculo accedemos al menú *Insertar* en la opción *Hipervínculo*

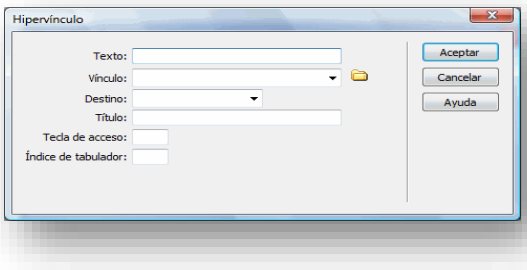



**Figura 85 - Ventana para crear Hipervínculo**

Por otro lado para crear ancles, en el mismo menú, seleccionamos la opción de Anclaje con nombre.

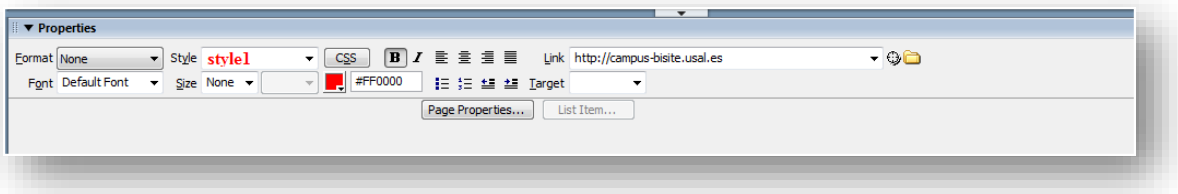



**Figura 86 - Ventana propiedades hipervínculo**

Las propiedades que se pueden definir son:

- **Texto:** es el texto que mostrará el enlace



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Vínculo:** es la página a la que irá redirigida el enlace, si se trata de un enlace externo deberás escribirla empezando siempre por HTTP://. Haz clic sobre el icono de carpeta para buscar los archivos que existan dentro del sitio. Por defecto dramweaver te creará un enlace relativo al documento. Si quieres que los enlaces sean relativos al sitio visita este avanzado .
- **Destino**: la página donde se abrirá la página, este campo se explica en el siguiente apartado.
- **Título**: se trata de la ayuda contextual del vínculo, es equivalente al atributo ALT de las imágenes.
- **Tecla de acceso**: atributo que facilita la accesibilidad a las páginas, habilita el acceso al enlace mediante la pulsación de la tecla Alt más la tecla de acceso indicada.
- **Índice de tabulador**: Como habrás podido observar puedes saltar a través de los enlaces pulsando la tecla Tabulador. En este campo podrás establecer un índice indicando la prioridad del enlace y así configurar el modo en el que actuará el Tabulador es sus diferentes saltos
- Destinos
    - _blank: Abre el documento vinculado en una ventana nueva del navegador.
    - _parent: Abre el documento vinculado en la ventana del marco que contiene el vínculo o en el conjunto de marcos padre.
    - _self: Es la opción predeterminada. Abre el documento vinculado en el mismo marco o ventana que el vínculo.
    - _top: Abre el documento vinculado en la ventana completa del navegador.

También es posible a través del menú *Insertar→Vínculo de correo electrónico*, hacer un enlace a una dirección de correo electrónico [36-28].

### 5.5.3 Imágenes

Para insertar una imagen en una web utilizamos menú *Insertar→Imagen*
La ruta en la que se encuentra la imagen aparecerá representada de una u otra forma en el campo URL de la ventana y en el campo Origen del inspector de propiedades, dependiendo de si ha sido insertada como relativa a la carpeta raíz del sitio o relativa al documento.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

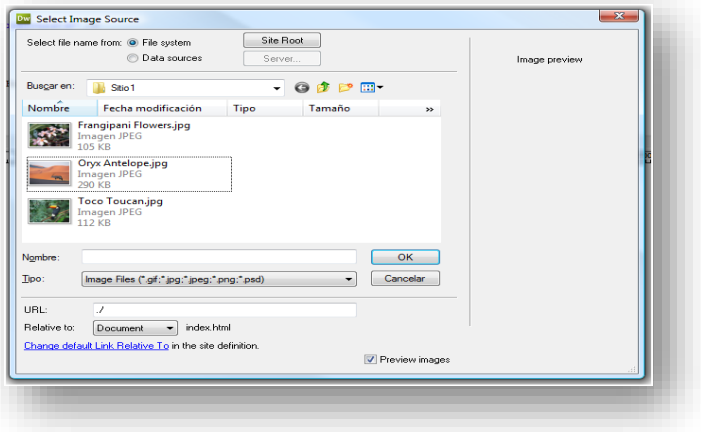



**Figura 87 - Ventana de Inserción de Imagen**

Existen otra serie de imágenes especiales que suelen utilizarse para crear menús, como pueden ser los botones Flash.
La inserción se realiza a través de menú *Insertar →Media →Botón Flash*. A través de *Estilo*: puede seleccionarse uno de los distintos formatos de botón que se ofrecen.

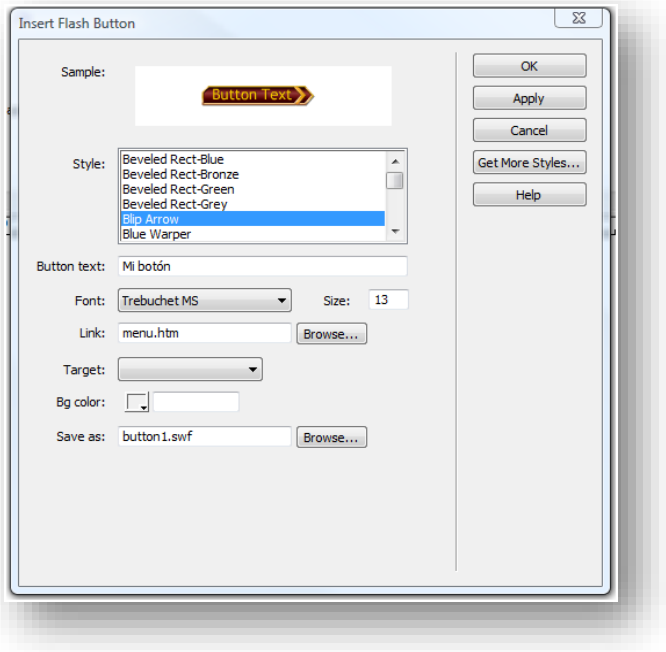



**Figura 88 - Insertar botón Flash**

Hay que especificar también el Texto que mostrará el botón (Texto del botón:), así como el nombre con el que será guardado (Guardar como:) y el vínculo asociado (Vinculo: y Destino:).
Es preferible guardar los botones Flash en el mismo directorio que los documentos HTML, en lugar de la carpeta destinada a almacenar imágenes, ya que de no ser así es posible que al intentar asignar un vínculo dentro del propio sitio, Dreamweaver no permita guardar el botón con ese vínculo en una ubicación diferente de la de dicho documento. Los botones deben guardarse con


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

la extensión SWF. A través del inspector de propiedades del botón Flash es posible editar de nuevo sus atributos.

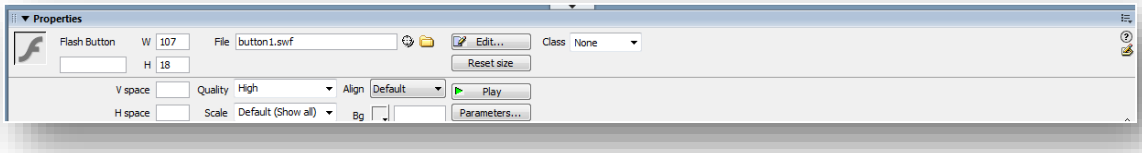



**Figura 89 - Ventana de propiedades de Flash**

### 5.5.4 Tablas

Para la inserción de tablas, se selecciona la opción *Tabla* del menú *Insertar*.

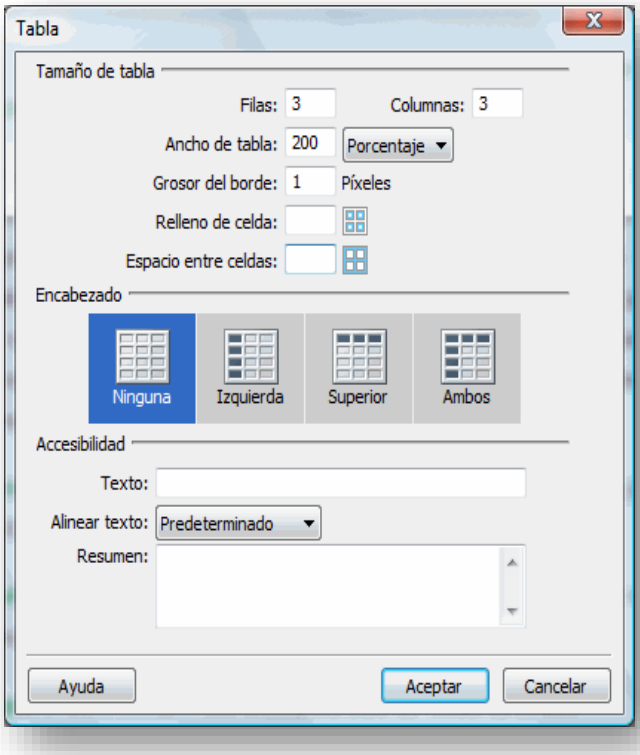



**Figura 90 - Ventana de inserción de Tabla**

Hay que especificar el número de Filas y Columnas que tendrá la tabla, así como el Ancho de la tabla .
El *Ancho* puede ser definido como Píxeles o como Porcentaje. La diferencia de uno y otro es que el ancho en Píxeles es siempre el mismo, independientemente del tamaño de la ventana del navegador en la que se visualice la página, en cambio, el ancho en Porcentaje indica el porcentaje que va a ocupar la tabla dentro de la página y se ajustará al tamaño de la ventana del navegador, esto permite que los usuarios que tengan pantallas grandes, aprovechen todo el ancho de pantalla.
*Grosor del Borde* indica el grosor del borde de la tabla en píxeles, por defecto se le asigna uno. *Relleno de celda* indica la distancia entre el contenido de las celdas y los bordes de éstas. *Espacio entre celdas* indica la distancia entre las celdas de la tabla.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

También se puede establecer si se quiere un encabezado para la tabla es recomendable utilizar encabezados en el caso de que los usuarios utilicen lectores de pantalla. Los lectores de pantalla leen los encabezados de tabla y ayudan a los usuarios de los mismos a mantener un seguimiento de la información de la tabla.
Si queremos incluir un título, lo indicamos en *Texto*, el título aparecerá fuera de la tabla.
En alinear texto indicamos dónde queremos alinear el título con respecto a la tabla. En *Resumen*, indicamos una descripción de la tabla, los lectores de pantalla leen el texto del resumen pero dicho texto no aparece en el navegador del usuario.
A la hora de trabajar con tablas Dreamweaver nos proporciona distintos modos de visualización. Del modo estándar se puede pasar a los otros modos a través del menú *Ver* opción *Modo de tabla*. Dentro de esta opción podemos elegir entre *Modo estándar*, *Modo de tablas expandidas* o *Modo de Diseño*.

- El Modo de diseño se utiliza para dibujar páginas con un diseño determinado, basándose en tablas. Utilizando esta vista, las tablas no han de insertarse obligatoriamente en líneas de texto, como hemos hecho hasta el momento, sino que es posible insertarlas en cualquier punto de la página, y Dreamweaver se encargará de rellenar el espacio vacío, para que sea posible que la tabla aparezca en esa posición.
- El Modo de tablas expandidas añade temporalmente relleno y espaciado de celda a las tablas de un documento y aumenta los bordes de las tablas para facilitar la edición. Este modo se puede utilizar para seleccionar elementos de las tablas o colocar el punto de inserción de forma precisa pero en este modo no vemos la página como quedará exactamente.

Siempre que se crea una página Web hay que tener en cuenta que va a ser visitada por usuarios con monitores de distintos tamaños y con distintas resoluciones.
Es imposible hacer que una página se vea exactamente igual en todos los monitores de los usuarios que la visitan, pero hay que intentar que su visualización sea lo más parecida posible. El tamaño de las tablas puede ser definido en píxeles o en porcentaje. Cuando el tamaño de la tabla sea definido en píxeles, la tabla se verá del mismo modo independientemente del tamaño de la ventana del navegador. En cambio, cuando el tamaño de la tabla sea definido en **porcentaje**, la tabla se ajustará al tamaño de la ventana del navegador, lo cual implica que el contenido de la tabla se descuadrará.

### 5.5.5 Marcos

Los marcos o frames sirven para distribuir mejor los datos de las páginas, ya que permiten mantener fijas algunas partes, como pueden ser el logotipo y la barra de navegación, mientras que otras sí pueden cambiar. Además de mejorar la funcionalidad, pueden mejorar también la apariencia.
Cada uno de los marcos de una página, contiene un documento HTML individual. Por ejemplo, en la imagen de la derecha puedes ver una página con dos marcos.

Para crear un marco, primero hay que abrir algún documento. Puede ser uno nuevo o uno ya existente. Después, dirigirse al menú *Insertar→HTML→Marcos*. A través de esta opción puede elegirse el tipo de marco que va a crearse.
Si pulsamos sobre Izquierda se creará un nuevo marco a la izquierda del documento actual. En este caso tendremos tres documentos: el de la izquierda, el de la derecha, y el que contiene el


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

grupo de marcos. El de la derecha es el documento que teníamos inicialmente, que está en el marco conocido como marco padre (MainFrame).
Para seleccionar los marcos del documento es necesario dirigirse al panel Marcos, que puede abrirse a través del menú Ventana. Si la opción Marcos no te aparece directamente en este menú, posiblemente esté dentro de la opción Otros. También puedes abrir el panel Marcos pulsando la combinación de teclas Mayús+F2.

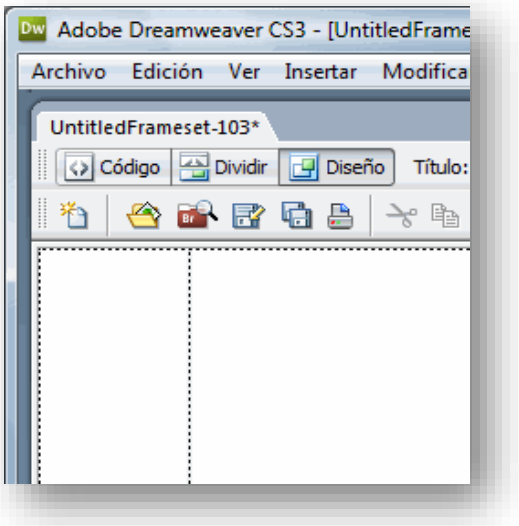



**Figura 91 - Definición de Marcos**

Todos los documentos que contienen marcos tienen que tener una página en cada uno de ellos. Es por esto que al crear algún marco, se cargan páginas nuevas por defecto en cada uno de ellos, a excepción del marco que contiene la página original.
Estas páginas nuevas pueden ser posteriormente sustituidas por otras ya existentes. Es necesario guardar la página que contiene el grupo de marcos, así como cada una de las páginas que están incluidas en sus marcos. No es conveniente guardar la primera vez los marcos con la opción *Guardar todo* , ya que podemos equivocarnos al dar los nombres a los nuevos documentos. Es preferible guardar cada documento uno por uno, a no ser que todos los marcos contengan alguna página ya existente, ya que en ese caso el único documento al que habrá que dar nombre será al que contiene el grupo de marcos. Para guardar el documento que contiene el grupo de marcos, hay que seleccionarlo previamente. Para guardar cada uno de los otros documentos, simplemente hay que situar el cursor en ellos antes de pulsar sobre *Guardar*.

Una vez seleccionado un marco a través del panel Marcos, pueden establecerse sus propiedades a través del inspector de propiedades. Cada marco tiene asignado un nombre, que puede cambiarse a través de *Nombre de marco*. El nombre no puede contener espacios en blanco. En *Origen* aparece el nombre del documento HTML que está contenido en el marco. En *Bordes* puede elegirse si aparecerá o no una línea separando el marco del resto de marcos. En el caso de que se muestre el borde, se puede especificar un color para éste a través de Color borde. *Desplaz.* indica si aparecerán o no las barras de desplazamiento cuando el documento del marco no pueda visualizarse completamente. Si la opción *Mismo tamaño* está activa, indica que los usuarios no podrán variar las medidas del marco desde el navegador. El *Ancho del margen* y el *Alto del margen* indican la separación que habrá entre el contenido del marco y sus bordes izquierdo-derecho y superior-inferior.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

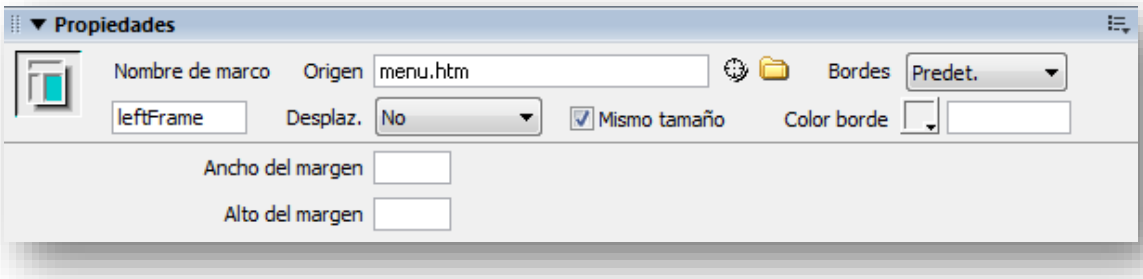



**Figura 92 - Propiedades Marcos**

Si lo que seleccionado es todo el conjunto de marcos (la página de marcos), el inspector de propiedades es algo diferente. En *Bordes* puede elegirse si aparecerá o no una línea separando los marcos entre sí y puede especificarse un color para este a través de *Color del borde*. También es posible establecer un grosor para el borde a través de *Ancho*. El campo *Columna* (o *Fila* dependiendo del marco elegido en Selección Fila Col.) sirve para especificar el tamaño del marco, que puede ser en Píxeles, Porcentaje (de la ventana) o Relativo (proporcional al resto de marcos).
Normalmente se utilizan dos marcos, uno de ellos con tamaño en Píxeles, que será el que contenga la barra de navegación, y el otro marco con tamaño Relativo, para que se ajuste a la ventana del navegante.

### 5.5.6 Formularios

Los formularios se utilizan para recoger datos de los usuarios, nos pueden servir para realizar un pedido en una tienda virtual, crear una encuesta, conocer las opiniones de los usuarios, recibir preguntas, etc.
Una vez el usuario rellena los datos y pulsa el botón para enviar el formulario se arrancará un programa que recibirá los datos y hará el tratamiento correspondiente.
Los elementos de formulario pueden insertarse en una página a través del menú *Insertar*, opción *Formulario*. A través de esta opción se puede acceder a la lista de todos los objetos de formulario que pueden ser insertados en la página.
Los elementos que se pueden incluir en un campo de texto son:

- **Campo de texto y Área de texto:** Permiten introducir texto. El Campo de texto solo permite al usuario escribir una línea, mientras que el Área de texto permite escribir varias.
- **Botón:** Es el botón tradicional de Windows. El botón puede tener asignadas tres opciones: Enviar formulario, Restablecer formulario (borrar todos los campos del formulario), o Ninguna (para poder asignarle un comportamiento diferente de los dos anteriores).
- **Casilla de verificación:** Es un cuadrito que se puede activar o desactivar
- **Botón de opción:** Es un pequeño botón redondo que puede activarse o desactivarse. Si hay varios del mismo formulario, sólo puede haber uno activado. Cuando se activa uno, automáticamente se desactivan los demás.
- **Lista/Menú:** Una lista o menú es un elemento de formulario que lleva asociada una lista de opciones.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Etiqueta:** Se utiliza para ponerle nombres al resto de elementos de formulario, para que el usuario pueda saber qué datos ha de introducir en cada uno de ellos

### 5.5.7 Multimedia

**Películas Flash**

Son animaciones, que al igual que los botones y el texto Flash, tienen la extensión SWF. Es frecuente verlas en las páginas iniciales de los sitios web, a modo de presentación hacia los usuarios, aunque se pueden utilizar para realizar cualquier tipo de animación.

Estas películas pueden crearse mediante el programa Flash de Macromedía, y necesitan que el usuario tenga instalado el plug-in para poder ser visualizadas.

Las películas Flash pueden insertarse en una página a través del menú *Insertar*, *Media*, opción *Flash*, o pulsando Ctrl+Alt+F. También pueden insertarse pulsando sobre la opción Flash que aparece en la pestaña *Común* del panel *Insertar*, botón *Media.*

El inspector de propiedades de las películas *Flash* es prácticamente igual que el de los botones y el texto Flash, pero existen dos opciones nuevas que hacen referencia a la visualización de la película. La opción *Bucle* indica que al finalizar la película, ésta volverá a comenzar desde el principio.

La otra es la opción *Rep. autom.* (reproducción automática), que al estar marcada indica que nada más cargarse la página comenzará a reproducirse la película Flash. Si esta opción no estuviera marcada, se mostraría únicamente el primer fotograma de la película. Interesa desmarcar esta opción cuando se desea que la reproducción sea activada por algún comportamiento.

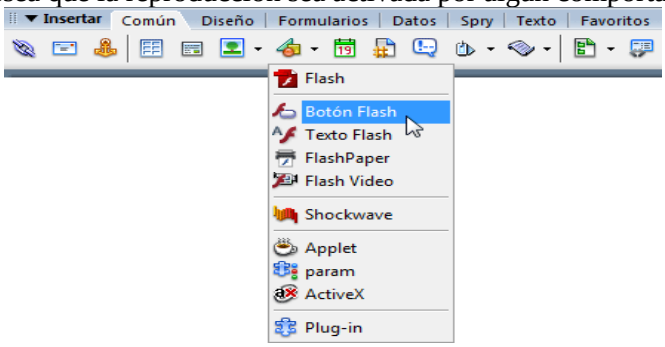



**Figura 93 - Inserción elementos Multimedia**

**Sonido**

No es muy habitual incluir sonido en una página web, ya que algunos usuarios suelen estar escuchando su propia música cuando navegan en Internet, por lo que el escuchar también sonido en cada página que se visita puede resultar algo molesto.

A pesar de ello, el incluir un sonido agradable, apropiado al contenido de la página, puede hacerla más atractiva. Muchas de las páginas que contienen sonido suelen ofrecer la posibilidad de activarlo o desactivarlo, para que aquellos usuarios que no deseen escuchar el sonido de la página puedan desactivarlo.

Los formatos de sonido más habituales en Internet son, fundamentalmente, el MP3, el WAV y en algunas ocasiones el MIDI, aunque existen otros formatos diferentes que también pueden utilizarse. Lo ideal es incluir algún archivo de audio que no ocupe mucho espacio, y que no por ello sea de mala calidad.

Para insertar un archivo de audio en un documento tienes que dirigirte al menú *Insertar*, *Medía*, opción *Plug-in.*

**Video**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

En ocasiones puede interesar incluir algún vídeo en una página web, pero hay que tener en cuenta que los vídeos suelen ocupar mucho espacio en disco, y por lo tanto, precisan de mucho tiempo para descargarse. Los formatos de vídeo que suelen utilizarse en Internet son el AVI, el MPEG y el MOV.

Para insertar un archivo de vídeo en un documento tienes que dirigirte al menú *Insertar, Medía,* opción *Plug-in*.

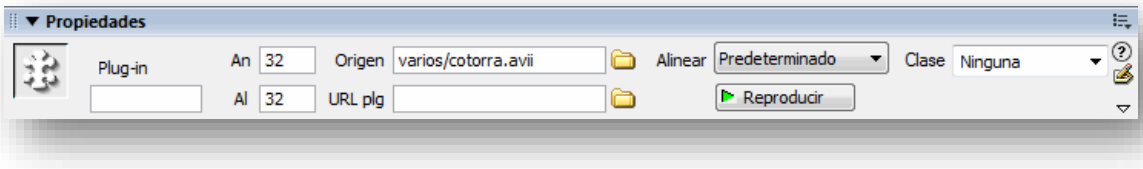



**Figura 94 - Propiedad Video**



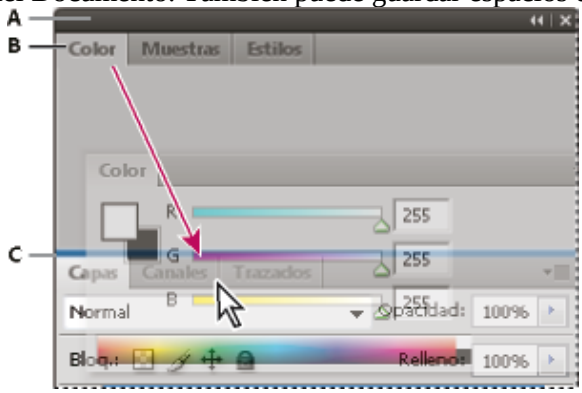
ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 6 Introducción a la edición de imágenes para web con PhotoShop

El espacio de trabajo está organizado para ayudar al diseñador a centrarse en la creación y edición de imágenes, incluye menús y una serie de herramientas y paneles para visualizar, editar y añadir elementos a las imágenes.
Las imágenes se manipulan empleando distintos elementos como paneles, barras y ventanas, cualquier disposición de estos elementos se denomina espacio de trabajo. El espacio de trabajo esta compuesto principalmente por:

- La barra Aplicación de la parte superior contiene un conmutador de espacio de trabajo, menús y otros controles de la aplicación.
- El panel Herramientas incluye utilidades para crear y editar imágenes, ilustraciones, elementos de página, etc.
- El panel Control muestra opciones específicas de la herramienta seleccionada en el momento.
- La ventana Documento exhibe el archivo en el que se trabaja. Las ventanas Documento se pueden organizar como fichas y, en ciertos casos, también se pueden agrupar y acoplar.
- Los paneles ayudan a controlar y modificar el trabajo.

El espacio de trabajo se puede personalizar moviendo y manipulando los panales y las ventanas del Documento. También puede guardar espacios de trabajo y pasar de uno a otro.




A. Barra de título
B. Fija
C. Zona de colocación

**Figura 95 - Photoshop: El espacio de trabajo**

## 6.1 Conceptos básicos sobre imágenes

Photoshop puede abrir e importar numerosos tipos de archivos de gráficos. Pero, para trabajar de forma eficaz con los diferentes tipos de imágenes es necesario estudiar previamente conceptos básicos relacionados con imágenes, empecemos con los tipos de imágenes:

- **Imágenes de mapa de bits.** Este tipo de imágenes también se denominan imágenes rasterizadas, utilizan una cuadrícula rectangular de elementos de imagen (píxeles) para representar imágenes. A cada píxel se le asigna una ubicación y una valor de color específicos.

  Estas imágenes dependen de la resolución, es decir, contienen un número fijo de píxeles y por lo tanto, pueden perder detalle y aparecer dentadas si se cambia la escala a grandes ampliaciones.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc


**Figura 96 - Imágenes de mapa de bits**

Las imágenes de mapas de bits son el medio más usado para las imágenes de tono continuo como fotografías o pinturas digitales.

- **Gráficos vectoriales.** Estos gráficos están compuestos de líneas y curvas definidas por objetos matemáticos denominados vectores, que describen una imagen de acuerdo con sus características geométricas.

  Son independientes de resolución, es decir, mantienen la nitidez cuando se les cambia el tamaño o se realice cualquier otro tipo de acción sobre ellos

Las imágenes de Photoshop tienen uno o varios **canales**, cada uno de los cuales almacena información sobre los elementos de color de la imagen. La cantidad de canales de color por defecto de una imagen depende de su modo de color. Estos canales son, en realidad, imágenes en escala de grises que representan cada uno de los componentes de color de la imagen. Por ejemplo, una imagen RGB dispone de canales independientes para cada valor de color: rojo, verde y azul.

La **profundidad de bits** especifica la cantidad de información de color que está disponible para cada píxel de una imagen. Cuantos más bits de información por píxel hay, más colores disponibles existen y más precisión en la representación del color se aprecia. Por ejemplo, una imagen con una profundidad de bits de 1 tiene píxeles con dos valores posibles: blanco y negro. Una imagen con una profundidad de bits de 8 tiene 28 o 256 valores posibles.

Las imágenes RGB se componen de tres canales de color. Una imagen RGB con 8 bits por píxel cuenta con 256 posibles valores para cada canal, lo que significa más de 16 millones de posibles valores de color. En ocasiones, las imágenes RGB con 8 bits por canal se denominan imágenes de 24 bits (8 bits x 3 canales = 24 bits de datos por píxel).



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 6.2 Operaciones básicas

### 6.2.1 Cambio de dimensiones

Como ya hemos comentado, las dimensiones en píxeles de las imágenes de mapa de bits son una medida del número de píxeles de altura y anchura de la imagen. La resolución es la *precisión del detalle* en las imágenes de mapa de bits, que se mide en píxeles por pulgada (ppp). Cuantos más píxeles por pulgada, mayor resolución. En general, las imágenes con más resolución producen una calidad de impresión mejor. En la imagen siguiente podemos ver una imagen a 72 ppp (izquierda) y una imagen a 300 ppp (derecha).

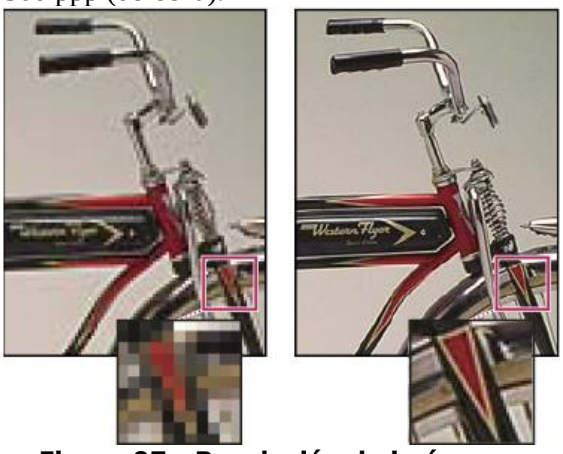
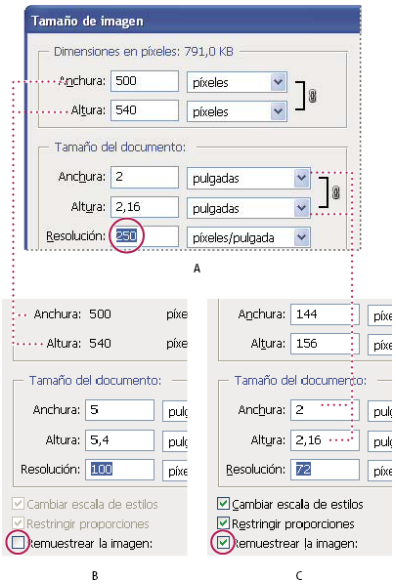


**Figura 97 - Resolución de imágenes**

La *combinación* de las dimensiones en píxeles y la resolución determina la cantidad de datos de las imágenes. Salvo en el caso de las imágenes remuestreadas, la cantidad de datos de la imagen permanece inalterada al cambiar las dimensiones en píxeles o la resolución. Si modifica la resolución de un archivo, su anchura y su altura cambian en consecuencia a fin de mantener la misma cantidad de datos de imagen, y al contrario.
En Photoshop, la relación entre el tamaño de la imagen y la resolución se ve en el cuadro de diálogo Tamaño de imagen (Imagen > Tamaño de imagen). Anule la selección de Remuestrear la imagen porque no desea alterar la cantidad de datos de imagen de la fotografía. A continuación, cambie la anchura o la altura o bien la resolución. Al modificar uno de los valores, los otros dos varían en consecuencia.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc



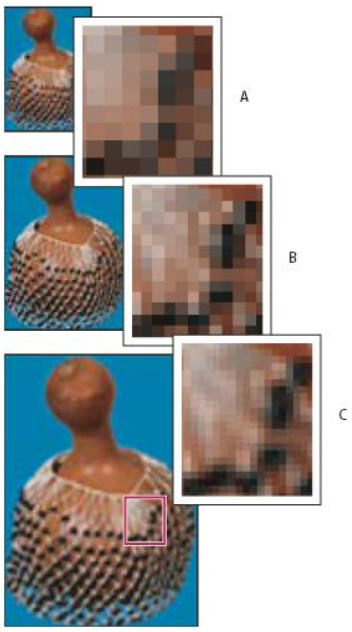

Las dimensiones en píxeles equivalen a la resolución multiplicada por el tamaño (de salida) del documento.
A. Dimensiones y resolución originales
B. Disminuir la resolución sin modificar las dimensiones en píxeles (sin remuestreos)
C. Disminuir la resolución con el mismo tamaño de documento disminuye las dimensiones de los píxeles (remuestreo).

**Figura 98 - Cambio de tamaño de imágenes**

El ***remuestreo*** consiste en alterar la cantidad de datos de imagen al cambiar las dimensiones en píxeles o la resolución de la imagen. Al *disminuir la resolución* (reducir el número de píxeles), se borra información de la imagen. Si *aumenta la resolución, (aumenta la cantidad de píxeles*), se añaden nuevos píxeles. Recuerde que el remuestreo puede tener como resultado una imagen de menor calidad.
Photoshop remuestrea las imágenes por medio de un método de interpolación para asignar valores de color a los píxeles nuevos en función de los valores de color de píxeles existentes. El método que se debe usar se selecciona en el cuadro de diálogo Tamaño de imagen.

- **Por aproximación.** Un método rápido aunque menos preciso que reproduce los píxeles de una imagen.
- **Bilineal.** Un método que añade píxeles mediante el cálculo de la media de los valores de color de los píxeles adyacentes. Produce resultados de calidad media.
- **Bicúbica.** Un método más preciso aunque más lento basado en un examen de los valores de los píxeles adyacentes.
- **Bicúbica más suavizada.** Un buen método para ampliar imágenes basado en la interpolación bicúbica pero diseñado para producir resultados más suaves.
- **Bicúbica más enfocada**. Un buen método para reducir el tamaño de una imagen basado en la interpolación bicúbica con un enfoque mejorado.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc




A: Disminución de la resolución
B: Original
C: Aumento de la resolución

**Figura 99 - Remuestreo de imágenes**

Si crea una imagen para soportes impresos, resulta útil especificar el tamaño de la imagen en términos de las dimensiones de impresión y la resolución de imagen. Estas dos medidas, a las que se denomina *tamaño del documento*, determinan la cantidad total de píxeles y, por lo tanto, el tamaño de archivo de la imagen; el tamaño del documento también determina el tamaño base con el cual una imagen se coloca dentro de otra aplicación.

### 6.2.2 Color

El color se puede describir de modos distintos usando un modelo de color que describen los colores que se ven en las imágenes digitales y con los que se trabaja. Cada modelo de color como, por ejemplo, RGB, CMYK o HSB representa un método diferente (por lo general, numérico) de descripción de los colores.
Los colores primarios aditivos son los tres colores de la luz (rojo, verde y azul), que producen todos los colores del espectro visible al unirse en distintas combinaciones. Al sumar partes iguales de rojo, azul y verde, se obtiene el color blanco. La total ausencia de rojo, azul y verde da como resultado el color negro. Los monitores de ordenador son dispositivos que emplean colores primarios aditivos para crear color.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

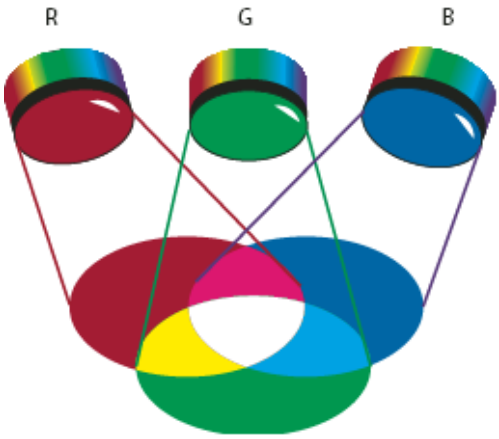



**Figura 100 - Colores primarios adictivos**

Los colores primarios sustractivos son pigmentos que crean un espectro de colores en diferentes combinaciones. Las impresoras emplean colores primarios sustractivos (pigmentos cian, magenta, amarillos y negros) para producir los colores mediante mezclas sustractivas.

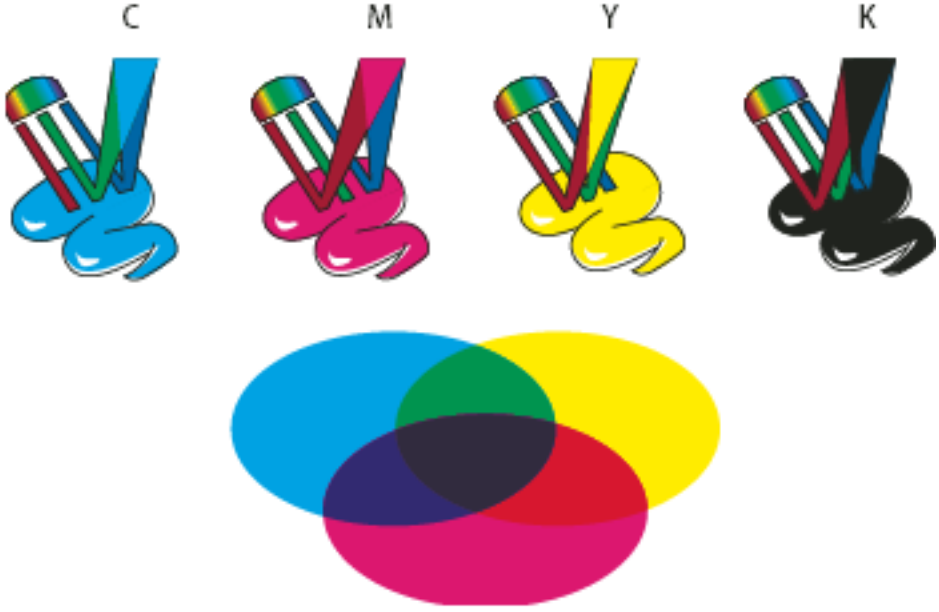



**Figura 101 - Colores primarios substractivos**

Los diagramas de la rueda de colores estándar le servirán de ayuda al trabajar con equilibrio de color. La rueda de colores se utiliza para predecir cómo un cambio en un componente de color afecta a otros colores y también cómo se convierten los cambios entre los modelos de color RGB y CMYK.



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

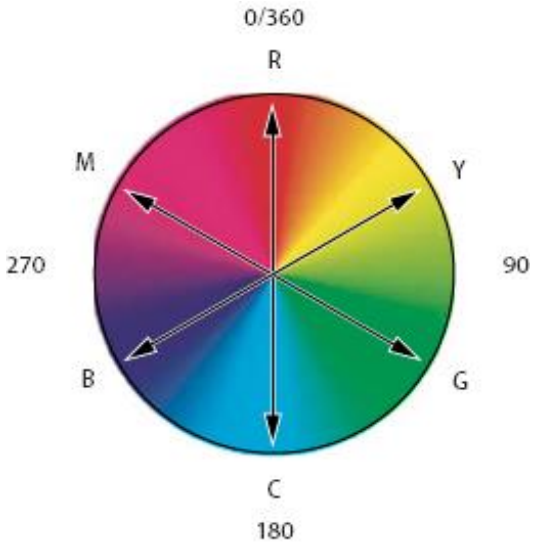



**Figura 102 - Diagrama de rueda de color**

Un **espacio de color** es una variante de un modelo de color que tiene una gama (rango) específica de colores. Por ejemplo, en el modelo de color RGB, hay un número de espacios de color: Adobe RGB, sRGB, ProPhoto RGB, etc. Los modos de color determinan el número de colores, el número de canales y el tamaño de archivo de una imagen. Seleccionar un modo de color también determina las herramientas y los formatos de archivo disponibles.
Cada dispositivo (ya sea un monitor o una impresora) dispone de su propio espacio de color, por lo que sólo puede reproducir los colores de su gama. Cuando una imagen pasa de un dispositivo a otro, los colores de la imagen pueden cambiar porque cada dispositivo interpreta los valores RGB o CMYK según su propio espacio de color.
En Photoshop, el modo de color de un documento determina el modelo de color utilizado para mostrar e imprimir la imagen en la que está trabajando. Los modos de color de Photoshop están basados en los modelos de color que resultan útiles en las imágenes que se utilizan en autoedición. Puede seleccionar entre RGB (rojo, verde, azul); CMYK (cian, magenta, amarillo, negro); Color Lab (basado en CIE L* a* b*) y Escala de grises.

- Las imágenes RGB utilizan tres colores o *canales* para reproducir los colores en la pantalla. En imágenes de 8 bits por canal, los tres canales se convierten en 24 (8 bits x 3 canales) bits de información del color por píxel. En imágenes de 24 bits, los tres canales pueden reproducir hasta 16,7 millones de colores por píxel. En imágenes de 48 bits (16 bits por canal) y 96 bits (32 bits por canal), pueden reproducirse incluso más colores por píxel.
- En el modo CMYK, a cada píxel se le asigna un valor de porcentaje para las tintas de cuatricromía. Los colores más claros (iluminaciones) tienen un porcentaje pequeño de tinta, mientras que los más oscuros (sombras) tienen porcentajes mayores. Por ejemplo, un rojo brillante podría tener 2% de cian, 93% de magenta, 90% de amarillo y 0% de negro. En las imágenes CMYK, el blanco puro se genera si los cuatro componentes tienen valores del 0%.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- El modo escala de grises utiliza distintos tonos de gris en una imagen. En imágenes de 8 bits, puede haber hasta 256 tonos de gris. Cada píxel de una imagen en escala de grises tiene un valor de brillo comprendido entre 0 (negro) y 255 (blanco).

El modelo HSB se basa en la percepción humana del color y describe tres características fundamentales del color:

- **Tono.** Color reflejado o transmitido a través de un objeto. Se mide como una posición en la rueda de colores estándar y se expresa en grados, entre 0° y 360°. Normalmente, el tono se identifica por el nombre del color, como rojo, naranja o verde.
- **Saturación.** A veces denominada cromatismo, es la fuerza o pureza del color. La saturación representa la cantidad de gris que existe en proporción al tono y se mide como porcentaje comprendido entre 0% (gris) y 100% (saturación completa). En la rueda de colores estándar, la saturación aumenta a medida que nos aproximamos al borde de la misma.
- **Brillo.** Luminosidad u oscuridad relativa del color y normalmente se expresa como un porcentaje comprendido entre 0% (negro) y 100% (blanco).

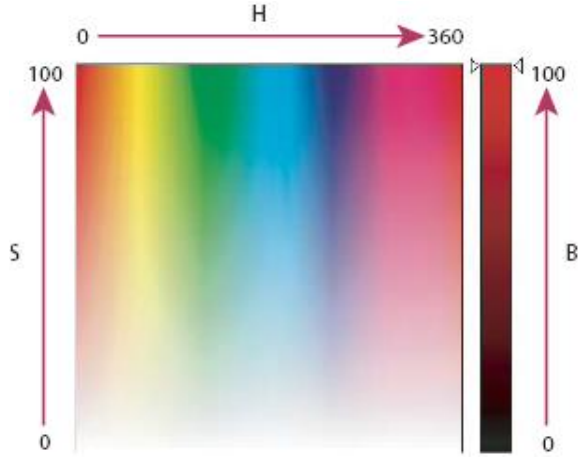



**Figura 103 - El modelo HSB**

Puede convertir el modo original de una imagen (modo de origen) en otro distinto (modo de destino). La selección de un modo de color distinto cambia de forma permanente los valores de color de la imagen. Se recomienda realizar todas las operaciones sobre la imagen y previamente a realizar la conversión realizar una copia de seguridad de la imagen. Para realizar la conversión elija Imagen > Modo y el modo que desee en el submenú. Los modos que no están disponibles para la imagen activa aparecen inhabilitados en el menú. Las imágenes se acoplan al convertirlas a los modos Multicanal, Mapa de bits o Color indexado porque estos modos no admiten capas.
Un histograma ilustra en un gráfico cómo están distribuidos los píxeles de la imagen mostrando la cantidad de píxeles en cada nivel de intensidad del color. Al mostrar los detalles de las sombras (en la parte izquierda), los medios tonos (en el centro) y las iluminaciones (en la parte derecha), el histograma ayuda a determinar si la imagen contiene suficientes detalles para realizar una corrección correcta.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

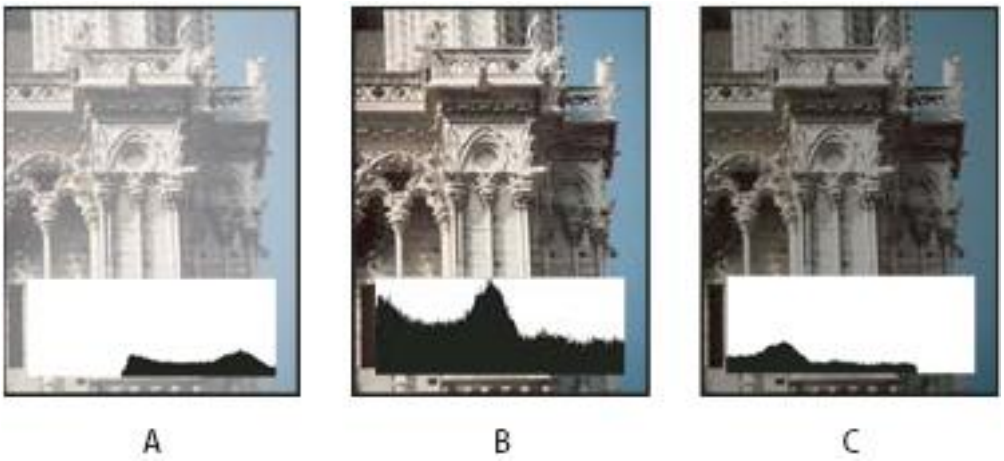



**Figura 104 - Histograma**

El histograma también ofrece una imagen rápida de la gama tonal, o tonalidad, de la imagen. El detalle de una imagen con tonalidad dominante oscura se concentra en las sombras; el detalle de una imagen con tonalidad dominante clara, en las iluminaciones; y el detalle de una imagen con tonalidad media, en los medios tonos. Una imagen con una gama tonal completa tiene píxeles en todas las áreas. Identificar la gama tonal ayuda a determinar las correcciones tonales correspondientes.

### 6.2.3 Transformación

Las funciones de retoque y transformación de Photoshop® le permiten alterar las imágenes para llevar a cabo una variedad de tareas: para mejorar una composición, corregir distorsiones o defectos, manipular de forma creativa elementos de la imagen, añadir o eliminar elementos, enfocar y desenfocar o combinar varias imágenes en un panorama.

#### 6.2.3.1 Ajuste del recorte, rotación y lienzo

Recortar es el proceso de eliminar partes de una imagen para crear un enfoque o reforzar la composición. Puede recortar una imagen mediante la herramienta Recortar y el comando Recortar. Para arrastre el puntero sobre la parte de la imagen que desea conservar para crear un marco. El marco no tiene que ser preciso. Puede ajustarlo posteriormente

**Figura 105 - Recorte de imágenes**

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

La herramienta Recortar tiene una opción que permite transformar la perspectiva de una imagen. Es muy útil para trabajar con imágenes que contengan *distorsión trapezoidal*. La distorsión trapezoidal se produce cuando se fotografía un objeto desde un ángulo en lugar de desde una vista recta. Por ejemplo, si realiza una fotografía de un edificio alto desde el nivel del suelo, los bordes del edificio se muestran más cercanos en la parte superior que en la parte inferior.

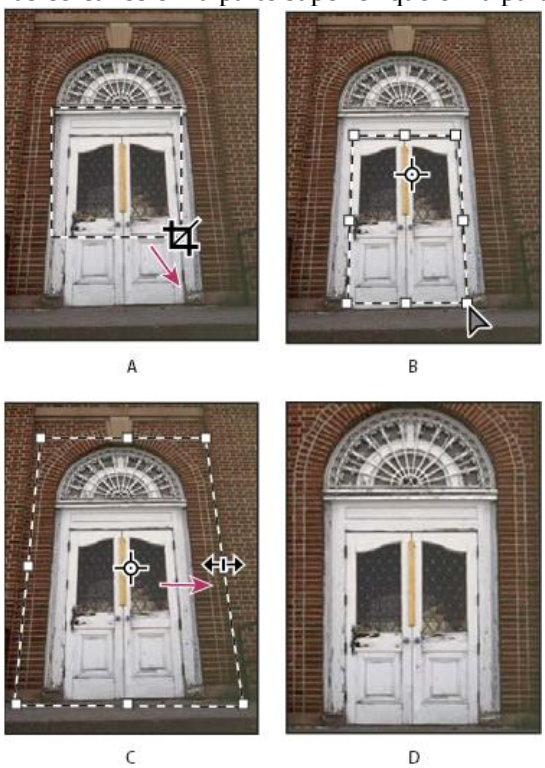



**Figura 106 - Perspectiva de imágenes**

6.2.3.2 Rotación de una imagen

Los comandos de Rotación de imagen le permiten rotar o voltear una imagen completa. Los comandos no funcionan en capas individuales ni en partes de capas, trazados o bordes de selección. Si desea rotar una selección o capa, utilice los comandos Transformar o Transformación libre.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

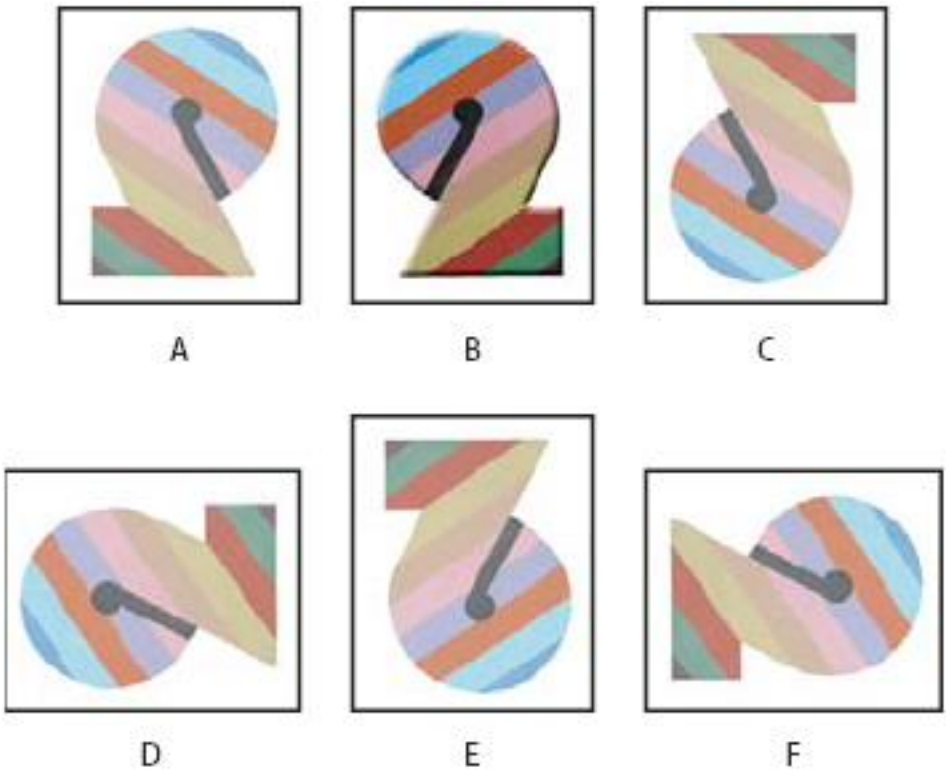



**Figura 107 - Volteo de imágenes**

- 180°. Rota la imagen media vuelta.
- 90° AC. Rota la imagen un cuarto de vuelta a la derecha.
- 90° ACD. Rota la imagen un cuarto de vuelta a la izquierda.
- Arbitrario. Rota la imagen según el ángulo especificado. Si selecciona esta opción, introduzca un ángulo comprendido entre -359,99 y 359,99 en el cuadro de texto Ángulo.

#### 6.2.3.3 Cambio del tamaño del lienzo

El tamaño del lienzo es el área editable completa de la imagen. El comando Tamaño de lienzo permite aumentar o reducir el tamaño del lienzo de la imagen. El aumento del tamaño del lienzo añade espacio alrededor de una imagen existente. Si se reduce el tamaño del lienzo de la imagen, se recortará en la imagen. Si aumenta el tamaño del lienzo de una imagen con un fondo transparente, el lienzo añadido será transparente. Si la imagen no tiene un fondo transparente, existen varias opciones para determinar el color del lienzo añadido. Las diferentes opciones se pueden realizar a través de la opción Imagen > Tamaño de lienzo.

#### 6.2.3.4 Enfoque de imágenes

El enfoque mejora la definición de los bordes de una imagen. Procedan de una cámara digital o de un escáner, la mayoría de las imágenes se pueden beneficiar con la operación de enfoque. El grado de enfoque necesario varía en función de la calidad de la cámara digital o del escáner. Tenga en cuenta que el enfoque no puede corregir una imagen muy desenfocada. Notas y sugerencias acerca del enfoque:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- Enfoque la imagen en una capa independiente para que pueda volver a enfocarla posteriormente si va a usar un medio de salida distinto.
- Si enfoca la imagen en una capa independiente, defina el modo de fusión de la capa como Luminancia para evitar variaciones de color en los bordes.
- Si necesita reducir el ruido de la imagen, hágalo antes de enfocar para no intensificar el ruido.
- Enfoque la imagen varias veces en pequeñas cantidades. La primera vez, enfóquela para corregir el desenfoque causado al capturar la imagen (escaneándola o tomándola con una cámara digital). Después de realizar correcciones de color y tamaño, enfoque la imagen de nuevo (o una copia de la misma) para añadir la cantidad adecuada de enfoque para el medio de salida que va a utilizar.
- A ser posible, utilice el medio de salida final para determinar la cantidad de enfoque necesaria, ya que ésta varía de un medio de salida a otro.

Puede enfocar toda la imagen o sólo una parte definida por una selección o máscara. Como los filtros Máscara de enfoque y Enfoque suavizado sólo pueden aplicarse a una capa cada vez, puede que necesite combinar capas o acoplar el archivo para enfocar todas las capas de la imagen en un archivo de varias capas.
Para obtener un mayor control, utilice el filtro Máscara de enfoque o el filtro Enfoque suavizado para enfocar las imágenes. Aunque Photoshop también dispone de las opciones de filtro Enfocar, Enfocar bordes y Enfocar más, estos filtros son automáticos y no proporcionan controles ni opciones.

- **Filtro de enfoque suavizado.** El filtro Enfoque suavizado ofrece controles de enfoque que no están disponibles en el filtro Máscara de enfoque. Puede definir el algoritmo de enfoque o controlar la cantidad de enfoque en las áreas de sombras e iluminaciones. Seleccione Filtro > Enfocar > Enfoque suavizado.
- **Mascara de enfoque.** El filtro Máscara de enfoque enfoca una imagen aumentando el contraste a lo largo de los bordes de la misma. El filtro Máscara de enfoque no detecta los bordes de una imagen. En lugar de ello, busca los píxeles cuyo valor es distinto al de los píxeles circundantes según el umbral que especifique. A continuación, aumenta el contraste de los píxeles adyacentes según la cantidad que especifique. De este modo, para los píxeles adyacentes, los píxeles más claros se vuelve más claros y los píxeles más oscuros se vuelven más oscuros.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

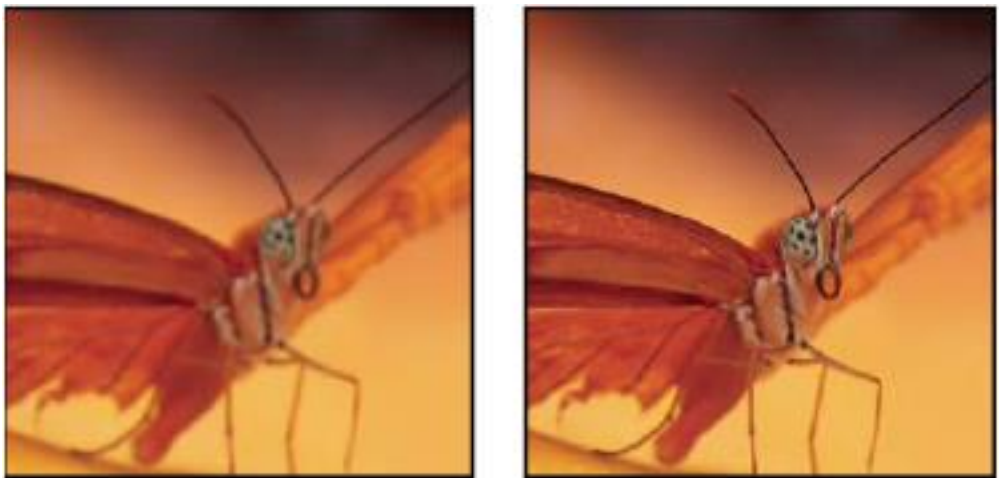


**Figura 108 - Enfoque de imágenes**

### 6.2.4 Retoque

#### 6.2.4.1 Tampón de clonar

La herramienta Tampón de clonar  pinta una parte de una imagen sobre otra parte de la misma imagen o sobre otra parte de cualquier documento abierto que tenga el mismo modo de colores. También puede pintar parte de una capa sobre otra capa. La herramienta Tampón de clonar resulta útil para duplicar objetos o eliminar defectos de una imagen.
Para utilizar la herramienta Tampón de clonar, defina un punto de muestra en el área de la que desea copiar (clonar) los píxeles y pinte en otra área. Para pintar con el punto de muestra más actual siempre que se detenga y vuelva a pintar, seleccione la opción Alineado. Anule la selección de la opción Alineado para pintar desde el punto de muestra inicial, independientemente de las veces que haya parado y reanudado la pintura.

**Figura 109 - Herramienta: tampón de clonar**

#### 6.2.4.2 Pincel corrector

La herramienta Pincel corrector permite corregir imperfecciones para que desaparezcan de la imagen. Al igual que las herramientas de clonar, la herramienta Pincel corrector se utiliza para pintar con píxeles muestreados de una imagen o un motivo. Sin embargo, la herramienta Pincel corrector

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

también hace coincidir la textura, iluminación, transparencia y sombreado de los píxeles muestreados con los píxeles que se están corrigiendo. Como resultado, los píxeles reparados se fusionan perfectamente con el resto de la imagen.

**Figura 110 - Herramienta: Pincel corrector**

6.2.4.3 Pincel corrector puntual

La herramienta Pincel corrector puntual elimina taras y otras imperfecciones de las fotografías rápidamente. La herramienta Pincel corrector puntual funciona de manera similar al Pincel corrector: pinta con píxeles muestreados de una imagen o motivo y hace coincidir la textura, iluminación, transparencia y sombra de los píxeles muestreados con los píxeles que se están corrigiendo. A diferencia del Pincel corrector, para el Pincel corrector puntual no necesita especificar un punto de la muestra. El Pincel corrector puntual muestrea automáticamente el área que rodea el punto retocado.

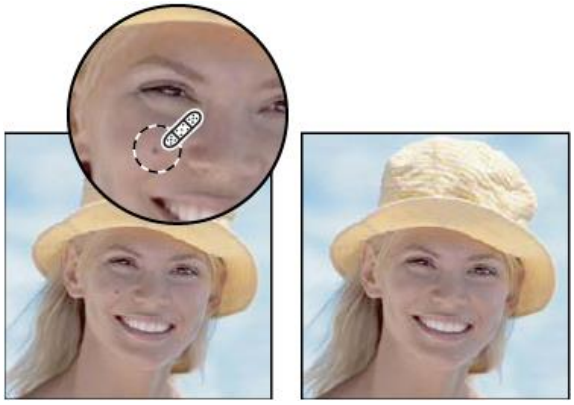


**Figura 111 - Herramienta: Pincel corrector puntual**

6.2.4.4 Eliminación de ojos rojos

La herramienta Pincel de ojos rojos elimina los ojos rojos de las fotografías con flash de gente y animales y los reflejos blancos o verdes de las fotografías con flash de animales.

1. Seleccione la herramienta Ojos rojos . (La herramienta Ojos rojos se encuentra en el mismo grupo que la herramienta Pincel corrector puntual . Haga clic en el triángulo de la parte inferior derecha de una herramienta para mostrar herramientas adicionales).


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

2. Haga clic en el ojo rojo. Si no está conforme con los resultados, deshaga la corrección, defina una o varias de las siguientes opciones en la barra de opciones y vuelva a hacer clic en el ojo rojo:
    - Tamaño de pupila. Aumenta o reduce el área afectada por la herramienta Pincel de ojos rojos.
    - Cantidad de oscurecimiento. Define la cantidad de oscurecimiento de la corrección.

#### 6.2.4.5 Herramienta dedo

La herramienta Dedo simula el efecto que se consigue al pasar un dedo por pintura fresca. La herramienta recoge el color en el punto donde empieza el trazo y lo empuja y extiende en la dirección del arrastre.

1. Seleccione la herramienta Dedo  .
2. Elija una punta de pincel y las opciones del modo de fusión en la barra de opciones.
3. Seleccione Muestrear todas las capas en la barra de opciones para emborronar utilizando los datos de color de todas las capas visibles. Si esta opción está deseleccionada, la herramienta Dedo sólo utiliza los colores de la capa activa.
4. Seleccione Pintar con los dedos en la barra de opciones para emborronar utilizando el color frontal al principio de cada trazo. Si esta opción está deseleccionada, la herramienta Dedo utiliza el color situado debajo del puntero al principio de cada trazo.
5. Arrastre el puntero en la imagen para emborronar los píxeles.

### 6.2.5 Selección de píxeles

Si desea aplicar cambios a partes de una imagen, primero debe seleccionar los píxeles que componen esas partes. Puede seleccionar los píxeles en Photoshop con las herramientas de selección o pintándolos en una máscara y cargando dicha máscara como una selección.
Una selección aísla una o varias partes de la imagen. Si selecciona áreas específicas, puede editar y aplicar efectos y filtros a partes de la imagen sin tocar las áreas que no ha seleccionado.
Photoshop proporciona varios conjuntos de herramientas separados para realizar selecciones de datos vectoriales o datos rasterizados. Por ejemplo, para seleccionar píxeles, puede utilizar las herramientas de marco o las de lazo. Utilice los comandos del menú Selección para seleccionar todos los píxeles, deseleccionar o volver a seleccionar.

#### 6.2.5.1 Herramienta marco

Las herramientas de marco le permiten seleccionar rectángulos, elipses y filas y columnas de 1 píxel.

1. Seleccione una herramienta de marco:
    - Marco rectangular 
    - Realiza una selección rectangular (o cuadrada, cuando se utiliza la tecla Mayús).
    - Marco elíptico 
    - Realiza una selección elíptica (o un círculo, cuando se utiliza la tecla Mayús).
    - Marco fila única  o Marco columna única 

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

2. Especifique un ajuste de desvanecimiento en la barra de opciones. Active o desactive el suavizado para la herramienta Marco elíptico. Consulte Suavizado de bordes de selecciones.
3. Para las herramientas Marco rectangular o Marco elíptico, seleccione un estilo en la barra de opciones:
    - Normal. Determina las proporciones del marco al arrastrar.
    - Proporción fija. Define una proporción altura-anchura. Introduzca valores (los decimales son válidos) para la proporción. Por ejemplo, para dibujar un marco con una anchura que sea dos veces la altura, introduzca 2 para la anchura y 1 para la altura.
    - Tamaño fijo. Especifica valores fijos para la altura y la anchura del marco. Introduzca los valores de píxeles en números enteros.
4. Para alinear la selección a guías, una cuadrícula, sectores o límites del documento, realice una de las acciones siguientes para ajustar la selección:
    - Seleccione Vista > Ajustar o Vista > Ajustar a y, a continuación, seleccione un comando del submenú. La selección realizada con una herramienta de marco se puede ajustar al límite o a una serie de extras de Photoshop, controlados en el submenú Ajustar a.
5. Para realizar una selección, lleve a cabo una de las acciones siguientes:
    - Con las herramientas Marco rectangular o Marco elíptico, arrastre sobre el área que desee seleccionar.
    - Mantenga pulsada la tecla Mayús mientras arrastra para restringir el marco a un cuadrado o a un círculo (suelte el botón del ratón antes de Mayús para limitar la forma de la selección).
    - Para arrastrar un marco desde su centro, mantenga pulsada la tecla Alt (Windows) u Opción (Mac OS) después de comenzar a arrastrar.

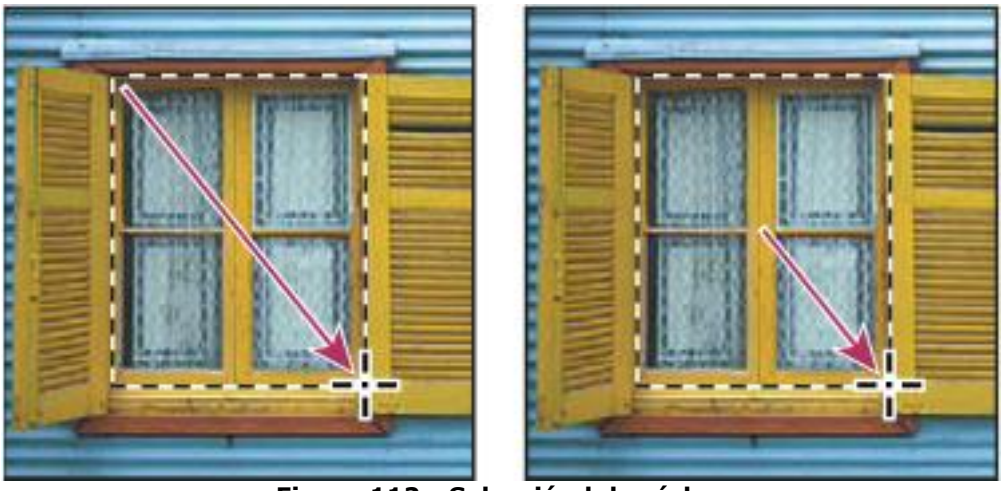


**Figura 112 - Selecciónd de píxles**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Arrastrar un marco desde la esquina de una imagen (izquierda) y desde el centro de una imagen (derecha) pulsando Alt/Opción mientras arrastra

6.2.5.2 Herramienta lazo

La herramienta Lazo resulta útil para dibujar segmentos de forma libre de un borde de selección.

1. Seleccione la herramienta Lazo  y, a continuación, seleccione opciones.
2. Arrastre para dibujar un borde de selección a mano alzada.
3. Especifique una de las opciones de selección en la barra de opciones.
4. (Opcional) Defina las opciones de calado y suavizado en la barra de opciones. Consulte Suavizado de bordes de selecciones.
5. Para dibujar un borde de selección rectilíneo cuando no hay ningún otro píxel seleccionado, pulse Alt (Windows) u Opción (Mac OS) y haga clic donde los segmentos deben empezar y terminar. Puede optar por dibujar a mano alzada o segmentos de borde rectilíneo.
6. Para borrar segmentos recién dibujados, mantenga pulsada la tecla Supr hasta que haya borrado los puntos de fijación del segmento deseado.
7. Para cerrar el borde de selección, suelte el botón del ratón sin mantener pulsada la tecla Alt (Windows) u Opción (Mac OS).
8. (Opcional) Haga clic en Perfeccionar borde para realizar más ajustes en el límite de selección o ver la selección con distintos fondos o como máscara. Consulte Precisión de bordes de selecciones.

6.2.5.3 Herramienta varita

La herramienta Varita mágica le permite seleccionar un área coloreada de forma coherente (por ejemplo, una flor roja) sin tener que trazar su contorno. Puede especificar el rango de color o la tolerancia para la selección de la herramienta Varita mágica, según la similitud del píxel en el que haga clic.

1. Seleccione la herramienta Varita mágica  .
2. Especifique una de las opciones de selección en la barra de opciones.
3. En la barra de opciones, especifique una de las siguientes opciones:
   - **Tolerancia.** Determina la similitud o la diferencia de los píxeles seleccionados. Introduzca un valor en píxeles, entre 0 y 255. Si define un valor bajo, seleccionará aquellos colores que más se asemejen al píxel en el que ha hecho clic. Cuanto más alto sea el valor, mayor será la gama de colores seleccionada.
   - **Suavizar.** Crea una selección de bordes más suaves.
   - **Contiguo.** Selecciona sólo zonas adyacentes con los mismos colores. De lo contrario, se seleccionan todos los píxeles de la imagen completa con los mismos colores.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Muestrear todas las capas.** Selecciona colores con datos de todas las capas visibles. De lo contrario, la herramienta Varita mágica selecciona colores sólo de la capa activa.

4. En la imagen, haga clic en el color que desea seleccionar. Si selecciona Contiguo, se seleccionan todos los píxeles adyacentes dentro del rango de tolerancia. En caso contrario, se seleccionan todos los píxeles del rango de tolerancia.
5. (Opcional) Haga clic en Perfeccionar borde para realizar más ajustes en el límite de selección o ver la selección con distintos fondos o como máscara. Consulte Precisión de bordes de selecciones.

### 6.2.6 Pintura

La pintura cambia el color de los píxeles de la imagen. Puede utilizar las herramientas y técnicas de pintura para retocar imágenes, crear o editar máscaras en canales alfa, usar la técnica rotoscope o pintar en cuadros de vídeo y pintar originales ilustraciones. Las puntas de pincel, los pinceles preestablecidos y muchas de las opciones de pincel le otorgan un control creativo para producir efectos de pintura impresionantes o simular el trabajo con los medios tradicionales.
Photoshop cuenta con varias herramientas para pintar y editar imágenes en color. La herramienta Pincel y la herramienta Lápiz funcionan como las herramientas de dibujo tradicionales aplicando color mediante trazos de pincel. La herramienta Degradado, el comando Rellenar y la herramienta Bote de pintura aplican color a áreas grandes. Herramientas como Borrador, Desenfocar o Dedo modifican los colores existentes de la imagen. Consulte Galería de herramientas de pintura.
En la barra de opciones de cada herramienta, puede definir la forma en la que se aplica el color a una imagen y elegir varias puntas de pincel preestablecidas.

- Ajustes preestablecidos de herramientas de pincel

Puede guardar un conjunto de opciones de pincel como un ajuste preestablecido para que pueda acceder rápidamente a las características de pincel que utiliza con frecuencia. Photoshop incluye varios ajustes preestablecidos de pincel de muestra. Puede empezar con estos ajustes preestablecidos y modificarlos para producir nuevos efectos. Muchos de los ajustes preestablecidos de pincel originales se pueden descargar desde la Web.
Puede seleccionar los ajustes preestablecidos desde el selector Pinceles preestablecidos, que almacena los ajustes preestablecidos de pincel y permite modificar temporalmente el diámetro y la dureza de un ajuste preestablecido de pincel.
Puede utilizar los ajustes preestablecidos cuando desee guardar las características de la punta de pincel junto con los ajustes de la barra de opciones, como opacidad, flujo y color. Para obtener más información sobre los ajustes preestablecidos de las herramientas, consulte Creación y uso de las herramientas preestablecidas.

- Opciones de punta de pincel

Junto con los ajustes de la barra de opciones, las opciones de punta de pincel controlan la forma en la que se aplica el color. Puede aplicar color de forma gradual, con bordes suaves, con grandes trazos de pincel, con distintas dinámicas de pincel, con diversas propiedades de fusión y con pinceles de distintos tamaños. Puede aplicar una textura con los trazos de pincel para simular la pintura en lienzo o papel de dibujo. También puede simular el pulverizado de pintura mediante un aerógrafo. Puede utilizar el panel Pinceles para definir las opciones de punta de pincel. Consulte Descripción general del panel Pinceles.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Si trabaja con una tableta de dibujo, puede controlar la forma en la que se aplica el color utilizando la presión de la pluma, el ángulo, la rotación o el rotativo de stylus. Puede definir las opciones para las tabletas de dibujo en el panel Pinceles.

### 6.2.7 Capas

Las capas son los elementos fundamentales que integran muchos flujos de trabajo de creación de imágenes. Quizá pueda prescindir de las capas si sólo realiza ajustes sencillos de imagen. No obstante, las capas ayudan a mejorar la eficiencia y resultan esenciales para la mayoría de tareas de edición no destructiva de imagen

Las capas de Photoshop semejan hojas apiladas de acetato. Puede ver las capas que se encuentran debajo a través de las áreas transparentes de una capa. Si mueve una capa a fin de colocar el contenido en ella, es como si deslizara una hoja de acetato dentro de una pila. Además, si lo desea, tiene la posibilidad de cambiar la opacidad de alguna capa de manera que su contenido sea parcialmente transparente.

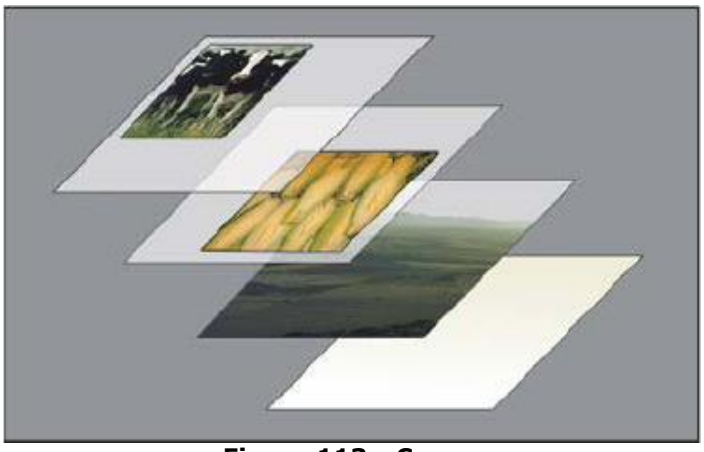


**Figura 113 - Capas**

Las capas sirven para realizar diversas tareas, entre otras, componer varias imágenes, añadir texto a una imagen o añadir formas de gráficos vectoriales. La aplicación de un estilo de capa facilita la adición de efectos especiales como sombras paralelas o algún tipo de resplandor.

- **Operaciones no destructivas**

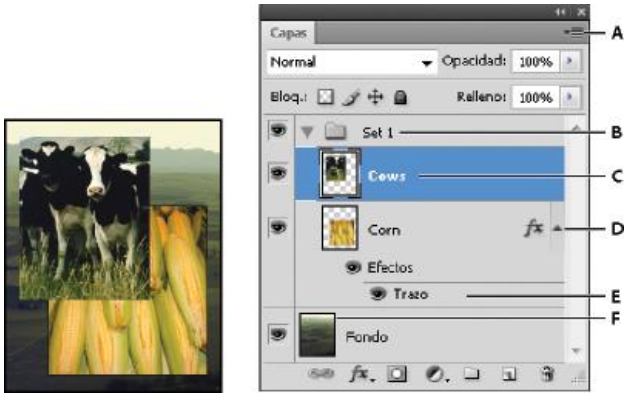
En ocasiones, parece como si algunas capas estuvieran desprovistas de contenido. Las capas de ajuste, por ejemplo, sólo contienen ajustes de color o tono que afectan a las capas inferiores. Así, en lugar de editar los píxeles de la imagen, se puede editar una capa de ajuste y mantener los píxeles subyacentes sin cambios.

- **Organización de capas**

Cada imagen nueva tiene una sola capa. El número de capas, conjuntos de capas y efectos de capa adicionales que puede añadir a la imagen está limitado sólo por la memoria de su equipo. Las operaciones con las capas se llevan a cabo en el panel Capas. Los grupos de capas le ayudan a organizar y gestionar las capas. Puede utilizar grupos para organizar las capas de manera lógica y despejar el panel Capas. Puede anidar grupos en otros grupos. También puede utilizar grupos para aplicar atributos y máscaras a varias capas a la vez.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc




A. Menú del panel Capas
B. Grupo de capas
C. Capa
D. Expandir/Contraer efectos de capa
E. Efecto de capa
F. Miniatura de la capa

**Figura 114 - Gestión de capas**

### 6.2.8 Filtros

Los filtros sirven para aplicar efectos especiales a las imágenes o para realizar tareas comunes de edición de imágenes como, por ejemplo, enfocar fotografías.
Los filtros se emplean para limpiar o retocar las fotografías, aplicar efectos especiales que doten a la imagen de la apariencia de un bosquejo o un cuadro impresionista o bien crear transformaciones exclusivas mediante distorsiones y efectos de iluminación. Los filtros que ofrece Adobe aparecen en el menú Filtro. Algunos filtros que proporcionan otros desarrolladores están disponibles en forma de plugins. Una vez instalados, estos filtros de plugins aparecen en la parte inferior del menú Filtro.
Los filtros inteligentes, aplicados a objetos inteligentes, permiten usar filtros no destructivos. Los filtros inteligentes se almacenan como efectos de capa en el panel Capas y se pueden volver a ajustar en cualquier momento a partir de los datos de la imagen original que contenga el objeto inteligente. Para obtener más información sobre los efectos de filtro inteligente y la edición no destructiva, consulte Edición no destructiva.
Para utilizar un filtro, seleccione el comando de submenú adecuado del menú Filtro. Las siguientes directrices le ayudarán a seleccionar los filtros:

- Los filtros se aplican a la capa activa visible o a una selección.
- En el caso de las imágenes de 8 bits por canal, la mayoría de los filtros se pueden aplicar de manera acumulativa mediante la Galería de filtros. Todos los filtros pueden aplicarse individualmente.
- No se pueden aplicar filtros a imágenes en modo de mapa de bits o de color indexado.
- Algunos filtros sólo funcionan en imágenes RGB.
- Todos los filtros pueden aplicarse a imágenes de 8 bits.
- Algunos filtros se procesan completamente en memoria RAM. Si no dispone de suficiente memoria RAM para procesar un efecto de filtro, es posible que reciba un mensaje de error.
-


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

# 7 Introducción a la animación multimedia con FLASH

Flash es una potente herramienta desarrollada por Adobe. La versión manejada en este manual es Flash CS4. Fue creado con el objeto de realizar animaciones y diseños vistosos para la web

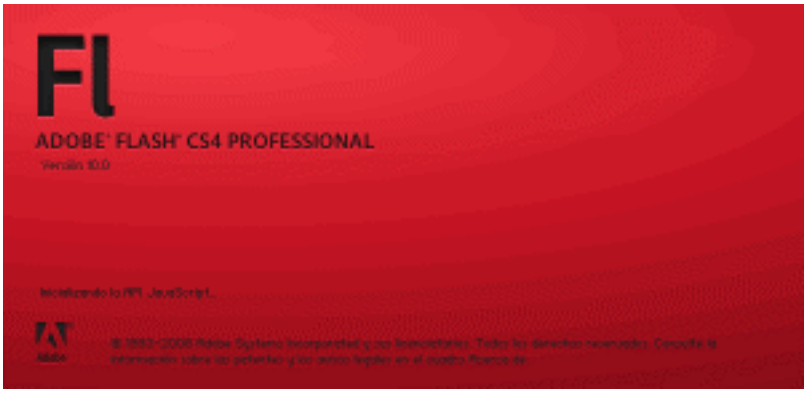



**Figura 115 - Ventana bienvenida Flash**

Flash ha conseguido hacer posible lo que más se echa en falta en Internet: Dinamismo, y con dinamismo no sólo nos referimos a las animaciones, sino que Flash permite crear aplicaciones interactivas que permiten al usuario ver la web como algo atractivo, no estático (en contraposición a la mayoría de las páginas, que están realizadas empleando el lenguaje HTML). Con Flash podremos crear de modo fácil y rápido animaciones de todo tipo, desde un botón a un complejo juego.

Además sus desarrolladores estan apostando muy fuerte por ActionScript, el lenguaje de programación Flash. A cada versión se mejora y ofrece un abanico de posibilidades cada vez mayor, por lo que además de dinamismo, Flash nos ofrece la posibilidad de ser la plataforma para aplicaciones web de un modo real.

El principal uso de Flash se da en el mundo de la web. Desde la creación de pequeños botones o banners publicitarios, hasta webs totalmente basadas en esta tecnología. Internet está repleta de animaciones Flash.

Como todo, Flash presenta tanto ventajas como inconvenientes:

- El tiempo de carga. Mientras que una página HTML puede ocupar unos 10-20 KB como media, una animación Flash ocupa mucho más. Depende del contenido que tenga, pero suelen superar los 100 KB con facilidad, y si además incorpora sonidos es fácil que la cifra se dispare. Al ocupar más espacio, el tiempo que tarda en estar visible el contenido Flash es mayor y no todos los visitantes están dispuestos a esperar... simplemente, se irán a otra página.

- Los buscadores. Son capaces de indexar el contenido de nuestra página, el texto, pero no el contenido del Flash, ya que no lo pueden leer, lo que afectará negativamente al posicionamiento de la página. Y hoy en día es crucial para una Web encontrarse bien posicionada. No obstante, los buscadores trabajan para solucionar este problema, pero de momento al mejor forma de solucionarlo es crear un diseño paralelo sin Flash, lo que aumenta el trabajo.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Flash requiere de plugins para poder visualizarse, y el hecho de no tenerlos instalados, o de que un navegador tenga los scripts deshabilitados por seguridad, impedirán la visualización del Flash. Este plugin lo suelen incorporar la mayoría de navegadores, es gratuito y se puede instalar de forma muy intuitiva, pero siempre habrá usuarios que prefieran salir de nuestra página si tienen que instalar "algo raro".
- Compatibilidad con distintos dispositivos. Cada vez es más frecuente acceder a la web con teléfonos móviles y PDAs, y muchos de ellos no soportan Flash.
- **Flash** es una tecnología propietaria **de Adobe**, por lo que su futuro depende de lo que esta compañía quiera hacer con él.
- Otro aspecto a tener en cuenta es la usabilidad de las páginas Flash, a veces se cae en la tentación de dar demasiada importancia al diseño y olvidarse de que la página debe ser fácil de usar, aunque es más un error de diseño que del propio Flash.

Por otro lado hay que tener en cuenta también las ventajas que aporta:

- La web se vuelve muy vistosa y atractiva, además de añadirle más interactividad.
- Con un poco de práctica, el desarrollo con Flash se vuelve rápido.
- Flash permite comportamientos que de otra forma no podríamos lograr.
- Compatibilidad con navegadores. Uno de los principales problemas en el diseño Web es que el resultado no tiene por qué verse igual en todos los navegadores. Con Flash, nos aseguramos de que lo que hemos creado es exactamente lo que se verá.
- Por tanto, se hace necesario cuando lo que se necesita es que el usuario pueda interactuar completamente con el contenido.

Flash tiene muchísimas aplicaciones más (hecho que ha provocado que otras compañías traten de sacar a la venta "clónicos" del Flash). Por ejemplo, se puede usar Flash para la creación de DVDs interactivos (como los que incluyen las revistas de informática, entre otras), la creación de banners publicitarios o lo que más está de moda ahora: la creación de dibujos animados. Además, Flash tiene uso industrial, pues se emplea para optimizar planos, crear diseños de interiores y trabajar con imágenes vectoriales en general.

Un consejo es emplear Flash únicamente en lo imprescindible en nuestra web dejando el contenido principal de la web y los elementos de navegación como HTML, que la web pueda funcionar si un usuario no puede ver los elementos en Flash y no se pierda nada realmente importante, a no ser que sea una web el la que su valor es justamente su contenido interactivo o visual en Flash.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.1 El entorno de Flash

Flash CS4 cuenta con un entorno o interfaz de trabajo renovada de lo más manejable e intuitiva. Además, tiene la ventaja de que es similar a la de otros programas de Adobe (Photoshop, Dreamweaver, Illustrator...), todo esto hace más fácil aprender Flash y más rápido su manejo y dominio. Esto es lo que nos encontraremos al abrir Flash CS4 por primera vez:

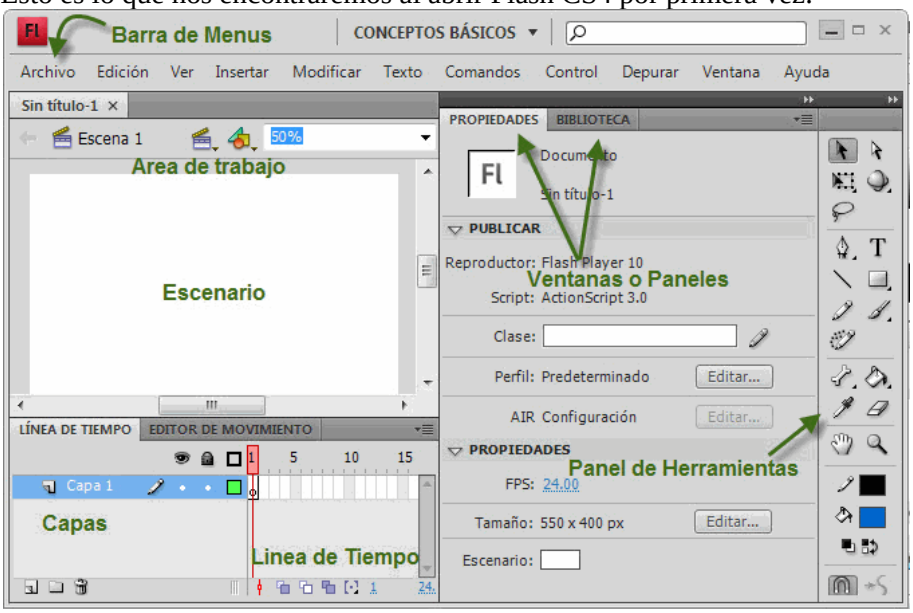



**Figura 116 - Pantalla principal Flash CS4**

En la imagen se puede ver la interfaz que nos encontraremos la primera vez que abramos el programa Flash. Flash recordará nuestras preferencias y abrirá el programa tal y como lo dejamos la última vez que lo utilizamos.

### 7.1.1 Barra de Menús

La **Barra de Menús** tiene como propósito facilitar el acceso a las distintas utilidades del programa. Es similar a la de cualquier otro programa de diseño Web o gráfico, aunque tiene algunas particularidades. Veamos los principales **Submenús** a los que se puede acceder:


**Figura 117 - Barra de menús**

- **Archivo**: Permite crear nuevos archivos, abrirlos, guardarlos... Destaca la potencia de la utilidad *Importar* que inserta en la película actual casi todo tipo de archivos (sonidos, vídeo, imágenes e incluso otras películas Flash), o la de *Configuración de Publicación* desde donde se pueden modificar las características de la publicación. También permite configurar la impresión de las páginas, imprimirlas...
- **Edición**: Es el clásico menú que te permite *Cortar, Copiar, Pegar*... tanto objetos o dibujos como fotogramas; también permite personalizar algunas de las opciones más comunes del programa.
- **Ver:** Además de los típicos Zooms, te permite moverte por los fotogramas y por las escenas. También incluye la posibilidad de crear una cuadrícula y unas guías. Esto se puede

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

seleccionar desde los submenús *Cuadrícula y Guías* desde donde también se pueden configurar sus opciones.

- **Insertar**: Permite insertar objetos en la película, así como nuevos fotogramas, capas, acciones, escenas...
- **Modificar**: La opción *Transformar* permite modificar los gráficos existentes en la película, y la opción *Trazar Mapa de Bits* convierte los gráficos en mapas vectoriales (este tema se tratará más adelante). El resto de opciones permite modificar características de los elementos de la animación *Suavizar, Optimizar* o de la propia película (*Capa, Escena* ...).
- **Texto**: Sus contenidos afectan a la edición de texto. Más adelante se tratará en profundidad.
- **Comandos**: Permite administrar los Comandos (conjunto de sentencias almacenadas que permiten emular lo que un usuario pueda introducir en el entorno de edición) que hayamos almacenado en nuestra animación, obtener otros nuevos de la página de Macromedia o ejecutar los que ya tengamos.
- **Control**: Desde aquí se modifican las propiedades de reproducción de la película. *Reproducir, Rebobinar, Probar Película....*
- **Depurar**: Aquí encontraremos las opciones de depuración de la película que te ayudarñan a encontrar errores de progrmación en ActionScrit, entre ellos se encuentran *Entrar, Pasar, Salir, Continuar...*
- **Ventana**: Este menú, además de las opciones clásicas acerca de cómo distribuir las ventanas, incluye accesos directos a todos los paneles y también la posibilidad de crear tus propios *Espacios de Trabajo*, guardarlos, cargarlos, etc.
- **Ayuda**: Desde aquí podemos acceder a toda la ayuda que nos ofrece Adobe, desde el manual existente, hasta el diccionario de Action Script, pasando por tutoriales, lecciones guiadas etc...

### 7.1.2 La Línea de Tiempo

La *Línea de Tiempo* representa una forma de ver los fotogramas de modo simplificado. Consta de 2 partes.

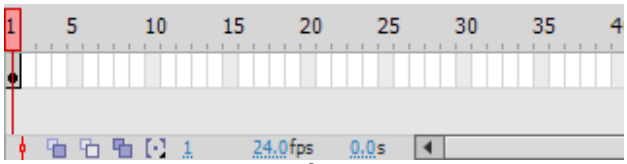



**Figura 118 - Línea de tiempo**

1) Los *Fotogramas (frames)* que vienen delimitados por líneas verticales (formando rectángulos)
2) Los *Números de Fotograma* que permiten saber qué número tiene asignado cada fotograma, cuánto dura o cuándo aparecerá en la película.



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Además, en la parte inferior hay herramientas para trabajar con *Papel cebolla* e información sobre el *Número de Fotograma actual* (1 en la imagen), la *Velocidad de los Fotogramas* (24.0 en la imagen) y el *Tiempo de película transcurrido* (0.0s en la imagen).
A nivel conceptual, la *Línea de Tiempo* representa la sucesión de Fotogramas en el tiempo. Es decir, la película Flash no será nada más que los fotogramas que aparecen en la *Línea de Tiempo* uno detrás de otro, en el orden que establece la misma Línea de Tiempo.

### 7.1.3 Las Capas

El concepto de **capa** es fundamental para manejar Flash de forma eficiente. Dada la importancia de estas, se le dedicará un tema completo. Aún así, veamos a grandes rasgos qué son las capas.
Una **capa** se puede definir como una película independiente de un único nivel. Es decir, una **capa** contiene su propia Línea de Tiempo (con infinitos fotogramas).
Los objetos que estén en una determinada **capa** comparten fotograma y por tanto, pueden "mezclarse" entre sí. Esto es interesante a menudo, pero otras veces es conveniente separar los objetos de modo que no interfieran entre sí. Para ello, crearemos tantas capas como sea necesario. El uso de múltiples capas además, da lugar a películas bien ordenadas y de fácil manejo (es conveniente colocar los el código ActionScript en una capa independiente llamada "acciones", por ejemplo).

### 7.1.4 Escenario

A continuación veremos las diferentes partes que conforman el Área de trabajo empezaremos con la más importante: el **Escenario**.
Sobre el escenario dibujaremos y colocaremos los diferentes elementos de la película que estemos realizando. Las propiedades del escenario coinciden con las **Propiedades del documento**. Para acceder a ellas, se hace clic con el botón derecho sobre cualquier parte del escenario en la que no haya ningún objeto y después sobre **Propiedades del documento**:

- Se puede asignar un **Título** y **Descripción**.
- **Dimensiones**: Determinan el tamaño de la película. El tamaño mínimo es de 1 x 1 píxeles y el máximo de 2880 x 2880 píxeles.
- **Coincidir**: Provocan que el tamaño de la película coincida con el botón seleccionado (tamaño por defecto de la Impresora, Contenidos existentes o los elegidos como Predeterminados)
- **Color de Fondo**: Selecciona el color de fondo de toda la película.
- **Velocidad de fotogramas**: número de fotogramas por segundo que aparecerán en la película.
- **Unidades de Regla**: Unidad que se empleará para medir las cantidades.
- **Transformar en predeterminado**: Este botón, permite almacenar las propiedades del documento actual y aplicarlas a todos los documentos nuevos que se creen a posteriori.

### 7.1.5 Paneles

Los **Paneles** son conjuntos de comandos agrupados según su función (por ejemplo, todo lo que haga referencia a las acciones, irá en el Panel **Acciones**). Su misión es simplificar y facilitar el uso de los comandos.
Estos paneles o ventanas se encuentran organizados en el lateral derecho del área de trabajo, pudiéndose expandir o minimizar con un clic de ratón.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Si no están visibles en el lateral derecho, puedes visualizarlos desplegando el menú **Ventana** y haciendo clic sobre el nombre del panel que quieras mostrar.
Para mejorar la experiencia del usuario, Flash permite a éste organizar los paneles y guardar esta configuración. Para ello deberás acceder a la opción **Ventana → Espacio de trabajo → Nuevo espacio de trabajo** y asignar un nombre a nuestro nuevo espacio.
Los paneles disponibles son:

- **Panel Alinear**: Coloca los objetos del modo que le indiquemos.
- **Panel Color**: Mediante este panel creamos los colores que más nos gusten.
- **Panel Muestras**: Nos permite seleccionar un color de modo rápido y gráfico. (Incluidas nuestras creaciones).
- **Panel Información**: Muestra el tamaño y las coordenadas de los objetos seleccionados, permitiéndonos modificarlas.
- **Panel Escena**: Modifica los atributos de las escenas que usemos.
- Si aún no sabes lo que son las Escenas, te lo explicamos en nuestro tema básico .
- **Panel Transformar**: Ensancha, encoge, gira ... los objetos seleccionados.
- **Panel Acciones**: De gran ayuda para emplear Action Script y asociar acciones a nuestra película.
- **Panel Comportamientos**: Permiten asignar a determinados objetos una serie de características (comportamientos) que después podrán almacenarse para aplicarse a otros objetos de forma rápida y eficaz.
- **Panel Componentes**: Nos permite acceder a los Componentes ya construidos y listos para ser usados que nos proporciona Flash. Los componentes son objetos "inteligentes" con propiedades características y muchas utilidades (calendarios, scrolls etc...)
- **Panel Cadenas**: Mediante este panel Flash aporta soporte multi-idioma a nuestras películas.
- **Panel Respuestas**: Macromedia pone a nuestra disposición ayuda y consejos accesibles desde este panel.
- **Panel Propiedades**: Nos muestra las propiedades del objeto seleccionado en ese instante, color de borde, de fondo, tipo de trazo, tamaño de los caracteres, tipografía, propiedades de los objetos (si hay interpolaciones etc...), coordenadas, tamaño etc...
- **Panel Explorador de Películas**: Nos permite acceder a todo el contenido de nuestra película de forma fácil y rápida


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.2 Dibujar y colorear

### 7.2.1 Barra de Herramientas

La Barra de Herramientas contiene todas las Herramientas necesarias para el dibujo. Algunas herramientas llevan asociados modificadores de herramientas, que permiten cambiar algunas propiedades de la herramienta, y que aparecen al final del panel de herramientas. Veamos las herramientas más importantes:

**Figura 119 - Barra de herramientas**

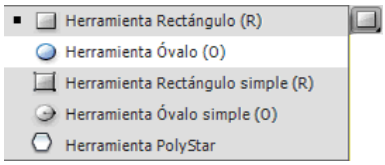
- **Herramienta Selección (flecha)**:  Es la herramienta más usada de todas. Su uso principal es para seleccionar objetos, permite seleccionar los bordes de los objetos (con doble clic), los rellenos (con un sólo clic), zonas a nuestra elección... Su uso adecuado puede ahorrarnos tiempo en el trabajo.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Herramienta Texto:**  Crea un texto en el lugar en el que hagamos clic. Sus propiedades se verán en el tema siguiente.

- **Herramienta Línea:**  Permite crear líneas rectas de un modo rápido. Las líneas se crean como en cualquier programa de dibujo, se hace clic y se arrastra hasta donde queramos que llegue la línea recta. Una vez creada la podemos modificar situando el cursor encima de los extremos para estirarlos y en cualquier otra parte cercana a la recta para curvarla.

- **Herramienta de forma** 

:Permite dibujar formas predefinidas como las que aparecen en la imagen, la última herramienta utilizada es la que aparece como icono en la barra de herramientas. Manteniendo pulsado el icono podremos acceder a las diferentes herramientas de forma. Por ejemplo:

- **Herramienta** Óvalo permite dibujar círculos o elipses de manera rápida y sencilla.

- **Herramienta Lápiz:**  Es la primera herramienta de dibujo propiamente dicho. Permite dibujar líneas con la forma que decidamos, modificando la forma de estas a nuestro gusto. El color que aplicará esta herramienta se puede modificar, bien desde el Panel Mezclador de Colores o bien desde el subpanel Colores que hay en la Barra de Herramientas**.**

- **Herramienta Pincel:**  Su funcionalidad es parecida a la del lápiz, pero por defecto su trazo es más grueso e irregular. Se suele emplear para aplicar rellenos. Se puede modificar su herramientas.

- **Herramienta Cubo de pintura:**  Permite aplicar rellenos a los objetos que hayamos creado. Al contrario que muchos otros programas de dibujo, no permite aplicar rellenos si la zona no está delimitada por un borde. El color que aplicará esta herramienta se puede modificar, bien desde el Panel Color o bien desde el subpanel Relleno que hay en la Barra de Herramientas.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Herramienta Borrador:**  Su funcionamiento es análogo a la Herramienta Pincel. Pero su función es la de eliminar todo aquello que "dibuje".

**Herramientas avanzadas**

- **Herramienta Lazo:**  Su función es complementaria a la de la herramienta Flecha, pues puede seleccionar cualquier cosa, sin importar la forma, (la Herramienta Flecha sólo puede seleccionar objetos o zonas rectangulares o cuadradas). En contrapartida, la Herramienta Lazo no puede seleccionar rellenos u objetos (a menos que hagamos la selección a mano).

  Al seleccionar esta herramienta, en el **Panel de Herramientas** aparecen estos botones:  . Esto es la herramienta **Varita Mágica**, tan popular en otros programas de dibujo. Permite hacer selecciones según los colores de los objetos. El tercer dibujo que aparece es este:  para seleccionar **Polígono**.

- **Herramienta Pluma:**  Crea polígonos (y por tanto rectas, rectángulos...) de un modo sencillo. Mucha gente encuentra esta herramienta complicada, aunque es una de las más potentes que ofrece Flash. Su empleo consiste en hacer clic en los lugares que queramos definir como vértices de los polígonos, lo que nos asegura una gran precisión. Para crear curvas, hay que señalar los puntos que la delimitan y posteriormente trazar las tangentes a ellas. Con un poco de práctica se acaba dominando.

- **Herramienta Subseleccionador:**  Esta Herramienta complementa a la Herramienta Pluma, ya que permite mover o ajustar los vértices que componen los objetos creados con dicha herramienta.

- **Herramienta Bote de Tinta:**  Se emplea para cambiar rápidamente el color de un trazo. Se aplica sobre objetos, si tienen borde, cambia al color mostrado de dicho borde, por el mostrado en el Panel Mezclador de Colores (que coincide con el subpanel Colores que hay en la Barra de Herramientas.)

- **Herramienta Cuentagotas:**  Su misión es "capturar" colores para que posteriormente podamos utilizarlos.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

Algunas Herramientas poseen unas opciones especiales que facilitan y potencian su uso. Para acceder a estas utilidades, a veces no basta con hacer clic en la Herramienta correspondiente sino que además debemos hacer clic en la línea o en el objeto que has dibujado.

Entonces aparecerá (o se iluminará si ya estaba presente) un submenú como este:

- **Ajustar a Objetos :**  Se usa para obligar a los objetos a "encajar" unos con otros, es decir, para que en caso de ser posible, sus bordes se superpongan, dando la sensación de estar "unidos".
- **Suavizar:**  Convierte los trazos rectos en líneas menos rígidas.
- **Enderezar:**  Realiza la labor inversa. Convierte los trazos redondeados en más rectilíneos.

### 7.2.2 Panel Color

El **Panel Color**, se usa para fabricar nuestros propios colores y para seleccionar los que más nos gusten.

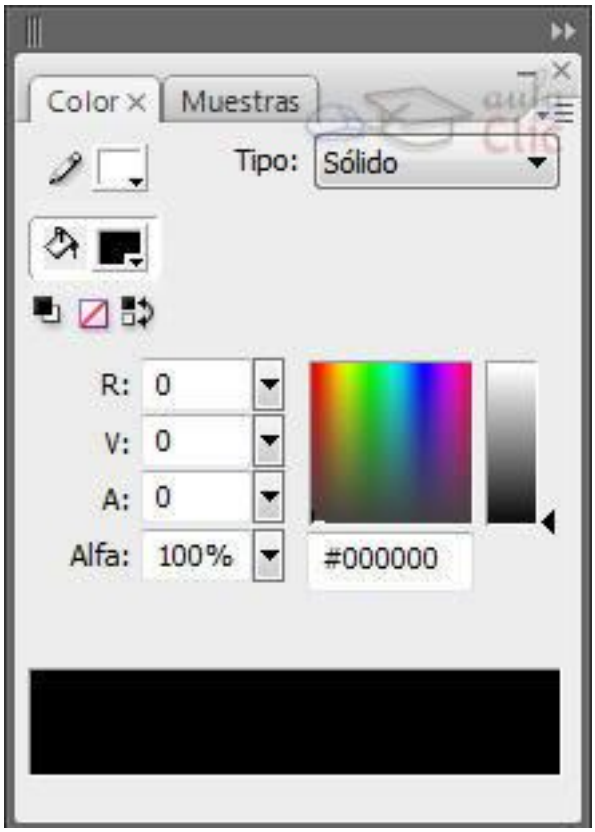
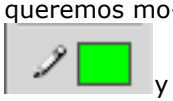
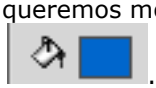



**Figura 120 - Panel Color**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Para seleccionar un color determinado, bastará con hacer clic en las pestañas que se encuentran junto a los iconos de las herramientas de Lápiz y de Bote de Pintura. (Si queremos modificar el color de un borde, pulsaremos sobre el color que está junto al lápiz  y si queremos modificar un relleno, haremos clic en el color que está junto al bote de pintura  . Al hacerlo aparecerá un panel con multitud de colores para que seleccionemos el que más nos gusta. También permite introducir el código del color según el estándar

Así mismo se puede determinar el tipo de relleno que aplicaremos a los objetos creados (mediante la herramienta Bote de Pintura).

Se pueden crear diferentes tipos de relleno:

- **Sólido**: Consiste en un relleno formado por un solo color.
- **Degradado Lineal**: Es un tipo especial de relleno, de modo que un color se degrada hasta convertirse en otro. Puede ir de arriba abajo o de un lado al otro
- **Degradado Radial**: Es igual que el anterior, pero los degradados tiene forma circular.
- **Mapa de Bits**: Permite colocar como relleno alguna imagen existente en la película (O ajena a ella si antes se "importa")

### 7.2.3 Panel Muestras

El **Panel Muestras** sirve para poder ver de un modo rápido y claro los colores de que disponemos, tanto sólidos (un solo color) como degradados (lineales o radiales). Además, cuando creemos un color mediante el **Panel Color**, podremos agregarlo a nuestro conjunto de muestras mediante **Agregar Muestra** (que se encuentra en un menú desplegable en la parte superior derecha del panel Color). Una vez esté agregado el color, pasará a estar disponible en nuestro conjunto de muestras y podremos acceder a él rápidamente cada vez que trabajemos con nuestra película.
Cada película tiene su propio conjunto de muestras y cada vez que la abramos para editarla, podremos usar las muestras que teníamos la última vez que trabajamos con dicha película.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

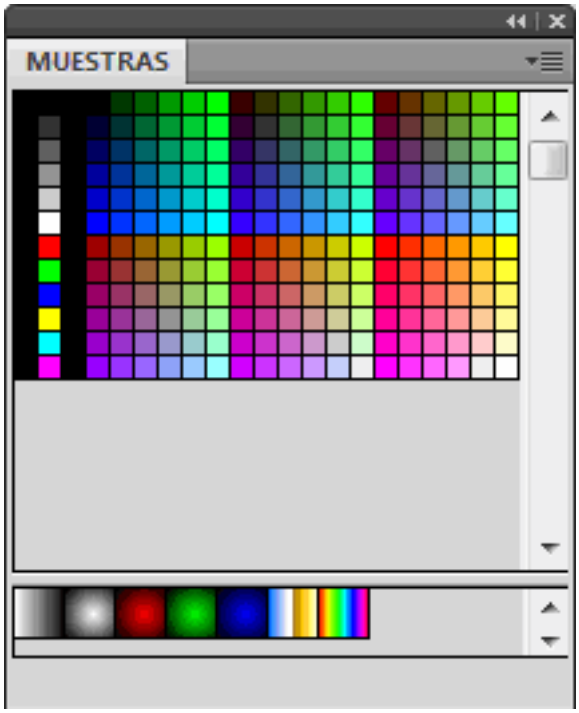


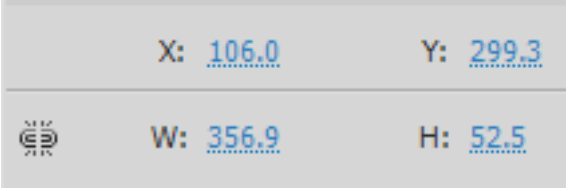

**Figura 121 - Panel Muestras**

## 7.3 Textos

Flash abarca todo aquello que nos pueda hacer falta a la hora de crear una animación, y por tanto, también todo aquello relativo a los textos. Sin embargo, Flash fue concebido para crear animaciones gráficas, de modo que tratará cualquier texto como si de un objeto más se tratase, listo para ser animado a nuestro gusto. Esto nos permitirá posteriormente animar textos y crear espectaculares animaciones con muy poco esfuerzo. Flash distingue entre 3 tipos de texto, texto estático o normal, texto dinámico y texto de entrada (para que el usuario introduzca sus datos, por ejemplo), también se puede crear texto que soporte formato HTML etc...

### 7.3.1 Propiedades de los textos

Para poder escribir, deberemos hacer clic en la **Herramienta Texto** y posteriormente en el punto del escenario en el que queramos empezar a escribir.
El **Panel Propiedades** contiene las principales propiedades de todos los objetos que empleemos durante nuestra película, de modo que si seleccionamos un texto, podremos ver en él todo lo que nos hace falta conocer sobre nuestro texto, respecto a la anterior versión la interfaz de las propiedades de texto a cambiado considerablemente.




- **Posición y Tamaño**:

  Dado que Flash trata los textos como objetos, éstos también tienen anchura, altura y coordenadas. Podemos modificarlos a nuestro gusto, pero debemos tener en cuenta que los


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

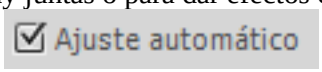
caracteres no deben estirarse sin motivo, pues las tipografías son una parte muy importante y deben respetarse.

- **Familia**: 


  Desde aquí, al igual que en los editores de texto más comunes podemos seleccionar el tipo de letra o "fuente" que más nos guste.

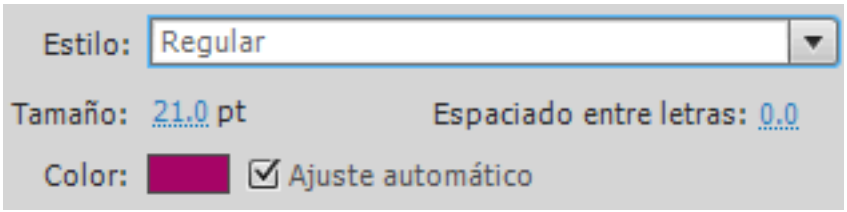
- **Espaciado entre letras**: 


  Determina el espaciado entre los caracteres. Útil cuando la tipografía que empleamos muestra las letras muy juntas o para dar efectos concretos al texto.

- **Ajuste automático**: 


  Activar esta casilla provoca que la separación entre caracteres se realice de modo automático.

- **Configuración:**


  Permiten convertir el texto en Negrita (B), Cursiva (I), cambiar el color del texto y el tamaño.

- **Suavizado:** Las opciones de suavizado resultan importantes a la hora de mostrar nuestro texto. Encontramos estas opciones:
  - *Utilizar fuentes de dispositivo*: Se emplean las fuentes de la impresora, lo que hace más liviano el archivo SWF al no incluir la fuente. Aunque nos exponemos a que la fuente no esté instalada, por lo que se empleará otra. Sólo deberíamos de utilizar esta opción para las más comunes.
  - *Texto de mapa de bits*, no realiza suavizado, ocupando más tamaño y perdiéndose resolución al redimensionar.
  - *Suavizado para animación*, es la mejor opción cuando pretendemos animar texto.
  - *Suavizado para legibilidad*, favorece la legibilidad, pero no debemos de emplearla en textos animados.
  - *Suavizado personalizado*, nos permite ajustarlo manualmente.

- **Orientación:** 

  Cambia la orientación del texto de horizontal a vertical, tanto de izquierda a derecha como de derecha a izquierda.

- **Seleccionable:** 

  Determina si el usuario pordrá seleccionar el texto.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

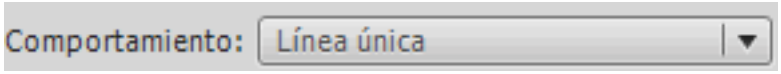
- **Posición:** 

  Nos permite convertir nuestro texto en subíndices o en superídices.
- **Vínculo:**

  Si queremos que al pulsar sobre nuestro texto, el usuario vaya a una dirección web, nada más fácil que escribirla ahí. Flash la asociará al texto que estés escribiendo en ese momento.
- **Destino:**

  Determina si la URL a la que el texto hace referencia se cargará en la misma ventana del navegador, en una nueva ...
- **Comportamiento:** 
  

  Si el texto que vamos a incluir es dinámico (de lo contrario aparece sin activar), esta opción nos permite determinar cómo queremos que aparezcan las líneas (línea única, multilínea o multilínea sin ajuste).

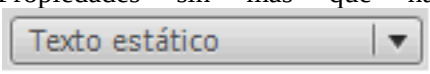
  Un **Párrafo** no es más que un conjunto de caracteres con propiedades comunes a todos ellos. Estos párrafos admiten ciertas opciones que nos permiten trabajar con bloques de texto. El **Panel Propiedades** nos proporciona las siguientes opciones para trabajar con párrafos (entre otras).
- **A la Izquierda**:  Todas las líneas empezarán tan a la izquierda como sea posible (dentro del recuadro de texto que hayamos definido).
- **Centrar**:  Las líneas se distribuyen a la derecha y a la izquierda del punto medio del párrafo.
- **A la derecha**:  Todas las líneas empezarán tan a la derecha como sea posible dentro del recuadro de texto que hayamos definido).
- **Justificado**:  El texto se ensancha si es necesario con tal de que no quede "dentado" por ninguno de sus límites.
- El resto de opciones nos permiten determinar los márgenes (izquierdo y derecho), los sangrados de párrafo y el espacio interlineal.



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

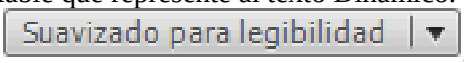
### 7.3.2 Tipos de textos

Como ya hemos comentado, Flash distingue entre diversos tipos de textos y les da un tratamiento especial, según el tipo que sean.
Todos los tipos de textos responden a las propiedades comentadas en los puntos anteriores, y es en el Tipo de texto en lo que se diferencian. El tipo de texto se puede modificar desde el Panel Propiedades sin más que haciendo clic sobre la pestaña Tipo de texto: 


### 7.3.3 Texto estático

El **Texto Estático** se caracteriza por no presentar ningún cambio a lo largo de la animación. Es importante que no confundamos la palabra "estático" con que el texto no se mueva o malinterpretemos la frase es el texto que no presenta cambios a lo largo de la animación.
Lo que queremos decir es que no cambia el contenido del recuadro de texto, es decir, que el texto puede estar animado (girar, cambiar de color...) y sin embargo ser estático.

### 7.3.4 Texto dinámico

El **Texto Dinámico** en contraposición al estático sí que puede cambiar su contenido (además de estar animado). Su uso es bastante más complejo que el del **Texto Estático**, ya que cada recuadro de **Texto Dinámico** puede ser una variable modificable mediante ActionScript, esto quiere decir que los valores y propiedades de este tipo de textos se pueden modificar mediante programación, lo que nos saca del objetivo de este curso. Un uso común que suelen tener es el de representar los textos introducidos mediante **Textos de Entrada**.
Tienen multitud de propiedades, accesibles desde el **Panel Propiedades**, se puede decidir el número de líneas que van a tener, se puede introducir texto HTML, se puede añadir fácilmente un borde al texto o dar nombre a la variable que represente al texto Dinámico.
Una opción interesante es el botón 

, que nos permite indicar qué caracteres incluir en el archivo SWF. Aquí debemos de indicar los carácteres que esperamos que muestre el texto dinámico.
Al editar nuestra película, distinguieremos los textos dinámicos por aparecer enmarcados en una línea discontinua.

### 7.3.5 Introducción de texto

El **Texto de Entrada** o **Introducción de Texto** tiene básicamente las mismas propiedades que el **Texto Dinámico**, junto con algunas propias de un tipo de texto orientado a la introducción de datos por parte de usuario, como por ejemplo el número máximo de líneas que puede introducir en ese campo de texto o si queremos que lo que el usuario escriba en dicho campo aparezca como asteriscos (para las contraseñas).
Evidentemente este tipo de texto se reflejará en nuestra película como un recuadro de texto SIN contenido, ya que lo deberá introducir el usuario.
Como hemos mencionado antes, este tipo de texto se puede combinar con el **Texto Dinámico.**
Flash permite crear animaciones de texto tan interactivas como las que pueden crear con imágenes (ya hemos visto lo sencillo que es convertir un texto en un enlace a otra página web). Aún así, crear animaciones con textos, que sobrepasen a las que Flash incorpora por defecto (colores, transparencias, brillos, rotaciones, movimientos o combinaciones de estos) puede resultar un trabajo de muchísimas horas y mucha paciencia.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

### 7.3.6 Incorporar Fuentes

Al crear un archivo de flash, y utilizar texto dinámico o textos de entrada, tenemos que tener presente que si hemos elegido una fuente que no está disponible en el sistema que se reproduce la película, se sustituirá por otra fuente disponible. Esto es un problema, ya que puede alterar bastante nuestro diseño.

Para evitar esto, podemos incluir las fuentes que hemos empleado en nuestro proyecto. Esto garantiza que el texto se vea tal y como lo creamos, independientemente de las fuentes del sistema. Esto sólo será necesario hacerlo para los textos dinámicos, ya que el texto estático se guarda como un gráfico vectorial, y no habrá problemas.

Para incorporar una fuente en la película hacemos clic derecho en alguna parte vacía de la **Biblioteca**, y seleccionamos **Nueva fuente.** Se abrirá el siguiente cuadro de diálogo:

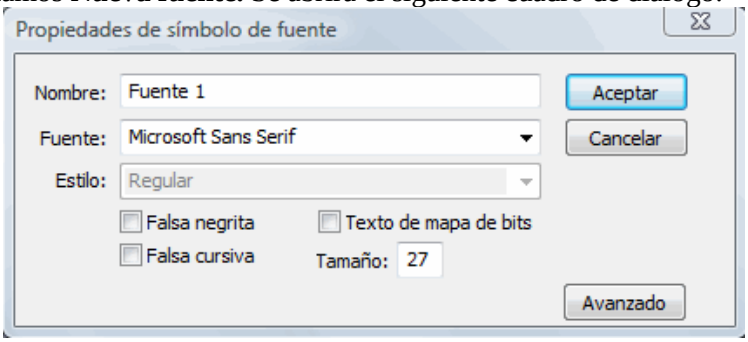



**Figura 122 - Propiedades de Fuente**

En el desplegable **Fuente** seleccionamos la fuente que deseamos utilizar. Seleccionamos los estilos que vamos a utilizar (Negrita, Cursiva y Tamaño).

El nombre que le demos se mostrará en la **Biblioteca** y estará lista para utilizarse. Cuando creemos un campo de texto dinámico o de entrada, seleccionamos nuestra fuente, que aparecerá señalada con un asterisco (*).

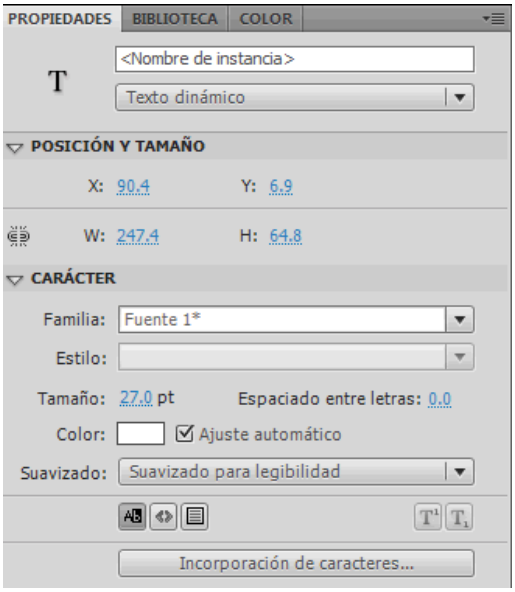



**Figura 123 - Definición de Fuentes**



ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.4 Sonidos

Hasta hace poco, los únicos sonidos que oíamos en las páginas Web eran los famosos "midis", de escaso tamaño y de escasa calidad. De hecho, eran sólo instrumentos musicales sin voz, de ahí su pequeño tamaño (y calidad). Aún así, siempre existía algún creador de páginas Web que se aventuraba a poner algún sonido complejo (.wav o .mp3) en su página web, su carga es tan lenta, que la mayoría de los visitantes se irían de la página sin llegar a escucharla.
Hoy, gracias a las conexiones de banda ancha orientadas a contenido multimedia, poner sonido a las páginas web es un poco más fácil, ¿qué aporta FLASH?
Flash nos permite insertar cualquier sonido que queramos en nuestras películas (.wav, .aiff, .mp3, etc...) de forma fácil y muy efectiva, ya que es capaz de acelerar la descarga del sonido siempre y cuando se descargue junto con nuestra película.
Podemos dar a la película efectos simples (el típico "clic" al pulsar un botón), efectos complejos (música de fondo) e incluso podemos hacer que la animación se desarrolle conjuntamente con una música (si bien esto último necesitaría que toda la película estuviera descargada previamente, para evitar "atascos" durante el desarrollo de la animación). En definitiva, Flash nos lo vuelve a poner fácil.

### 7.4.1 Importar Sonidos

Si alguna vez habeis intentado añadir un sonido a vuestra animación Flash probablemente os hayais llevado una gran decepción al no conseguirlo. Esto se debe a que no se tiene en cuenta que para poder emplear un objeto en nuestra película, o bien lo creamos nosotros (como llevamos haciendo hasta ahora) o bien lo conseguimos de cualquier otro modo y lo insertamos en nuestra película.
A esto último se le llama "**Importar**" y se puede hacer con sonidos, gráficos, e incluso con otras películas Flash. Importar por tanto, no es más que decirle a Flash que añada un determinado archivo a nuestra película, para que podamos usarlo cuando queramos. En realidad, lo añade a nuestra **Biblioteca**, que es el Panel en el que están todos los objetos que participan en la película (este Panel se verá más adelante).
Así pues si queremos manejar un sonido en nuestra película, deberemos importarlo previamente.
Para importar un sonido haz clic en el menú **Archivo → Importar → Importar a biblioteca.**
Se abrirá el cuadro de diálogo de **Importar a biblioteca**. Allí deberás seleccionar en **Tipo de archivo**, **Todos los formatos de sonido**. Se selecciona el archivo de audio haciendo clic sobre él y se pulsa el botón **Aceptar**.
El sonido estará listo para usarlo donde quieras, podrás encontrarlo en la **Biblioteca** (menú **Ventana → Biblioteca**).

### 7.4.2 Propiedades de los Sonidos

En Flash, todo lo referente a los sonidos lo podemos editar desde el Panel **Propiedades**. Aquí tenemos todo lo necesario para insertar, modificar y editar el sonido que se ha importado.
Para que aparezca la posibilidad de trabajar con sonidos, deberemos hacer clic en seleccionar el fotograma en el que queremos reproducir el sonido.
Veamos las partes que tiene este panel.

- **Nombre:** En este desplegable nos aparecerán las canciones que tenemos importadas, deberemos seleccionar la canción que pretendamos añadir a nuestra película (en el siguiente punto veremos cómo insertarlo). Si no queremos reproducir sonido, seleccionamos **Ninguno**.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Efecto**: Desde aquí podremos añadir algún efecto a nuestro sonido Si deseamos añadir efectos sonoros complejos, deberemos tratar adecuadamente el sonido con algún programa creado específicamente para este propósito antes de importarlo.
- **Sinc:** Esta opción nos permite determinar en qué momento comenzará a actuar nuestro sonido, estas son las opciones que tenemos:
    - **Evento:** Sincroniza nuestro sonido con un evento determinado. Es la opción por defecto y provoca que el sonido se empiece a reproducir al pasar la película por el fotograma en el que está situado. También se puede sincronizar el sonido con botones y los demás tipos de símbolos. El sonido se repetirá completamente, y si lo ejecutamos varias veces, se reproducirá de nuevo, aunque no hubiese acabado, provocando que el sonido se solape.
    - **Inicio:** Su funcionamiento es equivalente al de "Evento", se diferencian en que si ya se está reproduciendo, no se reproduce de nuevo.
    - **Detener:** Detiene el sonido seleccionado.
    - **Flujo:** Esta opción sincroniza el sonido con el o los objetos con los que esté asociado, por tanto, si la carga de la película es lenta y las imágenes no fluyen adecuadamente, el sonido se detendrá para sincronizarse con ellas. Este efecto puede dar la sensación de que la película se corta de un modo muy brusco. Por otra parte, es un efecto muy adecuado para algunas situaciones, por ejemplo, el efecto de que un personaje hable durante una película. En esta situación, es muy recomendable que el sonido y las imágenes estén sincronizadas.
    - **Repetir:** Determina el número de veces que se reproducirá el sonido según lo que escribas en la caja de texto de la derecha. También puedes seleccionar **Reproducir indefinidamente** para que el sonido se reproduzca en un bucle hasta llegar al siguiente fotograma clave.

### 7.4.3 Insertar un Sonido

Para que el sonido se reproduzca en la película, lo mejor será crear una nueva capa para cada sonido. Así lo controlaremos fácilmente. Después, se arrastra el sonido de la biblioteca al escenario, sobre la nueva capa.
Con el primer fotograma de la nueva capa seleccionado, accedemos a sus propiedades, a la sección de **Sonido**. Seleccionamos el Nombre del archivo, los efectos deseados, y las veces que lo queremos **Repetir**.
En el menú **Sinc**. seleccionamos la opción de sincronización deseada.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

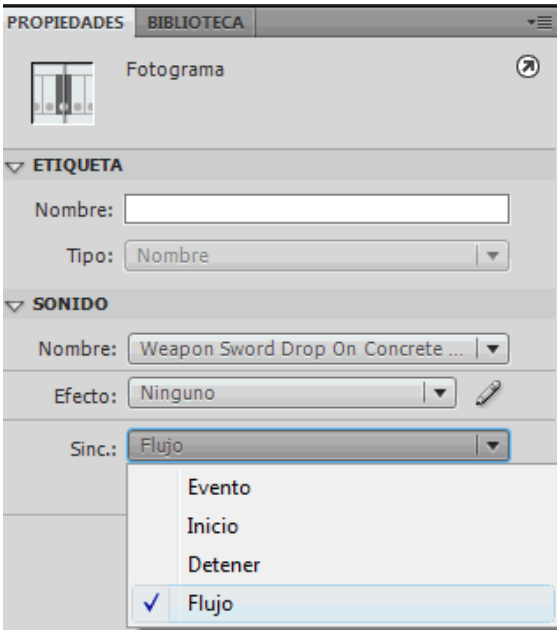



**Figura 124 - Propiedades del sonido**

Al seleccionar la opción **Flujo**, el sonido se reproducirá hasta que se acabe o llegue a un fotograma clave, siguiendo la película. Sólo es aconsejable esta opción si realmente la necesitamos.
Como los clips de película y botones tienen líneas de tiempo propias, como ya veremos, podemos insertar ahí los sonidos, por ejemplo, para hacer que al pulsar un botón se reproduzca un clic.

### 7.4.4 Editar Sonidos

Flash no fue creado como un editor de sonidos, por lo que su potencia en este campo es limitada. Aún así, los efectos que permite aplicar de un modo sencillo suelen ser suficientes para cualquier animación que queramos realizar. Disponemos de estos efectos:

- **Canal Izquierdo**: El sonido tan sólo se escuchará por el altavoz izquierdo.
- **Canal Derecho**: El sonido tan sólo se escuchará por el altavoz derecho.
- **Desvanecimiento de izquierda a derecha**: El sonido se reproduce inicialmente en el altavoz izquierdo para luego pasar al derecho.
- **Desvanecimiento de derecha a izquierda**: El sonido se reproduce inicialmente en el altavoz derecho para luego pasar al izquierdo.
- **Aumento Progresivo**: El volumen de nuestro sonido aumenta progresivamente.
- **Desvanecimiento**: El volumen de nuestro sonido disminuye progresivamente.
- **Personalizado**: Esta opción nos permite "editar" el sonido de un modo rápido y sencillo. Desde este editor podemos decidir qué volumen tendrá nuestro sonido y en qué altavoz. Podemos crear los efectos anteriores ajustándolos a nuestro gusto y podemos crear efectos más complejos a nuestra medida.

### 7.4.5 ¿Mp3 o Wav?

Como ya se ha comentado, los sonidos que Flash puede importar casi cualquier tipo de sonido, aunque los más empleados son MP3 y WAV. Por tanto, cuando queramos añadir un sonido a

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

nuestra película, deberemos decantarnos por uno de estos 2 formatos. Lo normal es que el sonido que pretendamos insertar ya esté en uno de ellos, pero no debemos olvidar que existen multitud de programas que convierten un sonido con extensión .mp3 a uno .wav y viceversa, por lo que no debe ser inconveniente el formato que tenga el sonido en un principio.
Si decidimos que nuestra película tenga sonidos, deberemos asumir una carga muy importante en cuanto a tamaño de la película y en consecuencia en cuanto a tiempo de descarga. Lo habitual es que los sonidos ocupen más de la mitad del espacio total y muchas veces no valdrá la pena insertarlos...
La lógica nos dice que insertemos el sonido que ocupe un espacio menor en el disco duro, ya que este espacio es el que ocupará en nuestra película. Esto es una aproximación, ya que Flash comprime todo aquello que insertamos en nuestras películas.
Como ya sabemos, los sonidos .mp3 ocupan un espacio mucho menor que los sonidos .wav (10 veces menos o más), por lo que parece recomendable insertar .mp3 en lugar de .wav, ahora bien ¿Puede Flash comprimir un sonido ya comprimido? Es decir, si un sonido .wav ha sido comprimido y ahora tiene extensión .mp3, ¿podrá Flash volver a comprimirlo? La respuesta es NO. Flash comprime los sonidos que insertamos en nuestras películas, pero si el sonido resulta estar ya comprimido, no podrá volver a comprimirlo (en realidad si que consigue comprimirlo, pero el sonido final es el mismo que el inicial).

## 7.5 Trabajar con objetos

### 7.5.1 Los objetos. Iniciación

En una página Web, en un catálogo para un DVD o en cualquier otra cosa, tendremos que trabajar con objetos. A grandes rasgos, podremos considerar un objeto todo aquello que aparezca en nuestra película y sea visible, de modo que podamos trabajar con él, por ejemplo, cualquier imagen que creemos o importemos, un botón, un dibujo creado por nosotros mismos etc...
Los objetos así considerados tienen 2 partes fundamentales:

- **El Borde:** Consiste en una delgada línea que separa el objeto del exterior, del escenario. Puede existir o no, según nos convenga. Cuando creamos un objeto, el borde se crea siempre y su color será el indicado en el **Color de Trazo** (dentro del **Panel Mezclador de Colores**). Si queremos dibujar creando Bordes deberemos emplear las **Herramientas Lápiz, Línea o Pluma** y si queremos que nuestro dibujo no tenga borde, bastará con seleccionar el borde y suprimirlo (ver siguiente punto).
- **El Relleno:** El relleno no es más que el propio objeto sin borde. Es, por tanto, la parte interna del objeto. Su existencia también es arbitraria, ya que podemos crear un objeto cuyo color de relleno sea transparente, como ya se vio en el tema de **Dibujar**, y por tanto, parecerá que dicho objeto no tiene relleno, aunque en realidad sí que exista pero sea de color transparente. Para dibujar **Rellenos** (sin borde) podemos usar herramientas tales como el **Pincel** o el **Cubo de Pintura**.

### 7.5.2 Seleccionar

Para poder trabajar con objetos, es fundamental saber seleccionar la parte del objeto que queramos modificar (mover, girar, cambiar de color...). Podremos observar que las partes de un objeto seleccionadas toman una apariencia con textura para indicar que están seleccionadas.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

**Figura 125 - Objeto SIN seleccionar**

**Figura 126 - Objeto con el BORDE seleccionado**

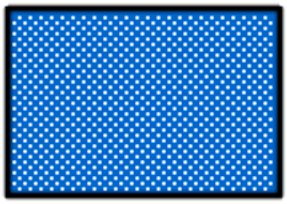


**Figura 127 - Objeto con el RELLENO seleccionado**

Veamos cómo seleccionar las diferentes partes de un objeto:

- **Seleccionar un Relleno o un Borde:** Basta hacer clic 1 vez en el Relleno o en el Borde que queramos seleccionar.
- **Seleccionar el Relleno y el Borde de un objeto**: Hacer doble clic en el Relleno.
- **Seleccionar todos los bordes (o líneas) de un mismo color que estén en contacto:** Hacer doble clic sobre una de las líneas que tenga el color que pretendamos seleccionar.
- **Seleccionar un Símbolo, un texto, o un grupo**: Clic en el Símbolo, en el texto o en el grupo. Estos tipos de objetos mostrarán un borde de color azul (por defecto, depende de la capa) al estar seleccionados.
- **Seleccionar Varios elementos**: Mantendremos pulsada la tecla SHIFT (Mayúsculas) mientras seleccionamos los objetos que queramos.
- **Seleccionar los objetos que se encuentran en una determinada zona**: Para ello usaremos la Herramienta Selección (flecha). Haremos clic en una parte del escenario y arrastraremos el cursor hasta delimitar el área que contenga los objetos que queremos seleccionar. Si esta área corta una forma, sólo quedará seleccionada la parte del objeto que esté

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

dentro del área que hemos delimitado. Si quisiéramos incluir dicho objeto, bastaría con usar la tecla SHIFT y seleccionar la parte del objeto que falta por seleccionar.

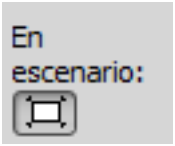
Este modo de seleccionar objetos permite seleccionar muchos objetos rápidamente, además de permitirnos seleccionar determinadas zonas de los objetos, para cortarlas, pegarlas ...

Además de la **Herramienta Flecha**, también podemos usar la Herramienta Lazo, más útil aún si cabe. Nos permite seleccionar cualquier zona de cualquier forma (la forma la determinamos nosotros) del objeto que sea. Al contrario que la **Herramienta Flecha**, las áreas seleccionadas no tienen porque ser rectángulares.

- **Seleccionar a partir de la Línea de Tiempo**: Si seleccionamos un determinado fotograma en la línea de tiempo, se seleccionan automáticamente todos los objetos que estén en dicho fotograma. Esto es útil para modificar de un modo rápido todos los elementos del fotograma.
- **Seleccionar Todo**: La forma más natural de seleccionar todo, consiste en hacer clic en el menú Edición → Seleccionar Todo.

### 7.5.3 Panel Alinear

Para colocar los objetos de un modo preciso (por no decir exacto), Flash pone a nuestra disposición el Panel Alinear. Este Panel lo podemos encontrar en el **Menú Ventana → Alinear**.
El **Panel Alinear** permite colocar los objetos tal y cómo le indiquemos. La opción **En Escena** permite decir a Flash que todas las posiciones que indiquemos para nuestros objetos tomen como referencia el escenario.




Si esta opción no está seleccionada, los objetos toman como referencia al conjunto de objetos en el que se encuentran, y se colocan en función de ellos. Lo más habitual es seleccionar **En Escenario**, para que los objetos se coloquen según los límites de la película, en el centro del fotograma etc...
Las distintas posibilidades del **Panel Alinear** son:

- **Alineamiento:**

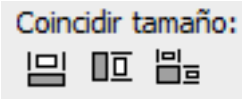

Sitúa los objetos en una determinada posición del fotograma (si está seleccionado En Escena). Las distintas opciones afectan a todos los elementos seleccionados y se emplean muy a menudo para situar los objetos en determinados sitios predefinidos.

- **Distribuir:**


Sitúa los objetos en el escenario en función de unos ejes imaginarios que pasan por cada uno de ellos, de modo que la distribución de los mismos sea uniforme.

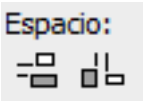
- **Coincidir Tamaño:**


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc




Hace coincidir los tamaños de los objetos. Si está activo "En Escena" estirará los objetos hasta que coincidan con el ancho y el largo de la película. Si no está activo, la referencia será el resto de objetos.

- **Espacio:**




Espacia los objetos de un modo uniforme.

## 7.6 Panel Información

Además de controlar la posición de los objetos desde el **Panel Alinear**, también podemos hacerlo, de un modo más exacto (más matemático) desde otro panel, el **Panel Información**.
A este Panel se puede acceder desde el Menú Ventana → **Información**. Las posibilidades de este Panel son limitadas, pero si buscamos exactitud en las medidas o no nos fiamos de las distribuciones de objetos que crea Flash, debemos acudir a él.

- **Medidas del Objeto:**  Aquí introduciremos un número que represente el tamaño de nuestro objeto en la medida seleccionada en las Propiedades del documento. An: hace referencia a la anchura y Al: a la altura.
- **Situación del objeto:**  Desde aquí controlamos la posición del objeto en el escenario. La X y la Y representan el eje de coordenadas (La X es el eje Horizontal y la Y el eje vertical). Las medidas también van en función de las medidas elegidas para la película.
- **Color Actual:**  Indica el color actual en función de la cantidad de Rojo (R), Verde (V), Azul (A) y efecto Alfa (Alfa) que contenga.

  Este indicador indica el color que tiene el objeto por el que en ese momento pasamos el cursor del ratón. Por tanto, podemos tener seleccionado un objeto (haciendo clic en él) y ver en el **Panel Información** su tamaño y su posición, pero al desplazar el ratón, el valor del color cambiará y ya no indicará el color del objeto seleccionado, sino el del objeto por el que pase el cursor. Tened esto en cuenta para no cometer errores o perder tiempo innecesario.
- **Posición del Cursor:**  Indica la posición del cursor. Es útil por si queremos que suceda algo en la película al pasar el cursor justo por una posición determinada o para situar partes del objeto en lugares específicos.

### 7.6.1 Los Grupos

Un **Grupo** no es más que un conjunto de objetos. Si bien no cualquier conjunto de objetos forman un grupo, ya que para crear un grupo, debemos indicarle a Flash que así lo queremos. Para ello,

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

basta seleccionar los objetos que queremos que formen parte de un grupo y después hacer clic en el Menú **Modificar → Agrupar** (Ctrl + G).
Tras hacer esto observaremos que desaparecen las texturas que indicaban que los objetos estaban seleccionados y observamos que el grupo pasa a ser un "todo", ya que resulta imposible seleccionar a uno de sus miembros sin que se seleccionen a su vez los demás. Además, aparece el rectángulo azul (por defecto) que rodea al grupo, definiéndolo como tal.
Crear grupos es muy útil, ya que nos permite, como ya hemos dicho, tratar al conjunto de objetos como un todo y por tanto, podemos aplicar efectos al conjunto, ahorrándonos la labor de hacerlo de objeto en objeto.
Al crear un grupo, simplemente estamos dando unas propiedades comunes a un conjunto de objetos y, en ningún caso perdemos nuestro objeto. En cualquier momento podemos deshacer el grupo, mediante el Menú **Modificar → Desagrupar**.
Además, Flash nos permite modificar los elementos de un grupo sin tener que desagruparlo. Para ello, seleccionamos el Grupo de elementos y hacemos clic en el **Menú Edición → Editar Seleccionado**. Podremos editar los objetos que componen el grupo por separado teniendo en cuenta que, como es lógico, los cambios realizados afectarán al grupo además de al elemento en cuestión.

## 7.7 Las Capas

### 7.7.1 Las Capas

Cada capa es un nivel en el que podemos dibujar, insertar sonidos, textos... con independencia del resto de capas. Hay que tener en cuenta que todas las capas comparten la misma **Línea de Tiempo** y por tanto, sus distintos fotogramas se reproducirán simultáneamente.
Otra razón para separar los objetos en capas, es que Flash nos obliga a colocar cada animación distinta en una capa. De lo contrario, todos los objetos que se encuentren en dicha capa formarán parte de la animación. Si queremos que un objeto no forme parte de una animación, deberemos quitarlo de la capa en la que se produce dicha animación.
Las capas además, tienen otras utilidades, nos permiten ordenar nuestra película de forma lógica, y nos ayudan en la edición de dibujos (evitando que se "fundan" en uno sólo, o bloqueando el resto de capas de modo que sólo podamos seleccionar la capa que nos interese).
Otro motivo es para organizar mejor nuestro contendio. Igual que creábamos una capa para los elementos de audio, crearemos capas para otros elementos, como el código ActionScript.

### 7.7.2 Trabajar con Capas

Veamos para qué sirven los distintos botones y cómo usarlos.

- **Nueva capa**  : Como su nombre indica, sirve para Insertar una nueva capa en la escena actual. Crea capas normales (en el siguiente punto se verán los distintos tipos de capas).
- **Crear carpeta**  : Nos sirve para crear carpetas, que nos ayudarán a organizar nuestras capas.
- **Borrar Capa**  : Borra la capa seleccionada.
- **Cambiar Nombre**: Para cambiar el nombre a una capa, basta con hacer doble clic en el nombre actual.
- **Propiedades de Capa**: Si hacemos doble clic en el icono  junto al nombre de la capa, podremos acceder a un panel con las propiedades de la capa en la que hayamos hecho


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

clic. Podremos modificar todas las opciones que hemos comentado anteriormente y alguna más de menor importancia.

Aquí puedes cambiar diferentes opciones sobre la capa, como su nombre o su color. También puedes bloquearla u ocultarla.

### 7.7.3 Trabajar con Capas. Opciones Avanzadas

- **Mostrar / Ocultar Capas**  : Este botón permite ver u ocultar todas las capas de la película. Es muy útil cuando tenemos muchas capas y sólo queremos ver una de ellas ya que permite ocultar todas a la vez, para después mostrar sólo la actual. Para activar la vista de una capa en concreto (o para ocultarla) basta con hacer clic en la capa correspondiente en el punto (o en la cruz) que se encuentra bajo el icono "Mostrar / Ocultar capas"

- **Bloquear Capas**  : Bloquea la edición de todas las capas, de modo que no podremos modificarlas hasta desbloquearlas. Para bloquear o desbloquear una capa concreta, procederemos como en el punto anterior, clic en el punto o icono "Cerrojo" situados en la capa actual bajo el icono "Bloquear Capas".

  Bloquear una capa es muy útil cuando tenemos varios objetos juntos y en capas distintas y queremos asegurarnos de que no modificamos "sin querer" alguno de ellos. Tras bloquear su capa podremos trabajar con la seguridad de no modificar dicho objeto, ni siquiera podremos seleccionarlo, de modo que editaremos con mayor facilidad el objeto que queramos.

- **Mostrar/Ocultar capas como contornos**  : Este botón nos muestra/oculta los contenidos de todas las capas como si sólo estuviesen formados por bordes. De este modo y ante un conjunto numeroso de objetos, podremos distinguirlos a todos de forma fácil y podremos ver en qué capa está cada uno de ellos.

### 7.7.4 Reorganizar las Capas

Como ya se ha comentado, las distintas capas tienen muchas cosas en común unas con otras. Lo primero y principal es la Línea de tiempo, todas las capas de una misma escena comparten la misma línea de tiempos y por tanto, los objetos de todos los fotogramas 1 de todas las capas se verán al mismo tiempo en la película superpuestos unos sobre otros. ¿y qué objeto está delante de los demás? Pues este criterio viene dado por la colocación de las Capas en la película. Los objetos que se mostrarán delante de todos los demás serán aquellos que se encuentren en la capa situada más arriba.

Para mover un fotograma de una capa a otra, basta con seleccionar el fotograma a mover y arrastrarlo hasta la capa donde queramos pegarlo. También se puede Copiar el fotograma y luego pegarlo en la capa de destino.

### 7.7.5 Tipos de Capas

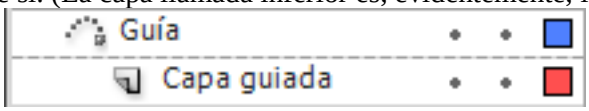
Como habréis podido comprobar al ver las propiedades generales de una capa o al hacer clic con el botón derecho del ratón sobre el icono de una capa cualquiera, existen varios tipos de capas.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Capas normales** : Son las capas por defecto de Flash y tienen todas las propiedades descritas en los puntos anteriores. Son las más usadas y se emplean para todo, colocar objetos, sonidos, acciones, ayudas...
- **Capas animación movimiento** : Son las capas que contienen una animación por interpolación de movimiento.
- **Capas Guía** / : Son capas especiales de contenido específico. Se emplean en las animaciones de movimiento de objetos y su único fin es marcar la trayectoria que debe seguir dicho objeto. Debido a que su misión es representar la trayectoria de un objeto animado, su contenido suele ser una línea (recta, curva o con cualquier forma).
- **Capas guiadas** (Guided Layers) : Definir una capa como guía no tiene sentido hasta que definamos también una capa guiada. Esto es, una capa que quedará afectada por la guía definida en la Capa guía.

  Si no definimos una capa guiada, la capa guía no tendrá ningún efecto y si bien no se verá en la película (por ser una capa guía) tampoco provocará ningún efecto en las demás capas.
  Las capas guía y las capas guiadas se relacionan entre sí de un modo evidente. A cada capa guía le corresponden una serie de capas guiadas.
  Al asociar una capa guía con una capa guiada, el icono que representa a la capa guía cambia, indicándonos que está realizando correctamente su labor.
  En la imagen podemos ver un ejemplo de capa guía y capa guiada correctamente asociadas entre sí. (La capa llamada inferior es, evidentemente, la capa con Guía)

  El funcionamiento de las Capas Guía y sus utilidades lo veremos a fondo en el tema de las Animaciones
- **Capas Máscara** : Estas capas se pueden ver como plantillas que tapan a las capas enmascaradas (las veremos enseguida). El funcionamiento de estas capas es algo complejo (tampoco demasiado) y se analizará en temas posteriores.

  Basta con decir que estas capas se colocan "encima" de las capas a las que enmascaran y sólo dejan que se vea la parte de éstas que tapan los objetos situados en las capas máscara (son como filtros). Al igual que las capas guía, los objetos existentes en este tipo de capas tampoco se ven en la película final. Sí se verán los objetos de su correspondiente capa enmascarada a los que estén "tapando".
- **Capas Enmascaradas** : Estas capas funcionan conjuntamente con las anteriores, ya que son las capas que están bajo las máscaras y a las que afectan. Al igual que las capas guía y las capas guiadas deben ir asociadas unas a otras para que su efecto sea correcto.

  Sus objetos sí que son visibles en la película final, pero sólo cuando algún objeto de la capa Máscara está sobre ellos.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.8 Símbolos

### 7.8.1 Qué son los símbolos

Los Símbolos provienen de objetos que hemos creado utilizando las herramientas que nos proporciona Flash.
Estos objetos al ser transformados en símbolos, son incluidos en una biblioteca en el momento en que son creados, lo que permite que sean utilizados en varias ocasiones, ya sea en la misma o en otra película. Los símbolos nos resultarán fundamentales a la hora de crear nuestras animaciones.

### 7.8.2 Cómo crear un símbolo

La acción de crear un nuevo símbolo es una de las más usadas en Flash ya que es uno de los primeros pasos para crear una animación, como veremos más adelante.
Seleccionamos el o los objetos que queramos convertir en un símbolo. Lo más habitual es partir de una forma.
Abrimos a la ventana **Convertir en símbolo**, accediendo al menú **Insertar → Nuevo Símbolo**, desde le menú contextual eligiendo **Convertir en símbolo**, o directamente con las teclas Ctrl + F8 o F8.
Una vez hecho esto nos aparecerá una ventana como la mostrada en la imagen. Introducimos el nombre del símbolo que vamos a crear, y que nos permitirá identificarlo en la biblioteca.
Sólo nos queda seleccionar el tipo de símbolo (desplegable **Tipo**) al que queremos convertir nuestro objeto. Podemos elegir entre Clip de **Película, Botón** y **Gráfico**. Sus características y las diferencias entre ellos las veremos en temas posteriores. Lo más habitual es Clips de película para los objetos que queremos mostrar en el escenario, y Botón si queremos que actúe como tal.

### 7.8.3 Las Bibliotecas

En Flash podemos encontrar dos tipos de bibliotecas, las bibliotecas comunes y de ejemplos y aquellas asociadas a las películas que hemos creado. Todas ellas las tenemos a nuestra disposición para utilizar los símbolos que contienen.
Para acceder a las bibliotecas comunes que nos ofrece Flash simplemente tenemos que ir a al menú **Ventana → Bibliotecas Comunes** y seleccionar alguna de las que se nos ofrecen. Las hay de todo tipo de símbolos: botones, clips o gráficos.
Para acceder a la biblioteca de símbolos de la película que estamos creando, de nuevo vamos a la Barra de **Menús, Ventana → Biblioteca**. En esta biblioteca aparecerán todos los símbolos que hemos creado hasta el momento.
Podemos comprobar como el nuevo símbolo que hemos creado en el ejercicio paso a paso de la unidad anterior se ha añadido a nuestra biblioteca accediendo a ella como acabamos de indicar.
Los símbolos contenidos en las bibliotecas están identificados por su nombre y por un icono que representa el tipo de símbolo que representan:

 Clip  Botón  Gráfico

Para utilizar un símbolo de una biblioteca basta con pulsar en el nombre de dicho símbolo y arrastrarlo a cualquier lugar del área de trabajo.

### 7.8.4 Diferencia entre símbolo e instancia

Cuando creamos un símbolo, Flash lo almacena en una biblioteca. Pues bien, cada vez que utilicemos ese objeto en una película, éste se convierte en una instancia del símbolo. Por tanto, podemos crear muchas instancias de un símbolo, pero a una instancia solo le corresponderá un símbolo.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

Aunque parece que sean lo mismo, la importancia de esta distinción es que cuando utilicemos un símbolo que hayamos creado previamente en una película y lo modifiquemos, solo alteraremos ésa instancia, mientras que el objeto seguirá intacto, tal y como era en el momento de su creación, de manera que podremos volverlo a utilizar en otro momento. En cambio, si modificamos el símbolo de la biblioteca, alteraremos todas sus instancias.

### 7.8.5 Modificar una instancia

No podemos modificar las instancias con las herramientas de dibujo de Flash, pero sí mediante el **Panel de Propiedades**, que permite la manipulación "externa" de la instancia.

Así, este panel, que como hemos visto resulta sumamente útil, no nos permite modificar la estructura básica de la instancia, pero sí otras propiedades, esto es, podremos hacer que la instancia tenga más brillo, pero no transformar una estrella en un círculo).
Esos cambios debemos de hacerlos directamente sobre el símbolo. Aunque sí podremos crear un símbolo a partir de una instancia, lo que desvinculará la instancia del símbolo original.

### 7.8.6 Panel propiedades de instancia

Para acceder al panel de propiedades de instancia, debemos seleccionar en primer lugar la instancia que queramos modificar y posteriormente abrir el panel **Propiedades**.
Si seleccionamos un objeto Flash que no se trate de un símbolo, el **Panel Propiedades** mostrará las propiedades del objeto en cuestión, pero no las características propias de los símbolos (cambios de color, intercambios etc...)
En el momento que seleccionemos un símbolo aparecerán una serie de propiedades y opciones que comentamos a continuación:

- **Nombre y su icono correspondiente.** El nombre de la instancia es muy importante, pues permite identificarla durante la película, y veremos más adelante que resulta imprescindible para acceder mediante ActionScript. El nombre, se introduce en el recuadro donde pone por defecto <Nombre de instancia>, y debe de ser único. El icono asociado nos permite saber qué tipo de símbolo es de un vistazo rápido.
- **Tipo de la instancia.** Por defecto se nos muestra el tipo al que pertenecía el símbolo original pero nosotros podemos cambiarlo para que cambie su comportamiento, aunque pueda seguir manteniendo su estructura inicial (en la imagen es Botón).
- **Símbolo de la Instancia seleccionada** (Instancia de:). Esta opción nos muestra el símbolo raíz del que proviene la instancia que estamos modificando.
- **Intercambiar.** Esta opción merece especial atención, pues es muy útil y nos ahorrará mucho trabajo cuando sea necesaria emplearla. Su función consiste en cambiar el símbolo de la instancia por cualquier otro que tengamos en nuestra Biblioteca, por lo que la instancia tomará el aspecto del nuevo símbolo. Puede parecer simple, pero durante el desarrollo de un trabajo profesional rápidamente surge la necesidad de probar situaciones y los diseños gráficos definitivos no suelen estar disponibles hasta bien avanzado el proyecto.

  Gracias a esta opción podemos trabajar tranquilamente con un "boceto" y sustituirlo de un modo efectivo (el nuevo símbolo hereda las propiedades del antiguo símbolo, incluido el nombre de instancia, las acciones que le afectarán, efectos gráficos etc...) cuando llegue


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

el momento.
En la imagen se puede observar el panel Intercambiar **Símbolo**.

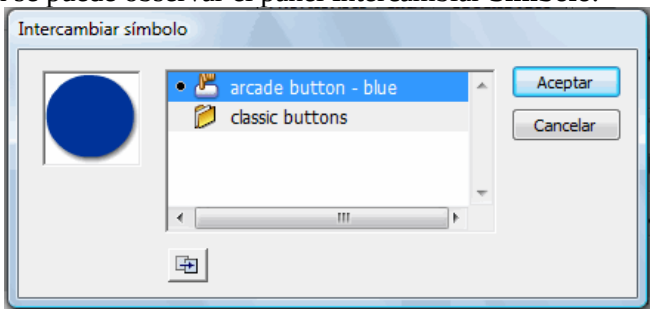



**Figura 128 - Panel Intercambiar Símbolo**

Este panel además, incorpora el botón **Duplicar Símbolo** cuya funcionalidad es la que nos podemos imaginar. Es muy útil cuando queremos hacer pruebas con un símbolo y no queremos perderlo.

### 7.8.7 Efectos sobre Instancias

Para acceder a los efectos aplicables sobre una instancia determinada, debemos acudir nuevamente al **Panel Propiedades**, desde aquí podremos acceder a todos los efectos que Flash nos proporciona.
Hay varios tipos de efectos. Si el símbolo se acaba de crear o si no tiene efecto asignado aparecerá en la pestaña **Ninguno**.
En el siguiente apartado comentamos los distintos efectos aplicables a una instancia.

- **Brillo.** Se puede modificar su valor desde -100% al 100%, esto es, completamente oscuro (negro) y completamente brillante (blanco). Puedes mover la barra deslizante o introducir su valor directamente en la casilla.
- **Tinta**. Esta opción permite cambiar el color de la instancia, pero puesto que, como dijimos, no podemos modificar la instancia internamente, al variar el color en la pestaña **Tinta** o bien mediante los valores RGB (cantidad de rojo, verde y azul), se cambiará el color de toda la instancia como si la estuviéramos tiñendo o poniendo una capa imaginaria de un color determinado. El grosor o intensidad de esta "capa" la podemos modificar en porcentaje mediante la primera pestaña que aparece a la derecha.
- **Alfa**. Representa el grado de visibilidad o transparencia que se tendrá de la instancia en cuestión. También se puede modificar mediante valor directo o con la barra deslizante y es muy útil para animaciones de aparición y desaparición de objetos. Si aplicamos un efecto alpha sobre una instancia que está encima de otro objeto, el objeto que antes estaba tapado se podrá ver a través de la instancia.
- **Avanzado**. Aquí podemos aplicar todos los efectos anteriores al mismo tiempo de manera más precisa, con la ventaja de que podemos ponerle un poco de cada uno, dando lugar a efectos de gran vistosidad.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.9 Gráficos

### 7.9.1 ¿Qué es un gráfico?

Los Gráficos son símbolos que nos permiten representar objetos estáticos y animaciones sencillas. En caso de que utilicemos un símbolo gráfico para realizar una animación, debemos tener en cuenta que ésta estará ligada a la línea de tiempo de la película en la que se encuentre. Es decir, la animación se reproducirá siempre y cuando la película original también se esté reproduciendo. Esto hace que, pese a tener su propia línea de tiempo, no puedan contener sonidos, controles ni otros símbolos gráficos.
Así pues, normalmente utilizaremos los gráficos para imágenes estáticas o para cuando nos convenga que una animación se reproduzca sólo cuando determinado frame de la línea de tiempo de la película esté en marcha, ya que para los casos que hemos comentado anteriormente en los que un gráfico no nos es útil, Flash nos ofrece otro tipo de símbolos como veremos en temas posteriores.

### 7.9.2 Tipos de gráficos

Los gráficos pueden ser:

**a**) **Estáticos**: estos gráficos se mantienen sin cambios cuando pasa el tiempo. Estos gráficos son los típicos en los fondos y en los objetos que no desempeñan ninguna función especial. Su tamaño y por tanto, el tiempo de carga de este tipo de gráficos, aunque siempre dependerá de la resolución, de sus dimensiones y de la forma en la que estén creados *, será en general reducido.

**b**) **Animaciones**: este tipo de gráfico varía su forma, posición u otras propiedades a medida que va pasando el tiempo. Puesto que para realizar la animación se deben usar varios gráficos más además del original o bien realizar determinadas acciones que modifiquen el estado inicial, el tamaño de esta clase de gráficos, para las mismas dimensiones y forma de creación, será mucho mayor que uno estático.

Las animaciones tienen dos inconvenientes:
1) Si se trata de un Mapa de Bits la web puede llegar a tener un tamaño excesivamente grande.
2) Aunque no se traten de mapas de bits, por ejemplo, si son animaciones típicas de Flash, cuyo tamaño no es excesivo, el hecho de poner muchas animaciones puede llegar a "marear" un poco al visitante de nuestro sitio y desviar su atención de lo que realmente importa, su contenido.

### 7.9.3 Introducir un Mapa de Bits

Flash permite importar mapas de bits de otros programas, cuando han sido guardados en formatos gráfico GIF, JPG, TIFF y muchos más. También nos permite modificarlos en cierto modo. Podemos cambiarle el tamaño y convertirlo en un símbolo para aprovechar las opciones que nos ofrece Flash aunque, teniendo en cuenta qué es un bitmap, no podremos modificarlo "internamente" pero podremos usarlo como un símbolo más.
Para importar un archivo de Mapa de Bits al escenario haz clic en el menú **Archivo → Importar → Importar a escenario.**
Se abrirá el cuadro de diálogo de **Importar**, allí deberás seleccionar el formato de imagen que quieres importar seleccionándolo en el desplegable **Tipo**. Luego navega por las carpetas hasta encontrarlo. Selecciónalo haciendo clic sobre él y pulsa el botón **Abrir**.
La imagen se incluirá en el escenario y estará lista para trabajar con ella.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 7.9.4 Introducir un archivo vectorial

Al igual que los mapas de bits, hay otros programas que trabajan con gráficos vectoriales como también hace Flash.
Si queremos traer un archivo vectorial creado en otro programa, por ejemplo Freehand o Illustrator, podemos hacerlo de manera muy sencilla.
Simplemente accedemos al menú **Archivo → Importar → Importar a escenario**. A continuación, seleccionamos el tipo de archivo correspondiente al gráfico vectorial que queramos importar. Pulsamos **Abrir**, y ya tenemos nuestro archivo vectorial.
Este archivo sí lo podremos modificar internamente ya que Flash es capaz de hacer gráficos de este tipo. Concretamente, Illustrator pertenece también a Adobe, igual que Flash, con lo que la compatibilidad en este caso es total.

### 7.9.5 Exportar un objeto Flash como mapa de bits

Para realizarlo seleccionamos el objeto que vamos a exportar y accedemos al menú **Archivo → Exportar → Exportar Imagen...** Luego introducimos en el campo Nombre el nombre que queremos que tenga nuestro nuevo bitmap. Seleccionamos el tipo de mapa de bits en que deseemos convertir nuestro objeto y pulsamos **Guardar**.

### 7.9.6 Exportar un objeto Flash como animación

Pues bien, con Flash también podemos crear una animación y exportarla como un GIF animado. Sin embargo, el GIF animado consiste en una secuencia de imágenes mostradas secuencialmente y es por esto que para exportar un objeto Flash como GIF animado es necesario que todos los fotogramas de esta animación sean clave, ya que el GIF no lo reconocerá en caso contrario y no veremos el efecto deseado.
Para exportar un símbolo y guardarlo como una imagen deberemos antes que nada seleccionarlo con la herramienta **Selección**.
Una vez seleccionado deberemos hacer clic en el menú **Archivo → Exportar → Exportar imagen...** y se abrirá un cuadro de diálogo.
En este cuadro de diálogo deberemos introducir el nombre del archivo que crearemos y seleccionar en el desplegable **Tipo** el formato de imagen con el que queremos guardarlo.
Una vez rellenados todos los campos y elegida la carpeta donde se guardará el archivo pulsa el botón **Guardar** y el archivo de imagen se creará y estará listo si quieres incluirlo en una página web estática o modificarlo con cualquier programa de imagen.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.10 Clips de Película

### 7.10.1 ¿Qué es un clip de película?

Un Clip de Película, simplemente Clip o *MovieClip* es una película en sí misma, como cualquiera de las que podamos haber creado hasta el momento en este curso. Normalmente nos referimos a ellas como clips cuando las incluímos en otra película, formando un símbolo. Por tanto, cualquier clip siempre podrá estar compuesto por otros clips insertados en él, que a su vez estén formados por otros, etc.

Al igual que los otros tipos de símbolos de Flash, los clips de película tienen su propia línea de tiempo. Sin embargo, y a diferencia de los Gráficos, esta línea temporal no está ligada a la línea de tiempo del documento que lo contiene, de tal forma que su ejecución es independiente, y en un fotograma de la película principal se puede estar reproduciendo repetidamente un clip.

Este tipo de símbolos puede contener cualquier otro tipo de símbolo: gráfico, clip o botón, así como cualquier objeto creado con Flash, ya que un clip es realmente una película.

Otra de las ventajas de los Clips la encontramos cuando realizamos películas de gran complejidad y tamaño, en la que intervienen un número muy elevado de fotogramas, debido a que en la vista general del documento, nosotros sólo veremos un fotograma por clip, el cual puede estar compuesto por muchos frames, lo que nos permitirá tener una mejor visión de cómo se desarrolla nuestra animación, y una línea de tiempo más clara y "limpia"

Los Clips son una de las herramientas que dan mayor potencia a Flash, permitiéndonos crear películas de gran complejidad y multiplicar los efectos visuales, ya que se pueden crear múltiples movimientos independientes entre sí y crear conexiones entre los diferentes Clips de un documento.

Todas aquellas cosas que no podíamos hacer con un símbolo de tipo Gráfico, lo podemos hacer con un Clip, además de poder realizar también todo aquello que nos permitía dicho símbolo. Por esto, normalmente se utilizan los clips para cualquier tipo de animación debido a su gran flexibilidad, dejando los gráficos sólo para imágenes estáticas.

### 7.10.2 Comprobar las propiedades de un Clip

La línea de tiempo de un clip no está ligada a la línea de tiempo del documento que lo contiene y, a la vez, veremos la diferencia entre los clips y los símbolos Gráficos, que supone una de las razones de la mayor utilización de los primeros en lo que se refiere a la creación de animaciones.

Para ello abriremos una nueva película Flash (**Archivo** → **Nuevo**, Archivo de Flash (AS 3.0)). Después importaremos una imagen cualquiera o bien crearemos una. La convertimos en Símbolo (botón derecho, Convertir en símbolo...) y seleccionamos en Tipo "Clip de Película". Arrastramos al escenario nuestro clip de película.

Editemos el Clip que insertamos en el documento, haciendo doble clic sobre él y examinemos su línea de tiempos. Aparecerá un único MovieClip. Podemos realizar algo similar a lo realizado con los gráficos. SI creásemos una animación de movimiento, como veremos más adelante, nos quedaría así:

La duración del clip que hemos insertado es mucho mayor que la película nueva que lo contiene. Si el símbolo fuera un gráfico ya hemos visto que al reproducir la película no ocurriría nada, porque sólo se reproduciría el primer fotograma de su línea de tiempo. En cambio, al tratarse de un clip, comienza a reproducirse al pasar por el primer fotograma, y como la línea de tiempo es independiente, sigue reproduciéndose aunque la línea de tiempo principal haya acabado.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

### 7.10.3 Crear un nuevo Clip

Se utiliza normalmente Clips para hacer animaciones. Se puede crear un símbolo Flash de la nada, igual que creamos un nuevo archivo, de forma que se quede en la biblioteca y podamos editarlo cuando nos convenga. Esto puede ser interesante en los clips, ya que a diferencia de los gráficos, su finalidad suele ser el movimiento y, en animaciones complejas, en ocasiones se les asignan acciones especiales en las cuales puede que no sea necesario crearlo en ese momento o convenga dejar el clip vacío.

Para insertar un clip vacío haz clic en **Insertar → Nuevo símbolo** y se abrirá el cuadro de diálogo de **Crear un nuevo símbolo**. Allí deberás darle un Nombre para identificarlo más tarde en la Biblioteca y seleccionar la opción Clip de Película en el desplegable Tipo.

A partir de este momento tendremos un nuevo clip (vacío) al cual podremos acceder desde nuestra Biblioteca (**menú Ventana → Biblioteca**), si hacemos clic derecho sobre él y seleccionamos Edición, podremos editarlo y trabajar con él.

### 7.10.4 Importar y Exportar Movie Clips de Biblioteca

Como para todos los símbolos los Clips se almacenan en la biblioteca del documento cuando son creados. Esto es muy importante en muchos casos ya que habitualmente los clips son muy reutilizables. Para importar clips debemos abrir primero la biblioteca en la que está contenido.

Hemos visto en el tema de Símbolos, dos tipos de bibliotecas: las que están asociadas a documentos u otras películas y las que nos proporciona Flash. No sólo podemos utilizar símbolos del mismo documento en el que estamos sino que podemos Importarlos de otros documentos de nuestro disco duro, lo que, en el caso que nos ocupa, puede resultar de gran utilidad. Obviamente la exportación mediante biblioteca se hace automáticamente ya que Flash deja los objetos creados en la biblioteca para que puedan ser reutilizados.

Para importar un Clip de un archivo del disco duro debemos ir al menú **Archivo → Importar → Abrir biblioteca externa...**, seleccionar el Archivo Flash (.fla) del que queremos importar sus símbolos de biblioteca y pulsar Abrir.

Aparecerá la biblioteca con la lista de los símbolos correspondientes a los gráficos, botones y clips del documento en cuestión.

Es importante destacar que cuando insertemos un clip de una biblioteca, se insertarán a su vez todos los símbolos que contenga, incluidos los clips.

## 7.11 Botones

### 7.11.1 ¿Qué es un Botón?

Los símbolos de tipo Botón son los que aportan la mayor parte de la interactividad de las películas Flash con aquel que la está visualizando. Un botón, en Flash, es igual que cualquier botón de cualquier entorno informático, sea web o cualquier otro.

Son elementos que se prestan a que el usuario los presione, desencadenando al hacerlo una serie de acciones. También es habitual ver cómo este tipo de elementos reaccionan cuando se les pasa el ratón por encima o cuando están pulsados.

Al igual que los otros símbolos de Flash, los botones tienen su propia línea de tiempo. Esta es independiente pero, sin embargo, está formada únicamente por cuatro fotogramas, uno para cada estado posible del botón.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

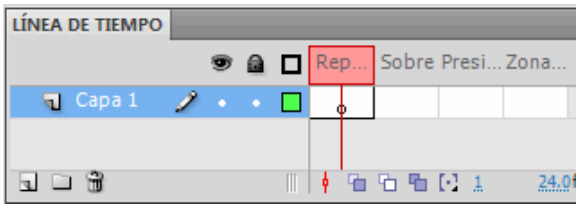



**Fig 1. Línea de tiempo Botón**

- **Reposo**. Aspecto por defecto del botón, es decir, cuando el puntero del ratón no está situado sobre él.
- **Sobre**. Aspecto del botón cuando situamos el puntero sobre él.
- **Presionado**. Apariencia que deseamos tenga nuestro botón mientras lo mantengamos pulsado.
- **Zona activa**. Aquí debemos indicar el área real en la que queremos que actúe nuestro botón. Esto es importante sobre todo en botones compuestos sólo por texto como veremos más adelante.

Los botones pueden contener a su vez otros símbolos, como clips o gráficos (también de tipo Bitmap).

### 7.11.2 Creación de un Botón

En la creación de un botón podemos considerar dos fases. En la primera vamos a convertir nuestro objeto a símbolo de tipo botón y posteriormente veremos cómo completarlo internamente, lo que nos ayudará a entender mejor dicha estructura.

Comenzamos creando el objeto que representará el aspecto por defecto de nuestro botón con las herramientas que nos ofrece Flash.

Seleccionaremos el objeto y accederemos al menú **Insertar** → **Convertir en Símbolo**, le daremos el Tipo Botón y asignaremos un nombre a nuestro nuevo símbolo.

De esta forma ya tenemos transformado el objeto para que se comporte como un botón. Ahora lo completaremos internamente.

Para determinar cómo debe reaccionar el botón en función de las acciones del ratón, lo editaremos haciendo clic con el botón derecho del ratón sobre nuestro nuevo botón y seleccionando la opción **Editar**.

Cuando tengamos delante la línea de tiempo del botón (observa que tiene el aspecto que hemos mostrado anteriormente), seleccionaremos cada uno de los frames (sobre, reposo, presionado y zona activa) y pulsaremos F6 para crear un fotograma clave en cada uno de ellos.

Ahora ya podemos modificar el aspecto inicial del botón para cada posición del cursor y marcar el área de acción del botón (fotograma Hit) en la que simplemente podremos dejar la misma figura que la inicial (en este caso sólo es importante la forma del objeto, no los colores u otras cosas) o bien dibujar con las herramientas de dibujos de Flash una nueva figura, en cuya superficie "se sentirá aludido" nuestro botón.

Si una vez creado el botón queremos observar sus distintos estados y todavía no hemos terminado la película entera y por tanto no deseamos tener que reproducirla toda podemos hacerlo accediendo a la Biblioteca de nuestra película y seleccionando el botón creado. Para ver


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

lo que comentábamos bastará con pulsar el icono  situado a la derecha de la vista previa del símbolo.

### 7.11.3 Formas en los Botones

Los botones son símbolos que pueden tener multitud de formas. Si bien lo más habitual es ver botones rectangulares, cuadrados y circulares, cuya creación es inmediata como vimos en el punto anterior, también hay otros muchos tipos de botones que, pese a ser menos utilizados, es muy habitual verlos en multitud de páginas web.

Entre estos están los creados mediante formas poligonales, aquellos que están formados por texto únicamente, dibujos con diferentes motivos, etc. Es interesante su uso para dar más vistosidad ya que algunos resultan más expresivos, y en esto Flash nos ayuda mucho, debido a la relativa sencillez de creación de botones que sus herramientas de dibujo nos ofrece.

Hay varias formas de botón también muy extendidas, como el botón con relieve sencillo o los botones en forma de píldora. Puesto que existen muchas formas de conseguir estos efectos, a continuación tienes un ejercicio paso a paso que muestra una forma de conseguir el relieve en un botón rectangular.

### 7.11.4 Incluir un clip en un botón

La inclusión de clips de película en los botones puede dotar a éstos de más vistosidad. Es habitual colocar un clip en el fotograma **Sobre** para indicar algún tipo de información extra o una animación para ir más allá de un cambio de color.

También es común ver un clip de película actuando como un botón. Esto caso se puede hacer por ejemplo poniendo el clip en el fotograma **Reposo**.

Incluir un Clip en un botón es muy sencillo. Tomando este ejemplo, y partiendo de que tenemos los dos textos como clips en la biblioteca, sólo tendremos que:

- Hacer doble clic sobre el botón para entrar en su modo de edición.

- Seleccionamos el estado Sobre para modificarlo. Selecciona la instancia del texto.

Ahora tenemos dos opciones:

- Pulsa la tecla SUPRIMIR para eliminarlo.

- Desde la Biblioteca (menú **Ventana** → **Biblioteca**) arrastramos el clip con el nuevo texto, para crear una instancia.

O la opción más práctica:

- En las propiedades de la instancia, pulsamos Intercambiar... y elegimos el nuevo objeto de texto.

- Por último, centramos el nuevo texto en el botón, si es necesario.

Por supuesto, en vez de cambiar el texto podemos añadir otros clips, que simulen movimientos, reflejos, etc.

### 7.11.5 Bitmaps y botones

Además de clips, los botones también pueden contener símbolos de tipo Gráfico.

Básicamente podemos hacer dos cosas:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

1) Incluir en cada uno de los fotogramas del botón un bitmap distinto, obteniendo un efecto como el que se consigue con lenguajes como javascript (siempre considerando la mayor sencillez de Flash).

2) Aprovechar las propiedades de los Gráficos en Flash. Para esto, deberíamos importar primero el Bitmap y después convertirlo a símbolo botón. Posteriormente lo editaríamos y, después de insertar cada fotograma clave, convertiríamos su contenido a símbolo Gráfico. Una vez hecho esto, variando los efectos de las instancias en Flash (Alfa, Tinta, Brillo) podremos conseguir efectos bastante buenos.

### 7.11.6 Acciones en los botones

Comenzamos por crear o añadir nuestro botón, y asignarle un nombre de instancia. El nombre que le demos es muy importante, porque nos permitirá acceder a él desde el código. Aunque podemos escribir el código en la misma capa, recomendamos crear una capa exclusivamente para el código.

Ahora, abrimos en el panel Acciones (menú **Ventana** → **Acciones**). Se mostrará un área en blanco en la que podemos escribir.

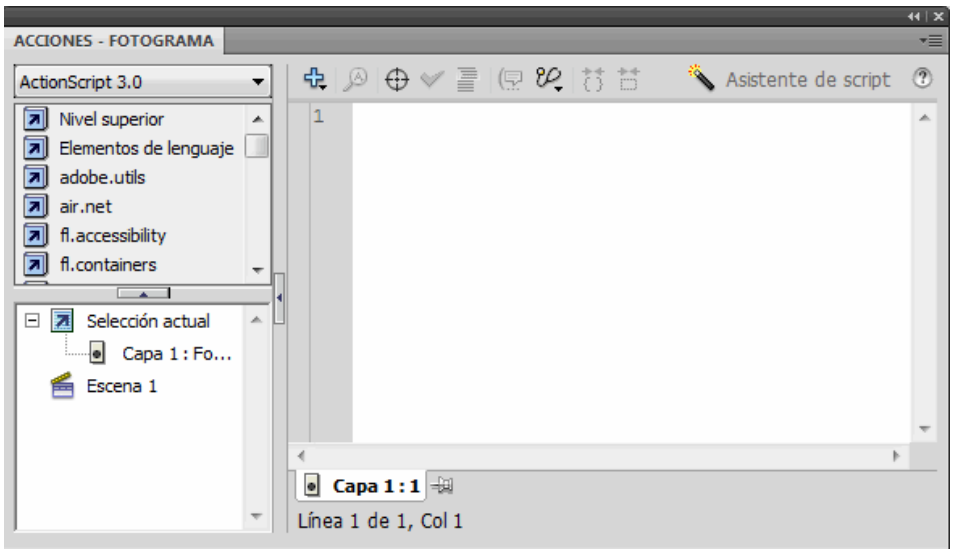



**Figura 129 - Ventana ActionScript**

Este es el código que debemos de escribir para asociar acciones a un botón:

```
miBoton.addEventListener('click', accionesMiBoton); function acciones-
MiBoton(event):void{   //Acciones }
```

Donde *miBoton* será el nombre de la instancia del botón.

*accionesMiBoton* contiene las acciones a realizar. Observa que aparece en dos sitios. El nombre que le hemos dado es el que queramos.

Si tenemos varios botones, a cada uno lo referiremos por su nombre de instancia que es único. También tenemos que dar un nombre único a *accionesMiBoton* para cada uno, si no todos realizarían las mismas acciones.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Ahora, cambiamos donde pone //Acciones por lo que queramos que haga. Veamos las más comunes:

1) **Abrir una página web**. Con esto conseguiremos abrir una página cualquiera de internet (o una película Flash), lo que nos servirá para irnos desplazando por webs que contengan más de una página, o permitir al usuario descargarse archivos entre otras cosas.

La instrucción en ActionScript que nos permite hacerlo es `navigateToURL(new URLRequest("http://www.mipagina.es"), "_blank");`. "http://www.mipagina.es" se refiere a la página que queremos abrir, y "_blank" indica que se abrirá en una página nueva.

2) **Controlar una película en curso**. Si estamos reproduciendo una película Flash y queremos permitir que el usuario la detenga, la ponga en marcha, avance, retroceda ...

Para ello podemos emplear las acciones:

`stop();` para detener.

`play();` para reproducir.

`gotoAndPlay(numeroFotograma);` para ir a un fotograma determinado.

### 7.11.7 Incluir sonido en un botón

Para ello, simplemente debemos editar nuestro botón y seleccionar el fotograma **Presionado**, e insertar el sonido. Como vimos en el tema correspondiente. Es este caso, el tipo de sincronización más conveniente suele ser **Evento**.

## 7.12 Animaciones de movimiento

### 7.12.1 Interpolación de movimiento

Es la acción básica de las animaciones en Flash. Permite desplazar un símbolo Flash de un lugar a otro del escenario, siendo necesarios únicamente dos fotogramas, lo que optimiza mucho el rendimiento de la película.

Es importante destacar que para que una **Interpolación de movimiento** se ejecute correctamente aquellos objetos que intervengan deberán haber sido previamente convertidos a símbolos. Los gráficos, clips de película, textos y botones son algunos de los símbolos que se pueden interpolar.

También se debe tener cuidado al realizar una interpolación con dos símbolos que se encuentren en la misma capa, ya que el motor de animación los agrupará como uno sólo y el resultado no será el esperado. Por esto es conveniente asegurarse de dos cosas:

- Separar en distintas capas los objetos fijos y los que estarán animados.
- Poner también en distintas capas objetos que vayan a ser animados con direcciones o formas distintas.

Una interpolación de movimiento, es el desplazamiento de un símbolo de uno a otro punto del escenario. El hecho de que sólo se necesiten dos fotogramas es debido a que Flash, únicamente con la posición inicial y final, "intuye" una trayectoria en línea recta y la representa (veremos que también se pueden realizar movimientos no rectilíneos).

Crearla es tan simple como hacer clic derecho sobre el fotograma que contiene los elementos y elegir **Crear interpolación de movimiento**. Por defecto, se añadirán unos cuantos fotogramas, rellenos de un color azulado.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

Ahora vamos al fotograma final, o creamos uno clave. Y desplazamos el símbolo. Veremos que aparece una línea punteada, por defecto recta, que representa el trazado de la animación.

Cuando realicemos la interpolación correctamente observaremos un aspecto como este en la línea de tiempo.

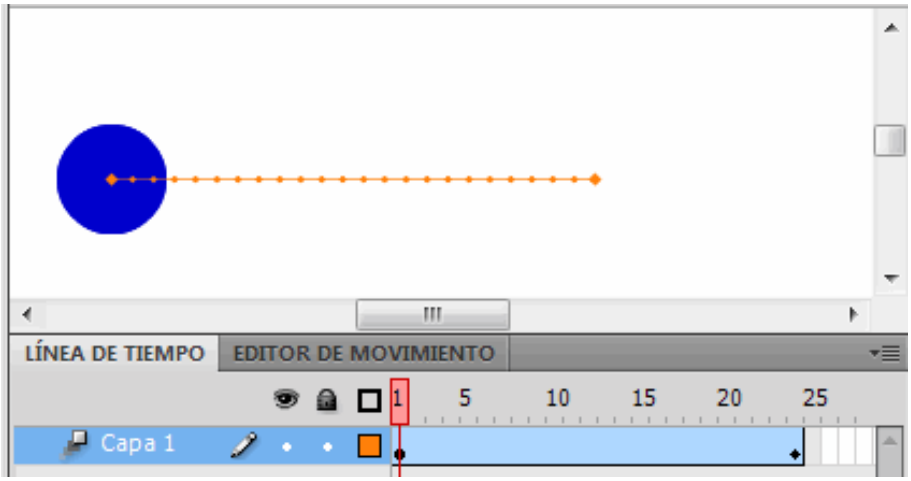
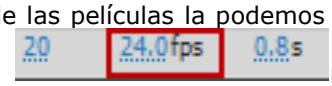



**Figura 130 - Interpolación del movimiento**

Esto indica que la animación cambiará la posición del símbolo del fotograma 1 hasta la posición del mismo símbolo en el fotograma 24, utilizando precisamente 24 fotogramas. El número de fotogramas que se usen en la interpolación indicará las subetapas de que constará la animación. Cuantas más subetapas más sensación de "continuidad" (menos saltos bruscos) pero a la vez menos velocidad en el movimiento.

La velocidad en el movimiento de las películas la podemos cambiar también modificando su parámetro en la línea de tiempo, 

pero esto no cambiará lo que hemos comentado anteriormente respecto al número de fotogramas.

La velocidad está expresada en Fotogramas Por Segundo (fps) y se puede modificar haciendo doble clic en el lugar que hemos indicado de la línea de tiempo. A mayor valor más velocidad, pero se deben poner siempre suficientes fotogramas para que se desarrolle la animación como queremos.

El trazado recto generado por defecto podemos modificarlo directamente haciendo clic y arrastrándolo, una vez seleccionada previamente la herramienta Selección .

En cualquier fotograma de la interpolación podemos cambiar la posición del símbolo (o cualquier otra propiedad), creando ahí un fotograma clave de propiedad, que se representa por un pequeño rombo en la línea de tiempo.

La interpolación de movimiento permite modificar muchos parámetros del movimiento mediante el **Editor de movimiento**, que veremos a continuación.

### 7.12.2 El editor de movimiento

Para acceder al **Editor de movimiento** basta tener seleccionada una interpolación de movimiento y hacer clic en la pestaña **Editor de movimiento** que aparece junto a la línea de tiempo. Como cualquier panel, también es accesible desde el menú **Ventanas**.

Este panel nos permite controlar multitud de propiedades y efectos que afectan a una animación con total precisión, fotograma a fotograma.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Podemos ver, a la izquierda una columna con las propiedades que podemos modificar, divididas en **Movimiento básico, Transformación, Efectos de Color, Filtros y Suavizados**.

Junto a estas propiedades, aparece una columna con los valores que toma esa propiedad en el momento seleccionado de la línea de tiempo. En la siguiente columna podemos establecer si el valor se aplica con aceleración o no.

En la columna **Fotogramas**, podemos recorrer o eliminar los distintos fotogramas clave. También los controles - y + que nos permiten añadir efectos.

Y a la derecha del todo encontramos la gráfica. Podemos ver que cada propiedad tiene una gráfica específica, que indica los fotogramas en horizontal y los valores de la propiedad en vertical. Si hacemos clic sobre una propiedad, veremos que su gráfica se expande para editarla con facilidad. En la gráfica encontramos los fotogramas clave marcados como un cuadrado negro, o verde cuando está seleccionado. Estirando de ellos, o de la línea de la gráfica podemos alterar los valores.

En la gráfica, vemos que los puntos suelen formar un vértice. Una opción muy interesante es poder transformarlos en puntos suavizados (desde el menú contextual del fotograma), creando una curva Bezier, lo que formará transiciones más suaves entre los picos de valor. Esto no es aplicable a las propiedades X,Y, Z.

### 7.12.3 Interpolación clásica

Una interpolación clásica, igual que una interpolación de movimiento, es el desplazamiento de un símbolo de uno a otro punto del escenario, muchos de los conceptos vistos en las interpolaciones de movimiento son los mismos para las interpolaciones clásicas. Por ejemplo, las animaciones también han de ser sobre símbolos y deben estar en una capa. *Los Fotogramas Por Segundo* (fps) tienen el mismo significado.

Para crear una interpolación clásica hay que hacer clic derecho sobre el fotograma que contiene los elementos y elegir *Crear interpolación clásica*.

Cuando realicemos la interpolación correctamente observaremos un aspecto como este en la línea de tiempo.

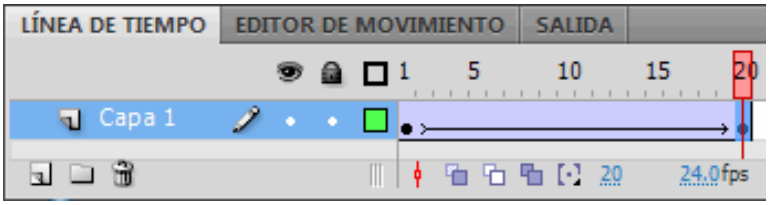



**Figura 131 - Interpolación clásica**

Vemos que la animación va desde el fotograma 1 hasta el fotograma 20. Aparece una flecha que no aparece en la interpolación de movimiento y el icono que hay a la derecha del nombre de la capa es distinto.

Al realizar una interpolación clásica el fotograma inicial y final deberán ser diferentes, en caso contrario no se creará ningún tipo de animación.

Si el objeto con el que queremos hacer la interpolación clásica no está convertido a símbolo nos encontraremos con algo así...

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

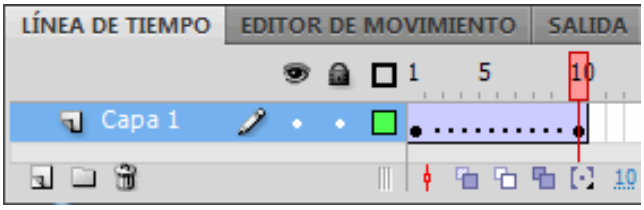



**Figura 132 - Interpolación clásica sin símbolo**

... y la animación no funcionará.

También podemos realizar la interpolación de otra forma, sin convertir previamente el objeto a símbolo, ya que Flash lo convierte a símbolo automáticamente si no lo hacemos, dándole el nombre "Animar" más un número. Esto quizá no sea lo más conveniente en películas grandes, debido a lo que ya comentamos de la gran cantidad de símbolos que pueden aparecer y la confusión que crean muchos símbolos con nombres parecidos.

Para crear una interpolación de este tipo, basta con tener un fotograma clave. Hacemos clic con el botón derecho sobre el fotograma en la línea de tiempo, y seleccionamos **Crear Interpolación Clásica**. Ahora, creamos un nuevo fotograma clave donde queremos que finalice la interpolación, y modificamos los símbolos en los fotogramas clave.

Si seleccionamos uno intermedio, se muestran los símbolos en su transición al fotograma final. Podemos decidir cómo mostrar el símbolo en ese fotograma, por ejemplo moviéndolo. Al hacerlo automáticamente se crea un fotograma clave. Esto hace que el movimiento ya no sea recto, y pueda ser en zig-zag. Si hacemos esto varias veces sobre varios fotogramas obtendremos varias trayectorias consecutivas más.

#### 7.12.3.1 Diferencias entre interpolación de movimiento e interpolación clásica

En general las interpolaciones de movimiento son más fáciles de utilizar y más potente, no obstante las interpolaciones clásicas tienen características que pueden hacerlas más interesentes para determinados usuarios.

Estas son algunas de las diferencias entre los dos tipos de interpolaciones:

- Las interpolaciones de movimiento incluyen el trazado del movimiento, mientras que en una animación clásica no existe el trazado, a menos que lo creemos expresamente.
- Sólo se permiten realizar interpolaciones con símbolos, si aplicamos una interpolación de movimiento a un objeto que no es un símbolo, Flash lo convertirá en un clip de película, mientras que si se trata de una interpolación clásica lo convertirá en un símbolo gráfico.
- En las interpolaciones clásicas cuando cambia una propiedad se crea un fotograma clave y cambia la instancia del objeto, mientras que en las interpolaciones de movimiento sólo hay una instancia de objeto y al cambiar una propiedad se crea un fotograma clave de propiedad.
- La interpolaciones de movimiento pueden trabajar con texto sin tener que convertirlo en símbolo, como ocurre en las clásicas.
- En un grupo de interpolación de movimiento no está permitido usar scripts de fotograma, mientras que sí es posible en las clásicas.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- Los grupos de interpolaciones de movimiento se pueden cambiar de tamaño en la línea de tiempo. Se tratan como un objeto único. Las interpolaciones clásicas están formadas por grupos de fotogramas que se pueden seleccionar de forma independiente.
- Las interpolaciones de movimiento sólo pueden aplicar un efecto de color por interpolación, mientras que las clásicas pueden aplicar más de uno.
- Los objetos 3D sólo pueden animarse en interpolaciones de movimiento, no en clásicas.
- Sólo las interpolaciones de movimiento se pueden guardar como configuraciones predefinidas de movimiento.

### 7.12.4 Cambio de forma en una Interpolación de movimiento

Acabamos de ver las interpolaciones de movimiento y las interpolaciones clásicas como un mecanismo para desplazar un símbolo Flash de un lado a otro del escenario. Sin embargo, podemos aprovechar una interpolación para realizar animaciones en las que nuestro objeto aumente o disminuya de manera progresiva su tamaño. Basta con modificar la instancia del símbolo en el último fotograma de la interpolación de movimiento o de la interpolación clásica, pero esta vez cambiándole el tamaño. Podemos utilizar la herramienta **Tranformación libre** para cambiar el tamaño.

Por supuesto, podemos aplicar ambos efectos a la vez, de manera que el cambio de tamaño se producirá mientras el objeto se desplaza. También podemos hacer el cambio de tamaño en varias fases o secuencias encadenadas como en las interpolaciones de movimiento comunes.

### 7.12.5 Interpolación guiada

Una guía es una capa especial que marca una trayectoria para los símbolos de la capa a la que afecta, para que dichos símbolos la sigan, durante el movimiento. Esta capa es invisible durante la reproducción y permite dibujar cualquier tipo de dibujo vectorial, que nos permitirá crear un movimiento no forzosamente rectilíneo.

Crear un movimiento mediante esta técnica es bastante sencillo. Partimos de una capa con una interpolación clásica:

- Sobre la capa en la línea de tiempo, hacemos clic derecho y seleccionamos **Añadir guía de movimiento clásica**.
- Creamos el trazado de la guía en la nueva capa, por ejemplo dibujando con el **Lápiz**.
- Y para acabar, hacemos coincidir el fotograma final de la interpolación con el final del trazado de la guía. Es importante que el símbolo esté en contacto con la guía. No es necesario colocarlos al principio del trazado ya que Flash lo hace automáticamente.

Una opción que puede resultar muy interesante es **Orientar según el trazado**, la cual encontramos en el panel de Propiedades. Esta opción hará que el símbolo vaya girando para seguir la línea del trazado, lo que en la animación anterior habría puesto el avión boca abajo al hacer el "loop".

Tanto si el trazado es de una interpolación de movimiento, como si es de una interpolación clásica, podemos modificarlo de varias formas:

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- Seleccionando la herramienta Selección y haciendo clic sobre el trazado y arrastrándolo.
- Seleccionando la herramienta Subselección y haciendo clic en el trazado, aparecerán los controladores de curva, moviéndolos modificaremos el trazado.
- Seleccionando la herramienta Transformación libre y haciendo clic en el trazado, aparecerán los controladores de tranformación libre, moviéndolos modificaremos el trazado.

Si se trata de un trazado de una interpolación de movimiento, adicionalmente disponemos de otros métodos para modificar el trazado:

- Moviendo la posición del objeto en el escenario, esto hace que se cree automáticamente un fotograma clave de propiedad y se modifique el trazado. Esta es, quizás, la forma más fácil de modificar un trazado.
- Utilizar el editor de movimiento para modificar los valores de X, Y, Z.

Si estamos modificando el trazado de una interpolación clásica, debemos tener cierto cuidado con no producir discontinuidades en el trazado, ya que esto haría que la animación se detuviera en ese punto. En general es más fácil y flexible utilizar los trazados con las interpolaciones de movimiento que con las interpolaciones clásicas.

## 7.13 Animación por Forma

### 7.13.1 Interpolación por forma

Cuando lo que queramos no sea cambiar la posición de un objeto en el escenario, sino su forma de manera progresiva (o ambas cosas a la vez), Flash nos ofrece la técnica de la **Interpolación por Forma**, que consiste simplemente en ir transformando el contorno de un objeto creado en su interfaz hasta que sea igual que el contorno de otro objeto distinto.

Realizar una interpolación por forma, es muy semejante a crear una interpolación de movimiento. Flash genera fotogramas intermedios en los que va variando ligeramente la forma del fotograma anterior. Sólo necesitamos dos fotogramas clave. Colocaremos en el primer fotograma el objeto con su aspecto original, y en el último la apariencia final que queremos que tenga.

Es importante destacar que para que una **Interpolación por Forma** funcione como es debido aquellos objetos que intervengan deberán ser objetos vectoriales (no símbolos Flash).

Debemos tener también dos aspectos en cuenta:

1) Separar en distintas capas los objetos fijos y los que estarán animados.

2) Poner en distintas capas objetos que vayan a ser transformados con formas distintas, ya que Flash transformará todos los objetos vectoriales del primer fotograma en aquello que haya en el último fotograma de la interpolación.

Para crear la interpolación, una vez tengamos los fotogramas de inicio y fin de la animación hacemos clic derecho en alguno de los fotogramas intermedios, y seleccionamos la opción del menú **Crear interpolación de forma**.

### 7.13.2 Transformar textos

Para realizar una interpolación de forma es necesario que el objeto sea de tipo vectorial. Este no es el caso de los textos, que se basan en fuentes y son creados en función de los valores


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

de la tabla ASCII y un tipo de letra asociado. Para solucionar este problema, deberemos transformar el texto previamente en un objeto vectorial, es decir, como si hubiéramos repasado con la herramienta lápiz el contorno de cada letra.

Debemos asegurarnos de dar suficientes fotogramas de margen para que se note la transición. También es conveniente dar un tiempo para que cada letra sea mostrada y pueda ser visualizada y "entendida" por aquel que vea la película, porque si no lo hacemos corremos el riesgo de que la animación parezca sólo un garabato revolviéndose.

Cuando creemos interpolaciones de forma y queramos incluir textos deberemos actuar de un modo especial dado que un objeto de texto no se considera como una forma. Para ello, y una vez tengamos la animación creada con una forma en el fotograma inicial y un texto en el final veremos que la interpolación aparece como incorrecta. Deberemos, entonces seleccionar el fotograma donde se encuentre el texto y seleccionarlo.

Haremos clic en el menú **Modificar** → **Separar** para convertir el texto en un conjunto de puntos que creen una forma. Y la interpolación ya estará lista.

### 7.13.3 Consejos de Forma

En el caso de la interpolación por forma, es posible que Flash realice la transformación de los objetos de una forma que no es la que esperábamos y que no nos conviene para llevar a cabo nuestro propósito.

Pues bien, para estos casos Flash nos ofrece una herramienta para solucionarlo: **los consejos de forma.**

Los consejos de forma son marcas que indican a Flash qué puntos de la imagen inicial son los que deben corresponderse con otros tantos puntos en la figura final. Al indicar esto, estamos controlando las formas intermedias que se generarán.

Para añadir un consejo de forma a una interpolación nos situaremos en el fotograma inicial y pulsaremos **Control + Shift + H** (o bien ir al menú **Modificar** → **Forma** → **Añadir Consejo de Forma**).

Aparecerá un círculo de color con una letra dentro en el centro de la figura. Su color será rojo mientras no sea colocado en el contorno del objeto (los extremos, si es que no hay contorno), que es donde tiene efecto. Cuando están colocados sobre una curva (vector) que forma un contorno son amarillos para el objeto inicial y verdes para el final.

Dicho círculo debemos colocarlo en un punto del contorno de la figura inicial. Automáticamente aparecerá un punto con la misma letra dentro en la figura final (este punto será el que se deba corresponder con el punto del objeto de origen) y deberemos colocarlo en el punto que deseemos del contorno.

El primer consejo de forma está marcado con la letra "a". Si creamos más de uno serán marcados con las letras "b", "c", "d" ... hasta la z, que es el límite (26 como máximo).

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.14 Efectos sobre animaciones

### 7.14.1 Efectos sobre la interpolación

Si seleccionamos un fotograma y un objeto perteneciente a una interpolación, y abrimos el panel **Propiedade**s, desde el botón de la parte superior derecha, o bien desde el menú **Ventana, Propiedades**, encontraremos las distintas propiedades aplicables a ese objeto en ese fotograma. Las propiedades variarán según el tipo de interpolación, según puedes ver en las siguientes imágenes correspondientes a las propiedades de la interpolación de movimiento, interpolación guiada e interpolación de forma:

En las interpolaciones de movimiento podemos darle un nombre de etiqueta, nos ayudará a identificarla en la barra de tiempo. Aparecerá a lo largo de todos los fotogramas que la componen.

La propiedad tamaño se representa por los valores W (anchura) y H (altura) y para modificarla basta colocar el cursor sobre el valor y aparecerá un deslizador que podemos mover a izquierda o derecha, también podemos hacer clic sobre el valor y teclear el nuevo valor.

Desde el **Editor de movimiento** también podemos modificar el tamaño de un objeto desde las propiedades Escala X (anchura) y Escala Y (altura) que tienen valores en tanto por ciento.

Por último, con el uso de la herramienta **Tranformación libre** también podemos variar las dimensiones del objeto.

En las interpolaciones guiadas, al activar la casilla de **Escala**, permite un incremento/decremento progresivo del tamaño del objeto inicial cuando sus dimensiones son distintas a las del objeto que está en la posición final de la animación. Si la casilla está desactivada podríamos encontrarnos con una animación como la siguiente.

En las interpolaciones de movimiento esta opción está activada defecto.

Como vemos Flash hace el desplazamiento de posición, pero el cambio de tamaño se produce bruscamente en el último fotograma.

La barra deslizante nos permite determinar la aceleración del primer tramo de la animación. Podemos elegir valores entre -100 y 100. Si el valor de aceleración es negativo el símbolo se moverá despacio primero e irá acelerando de manera progresiva. Si dicho valor es positivo provocará un efecto opuesto.

La opción de **Rotación** da la posibilidad de aplicar una rotación al objeto mientras se produce el movimiento. Podemos especificar el número de veces o vueltas que se produzca la rotación durante los fotogramas que dure la interpolación, o también podemos indicar los grados que queremos que rote el objeto.

Si se trata de una interpolación de movimiento, nos presenta un submenú **Dirección**, con tres opciones:

- **Ninguno.** Con esto le indicamos a Flash que no aplique rotación alguna sobre el símbolo en movimiento. Es la opción por defecto en las interpolaciones de movimiento.
- **Antihorario** (CCW). Permite realizar el número de rotaciones completas o el valor de grados en la dirección contraria a la de las agujas del reloj,
- Horario (CW). Realiza lo mismo que la opción anterior pero en la dirección de las agujas del reloj (hacia la derecha).


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Si se trata de una interpolación clásica, nos presenta un submenú Dirección, con una opción más:

- **Auto**. Marcando esta opción hacemos que se produzca la rotación en aquella dirección que necesite menos movimientos. Si cuesta lo mismo hacerlo por un lado o y por el otro, es decir, cuando la imagen inicial y final está en la misma posición (en cuanto a rotación se refiere), el hecho de activar esta opción no tendrá ningún efecto.

  Esta opción es la que está marcada por defecto en las interpolaciones clásicas y por eso no hemos visto hasta ahora un objeto rotar en nuestras interpolaciones clásicas. Para que hubiera tenido efecto deberíamos haber rotado la última imagen de la interpolación.

Desde el Editor de movimiento, también podemos especificar los grados de una rotación.

Si activamos la casilla **Orientar según trazado**, tanto si la interpolación sobre la que actuamos es una interpolación guiada, como si es de movimiento, Flash hará que el símbolo tome la dirección de la guía, rotando para orientarse en la misma posición que adopta la línea.

Con la opción **Sincronizar** evitamos que no se reproduzca el último bucle de un símbolo gráfico incluido en nuestra película con una animación en su línea de tiempo interna, cuando el número de fotogramas que ocupa en la línea principal no es múltiplo de los fotogramas que contiene la instancia.

Si activamos la opción **Ajustar**, el centro de la instancia (identificado con una cruz) se ajustará forzosamente a la guía que hemos marcado en la correspondiente capa.

### 7.14.2 Efectos sobre el símbolo interpolado

Podemos aplicar los efectos sobre cualquier símbolo de cualquier fotograma de la animación, tanto sobre el fotograma inicial o el final como en los intermedios. En este último caso, si se trata de una interpolación clásica, deberemos convertir a fotograma clave aquel que contenga la instancia y después aplicar el efecto. El efecto se irá aplicando de forma gradual. De hecho, es frecuente utilizar interpolaciones clásicas únicamente para aplicar efectos.

### 7.14.3 Efecto tinta

El efecto de tinta, tiene un amplio marco de posibilidades de uso. Es muy usado en textos y en botones, por ejemplo haciendo que cambien progresivamente de color al pasarles el ratón por encima o simplemente efectos de cambio de color en presentaciones.

El tintar colores supone un toque alegre y muy vistosos en las presentaciones y si se usan varios efectos, combinados adecuadamente, dota de un ritmo rápido a la animación, como una explosión de color que sorprenda al receptor de la película.

Otro interesante uso de este efecto es simular la proyección de una luz de color sobre un objeto que se va acercando al foco.

### 7.14.4 Efecto transparencia (Alfa)

Este es probablemente el efecto más utilizado debido a la versatilidad del hecho de controlar el grado de visibilidad de los objetos. Podemos, por ejemplo, simular un foco apuntando a un escenario o, lo más común, hacer aparecer objetos de la nada y también hacer que se desvanezcan poco a poco.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## 7.15 Generar y publicar películas

### 7.15.1 Consideraciones sobre el tamaño de las películas

**CONSIDERACIONES EN EL DIBUJO:**

- Los degradados requieren más memoria, por lo que debemos evitar su uso excesivo, en la medida de lo posible.
- La herramienta **Pincel** gasta más memoria que el resto de herramientas de dibujo, por lo que deberíamos elegir estas últimas en la medida de lo posible.
- El uso de líneas que no sean las definidas por defecto y que usamos en el capítulo que hemos comentado, hará que el tamaño de la descarga aumente. Por tanto evitemos las líneas discontinuas, de puntos ...
- Dibujar las curvas con el menor número de nodos posible.

**CONSIDERACIONES EN LA ORGANIZACIÓN:**

- Agrupar los objetos que estén relacionados, con el comando **Modificar → Agrupar**.
- Si hemos creado un objeto que va a aparecer varias veces, deberíamos convertirlo a símbolo, ya que como hemos visto, Flash lo colocará en la biblioteca y cada vez que quiera mostrarlo, hará referencia a una única posición de memoria.
- Ya hemos comentado el mayor tamaño de los mapas de bits, lo que hace que debamos minimizar el número de apariciones de éstos en nuestra película.

**CONSIDERACIONES EN LOS TEXTOS:**

- Hemos podido observar, cuando manejábamos textos, que cuando abrimos el menú de tipos de letras, las tres primeras son siempre "_sans", "_serif" y "_typewriter". Esto no es una casualidad. Están colocadas ahí para resaltar que estas fuentes ocupan un mínimo de memoria, por lo que se recomienda su uso.

**CONSIDERACIONES EN LA ANIMACIÓN:**

- Utilizar lo más que podamos las interpolaciones de movimiento y las guías para reducir el número de fotogramas clave y el tamaño de la película.
- Evitar el uso de la interpolación por forma para animaciones de cambio de color, cuando sea posible.
- Independientemente de la optimización que hagamos, a veces no se puede evitar que el tamaño de la película aumente. Es recomendable entonces hacer un preloader (precarga) cuando la película que queramos publicar sea de tamaño superior a unos 80KB.

### 7.15.2 Preloader. Cargar la Película entera antes de reproducirla

Un preloader se usa principalmente para evitar la carga parcial de la película, mientras ésta se está reproduciendo, lo que, en ocasiones en las que la película es de un tamaño considerable, hace que la película se vea entrecortada.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

Normalmente, los preloaders se hacen vistosos para que el observador no se aburra y deje de lado la opción de visitar nuestra Web. Suelen llevar alguna animación sencilla que se va reproduciendo mientras se está cargando simultáneamente la película principal, mucho más grande.

Se pueden complicar mucho más, pero nosotros haremos uno sencillo que nos sirva para entender bien el concepto, y la manera de hacerlo.

Partimos de que ya tenemos nuestra película terminada. Si queremos saber su tamaño podemos ir a **Archivo → Configuración de publicación** seleccionar la pestaña Flash, y marcar la casilla **Generar Informe de Tamaño**. Si pulsamos el botón **Publicar**, aparecerá en nuestro directorio un archivo de texto donde se explica con detalle el tamaño de nuestra película.

Ahora insertaremos una nueva escena (**Insertar→ Escena**). Deberá ser la primera que se ejecute. Para asegurarnos de ello accedemos a **Ventana → Otros Paneles→ Escena**, y en la ventana que aparece arrastramos la escena que acabamos de crear hasta que esté la primera.

En la escena recién creada insertaremos otra capa, de manera que nos queden dos capas a las que llamaremos, "Acción" y "Cargando".

En la capa "Cargando" crearemos una animación sencilla. Por ejemplo, hagámosle honor al título y escribamos "Cargando ..."; puedes aplicarle la animación que prefieras, siempre que no sea muy compleja. En nuestro ejemplo, esta capa tiene por lo menos dos fotogramas.

En la capa "Acción" diseñaremos el "corazón" del preloader. Vamos a hacer que la animación de nuestra escena de carga se ejecute repetidas veces, hasta que se haya cargado la escena que contiene la película principal, mediante las acciones ActionScript 3 de Flash. Para ello abrimos el panel de Acciones.

Vamos a emplear la función `gotoAndPlay` de ActionScript, que como ya comentamos nos permite ir a un fotograma determinado. Podemos escribirlo `gotoAndPlay(1, "Escena")` para ir al primer fotograma de la escena indicada.

Podemos saber cuántos fotogramas se han cargado hasta ahora con la propiedad `this.framesLoaded`, y cuantos fotogramas hay en total con `this.totalFrames`.

Conociendo estos datos, lo único que tenemos que hacer es preguntar si los fotogramas cargados igualan a los totales . Si es que sí, ya podemos avanzar hasta la siguiente escena. Y si no, podemos volver al principio de nuestro loader, lo que lo irá repitiendo en bucle.

Para expresar esto en ActionScript, lo haríamos así:

```
if(this.framesLoaded==this.totalFrames) {     gotoAndPlay(1, "Película") }
else {       gotoAndPlay(1, "Cargador") }
```

Lo que hará esta instrucción es reproducir la escena Cargador, y al final comprobar el estado de la carga. Si se no se ha completado, vuelve al principio del cargador, lo que hará que vuelva a pasar por la instrucción. Cuando la carga esté completa, iniciamos la Película.

### 7.15.3 Distribución como archivo swf en un reproductor autónomo

Para poder distribuir películas creadas en Flash que la gente pueda ver, son necesarias dos cosas: crear un archivo SWF y que el que la quiera visualizar tenga instalado el Reproductor de Flash.

Flash nos ofrece varias opciones y funcionalidades para la creación de un archivo SWF. Estas opciones se pueden ver en el panel de Configuración de Publicación, al que podemos acceder mediante el menú **Archivo → Configuración de Publicación** (Pestaña Flash).

Veamos cuáles son estas opciones:

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Reproductor**: Si queremos publicar nuestra película para que sea vista con versiones anteriores de Flash, debemos seleccionar aquí la versión deseada.
- **Versión de ActionScript**: El uso de ActionScript 3 nos permitirá usar las novedades relativas a objetos, clases etc... Si hemos introducido código ActionScript debemos de respetar la versión elegida al crear el archivo, si no se pueden producir errores.
- **Calidad JPEG**: Si en el panel de propiedades del mapa de bits no hemos indicado una compresión concreta, aquí podremos determinar su grado de compresión, que determinará a su vez el espacio ocupado en memoria por este tipo de imágenes. A mayor compresión, menos espacio en memoria ocupará la imagen, pero también su calidad será menor.
- **Establecer Flujo de Audio o Evento de Audio**: Esta opción nos permite acceder al Panel "Configuración de Sonido" desde donde podemos configurar, para cada tipo de sonidos, sus características.
- **Suplantar configuración de sonido**: Con esto se suplantarán los niveles de compresión seleccionados para cada archivo de sonido de nuestro documento.
- **Comprimir película**: Comprime la película al máximo posible.
- **Generar Informe de tamaño**: Esta opción la hemos usado en el apartado anterior. Si la activamos, se creará un archivo de texto con una relación detallada del tamaño del documento.
- **Proteger Frente a Importación**: Activando está casilla hará que cuando otro usuario (o nosotros mismos) queramos importarla no podamos o tengamos que introducir una contraseña si se ha escogido alguna.
- **Omitir acciones de traza**: Las acciones de traza se emplean para comprobar el correcto funcionamiento de la película durante la creación de esta (durante las pruebas). También se consideran trazas los comentarios que insertemos en el código ActionScript. Si activamos esta señal, la película creada no los incluirá, ocupará menos tamaño y ahorraremos tiempo innecesario. Es recomendable cuando se publique la película de un modo definitivo.
- **Permitir depuración**: Permite que se pueda depurar el archivo SWF. También exige la introducción de una contraseña ya que se debe tener permiso del creador para Importar el archivo y depurarlo.

### 7.15.4 Distribución para páginas Web

Para publicar una película Flash en Internet de manera que forme parte de una página Web deberemos insertarla en un archivo típico de páginas Web cuyo lenguaje de programación sea del estilo del HTML. Para ello debemos atender a las opciones de publicación HTML que nos ofrece Flash, y que nos ayudarán a que nuestra película se visualice como realmente queremos.

Las opciones de este tipo de publicación están en **Archivo → Configuración de Publicación...** (Pestaña HTML).


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Plantilla:** Para incrustar una película Flash en un documento HTML, hay que escribir una serie de códigos de programa algo complejos y laboriosos de hacer a mano.

  Para facilitarnos esta tarea Flash hace esto automáticamente pero, puesto que cada Web es distinta y nuestras necesidades van a ser muy distintas, los códigos también serán muchos y distintos, por esto Flash incluye Plantillas, que crean este código automáticamente según el tipo de publicación que deseemos:

  En el botón Información que está a la derecha de la pestaña "Plantilla" se nos muestra información muy útil sobre cada tipo de plantilla. Estas son las plantillas más comunes:

  - **Sólo Flash**: Esta es la opción predeterminada y utiliza el reproductor Flash.
  - **Flash con FSCommand**: Imprescindible cuando se usen FSCommands. Permiten controlar el reproductor desde la película.
  - **Mapa de Imágenes**: Si hemos incluido una Image Map (imágenes completas que ejecutan distintas acciones según la coordenada que se pulse) debemos activar esta opción.
  - **QuickTime:** Permite incluir una película QuickTime.
  - etc ...

- **Detectar Versión de Flash**: Desde aquí podemos seleccionar si queremos que nuestra película detecte la existencia o no existencia del plugin de Flash en el ordenador del usuario, así como las páginas Web donde se insertará el código encargado de comprobarlo y las páginas Web a las que se irá en caso de disponer del Plugin o no disponer de él.
- **Dimensiones**: Especifica la unidad en la que mediremos las dimensiones del Documento.
- **Anchura X Altura**: Aquí introduciremos la anchura y altura, teniendo en cuenta que a veces un objeto más grande que estas dimensiones provocará un cambio en éstas.
- **Reproducción**: Permite realizar determinados cambios en cuanto a la reproducción de la película:
  - **Pausa al Comienzo**: Permite que sea el usuario quien haga que se inicie la reproducción, que inicialmente aparecerá detenida.
  - **Reproducción Indefinida**: Cuando la película termine, volverá a empezar desde el principio. Esto lo hará infinitamente.
  - **Visualizar Menú**: Permite que al hacer el usuario clic con el botón derecho del ratón sobre la película, el menú emergente tenga todas las opciones por defecto. Si la desactivamos sólo aparecerá la opción "Acerca de Flash".
  - **Fuentes de Dispositivo**: Sustituye las fuentes utilizadas en los textos sin animación de la película por las fuentes predeterminadas en la máquina de quien la visualice.

ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242

Ediciones Universidad de Salamanca - cc by nc dc

- **Calidad**: Aquí podemos modificar la calidad de visualización de la película, que depende del suavizado de la imagen y el tiempo de reproducción. Las opciones son:
  - **Baja No hay suavizado**. El tiempo de reproducción es el de máxima velocidad.
  - **Baja Automática**: El reproductor detecta si la máquina soporta en cada instante un ligero suavizado, si lo soporta, lo aplica.
  - **Alta Automática**: Pone al mismo nivel el tiempo y el suavizado, pero si hay alguna limitación, siempre dará preferencia a la velocidad.
  - **Media**: Valores intermedios de velocidad y suavizado. No suaviza los bitmaps.
  - **Alta**: Usa siempre el suavizado, los mapas de bits se suavizan sólo si no hay animación. Da preferencia a la buena visualización.
  - **Óptima**: Se suaviza todo, incluidos los mapas de bits en cualquier caso. Total preferencia de la apariencia frente a la velocidad.
- **Modo de Ventana**: Opciones para la reproducción dentro de las ventanas de Windows:
  - **Ventana**: Se reproduce la película en la ventana de la Web en la que está insertada.
  - **Opaco sin Ventanas**: Hace que los objetos situados en capas situadas detrás de la película no se vean (en páginas DHTML).
  - **Transparente sin Ventanas**: Es el opuesto al anterior. Permite que los objetos situados detrás se vean.
- **Alineación HTML**: Posición relativa de la película dentro de la página Web HTML. Tenemos varias opciones:
  - **Predeterminada**: Centra la película en la página. Si no cabe se cortan los extremos.
  - **Izquierda**: Alineación a la izquierda. También se recortan los bordes si no cabe.
  - **Derecha**: Alineación a la derecha.
  - **Superior**: Alineación en el borde superior de la página.
  - **Inferior**: Alineación en el borde inferior de la página.
- **Escala**: Si hemos especificado el tamaño en píxeles o en tanto por ciento, podemos decirle a Flash cómo distribuir la película en el rectángulo que hemos decidido que la contenga:
  - **Predeterminada**: Se ve toda la película guardando las proporciones originales.
  - **Sin Borde**: Recorta (en caso de que la película sea más grande que el rectángulo) todo lo que sobre con el fin de mantener las proporciones.


ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

- **Ajuste Exacto**: Distorsiona las proporciones si es necesario para conseguir que la película ocupe el rectángulo completo.

- **Alineación Flash**: Se hace necesario alinear la película cuando esta no tiene las mismas dimensiones que el rectángulo definido. Las opciones son:
  - **Alineación Horizontal**: Podemos escoger entre Centro, Izquierda o Derecha.
  - **Alineación Vertical**: Podemos escoger entre Centro, Superior o Inferior.
- **Mostrar Mensajes de advertencia**: Permite que se muestren los posibles mensajes de error de código ActionScript.

Además, Flash creará el código JavasScript que detecte la versión de Flash y otras acciones.

De todas formas, lo más habitual es utilizar un editor Web, como Dreamweaver para crear la página Web, y desde ahí insertar el archivo SWF, por lo que será el propio editor el que se encargue de generar este código.

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

## References


1. Anderson Sergio, Sidartha Carvalho, Marco Rego (2014). On the Use of Compact Approaches in Evolution Strategies. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 4
2. Buciarelli, E., Silvestri, M., & González, S. R. (2016). Decision Economics, In Commemoration of the Birth Centennial of Herbert A. Simon 1916-2016 (Nobel Prize in Economics 1978): Distributed Computing and Artificial Intelligence, 13th International Conference. Advances in Intelligent Systems and Computing (Vol. 475). Springer.
3. Casado-Vara, R., Chamoso, P., De la Prieta, F., Prieto J., & Corchado J.M. (2019). Non-linear adaptive closed-loop control system for improved efficiency in IoT-blockchain management. Information Fusion.
4. Casado-Vara, R., Novais, P., Gil, A. B., Prieto, J., & Corchado, J. M. (2019). Distributed continuous-time fault estimation control for multiple devices in IoT networks. IEEE Access.
5. Casado-Vara, R., Vale, Z., Prieto, J., & Corchado, J. (2018). Fault-tolerant temperature control algorithm for IoT networks in smart buildings. Energies, 11(12), 3430.
6. Casado-Vara, R., Prieto-Castrillo, F., & Corchado, J. M. (2018). A game theory approach for cooperative control to improve data quality and false data detection in WSN. International Journal of Robust and Nonlinear Control, 28(16), 5087-5102.
7. Chamoso, P., González-Briones, A., Rivas, A., De La Prieta, F., & Corchado J.M. (2019). Social computing in currency exchange. Knowledge and Information Systems.
8. Chamoso, P., González-Briones, A., Rivas, A., De La Prieta, F., & Corchado, J. M. (2019). Social computing in currency exchange. Knowledge and Information Systems, 1-21.
9. Chamoso, P., González-Briones, A., Rodríguez, S., & Corchado, J. M. (2018). Tendencies of technologies and platforms in smart cities: A state-of-the-art review. Wireless Communications and Mobile Computing, 2018.
10. Chamoso, P., Rodríguez, S., de la Prieta, F., & Bajo, J. (2018). Classification of retinal vessels using a collaborative agent-based architecture. AI Communications, (Preprint), 1-18.
11. Constantino Martins, Ana Rita Silva, Carlos Martins, Goreti Marreiros (2014). Supporting Informed Decision Making in Prevention of Prostate Cancer. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 3
12. Eva L. Iglesias, Lourdes Borrajo, R. Romero (2014). A HMM text classification model with learning capacity. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 3
13. García, O., Chamoso, P., Prieto, J., Rodríguez, S., & De La Prieta, F. (2017). A serious game to reduce consumption in smart buildings. In Communications in Computer and Information Science (Vol. 722, pp. 481–493). https://doi.org/10.1007/978-3-319-60285-1_41
14. Gonzalez-Briones, A., Chamoso, P., De La Prieta, F., Demazeau, Y., & Corchado, J. M. (2018). Agreement Technologies for Energy Optimization at Home. Sensors (Basel), 18(5), 1633-1633. doi:10.3390/s18051633
15. González-Briones, A., Chamoso, P., Yoe, H., & Corchado, J. M. (2018). GreenVMAS: virtual organization-based platform for heating greenhouses using waste energy from power plants. Sensors, 18(3), 861.
16. Gonzalez-Briones, A., Prieto, J., De La Prieta, F., Herrera-Viedma, E., & Corchado, J. M. (2018). Energy Optimization Using a Case-Based Reasoning Strategy. Sensors (Basel), 18(3), 865-865. doi:10.3390/s18030865
17. Jamal Ahmad Dargham, Ali Chekima, Ervin Gubin Moung, Sigeru Omatu (2014). The Effect of Training Data Selection on Face Recognition in Surveillance Application. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 4
18. Juan Carlos Alvarado-Pérez, Diego H. Peluffo-Ordóñez, Roberto Therón (2015). Bridging the gap between human knowledge and machine learning. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 4, n. 1
19. Juan Castro, Pere Marti-Puig (2014). Real-time Identification of Respiratory Movements through a Microphone. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 3
20. Margherita Brondino, Gabriella Dodero, Rosella Gennari, Alessandra Melonio, Daniela Raccanello, Santina Torello (2014). Achievement Emotions and Peer Acceptance Get Together in Game Design at School. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 4
21. Miki Ueno, Naoki Mori, Keinosuke Matsumoto (2014). Picture models for 2-scene comics creating system. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 2




ISBN: 978-84-9012-862-6
Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc

22. Ming Fei Siyau, Tiancheng Li, Jonathan Loo (2014). A Novel Pilot Expansion Approach for MIMO Channel Estimation. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 3, n. 3
23. Mohamed Frikha, Mohamed Mhiri, Faiez Gargouri (2015). A Semantic Social Recommender System Using Ontologies Based Approach For Tunisian Tourism. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 4, n. 1
24. Sittón-Candanedo, I., Alonso, R. S., Corchado, J. M., Rodríguez-González, S., & Casado-Vara, R. (2019). A review of edge computing reference architectures and a new global edge proposal. Future Generation Computer Systems, 99, 278-294.
25. Palomino, C. G., Nunes, C. S., Silveira, R. A., González, S. R., & Nakayama, M. K. (2017). Adaptive agent-based environment model to enable the teacher to create an adaptive class. Advances in Intelligent Systems and Computing (Vol. 617). https://doi.org/10.1007/978-3-319-60819-8_3
26. Román, J. A., Rodríguez, S., & de da Prieta, F. (2016). Improving the distribution of services in MAS. Communications in Computer and Information Science (Vol. 616). https://doi.org/10.1007/978-3-319-39387-2_4
27. Silvia Rossi, Francesco Barile, Antonio Caso (2015). Dominance Weighted Social Choice Functions for Group Recommendations. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal (ISSN: 2255-2863), Salamanca, v. 4, n. 1
28. Sittón, I., & Rodríguez, S. (2017). Pattern Extraction for the Design of Predictive Models in Industry 4.0. In International Conference on Practical Applications of Agents and Multi-Agent Systems (pp. 258–261).

ISBN: 978-84-9012-862-6

Colección Aquilafuente: AQ-242
Ediciones Universidad de Salamanca - cc by nc dc