Índice de tutoriales

Introducción

Flash CS4 es una potente herramienta que nos permite crear diferente tipo de contenido, como por ejemplo gráficos vectoriales, animaciones, recursos interactivos, aplicaciones multimedia, juegos, etc.

En esta guía comenzaremos conociendo las herramientas de edición gráfica, para después adentrarnos en las nuevas posibilidades de animación que ofrece Flash CS4, como por ejemplo el editor de movimiento, la animación de huesos y las herramientas 3D. También añadiremos sonido a nuestras animaciones

Para añadir interactividad, nos introduciremos en la programación con ActionScript. Aprenderemos a controlar desde la reproducción de una animación, hasta el caso más complejo de la creación de un juego. También aprenderemos a trabajar con contenido externo (texto, imágenes y vídeos) para conseguir aplicaciones dinámicas.

Por último, conoceremos distintas formas de publicar el contenido que hayamos creado.

Tutorial 1. Creación de una ilustración vectorial

Paso 1 de 14

Esta es la pantalla de bienvenida que aparece al ejecutar Flash. En esta pantalla tenemos cinco áreas:

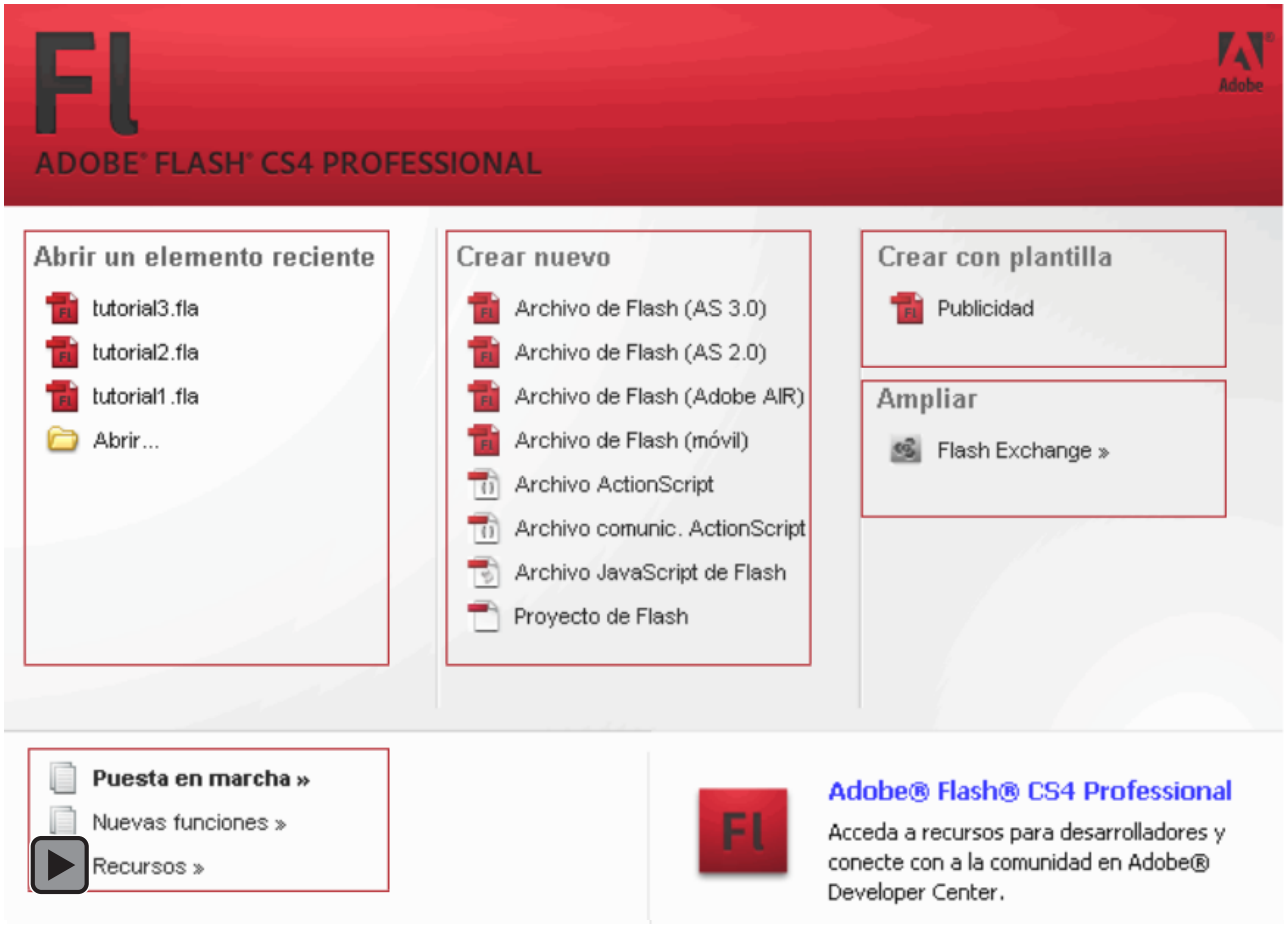
-**Abrir un elemento reciente**. Acceso directo a archivos recientes y opción **Abrir** para buscar otros archivos.

-**Crear nuevo**. Creación de un documento nuevo de Flash del tipo que seleccionemos. Para comenzar seleccionaremos **Crear nuevo > Archivo de Flash (AS 3.0)**.

-**Crear con plantilla**. Abre un nuevo archivo a partir de una plantilla.

-**Ampliar**. Vínculo a Flash Exchange, un sitio de intercambio de recursos.

-**Puesta en marcha**, **Nuevas funciones** y **Recursos.** Vínculos a sitios de ayuda de Flash.



Desplaza el cursor por encima de las diferentes áreas para una información más detallada.

Tutorial 1. Creación de una ilustración vectorial

Paso 2 de 14

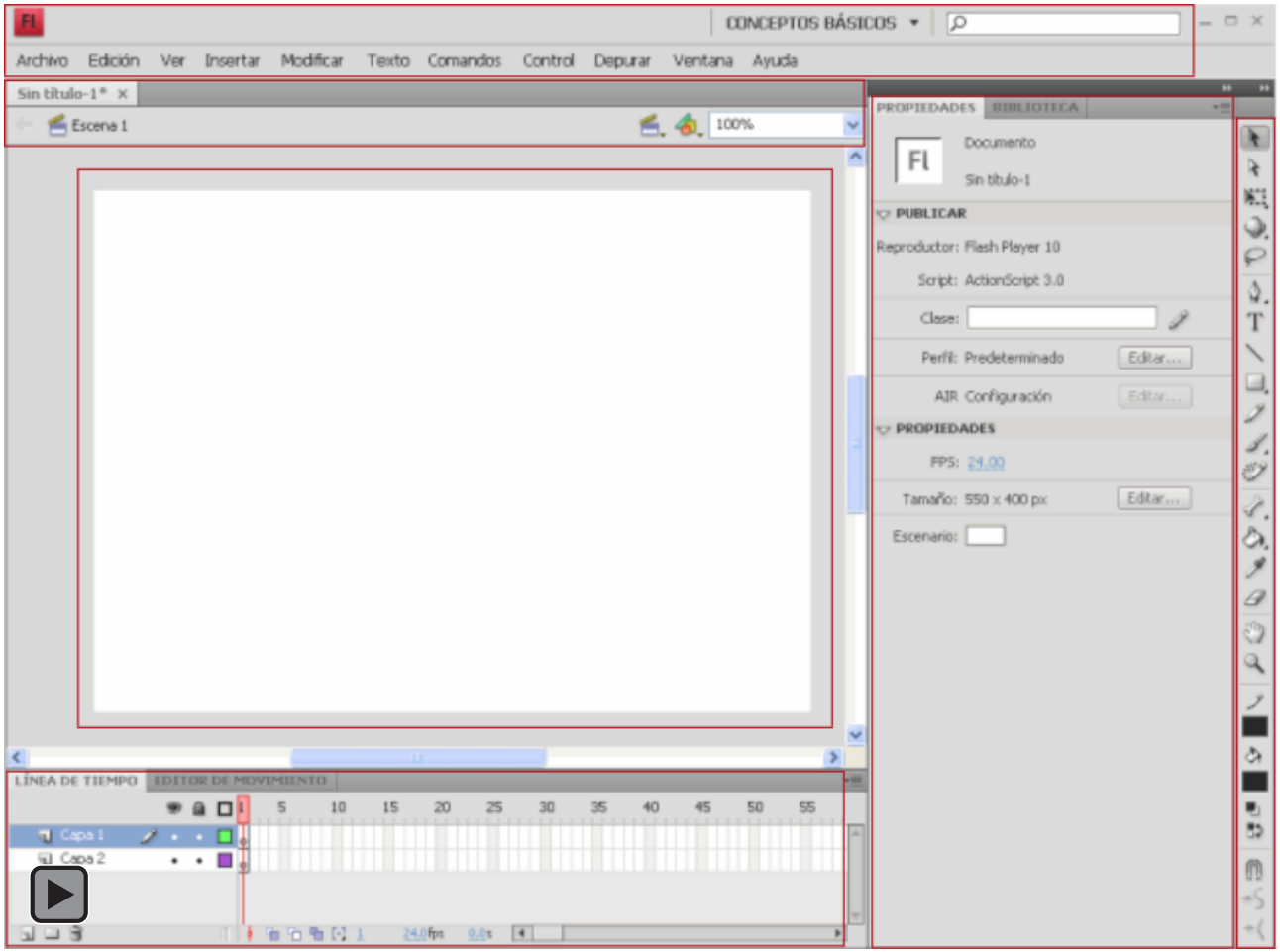
En Flash CS4 podemos elegir entre varios espacios de trabajo , así como crear nuestro espacio de trabajo personalizado.

Cada espacio de trabajo tiene una distribución diferente de las diferentes áreas, y puede mostrar por defecto unos paneles u otros.

Podemos cambiar a diferentes espacios de trabajo seleccionando en el desplegable, situado en la parte superior al lado del buscador, el espacio de trabajo que queramos.

En esta guía trabajaremos con el espacio de trabajo llamado **Conceptos básicos**.

Para ver una descripción de cada zona de este espacio, desplazad el cursor por las diferentes áreas.



Tutorial 1. Creación de una ilustración vectorial

Paso 3 de 14

Generalmente, el primer paso que haremos al trabajar con un archivo Flash es configurar las características principales de la película.

Podemos ver las características de nuestra película en el **inspector de propiedades**. Si no vemos este panel, pulsaremos sobre el menú **Ventana > Propiedades***.

Si en este panel vemos otros datos que no corresponden con la imagen, seleccionaremos la herramienta **Selección** pulsando sobre la flecha superior de la barra de herramientas o bien pulsando la tecla de acceso directo **V**. Este acceso directo puede ser muy útil, ya que es una de las herramientas que más utilizaremos.

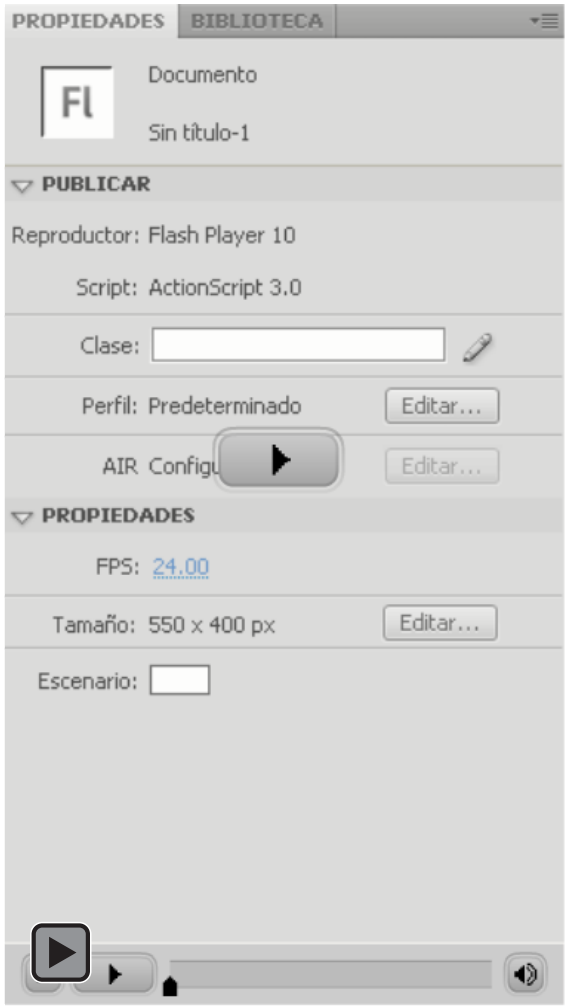
Con la herramienta **Selección**, pulsaremos sobre un lugar vacío del escenario para asegurar que no haya nada seleccionado.

En la parte superior de este panel vemos el título de documento. Después está el área **Publicar**, de la que hablaremos en otros tutoriales. Por último está el área de **Propiedades**.

FPS muestra los fotogramas por segundo a los que está configurada la película. A mayor número de FPS, mayor velocidad y fluidez de la animación. Para algunas animaciones puede ser suficiente un valor de 12 o 18 FPS, pero de momento dejamos este valor en 24 fotogramas por segundo.

Después tenemos el **tamaño** del escenario. Si queremos cambiar este valor, pulsaremos sobre el botón Editar que tenemos a su lado. Para este tutorial no vamos a modificar el tamaño.

Escenario muestra el color de fondo de nuestra película. Vamos a cambiar el color de fondo pulsando sobre el cuadro de color y seleccionando un color azulado, tal y como se puede ver en este vídeo.



*Siempre que un panel no está visible, podemos abrirlo clicando en su nombre desde el menú **Ventana**.

Tutorial 1. Creación de una ilustración vectorial

Paso 4 de 14

Para comenzar vamos a dibujar una nube trazando varios círculos superpuestos.

En primer lugar seleccionaremos la **herramienta Óvalo**, que se encuentra agrupada junto con la herramienta **Rectángulo**. Todas las herramientas que tienen otras herramientas agrupadas muestran en la barra de herramientas una pequeña flecha negra. Manteniendo pulsada la herramienta correspondiente podemos acceder al resto de herramientas que están agrupadas junto a ella.

Una vez seleccionada la herramienta **Óvalo**, el inspector de propiedades mostrará atributos relacionados con esta herramienta. Lo único que vamos a asegurar de momento es que el **trazo** sea negro y el **relleno** blanco. De no serlo, pulsaremos sobre el cuadro de color correspondiente para cambiarlo. Estos valores también se pueden cambiar directamente desde el panel de Herramientas. El trazo se refiere al color del perfil del objeto, y el relleno al color que muestra en su interior.

Las **Opciones de óvalo** sólo pueden modificarse antes de dibujar el óvalo, salvo que hubiéramos elegido la herramienta **Óvalo simple**, en cuyo caso podríamos modificar estos valores después de haber dibujado el objeto.

Sin embargo, con Óvalo simple el objeto permanece agrupado e independiente, por lo que no se combina con otras formas al superponerse.

Entre Rectángulo y Rectángulo simple existen estas mismas diferencias.

En este caso vamos a superponer y combinar varios círculos, por lo que hemos elegido la **herramienta Óvalo**.

Tutorial 1. Creación de una ilustración vectorial

Paso 5 de 14

Dibujo de objeto, un modificador que aparecerá en la parte inferior de la barra de Herramientas, deberá estar también desactivado, ya que de lo contrario la forma creada también permanecerá agrupada y, por tanto, no la podremos combinar con otras formas.

Desactivaremos también el modificador **Ajustar a objetos** para tener una mayor libertad a la hora de crear las formas ovaladas y situar los círculos que dibujemos. Por ejemplo, si este modificador está activado, los óvalos casi redondos se convertirán en círculos perfectos, lo cual en este caso no nos interesa.

Ahora dibujaremos una nube situándola en la esquina superior izquierda del escenario.

Con la herramienta **Óvalo** seleccionada, dibujamos cuatro óvalos en el escenario de forma similar a como se muestra en el vídeo.

Después, con la herramienta **Selección** (la flecha superior de la barra de herramientas, que también se puede activar pulsando la tecla **V**), clicamos sobre alguno de los trazos negros. Como podremos ver, quedará seleccionado un tramo del trazo, ya que los óvalos se han combinado para formar ahora un solo dibujo. Eliminamos el trazo seleccionado pulsando la tecla **Supr**.

Podemos eliminar el resto de los trazos de la misma manera, es decir, seleccionando cada tramo y borrándolo con la tecla Supr.

Sin embargo, como todos los trazos se encuentran unidos por algún punto, una forma más rápida de borrarlos es haciendo **doble clic** sobre cualquier trazo, y con ello quedarán seleccionados todos los trazos que estén unidos. De esta forma, al pulsar la tecla **Supr** se borrarán todos los trazos seleccionados.

Tutorial 1. Creación de una ilustración vectorial

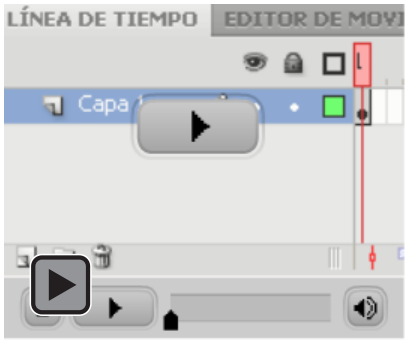
Paso 6 de 14

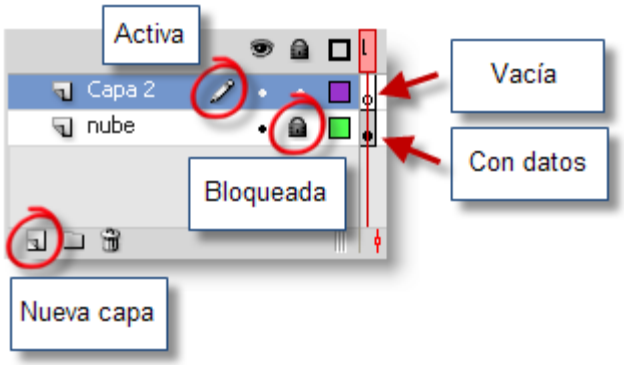
Si observamos la **línea de tiempo**, podemos ver que ahora el primer fotograma de la *Capa 1* tiene un punto negro sobre un fondo gris, lo que indica que nuestro dibujo ha quedado incluido en este fotograma.

Las capas, además de definir el orden de apilamiento de los distintos elementos que tengamos en el escenario, también son una buena forma de organizar y mantener independientes las diferentes partes de una ilustración. Es por tanto muy recomendable crear una capa diferente para cada elemento.

En primer lugar **renombramos** la capa actual como *nube* haciendo doble clic sobre su nombre. Para evitar modificar su contenido accidentalmente, la **bloqueamos** haciendo clic sobre el punto que hay bajo el candado.

Por último, crearemos una **nueva capa**. Si seleccionamos esta nueva capa, un pequeño lápiz junto a su nombre indicará que se encuentra activa.





Antes de continuar, guardad el archivo con el nombre *tutorial1.fla* seleccionando **Archivo > Guardar**.

Es conveniente que nos acostumbremos a guardar el archivo regularmente a medida que vayamos avanzando con el tutorial.

Tutorial 1. Creación de una ilustración vectorial

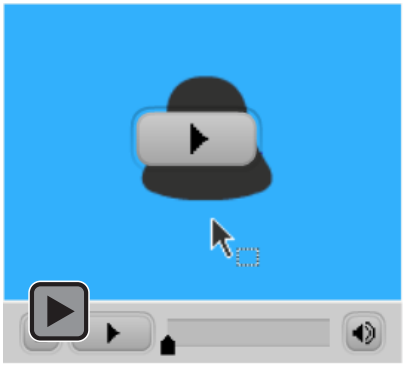
Paso 7 de 14

Vamos a dibujar un pájaro con las alas extendidas, comenzando por el cuerpo.

En primer lugar vamos a asignar que no haya **trazo** (representado por un rectángulo blanco con una franja roja) y un color de **relleno** negro.

Esta vez cambiaremos los colores en la propia barra de herramientas. El resultado es el mismo que si lo hacemos en el inspector de propiedades.

Una de las formas más sencillas de comenzar a dibujar es empezar dibujando formas básicas, como por ejemplo óvalos o rectángulos, y modificando después esas formas con la herramienta Selección. Vamos a dibujar el pájaro de esta forma.



Para crear el cuerpo del pájaro, dibujaremos una forma ovalada con la **herramienta Óvalo**. Después modificaremos el óvalo con la **herramienta Selección (V)** hasta conseguir una forma acampanada similar a la del vídeo.

Cuando acercamos el puntero a una forma con la herramienta Selección, el puntero cambia para indicar el tipo de modificación que se puede realizar en la línea o el relleno: si aparece una esquina podremos modificar un extremo, si aparece una curva podremos ajustar una curva, y si aparece una cruz podremos desplazar el objeto.

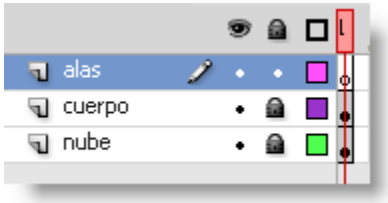
Siempre que queramos hacer una de estas modificaciones tendremos que clicar cuando el puntero adopte la forma que busquemos y arrastrar hasta el lugar deseado.

En este caso, situándonos en el borde del óvalo, sólo aparecía la forma curvada, ya que al tratarse de una forma ovalada no hay ninguna esquina.

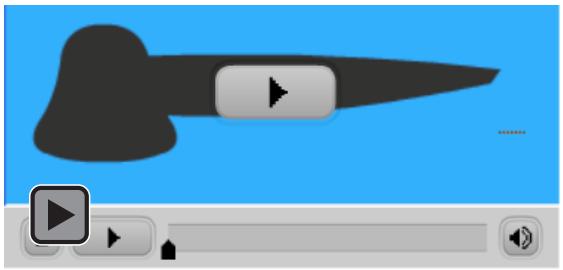
Tutorial 1. Creación de una ilustración vectorial

Paso 8 de 14

De forma similar a como hicimos en el paso 6, llamaremos *cuerpo* a la capa actual, la bloquearemos, y crearemos una nueva capa llamada *alas*.



Para dibujar el ala utilizaremos la **herramienta Rectángulo**, y posteriormente modificaremos la forma creada con la **herramienta Selección**.



Como podemos ver en el vídeo, en este caso sí que podemos modificar tanto esquinas como curvas. La forma del puntero ayuda a encontrar las diferentes curvas y esquinas incluso en el área en la que se superponen ambas formas negras (alas y cuerpo).

Por ahora nos interesa que el ala se muestre totalmente extendida aunque quede menos realista. Más adelante, en los tutoriales de animación, la doblaremos y moveremos convenientemente.

El ala izquierda la haremos a partir de una copia del ala derecha.

Normalmente, cuando vayamos a reutilizar un objeto, lo más adecuado es convertirlo previamente en símbolo. Sin embargo, para este primer tutorial vamos a copiar y pegar el ala directamente, sin convertirla en símbolo. El uso de los símbolos lo estudiaremos en el próximo tutorial.

Tutorial 1. Creación de una ilustración vectorial

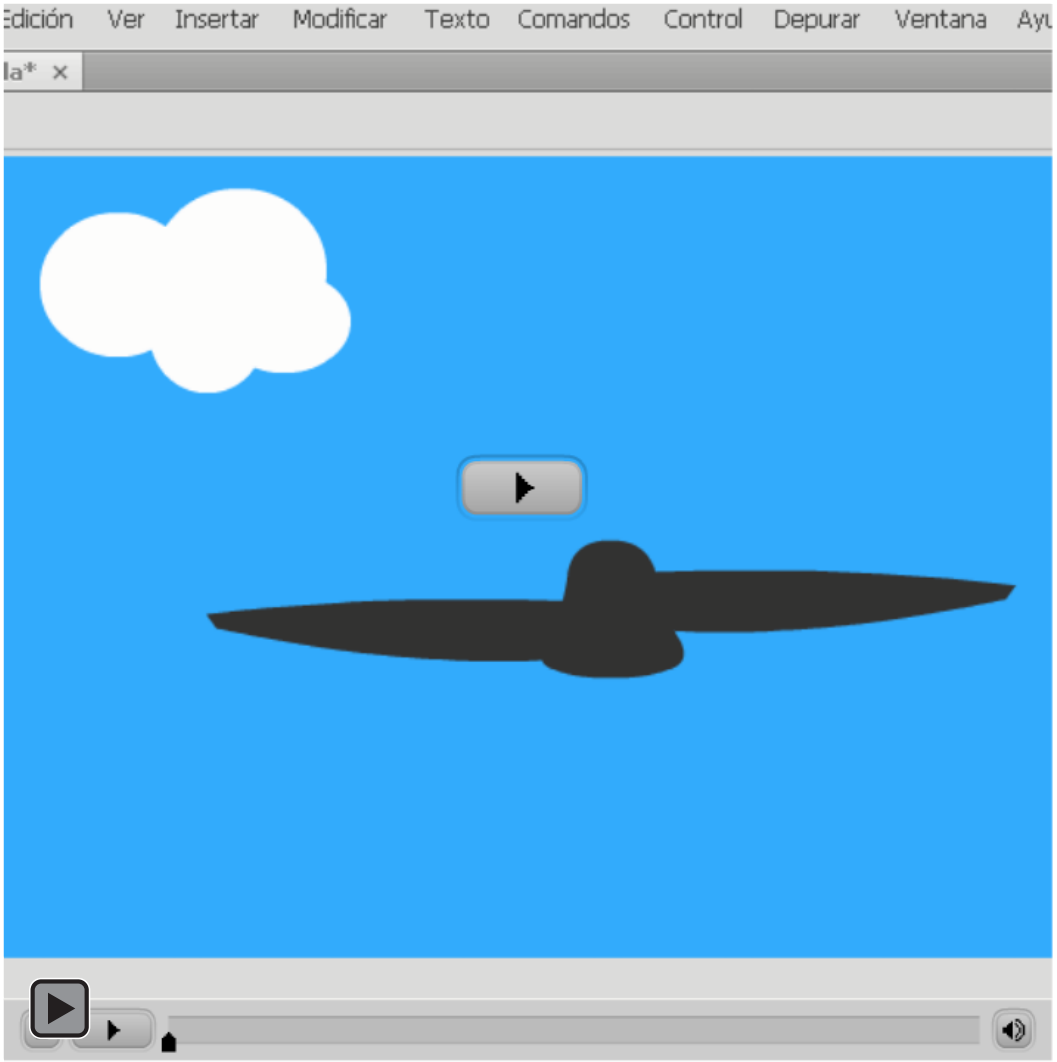
Paso 9 de 14

Clicamos sobre el ala con el **botón derecho** del ratón y seleccionamos **Copiar** del menú contextual.

Después, sobre algún lugar vacío del escenario, clicamos de nuevo con **el botón derecho** del ratón y seleccionamos **Pegar**.

Con la copia del ala seleccionada, seleccionamos **Modificar > Transformar > Voltear horizontalmente**.

Clicamos sobre el ala y la arrastramos hasta una posición aproximada a su ubicación final. En el siguiente paso, con ayuda del **panel Alinear**, la colocaremos en su posición definitiva.



Tutorial 1. Creación de una ilustración vectorial

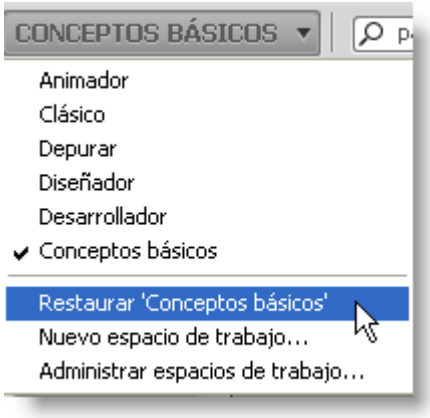
Paso 10 de 14

Abrimos el panel Alinear seleccionando **Ventana > Alinear**.

En Flash podemos colocar los paneles en el lugar que nos resulte más cómodo, pudiendo personalizar de esta forma nuestro espacio de trabajo según nuestras necesidades.

En el vídeo se muestran algunas de las opciones de visualización que tienen todos los paneles, tales como desplazarse y minimizarse de varias maneras.

También es posible agrupar los paneles junto con otras áreas del espacio de trabajo. Si en cualquier momento queremos volver al espacio de trabajo por defecto, seleccionaremos **Restaurar** en el desplegable del espacio de trabajo.



Si no disponemos de mucho espacio para trabajar, lo más cómodo será abrir los paneles a medida que los vayamos necesitando, reubicarlos en un lugar que nos resulte cómodo, y cerrarlos tras su uso.

Tutorial 1. Creación de una ilustración vectorial

Paso 11 de 14

Volviendo a nuestro pájaro, vamos en primer lugar a alinear las alas con ayuda del **panel Alinear**.

Lo primero que debemos comprobar es que la casilla *En escena* esté desactivada, ya que de lo contrario los elementos no se alinearán entre sí, sino respecto al escenario.

Seleccionamos las dos alas haciendo clic sobre ellas con la **herramienta Selección**. Para hacer una selección múltiple deberemos mantener pulsada la tecla **Mayúsculas**.

Con ambas alas seleccionadas hacemos clic en **Alinear borde superior**.

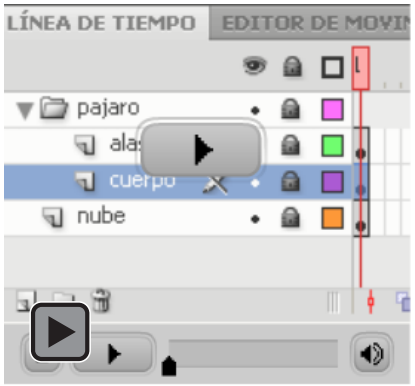
Desbloqueamos la capa del cuerpo del pájaro para poder seleccionarlo, y con los tres elementos seleccionados (el cuerpo y las alas) hacemos clic en **Distribuir horizontalmente respecto al centro**.

Para una mejor organización de los elementos de una película, podemos agrupar las capas en carpetas.

Vamos a crear en la **línea de tiempo** una **nueva carpeta** llamada *pájaro* que incluirá las capas de las alas y del cuerpo. Para ello, después de crear la carpeta, arrastraremos las capas a su interior.

Al bloquear la carpeta podemos observar que se bloquean todas las capas que contiene.

Después crearemos una nueva capa para añadir más elementos al escenario.



Recordad guardar regularmente vuestro trabajo con **Archivo > Guardar**.

.

Tutorial 1. Creación de una ilustración vectorial

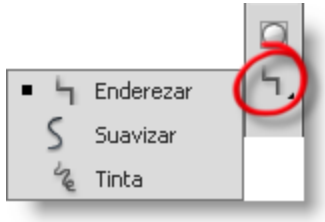
Paso 12 de 14

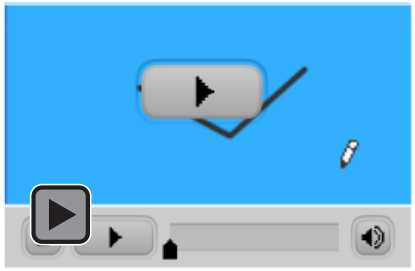
Vamos a hacer una pequeña aproximación a otras dos herramientas vectoriales de Flash, comenzando por la **herramienta Lápiz**.

Al activar el lápiz, veremos que en el **inspector de Propiedades** el relleno aparece automáticamente con una franja roja que significa que no hay relleno, ya que el lápiz sólo puede dibujar trazos. Seleccionaremos un color negro para el trazo, y un grosor de 3 píxeles.

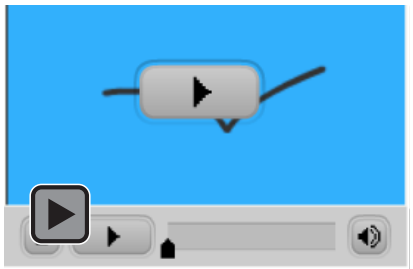
En la parte inferior de la barra de herramientas aparecerán los modificadores asociados a la herramienta lápiz.

Vamos a tratar de dibujar algunos pájaros con estos tres modificadores.

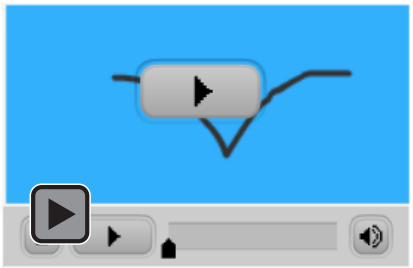




El modificador **Enderezar** convierte las líneas en rectas o en figuras geométricas comunes.



El modificador **Suavizar** dibuja curvas suaves.



El modificador **Tinta** dibuja curvas a mano alzada, sin aplicar ninguna modificación.

Una vez dibujada una forma, sea con estas o con otras herramientas, siempre podrá ser suavizada o enderezada clicando repetidamente en los modificadores **Suavizar** y **Enderezar** que aparecerán al seleccionar una forma en el escenario.

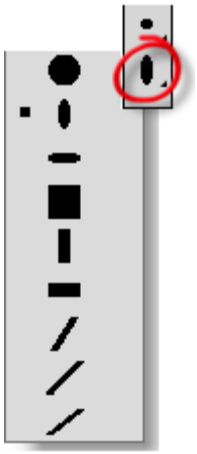
Tutorial 1. Creación de una ilustración vectorial

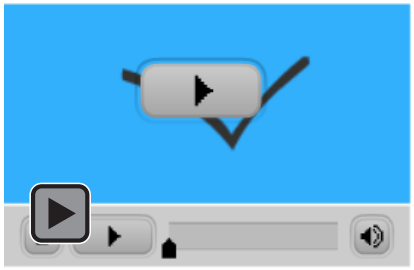
Paso 13 de 14

Por último vamos a utilizar la **herramienta Pincel**.

En la parte inferior de la barra de herramientas aparecerán los modificadores asociados a la herramienta Pincel.

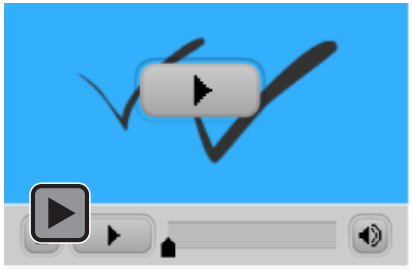
Los dos modificadores inferiores se refieren al **tamaño** y la **forma** del pincel. Vamos a seleccionar como **forma del pincel** una forma ovalada vertical.





El dibujo realizado con el pincel queda generalmente suavizado, por lo que es una buena herramienta para hacer este tipo de dibujos.

A diferencia del lápiz, el pincel pinta rellenos, y no trazos. Por tanto, el dibujo realizado con el pincel puede ser modificado como un relleno.

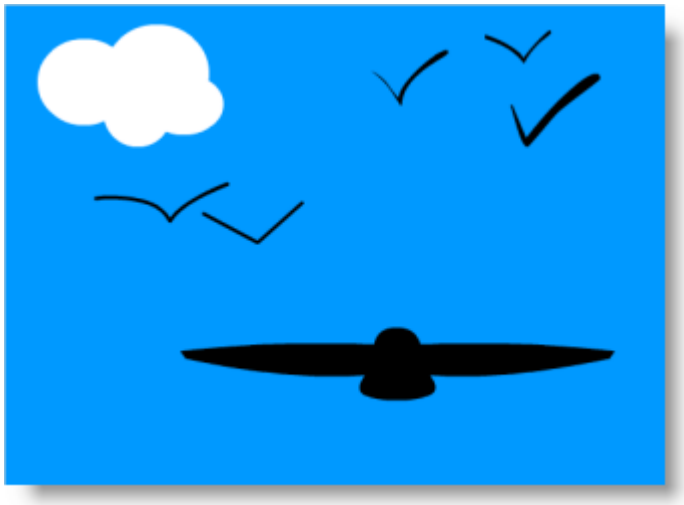


Si se tiene una tableta sensible a la presión aparecerán dos nuevos modificadores, el modificador **Presión** y el modificador **Inclinación**. Si se activan estos modificadores podremos variar la anchura y la inclinación de los trazos del pincel dependiendo de la presión e inclinación que ejerzamos con nuestro lápiz.

Los pájaros de estos dos últimos pasos los hemos dibujado solamente para experimentar con la herramienta lápiz y la herramienta pincel, pero el primer pájaro y la nube los utilizaremos en posteriores tutoriales sobre animación, así que es muy importante que guardéis vuestro trabajo. Para ello selecciona **Archivo > Guardar** o **Ctrl+S**.

Tutorial 1. Creación de una ilustración vectorial

Paso 14 de 14



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Dibujad otra nube y modifica después su forma con la herramienta Selección.
2. Modificad también la forma de los pájaros que hemos creado con el lápiz y con el pincel para comprender mejor las diferencias entre estas dos herramientas.
3. Cread una forma nueva partiendo de un óvalo o de un rectángulo.

Tutorial 2. Importación de archivos externos

Paso 1 de 24

En este tutorial vamos a crear un paisaje que utilizaremos en otros tutoriales. Por ello es importante que guardemos el resultado.

Para crear este paisaje vamos a importar y modificar archivos externos. Por un lado importaremos un gráfico vectorial creado con Adobe Illustrator que contiene la hierba y los árboles. Por otro lado trabajaremos con una imagen importada de mapa de bits para crear la luna.

Los gráficos vectoriales representan la imagen mediante líneas y curvas (vectores), mientras que los mapas de bits están formados por puntos (píxeles). Por ello la forma de trabajar con ambos tipos de formatos es diferente.

Para completar el paisaje utilizaremos la herramienta pluma para crear la montaña, y trabajaremos con los degradados de color tanto para el cielo como para la montaña.

Tutorial 2. Importación de archivos externos

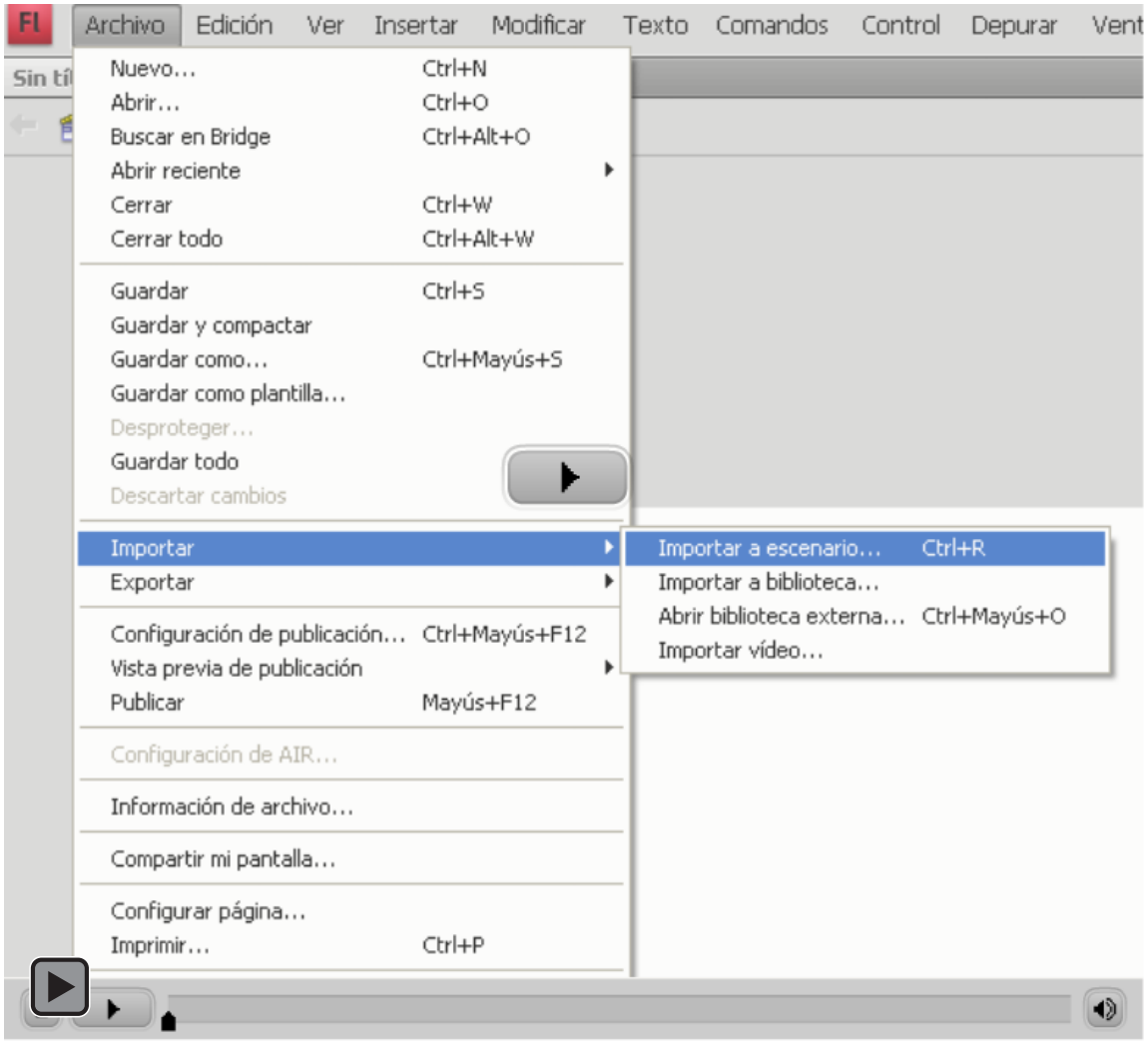
Paso 2 de 24

En primer lugar importaremos el archivo vectorial *arbol.ai* que se encuentra en la carpeta *tutorial2*. Para ello seleccionamos **Archivo > Importar > Importar a escenario**.

Durante la importación nos aseguramos de que la opción de convertir capas en Capas de Flash está seleccionada.

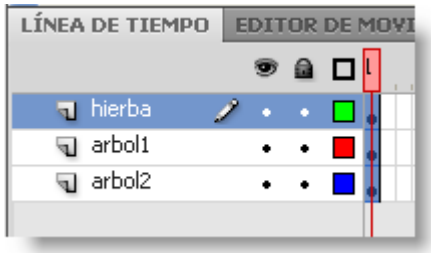
Una vez importado, guardamos el archivo con el nombre *tutorial2.fla.* Para ello seleccionamos **Archivo > Guardar** (Ctrl+S).

Es conveniente que nos acostumbremos a guardar el archivo regularmente a medida que vayamos avanzando con el tutorial.



Tutorial 2. Importación de archivos externos

Paso 3 de 24



Las capas que tenía originalmente el archivo que hemos importado aparecerán ahora en la **línea de tiempo**.

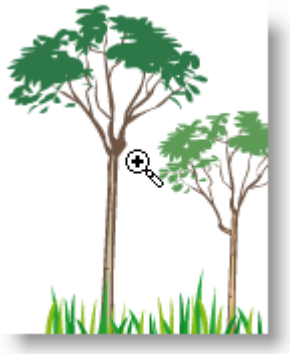
El **orden de las capas** determina cómo se superponen los elementos de la animación. Las capas inferiores se mostrarán más al fondo en el escenario. En este caso, la hierba se mostrará en primer plano, y el árbol 2 estará por detrás del árbol 1.

El archivo vectorial que hemos importado se puede editar directamente con las herramientas de Flash, ya que mantiene la información sobre trazos y formas del archivo original.

Vamos a modificar una curva del árbol 1 (el árbol que se encuentra más a la izquierda). Para facilitar la edición, ampliaremos la zona que vamos a modificar.

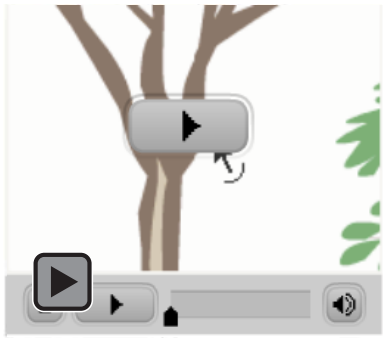
Selecciona la herramienta **Zoom** de la barra de herramientas. En la parte inferior de la barra aparecerán ahora los modificadores aumentar y reducir.

Con **Aumentar** seleccionado, acercaremos la lupa a la parte superior del tronco del árbol 1. Clicamos dos veces para ampliar suficientemente esa zona.



Una vez ampliada la zona, activaremos la **herramienta Selección** (la flecha superior de la barra de herramientas).

Con la flecha nos acercamos a la protuberancia del tronco, hasta que la flecha muestra una pequeña curva que indica que podemos hacer una modificación de la curva del árbol. Clicamos y arrastramos hasta disminuir ligeramente esa protuberancia.



Tutorial 2. Importación de archivos externos
Paso 4 de 24

Clicamos con la herramienta **Zoom > Reducir** para visualizar todo el escenario. En pasos posteriores vamos a cubrir de hierba la parte inferior del escenario y vamos a posicionar los árboles en su ubicación definitiva. Lo primero que advertimos es que vamos a necesitar duplicar la hierba. Para ello la convertiremos previamente en **símbolo**.

Los símbolos son elementos que se almacenan en la **Biblioteca** y que podemos reutilizar tantas veces como sea necesario, optimizando de esta manera el tamaño del archivo. Podemos crear tres tipo de símbolos:
-**Gráfico**. Se utilizan principalmente para imágenes estáticas. También permiten utilizar efectos de color.
-**Botón**. Para los símbolos que respondan a acciones interactivas. Un clip de película también puede responder a acciones interactivas, pero la particularidad del botón es que posee una línea de tiempo para mostrar cada estado del botón (reposo, sobre, presionado y zona activa).
-**Clip de película**. Son el tipo de símbolo más adecuado para símbolos que contienen animaciones. También permiten utilizar herramientas 3D, filtros y mezclas, y se pueden controlar mediante programación, por lo que puede resultar de utilidad aunque no contengan animaciones.

Para este paisaje vamos a crear un símbolo gráfico para la hierba y también para cada árbol, ya que, aunque no vayamos a duplicar los árboles, les daremos un efecto de color en próximos pasos.

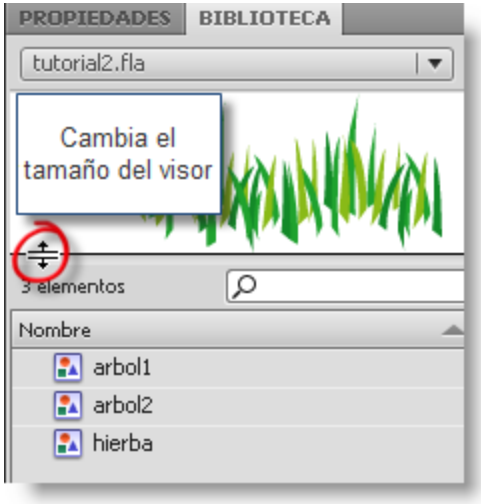
Para crear cada símbolo gráfico en primer lugar tenemos que seleccionar todos sus componentes. Una forma rápida de hacer esta selección, aprovechando el hecho de que cada objeto se encuentra en una capa diferente, es clicar sobre el fotograma que contiene cada elemento.

Por ejemplo, seleccionamos el fotograma de la capa *arbol2*. Una vez seleccionado, seleccionaremos **Modificar > Convertir en símbolo** o pulsamos directamente **F8**. En la siguiente pantalla daremos el nombre *arbol2* al símbolo, y seleccionaremos tipo **Gráfico**. Repetimos el mismo proceso para el árbol 1 y para la hierba.

Tutorial 2. Importación de archivos externos

Paso 5 de 24

Ahora tendremos almacenados en la **Biblioteca** los tres símbolos que hemos creado en el paso anterior. Si no vemos este panel, seleccionamos **Ventana > Biblioteca**. Si pulsamos en el nombre de uno de ellos podremos ver en el visor una miniatura de su contenido. Si es necesario, podemos ampliar la zona del visor.

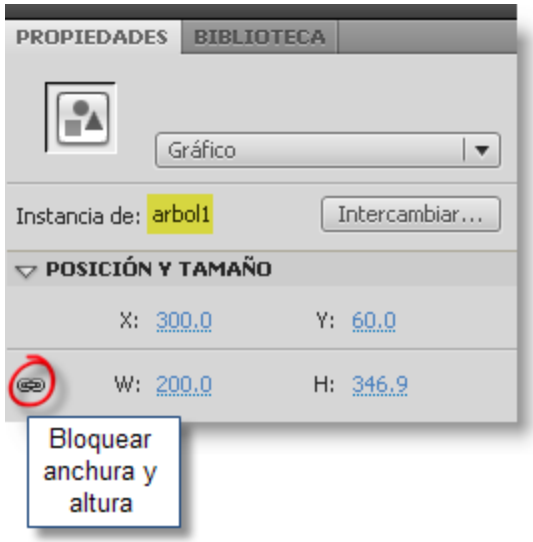


Vamos a modificar desde el inspector de **Propiedades** la posición y el tamaño del árbol 1 (el que está situado a la izquierda). Si no vemos este panel seleccionamos **Ventana > Propiedades**.

En primer lugar seleccionamos el árbol en el escenario pulsando sobre él con la herramienta **Selección** (la flecha superior).

Cuando lo hayamos seleccionado, el panel Propiedades indicará que tenemos seleccionada una instancia del símbolo gráfico *arbol1*. Las **modificaciones** se realizarán en la **instancia** del árbol que tenemos en el escenario, pero el símbolo en la biblioteca se mantendrá sin cambios.

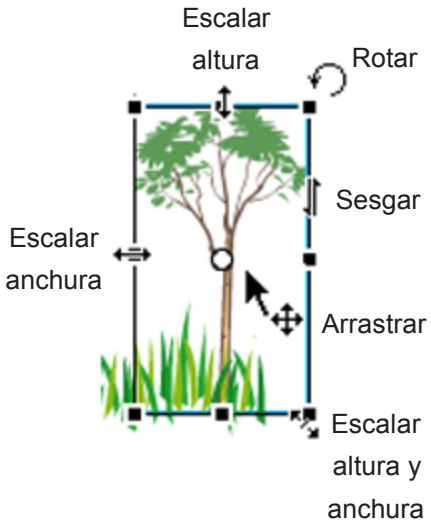
A continuación introducimos los valores que se ven en la imagen. Situamos el árbol en las coordenadas X:300px e Y:60px. Le damos una anchura (W) de 200px, y si tenemos bloqueada la anchura y la altura, el valor de la altura tomará el valor apropiado para que no se distorsione la imagen.



Tutorial 2. Importación de archivos externos
Paso 6 de 24

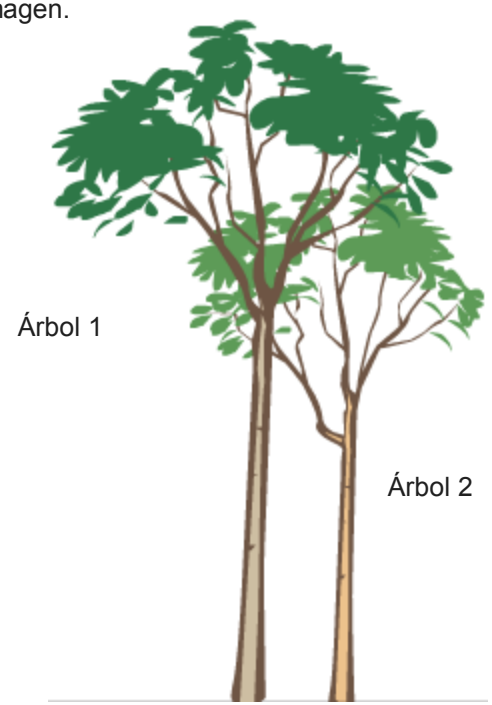
Otra forma de realizar modificaciones es con la herramienta **Transformación libre**. Seleccionamos esta herramienta y después pulsamos sobre el árbol 2 en el escenario.

Según dónde acerquemos el cursor, veremos que adquiere diferentes formas que nos indican qué tipo de transformación se aplicaría en cada caso.



Acercamos el cursor a la esquina inferior derecha del árbol para **escalar altura y anchura** conjuntamente. Mantenemos pulsada la tecla **Mayúsculas** mientras escalamos para **mantener la proporción** del árbol.

Una vez escalado, lo arrastramos hasta su posición final, de forma que quede similar a la imagen.



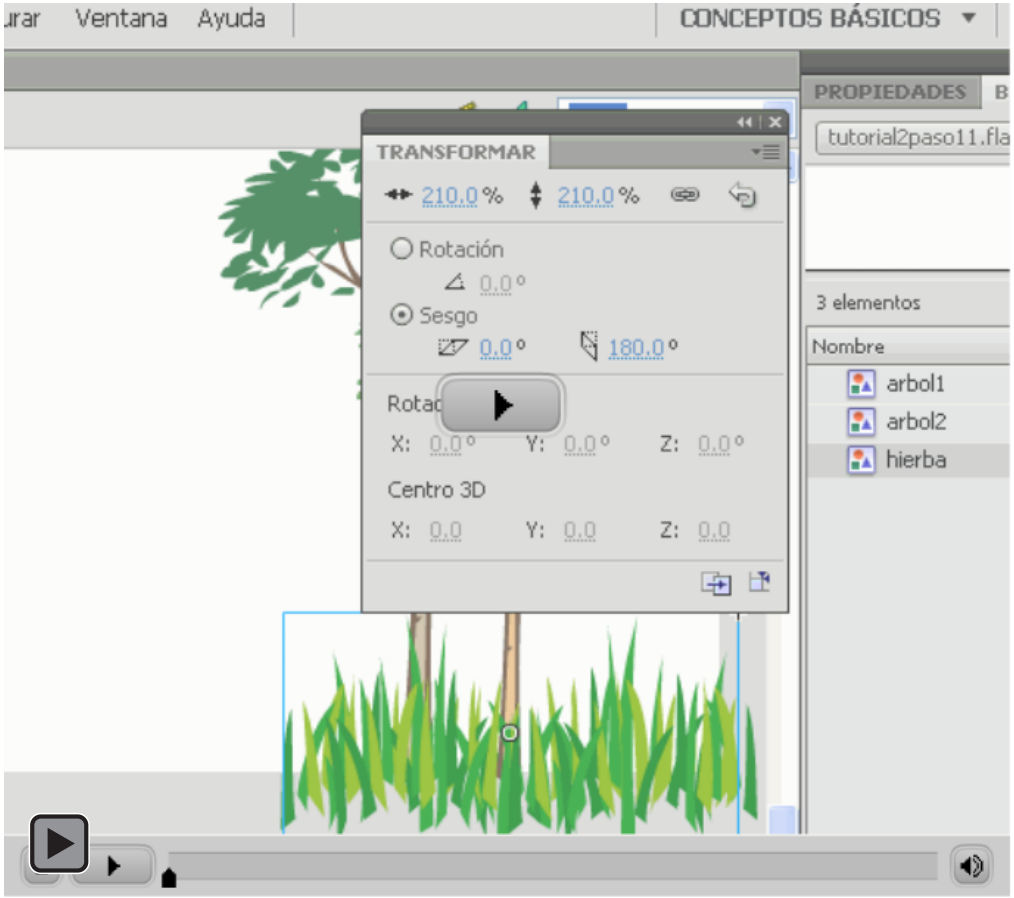
Tutorial 2. Importación de archivos externos

Paso 7 de 24

Por último, vamos a crear seis instancias del gráfico *hierba* y las vamos a modificar desde el panel Transformar, al que accederemos desde **Ventana > Transformar**.

Daremos a cada instancia valores de escalado ligeramente diferentes, y algunas de las instancias las reflejaremos horizontalmente para que las copias no se vean exactamente iguales.

Posicionaremos las instancias en la parte inferior del escenario, rebasando ligeramente los bordes del mismo.



Tutorial 2. Importación de archivos externos
Paso 8 de 24

Vamos a modificar el símbolo gráfico *hierba*, del cual hay seis instancias en el escenario. Al **modificar el símbolo original**, veremos que **los cambios se aplicarán a todas las instancias del símbolo**.

Una forma de modificar un símbolo es haciendo doble clic sobre su nombre en la **Biblioteca**. Esto nos abriría el símbolo desde su propia línea de tiempo.

Otra forma de editar un símbolo, que es la que vamos a utilizar, es haciendo **doble clic** sobre cualquiera de las **instancias** del símbolo que se encuentran en el escenario. De esta forma, podemos editar el símbolo sin perder de vista el contexto en el que se encuentra. Los elementos que no forman parte del símbolo aparecerán con colores apagados y no se podrán modificar.

Cuando estemos editando un símbolo, el nombre del símbolo aparecerá en la barra de edición, situada sobre el escenario. Desde esta misma barra podremos volver a la escena principal después de la edición del símbolo.

Hacemos doble clic sobre una instancia del gráfico *hierba* para editar su símbolo original.

Bajamos la esquina superior de las hierbas más altas. Veremos que los cambios se aplicarán al resto de las instancias.

Volvemos a la escena principal clicando sobre **Escena 1**.

Tutorial 2. Importación de archivos externos
Paso 9 de 24

Creamos una **nueva capa** a la que llamaremos *montaña*.

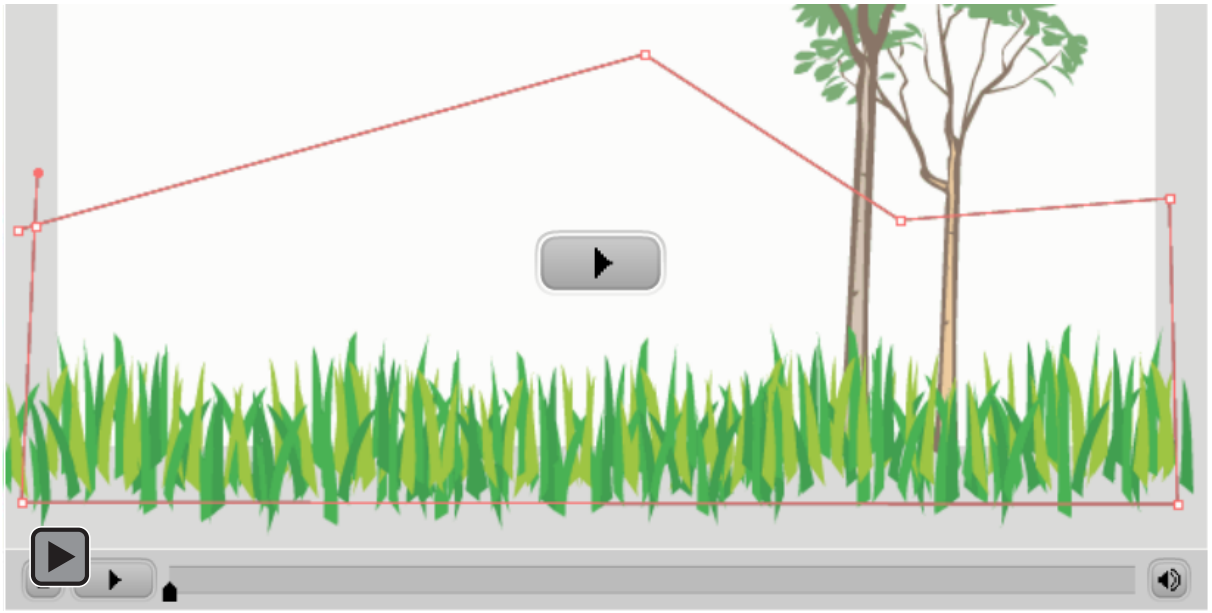
Movemos la capa hacia abajo para que quede por detrás de la hierba y de los árboles.

Bloqueamos el resto de las capas con el candado para evitar modificarlas accidentalmente.

El lápiz indica qué capa es la que se encuentra activa

Vamos a dibujar la montaña con líneas rectas. Con la herramienta **Pluma** hacemos clic para ir creando los puntos de ancla. Se irán creando segmentos rectilíneos conectados por los vértices.

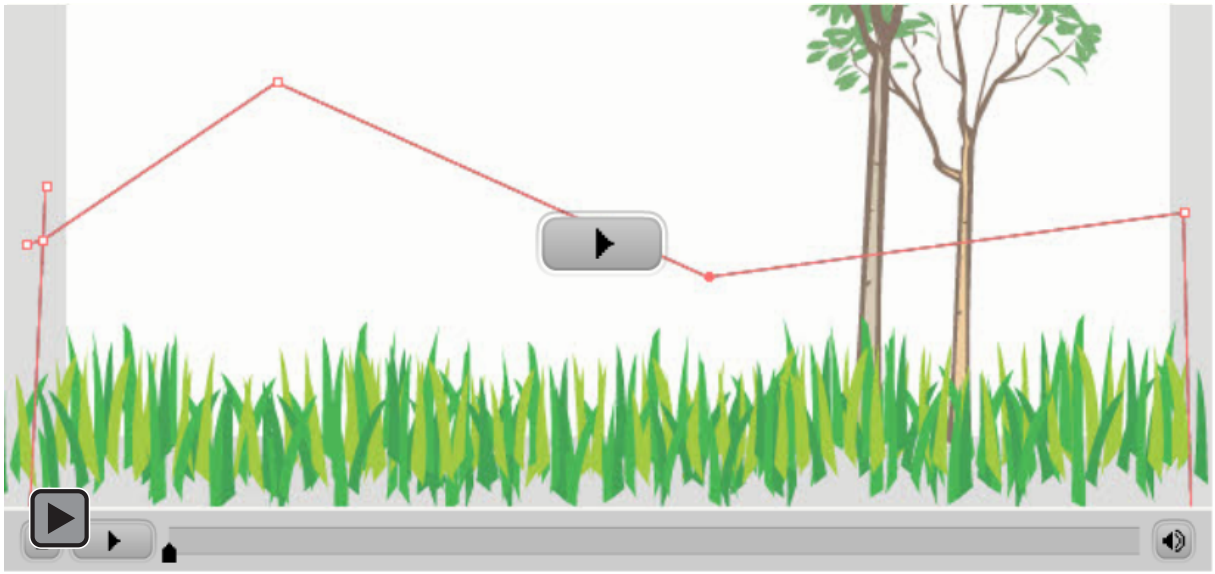
Dibujamos fuera de los límites del escenario sin preocuparnos, ya que lo que esté fuera del escenario no aparecerá en el archivo final. Para cerrar un trazado normalmente nos acercamos al vértice inicial hasta que aparezca un pequeño círculo y clicamos. En este caso, como el lugar del cierre queda fuera del escenario, cruzaremos el último segmento para asegurarnos de que la forma queda perfectamente cerrada.

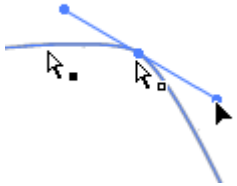


Tutorial 2. Importación de archivos externos

Paso 10 de 24

Activamos la herramienta **Subselección**. Cuando pulsamos con esta herramienta sobre un trazado, nos aparecerán los **puntos de ancla** y los **tiradores** de ese trazado. Los tiradores permiten determinar el ángulo de la curva comprendida entre dos puntos de ancla.



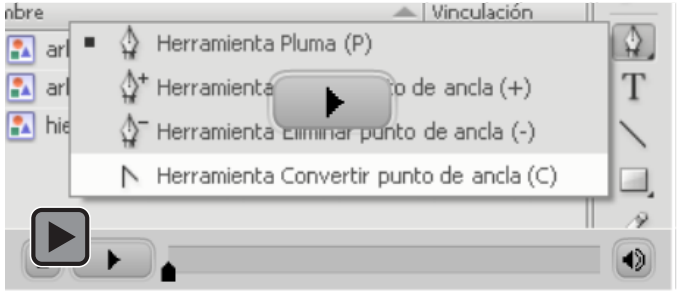


La **herramienta Subselección** adoptará diferentes formas según acerquemos el cursor a diferentes puntos del trazado. Si la flecha muestra un pequeño cuadrado blanco estaremos ante un punto de ancla que podremos desplazar. Si aparece un cuadrado negro podremos desplazar el trazado en su totalidad. Si el puntero se convierte en una flecha negra podremos modificar los tiradores para ajustar la curva.

En nuestro caso la montaña está formada, por ahora, por tramos rectos, por lo que sólo aparecen los puntos de ancla, pero no los tiradores. Con la herramienta Subselección vamos a desplazar los puntos de ancla que habíamos creado en el paso anterior, y finalmente desplazaremos ligeramente todo el trazado.

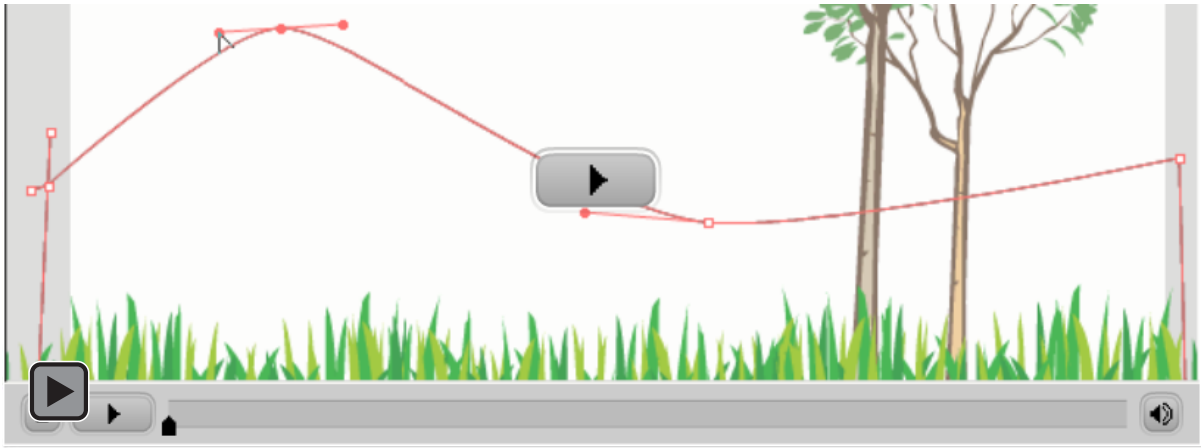
Tutorial 2. Importación de archivos externos
Paso 11 de 24

Ahora vamos a suavizar la forma de la montaña con ayuda de la herramienta **Convertir punto de ancla**. Esta herramienta se encuentra en el menú emergente de la herramienta **Pluma**.



La herramienta **Convertir punto de ancla** convierte un punto de esquina sin tiradores en un punto de línea curva con tiradores.

Para ello tenemos que pulsar y arrastrar sobre los puntos de ancla creados anteriormente, hasta conseguir el tipo de curva deseada.



Tutorial 2. Importación de archivos externos

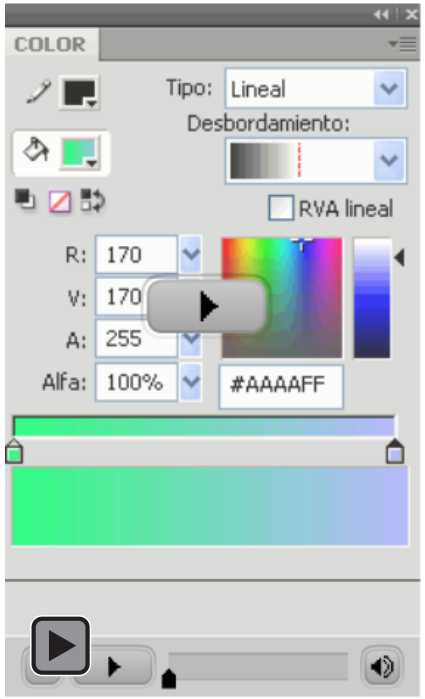
Paso 12 de 24

Vamos a rellenar la montaña con un color **degradado**. Un degradado es un relleno multicolor, en el que hay un cambio gradual entre varios colores.

Abrimos el panel Color seleccionando **Ventana > Color**. Recordemos que podemos arrastrar el panel pulsando sobre la barra gris oscuro superior del panel.

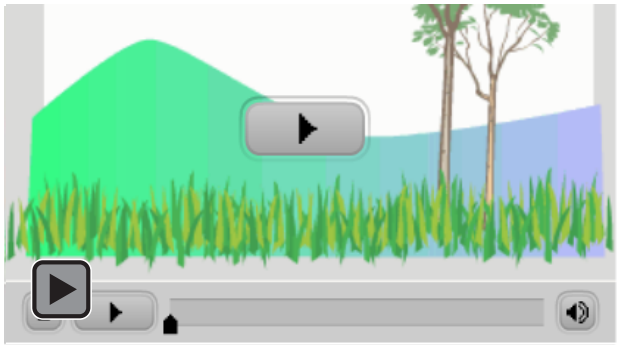
Seleccionamos un color de relleno **Tipo: Lineal**. En la barra inferior aparecerán los colores extremos del degradado.

Clicamos dos veces sobre uno de los extremos para abrir la paleta de muestras, y seleccionamos el color #00FF66. El otro extremo lo vamos a cambiar dando directamente nuevos valores RVA (rojo, verde, azul) directamente en el panel: R:170, V:170 y A:255.



Activamos la herramienta **Cubo de pintura**. Clicamos sobre el interior de la montaña para rellenarla con el degradado que acabamos de crear.

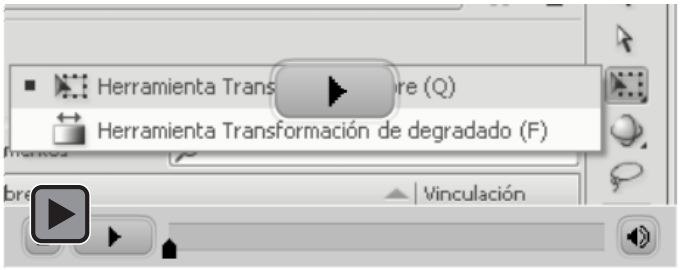
Vamos a eliminar el trazado de la montaña y a dejar sólo el relleno. Para ello activamos la herramienta **Selección** desde la barra de herramientas o pulsando la letra **V**. Hacemos doble clic sobre el trazado de la montaña para seleccionarlo por completo, y pulsamos **Supr** para eliminar el trazado.

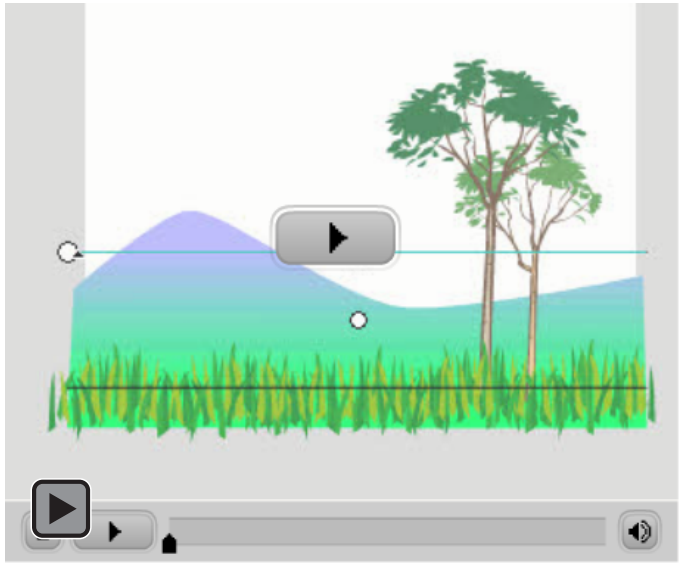


Tutorial 2. Importación de archivos externos
Paso 13 de 24

Ahora utilizaremos la herramienta **Transformación de degradado** para reorientar el degradado de la montaña, de tal forma que el color azulado quede en la parte superior y el verde en la inferior.

Esta herramienta se encuentra agrupada junto con la herramienta de **Transformación libre**, por lo que debemos mantener pulsada la herramienta de **Transformación libre**, y seleccionar **Transformación de degradado** en el menú emergente.



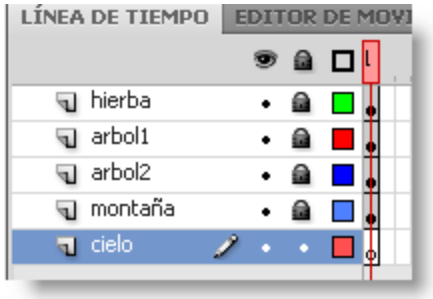


Una vez seleccionada esta herramienta, pulsaremos sobre el relleno degradado de la montaña. El primer paso será rotar el degradado, de tal forma que el color más azulado quede en la parte superior.

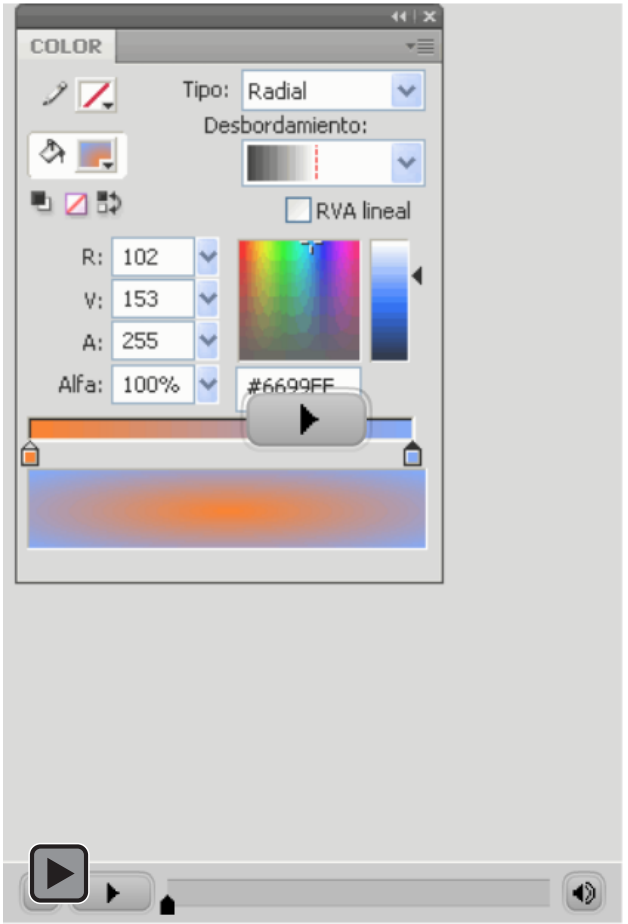
Por último estrecharemos el área en la que se aplica el degradado, de tal forma que las dos líneas paralelas de delimitación queden dentro de la montaña. Estas líneas son las que limitan el área en la que se produce la transición entre un color y otro. La parte del relleno que queda fuera de esas dos líneas paralelas tendrán el color puro del extremo que les corresponda.

Tutorial 2. Importación de archivos externos

Paso 14 de 24



Creamos una nueva capa a la que llamaremos *cielo* de la misma forma que creamos la capa para la montaña en el paso 9 de este tutorial. Arrastramos la capa a la parte inferior, ya que debe quedar por detrás de todos los demás elementos del paisaje. Bloqueamos el resto de las capas y la seleccionamos para que sea la capa activa.

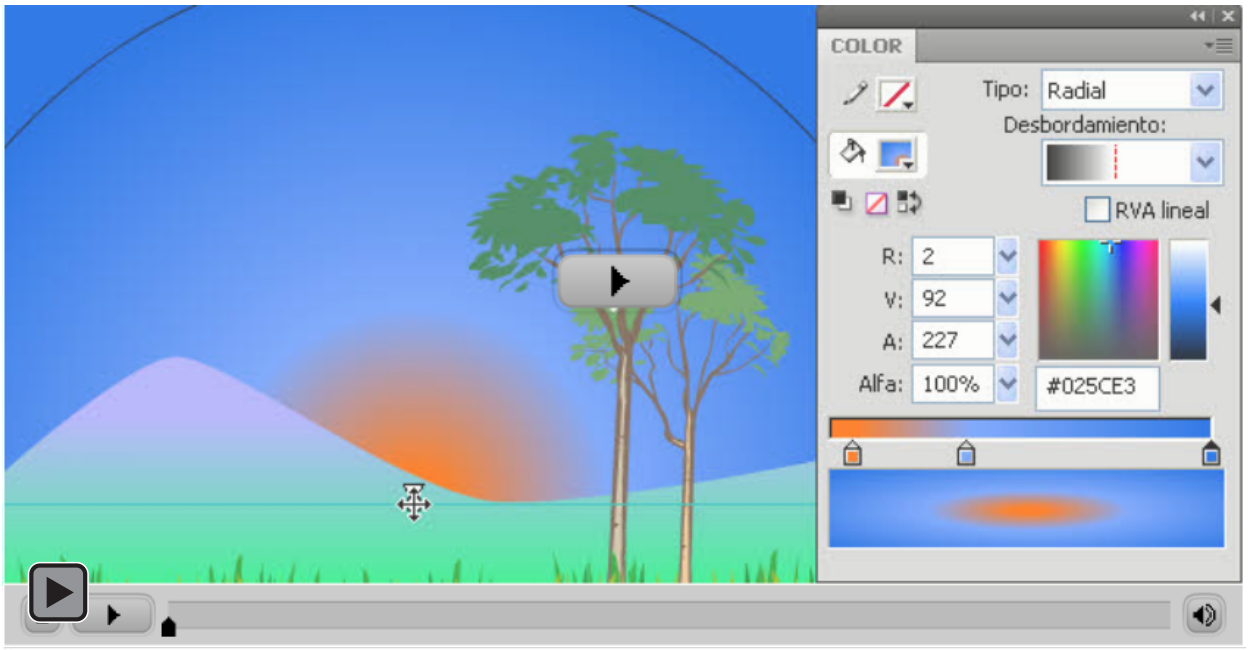


Para el cielo dibujaremos un rectángulo sin línea de trazo (sin perfil) y con un relleno degradado, aunque en este caso será un degradado radial. Para ello utilizaremos de nuevo el panel Color (**Ventana > Color**). Para el trazo seleccionaremos **Tipo: Ninguno** o bien la franja roja que indica que no hay color. Para el relleno seleccionamos **Tipo: Radial**, y damos a un extremo el valor #FF6600 y al otro extremos el valor #6699FF.

Activamos la herramienta **Rectángulo** y trazaremos un rectángulo que será el cielo, traspasando de nuevo los límites del escenario por arriba y por ambos lados, y de tal forma que la parte inferior quede por detrás de las montañas, sin ver su límite.

Tutorial 2. Importación de archivos externos

Paso 15 de 24

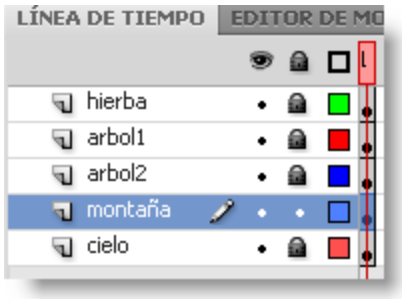


Vamos a hacer cambios en el degradado del cielo. Con la herramienta **Selección (V)** seleccionamos el cielo para poder ir visualizando los cambios a medida que los hagamos. Desplazamos el cursor azul hacia el naranja para hacer más pequeño el círculo que representará al sol. También desplazamos ligeramente el color naranja hacia el centro. Clicamos en la barra de degradado para crear un nuevo color. Nos aparecerá el mismo azul que teníamos, y lo oscureceremos ligeramente. Por último, con la herramienta **Transformación de degradado** desplazaremos el centro del degradado hasta situarlo tras la montaña.

Tutorial 2. Importación de archivos externos
Paso 16 de 24

En contraste con el cielo, vemos que el resto de los elementos del paisaje tienen colores demasiado luminosos, por lo que vamos a oscurecerlos.

Primero, al igual que hicimos en el paso 4 de este tutorial, vamos a convertir a la montaña en un símbolo gráfico. Desbloqueamos la capa montaña y bloqueamos el resto de las capas. Seleccionamos la montaña en el escenario o bien el fotograma en la línea de tiempo. Pulsamos **F8** o **Modificar > Convertir en símbolo**, seleccionamos **Tipo: Gráfico** y le damos el nombre *montaña*.

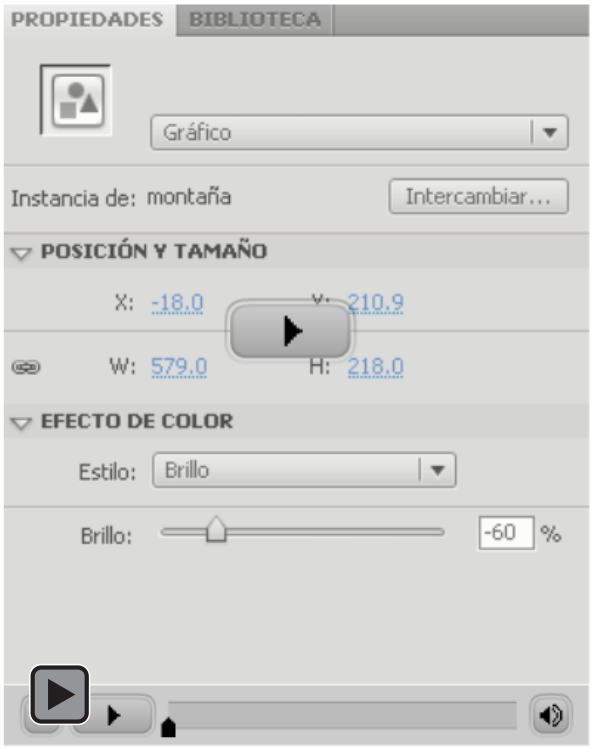


Con la instancia de la montaña seleccionada en el escenario, le asignamos en el inspector de **Propiedades** un **efecto de color** de estilo **Brillo**, con un valor de -60. De esta forma oscureceremos el símbolo sin necesidad de variar los colores originales. Podemos probar varios valores de brillo hasta que encontremos un color que nos guste.

Repetimos el mismo proceso con los árboles y con la hierba. Para ello tendremos que ir desbloqueando cada capa, seleccionando todas las instancias, y asignándoles en efectos de color el mismo brillo negativo.

También podemos desbloquear al mismo tiempo las capas de los árboles y de la hierba, y abarcar todos los elementos con la herramienta Selección, y después aplicar el efecto de Brillo, que afectaría a todos los elementos seleccionados.

Esto no funcionaría si tenemos también seleccionado el cielo, ya que al no ser un símbolo no admite este efecto. Por ello, es importante mantener bloqueada la capa que contiene el cielo.

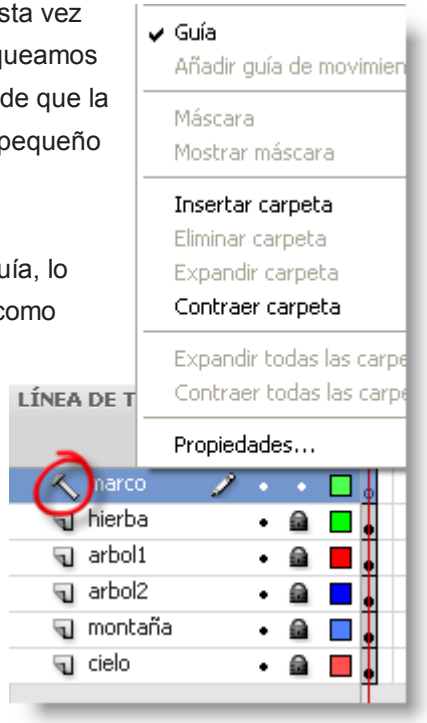


Tutorial 2. Importación de archivos externos

Paso 17 de 24

Antes de añadir la luna, que será el último elemento del paisaje, vamos a hacer un marco de referencia para saber dónde están los límites del escenario, ya que ahora aparecen cubiertos por el cielo y las montañas.

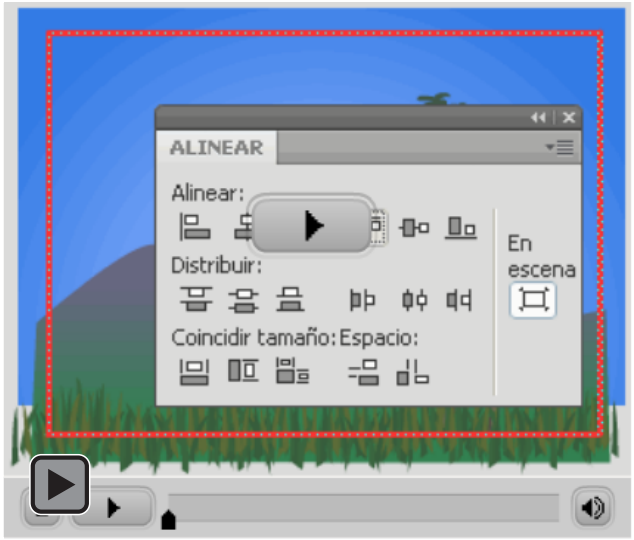
Al igual que como hemos visto en otros pasos, creamos una nueva capa llamada marco, y la situamos esta vez por encima de todas las demás. Bloqueamos el resto de capas y nos aseguramos de que la capa marco está activa (muestra un pequeño lápiz).



Después convertimos esa capa en guía, lo que significa que nos puede ayudar como guía visual, pero no aparecerá en el Archivo que se publique. Para ello clicamos con el botón derecho del ratón sobre el nombre de la capa, y seleccionamos **Guía**. A la izquierda del nombre de la capa aparecerá un pequeño icono para indicar que esa capa ahora es una guía.

Seleccionamos la herramienta **Rectángulo**, trazo rojo y sin color de relleno, y trazamos un rectángulo en el escenario, sin preocuparnos de su tamaño ni ubicación. Una vez dibujado, lo seleccionamos y le damos en el inspector de **Propiedades** los siguientes valores: *X:0 - Y:0* para que se posicione en el extremo superior izquierdo del escenario, y *W:550 - H:400*, que son las medidas del escenario.

También podría llegarse al mismo resultado con el panel **Alinear (Ventana > Alinear)**, alineándolo a la escena tal y como se ve en este vídeo.

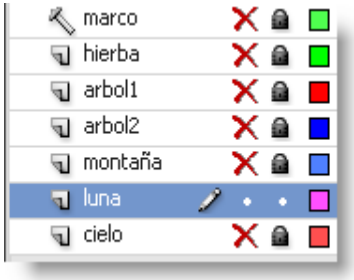


Tutorial 2. Importación de archivos externos

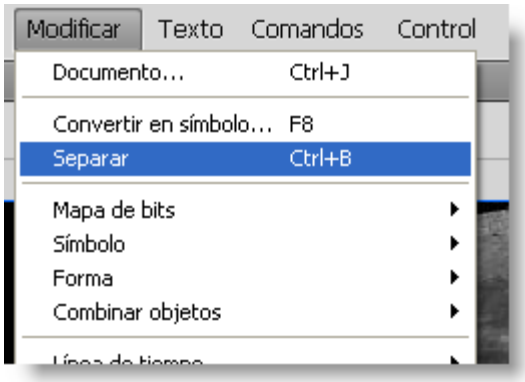
Paso 18 de 24

Creamos una nueva capa llamada luna, que situaremos entre la montaña y el cielo.

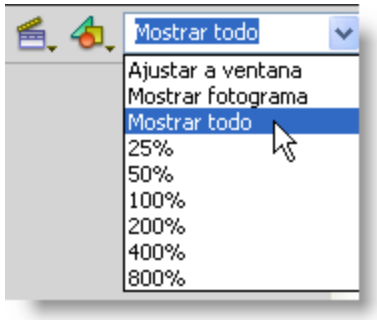
Importamos el archivo de mapa de bits *luna_llena.jpg* que se encuentra en la carpeta *tutorial 2*. Para ello seleccionamos **Archivo > Importar > Importar a escenario**.



Ocultamos todas las capas excepto la de la luna marcando los puntos que se encuentran al lado de los candados.



Con la imagen de la luna seleccionada, elegimos **Modificar > Separar**, o bien pulsamos sobre ella con el botón derecho del ratón y marcamos Separar en el menú contextual.



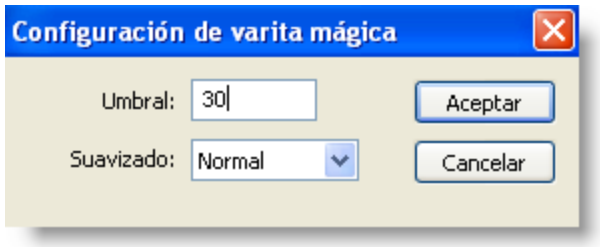
Seleccionamos **Mostrar todo** en la barra de edición, para mostrar la luna completamente y ocupando toda el área disponible.

Pulsamos en cualquier lugar del área del escenario pero fuera de la imagen para deseleccionarla.

Tutorial 2. Importación de archivos externos
Paso 19 de 24

Cuando separamos un mapa de bits, los píxeles de la imagen se convierten en áreas que se pueden seleccionar y modificar por separado. Por lo tanto, ahora podemos editar la imagen de la luna.

Seleccionamos la herramienta **Lazo**, y en la parte inferior de la barra de herramientas aparecerán los modificadores asociados a esta herramienta. Seleccionamos en primer lugar **Propiedades de varita mágica**.



El **umbral** se refiere a la diferencia de color entre píxeles adyacentes, y el **suavizado** se refiere al nivel de detalle de los bordes de la selección. Dejamos un valor de umbral de 30 y suavizado normal y pulsamos Aceptar.

Seleccionamos la herramienta **Varita mágica**, y con ella vamos marcando las cuatro grandes áreas negras de la imagen. Después de cada selección, pulsamos **Supr** para borrar esas áreas.

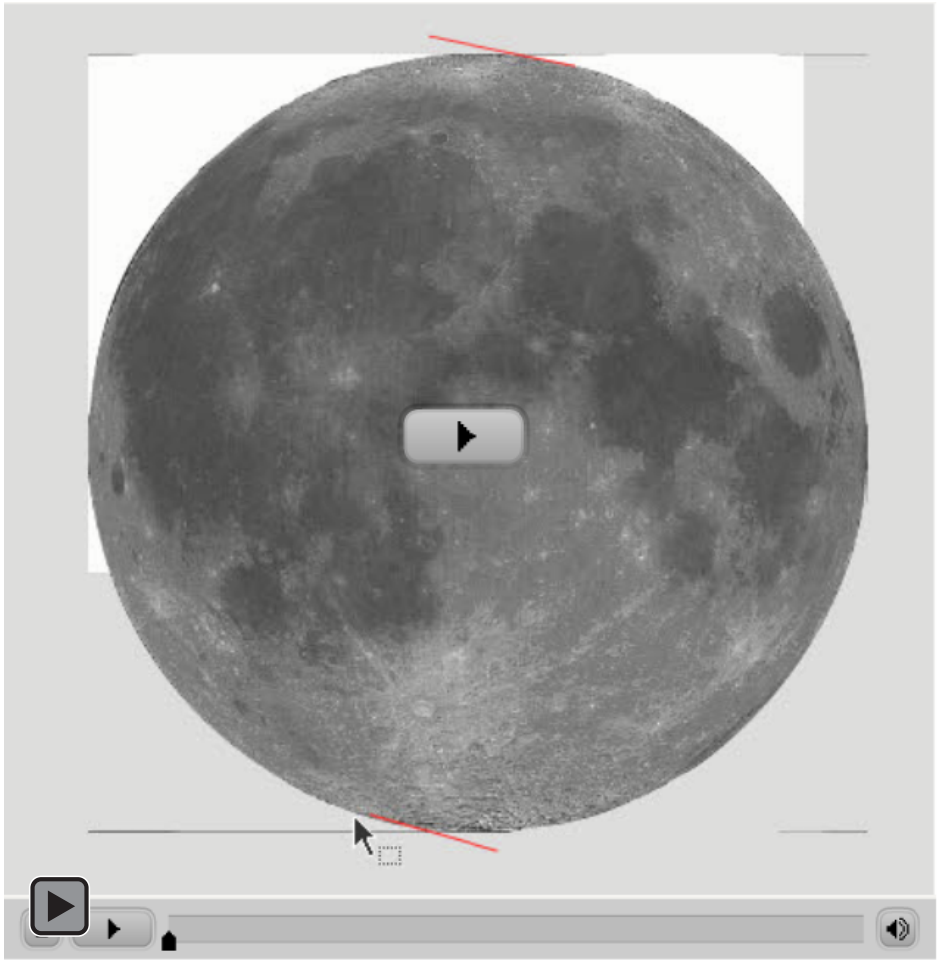
Tutorial 2. Importación de archivos externos
Paso 20 de 24

Aunque ya casi tenemos la luna sin el fondo, vemos que nos han quedado pequeñas líneas que tenemos que eliminar. Para ello vamos a dividir la imagen con la ayuda de la herramienta **Línea**.

Dibujamos dos líneas de corte para separar las pequeñas franjas negras que han quedado junto a la luna.

Después, con la herramienta **Selección**, vamos marcando todas las áreas externas a la luna y las borramos con la tecla **Supr**.

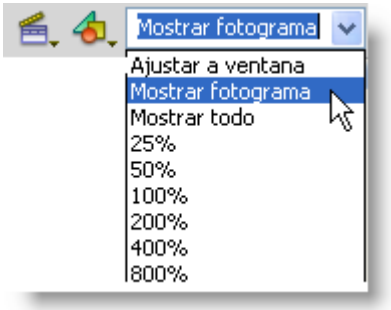
Por último, borramos también las propias líneas que hemos utilizado para efectuar los cortes en la imagen. Para seleccionar completamente un trazo que está dividido en segmentos, la forma más rápida es haciendo doble clic sobre el trazo. En este caso, las líneas rojas se habían dividido en segmentos al entrar en contacto con parte del dibujo de la luna, por ello hacemos doble clic para seleccionarlas.



Tutorial 2. Importación de archivos externos
Paso 21 de 24

Ahora que ya tenemos la luna totalmente separada del fondo, la seleccionamos y en el panel **Propiedades**, en posición y tamaño, le damos los valores X: 80, Y: 60, W: 60 y H:60.

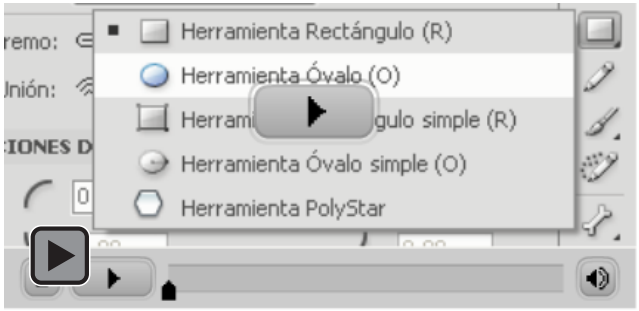
Hacemos visibles las capas que teníamos ocultas.

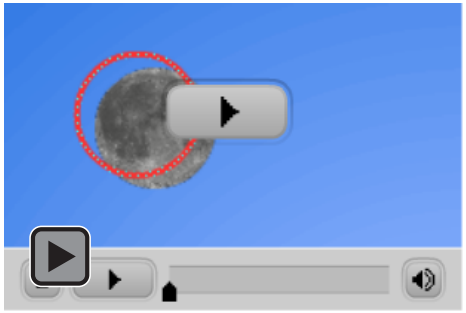


Cambiamos el grado de aumento del escenario seleccionando **Mostrar fotograma** en la barra de edición.

Ahora tendremos un paisaje similar al de esta imagen. En los próximos pasos modificaremos la luna llena dejando sólo una pequeña porción de luna creciente, ya que es el aspecto que muestra cuando se ve cerca del horizonte al atardecer. Por último le daremos un toque de luz utilizando un efecto de color avanzado.

Tutorial 2. Importación de archivos externos
Paso 22 de 24





En la misma capa que la luna, dibujaremos cerca de ella un pequeño círculo con la herramienta **Óvalo**, con trazo rojo y sin relleno. Para que la forma sea circular y no ovalada, mantendremos pulsada la tecla Mayúsculas mientras lo dibujamos.

Seleccionamos este círculo, y en el inspector de Propiedades le damos un valor de altura y anchura de 60, que es el mismo tamaño que tiene la luna.

Arrastramos el círculo hasta cruzarlo con la imagen de la luna, de tal forma que a la derecha de la imagen quede el pequeño gajo que queremos mostrar.

Una vez posicionado el círculo, seleccionamos y borramos con **Supr** la parte de la luna que queremos eliminar.

Por último eliminamos el propio círculo que hemos utilizado para cortar la imagen.

Tutorial 2. Importación de archivos externos

Paso 23 de 24

Por último aplicaremos efectos de color a la luna, con lo que, como hemos hecho en pasos anteriores, convertimos la imagen de la luna en un **símbolo gráfico**.

Para iluminar la imagen, seleccionamos en el inspector de **Propiedades** un efecto de color **Estilo: Avanzadas**.

Asignamos un valor de desplazamiento del rojo de 150, desplazamiento del verde de 150, y 75 en el desplazamiento del azul.

¡No olvidéis guardar vuestro trabajo!

Tutorial 2. Importación de archivos externos

Paso 24 de 24

Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

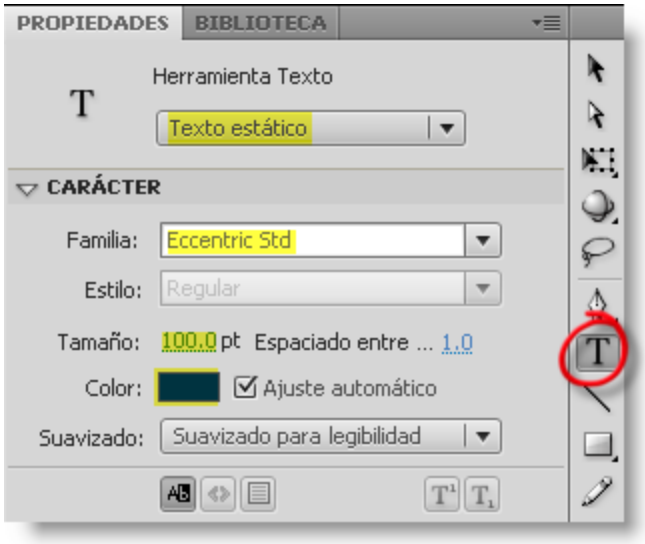
1. Colocad otra instancia de uno de los árboles en el escenario, y cambiad su tamaño y posición.
2. Modificad la forma de la montaña con la herramienta Subselección.
3. Experimentad con diferentes degradados para el cielo y para la montaña, así como con el efecto de Brillo, para simular diferentes horas del día.

Tutorial 3. Creación de un logotipo textual

Paso 1 de 13

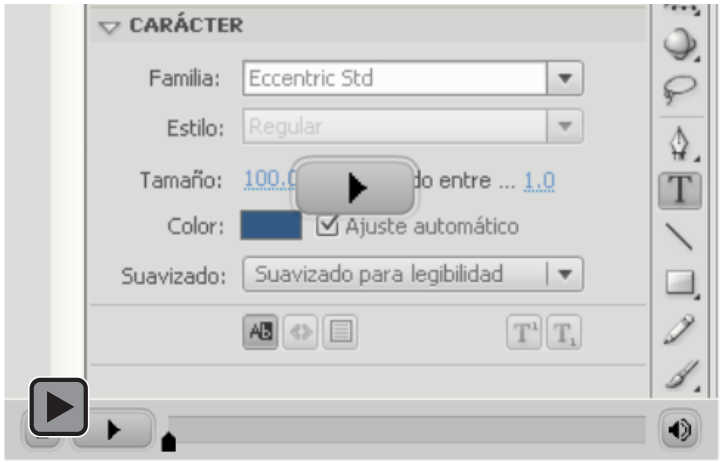
En este tutorial vamos a crear un logotipo basado en un texto al que aplicaremos algunos efectos utilizando filtros y mezclas.

Para comenzar seleccionaremos la **herramienta Texto**. El **inspector de Propiedades** mostrará los diferentes atributos de esta herramienta. Modificaremos algunos atributos del bloque **Carácter**.



En el primer desplegable que aparece en el inspector de Propiedades seleccionamos **Texto estático**. Los otros tipos de texto (texto dinámico e introducción de texto) los veremos en los tutoriales de programación.

En el campo **Familia** seleccionaremos **Eccentric Std**. Para ello podemos buscar la fuente en el desplegable o escribir directamente las primeras letras de su nombre. Si no disponemos de esta fuente podemos elegir cualquier otra. Después le asignaremos un **tamaño** de **100 pt** y un **color** azul oscuro (**#003366**).



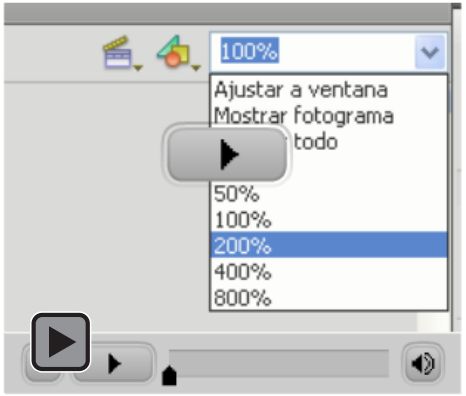
Tutorial 3. Creación de un logotipo textual
Paso 2 de 13

Con la **herramienta Texto** configurada, clicamos con ella en el escenario, y tecleamos la palabra *logo*.

La palabra se mostrará con los atributos que hemos seleccionado en el inspector de Propiedades.

También sería posible cambiar estos atributos después de haber creado el texto. Para ello seleccionamos el campo de texto que se encuentra en el escenario y realizamos las modificaciones que queramos en el inspector de Propiedades.

Para poder editar el logo con mayor comodidad, acercaremos el escenario, seleccionando un **zoom** del **200%**.



Si el escenario no ha quedado bien posicionado para trabajar después de hacer el zoom, podemos desplazarlo con la **herramienta Mano**.

Tutorial 3. Creación de un logotipo textual

Paso 3 de 13

El siguiente paso va a ser modificar la letra *g*. Para ello debemos convertir previamente el texto en un gráfico editable.

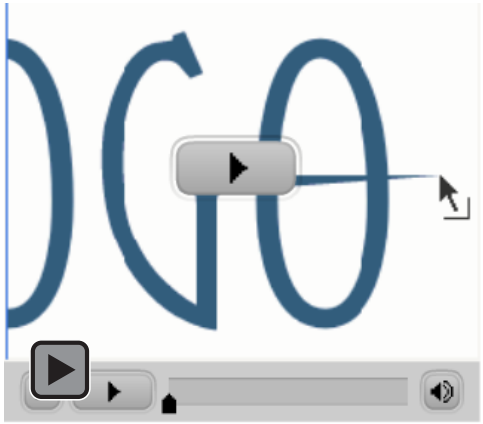
Seleccionamos el campo de texto en el escenario y seleccionamos **Modificar > Separar**, o bien clicamos sobre el texto con el botón derecho del ratón y seleccionamos **Separar** en el menú contextual.

Cuando separamos un texto se crea un campo de texto por cada carácter. En este punto podríamos por ejemplo cambiar una letra por otra, elegir una fuente diferente para alguna letra individual, posicionar cada letra en diferentes lugares, animar caracteres individuales, etc. Cada carácter se encontrará por tanto en un campo de texto independiente.

Si lo que queremos es manipular la forma de una letra, deberemos volver a separar ese campo de texto. En este caso, como sólo vamos a manipular la forma de la letra *g*, sólo separaremos de nuevo esta letra, manteniéndose el resto como caracteres individuales. Para ello seleccionamos la letra *g* y después **Modificar > Separar**.

Tutorial 3. Creación de un logotipo textual
Paso 4 de 13

Vamos a cambiar un pequeño detalle de la letra *g*. Con la **herramienta Selección** arrastramos una esquina para modificar su posición, tal y como se muestra en el vídeo.



El paso siguiente va a ser duplicar los campos de texto que contienen la letra *o*. Para que la distancia entre las letras duplicadas sea la misma que entre las letras originales, copiaremos y pegaremos ambos campos al mismo tiempo.

Para ello seleccionamos uno de ellos, y con la tecla **Mayúsculas** pulsada, seleccionamos el otro. Hacemos clic con el **botón derecho del ratón** y seleccionamos **Copiar** del menú contextual. Después, en cualquier punto del escenario, hacemos clic de nuevo con el botón derecho del ratón y seleccionamos **Pegar**.

Ahora tendremos seleccionados en el escenario los dos campos que hemos pegado.

En el **inspector de Propiedades** asignamos un **tamaño** de **50 pt** a estos campos de texto. Esto no hubiera sido posible con la letra *g*, ya que al separarla de nuevo ya no es un campo de texto.

Por último, con ambos campos aún seleccionados, arrastramos estas copias hasta situarlas dentro de las letras *u* originales. Para afinar mejor la posición podemos ayudarnos de las flechas del teclado.

Tutorial 3. Creación de un logotipo textual
Paso 5 de 13

Ahora que ya hemos realizado la parte superior del logo, nos queda hacer el reflejo de la parte inferior. Para crear este reflejo utilizaremos una copia del logo que ya hemos creado, y a esta copia le añadiremos efectos especiales utilizando **filtros** y **mezclas**.

Para poder añadir estos efectos necesitamos crear un **clip de película**. Utilizar un clip de película tendrá también la ventaja de que si decidimos cambiar el logo original, su reflejo también se verá modificado, ya que será una instancia del mismo clip de película.

Seleccionamos el logo abarcándolo con la **herramienta Selección** o bien clicando sobre el fotograma que lo contiene en la línea de tiempo.

Después seleccionamos **F8** o **Modificar > Convertir en símbolo**, y le damos el nombre *logo*. Como Tipo seleccionamos **Clip de película**.

Damos el nombre de *superior* a la capa actual y creamos una **nueva capa** a la que llamamos *reflejo*. Situamos esta nueva capa por debajo de la capa *superior* y la seleccionamos para que sea la capa activa.

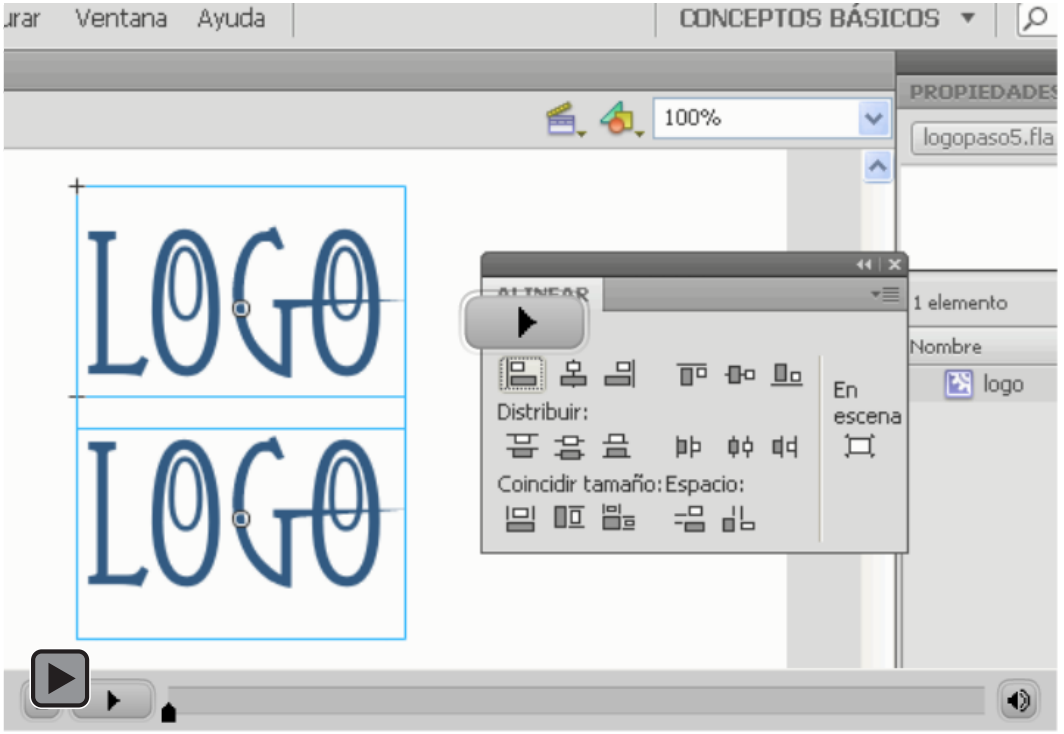
Tutorial 3. Creación de un logotipo textual
Paso 6 de 13

Con la capa *reflejo* activa, arrastramos desde la biblioteca al escenario una **instancia** del clip de película *logo*.

Seleccionamos **Ventana > Alinear**. Nos aseguramos de que el botón **En escena** no esté seleccionado.

Seleccionamos la instancia del logo que se encuentra en la capa *superior*. Recordemos que para hacer una selección múltiple tenemos que tener la tecla **Mayúsculas** pulsada, y las capas donde se encuentran los elementos que vamos a seleccionar no deben estar bloqueadas.

En el panel **Alinear** seleccionamos **Alinear borde izquierdo**.



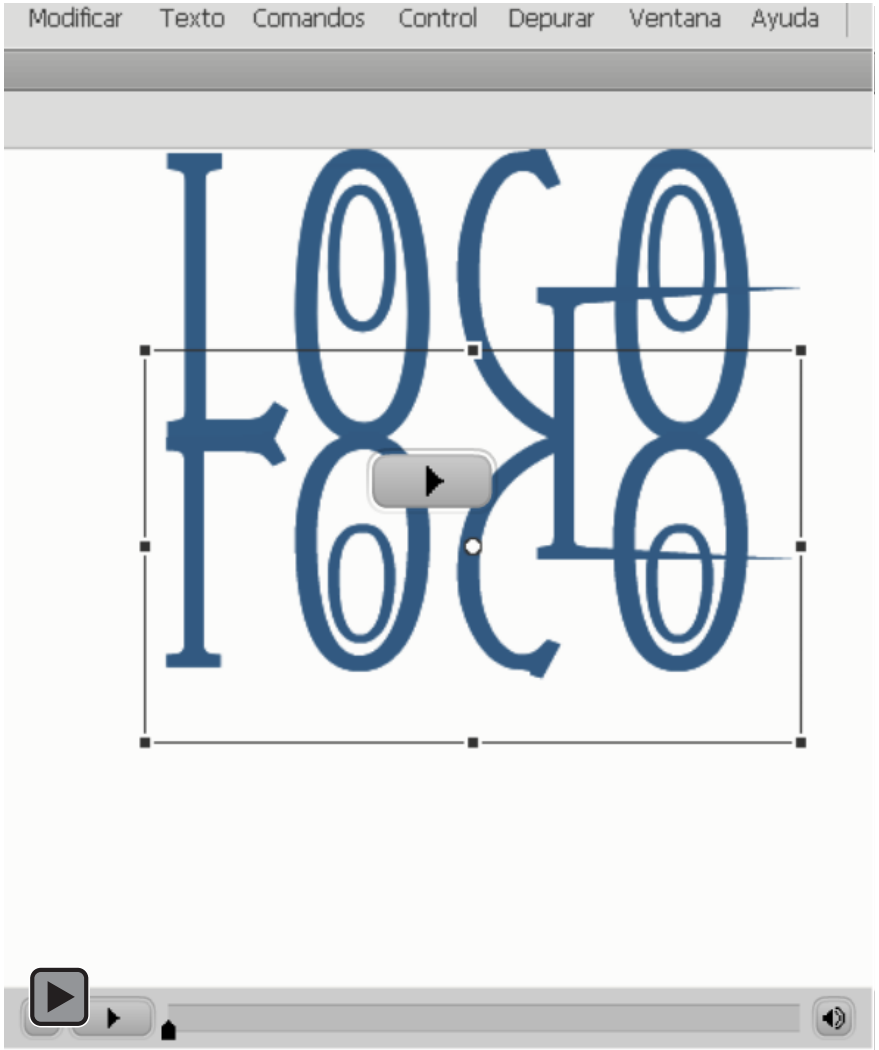
Tutorial 3. Creación de un logotipo textual
Paso 7 de 13

Seleccionamos la instancia del logo correspondiente al reflejo con la **herramienta Transformación libre** y reducimos ligeramente su altura.

Seleccionamos **Modificar > Transformar > Voltear verticalmente**.

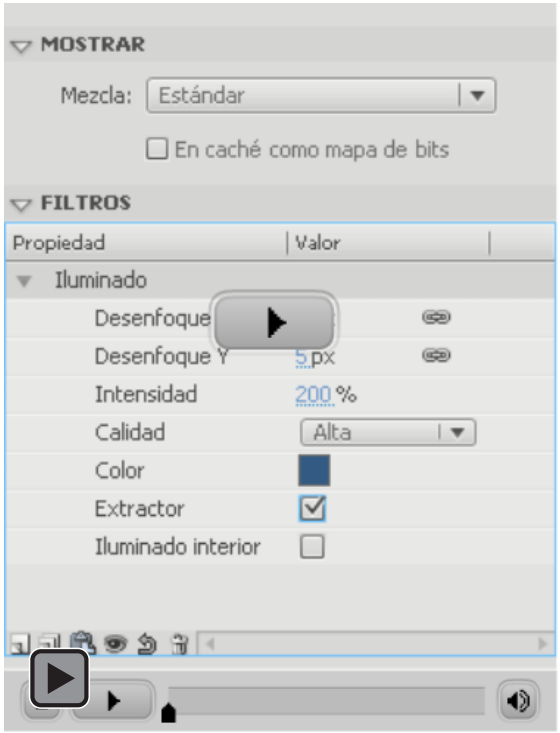
Ajustamos la posición vertical del reflejo con la ayuda de **las flechas arriba y abajo del teclado**. Podemos mantener una flecha pulsada para avanzar rápidamente, y después podemos ajustar la posición con pequeñas pulsaciones de la flecha.

Subimos la instancia del reflejo hasta eliminar el espacio entre la base de la letra *l* y su reflejo.



Tutorial 3. Creación de un logotipo textual

Paso 8 de 13



Vamos a añadir un **filtro** a la instancia del reflejo. Los filtros se pueden añadir a campos de texto, botones y clips de película. Si el texto no lo hubiéramos separado, podríamos también haber añadido un filtro sin necesidad de convertirlo previamente en clip de película.

Seleccionamos la instancia del reflejo en el escenario,

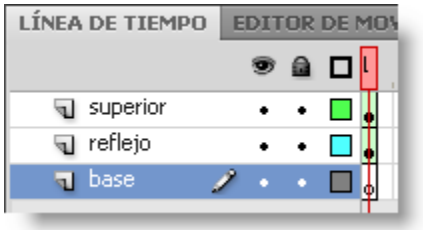
Seleccionamos **Añadir filtro** en el área de **Filtros** del **inspector de Propiedades**. Se abrirá un menú desplegable que mostrará los filtros disponibles. Vamos a añadir un filtro de tipo **Iluminado**.

Vamos a efectuar algunos cambios en los atributos de este filtro. Para comenzar vamos a asignar una **intensidad** del **200%**. Seleccionamos **calidad alta**. Modificamos el **color** seleccionando el mismo color que tenía el logo (**#003366**). Por último, marcamos la casilla **Extractor** para mostrar solamente el iluminado.

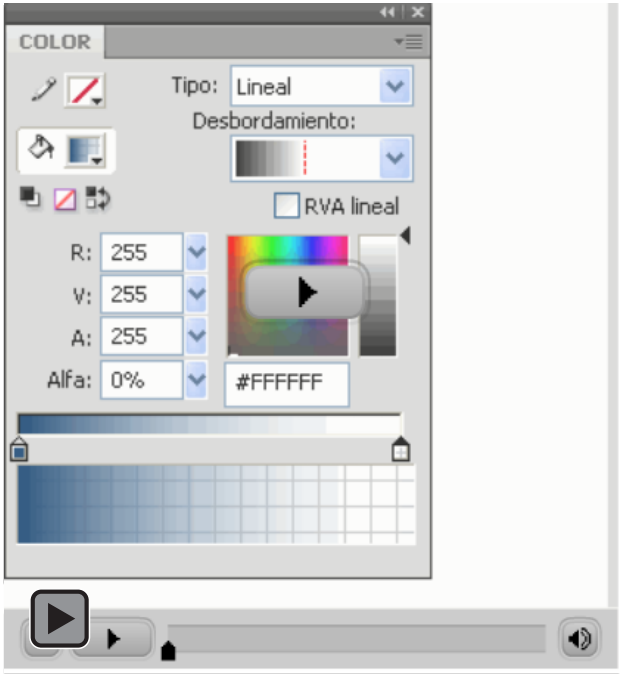
Tutorial 3. Creación de un logotipo textual

Paso 9 de 13

Para dibujar el rectángulo donde parece apoyarse el logo creamos una **nueva capa**, a la que llamaremos *base*, y la situamos por debajo de las otras dos capas.



Abrimos el **panel Color** (**Ventana > Color**). **Eliminamos el trazo** y para el **relleno** seleccionamos un **degradado lineal** con un extremo con el color azul que estamos utilizando en este tutorial (**#003366**) y con otro extremo transparente (**alfa 0**). De esta forma el color azul se irá transparentando progresivamente.



Tutorial 3. Creación de un logotipo textual

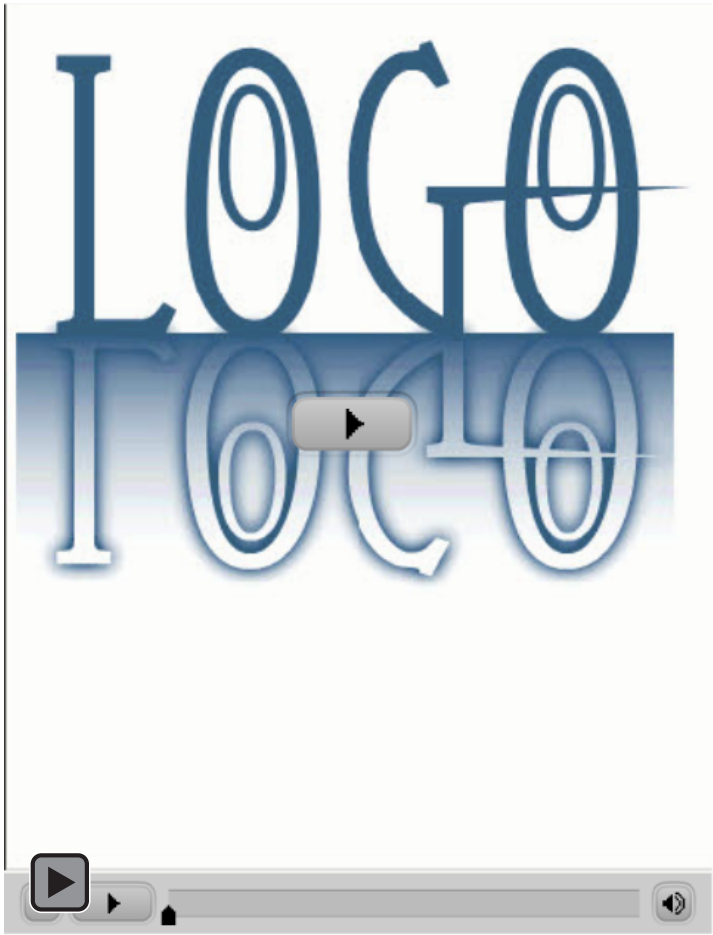
Paso 10 de 13

Dibujamos en la capa base un pequeño rectángulo con **la herramienta Rectángulo**, que aparecerá con las opciones de color que habíamos seleccionado en el paso anterior.

Con la **herramienta Transformación de degradado** modificamos el degradado lineal, de tal forma que el color azul quede arriba y el color transparente abajo. También cambiaremos la altura del degradado para que coincida con la del rectángulo.

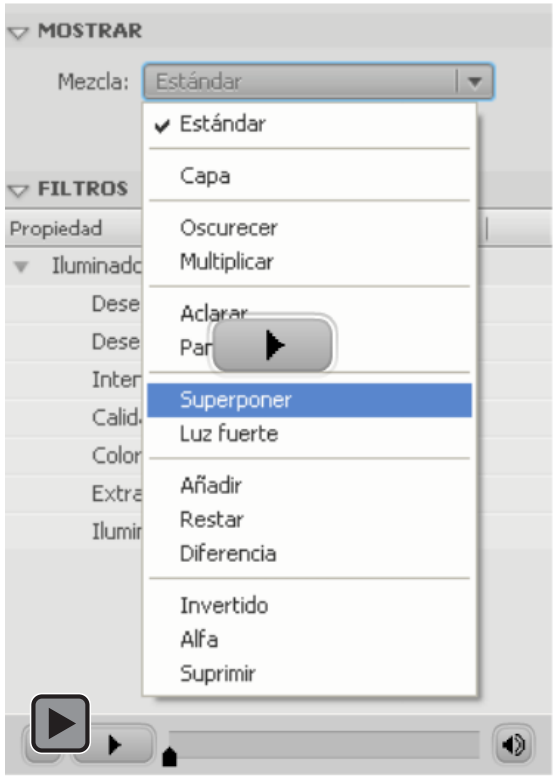
Después, con la **herramienta Transformación libre** situamos el rectángulo y le damos las proporciones adecuadas.

Recordad que ambas herramientas se encuentran agrupadas en el mismo lugar de la barra de herramientas. Si tenéis alguna duda sobre su uso repasad el tutorial anterior.



Tutorial 3. Creación de un logotipo textual

Paso 11 de 13



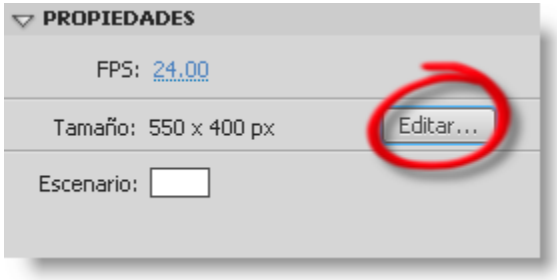
Para finalizar el logo, vamos a crear una imagen compuesta entre el reflejo y el rectángulo de la base.

Seleccionamos en el escenario la imagen reflejada del logo. Podemos bloquear las otras capas para asegurarnos de que estamos seleccionando el reflejo.

En el **inspector de Propiedades**, en el área **Mostrar**, seleccionamos **Mezcla > Superponer**.

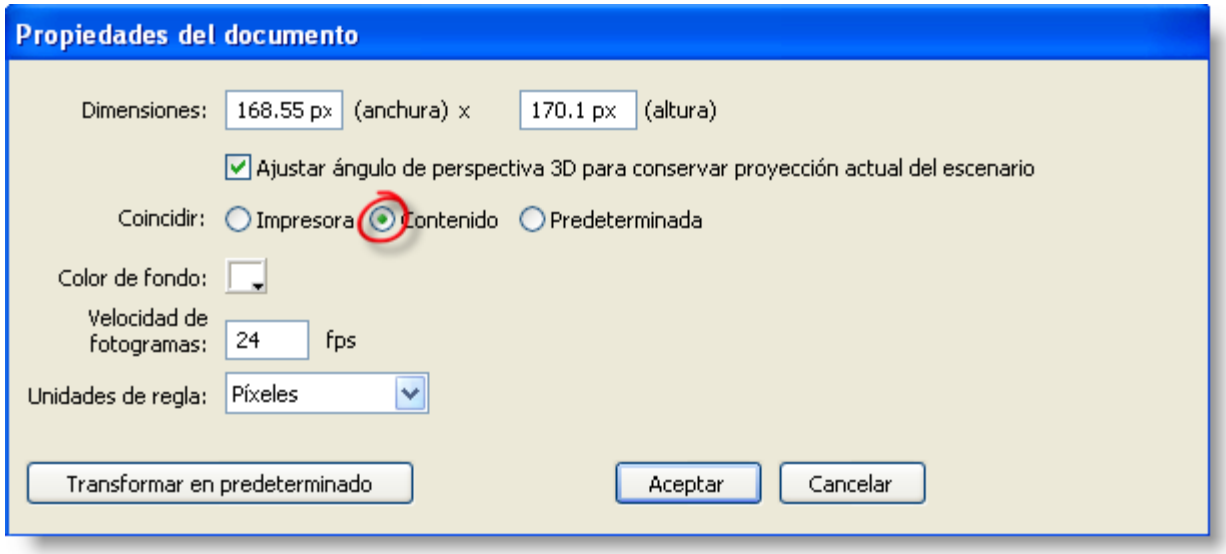
Tutorial 3. Creación de un logotipo textual

Paso 12 de 13



Vamos a cambiar el tamaño del escenario, ya que nos ha quedado demasiado grande en comparación con el logotipo. Seleccionamos un lugar vacío del escenario para que el **inspector de Propiedades** nos muestre la información sobre nuestra película. Pulsamos sobre el botón **Editar** que se encuentra al lado del tamaño del archivo.

En el panel de **Propiedades del documento** seleccionamos **Coincidir: Contenido**. De esta forma el documento adaptará automáticamente su tamaño al de nuestro logo.



Guarda el archivo como *tutorial3.fla*. Según el uso que le quisiéramos dar al logotipo, podríamos por ejemplo guardar su imagen seleccionando **Archivo > Exportar > Exportar imagen**, y en **Tipo** seleccionar **PNG**.

Tutorial 3. Creación de un logotipo textual

Paso 13 de 13

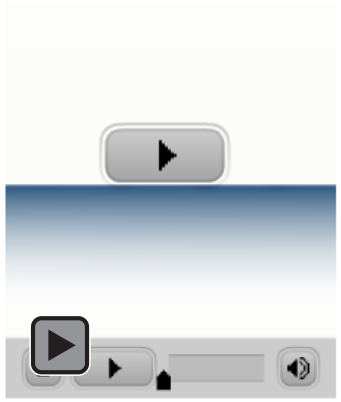
Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Haced alguna otra modificación en las letras del símbolo original del logotipo, y observad cómo ese cambio se aplica tanto al logotipo como a su reflejo.
2. Experimentad con diferentes tipos de filtros y de valores.
3. Probad los efectos de los diferentes tipos de mezclas.

Tutorial 4. Animación del logotipo

Paso 1 de 13

Para introducirnos en el campo de las animaciones, vamos a comenzar realizando una sencilla animación partiendo del logotipo que hemos creado.



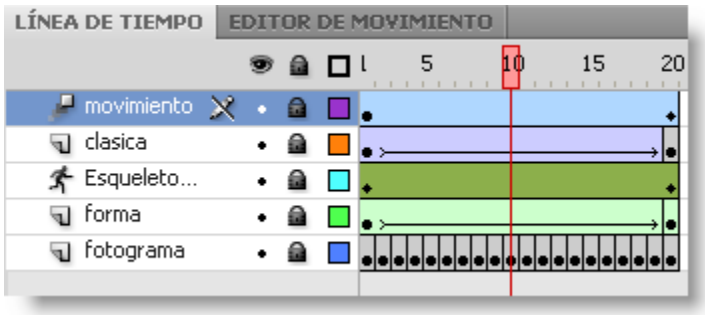
Para ello, en primer lugar abriremos el archivo *tutorial3.fla* que habíamos guardado en el tutorial anterior, y lo guardaremos como *tutorial4.fla*. De esta forma mantendremos intacto el archivo del tutorial 3 por si necesitamos en algún momento volver a él.

En Flash existen diferentes formas de crear animaciones, ofreciendo cada una de ellas diferentes posibilidades:

-**Interpolaciones de movimiento**. Si asignamos distintos valores en distintos fotogramas a una o varias propiedades de un objeto, como por ejemplo la posición, la escala, la transparencia, etc., Flash calculará los valores intermedios entre ambos fotogramas, de tal forma que el cambio sea gradual. Es el tipo de interpolación que crearemos en este tutorial.
-**Interpolaciones clásicas**. Son la forma en la que se creaban interpolaciones en las versiones anteriores de Flash. Son similares en varios aspectos a las interpolaciones de movimiento, aunque son más difíciles de crear y manipular.
-**Poses de cinemática inversa**. Son animaciones mediante *huesos* que trataremos en el siguiente tutorial.
-**Interpolaciones de forma**. Son animaciones en las que pasamos de una forma a otra diferente. Al igual que en otro tipo de animaciones, Flash crea los pasos intermedios entre dos fotogramas, en este caso entre formas diferentes.
-**Fotograma a fotograma**. Flash muestra simplemente cada fotograma en un momento en el tiempo, por lo que, como en el cine, se puede crear la sensación de animación al mostrar cada fotograma un contenido diferente. Es la forma más laboriosa de animar.

Tutorial 4. Animación del logotipo
Paso 2 de 13

Esta es la forma en la que Flash muestra en la **línea de tiempo** los diferentes tipos de animaciones:



Salvo en la animación fotograma a fotograma, en el resto sólo hemos indicado el primer y el último fotograma, y es Flash el que ha creado automáticamente los pasos intermedios (interpolaciones).

En este caso podemos ver que todas las animaciones tienen una duración de 20 fotogramas. La velocidad de las animaciones dependerá de a cuántos fotogramas por segundo tengamos configurado nuestro documento. Si lo tuviéramos configurado a 20 FPS, estas animaciones durarían exactamente 1 segundo.

Podemos observar también que la cabeza lectora se encuentra en este momento en el fotograma 10.

Comenzando ya con nuestra animación, seleccionamos la instancia del **clip de película** *logo* que se encuentra en la parte superior, hacemos clic con el **botón derecho del ratón**, y seleccionamos **Crear interpolación de movimiento**.

Otras formas de crear esta interpolación es seleccionando **Insertar > Interpolación de movimiento** o bien seleccionando el fotograma en la línea de tiempo y, también haciendo clic con el **botón derecho del ratón sobre el fotograma** en el que se encuentra la instancia, seleccionar **Crear interpolación de movimiento**.

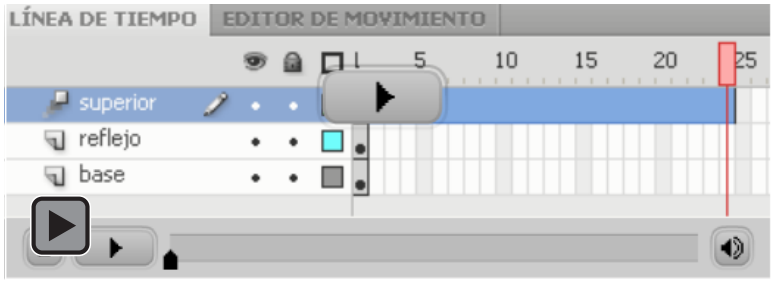
Para poder hacer este tipo de interpolaciones debemos tener cada objeto que queramos animar en una capa diferente. En este caso ya teníamos el logo distribuido en capas, por lo que no tendremos ningún problema.

Tutorial 4. Animación del logotipo
Paso 3 de 13

Tras crear la interpolación, vemos que se han creado 24 fotogramas en la capa en la que teníamos la parte superior del logo. El número de fotogramas que se añaden automáticamente depende de los fotogramas por segundo a los que tengamos configurado el documento (24 FPS por defecto).

La cabeza lectora se ha desplazado también al fotograma 24. En el escenario sólo veremos la parte superior del logo, ya que las otras dos capas sólo se muestran en el primer fotograma.

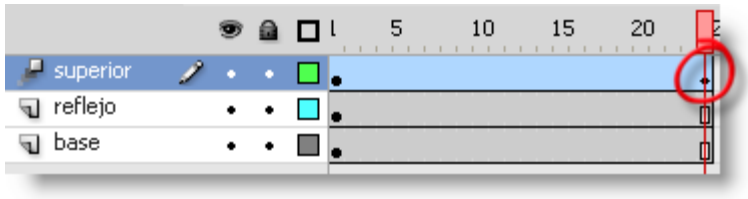
Para poder ver el resto de las capas, tenemos que insertar fotogramas para que se mantengan visibles hasta el fotograma 24. Para ello, nos situamos en el fotograma 24 de cada capa y seleccionamos **Insertar > Línea de tiempo > Fotograma**, o bien pulsamos **F5**.



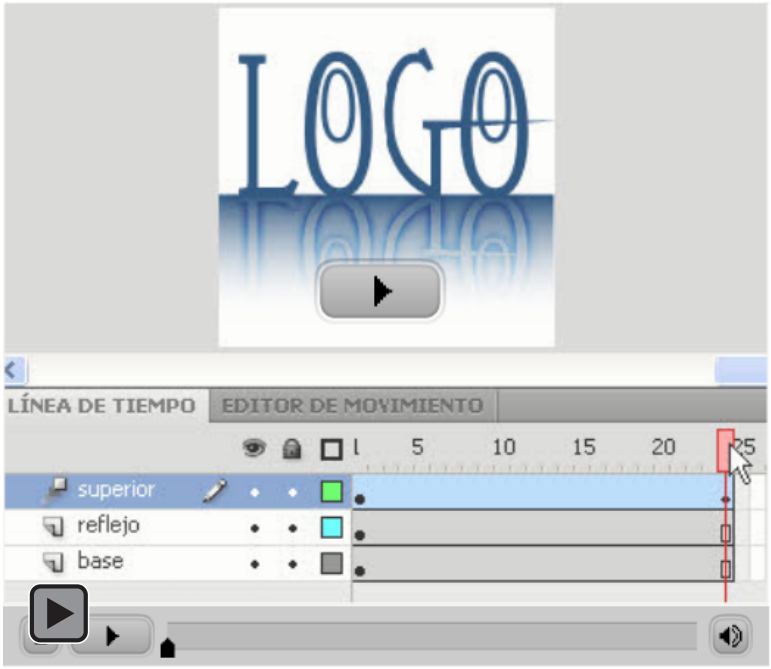
Seleccionamos de nuevo la capa superior. Teniendo la cabeza lectora en el fotograma 24, si ahora cambiáramos el logo de posición se crearía automáticamente una interpolación de movimiento desde la posición de inicio a la nueva posición.

Sin embargo en este caso queremos que la animación termine con el logo en la posición en la que se encuentra actualmente, por lo que el proceso va a ser crear en primer lugar un **fotograma clave** para mantener la posición actual en el fotograma 24.

Para ello seleccionamos **Insertar > Línea de tiempo > Fotograma clave**. Aparecerá un pequeño rombo en el fotograma 24 de la primera capa que indica que se ha creado un **fotograma clave de propiedad**, es decir, que en ese fotograma se almacenan los valores actuales de las diferentes propiedades de la instancia del logo.



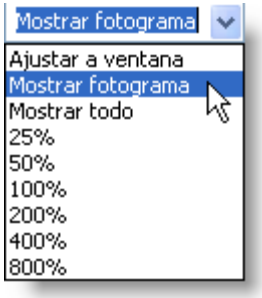
Tutorial 4. Animación del logotipo
Paso 4 de 13



Desplazamos la cabeza lectora al primer fotograma, y bajamos la posición del logo hasta que quede por debajo de la base. Para desplazarlo con facilidad y asegurarnos de que sólo variamos la altura, utilizaremos la **flecha hacia abajo del teclado**.

De momento el logo no se ocultará tras la base, sino que se verá por delante de ella, pero esto lo solucionaremos más adelante.

Para poder definir la posición con mayor detalle, podemos ampliar el tamaño del escenario seleccionando **Mostrar fotograma**.

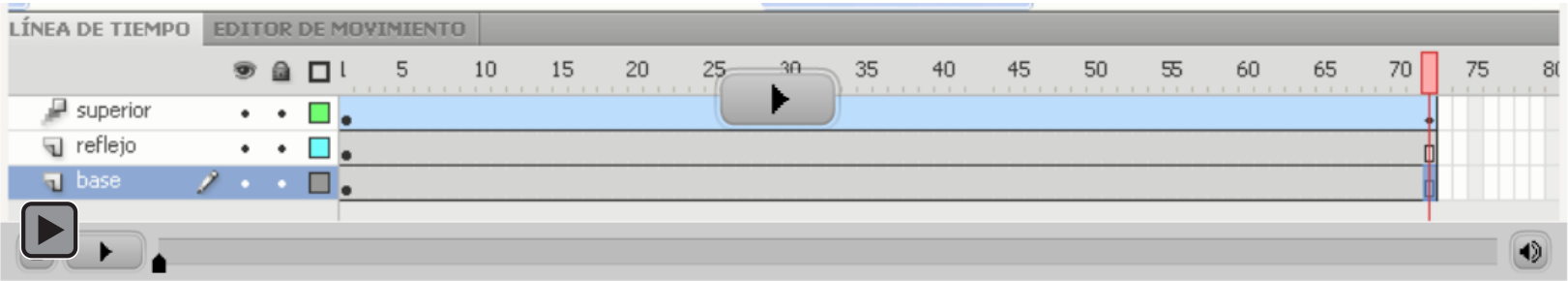


A medida que vayamos desplazando la posición vertical del logo, observaremos cómo aparece en el escenario una línea con puntos en su recorrido. Los puntos inicial y final de la línea muestran la posición del logo en los fotogramas clave de inicio y final de la animación, y los puntos intermedios representan las posiciones que tendrá el logo en cada fotograma de la animación.

Si desplazamos la cabeza lectora podremos ver el recorrido que realiza el logo a lo largo de los fotogramas. Si queremos ver la animación en tiempo real, nos posicionamos en el fotograma 1 y pulsamos la tecla **intro**.

Tutorial 4. Animación del logotipo

Paso 5 de 13



Al visualizar la animación en tiempo real, vemos que va demasiado rápido, por lo que vamos a alargar su duración a 72 fotogramas (tres segundos). Para ello acercamos el cursor al último fotograma de la animación y simplemente pulsamos y arrastramos el fotograma hasta su nueva posición.

Veremos que aparecen más puntos en la línea que muestra la interpolación, ya que ahora la animación transcurre a lo largo de más fotogramas.

Añadimos fotogramas a las otras dos capas tal y como hicimos en el paso 3, es decir, seleccionando el fotograma vacío correspondiente y pulsando **F5**, ya que, al no ser una interpolación de movimiento, no podemos extender directamente la duración de la capa.

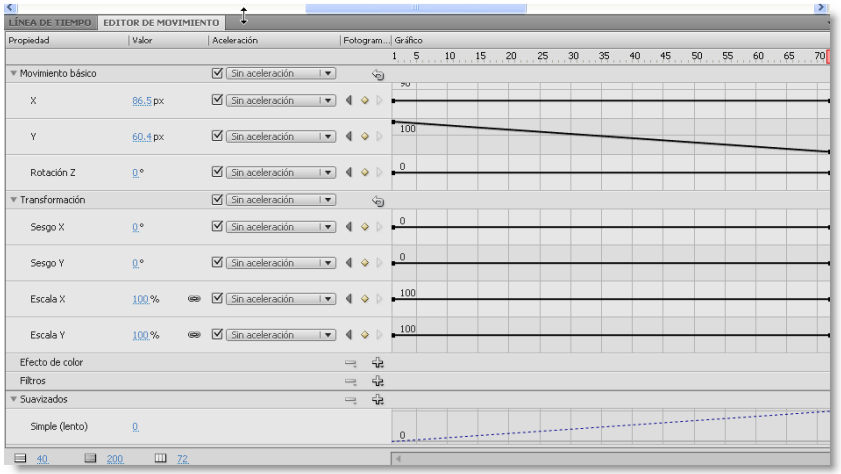
Por el momento, la parte superior del logo se desplaza con velocidad uniforme desde la posición inicial, bajo la línea superior de la base, hasta su posición final en la parte superior, y tarda tres segundos en realizar este recorrido.

En los pasos siguientes, con ayuda del editor de movimiento, transformaremos este movimiento uniforme en un movimiento que simulará un pequeño rebote en la parte superior.

Con el logo superior o la línea verde seleccionada, clicamos sobre la pestaña **Editor de movimiento** que se encuentra junto a la pestaña Línea de tiempo.

Tutorial 4. Animación del logotipo

Paso 6 de 13



Para poder visualizar un área más amplia de este panel, podemos arrastrar la parte superior del panel (la parte superior de la barra gris oscuro).

En la parte inferior del panel podemos seleccionar diferentes valores para la visualización de los diferentes elementos del panel. A **tamaño de gráfico** le daremos un valor de 20, y a **fotogramas visibles** un valor de 72, para poder de esta forma ver toda la evolución de la animación a lo largo de la línea de tiempo.

En la columna **Propiedad** podemos ver las distintas propiedades que podemos modificar en una interpolación de movimiento (X, Y, Rotación Z, Sesgo Y, etc.), agrupadas en diferentes categorías.

En la columna **Valor** vemos los valores de las diferentes propiedades en el fotograma en el que se encuentra la cabeza lectora.

La columna **Aceleración** muestra si la interpolación de la propiedad correspondiente tiene asignada algún tipo de aceleración.

En la columna **Fotogramas** podemos añadir o eliminar fotogramas clave, así como desplazarnos entre ellos.

El área principal, la correspondiente a **Gráfico**, muestra la evolución de los valores de cada propiedad a lo largo del tiempo. En este caso podemos ver que todo son líneas rectas excepto la propiedad Y, que muestra una línea descendente. Esto significa que la propiedad Y (altura) del logo comienza con un valor alto, que va disminuyendo según avanza la animación (los valores más bajos de Y corresponden a posiciones más altas en el escenario). Las demás propiedades no presentan ninguna modificación.

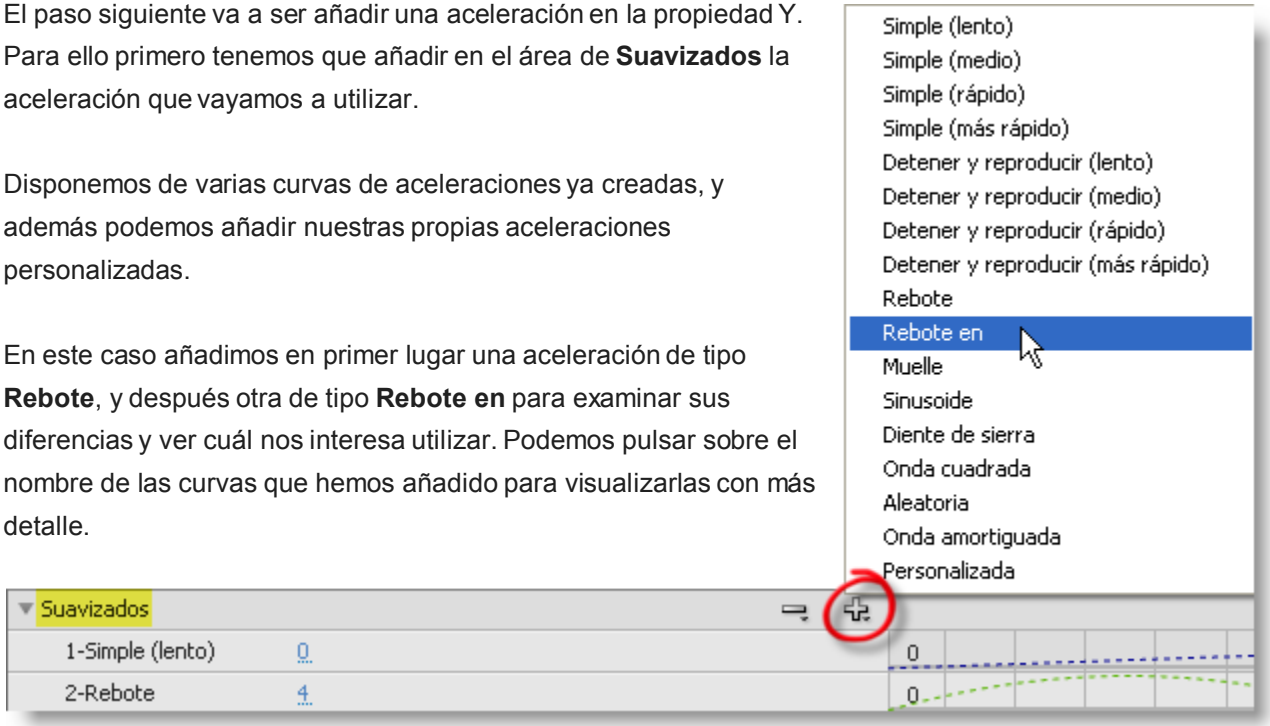
Tutorial 4. Animación del logotipo

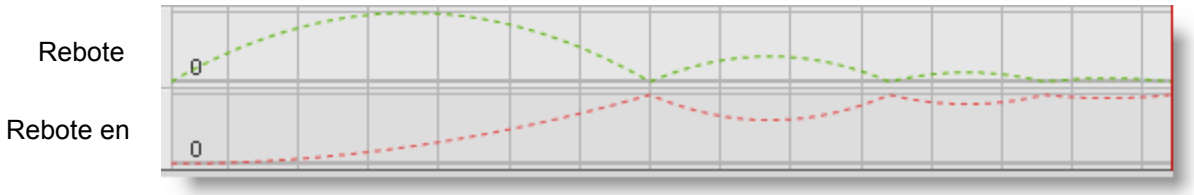
Paso 7 de 13

El paso siguiente va a ser añadir una aceleración en la propiedad Y. Para ello primero tenemos que añadir en el área de **Suavizados** la aceleración que vayamos a utilizar.

Disponemos de varias curvas de aceleraciones ya creadas, y además podemos añadir nuestras propias aceleraciones personalizadas.

En este caso añadimos en primer lugar una aceleración de tipo **Rebote**, y después otra de tipo **Rebote en** para examinar sus diferencias y ver cuál nos interesa utilizar. Podemos pulsar sobre el nombre de las curvas que hemos añadido para visualizarlas con más detalle.





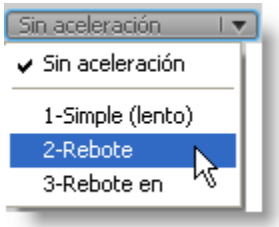
En el área de Suavizados, el valor más bajo de cada curva (0) indica el valor de la propiedad en el fotograma inicial de la animación, y el valor más alto (100) indica el valor de la propiedad en el fotograma final.

En el caso de la curva **Rebote,** vemos que la animación pasa del valor del primer fotograma al último para volver al valor inicial, y la animación termina, después de unos rebotes, en el valor que tenía la propiedad en el primer fotograma.

En el caso de **Rebote en**, la animación rebota en el valor del último fotograma, y es en éste en el que termina. Por lo tanto será este tipo de curva el que nos interesa para nuestra animación.

Tutorial 4. Animación del logotipo
Paso 8 de 13

Para entender mejor el significado de las curvas de aceleración, vamos a aplicar ambos tipos de curvas en nuestra animación.



En la fila correspondiente a la **propiedad Y**, seleccionaremos **Rebote** en el desplegable de la aceleración. Vemos que en el desplegable sólo aparecen las aceleraciones que hemos añadido en el área de suavizados.

Para ver la curva con más detalle pulsamos sobre el nombre de la propiedad Y (podemos volver a pulsar sobre el nombre para visualizar de nuevo un tamaño reducido).

En este caso vemos que la animación comienza con el valor inicial, baja después al valor final, sube después al valor inicial, baja ligeramente, etc., hasta terminar en el valor inicial. Pulsamos la tecla **intro** para ver el resultado de esta animación. El resultado es que el logo sube para finalmente volver a caer.

Ahora seleccionamos **Rebote en** en el desplegable de la aceleración de la propiedad Y. Aquí podemos ver cómo la animación rebota y termina en la posición final. Pulsamos de nuevo la tecla **intro** para ver la animación resultante y vemos que ya se asemeja al resultado que queremos conseguir.

Si queremos cambiar la velocidad del rebote, podemos asignar diferentes valores a la aceleración en el área de suavizado. Por ejemplo, vamos a asignar un valor de 10 a **Rebote en** en el área de **Suavizados**. Prueba con diferentes valores para observar las diferencias en la curva y, por tanto, en la animación.

Si necesitamos ajustar la posición inicial o final del logo, podemos hacerlo directamente desde el área **Gráfico** de la propiedad Y, arrastrando directamente la posición de los pequeños cuadrados negros que representan a los fotogramas clave.

Tutorial 4. Animación del logotipo
Paso 9 de 13

En este paso vamos a ocultar el logo tras la base, de tal forma que sólo se vea la parte del logo superior que quede por encima de la base, y para ello vamos a utilizar una **máscara**.

Una máscara es una capa que determina qué áreas de las capas asociadas bajo ella son visibles. Las zonas rellenas de la máscara corresponden a las zonas visibles de las capas asociadas; allí donde la máscara está vacía, no es visible el contenido de las capas asociadas.

En primer lugar, creamos una nueva capa a la que llamamos *máscara* (el nombre puede ser otro).

Clicamos con el **botón derecho** del ratón sobre el nombre de la nueva capa y seleccionamos **Máscara** en el menú contextual.

Automáticamente cambiará el icono de la capa para mostrar que se trata de una capa tipo Máscara, y se asociará a ella la capa inmediatamente inferior. Podríamos arrastrar más capas bajo la máscara para que también se vean afectadas por ella, aunque en este caso no lo vamos a hacer.

Desbloqueamos la capa *mascara* clicando sobre el icono del candado. En esta nueva capa dibujamos un rectángulo sin trazo y con relleno rojo que ocupe exactamente la parte superior, es decir, la parte en la que queremos que se muestre la capa *superior*, que es la que contiene la animación que hemos realizado hasta el momento.

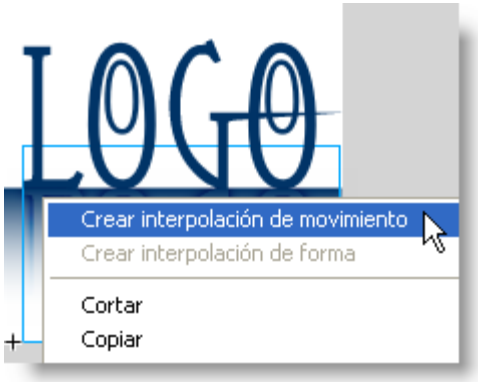
Para ver el efecto de la máscara que hemos creado, tenemos que bloquear de nuevo tanto la máscara como las capas que estén asociadas a ella.

Pulsando **intro**, vemos que sólo se muestra el logo superior cuando está en el área que habíamos definido con el rectángulo rojo.

Tutorial 4. Animación del logotipo
Paso 10 de 13

Ahora realizaremos la animación del reflejo de forma similar a como hemos animado la parte superior del logo.

Bloqueamos todas las capas excepto la capa *reflejo*, y clicamos sobre el reflejo en el escenario con el **botón derecho** del ratón. Seleccionamos **Crear interpolación de movimiento** en el menú contextual.

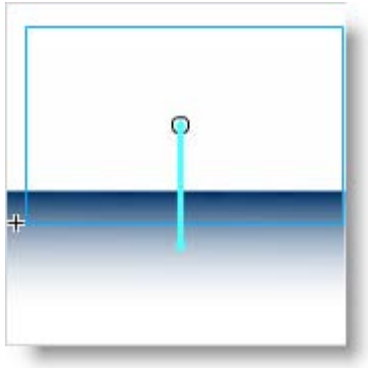


En este caso vemos que la animación ocupa automáticamente los 72 fotogramas que ocupaba esa capa. Por lo tanto, si una capa dispone de un número de fotogramas mayor que 1, las animaciones que creemos en esa capa ocuparán por defecto el número de fotogramas disponibles para esa capa, y no el número de FPS del documento.

Estando en el fotograma 72 subimos el reflejo en el escenario con ayuda de la **flecha hacia arriba** del teclado, hasta que desaparezca completamente por encima de la base. Automáticamente se creará un **fotograma clave de propiedad**.

Después seleccionamos la interpolación en la línea de tiempo y, pulsando con el **botón derecho** del ratón, seleccionamos **Invertir fotogramas clave**. Ahora el reflejo comenzará invisible arriba y acabará abajo. Esta es otra manera de conseguir que el primer fotograma pase a ser el último de la animación.

En este caso el reflejo desaparece sin utilizar una máscara, ya que habíamos usado para esta capa la mezcla **Superponer** en el tutorial anterior, y eso hace que no sea visible en un fondo blanco.

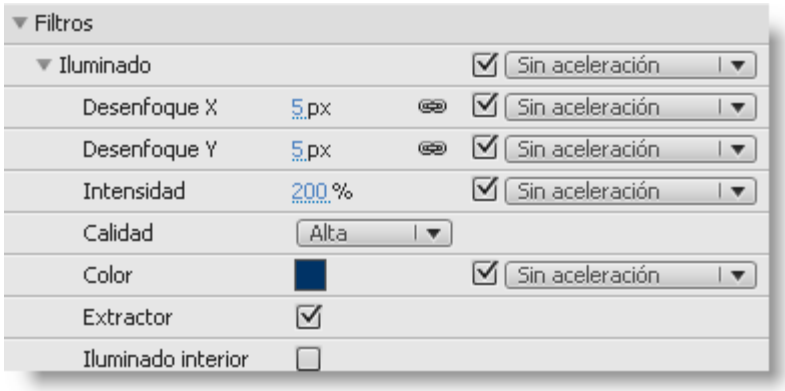


Tutorial 4. Animación del logotipo
Paso 11 de 13

Si pulsamos la tecla **intro** para probar la animación, veremos que el reflejo va de arriba abajo en el escenario de forma uniforme.

Para añadir el mismo efecto que en la parte superior, volveremos a realizar los mismos pasos que hicimos para animar la parte superior.

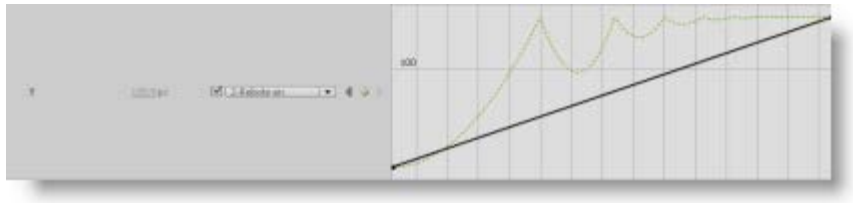
Seleccionamos el reflejo en el escenario y abrimos el **Editor de movimiento**. Al observar este panel podemos ver que hay algunos cambios respecto a la anterior animación. Por ejemplo, ahora aparece un valor en negativo en la propiedad **Escala Y** que indica que hemos invertido el logotipo. También aparecerá el filtro **Iluminado** que habíamos aplicado, y que también podríamos animar.



Añadimos **Rebote en** primero en el área de **Suavizados**. Le damos un valor de 10 (o el valor que hayamos decidido darle en el paso 8 a la parte superior del logo). En el desplegable **Aceleración** de la propiedad **Y** seleccionamos **Rebote en**.

En este caso vemos que la forma que adopta la curva en la propiedad Y es diferente, ya que comienza con un valor de Y bajo, y termina con un valor alto.

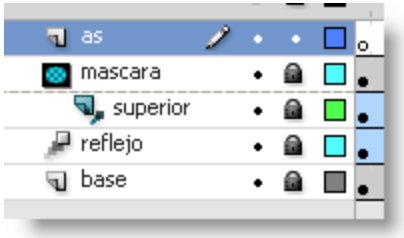
Para una mayor precisión, recordemos que podemos ajustar las posiciones de principio o de final de la animación desplazando los recuadros negros que representan a los fotogramas clave de inicio y fin de la animación.



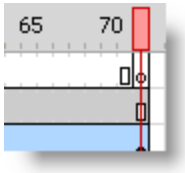
Tutorial 4. Animación del logotipo
Paso 12 de 13

Seleccionamos **Control > Probar película** para probar nuestra animación. Esta opción genera automáticamente un archivo **swf** en la misma carpeta en la que tengamos nuestro archivo **fla**.

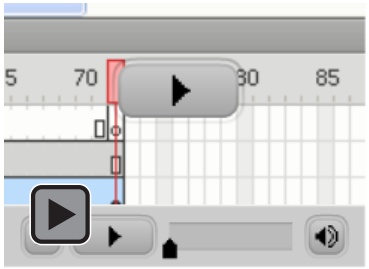
En el archivo swf que se ha generado podemos ver que la animación se repite indefinidamente. Para mostrar la animación una sola vez necesitaremos detener la línea de tiempo en el último fotograma de la animación.



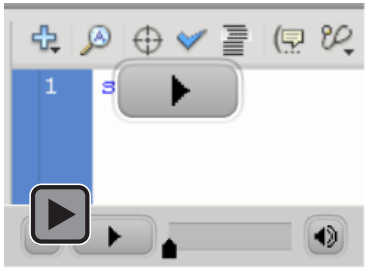
Creamos una nueva capa a la que llamaremos *as* (por ActionScript), y en la que escribiremos el código necesario para detener la película.



En el fotograma 72 de esta capa, pulsamos **F6** para insertar un **fotograma clave**. Se mostrará un pequeño círculo vacío, ya que de momento ese fotograma no tiene contenido.



Con el fotograma seleccionado, hacemos clic con el **botón derecho** del ratón y seleccionamos **Acciones**, o bien seleccionamos **Ventana > Acciones** para abrir el panel Acciones.

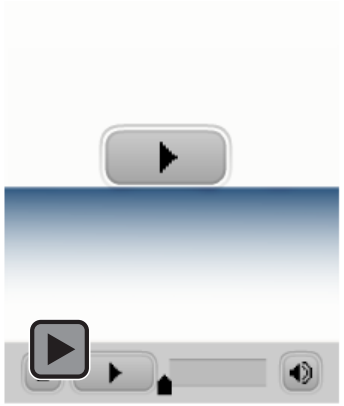


Escribimos **stop();** y cerramos el panel. El fotograma mostrará una pequeña *a* que indica que ese fotograma tiene asignada una acción de fotograma.

Seleccionamos de nuevo **Control > Probar película** para probar nuestra animación. Ahora la animación se detiene al llegar al último fotograma, ya que es el que tiene la acción *stop()* asignada.

Ahora ya tendremos un archivo swf que contiene nuestra animación del logotipo. En tutoriales más avanzados veremos con detalle las diferentes opciones para publicar nuestras películas.

Tutorial 4. Animación del logotipo
Paso 13 de 13



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Probad los diferentes tipos de suavizados disponibles para explorar su funcionamiento.
2. Cread una animación para el filtro iluminado del reflejo modificando los valores del fotograma 1 en el editor de movimiento.
3. Cread desde el editor de movimiento un fotograma clave en un punto intermedio de la línea de tiempo en el logo superior, y variad su posición para observar los efectos en la animación.

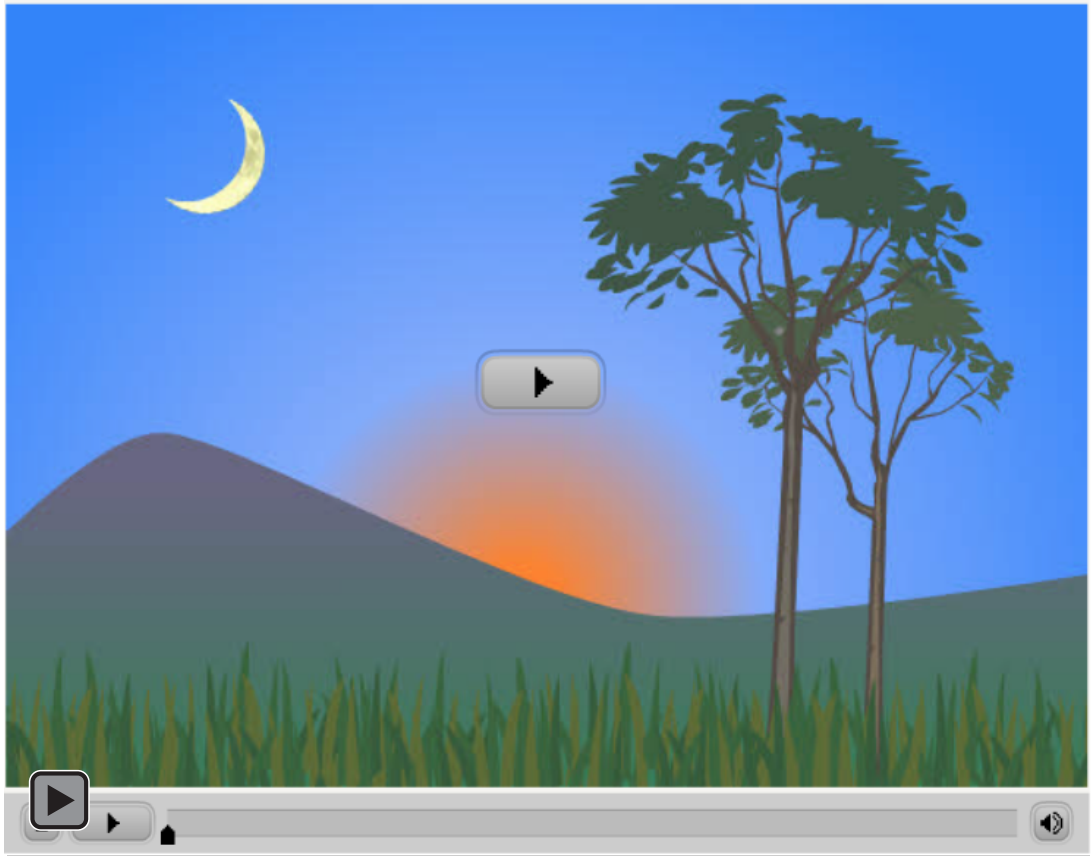
Tutorial 5. Cinemática inversa y herramientas 3D

Paso 1 de 21

En este tutorial vamos a conocer algunas herramientas que pueden resultar muy útiles para realizar animaciones, como la **herramienta hueso** o las **herramientas de rotación y traslación 3D**.

También veremos las ventajas de utilizar clips de película anidados para realizar algunos tipos de animación.

Para hacer este tutorial vamos a utilizar el pájaro que creamos en el tutorial 1 y el paisaje que creamos en el tutorial 2.



Tutorial 5. Cinemática inversa y herramientas 3D
Paso 2 de 21

En muchos casos en los que necesitemos realizar una animación será conveniente crear una estructura jerárquica de clips de película, teniendo cada uno de ellos su propia línea de tiempo.

Supongamos por ejemplo el caso de la animación de un coche. Tendremos en primer lugar un clip de película llamado *coche* que se desplazará de un lado a otro del escenario. El clip de película estará formado posiblemente de diversas capas con sus distintos elementos (ventanas, ruedas, etc.). Al desplazar el clip *coche* por el escenario se desplazarán al unísono todas las capas que contiene, facilitando de esa forma la animación.

Necesitaremos también que las ruedas giren mientras el coche se desplaza. Podemos convertir las ruedas en un clip de película, y en su línea de tiempo hacer que gire sobre sí misma. Es suficiente con una vuelta completa, ya que los clips de película se reproducirán por defecto como un bucle.

Ahora tendremos en el escenario una línea de tiempo principal en la que un clip llamado *coche* se desplaza por el escenario, y a su vez el clip *coche* estará formado por varios objetos, entre los que se encuentran los clips de las *ruedas*, que a su vez tienen una línea de tiempo propia en la que giran.

De esta forma, vaya donde vaya el coche en la línea de tiempo principal, veremos cómo giran las ruedas, ya que es una línea de tiempo que está anidada dentro del clip *coche*.

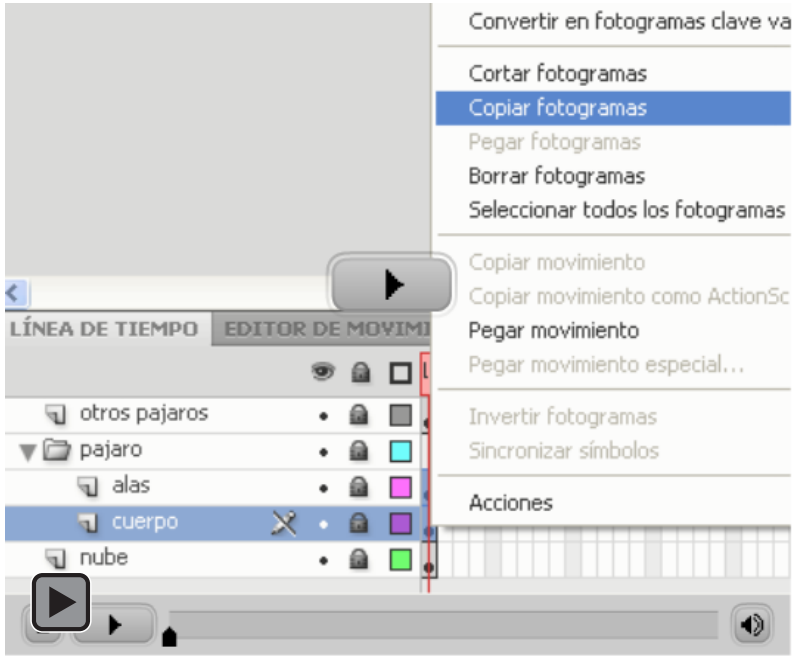
Los clips anidados pueden ser útiles en muchos tipos de animaciones, por ejemplo, el movimiento o parpadeo de los ojos estaría dentro del clip de una cabeza, que a su vez estaría, junto con otras partes del cuerpo, dentro de un clip de un cuerpo humano.

En nuestro caso vamos a desplazar en el escenario un clip llamado *pájaro*, que a su vez estará formado por el cuerpo y las alas, que tendrán a su vez su propia línea de tiempo en la que crearemos el movimiento del batir de las alas. De esta forma, al desplazar al pájaro por el escenario principal, siempre veremos el batir de las alas, ya que serán un clip de película anidado dentro del clip *pájaro*.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 3 de 21

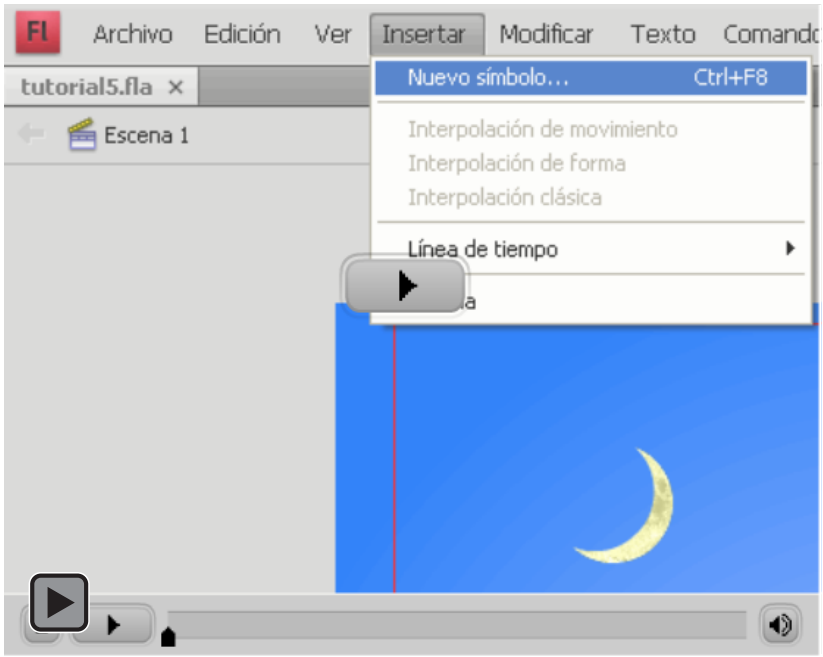
Abrimos el documento *tutorial1.fla* y desplegamos la carpeta que contiene el pájaro principal del documento. Seleccionamos los fotogramas de las dos capas clicando sobre ellos con la tecla **Ctrl** pulsada.

Después hacemos clic con el **botón derecho** del ratón y seleccionamos **Copiar fotogramas** del menú contextual.



Ahora ya podemos cerrar el *tutorial1.fla* sin necesidad de guardar los cambios. Abrimos el *tutorial2.fla*, que contiene el paisaje, y lo guardamos como *tutorial5.fla*, que será el documento en el que vamos a hacer la animación.

Creamos un nuevo clip de película vacío seleccionando **Insertar > Nuevo Símbolo**, y lo llamamos *pájaro*.

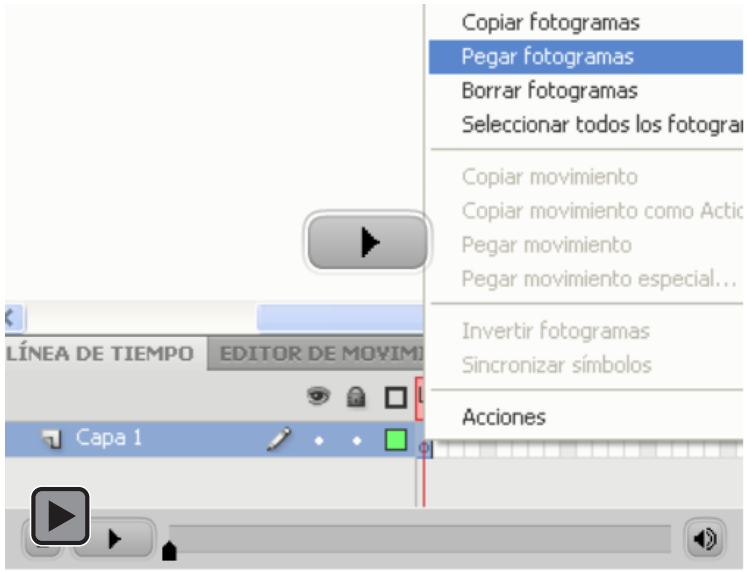


Tutorial 5. Cinemática inversa y herramientas 3D
Paso 4 de 21

Ahora nos encontraremos dentro del clip de película que hemos creado.

Al estar dentro de un clip de película, la línea de tiempo mostrará el contenido de ese clip de película. En este caso veremos que de momento sólo hay una capa y no tiene contenido.

Seleccionamos el fotograma vacío con el **botón derecho** del ratón, y seleccionamos **Pegar fotogramas**.



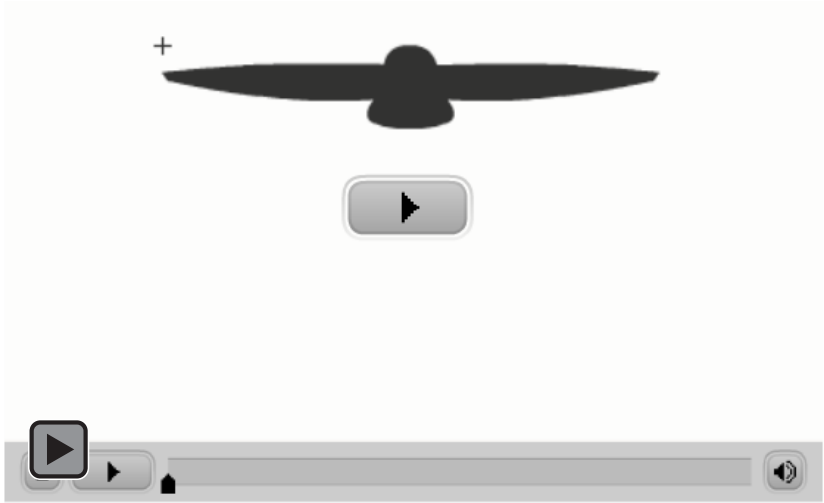
Los dos fotogramas que habíamos copiado del *tutorial1.fla* aparecerán automáticamente en este clip de película. Los nombres de las capas también se copiarán del archivo de origen.

Todavía dentro del clip *pájaro*, seleccionamos el cuerpo y las alas en el escenario y situamos el conjunto seleccionado en la posición X:0 e Y:0. Con la anchura y altura bloqueadas, asignamos una anchura total de 200.

El motivo de realizar este paso es que en el tutorial 1 pudimos hacer pájaros con tamaños muy diversos, y de esta forma todos los pájaros tendrán una anchura de 200 con las alas extendidas, por lo que podremos seguir el tutorial partiendo de los mismos valores.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 5 de 21

Borramos el ala izquierda, seleccionamos el ala derecha y la convertimos en un clip de película llamado *ala*.



El ala izquierda será una instancia del mismo clip de película ala que hemos creado a partir del ala derecha. Para ello arrastramos una instancia del clip de película *ala* de la biblioteca al escenario.

Volteamos esta instancia seleccionando **Modificar > Transformar > Voltear horizontalmente**, y por último la alineamos con ayuda del panel **Alinear**, tal y como hicimos en el paso 9 del tutorial 1.

En este momento tendremos un clip de película llamado *pájaro*, que a su vez está compuesto del cuerpo y de dos instancias del clip de película *ala*.

Hacemos doble clic sobre el ala derecha para poder editar el clip de película sin perder de vista al resto del pájaro. La barra de edición mostrará que estamos editando el clip *ala*, dentro del contexto del clip *pájaro*.

Esta barra también mostrará una referencia a la escena principal para poder volver a ella, incluso aunque todavía no hayamos incluido el clip en la escena principal.

Tutorial 5. Cinemática inversa y herramientas 3D

Paso 6 de 21

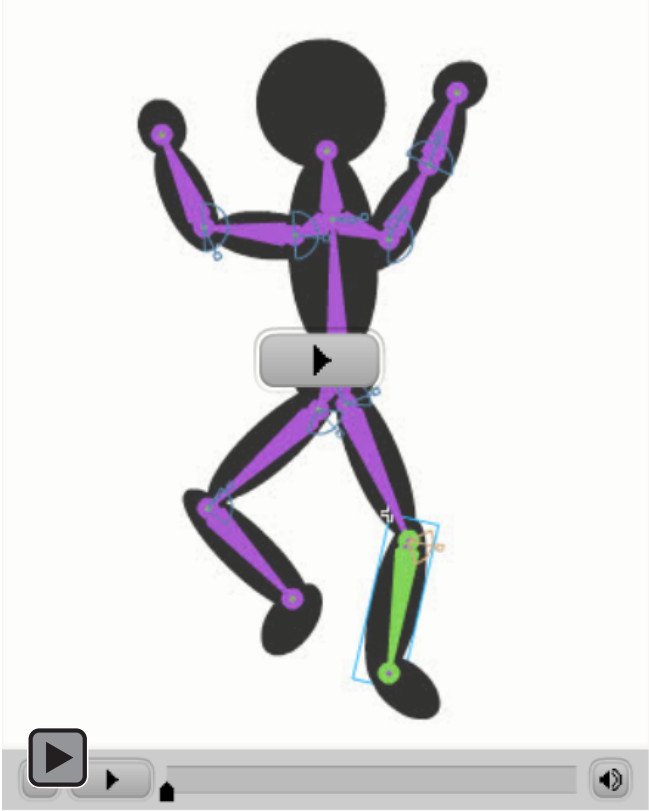
Vamos a animar el ala utilizando la **cinemática inversa**, que es un método de animación a través de una estructura articulada de huesos.

Los huesos pueden conectar diferentes instancias de símbolos, como se puede ver en el vídeo de ejemplo, o bien se pueden añadir al interior de una forma, como haremos en este tutorial.

Mediante la **herramienta Hueso** podemos generar un **esqueleto**, es decir, una cadena de huesos vinculados entre sí. Después podemos definir diferentes poses en distintos fotogramas, y Flash interpolará las posiciones de los fotogramas intermedios.

Mediante la cinemática inversa podemos crear animaciones más complejas y naturales con mayor facilidad. Para un mayor control, podemos añadir restricciones de movimiento, limitando por ejemplo el ángulo en que permitimos que una articulación pueda doblarse.

En este tutorial vamos a hacer tan sólo una pequeña aproximación a esta herramienta, por lo que recomendamos experimentar con todas las posibilidades que ofrece.



Tutorial 5. Cinemática inversa y herramientas 3D
Paso 7 de 21

Volviendo a nuestro clip *ala*, hacemos en primer lugar un zoom suficiente para trabajar con comodidad y detalle suficiente.

Seleccionamos la **herramienta Hueso** en la barra de herramientas.

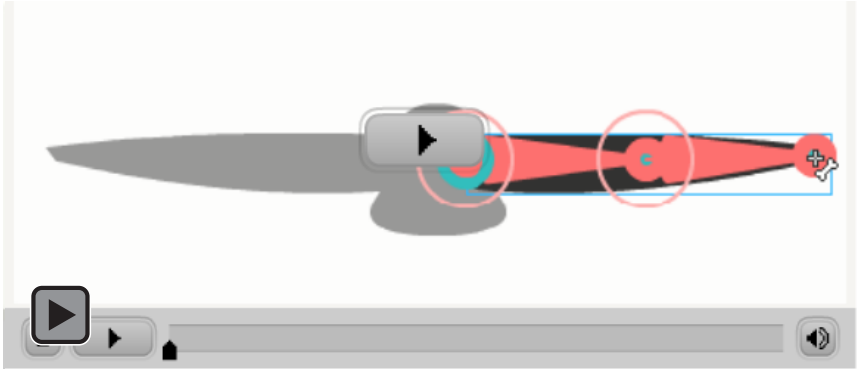
Hacemos clic en el centro del extremo interior del ala, y arrastramos hasta aproximadamente el punto central del ala, que aparecerá marcado con una pequeña cruz.

Después hacemos clic sobre el mismo punto en el que hemos acabado el hueso anterior, y arrastramos hasta el extremo exterior del ala.

Todos los elementos a los que se les añade un hueso pasan automáticamente a una nueva **capa de pose**. Esta capa tendrá un icono con una figura humana, y los fotogramas en la línea de tiempo tendrán un fondo verde.

En este caso, como hemos añadido huesos al único elemento que había, la capa original queda vacía y todo su contenido pasa automáticamente a la nueva capa de pose. Borramos la capa que ha quedado vacía.

Los huesos mostrarán el color aleatorio que haya sido asignado al contorno de la nueva capa. Si queremos podemos cambiarlo pulsando sobre el icono de color de la capa.

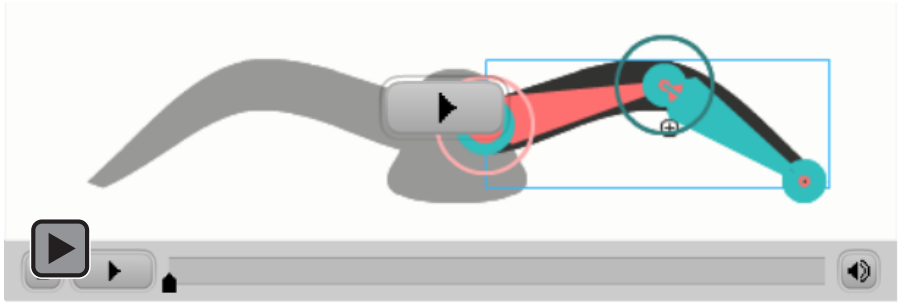


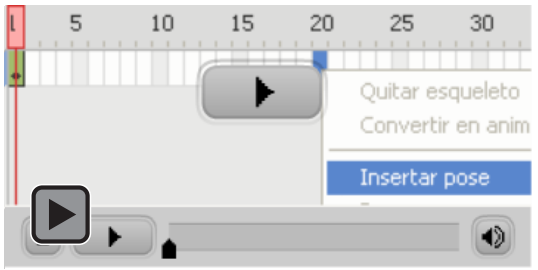
Tutorial 5. Cinemática inversa y herramientas 3D
Paso 8 de 21

Vamos a comenzar la animación con una posición ligeramente doblada del ala.

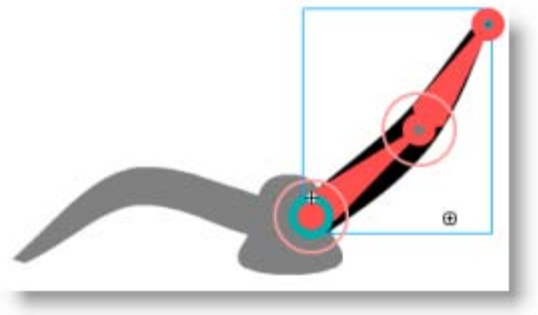
Con la **herramienta Selección (V)** clicamos y arrastramos la posición del ala hasta conseguir una posición similar a la que se muestra en el vídeo.

Podemos ver que ambas alas adoptan la misma posición, ya que se trata de instancias del mismo clip de película.





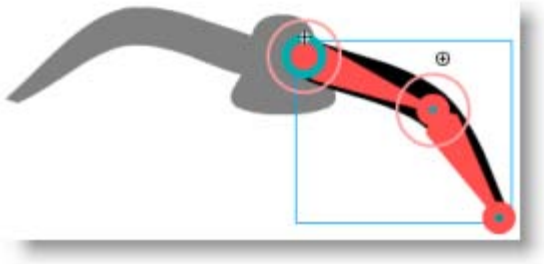
Clicamos con el **botón derecho** del ratón sobre el fotograma 20 y seleccionamos **Insertar pose**. En este fotograma modificamos la posición hasta conseguir que sea semejante a la de la imagen.

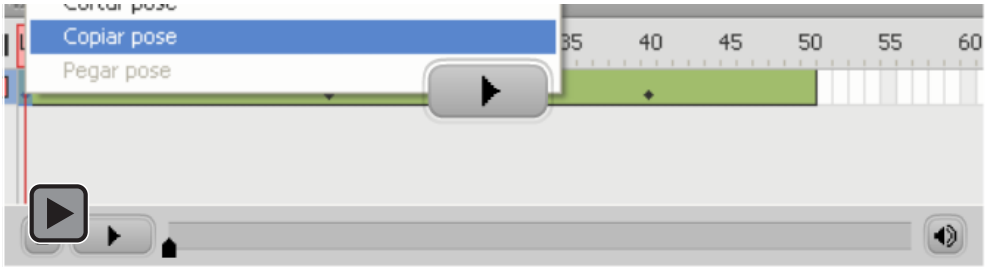


En este caso el ala izquierda no mostrará el cambio de posición, ya que cuando editamos un clip dentro de un contexto, el resto de los elementos permanecen en el fotograma en el que se encontraban de su propia línea de tiempo (el primer fotograma en este caso).

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 9 de 21

De la misma manera, insertamos una nueva pose en el fotograma 40, y bajamos el ala hasta una posición como la de la imagen.





En el fotograma 50 vamos a repetir la pose que habíamos creado en el fotograma 1. Para ello en primer lugar insertamos los fotogramas necesarios pulsando **F5** en el fotograma 50.

Para seleccionar un fotograma de una pose, hacemos clic sobre él mientras mantenemos pulsada la tecla **Ctrl**. De esta forma seleccionamos el fotograma 1, después hacemos clic con el **botón derecho** del ratón y seleccionamos **Copiar pose**.

Seleccionamos el fotograma 50 de la misma manera, pulsamos con el **botón derecho** del ratón y seleccionamos **Pegar pose**.

Para probar la animación del ala, pulsamos la tecla **Intro**. De esta forma la animación del ala se mostrará una sola vez.

Para mostrar la animación con un **bucle**, que es como se reproducirá en la película final, seleccionamos **Control > Reproducir indefinidamente**, y pulsamos de nuevo la tecla **Intro** tanto para iniciar como para detener la reproducción.

Si fuera necesario, modificaremos alguna de las poses hasta que nos parezca tener un movimiento natural.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 10 de 21

Al visualizar el bucle, podemos observar que la animación parece detenerse un instante en la pose inicial.

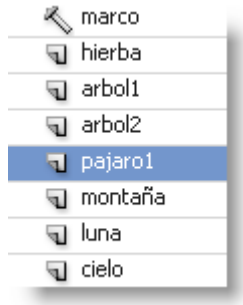
Esto se debe a que el fotograma 50 y el fotograma 1 tienen la misma pose, y después del fotograma 50 se reproduce de nuevo el fotograma 1, que tiene el mismo contenido.

Para solucionar este problema, insertaremos una nueva pose en el fotograma 49 seleccionando **Insertar pose**, que mostrará por defecto la pose calculada por Flash para el fotograma justo anterior al de la pose inicial (la misma del fotograma 50).

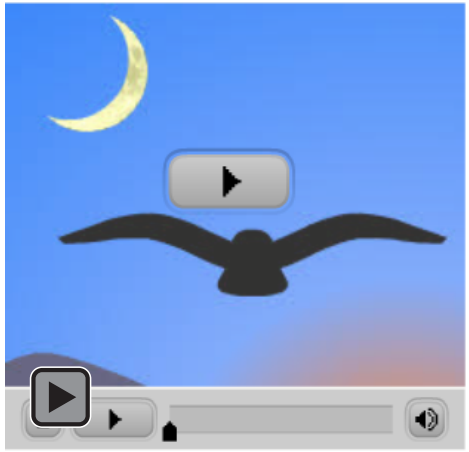
Después ya podemos eliminar el fotograma 50, seleccionando **Quitar fotogramas** en el menú contextual.

El bucle mostrará ahora el paso del fotograma 49 al fotograma 1, sin que haya ningún fotograma repetido, y con la última pose del fotograma 49 previa al fotograma 1 perfectamente calculada.

Ahora que ya tenemos creado el bucle con el batir de las alas, volvemos a la escena principal pulsando **Escena 1** en la barra de edición.



Creamos una nueva capa llamada *pajaro1* entre la capa *montaña* y la capa *arbol2*. Con esa capa seleccionada, arrastramos desde la biblioteca al escenario una instancia del clip de película *pájaro.*



Si pulsamos ahora la tecla **Intro** no veremos ninguna animación, ya que la línea de tiempo actual tiene sólo un fotograma.

Si seleccionamos **Control > Probar película** podremos ver todas las líneas de tiempo anidadas de la película, incluidas las alas.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 11 de 21

Vemos que el movimiento del ala va un poco lento, así que vamos a acelerar ligeramente la velocidad del bucle.

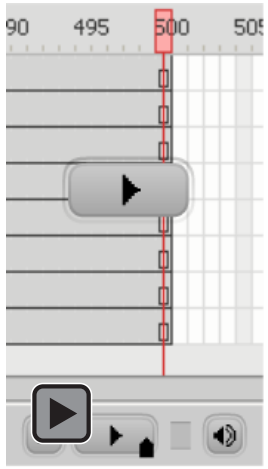
Podemos editar directamente el clip *ala* haciendo doble clic sobre su nombre en la **biblioteca**, ya que para aumentar la velocidad no necesitamos tener una referencia visual del resto del pájaro.

Vamos a reducir la duración del bucle en 10 fotogramas. Para ello clicamos sobre el extremo de la animación y lo arrastramos hasta el fotograma 39. Vemos que la posición del resto de los fotogramas se han reajustado proporcionalmente a la nueva duración.

Si probamos de nuevo la película veremos que ahora las alas se mueven más rápido. Como hemos podido ver, podemos variar en cualquier momento alguna de las poses de un clip o su duración, y estos cambios se reflejarán en las instancias que estemos utilizando.

El siguiente paso va a ser crear la animación con el desplazamiento del pájaro en la línea de tiempo principal, es decir, en la línea de tiempo que contiene el paisaje.

Como punto de partida situamos fuera del escenario, a la altura de la luna, la instancia del pájaro que tenemos en la capa *pajaro1.*

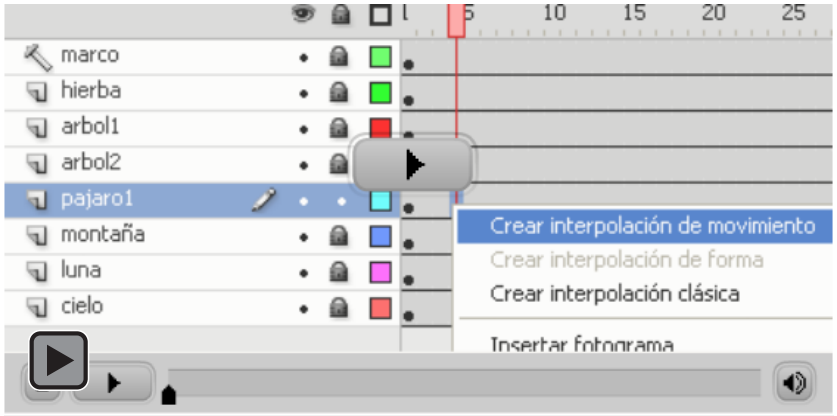


Vamos a insertar 500 fotogramas en todas las capas de la escena, ya que es la duración prevista para nuestra animación.

La forma más rápida para hacerlo es situarnos en el fotograma 500 con ayuda de la barra inferior de desplazamiento, pulsar sobre el fotograma 500 de una de las capas y, sin soltar, arrastrar para seleccionar el fotograma 500 del resto de las capas. Por último, pulsamos **F5** para añadir fotogramas en todas las capas seleccionadas.

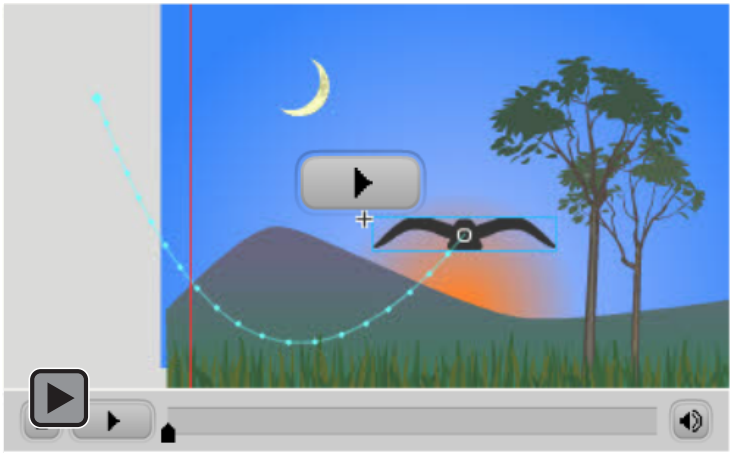
Tutorial 5. Cinemática inversa y herramientas 3D
Paso 12 de 21

Seleccionamos con el **botón derecho** del ratón cualquier punto de la capa *pajaro1* y seleccionamos **Crear interpolación de movimiento**.



Situamos la cabeza lectora en el fotograma 500 y, con la herramienta **Selección**, desplazamos la instancia del clip *pájaro* a un punto más o menos central del escenario.

Aparecerá una línea de interpolación recta con varios puntos intermedios equidistantes. A diferencia del tutorial anterior, en este caso no aparece un punto por cada fotograma, ya que tenemos un número demasiado elevado de éstos.



Vamos a curvar la trayectoria de la interpolación pulsando y arrastrando directamente la línea de interpolación. Con esto definimos un recorrido en el que se desplaza hacia la derecha, primero descendiendo y finalmente ascendiendo.

Como cualquier otra curva, podríamos afinar su forma clicando sobre sus puntos extremos con la herramienta **Subselección**, aunque en este caso no va a ser necesario.

Si introdujéramos nuevos fotogramas clave de propiedad en puntos intermedios de la interpolación, podríamos definir curvas complejas con facilidad.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 13 de 21

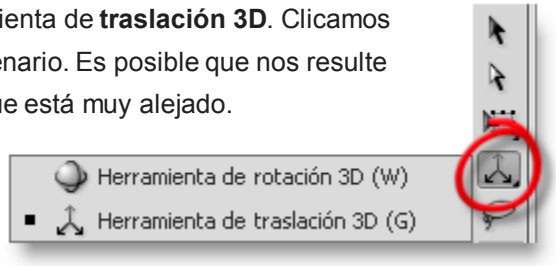
En el fotograma 500, seleccionamos la instancia del pájaro en el escenario. En el **inspector de Propiedades**, en el área de **Posición y vista 3D**, vamos a asignar un valor de **profundidad** (**Z**) de 5000. Cuanto mayor sea este valor, más parecerá alejarse el clip.

La forma de la curva se habrá suavizado considerablemente, ya que ahora el recorrido de la curva lo sigue mientras se aleja.

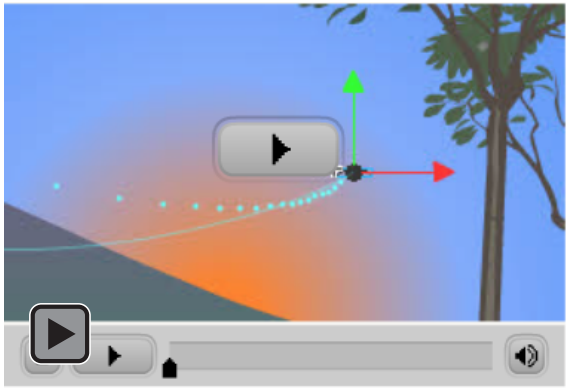
Con este nuevo valor, veremos que los puntos de la curva de interpolación se han redistribuido, apareciendo al principio de la curva más distanciados que al final de la curva. Esto significa que al principio de la animación el clip de película avanzará más rápido respecto a la curva, y que se irá ralentizando al final.

Esta desaceleración simulará perfectamente el alejamiento del clip, ya que un objeto cercano varía su posición respecto al observador en mayor medida que un objeto lejano. Por ejemplo, un avión a lo lejos parece desplazarse con mayor lentitud que si pasa junto a nosotros.

Seleccionamos la herramienta de **traslación 3D**. Clicamos sobre el pájaro en el escenario. Es posible que nos resulte difícil clicar sobre él ya que está muy alejado.



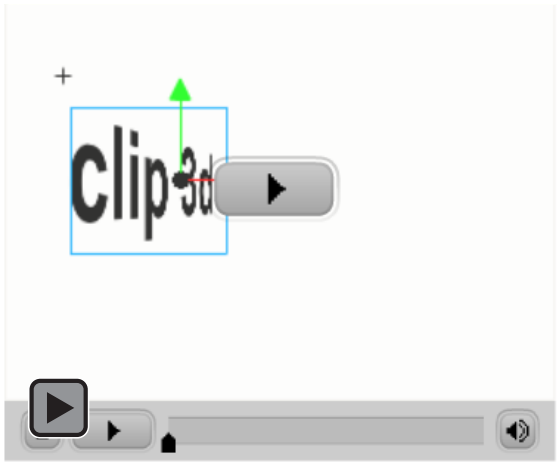
El eje Z aparecerá como un punto negro central, pero no lo vamos a desplazar ya que lo hemos definido manualmente en el inspector de propiedades. Pulsamos y arrastramos sobre las puntas de las flechas para desplazar el clip en los **ejes X e Y** hasta la posición que queramos. Tendremos que arrastrar tramos amplios para que el clip se desplace, ya que se encuentra en un punto muy lejano del eje Z.



Tutorial 5. Cinemática inversa y herramientas 3D
Paso 14 de 21

Puede parecer similar el hecho de desplazar un objeto con la herramienta **Selección**, a hacerlo con la herramienta de **Traslación 3D** en los ejes horizontal (X) y vertical (Y). Sin embargo, si el objeto a trasladar tuviera una rotación 3D, podríamos observar claramente la diferencia que supone utilizar esta herramienta.

En el vídeo de ejemplo podemos observar que al trasladar un objeto en estos ejes con la herramienta de traslación 3D, cambia su perspectiva. Si esto lo hubiéramos realizado con la herramienta Selección la perspectiva que nos muestra el clip no hubiera variado.



Vamos a asignar en el **inspector de Propiedades** un valor de **Alfa** de 0 a la instancia del clip *pájaro* en el fotograma 500.

Los valores posibles de la propiedad alfa (transparencia) oscilan entre 100 (totalmente opaco) a 0 (transparente).

Al asignar un alfa de 0 en el último fotograma de la animación de la instancia del pájaro, conseguimos que, a medida que avanza la animación, es decir, según se va alejando el pájaro, se irá haciendo cada vez más transparente, llegando a desaparecer completamente en el último fotograma.

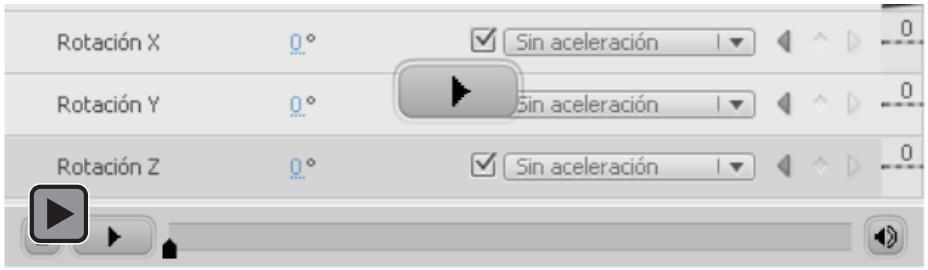
Tutorial 5. Cinemática inversa y herramientas 3D
Paso 15 de 21

Si probamos la película veremos que la animación no resulta muy natural en los primeros fotogramas, ya que parece que el pájaro vuela de lado.

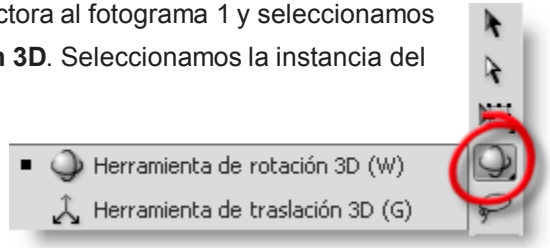
Para solucionar este problema, modificaremos la rotación 3D de la instancia en el primer fotograma. Antes de hacerlo, y para que la posición final en el fotograma 500 no se modifique, guardaremos las propiedades actuales de rotación en el último fotograma.

Para ello, en el **Editor de movimiento**, añadiremos **fotogramas clave de propiedad** para los tres tipos de **rotación 3D** en el fotograma 500.

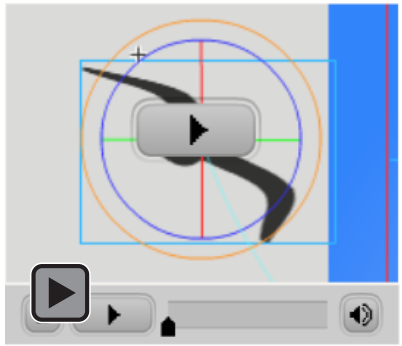
De esta forma, la animación de la instancia terminará con una rotación con valor de 0 en los tres ejes, independientemente de los cambios que hagamos posteriormente en el fotograma 1.

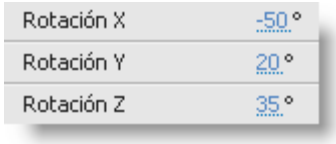


Desplazamos la cabeza lectora al fotograma 1 y seleccionamos la herramienta **de rotación 3D**. Seleccionamos la instancia del pájaro en el escenario.



Modificamos la rotación en los ejes **X** (rojo), **Y** (verde) y **Z** (azul).





También podríamos haber asignado los valores directamente en el **Editor de movimiento**.

Tutorial 5. Cinemática inversa y herramientas 3D

Paso 16 de 21

Para hacer la animación de un segundo pájaro, creamos en primer lugar una nueva capa llamada *pajaro2* por encima de la capa *pajaro1*.

Arrastramos otra instancia del clip pájaro desde la biblioteca al escenario en esta nueva capa, y lo situamos bajo el primer pájaro, fuera de los límites del escenario.

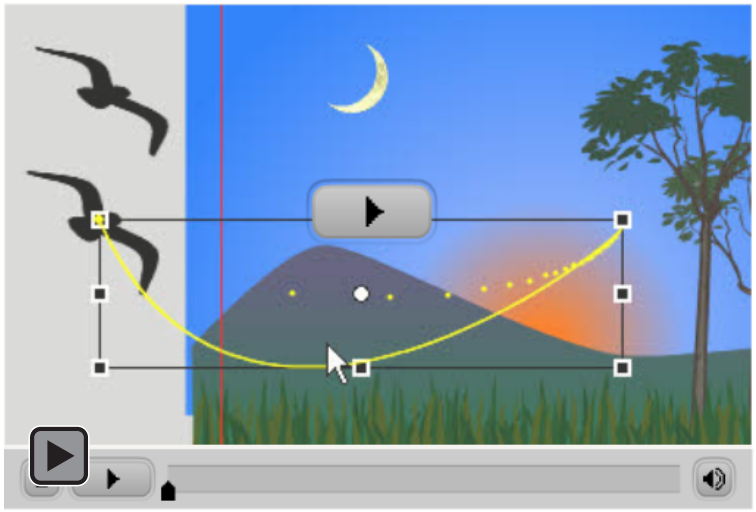
Clicamos sobre el pájaro de la capa 1 con el **botón derecho** del ratón, y seleccionamos **Copiar movimiento** en el menú contextual.

Después, clicamos sobre nuestro nuevo pájaro con el **botón derecho** del ratón y seleccionamos **Pegar movimiento** en el menú contextual.

Podemos modificar las interpolaciones de movimiento como si fueran cualquier objeto gráfico.

En este caso vamos a rotar ligeramente la interpolación del segundo pájaro, seleccionándola en el escenario con la herramienta **Transformación libre**.

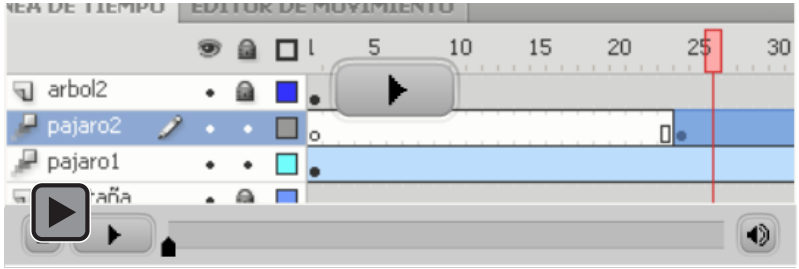
Al haber rotado la interpolación, el segundo pájaro empezará su vuelo por debajo del primer pájaro, pero terminará en una posición superior.



Tutorial 5. Cinemática inversa y herramientas 3D
Paso 17 de 21

Para que el segundo pájaro comience su animación más tarde que el primero, clicamos la interpolación en la línea de tiempo y la arrastramos a una nueva posición.

Si arrastramos la interpolación para que comience en el fotograma 24 y tenemos configurado el documento a 24 FPS, la animación de este pájaro comenzará un segundo más tarde que el anterior.



Al desplazar la animación, ahora la película se extenderá más de 500 fotogramas (524 en este caso). Debemos por tanto volver a añadir fotogramas (**F5**) en el resto de las capas.

Vamos a añadir fotogramas para todas las capas excepto las de los pájaros, extendiendo la duración hasta los 600 fotogramas.

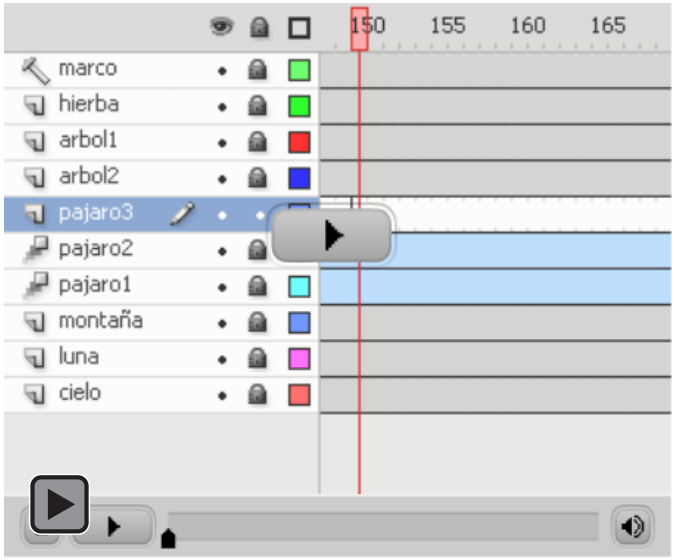
De esta forma el paisaje continuará siendo visible un tiempo después de que desaparezcan ambos pájaros.

Recordemos que para añadir fotogramas sin interpolación de movimiento, debemos clicar sobre el fotograma que queramos que sea el final de cada capa, y pulsar **F5** o bien, si pulsamos sobre él con el **botón derecho** del ratón, seleccionar **Insertar fotograma** en el menú contextual.

Tutorial 5. Cinemática inversa y herramientas 3D
Paso 18 de 21

Vamos a realizar la animación de un último pájaro. Creamos una nueva capa a la que llamaremos *pajaro3*.

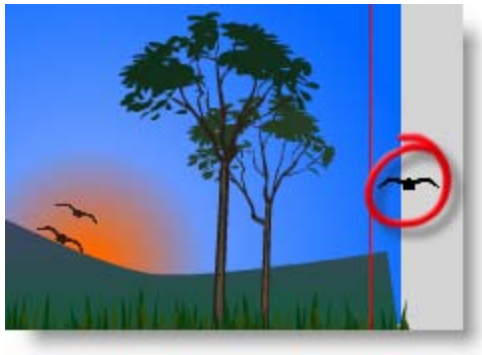
Esta vez la animación comenzará en el fotograma 150. Una forma de controlar el punto de inicio de una animación es insertando en primer lugar un fotograma clave en el fotograma en el que va a comenzar la animación. En este caso clicamos con el **botón derecho** del ratón sobre el fotograma 150, que se convertirá en el primero de la animación, y seleccionamos **Insertar fotograma clave**.



Al no tener contenido previo en esa capa, se creará un fotograma clave vacío.

Arrastramos una instancia del clip pájaro de la biblioteca al escenario, y veremos que se posiciona en el fotograma clave que hemos creado, es decir, en el fotograma 150. El contenido siempre se posicionará en el último fotograma clave que encuentre para esa capa y, de no haberlo, se posicionará en el fotograma 1.

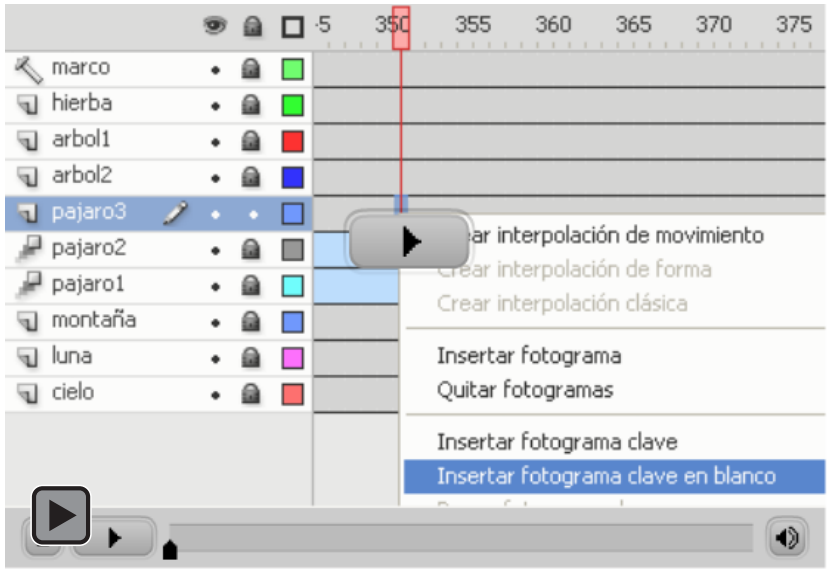
En el **inspector de Propiedades,** en el área de **Posición y vista 3D**, le asignamos un valor de **profundidad (Z)** de 1000. Después, con ayuda de la herramienta de **traslación 3D**, lo posicionamos en la parte derecha fuera de los límites del escenario, en una altura inferior a la copa de los árboles.



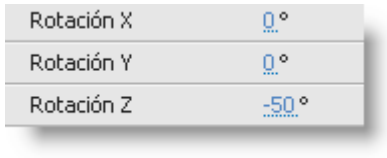
Tutorial 5. Cinemática inversa y herramientas 3D
Paso 19 de 21

Si ahora hiciéramos una interpolación de movimiento, ésta se extendería hasta el fotograma 600. En nuestro caso vamos a hacer que la animación termine en el fotograma 350.

Para ello, antes de crear la interpolación clicamos con el **botón derecho** del ratón en el fotograma 351, y seleccionamos **Insertar fotograma clave en blanco**. Después clicamos con el **botón derecho** sobre cualquier punto de la capa donde se encuentra el clip (en gris), y seleccionamos **Crear interpolación de movimiento**.



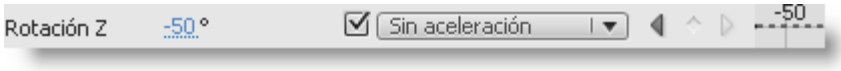
Volviendo al fotograma 150, es decir, al inicio de la animación de este pájaro, asignamos una **rotación Z** de -50 en el **editor de movimiento**



Por último, de nuevo en el fotograma 350, asignamos en el **inspector de Propiedades**, en el área de **Posición y vista 3D**, los valores de **X:-100**, **Y:100** y **Z:-100**

De esta forma el pájaro volará acercándose hasta desaparecer por la esquina superior izquierda. Como podemos ver, los valores negativos de **Z** hacen que un clip aparezca más cerca.

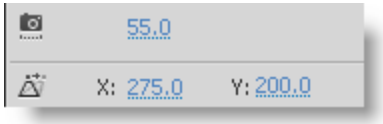
Si observamos el editor de movimiento, vemos que la línea de la propiedad de rotación Z aparece punteada, lo que significa que no tiene interpolación, y que se mantiene la misma rotación en toda la animación.



Tutorial 5. Cinemática inversa y herramientas 3D
Paso 20 de 21

Hay varias opciones interesantes relacionadas con las interpolaciones que no hemos tratado en este tutorial, pero que es interesante conocer. Vamos a hacer un breve resumen sobre algunas de las opciones sobre las que recomendamos experimentar.

Para comenzar, en el área de **Posición y vista 3D** que aparece en el **inspector de Propiedades** cuando seleccionamos un clip de película, existen también otras áreas que no hemos utilizado.

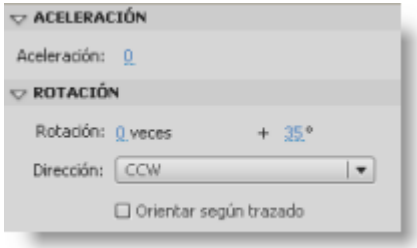


En primer lugar tenemos el **Ángulo de perspectiva**. Esta propiedad define el ángulo de visión de forma similar a como podría hacerlo un zoom fotográfico. El valor del ángulo puede oscilar entre 1 y 180. Los valores más altos producen un escenario con mayor profundidad.

Tenemos también la posición X e Y del **Punto de desvanecimiento**. Esta posición se refiere a la orientación del eje Z, que por defecto es el centro del escenario.

Si modificamos el ángulo de perspectiva o el punto de desvanecimiento, estos cambios afectarán a todos los clips 3D de nuestra película.

Por otro lado, cuando seleccionamos una interpolación, podemos modificar su **aceleración** y **rotación** en el **inspector de Propiedades**.



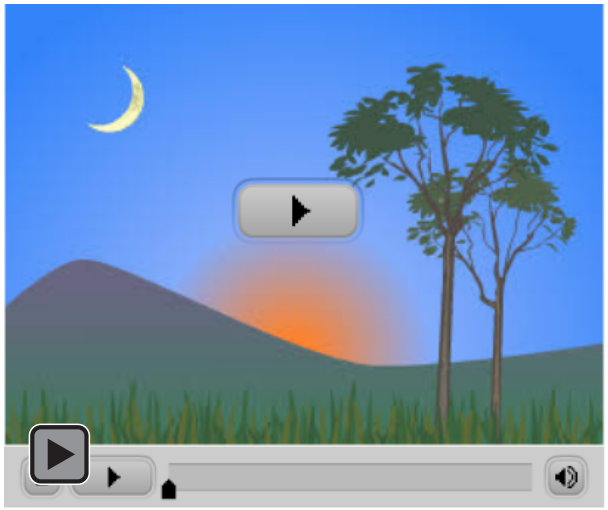
Otra opción muy interesante es el nuevo panel que se encuentra en **Ventana > Configuración predefinida de movimiento**.

Con este panel podemos asignar fácilmente un movimiento predefinido a un clip, pudiendo después modificar la interpolación creada de la misma forma que si la hubiéramos creado nosotros.

También permite guardar nuestras propias interpolaciones para poder reutilizarlas con otros clips.

Tutorial 5. Cinemática inversa y herramientas 3D

Paso 21 de 21



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Cambiad el punto de desvanecimiento y el ángulo de perspectiva.
2. Añadid un fotograma clave de propiedad en un punto intermedio de una interpolación, y modifica las propiedades de rotación en ese punto. Podéis utilizar para ello el editor de movimiento.
3. Añadid nuevas poses en el clip *ala*, añadiendo un nuevo ciclo del batir de las alas que no sea exactamente igual al primero.

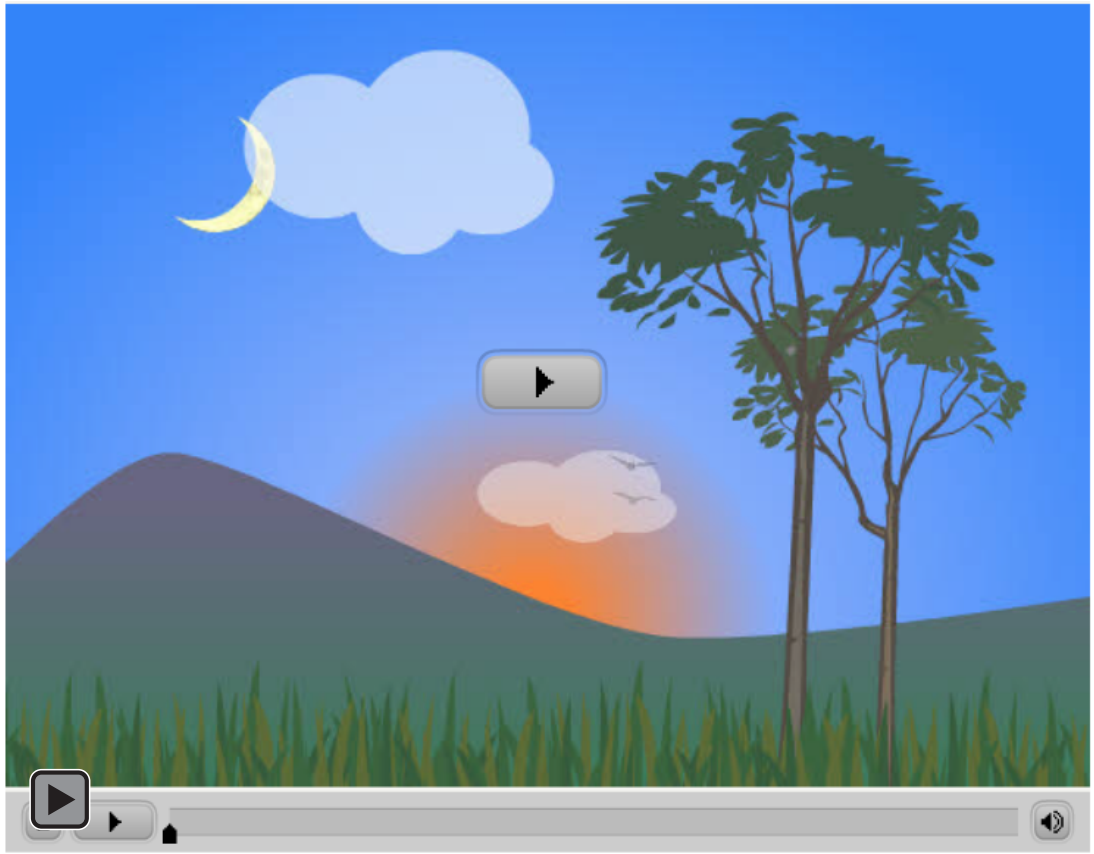
Tutorial 6. Animación con ActionScript 3.0

Paso 1 de 19

En este tutorial vamos a hacer una primera aproximación a la programación con ActionScript 3.0, la última versión del lenguaje de programación de Flash.

Con este lenguaje de programación podemos añadir interactividad a nuestras películas, controlar el aspecto o movimiento de objetos en el escenario, crear aplicaciones complejas, etc.

En primer lugar vamos a aprender a controlar los clips que se encuentran en el escenario mediante programación. En este caso controlaremos un clip realizado a partir de la nube que creamos en el tutorial 1.



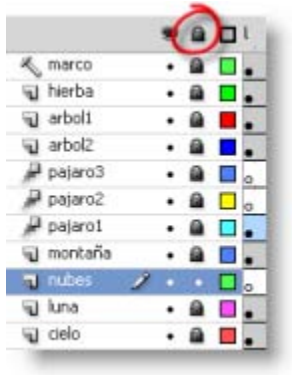
Tutorial 6. Animación con ActionScript 3.0
Paso 2 de 19

Abrimos el documento *tutorial1.fla*. Hacemos clic sobre la nube con el **botón derecho** del ratón y seleccionamos **Copiar**. Si no podemos seleccionarla será posiblemente porque tenemos bloqueada la capa nube en la línea de tiempo, así que tendremos que desbloquearla previamente.

Una vez copiada, ya podemos cerrar el documento sin necesidad de guardar los cambios. Abrimos el archivo *tutorial5.fla*, que contiene la animación de los pájaros, y lo guardamos como *tutorial6.fla.*

Creamos una nueva capa por detrás de la montaña y le damos el nombre *nubes*.

Bloqueamos el resto de las capas para evitar posibles modificaciones accidentales. La forma más rápida es clicar sobre el candado que está sobre todas las capas, y después desbloquear la capa que nos interesa.

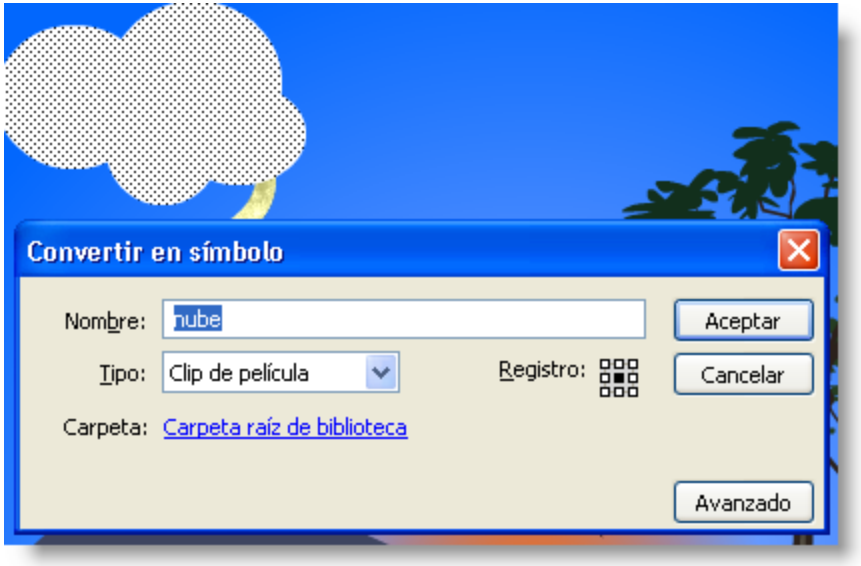


Con la capa nubes activa, seleccionamos cualquier punto del escenario con el **botón derecho** del ratón y seleccionamos **Pegar**.

Situamos la nube en la esquina superior izquierda del escenario.

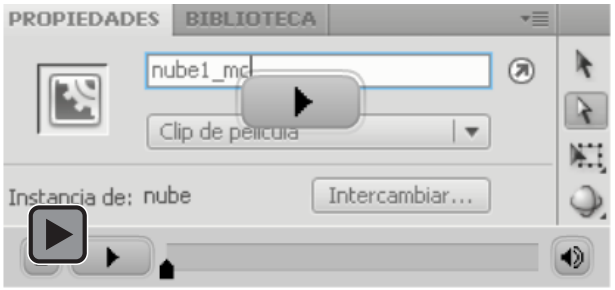
Tutorial 6. Animación con ActionScript 3.0
Paso 3 de 19

Con la nube seleccionada, pulsamos **F8** o **Modificar > Convertir en símbolo.** Le damos el nombre *nube* y como tipo seleccionamos **Clip de película**.



Ahora tendremos un clip de película llamado nube en la biblioteca, mientras que en el escenario tendremos una instancia de ese clip de película, tal y como nos indicará la parte superior del **inspector de Propiedades**.

Para poder controlar una instancia con ActionScript, lo primero que tenemos que hacer es darle un nombre de instancia en el inspector de Propiedades. Le vamos a dar el nombre *nube1_mc*.



Es conveniente que los nombres de instancia siempre comiencen por una letra minúscula y que terminen de la siguiente manera:

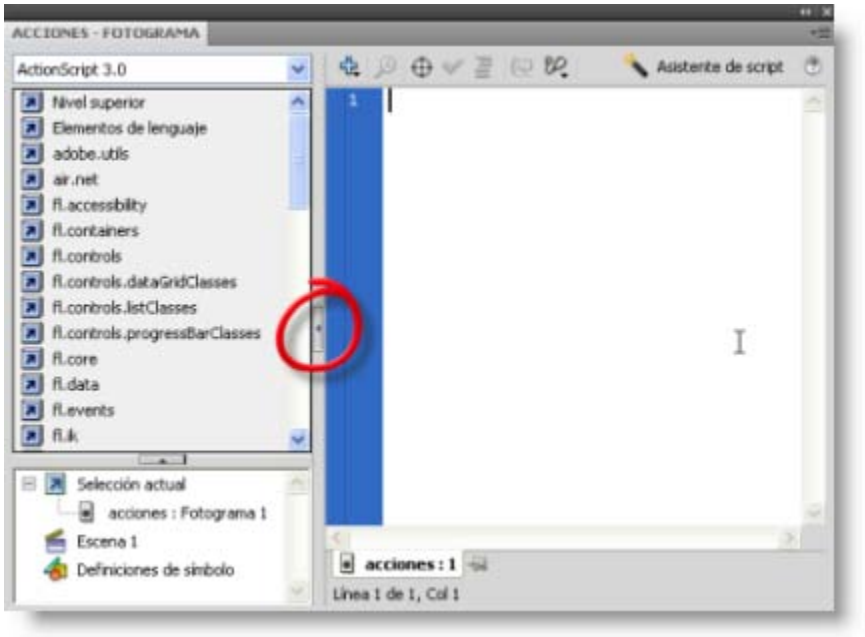
- *_mc* para clips de película (*movie clip*)
- *_btn* para botones
- *_txt* para campos de texto

La razón para ello es que cuando escribamos código sobre una instancia con una de estas terminaciones, Flash detectará el tipo de objeto del que se trata, ayudando con las sugerencias de código adecuadas a ese tipo de objeto.

Tutorial 6. Animación con ActionScript 3.0
Paso 4 de 19

Vamos a escribir las acciones en una **nueva capa** que situaremos por encima de todas las demás, y a la que llamaremos *acciones*. Es conveniente tener todas las acciones separadas en una capa para facilitar la organización del documento.

Abrimos el **panel Acciones** pulsando **F9** o **Ventana > Acciones**.



En la columna izquierda de este panel tenemos en primer lugar un acceso exhaustivo a las diferentes clases y funciones disponibles. En nuestro caso no vamos a utilizar esta forma de introducir código.

En la parte inferior de la misma columna tenemos información sobre la selección actual (en nuestro caso el fotograma 1 de la capa *acciones*). Cuando tengamos acciones distribuidas en más fotogramas, desde aquí tendremos un acceso cómodo y directo que nos permitirá acceder a la programación de los diferentes fotogramas.

En la parte central tenemos el editor, que es el área sobre la que escribiremos el código. Sobre esta área tenemos algunas herramientas para comprobar el código, insertar comentarios, acceder a la ayuda, etc. Recomendamos tener desactivado el asistente de script.

Para disponer de más espacio para escribir el código, recomendamos colapsar la columna izquierda clicando sobre la pequeña flecha que separa ambas columnas. Podremos expandirla de nuevo cuando lo necesitemos.

Tutorial 6. Animación con ActionScript 3.0
Paso 5 de 19

Vamos a comenzar asignando un valor de *.6* a la propiedad *alpha*. Este valor significa una opacidad del 60%. Este valor puede oscilar entre 0 (transparente) y 1 (opaco).

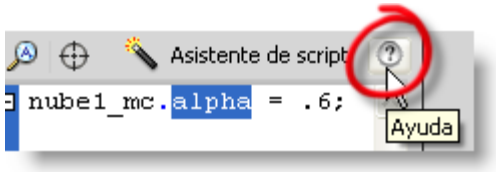
Comenzamos escribiendo el nombre de la instancia seguido por un punto. Al escribir el punto, aparecerá automáticamente un desplegable con diferentes métodos y propiedades que podemos asignar a un clip de película. Esta ayuda no aparecería si no hubiéramos puesto *_mc* como final del nombre de la instancia.

Tras el punto se escribe el método que queremos ejecutar o la propiedad que queremos asignar. En este caso hemos seleccionado la propiedad *alpha*. Las palabras clave del lenguaje aparecen en color azul.

Después, tras un signo =, escribimos el valor para esa propiedad. En este caso hemos escrito *.6* (también podríamos haber escrito *0.6*). Los espacios antes y después del signo = no son necesarios, pero mejoran la legibilidad del código. Finalizamos cada sentencia con un *punto y coma ;*.

Si tenemos problemas con el tamaño del texto del código, podemos abrir las **preferencias (Ctrl+U)** y, en la categoría de **ActionScript**, cambiar el tipo de texto, el tamaño de la letra o los colores. Es preferible mantener el coloreado del código, ya que nos ayuda a distinguir palabras clave, comentarios, etc.

Una forma rápida de acceder a la ayuda sobre alguna palabra clave es seleccionarla directamente en el editor y clicar sobre el icono de la ayuda. En la página de ayuda podremos ver definiciones y ejemplos de uso de cada palabra clave.



Tutorial 6. Animación con ActionScript 3.0
Paso 6 de 19

Si contraemos el **panel Acciones** pulsando sobre la barra gris superior del panel para ver el escenario, no observamos ningún cambio en la nube. Esto se debe a que la programación se ejecuta cuando probamos o publicamos la película (tiempo de ejecución), y no directamente en el escenario de edición (tiempo de edición).

Para comprobar cómo ha cambiado el alfa de la nube tendremos que probar la película con **Control > Probar película (Ctrl+Intro)**.

Si hemos asignado un valor para una propiedad en el código, este valor prevalecerá sobre lo que hayamos definido en tiempo de edición. Por ejemplo, en el caso del alfa, podemos asignar diferentes valores en el inspector de propiedades, pero cuando probemos la película siempre veremos la nube con el 60% del alfa, ya que es el valor que hemos introducido en el código.

De la misma forma, si asignamos en el código valores para otras propiedades, como por ejemplo las posiciones X e Y del clip, éste aparecerá en las coordenadas que especifiquemos en la programación, y no en las coordenadas en que lo hayamos situado en el escenario.

Para añadir **interactividad** a una película, lo más probable es que necesitemos **detectar** cuándo ocurre un determinado **evento** (que se haya clicado sobre un botón, que se haya pulsado una tecla, etc.), y ejecutar después una **acción** en base a ello.

Para ello tenemos que agregar un ***listener***, es decir, añadir a un objeto la capacidad de detectar si algo ha ocurrido, y que llame a una función cuando ello ocurra.

La forma general de crear un *listener* es la siguiente:

```
nombreClip.addEventListener(nombreEvento,nombreFuncion);
```

En nuestro caso vamos a hacer que nuestra nube se arrastre cuando estemos pulsando sobre ella.

Para ello añadiremos la siguiente línea al código:

```
nube1_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
```

Con este código hemos añadido a nuestra instancia *nube1_mc*, un detector para un evento del ratón (MouseEvent). En concreto, el evento del ratón que queremos que detecte es si el usuario ha pulsado sobre ella (MOUSE_DOWN). Si es así, ejecutará una función a la que hemos llamado *arrastrar*, y que definiremos más adelante.

Tutorial 6. Animación con ActionScript 3.0
Paso 7 de 19

Un evento de ratón similar es *MouseEvent.CLICK*. La diferencia es que con *CLICK* se detecta si se ha pulsado y soltado el ratón, es decir, si se ha hecho clic. Más adelante veremos que en nuestro caso vamos a ejecutar acciones diferentes para las acciones de pulsar y soltar el ratón, y es por ello por lo que hemos elegido el evento *MOUSE_DOWN* (y después utilizaremos *MOUSE_UP*).

Ahora pasaremos a definir nuestra función *arrastrar*. Una **función** es un bloque de instrucciones agrupadas, y que se ejecuta al ser llamada desde otra función o método. Al escribir el código, las funciones y métodos se diferencian de las propiedades porque tras el nombre de los primeros hay un paréntesis. El paréntesis puede estar vacío, o bien puede recibir uno o varios parámetros separados por comas. Por ejemplo, *alpha* es una propiedad, mientras que *addEventListener* es un método con dos parámetros (evento y función).

Para crear una función tenemos que comenzar con la palabra clave *function*. Tras ella, escribimos el nombre que queremos dar a nuestra función. En este caso le damos el nombre *arrastrar*. Tras el nombre, escribimos los paréntesis:

```
function arrastrar()
```

Las funciones que vayan a ser llamadas desde un *addEventListener*, como es nuestro caso, reciben un parámetro. Este parámetro es un objeto del tipo de evento que ha desencadenado la función, en este caso *MouseEvent*. Entre los paréntesis escribiremos el nombre del objeto (el nombre que queramos), dos puntos (:), y el tipo de objeto.

Por lo tanto, de momento nuestra función comenzará asi:

```
function arrastrar(e:MouseEvent)
```

Le hemos dado al parámetro el nombre *e* por *evento*, pero podríamos haberle dado otro nombre. Generalmente se suele dar a este parámetro el nombre *e* o *event*.

Tras el nombre de la función es conveniente especificar el tipo de datos que devolverá ésta. En nuestro caso, la función no va a devolver ningún resultado, sino que simplemente va a ejecutar una acción. En estos casos, como tipo de datos se asigna *void* (vacío).

Nuestra función queda como sigue:

```
function arrastrar(e:MouseEvent):void
```

Tutorial 6. Animación con ActionScript 3.0
Paso 8 de 19

Ahora nos queda especificar las acciones que queremos que ejecute esta función. Todo el contenido de una función se escribe entre llaves, así que para asegurarnos de que no se nos olvidan, es conveniente escribir primero las llaves, y pasar después a escribir entre ellas las distintas instrucciones.

```
function arrastrar(e:MouseEvent):void
{
   //aquí irá nuestro código
}
```

Las dos barras que hemos añadido en el código anterior antes de la frase “aquí irá nuestro código”, indican que se trata de un comentario de una sola línea. Podemos añadir los comentarios que queramos en nuestro código, ya que no se ejecutarán. Para comentarios de varias líneas utilizaremos /* y */ para delimitar el texto que queremos que sea un comentario.

Los comentarios pueden ser muy útiles no sólo para añadir información, sino también para hacer que algunas líneas del código no se ejecuten temporalmente, lo cual nos ayudará a comprobar el funcionamiento de nuestro código.

Probemos a escribir el código siguiente:

```
function arrastrar(e:MouseEvent):void
{
   trace("Estoy pulsando sobre la nube");
}
```

La función *trace* mostrará en un panel llamado **Salida** lo que se encuentre dentro del paréntesis. Si escribimos entre comillas el contenido del trace, en el panel Salida se mostrará exactamente la frase que hayamos escrito. Las comillas permiten escribir cadenas de caracteres, y las veremos en color verde en el panel Acciones.

Los datos que aparezcan en este panel no los verá el usuario final, pero sí cuando probemos la película. Si aparece esa frase quiere decir que esa línea de código se está ejecutando correctamente.

Recapitulando: hemos añadido a la nube un detector del evento de pulsar sobre ella con el ratón, y cuando ese evento sea detectado, se ejecutará la función *arrastrar*, que tiene la instrucción de mostrar en el panel Salida la frase *Estoy pulsando sobre la nube*. Probamos película con **Control > Probar película** y clicamos sobre la nube para comprobar el funcionamiento de nuestro código.

Tutorial 6. Animación con ActionScript 3.0
Paso 9 de 19

Ahora que hemos comprobado que nuestro código funciona correctamente, sustituimos la función *trace* por la acción que realmente queremos que se realice, que es que la nube se arrastre.

```
function arrastrar(e:MouseEvent):void
{
    nube1_mc.startDrag();
}
```

Si probamos de nuevo nuestra película, podremos ver que al pulsar sobre la nube, ésta comienza a arrastrarse junto con el ratón, y no deja de arrastrarse en ningún momento.

Para que cese el arrastre, tendremos que introducir una nueva función que permita que cuando se suelte el botón del ratón, la nube deje de arrastrarse.

El funcionamiento será el mismo que el anterior. Tendremos que crear otro detector para el evento de soltar el ratón, y asignarle una función que detenga el arrastre de la nube.

Por legibilidad del código podemos juntar por un lado las líneas con la creación de los *listeners*, y por otro lado las funciones.

La forma más rápida para escribir este nuevo código es copiar y pegar el que ya habíamos creado, ya que es en muchos aspectos similar. Sustituiremos el *MOUSE_DOWN* por *MOUSE_UP*, y el nombre de la función nueva será *soltar* en vez de *arrastrar*. Por último, la función que detiene un arrastre activo es *stopDrag*().

El código que hemos creado hasta ahora en el primer fotograma de la capa *acciones* es el siguiente:

```
nube1_mc.alpha = .6;

nube1_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube1_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);

function arrastrar(e:MouseEvent):void
{
     nube1_mc.startDrag();
}

function soltar(e:MouseEvent):void
{
     nube1_mc.stopDrag();
}
```

Tutorial 6. Animación con ActionScript 3.0
Paso 10 de 19

Además del arrastre interactivo de la nube, vamos a añadir un desplazamiento continuo de la nube por el escenario mediante una función a la que llamaremos *desplazar*.

Existe un tipo de evento llamado *ENTER_FRAME*, que pertenece a una categoría genérica de eventos llamada *Event,* que se ejecuta cada vez que la cabeza lectora se desplaza un fotograma. Si la cabeza lectora está detenida, pero el objeto al que se asocia está en el escenario (esté o no visible), también se ejecutará con la misma frecuencia que los fotogramas por segundo que tengamos definidos.

En el caso de nuestra película, una función que se asocie a un *ENTER_FRAM*E se ejecutará cada vez que avance la cabeza lectora, es decir, 24 veces por segundo. Cambiaremos ligeramente la posición de la nube cada 1/24 de segundo, por lo que mostrará una animación fluida.

Agregamos este nuevo *listener* bajo los otros dos que habíamos creado. Para ello escribimos este código:

```
nube1_mc.addEventListener(Event.ENTER_FRAME,desplazar);
```

A falta de definir las instrucciones que ejecutará la función *desplazar*, su definición quedará como sigue:

```
function desplazar(e:Event):void
{
}
```

Como podemos ver, el evento *ENTER_FRAME* pertenece a una categoría llamada *Event,* y no a eventos de ratón *(MouseEvent)*. Este *Event* aparecerá tanto en la creación del listener como en el parámetro de la función.

Para que la nube avance hacia la derecha tendremos que variar su posición x poco a poco dentro de la función *desplazar*. Por ejemplo, podemos especificar que la posición x de la nube aumente de uno en uno su valor, lo que traducido a código sería:

```
nube1_mc.x = nube1_mc.x + 1;
```

o bien, más sencillo:

```
nube1_mc.x += 1;
```

que significa sumar su valor más el número que se encuentra tras el =.

Tutorial 6. Animación con ActionScript 3.0
Paso 11 de 19

De momento tendremos la función desplazar como sigue:

```
function desplazar(e:Event):void
{
     nube1_mc.x += 1;
}
```

Probamos la película para comprobar cómo se desplaza la nube hacia la derecha, mientras que sigue siendo arrastrable. Podemos probar con distintos valores hasta encontrar una velocidad que nos parezca adecuada (por ejemplo 0.4).

Si mantenemos pulsada la nube para arrastrarla pero no soltamos el ratón, la nube continuará avanzando hacia la derecha fuera del ratón. Aunque después soltemos el ratón, el evento MOUSE_UP ya no se producirá, ya que significa levantar el botón del ratón encima de la nube (no fuera de ella). De esta forma ya no podremos soltar el arrastre de la nube.

Para solucionarlo podemos asociar a la función soltar otro evento de ratón llamado *ROLL_OUT*, que significa estar pulsando sobre la nube pero que el ratón se arrastre fuera de ella. Podemos crear el nuevo evento bajo el evento MOUSE_UP.

Por ahora los *listeners* que hemos creado serán los siguientes:

```
nube1_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube1_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);
nube1_mc.addEventListener(MouseEvent.ROLL_OUT,soltar);
nube1_mc.addEventListener(Event.ENTER_FRAME,desplazar);
```

En este caso no necesitaremos crear una nueva función, ya que este *listener* llamará a la misma función *soltar* que habíamos creado previamente.

Si probamos la película veremos que si en algún momento dejamos de arrastrar la nube, ésta terminará desapareciendo por la parte derecha del escenario. Aunque deje de visualizarse, la instancia seguirá ejecutando el código.

Lo siguiente que vamos a programar va a ser que cuando haya desaparecido por la parte derecha, vuelva a aparecer por la parte izquierda del escenario.

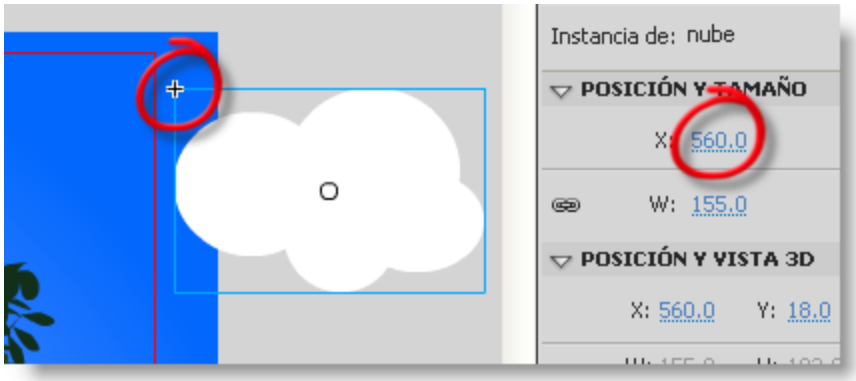
Para ello vamos a introducir una sentencia condicional dentro de la función desplazar, de tal forma que si se cumplen unos determinados requisitos, la nube se posicione en la parte izquierda del escenario.

Tutorial 6. Animación con ActionScript 3.0
Paso 12 de 19

En nuestro caso, vamos a definir que si la posición x de la nube alcanza determinado valor (cuando haya desaparecido por la derecha), entonces que vuelva a la parte izquierda del escenario.

Una forma rápida de hacer el cálculo de las posiciones de inicio y de fin es colocar la nube en un extremo y otro del escenario, y anotar el valor de x que aparece en el inspector de Propiedades.

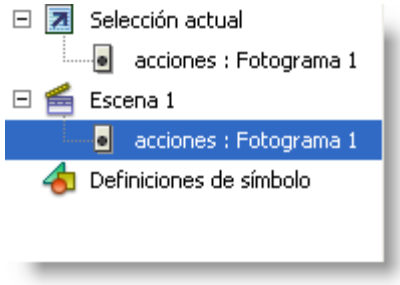
La posición del clip la marca una pequeña cruz. Esta posición depende del punto de registro que hayamos seleccionado al crear el clip (por defecto es la parte superior izquierda del clip).



En este caso, vamos a seleccionar como extremo derecho 560 (algo más que la anchura del escenario), y como extremo izquierdo -160 (una posición en la que todavía no se ve la nube). Estos valores dependerán del tamaño de vuestra nube.

Podríamos hacer estos cálculos utilizando como datos la anchura del clip y la anchura del escenario. El motivo de no hacerlo de esta manera es que en próximos pasos añadiremos otra nube a la que programaremos profundidad en el eje z, y entonces la posición x del objeto ya no será equivalente a los píxeles del escenario, ya que se moverá en un escenario que simulará más profundidad, y por tanto más anchura que los 550 píxeles de nuestro escenario.

Colocamos la nube en el lugar en el que queramos que comience su animación y volvemos al **panel Acciones (F9)**. Recordemos que podemos navegar por las distintas acciones de nuestra película pulsando sobre la línea correspondiente en la parte inferior izquierda del panel.



En este caso, si no tuviéramos seleccionado el fotograma 1 de la capa acciones en la línea de tiempo, podríamos ir a él pulsando acciones: Fotograma 1

Tutorial 6. Animación con ActionScript 3.0
Paso 13 de 19

La sentencia if, que es la que vamos a utilizar, evalúa si se cumple una condición, y en caso de cumplirse, ejecuta las instrucciones que indiquemos entre las llaves. La estructura es la siguiente:

```
if (condicion)
{
     //instrucciones
}
```

En la sentencia, por tanto, tendremos que especificar que si la posición x actual de la nube es mayor que 560 (el signo > significa mayor que), entonces que la posición pase a ser -160.

La función *desplazar* quedará de esta manera:

```
function desplazar(e:Event):void
{
     nube1_mc.x += .4;
     if (nube1_mc.x > 560)
          {
               nube1_mc.x = -160;
          }
}
```

Vemos que la función tiene sus propias llaves, y que dentro de ella también hay un condicional con sus propias llaves de inicio y de fin.

Probamos ahora la película con **Control > Probar película**. Vemos que al poco de desaparecer por el extremo derecho, aparece de nuevo por el extremo izquierdo.

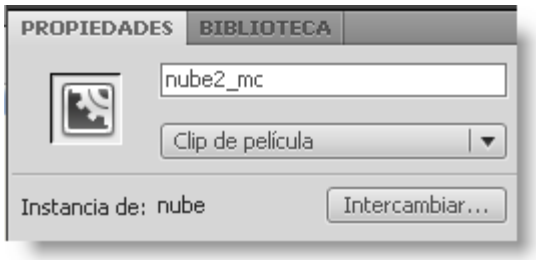
A diferencia de la línea de tiempo principal, que tiene una extensión de 600 fotogramas, vemos que la animación del movimiento de la nube es independiente de esta extensión. En cada repetición del vuelo de los pájaros, la nube puede encontrarse en un punto diferente.

Para animar mediante programación es suficiente con un solo fotograma en la línea de tiempo. Si hubiéramos hecho esta animación en la línea de tiempo, hubiéramos necesitado extender mucho más el número de fotogramas.

Tutorial 6. Animación con ActionScript 3.0

Paso 14 de 19

Arrastramos otra instancia del clip *nube* al escenario en la misma capa *nubes* que la nube anterior, y en el **inspector de Propiedades** le damos el nombre de instancia *nube2_mc*.



Al principio de la programación, tras la línea en la que habíamos definido el alfa de la primera instancia, añadimos algunas propiedades para la nueva instancia.

```
nube1_mc.alpha = .6;
nube2_mc.alpha = .4; //más transparente que la primera nube
nube2_mc.scaleY = .7;  //para disminuir verticalmente (70%)
nube2_mc.z = 300;  //para añadir profundidad
```

Después añadimos los mismos *listener* para esta segunda instancia. Para ello basta con copiar y pegar los *listeners* ya creados y sustituir *nube1_mc* por *nube2_mc*:

```
nube1_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube1_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);
nube1_mc.addEventListener(MouseEvent.ROLL_OUT,soltar);
nube1_mc.addEventListener(Event.ENTER_FRAME,desplazar);

nube2_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube2_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);
nube2_mc.addEventListener(MouseEvent.ROLL_OUT,soltar);
nube2_mc.addEventListener(Event.ENTER_FRAME,desplazar);
```

Dentro de cada función hacíamos también referencia a la instancia *nube1_mc*. En vez de repetir las líneas con la programación para nube2_mc dentro de las funciones, lo que vamos a hacer es que la programación haga referencia al objeto que inició el evento.

Es decir, si la acción arrastrar ha sido llamada por un listener de la *nube1_mc*, arrastraremos esa nube, pero si la acción ha sido iniciada por la *nube2_mc*, será esta nube la que arrastremos.

Tutorial 6. Animación con ActionScript 3.0
Paso 15 de 19

Para ello tenemos que escribir en primer lugar el nombre del parámetro del evento que hemos puesto en la función (*e* en nuestro caso). Después añadimos la propiedad *target*. Con esto se hará referencia al objeto que disparó la acción al recibir el evento.

Por lo tanto, si sustituimos dentro de las funciones el nombre de la instancia por *e.target*, conseguiremos tener una referencia directa a los objetos que han enviado la función.

Para comprobarlo en la función *arrastrar*, además de sustituir *nube1_mc* por *e.target*, añadiremos un *trace* que nos muestre el nombre (*name*) de la instancia que desencadena el evento.

```
function arrastrar(e:MouseEvent):void
{
     e.target.startDrag();
     trace(e.target.name);
}
```

Al no estar el contenido del *trace* entre comillas, la función no escribe el valor literal que hemos escrito entre paréntesis, sino su valor (en este caso es un nombre).

De esta manera, veremos que al pulsar sobre cada nube el panel de salida nos muestra su nombre.

Añadimos *e.target* a todas las funciones. Ahora tenemos un problema con las posiciones que habíamos determinado en el condicional *if*, ya que al tener la segunda nube más profundidad deberíamos distanciar ambos extremos.

Una forma de saber aproximadamente estos valores es cambiar el *trace* de la función arrastrar para que nos muestre la posición x del clip sobre el que clicemos:

```
trace(e.target.x);
```

Después desactivamos temporalmente el *if* de la función *desplazar* convirtiéndolo en un comentario añadiendo */** antes del *if* y ***/** después de la llave de cierre del *if*.

```
/*if (e.target.x > 560)
{
     e.target.x = -160;
}*/
```

Tutorial 6. Animación con ActionScript 3.0
Paso 16 de 19

Ahora probamos la película y tratamos de pulsar sobre la segunda nube en posiciones muy cercanas al límite del escenario por ambos lados para hacernos una idea de los valores de x adecuados.

El **panel Salida** mostrará el valor x para esta nube, que como podemos ver, ha variado considerablemente al estar programada para mostrarse en un plano más profundo.

De hecho, esa diferencia entre la propiedad x dependiendo de la profundidad la podemos observar también en la animación, ya que aunque ambas nubes tienen asignado el mismo incremento en el valor x, la más lejana parece avanzar más despacio que la más cercana.

En nuestro caso, como valor mínimo seleccionaremos -315 y como valor máximo 715, pero esos valores dependerán del tamaño de vuestra nube.

Ahora la primera nube tardará más en aparecer. Podríamos añadir un condicional, y dependiendo de si el target es la primera nube o la segunda, tomar unos límites u otros. En este caso no tiene importancia que una nube tarde más en aparecer de nuevo, así que dejamos como límites los de la nube más lejana.

Después de borrar el *trace*, pues ya no lo necesitamos, las funciones quedarán así:

```
function arrastrar(e:MouseEvent):void
{
     e.target.startDrag();
}

function soltar(e:MouseEvent):void
{
     e.target.stopDrag();
}

function desplazar(e:Event):void
{
     e.target.x += .4;
     if (e.target.x > 715)
     {
          e.target.x = -315;
     }
}
```

Tutorial 6. Animación con ActionScript 3.0
Paso 17 de 19

Como último detalle, vamos a hacer que se muestre un cursor con forma de mano cuando estemos sobre la nube, para que parezca un botón y sea más fácil intuir que la nube se puede arrastrar.

Para ello, antes de los *listener* añadimos las siguientes propiedades:

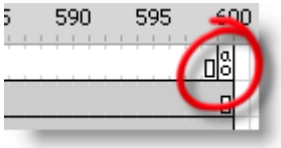
```
nube1_mc.buttonMode = true;
nube2_mc.buttonMode = true;
```

Con esto ya veremos que el cursor adopta la forma de una mano cuando estamos encima de una nube, ya que trata visualmente a las nubes como si fueran un botón.

Analicemos lo que ocurre con nuestra película. Comienza en el fotograma 1 y Flash lee toda la programación que hemos escrito y la va ejecutando. Después la línea de tiempo continua durante 600 fotogramas, y de vuelta al fotograma 1, donde vuelve a leer toda la programación.

Para no sobrecargar a Flash con información repetitiva, podemos hacer que el bucle de la línea de tiempo principal vaya del fotograma 600 al fotograma 2, sin pasar de nuevo por el fotograma 1, que es donde hemos escrito toda nuestra programación.

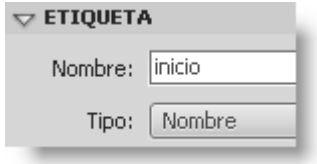
Para ello creamos un fotograma clave en el fotograma 600 de la capa *acciones*, y escribimos la siguiente programación:




```
gotoAndPlay(2);
```

Esto significa que cuando la cabeza lectora llegue al fotograma 600, donde está escrita la programación, irá al fotograma 2 y continuará con la reproducción.

Si hubiéramos creado un fotograma clave y le hubiéramos asignado en el inspector de propiedades el nombre *inicio*, también podríamos haber ido a ese fotograma escribiendo:




```
gotoAndPlay("inicio");
```

Este paso no es necesario en este caso, pero puede resultar útil conocer esta opción para crear fácilmente menús de navegación.

Tutorial 6. Animación con ActionScript 3.0

Paso 18 de 19

Este es el aspecto completo que tendrá la programación completa de este tutorial:

```
nube1_mc.alpha = .6;
nube2_mc.alpha = .4;
nube2_mc.scaleY = .7;
nube2_mc.z = 300;
nube1_mc.buttonMode = true;
nube2_mc.buttonMode = true;

nube1_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube1_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);
nube1_mc.addEventListener(MouseEvent.ROLL_OUT,soltar);
nube1_mc.addEventListener(Event.ENTER_FRAME,desplazar);

nube2_mc.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
nube2_mc.addEventListener(MouseEvent.MOUSE_UP,soltar);
nube2_mc.addEventListener(MouseEvent.ROLL_OUT,soltar);
nube2_mc.addEventListener(Event.ENTER_FRAME,desplazar);
```

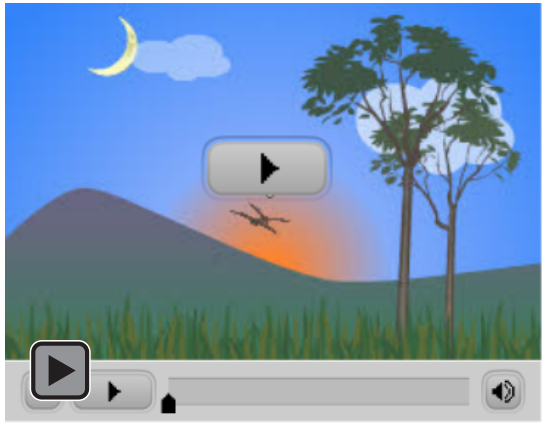
```
function arrastrar(e:MouseEvent):void
{
     e.target.startDrag();
}

function soltar(e:MouseEvent):void
{
     e.target.stopDrag();
}

function desplazar(e:Event):void
{
     e.target.x += .4;
     if (e.target.x > 715)
     {
          e.target.x = -315;
     }
}
```

Tutorial 6. Animación con ActionScript 3.0

Paso 19 de 19



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Cambiad la programación para que las nubes se desplacen de derecha a izquierda.
2. Añadid una nueva nube en el escenario que también se desplace, pero que no se pueda arrastrar.
3. Haced que las nubes se vuelvan más transparentes al desplazarse, pero que al aparecer de nuevo por el otro lado del escenario tengan de nuevo su valor alfa original.

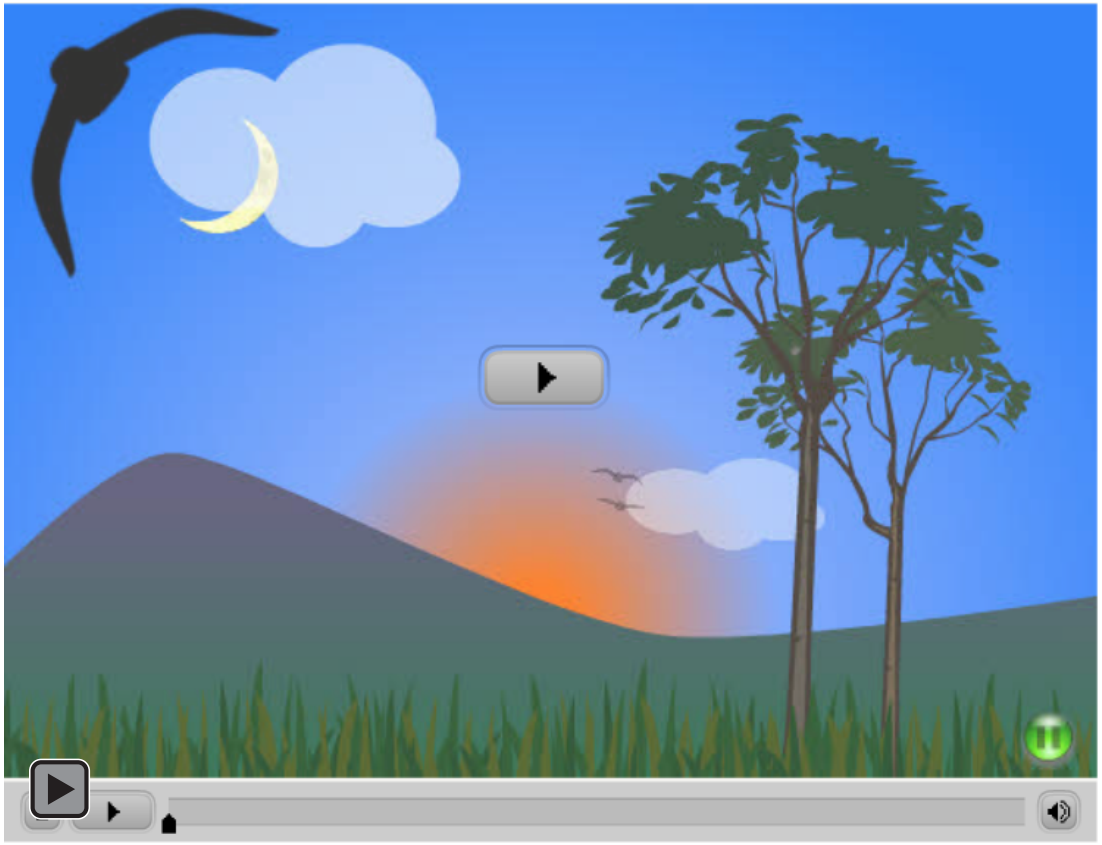
Tutorial 7. Control de la línea de tiempo
Paso 1 de 11

En este tutorial vamos a añadir dos botones para reproducir y reanudar nuestra animación.

Aprenderemos cómo controlar tanto la línea de tiempo principal de una película, como las líneas de tiempo de clips anidados dentro de ella.

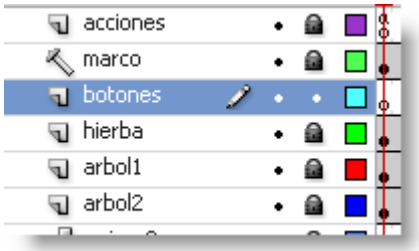
También aprenderemos cómo controlar la reproducción de animaciones creadas mediante programación.

Para hacer este tutorial, abrimos el archivo creado en el tutorial anterior (*tutorial6.fla*) y lo guardamos como *tutorial7.fla*.



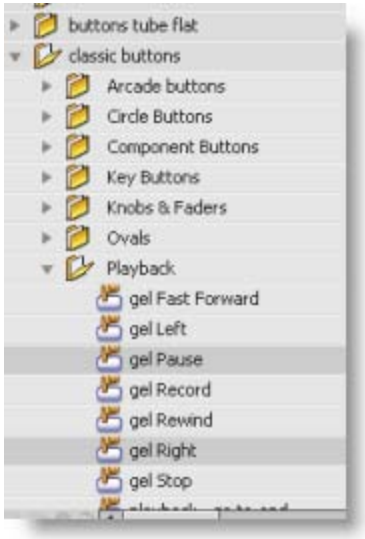
Tutorial 7. Control de la línea de tiempo
Paso 2 de 11

En primer lugar vamos a crear una nueva capa por encima de la hierba, en la que colocaremos los botones con los que controlaremos la línea de tiempo.



Para estos botones vamos a utilizar botones ya creados que podemos encontrar seleccionando **Ventana > Bibliotecas comunes > Botones**. En otro tutorial aprenderemos a modificar y crear nuestros propios botones, y entraremos en detalle sobre su funcionamiento.

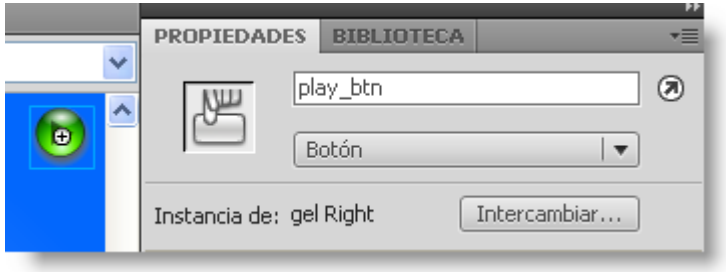
Arrastramos al escenario los botones *gel Pause* y *gel Right* que se encuentran en la carpeta **classic buttons > Playback**.



Ahora estos botones aparecerán en la biblioteca de nuestro documento. Por tanto ya podemos cerrar la biblioteca de los botones.

Seleccionamos la instancia de *gel Pause* en el escenario y le damos el nombre de instancia *pausa_btn* en el **inspector de Propiedades**.

A la instancia del botón *gel Right* le damos el nombre *play_btn*



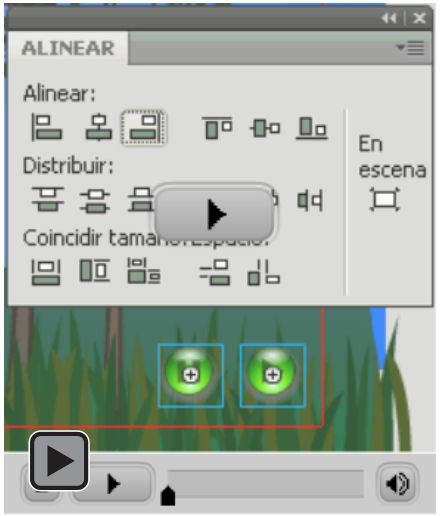
Cuando la línea de tiempo se esté reproduciendo queremos que esté visible el botón para pausar la película, y cuando la película esté pausada necesitaremos tener visible el botón para reanudarla.

Es decir, vamos a hacer que ambos botones se sitúen en el mismo lugar, y que se muestre uno u otro dependiendo del estado de la reproducción, como si fuera un conmutador.

Tutorial 7. Control de la línea de tiempo
Paso 3 de 11

Seleccionamos los botones con la **tecla Mayúsculas** pulsada para poder seleccionar ambos al mismo tiempo. Los situamos cerca de la esquina inferior derecha del escenario. Recordemos que los límites del escenario están definidos en este caso por el marco rojo.

Con ayuda del **panel Alinear**, colocamos ambos botones en el mismo lugar del escenario, uno encima del otro.



Abrimos el **panel Acciones (F9)**. Escribiremos tras la última línea que habíamos programado en el tutorial anterior, en el fotograma 1 de la capa *acciones*.

Para diferenciar ambos tutoriales y mejorar la legibilidad del código, vamos a introducir, además de alguna línea en blanco, un comentario que separe ambas partes del código y que nos indique en qué zona de la programación nos encontramos. Por ejemplo podemos escribir el siguiente comentario:

```
//botones de control de la línea de tiempo
```

Cuando probamos la película, ésta se reproduce automáticamente, por lo que el botón que queremos que esté visible en un primer momento es el botón para pausar la animación. Por tanto, tendremos que hacer invisible el botón *play_btn*. Para ello escribiremos este código:

```
play_btn.visible = false;
```

La propiedad *visible* admite los valores *true* (visible) y *false* (invisible).

Tutorial 7. Control de la línea de tiempo
Paso 4 de 11

Un botón invisible no sólo no se verá, sino que además no estará activo. Sin embargo, un botón con un valor 0 de alfa podrá pulsarse aunque no sea visible. Es por ello que, cuando queremos desactivar un objeto además de hacerlo invisible, sea preferible utilizar la propiedad *visible* en lugar de la propiedad *alpha*.

El siguiente paso es añadir detectores del evento de ratón CLICK para ambos botones, llamado el botón *play_btn* a una función a la que llamaremos reproducir, y el botón *pausa_btn* a una función llamada detener:

```
pausa_btn.addEventListener(MouseEvent.CLICK,detener);
play_btn.addEventListener(MouseEvent.CLICK,reproducir);
```

La función *detener* parará la película. Cuando la película esté parada necesitaremos tener visible el botón para reanudar la reproducción, mientras que ya no será necesario mostrar el botón para pausarla.

En el caso de la función reproducir, en primer lugar reanudará la reproducción de la película. Cuando la película se reproduzca de nuevo, necesitaremos tener visible el botón para pausarla, pero ya no será necesario tener visible el botón para reproducirla.

Las funciones *stop()* y *play()* detienen y reproducen respectivamente la línea principal de tiempo.

Por lo tanto, de momento las funciones detener y reproducir quedarán de la siguiente manera:

```
function detener(e:MouseEvent):void
{
     stop();
     play_btn.visible = true;
     pausa_btn.visible = false;
}

function reproducir(e:MouseEvent):void
{
     play();
     pausa_btn.visible = true;
     play_btn.visible = false;
}
```

Si hay alguna duda con esta parte de este código, recomendamos repasar el tutorial anterior.

Tutorial 7. Control de la línea de tiempo
Paso 5 de 11

La línea `play_btn.visible = false;` que habíamos colocado antes de las funciones (paso 3), se ejecutará directamente al probar la película cuando la cabeza lectora esté en el primer fotograma. Sin embargo, el resto de instrucciones sobre la propiedad visible, al estar dentro de una función, sólo serán ejecutadas cuando la función que las contiene sea llamada.

Seleccionamos **Control > Probar película (Ctrl+Intro)**.

Podemos comprobar por un lado cómo se alterna la visibilidad de los botones *play_btn* y *pausa_btn* al pulsarlos.

Observamos que pulsando *pausa_btn* se detiene la línea de tiempo principal, es decir, el avance de los pájaros. Sin embargo las alas siguen en movimiento, ya que la función *stop()* hace referencia a la línea de tiempo principal, pero no a las líneas de tiempo anidadas dentro de ella (como en este caso las alas).

Por otro lado, también continúa el avance de las nubes por el escenario, ya que su movimiento depende del evento *enter frame*, y este evento es independiente del desplazamiento de la cabeza lectora en la línea de tiempo principal.

Para solucionar el problema del movimiento de las alas, primero tendremos que saber cómo acceder a ellas desde la programación en la línea de tiempo principal.

Para acceder a una línea de tiempo anidada dentro de otra, en la programación escribiremos el nombre de la instancia principal, seguida de un punto, y seguida del elemento que está contenido dentro de ella.

Por ejemplo, supongamos que queremos reproducir la línea de tiempo de la mano derecha que se encuentra dentro del clip del brazo derecho que a su vez pertenece a un cuerpo. Para ello tendríamos que escribir lo siguiente:

```
cuerpo.brazo_derecho.mano_derecha.play();
```

De esa forma la función *play()* haría referencia a esa mano.

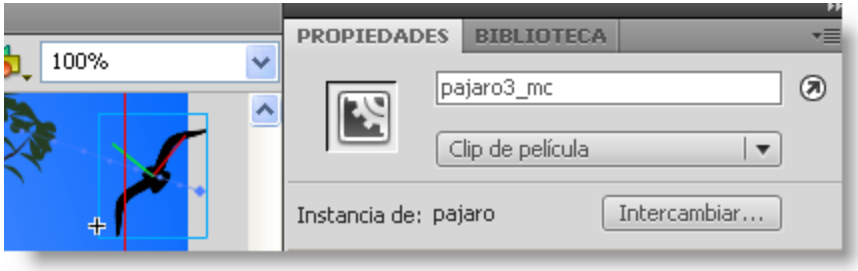
Por lo tanto, para poder acceder a las alas de cada pájaro tendremos que dar un nombre de instancia en primer lugar a cada pájaro, y después a las alas.

Tutorial 7. Control de la línea de tiempo
Paso 6 de 11

Para poder dar un nombre de instancia a los pájaros, desbloqueamos en primer lugar las capas que los contienen.

Seleccionamos cada pájaro en el escenario y les asignamos los nombres de instancia *pajaro1_mc*, *pajaro2_mc* y *pajaro3_mc*, siguiendo la numeración de los nombres de las capas en las que se encuentran.

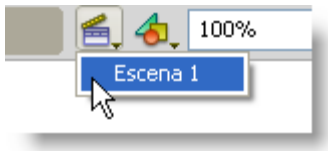
Para seleccionar a los pájaros 2 y 3 tendremos que desplazar la cabeza lectora, ya que no aparecen en el escenario hasta los fotogramas 24 y 150 respectivamente.



Si clicamos dos veces sobre cualquiera de los pájaros del escenario, entraremos a la edición del clip *pájaro* original.

Este clip de película principal tiene en su interior dos instancias del clip de película *ala*. Seleccionamos el ala izquierda y le damos el nombre de instancia *ala1_mc*. Al ala derecha le damos el nombre *ala2_mc*.

Volvemos a la escena principal pulsando sobre *Escena1* en la barra de edición, o bien seleccionándola en el desplegable.



Tutorial 7. Control de la línea de tiempo
Paso 7 de 11

Si pulsamos dos veces sobre cualquiera de los otros pájaros, veremos que nos lleva al mismo clip *pájaro* original, al que ya hemos dado los nombres de instancia *ala1_mc* y *ala2_mc*.

Probemos a detener un ala de un pájaro. Para ello tendremos que añadir la siguiente línea de código dentro de función *detener:*

```
pajaro1_mc.ala1_mc.stop();
```

de tal forma que la función *detener* queda como sigue:

```
function detener(e:MouseEvent):void
{
     stop();
     play_btn.visible = true;
     pausa_btn.visible = false;
     pajaro1_mc.ala1_mc.stop();
}
```

Probamos la película y pulsamos sobre el botón *pausa_btn*. El ala izquierda del primer pájaro se detendrá a la vez que la línea de tiempo principal. Aunque reanudemos la película, este ala no volverá a reproducirse ya que no hemos escrito instrucciones para ello.

Antes de continuar, probemos a sustituir en la programación *pajaro1_mc* por *pajaro3_mc*, quedando la línea como sigue:

```
pajaro3_mc.ala1_mc.stop();
```

Si volvemos a probar la película, y pulsamos el botón *pausa_btn* antes de que el pájaro 3 aparezca en escena, nos encontraremos con el siguiente error en el **panel Salida**:

```
TypeError: Error #1009: No se puede acceder a una
propiedad o a un método de una referencia a un objeto
nulo.
     at tutorial7_fla::MainTimeline/detener()
```

Esto quiere decir que al pulsar el botón para detener la película y ejecutarse la función *detener*, se hace referencia a un objeto que no existe en ese momento (*pajaro3_mc* en este caso), y no puede por tanto ejecutar la instrucción de detener el ala de ese pájaro.

Pese a que esto no impide el funcionamiento correcto de nuestra película, para evitar este error comprobaremos qué pájaros están en el escenario en el momento de pulsar los botones, y sólo ejecutaremos las instrucciones para detener las alas de esos pájaros.

Tutorial 7. Control de la línea de tiempo
Paso 8 de 11

Para saber qué pájaros se encuentran en el escenario en el momento de pulsar un botón será necesario saber en primer lugar en qué fotograma nos encontramos en ese momento.

Para conocer en qué fotograma se encuentra la cabeza lectora en un momento dado, utilizaremos la propiedad *currentFrame*.

Sabemos que el pájaro 1 está desde el primer fotograma y que desaparece en el fotograma 500. Por lo tanto, si la propiedad *currentFrame* devuelve un número menor que 501, el pájaro 1 estará en escena.

El pájaro 2 aparece en el fotograma 24 y desaparece en el 524, así que si la cabeza lectora se encuentra entre esos dos valores, también podremos asegurar que el pájaro 2 está en el escenario. De la misma forma, el pájaro 3 está presente entre los fotogramas 150 y 350.

Para considerar que un pájaro está en escena no es necesario que se encuentre dentro del marco visible del escenario, sino que lo que se tiene en cuenta es que esté presente en la línea de tiempo.

Teniendo en cuenta que < significa menor que, > significa mayor que, y && equivale al *AND* lógico, la función detener quedará como sigue:

```
function detener(e:MouseEvent):void
{
     stop();
     play_btn.visible = true;
     pausa_btn.visible = false;

     if (currentFrame < 501)
     {
          pajaro1_mc.ala1_mc.stop();
          pajaro1_mc.ala2_mc.stop();
     }
     if (currentFrame > 23 && currentFrame < 525)
     {
          pajaro2_mc.ala1_mc.stop();
          pajaro2_mc.ala2_mc.stop();
     }
     if (currentFrame > 149 && currentFrame < 351)
     {
          pajaro3_mc.ala1_mc.stop();
          pajaro3_mc.ala2_mc.stop();
     }
}
```

Tutorial 7. Control de la línea de tiempo
Paso 9 de 11

Vamos a explicar uno de los condicionales para comprender mejor el código. Por ejemplo, con la condición

```
if (currentFrame > 23 && currentFrame < 525)
```

estamos comprobando si el fotograma actual es mayor que el fotograma 23, y, además, es menor que el fotograma 525.

Si se cumplen ambos requisitos, entonces la condición completa se cumple, y pasarán a ejecutarse las instrucciones que estén entre las llaves de ese *if*. Si alguna de las partes de la condición no se cumple, no se ejecutarán las instrucciones correspondientes.

Según en qué fotograma estemos, puede que se cumplan las condiciones que hemos escrito para todos los pájaros (por ejemplo el fotograma 200), para sólo alguno de ellos (por ejemplo el fotograma 15), o para ninguno de los 3 (por ejemplo el fotograma 580).

Si probamos la película, veremos que ahora se paran correctamente las alas de los tres pájaros, y no se nos informa de ningún error.

Ahora nos falta volver a activar el movimiento de las alas cuando volvemos a reproducir la película.

Para activar de nuevo el movimiento de las alas, copiamos en la función *reproducir* las sentencias condicionales que hemos utilizado en la función *detener*. Después, simplemente sustituimos el método *stop()* por el método *play(),* por ejemplo:

```
if (currentFrame < 501)
{
    pajaro1_mc.ala1_mc.play();
    pajaro1_mc.ala2_mc.play();
}
```

Probamos de nuevo la película. Ahora con los botones *play_mc* y *pausa_mc* podemor detener y reanudar tanto el avance de los pájaros por el escenario, como el movimiento de sus alas.

Además de parar con *stop()* y reanudar con *play(),* existen otros métodos interesantes a la hora de controlar una línea de tiempo, como por ejemplo:
- *nextFrame()* y *prevFrame()*, para avanzar y retroceder un fotograma
- *gotoAndPlay()* y *gotoAndStop()*, para ir a un fotograma concreto, y una vez allí continuar o detenernos, escribiendo dentro del paréntesis el número de fotograma al que queremos ir o su nombre (etiqueta).

Tutorial 7. Control de la línea de tiempo
Paso 10 de 11

Hasta aquí hemos visto cómo podemos controlar la cabeza lectora tanto de la línea de tiempo principal como de líneas de tiempo anidadas.

Ahora nos queda detener y reanudar el movimiento de las nubes, ya que este movimiento es independiente del desplazamiento de la cabeza lectora en la línea de tiempo principal.

El movimiento de cada nube lo habíamos creado en el tutorial anterior con el siguiente código:

```
nube1_mc.addEventListener(Event.ENTER_FRAME,desplazar);
nube2_mc.addEventListener(Event.ENTER_FRAME,desplazar);
```

Para que las nubes se detengan, será suficiente con eliminar los detectores del evento *enter frame*. Para eliminarlos basta con copiar el mismo código que se usó para crearlos, y sustituir *addEventListener* por *removeEventListener*.

```
nube1_mc.removeEventListener(Event.ENTER_FRAME,desplazar);
nube2_mc.removeEventListener(Event.ENTER_FRAME,desplazar);
```

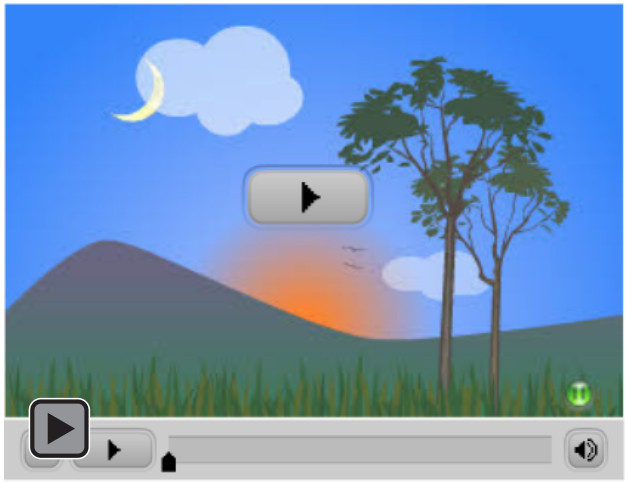
Por lo tanto, añadiremos las dos sentencias con *removeEventListener* a la función *detener* antes de la última llave que marca el final de la función. Para que el movimiento se reanude después, añadiremos de nuevo en la función *reproducir* las dos sentencias con *addEventListener.*

Si además de querer detener el movimiento, quisiéramos que las nubes dejaran de ser arrastrables, tendríamos que seguir el mismo procedimiento, es decir, copiar la misma sentencia que utilizamos para agregar el *listener* y sustituir *add* por *remove.* En este caso vamos a permitir que las nubes puedan arrastrarse aunque la línea de tiempo esté detenida, así que no vamos a eliminar esos *listeners*.

El método *removeEventListener* puede ser muy útil en diversas circunstancias, por ejemplo para eliminar la interactividad de un botón o un clip sin necesidad de esconderlo ni eliminarlo del escenario.

Seleccionamos **Control > Probar película**. Ahora podemos detener y reanudar todas las animaciones (pájaros, alas y nubes).

Tutorial 7. Control de la línea de tiempo
Paso 11 de 11



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Añadid botones para avanzar y retroceder un fotograma
2. Añadid un botón que vaya al principio de la película
3. Haced que la nubes se puedan arrastrar sólo cuando la película esté detenida.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo

Paso 1 de 14

En este tutorial vamos a aprender a añadir un sonido de fondo a nuestra película. También añadiremos sonido a nuestros botones.

Crearemos nuevos botones para activar y desactivar el sonido general de la película. Estos nuevos botones los crearemos duplicando los botones que ya teníamos en nuestra biblioteca.

En primer lugar vamos a abrir el archivo *tutorial7.fla* que creamos en el tutorial anterior, y lo guardamos como *tutorial8.fla*.

Los sonidos que vamos a utilizar se encuentran en la carpeta *tutorial8.* Para importarlos seleccionamos **Archivo > Importar > Importar a biblioteca**, y seleccionamos los archivos *fondoMusical.mp3* y *Mouse.mp3.*

Los sonidos pasarán automáticamente a formar parte de nuestra biblioteca.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 2 de 14

Vamos a organizar el contenido de la biblioteca en carpetas.

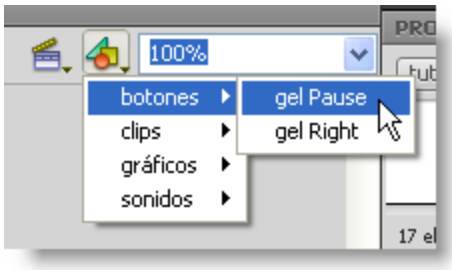
Para crear una carpeta pulsaremos sobre el icono inferior **Nueva carpeta**. Crearemos cuatro carpetas, una para gráficos, otra para clips, otra para botones y otra para sonidos.

Seleccionamos los distintos objetos que queremos incluir en cada una de las carpetas pulsando sobre ellos con la tecla **Ctrl** pulsada para hacer una selección múltiple.

Después arrastraremos los objetos seleccionados sobre la carpeta en la que queremos incluirlos. El mapa de bits de la luna lo incluiremos también en la carpeta de gráficos.

Ahora vamos a acceder a la línea de tiempo de los botones para añadir el sonido del clic del ratón. Para acceder a estas líneas de tiempo tenemos varios caminos posibles, como por ejemplo buscar el símbolo correspondiente en la biblioteca y hacer doble clic sobre su nombre.

Otra forma de acceder a cada botón es seleccionando su nombre desde el icono **Editar símbolos** de la barra de edición. Aquí aparecerá la misma estructura de carpetas que tengamos en la biblioteca.



También podemos hacer doble clic sobre el objeto en el escenario. Para poder acceder a un objeto que se encuentra bajo otro en la misma capa, pulsamos sobre el que está en primer plano con el **botón derecho** del ratón, y seleccionamos **Organizar > Enviar al fondo**.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 3 de 14

Accedemos a la línea de tiempo del botón *gel Pause* utilizando cualquiera de las formas antes descritas.

Hacemos zoom sobre el escenario para ver con más detalle el aspecto del botón

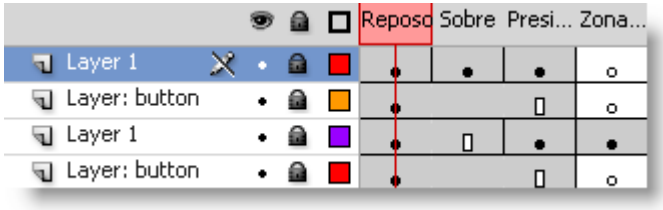
Las líneas de tiempo de los botones tienen cuatro fotogramas que representan diferentes estados de un botón:

• **Reposo.** Aspecto inicial del botón. Representa el momento en que el puntero no está sobre el botón.

• **Sobre.** Aspecto del botón cuando situamos el puntero sobre él.

• **Presionado.** Aspecto del botón cuando hacemos clic sobre él.

• **Zona activa** define el área que responde al clic del ratón. Esta zona no será visible para el usuario final. Permite crear áreas de clic con diferente forma o tamaño que el propio aspecto del botón. También permite que zonas aparentemente vacías puedan ser clicables.



Los botones, al igual que otras líneas de tiempo, pueden tener el número de capas que queramos.

Los aspectos del botón pueden variar de un fotograma a otro o permanecer sin cambios. Aquí podemos ver que entre el fotograma *Reposo* y el fotograma *Sobre* sólo hay cambios en la capa superior, donde un punto negro indica que hay un fotograma clave; mientras que en el estado *Presionado* hay diferencias en dos de las capas.

Aunque podemos añadir interactividad tanto a los clips de película como a los botones, es la diferencia en las características de la línea de tiempo lo que hace que pueda ser más adecuado utilizar un tipo u otro de símbolo dependiendo de nuestras necesidades.

Si queremos que, de forma sencilla y sin programación, el símbolo muestre distintos aspectos dependiendo de su estado (reposo, sobre, presionado), entonces los botones son una buena elección.

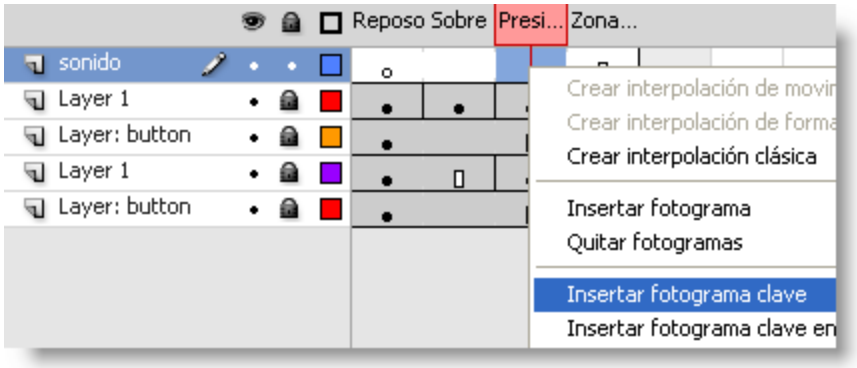
Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 4 de 14

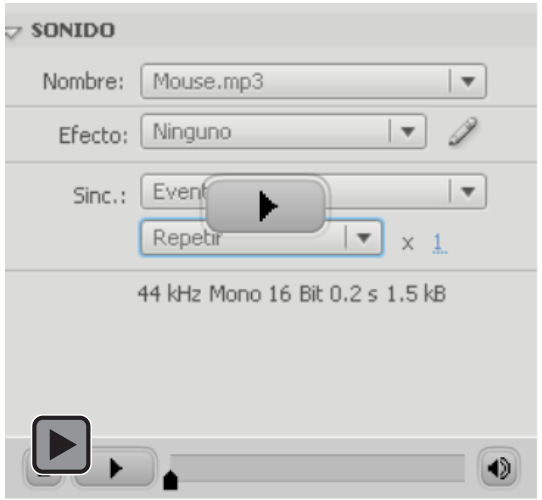
Vamos a añadir un pequeño sonido de clic en el momento de presionar el botón.

En primer lugar, añadimos una **nueva capa** a la que llamaremos *sonido*.

Como queremos que el sonido se oiga cuando hagamos clic sobre el botón, insertaremos un fotograma clave para el estado *Presionado*.

Para ello, hacemos clic con el **botón derecho** del ratón en el fotograma *Presionado* de la capa *sonido*, y seleccionamos **Insertar fotograma clave** en el menú contextual.





Con ese fotograma seleccionado, vamos al área **Sonido** del **inspector de Propiedades**. Seleccionamos en el desplegable **Nombre** el archivo **Mouse.mp3** que ya tenemos en nuestra biblioteca.

Como **sincronización** elegimos **Evento**, que significa que el sonido se reproducirá completamente cuando la cabeza lectora alcance el fotograma en el que se encuentra el sonido (cuando presionemos el botón en este caso).

Explicaremos más adelante las diferentes opciones de sincronización.

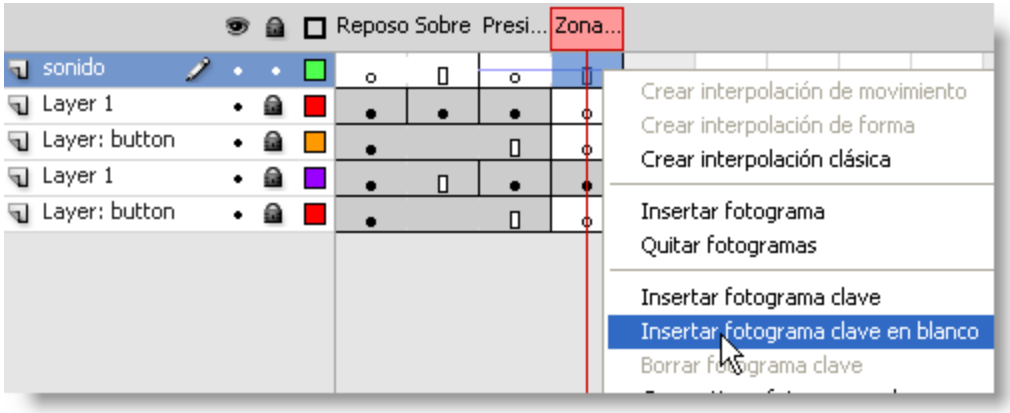
Tutorial 8. Añadir sonido a botones y a la línea de tiempo

Paso 5 de 14

La capa *sonido* mostrará, comenzando en el fotograma *Presionado*, una pequeña línea azul que representa la onda del sonido seleccionado para ese fotograma.

Podemos observar que el sonido se extiende al fotograma de la *Zona activa*. En este caso, esto no tendrá repercusión sobre el sonido, pero para mayor claridad insertaremos un fotograma clave vacío en el fotograma de la *Zona activa* de la capa *sonido*.

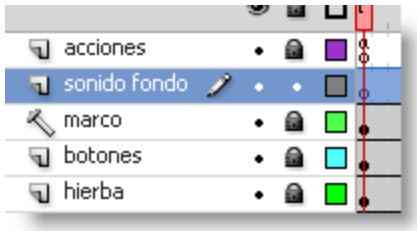
Para ello, seleccionamos el fotograma con el **botón derecho** del ratón, y seleccionamos **Insertar fotograma clave en blanco**.



Repetimos el mismo proceso para añadir sonido al botón *gel Right* y probamos la película **(Control > Probar película).**

Ahora se oirá un pequeño clic cuando pulsemos sobre cualquiera de los botones.

Volvemos a la línea de tiempo principal de la película, y añadimos una nueva capa para el sonido. Llamamos a esta capa *sonido fondo*.



En el inspector de propiedades le asignamos el sonido *fondoMusical.mp3*. Veremos cómo la línea azul de la onda del sonido se extiende por los 600 fotogramas de nuestra película.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 6 de 14

Analicemos las diferentes opciones de sincronización que podemos asignar a un sonido:

-**Evento**. El sonido comenzará cuando la cabeza lectora alcance el fotograma que contiene el inicio del sonido. Una vez haya comenzado, el sonido se reproducirá en su totalidad, independientemente de la cabeza lectora. Por tanto, es suficiente que el sonido se encuentre en un solo fotograma para que se reproduzca en toda su extensión. Si la cabeza lectora lee de nuevo el fotograma que contiene el sonido antes de que éste haya terminado, comenzará una nueva reproducción del sonido que se mezclará con la anterior.

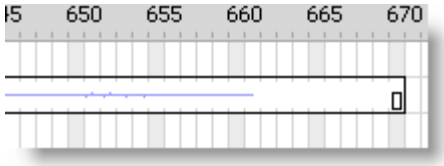
-**Inicio**. Es similar a *Evento*, con la diferencia de que si el sonido se está reproduciendo previamente, no se iniciará una nueva reproducción.

-**Detener**. Detiene el sonido especificado.

-**Flujo**. El sonido estará sincronizado con la línea de tiempo. Por lo tanto, si la línea de tiempo está detenida, el sonido se detendrá también. En este caso los fotogramas tienen que extenderse durante todo el tiempo que queremos que se escuche el sonido. De esta forma podemos sincronizar el sonido con las imágenes.

Seleccionamos en el **inspector de Propiedades** una sincronización de tipo **Flujo** para el sonido *fondoMusical.mp3* y probamos la película.

Podemos comprobar que la música se detiene y se reanuda junto con el resto de la línea de tiempo.

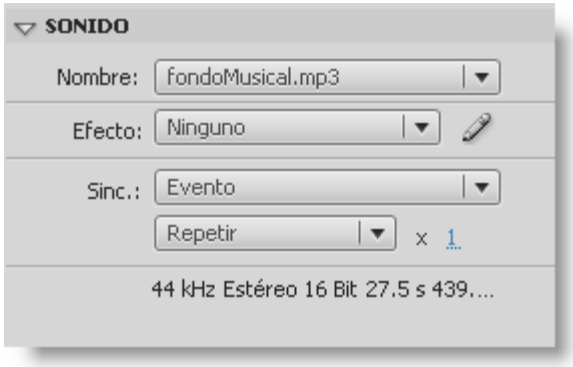


La duración de la pista de música es mayor que los 600 fotogramas disponibles, por lo que no hay ningún momento de silencio durante la reproducción. Si la línea de tiempo tuviera más fotogramas, podríamos ver que la música se extiende hasta el fotograma 661 aproximadamente, y si la animación continuara más allá de estos fotogramas, la música cesaría.

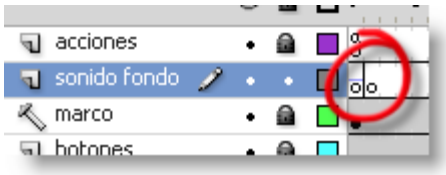
Nuestra línea de tiempo llega al fotograma 600, y allí el código la envía de nuevo al fotograma 2 con la instrucción *gotoAndPlay(2)*. Junto con el resto de la línea de tiempo, el sonido también vuelve a ese fotograma. En este caso no encaja musicalmente el final del fotograma 600 con el inicio en el fotograma 2, por lo que se escucha un salto en la música. Por tanto, en los casos de bucles de fondo que no necesiten estar sincronizados con la animación, la opción de sincronización como flujo no es la más adecuada.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 7 de 14

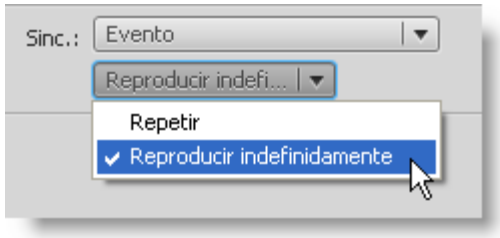
Cambiamos el tipo de sincronización de la pista a una sincronización de tipo **Evento** con una repetición.



Probamos de nuevo la película **(Ctrl+Intro)**. Ahora la reproducción del sonido es independiente de la reproducción de la línea de tiempo. Por ello, podemos **insertar un fotograma clave en blanco** en el segundo fotograma de la capa *sonido fondo*.



Para que el sonido se repita de forma continua en un bucle, seleccionamos el fotograma que contiene el sonido, y seleccionamos **Reproducir indefinidamente**. El sonido se reproducirá ahora de forma continua.



Si no tuviéramos en el fotograma 600 la instrucción gotoAndPlay(2), la línea de tiempo volvería al fotograma 1, fotograma en el que se encuentra el inicio del sonido de fondo. Al tener seleccionada una sincronización tipo Evento, el sonido comenzaría de nuevo, sin por ello parar la reproducción que estuviera en marcha, sonando de forma simultánea. En ese caso la sincronización más adecuada sería *Inicio*.

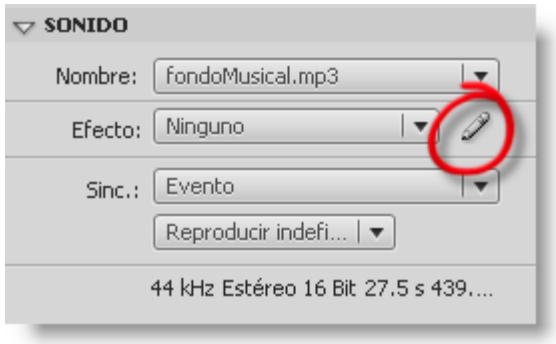
Como en nuestra película la cabeza lectora sólo pasa por el primer fotograma una vez, podemos mantener la sincronización como tipo *Evento*.

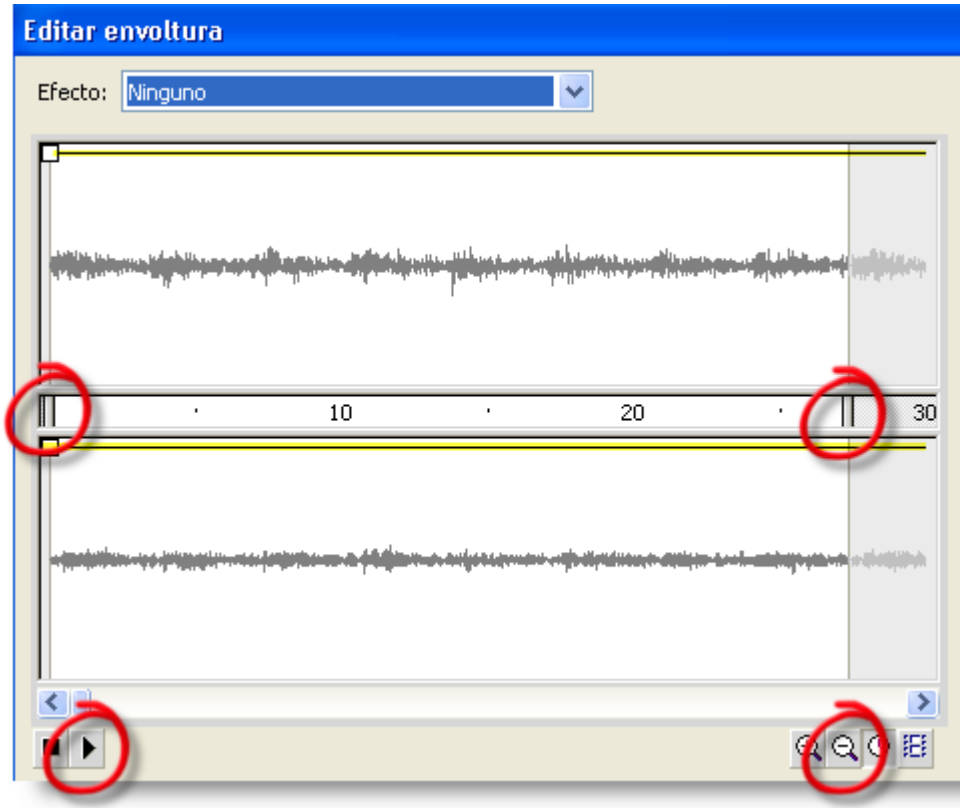
Tutorial 8. Añadir sonido a botones y a la línea de tiempo

Paso 8 de 14

En cada repetición del sonido, que dura unos 27 segundos, podemos escuchar una ligera desincronización.

Podemos arreglar este pequeño desajuste con ayuda de la pantalla **Editar envoltura**. Para abrir esta pantalla pulsaremos en el icono del lápiz que hay junto al desplegable **Efecto**.





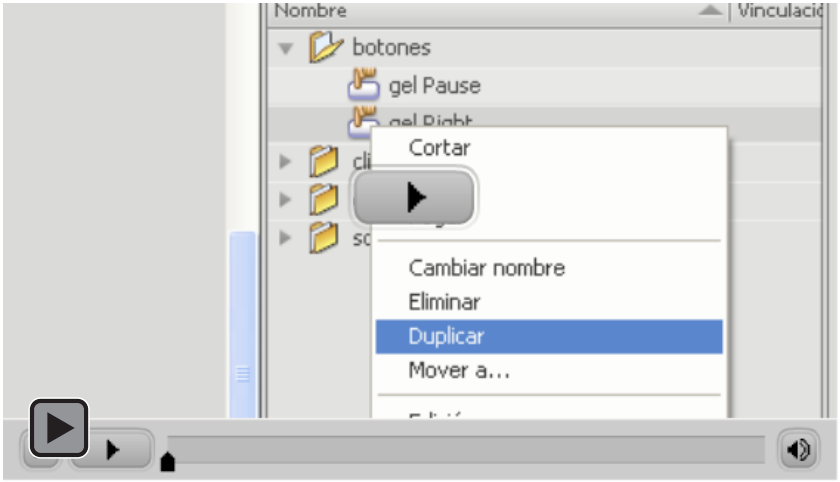
En esta nueva pantalla, podemos ajustar el punto inicial y final del sonido. Podemos reducir el zoom para visualizar toda la onda de sonido. Tras el final de la pista de audio (segundo 27), podremos ver en tono gris el comienzo de la siguiente repetición. Probamos con el botón *play* nuestros ajustes. Cuando nos guste el resultado, clicamos en **Aceptar**. En esta pantalla también podríamos haber ajustado los volúmenes de ambos canales seleccionando algún efecto predeterminado, o bien haciendo clic en la línea superior de cada canal para crear nuevos puntos de control que podemos arrastrar para controlar el volumen.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 9 de 14

El sonido de fondo puede resultar molesto en ocasiones, así que añadiremos dos nuevos botones para activar y desactivar el sonido de la película.

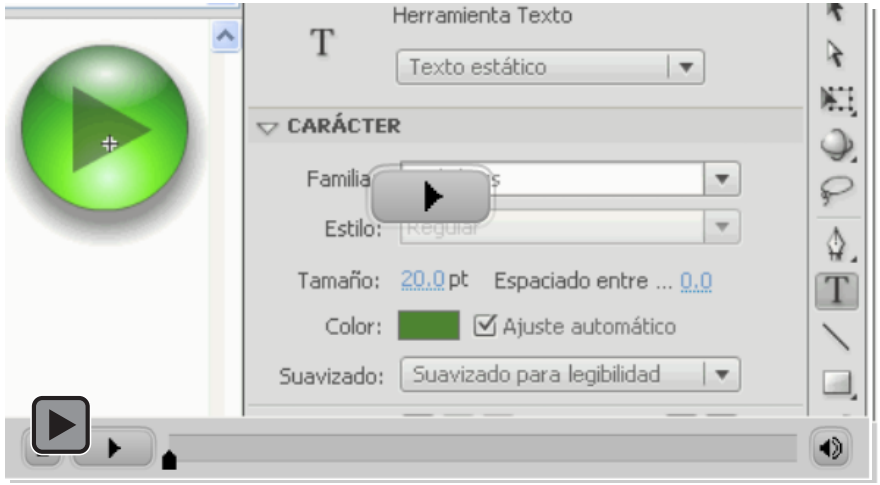
Para que sean similares a los botones que ya tenemos, la forma más rápida es duplicar uno de los botones, y después transformarlo para que se adapte a nuestras necesidades.

En la biblioteca, seleccionamos el botón *gel Right* con el **botón derecho** del ratón, y seleccionamos **Duplicar** en el menú contextual. Al nuevo botón le damos el nombre *gel Volumen.*



Para editar este símbolo hacemos **doble clic** sobre su nombre *(gel Volumen)* en la biblioteca. Vamos a sustituir el icono del play en el fotograma *Reposo* por el icono de un altavoz. Activamos la capa *Layer 4* que es la que contiene el icono del play.

Para dibujar este altavoz, seleccionamos la **herramienta Texto, familia Webdings y tamaño de 20 pt.** Para seleccionar el color, pinchamos sobre la casilla de **Color**, y después con el cuentagotas sobre el icono para que guarde su color. Por último, pulsamos sobre el escenario para crear un cuadro de texto, y pulsamos la **tecla X mayúscula**, que es el icono del altavoz.



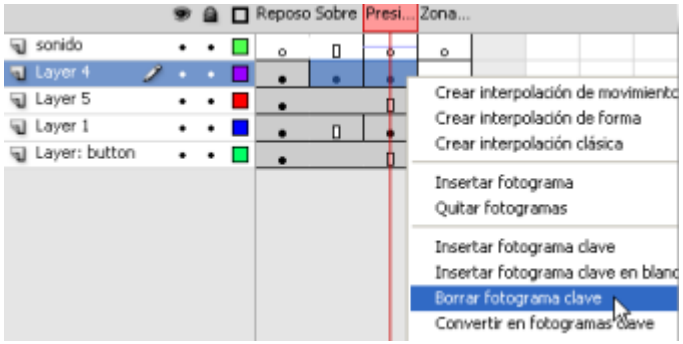
Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 10 de 14

Seleccionamos con el botón derecho del ratón el icono del altavoz que hemos creado, y seleccionamos **Separar** para convertirlo en una forma.

Después borramos el icono del *play*, y situamos el icono del altavoz en el centro del botón.

Si observamos los fotogramas *Sobre* y *Presionado*, veremos que el icono play aparecía con color blanco en estos fotogramas.

Seleccionamos los fotogramas *Sobre* y *Presionado* en los que todavía se encuentra el icono del *play* y, haciendo clic con el **botón derecho** del ratón, seleccionamos **Borrar fotograma clave**.



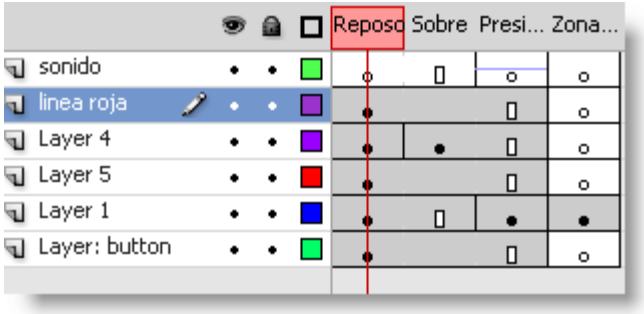
Ahora tendremos el icono del altavoz en el fotograma *Reposo*, que se mostrará en los tres fotogramas. Hacemos clic sobre el fotograma *Sobre* con el botón derecho del ratón y seleccionamos **Insertar fotograma clave** para crear un fotograma clave con el mismo contenido que el fotograma *Reposo*.

Seleccionamos el altavoz en el fotograma *Sobre* y le cambiamos el color a blanco en el **inspector de Propiedades**. Este color se mantendrá sin cambios en el fotograma *Presionado*.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 11 de 14

Para crear el botón que muestre que el audio está desactivado, **duplicaremos** el botón *gel Volumen* que acabamos de crear, y nombramos al nuevo botón *gel Volumen Off.*

Después añadimos una **nueva capa** por encima de la capa *Layer 4*, y dibujamos una línea roja transversal que se mantenga durante los fotogramas *Reposo, Sobre* y *Presionado.*



Arrastramos instancias de ambos botones en el escenario, en la capa *botones*.

Al altavoz con la línea roja le daremos el **nombre de instancia** *encender_btn*, ya que cuando se muestre estaremos sin audio, y por lo tanto, si lo pulsamos será para encender de nuevo el audio. Al otro botón le damos el nombre de instancia *apagar_btn.*

Con ayuda del panel **Alinear**, colocamos un altavoz sobre el otro para que haga el efecto de conmutador, y ambos al lado del otro botón.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo
Paso 12 de 14

Añadimos la programación en el fotograma 1 de la capa acciones, bajo la programación que hemos creado en tutoriales anteriores, que será como sigue:

```
encender_btn.visible = false;
encender_btn.addEventListener(MouseEvent.CLICK,encender);
apagar_btn.addEventListener(MouseEvent.CLICK,apagar);

function encender(e:MouseEvent):void
{
	apagar_btn.visible = true;
	encender_btn.visible = false;
	SoundMixer.soundTransform = new SoundTransform(1);
}

function apagar(e:MouseEvent):void
{
	encender_btn.visible = true;
	apagar_btn.visible = false;
	SoundMixer.soundTransform = new SoundTransform(0);
}
```

Como podemos ver, la parte de la programación que se refiere a mostrar u ocultar cada botón es similar a la de los botones *play_btn* y *pausa_btn* que hicimos en el tutorial anterior.

Las únicas líneas nuevas están relacionadas con una *clase* llamada *SoundTransform:*

```
SoundMixer.soundTransform = new SoundTransform(1);
SoundMixer.soundTransform = new SoundTransform(0);
```

La utilización de las *clases* excede de los límites de este tutorial. Sin embargo, es útil conocer que simplemente con esta línea de código podemos modificar el volumen general de todos los sonidos de una película. El volumen varía entre 0 (apagado) y 1 (volumen máximo). Estos números los colocaremos en el paréntesis del final de la línea.

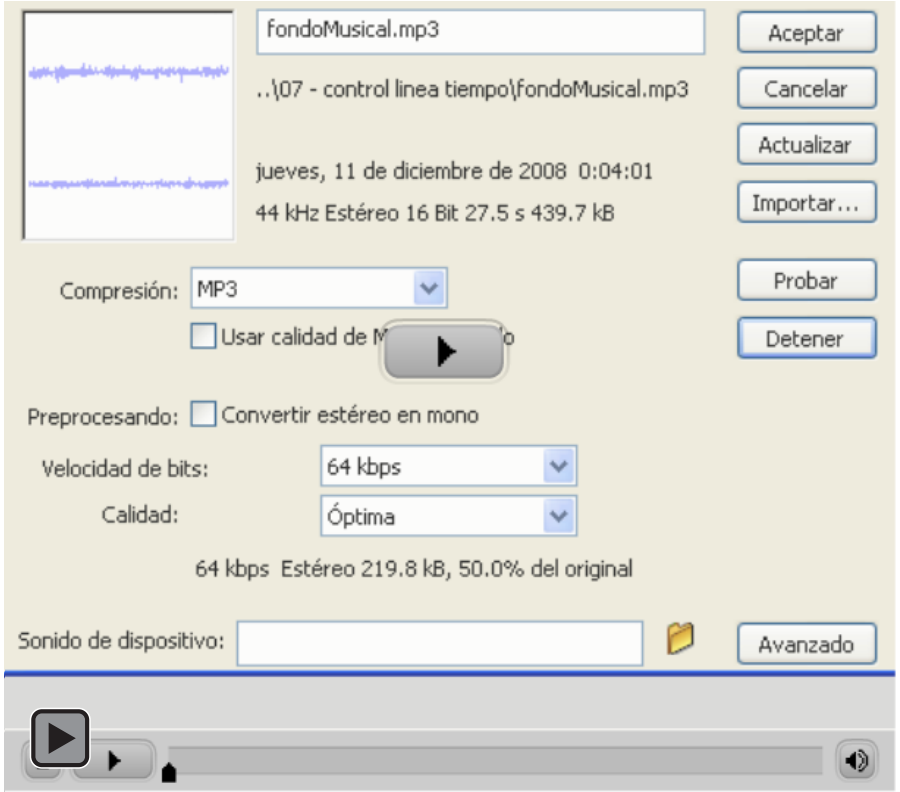
Probamos la película y comprobamos que si pulsamos el botón para apagar el audio, dejará de sonar no sólo la pista de la música de fondo, sino también el clic de los botones.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo

Paso 13 de 14

Para optimizar el peso final de la película, podemos variar la calidad de compresión de los archivos de audio que utilicemos.

Una forma de hacerlo es en la ventana **Propiedades de sonido**, a la que se accede pulsando con el **botón derecho** del ratón sobre el nombre del sonido en la **biblioteca** (*fondoMusical.mp3* en este caso), y seleccionando **Propiedades** en el menú contextual.



En primer lugar **desactivamos** la casilla **Usar calidad de MP3 importado.** Es posible que nos aparezca una velocidad de 16 kbps y calidad rápida. Seleccionamos **Probar** y escuchamos el principio del sonido. Después pulsamos **Detener.** Haremos diversas pruebas con distintos valores hasta encontrar una calidad aceptable sin demasiado peso.

La calidad se refiere a la velocidad con la que comprime el archivo. A mayor velocidad, menor calidad en la compresión. Por ello es recomendable utilizar una calidad óptima.

Debajo de la casilla de calidad podremos ver el peso del archivo con las condiciones actuales de compresión.

En este caso, una opción adecuada podría ser un archivo estéreo, con una velocidad de bits de 64 kbps y con calidad óptima.

Tutorial 8. Añadir sonido a botones y a la línea de tiempo

Paso 14 de 14

Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer la siguiente actividad:

1. Cread en el escenario un clip que contenga un texto con información sobre la película, y hacedlo invisible al inicio de la programación.
2. Añadid botones para mostrar y ocultar esta información.

Tutorial 9. Creación de un juego (I)

Paso 1 de 26

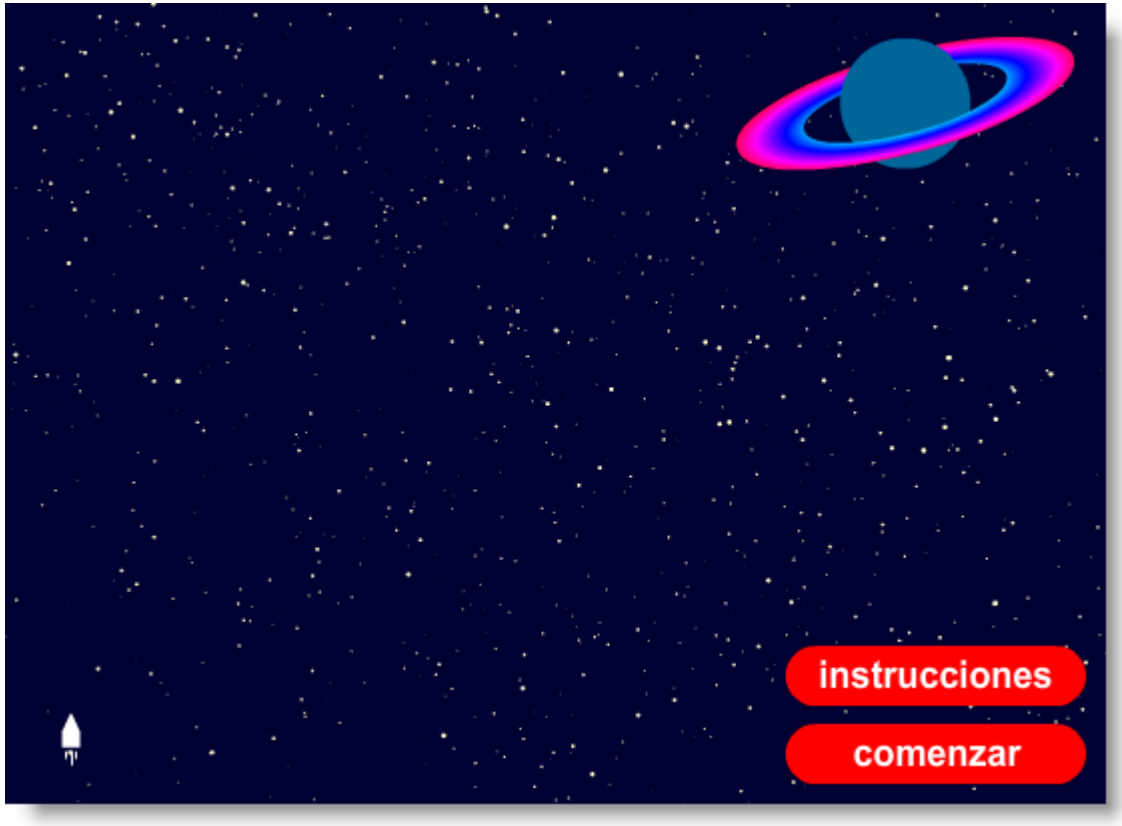
En este tutorial vamos a crear la primera parte de un juego en el que tenemos que llevar una nave hasta un planeta, con ayuda de las flechas del teclado.

Comenzaremos creando los elementos gráficos necesarios para el juego, utilizando herramientas como el óvalo y rectángulo simple, y el pincel rociador.

En lo referente a programación, aprenderemos cómo controlar un objeto con el teclado, así como a detectar su posición o si choca con algún otro objeto.

También aprenderemos a crear botones que nos muestren información al situar el puntero sobre ellos.

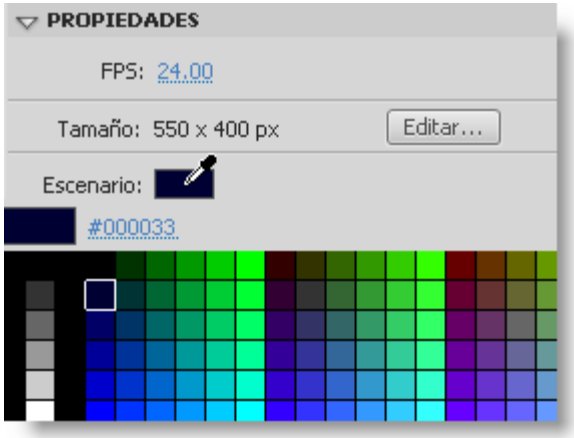
En el siguiente tutorial añadiremos complejidad a nuestro juego añadiendo obstáculos con movimientos aleatorios.



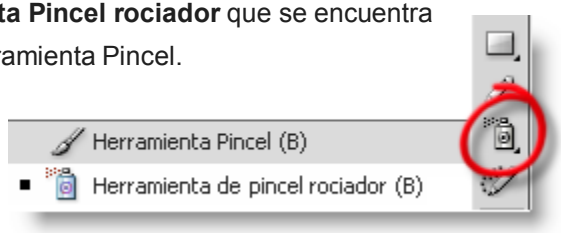
Tutorial 9. Creación de un juego (I)
Paso 2 de 26

Creamos un nuevo **archivo de Flash (AS 3.0)** desde el área **Crear nuevo** de la pantalla de bienvenida o bien seleccionando **Archivo >Nuevo**. Guardamos el archivo como *tutorial9.fla*.

Cambiamos el **color del escenario** en el inspector de propiedades. Le asignamos un color azul oscuro *(#000033)*.

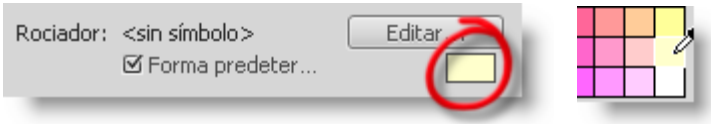


Vamos a crear en primer lugar los elementos gráficos de nuestro juego, comenzando por el fondo de estrellas. Para crear este fondo utilizaremos la **herramienta Pincel rociador** que se encuentra agrupada junto con la herramienta Pincel.



Esta herramienta emite de forma predeterminada un spray de puntos de partículas con el color que seleccionemos.

Con la herramienta de Pincel rociador seleccionada, cambiamos en el **inspector de Propiedades** el color de la partícula a un amarillo claro *(#FFFFCD)*.



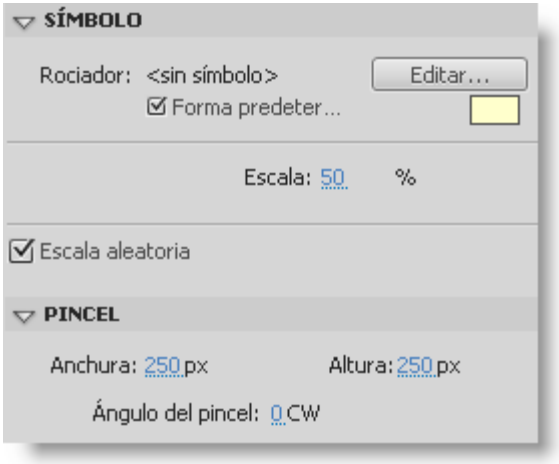
Con el botón *Editar* podríamos seleccionar como partícula emitida un símbolo que tuviéramos en la biblioteca. En este caso mantendremos la forma predeterminada.

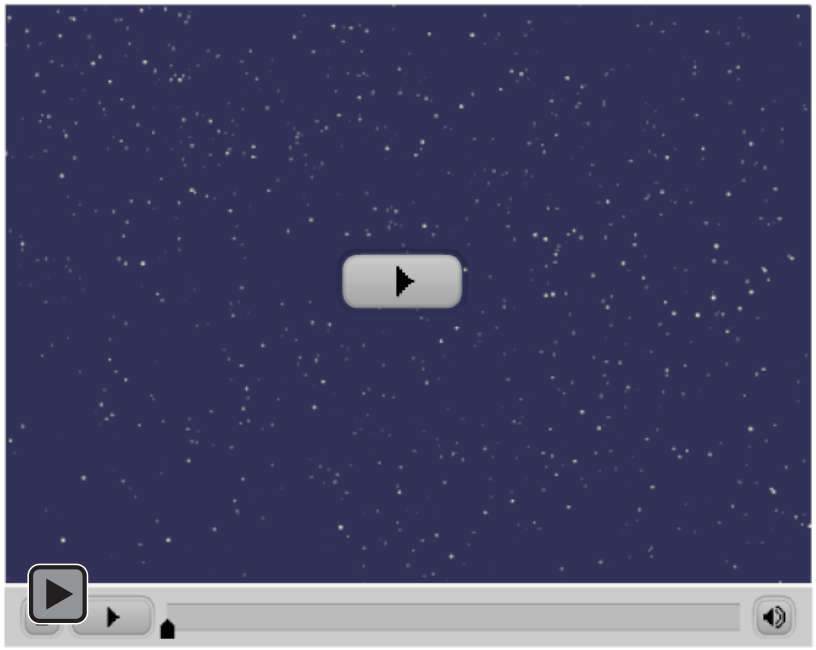
Tutorial 9. Creación de un juego (I)
Paso 3 de 26

Cambiamos la **Escala** a *50%*. La escala se refiere al tamaño de las partículas emitidas.

Seleccionamos la casilla **Escala aleatoria**. Esto hace que las partículas sean de diferentes tamaños, aunque el tamaño máximo será el que hemos indicado previamente en la casilla *Escala*.

Por último, en el área *Pincel*, seleccionamos una **anchura** y **altura** de *250 px*. Esto hará que las partículas estén más distanciadas entre sí.





Pulsamos y arrastramos sobre el escenario con el pincel rociador hasta conseguir un fondo estrellado que nos guste.

Damos a esta capa el nombre *estrellas*. Después bloqueamos la capa para evitar cambios accidentales.

Tutorial 9. Creación de un juego (I)
Paso 4 de 26

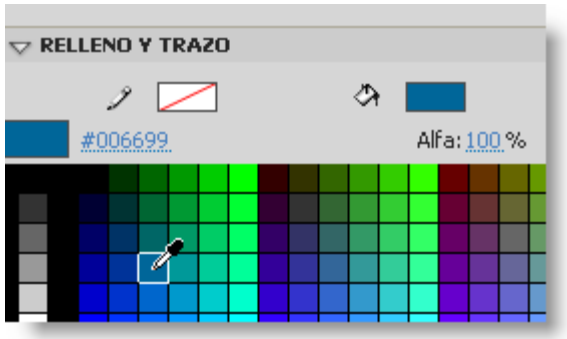
Creamos una nueva capa a la que llamaremos *planeta*.

Nuestro planeta estará formado por la parte central y los anillos, es decir, que contendrá más de una capa. Para que todas las capas formen parte del mismo objeto, lo más cómodo es crearlas directamente dentro de un mismo símbolo.

Seleccionamos **Insertar > Nuevo símbolo**, le damos el nombre *planeta* y como **Tipo** seleccionamos **Clip de película.**

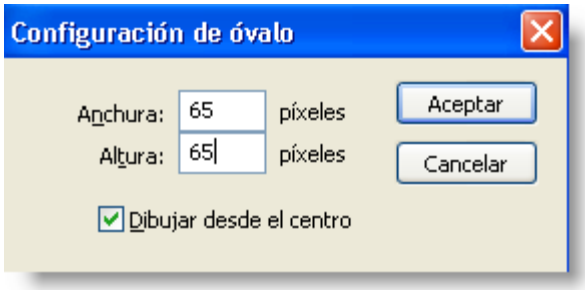
Dentro del símbolo *planeta*, dibujamos en primer lugar la parte central del planeta, y para ello seleccionamos la **herramienta Óvalo**.

En el **inspector de Propiedades** eliminamos el color del trazo y seleccionamos como color de relleno el color *#006699*.

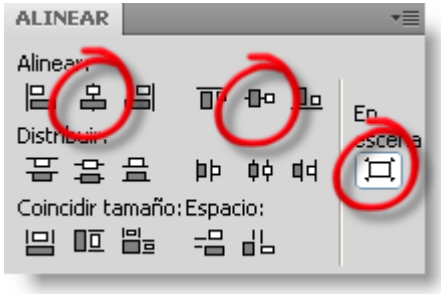


Con la herramienta Óvalo seleccionada, pulsamos la tecla **Alt** y, sin soltarla, hacemos clic sobre el escenario.

Se abrirá la pantalla **Configuración de óvalo**, que permite definir directamente el tamaño de un óvalo antes de dibujarlo. En este caso le vamos a asignar una anchura y altura de 65 px.



Seleccionamos el óvalo que hemos creado y lo centramos dentro de su propio escenario con ayuda del **panel Alinear**.

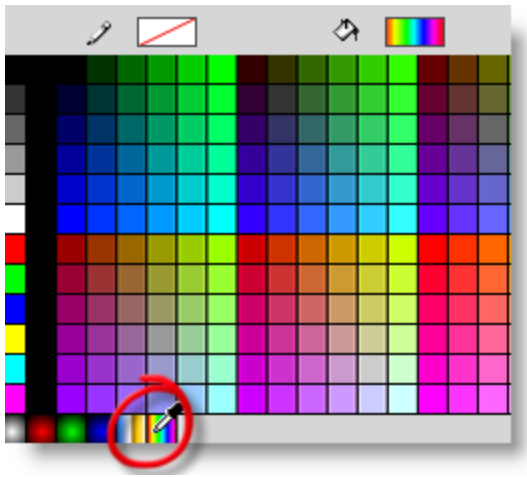


Tutorial 9. Creación de un juego (I)
Paso 5 de 26

Llamamos a esta capa *centro*, y añadimos una nueva capa a la que llamaremos *anillos*.

Seleccionamos esta vez la **herramienta Óvalo simple** para poder modificar algunas características del óvalo después de dibujarlo.

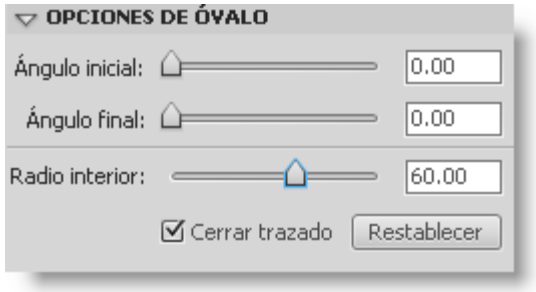
Esta vez tampoco seleccionaremos un trazo en el **inspector de Propiedades**. Como color de relleno seleccionaremos el degradado multicolor que aparece en la parte inferior de la paleta.



Al igual que en el paso anterior, pulsamos la tecla **Alt** y, sin soltarla, hacemos clic sobre cualquier punto del escenario. De momento no es necesario que el óvalo de los anillos esté centrado.

En la pantalla **Configuración de óvalo** asignamos una anchura y altura de 150 px. Aceptamos para crear el óvalo.

En **Opciones de óvalo** establecemos un **Radio interior** de 60. Recomendamos probar las opciones de ángulo inicial y final para conocer las posibilidades que esto ofrece, aunque finalmente dejaremos los ángulos a 0.

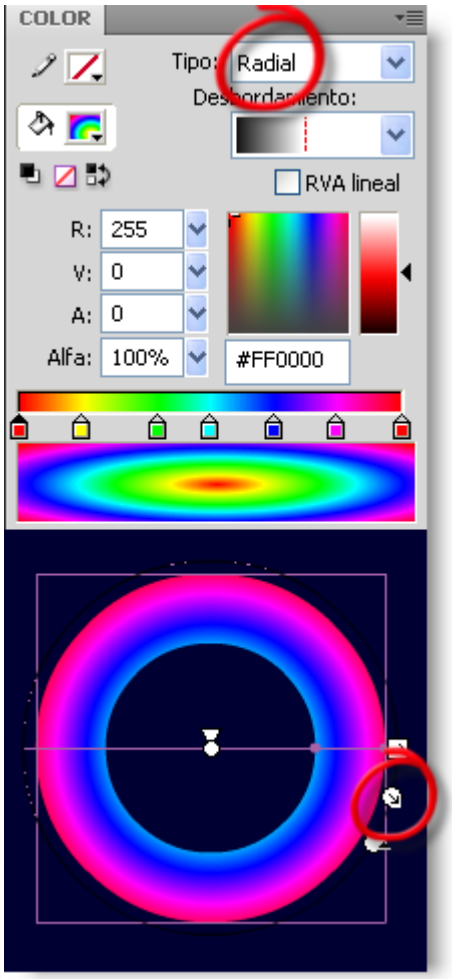


Tutorial 9. Creación de un juego (I)
Paso 6 de 26

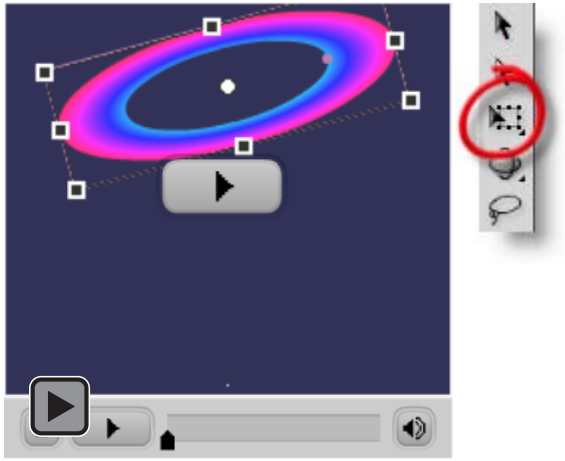
El relleno multicolor que hemos utilizado es por defecto un degradado lineal.

Para convertir el degradado en radial, abrimos en **panel Color** (Ventana > Color) y, con el óvalo de los anillos seleccionado, cambiamos el **Tipo** a **Radial**.

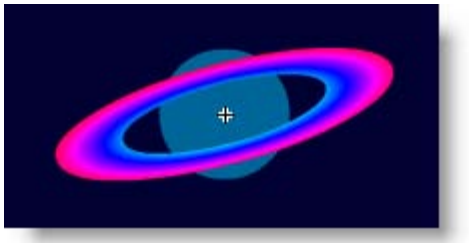
Si queremos cambiar los colores de los anillos, podemos probar con la herramienta **Transformación de degradado**, y cambiar la amplitud del degradado para que se muestren diferentes colores.



Con la **herramienta Transformación libre** modificamos la forma de los anillos, tal y como se muestra en el vídeo.



Centramos los anillos en el escenario con ayuda del **panel Alinear**. La parte central del planeta deberá asomar ligeramente tanto por encima como por debajo de los anillos. De no ser así, haremos las rectificaciones pertinentes en los anillos con la herramienta transformación libre.



Tutorial 9. Creación de un juego (I)
Paso 7 de 26

La parte central del planeta queda completamente por detrás de los anillos. Sin embargo, para que parezca que esa parte se encuentra dentro de los anillos, una parte del planeta debería verse por delante de los anillos, y otra parte por detrás.

Para conseguir este efecto, duplicaremos el centro del planeta en una nueva capa superior, a la que también podemos llamar *centro.*

Copiamos el óvalo del centro del planeta clicando sobre él con el **botón derecho** del ratón y seleccionando **Copiar**.

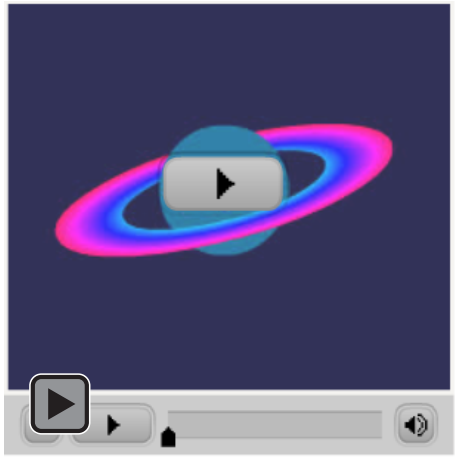
Bloqueamos las dos capas inferiores y, con la capa superior activa, clicamos con el botón derecho sobre el escenario y seleccionamos **Pegar in situ**.

Esto copiará el dibujo del centro del planeta en la misma posición que tenía en la capa inferior.

Si ahora borramos de la capa superior una parte del planeta, podremos ver lo que se encuentra bajo esa parte, es decir, los anillos en primer lugar, y tras ellos el centro del planeta en la capa inferior.

Probamos a eliminar la parte superior o inferior del centro del planeta en la capa superior, según la perspectiva que queramos dar a nuestro planeta (visto desde arriba o visto desde abajo).

Observad el vídeo para entender mejor lo que ocurre al borrar una zona del planeta en la capa superior. Elegid la perspectiva que más os guste.



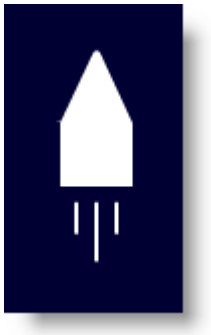
Tutorial 9. Creación de un juego (I)
Paso 8 de 26

Volviendo a la escena principal, y con la capa *planeta* activa, arrastramos desde la biblioteca una instancia del clip *planeta.*

Situamos el clip en la parte superior derecha del escenario, y le damos el nombre de instancia *planeta_mc.*

Creamos una nueva capa llamada *nave* en la que situaremos la pequeña nave que controlará el usuario.

Esta nave tendrá una sola capa, así que podemos crearla en la capa nave de la escena principal.



Dibujamos una nave de aspecto similar al de esta imagen. Para ello podemos utilizar herramientas como el rectángulo y la línea.

La imagen se encuentra muy ampliada. La altura total de la nave debe ser aproximadamente de 25 px, y la anchura unos 10 px.

Una vez dibujada, seleccionamos **Modificar > Convertir en símbolo** o pulsamos **F8**. Le damos al clip el nombre *nave*, y como **Registro** le asignamos el punto central. Esto hará que el clip quede centrado en su propio escenario.

Situamos la nave en la parte inferior izquierda del escenario. En el **inspector de Propiedades** le asignamos el nombre *nave_mc*.

Con estos dos clips ya podemos comenzar la programación de nuestro juego. Creamos una nueva capa a la que llamaremos *acciones*, y con el primer fotograma de esta capa seleccionado, pulsamos **Ventana > Acciones** o **F9**.

Tutorial 9. Creación de un juego (I)
Paso 9 de 26

Queremos que la nave gire y se desplace al pulsar las flechas del teclado, así que el primer paso será convertir la nave en un detector de eventos del teclado.

Programamos que cuando se detecte una pulsación de tecla, se ejecute una función a la que hemos llamado *flechas*. La forma de añadir un evento de teclado es similar a lo que hemos visto hasta ahora.

```
nave_mc.addEventListener(KeyboardEvent.KEY_DOWN,flechas);

function flechas(e:KeyboardEvent):void
{
    trace("He pulsado una tecla");
}
```

Con esta programación le estamos diciendo a la nave que, cuando detecte que se ha pulsado una tecla, ejecute una función que se llama *flechas*. Esta función, que recibe un evento de teclado, tiene a su vez la orden de escribir en el panel Salida la frase *He pulsado una tecla*.

Sin embargo, si probamos la película y pulsamos las flechas del teclado no ocurrirá nada.

Esto se debe a que para que funcione un detector de eventos de teclado, el objeto encargado de detectarlo debe recibir el foco de la película, es decir, debe estar seleccionado. Para seleccionarlo añadimos al inicio de la programación el siguiente código:

```
stage.focus = nave_mc;
```

Ahora que la nave tiene el foco, detectará las pulsaciones de nuestro teclado, escribiendo tras cada pulsación la frase que habíamos puesto en el *trace*. Cuando estamos en fase de pruebas, esta detección no funcionará con las teclas que tengamos en Flash como métodos abreviados de teclado. Por ello en la fase de pruebas nos limitaremos también a las flechas del teclado.

Al tener ahora la nave el foco, podemos ver que aparece un rectángulo alrededor de la nave para indicar que tiene el foco. Para que no aparezca este rectángulo añadiremos, antes de dar el foco a la nave, una sentencia que especifique que, aunque nuestra nave tenga el foco, no muestre un rectángulo. Las dos primeras líneas de nuestra programación quedarán así:

```
nave_mc.focusRect = false;
stage.focus = nave_mc;
```

Tutorial 9. Creación de un juego (I)
Paso 10 de 26

Ahora que hemos probado que el *trace* funciona, y que la nave no muestra el rectángulo del foco, procederemos a sustituir el *trace* por las sentencias que queremos ejecutar.

Queremos evaluar qué flecha se ha pulsado, y ejecutar acciones diferentes según sea el caso de haber pulsado una flecha u otra. Para este tipo de problemas es muy útil una sentencia llamada *switch*.

Supongamos el caso de un test con respuestas *a*, *b* y *c:*

```
switch (respuesta)
{
    case a :
        trace("el sujeto ha elegido a");
        break;
    case b :
        trace("el sujeto ha elegido b");
        break;
    case c :
        trace("el sujeto ha elegido c");
        break;
    default :
        trace("ha ocurrido un error");
}
```

Lo que haría en este caso la sentencia *switch* es evaluar la *respuesta* (variable que está entre paréntesis).

En el caso de que la respuesta sea *b,* se analizarán los casos hasta encontrar uno que sea correcto *(case b)*, y entonces se ejecutarán la sentencias que se encuentren dentro de ese *case* (en este caso un *trace* que informa de que se ha respondido *a*), y ya no se revisarán los demás casos (*break*).

Cada *case* debe terminar siempre con una sentencia *break* para que se omita el resto de la sentencia *switch* cuando encontremos un *case* correcto. Así evitamos que vaya comprobando el resto de los casos.

Si ningún *case* fuera correcto, entonces se ejecutará lo que se indique en *default*. Aquí las únicas respuestas posibles son *a*, *b* o *c*, por lo que si *respuesta* es cualquier otra cosa, hemos programado un *trace* que indique que ha habido un error. Aquí no haría falta un *break*.

Siempre hay que incluir *default*, aunque no preveamos que vaya a ocurrir nada fuera de los *case*.

Es importante no olvidar que esta sentencia tiene también sus propias llaves al inicio y al final.

Tutorial 9. Creación de un juego (I)
Paso 11 de 26

Vamos a evaluar qué tecla ha detectado nuestro *listener* que se ha pulsado (*e.keyCode)*. Según la tecla pulsada, rotaremos el clip hacia un lado o hacia otro. Si se ha pulsado cualquier otra tecla, no haremos nada, salvo salir de la sentencia. Nuestra sentencia, que estará dentro de la función *flechas* (sustituyendo al *trace*), quedará como sigue:

```
switch (e.keyCode)
{
    case Keyboard.RIGHT :
          e.target.rotation = 90;
          break;
    case Keyboard.LEFT :
          e.target.rotation = -90;
          break;
    case Keyboard.UP :
          e.target.rotation = 0;
          break;
    case Keyboard.DOWN :
          e.target.rotation = 180;
          break;
    default :
          break;
}
```

La rotación se mide en grados (360 grados sería una vuelta completa). El clip rotará sobre su punto de registro, que este caso está en el centro del clip.

Probamos la película **(Ctrl+Intro)**. La nave ya se orienta hacia un lado u otro dependiendo de la flecha que hayamos pulsado.

El paso siguiente será añadir sentencias para que la nave, además de rotar, se desplace. Para conseguirlo podemos utilizar sentencias similares a las que utilizamos para desplazar la nube en el tutorial 6.

Por ejemplo, podemos añadir las siguientes sentencias en cada *case*, según corresponda:

```
e.target.x += 3; //para que vaya 3px hacia la derecha
e.target.x -= 3; //para que vaya 3px hacia la izquierda
e.target.y -= 3; //para que suba 3px
e.target.y += 3; //para que baje 3px
```

El primer *case* quedaría por tanto de la siguiente forma :

```
case Keyboard.RIGHT :
    e.target.rotation = 90;
    e.target.x += 3;
    break;
```

Tutorial 9. Creación de un juego (I)
Paso 12 de 26

Completamos los cuatro *case* con las sentencias correspondientes y probamos de nuevo la película. Es conveniente probar la película con cada pequeño paso que hagamos, para poder comprobar qué hace exactamente cada parte del código que vamos añadiendo.

La nave ya se mueve hacia los cuatro lados, dependiendo de la flecha que pulsemos. Sin embargo, el movimiento no es muy fluido.

Si queremos que al pulsar una flecha, el movimiento hacia ese lado se mantenga de forma continua hasta pulsar otra flecha, entonces necesitaremos añadir un *enter frame.*

Añadimos a la nave un detector del evento *enter frame*, que llame a una función a la que llamaremos *navegar:*

```
nave_mc.addEventListener(Event.ENTER_FRAME,navegar);
```

La función *navegar* tiene que detectar el último movimiento que se ha activado, y continuar en la misma dirección. Por ejemplo, en el caso de que el movimiento sea hacia la derecha, entonces continuar avanzando hacia la derecha.

Vamos a crear una variable llamada *movimiento*, que almacene un texto con la dirección del movimiento.

Las variables se crean con la siguiente estructura:

```
var nombreVariable:tipoVariable = valor inicial;
```

El valor inicial es opcional. Podría poner simplemente un punto y coma tras la declaración de la variable. Por legibilidad del código, es conveniente declarar las variables que vayamos a utilizar en la parte superior del código.

Por lo tanto, añadiremos esta primera línea a nuestro código:

```
var movimiento:String = "";
```

El tipo de variable *String* representa cadenas de caracteres. Las cadenas de caracteres siempre van entre comillas. Como valor inicial hemos creado una cadena vacía (sin ningún contenido).

Tutorial 9. Creación de un juego (I)
Paso 13 de 26

Como ahora queremos que los desplazamientos de la nave los haga otra función que recibe un *enter frame* (la función navegar), en la función *flecha* sólo guardaremos el dato sobre hacia dónde se debe dirigir el movimiento.

Por lo tanto, sustituimos las sentencias *case* del desplazamiento en la función *flechas* de la siguiente manera:

```
case Keyboard.RIGHT :
	e.target.rotation = 90;
	movimiento = "derecha";
	break;
```

En los demás casos, asignamos a la variable *movimiento* los valores *"izquierda", "arriba"* y *"abajo"* respectivamente.

La función *navegar*, que crearemos a continuación, moverá de forma continua el clip hacia un lado u otro dependiendo del último valor asignado a la variable *movimiento*, ya que se ejecutará continuamente al ser llamada por un evento *enter frame*.

La función *navegar* evaluará el contenido de la variable *movimiento*, y en base a eso moverá la nave.

```
function navegar(e:Event):void
{
	switch (movimiento)
	{
		case "derecha" :
			e.target.x += 3;
			break;
		case "izquierda" :
			e.target.x -= 3;
			break;
		case "arriba" :
			e.target.y -= 3;
			break;
		case "abajo" :
			e.target.y += 3;
			break;
		default :
			break;
	}
}
```

El valor *3* es la cantidad de píxeles que se desplaza la nave en cada enter frame. Es, por tanto, la velocidad de la nave..

Tutorial 9. Creación de un juego (I)
Paso 14 de 26

Podríamos por tanto crear una variable llamada *velocidad,* que especifique el número de píxeles en que varía la posición de la nave en cada momento.

Podemos crear esta variable en la parte superior de la programación, tras la declaración de la variable *movimiento*:

```
var movimiento:String = "";
var velocidad:Number = 3;
```

Dentro de la función *navegar*, sustituimos el número de píxeles por la variable *velocidad* en los cuatro *case*, que tomará el valor que le hemos asignado cuando creamos la variable:

```
case "derecha" :
      e.target.x += velocidad;
      break;
```

De esta forma, si cambiamos el valor de la variable velocidad, cambiará la velocidad de los desplazamientos en las cuatro direcciones. Podemos probar diferentes valores hasta que nos parezca una velocidad adecuada.

El siguiente paso va a ser detectar si la nave llega al planeta, que será la meta del juego. Para ello, mientras la nave se desplaza debemos ir comprobando si choca con el planeta.

Dentro de la función navegar, por debajo de la llave de cierre de la sentencia *switch*, añadiremos este código:

```
if (e.target.hitTestObject(planeta_mc))
{
      trace("He llegado al planeta. He ganado.");
}
```

Esto es un condicional que comprueba si la nave (*e.target*) choca *(hitTestObject)* con el planeta (*planeta_mc)*. Si choca, entonces se mostrará la frase *He llegado al planeta* en el panel Salida.

Vamos a añadir otro condicional, esta vez para detectar si la nave se sale fuera de los límites del escenario, que mide 550 x 400. Por lo tanto deberemos comprobar si la propiedad *x* (posición horizontal) de la nave tiene un valor inferior a 0 o superior a 550, y si la propiedad *y* (posición vertical) tiene un valor por debajo de 0 o por encima de 400. En cualquiera de los cuatro casos, la nave estará posicionada fuera del escenario.

Tutorial 9. Creación de un juego (I)
Paso 15 de 26

Este nuevo condicional, escrito también dentro de la función *navegar*, quedará de la siguiente manera (las barras verticales ||, que se escriben con *Alt Gr+1*, equivalen al *OR* lógico):

```
if (e.target.x < 0 || e.target.x > 550 || e.target.y < 0
|| e.target.y > 400)
{
      trace("Estoy fuera del escenario. He perdido.");
}
```

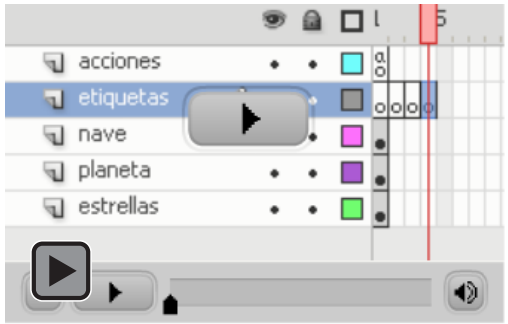
Probamos la película para comprobar su correcto funcionamiento. La nave se desplazará por el escenario dependiendo de las flechas que pulsemos en el teclado. Si llegamos al planeta, aparecerá la frase de que hemos ganado en el panel Salida, y si estamos fuera de los límites del escenario, aparecerá la frase de que hemos perdido.

Ahora el juego comienza directamente, pero sería adecuado disponer de una pantalla previa con las instrucciones del juego, y que al llegar al planeta (ganar) o salirnos del escenario (perder) se mostrara una pantalla diferente y el juego terminara, dando opción a volver a jugar.

Vamos por tanto a crear cuatro fotogramas diferentes:
-un fotograma inicial con dos botones, uno con las instrucciones del juego y otro para comenzar a jugar.
-un segundo fotograma en el que se desarrollará el juego.
-un tercer fotograma indicando que hemos ganado.
-un último fotograma indicando que hemos perdido.

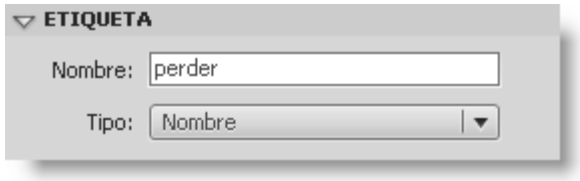
Estos dos últimos fotogramas tendrán un botón para volver a jugar.

Insertamos una nueva capa a la que llamaremos *etiquetas*. Insertamos **fotogramas clave** en los fotogramas 2, 3 y 4. Podemos crear estos fotogramas clave fácilmente si hacemos clic sobre cada fotograma y pulsamos la tecla **F6**.



Tutorial 9. Creación de un juego (I)
Paso 16 de 26

Pulsamos sobre cada fotograma de la capa *etiquetas*, y asignamos a cada uno un nombre de etiqueta en el **inspector de Propiedades**. Al número 1 le llamaremos *inicio*, al 2 *juego*, al 3 *ganar*, y al 4 *perder*.



Los fotogramas que tengan una etiqueta mostrarán una pequeña bandera roja en la línea de tiempo.

La nave y el planeta tendrán que mantenerse visibles y sin cambios en los fotogramas *inicio* y *juego*, mientras que las estrellas se verán como fondo en los 4 fotogramas. En los fotogramas *ganar* y *perder* utilizaremos de nuevo una imagen del planeta, pero esta vez lo situaremos a tamaño mayor y en el centro de la pantalla.

Por lo tanto insertamos los fotogramas correspondientes pulsando **F5**, y después insertaremos un fotograma clave con **F6** para hacer un cambio en el tamaño planeta que se mantendrá en los fotogramas 3 y 4.

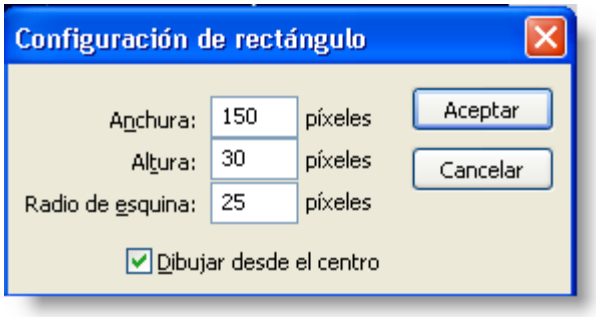
Tutorial 9. Creación de un juego (I)
Paso 17 de 26

Cambiamos la posición y tamaño del planeta en el fotograma 3, posicionándolo en el centro del escenario y a mayor tamaño.

Creamos una **nueva capa** llamada *texto*. En el fotograma 3 de esta nueva capa creamos un **fotograma clave (F6)**. Con la **herramienta Texto** creamos en este fotograma un **campo de texto estático** con el texto *has ganado!!!*. En el fotograma 4 de la capa *texto* creamos un nuevo fotograma clave (F6), y cambiamos el texto a *has perdido!!!*

Creamos una **nueva capa** llamada *botones*. Para crear un rectángulo que después será el fondo del botón, seleccionamos la **herramienta Rectángulo**, sin trazo y con color de relleno rojo. Con la tecla **Alt** pulsada hacemos clic sobre el escenario.

En la configuración del rectángulo, establecemos una **anchura** de 150 px, **altura** de 30 px, y **radio de esquina** de 25 px, para que aparezcan las esquinas redondeadas.



Tutorial 9. Creación de un juego (I)
Paso 18 de 26

Seleccionamos el rectángulo rojo que hemos creado, y pulsamos **F8** para **convertirlo en símbolo**. Le damos el nombre *comenzar*, y seleccionamos que sea tipo **Botón**.

Añadimos una nueva capa sobre el fondo del botón, en la que situaremos el texto del botón.

Con la **herramienta Texto** creamos un campo de texto estático, y escribiremos *comenzar* en color blanco. Centramos el campo de texto en el rectángulo de fondo.

Este texto se mantendrá tanto en el fotograma *Reposo* como en el fotograma *Sobre*.

Hacemos doble clic sobre el botón en el escenario para editarlo.

Insertamos un fotograma clave pulsando **F6** en el estado *Sobre*, y cambiamos el color del rectángulo a azul en el **inspector de Propiedades**.

Volvemos a la escena principal, y situamos el botón en la parte inferior derecha del escenario. En el **inspector de Propiedades** le asignamos el **nombre de instancia** *comenzar_btn.*

Tutorial 9. Creación de un juego (I)
Paso 19 de 26

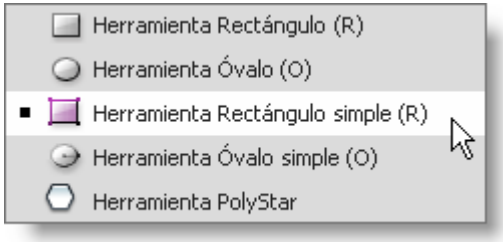
Creamos un nuevo botón llamado *instrucciones* con las mismas características que el anterior, pero esta vez con el texto *instrucciones*. Lo situamos en el escenario sobre el botón *comenzar*.

Vamos a hacer que cuando situemos el puntero sobre el botón *instrucciones*, aparezcan directamente las instrucciones del juego. La forma más sencilla de conseguir este efecto es añadiendo el contenido que queremos mostrar en el fotograma *Sobre* del botón.

Hacemos doble clic sobre el botón *instrucciones* para añadir algunos cambios a su fotograma *Sobre*.

Añadimos una nueva capa en la línea de tiempo del botón, e insertamos un **fotograma clave** en el estado *Sobre*.

Seleccionamos la **herramienta Rectángulo simple**, para poder efectuar modificaciones en las características del rectángulo después de dibujarlo.



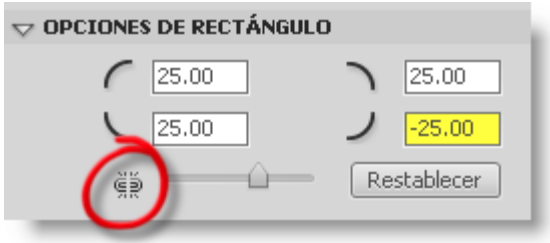
En el fotograma *Sobre* de esta nueva capa que hemos creado, dibujamos un rectángulo con el mismo color azul que habíamos utilizado para el botón.

Tutorial 9. Creación de un juego (I)
Paso 20 de 26

Con el rectángulo recién creado seleccionado en el escenario, cambiamos el valor de sus esquinas en el **inspector de Propiedades**.

Para poder asignar un valor diferente a una esquina, debemos en primer lugar desactivar el bloqueo del radio de las esquinas.

Después asignamos un valor de 25 a todas las esquinas excepto a la esquina inferior derecha, a la que asignaremos un valor de -25. De esta forma conseguiremos una esquina con el radio inverso.

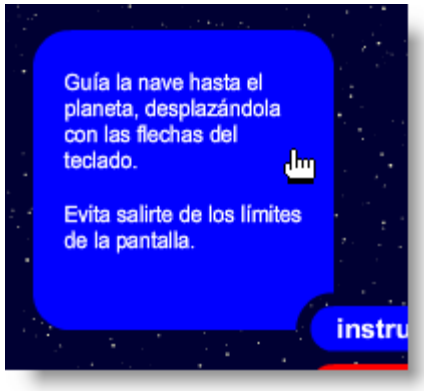


Creamos otra capa por encima de este rectángulo para escribir el texto de las instrucciones. Insertamos un fotograma clave en el estado *Sobre* de esta capa.

El texto, en color blanco para destacar sobre el recuadro azul, podría ser algo similar a lo siguiente: *Guía la nave hasta el planeta, desplazándola con las flechas del teclado. Evita salirte de los límites de la pantalla.*

Volviendo a la escena principal, seleccionamos **Control > Habilitar botones simples** para poder comprobar el efecto de situar el puntero sobre un botón sin necesidad de probar la película.

Podemos comprobar que si situamos el puntero sobre el área donde se muestran las instrucciones, el botón también se activa.



Sin embargo, queremos que sólo se muestren las instrucciones si nos situamos sobre el rectángulo rojo del botón instrucciones en su estado de reposo.

Tutorial 9. Creación de un juego (I)
Paso 21 de 26

Seleccionamos de nuevo **Control > Habilitar botones simples** para desactivar los botones, y así poder hacer doble clic sobre el botón *instrucciones* para editarlo.

Insertamos fotogramas de tal forma que en la zona activa se muestre el fondo del botón, es decir, el rectángulo redondeado que servía de base al botón.

Insertamos fotogramas en el resto de las capas para que se muestren también en el estado *Presionado,* y así evitar que en ese estado se muestre sólo el fotograma del fondo del botón.

Si ahora habilitamos de nuevo los botones simples para hacer la prueba, veremos que el botón se comporta de la manera esperada.

Esta forma de utilizar el estado *Sobre de un botón*, también llamado *rollover*, puede resultar muy útil para crear aplicaciones educativas, ya que podemos mostrar diferentes tipos de información al situarnos sobre un área concreta.

Tutorial 9. Creación de un juego (I)
Paso 22 de 26

Volvemos a la escena principal. Los botones *instrucciones* y *comenzar* sólo deben verse en el primer fotograma, ya que durante el juego no deben estar visibles.

Creamos el botón que nos falta, al que llamaremos *repetir*. Será un botón igual que *comenzar*, salvo que en el texto escribiremos *jugar de nuevo*. Este botón deberá estar visible en los fotogramas *ganar* y *perder* (3 y 4).

Colocamos este tercer botón en el tercer fotograma de la capa *botones*, en la misma posición que tenía el botón *comenzar* en el primer fotograma y le damos el nombre de instancia *repetir_btn*. Al botón *instrucciones* no le dimos nombre de instancia porque no va a llevar asociada ninguna programación.

Insertamos un fotograma en blanco en el segundo fotograma (en el que se desarrolla el juego), para que en ese fotograma no se muestre ningún botón.

Repasando el contenido que hemos creado, en el primer fotograma (*inicio*) se verán, además de la nave y el planeta, los botones *instrucciones* y *comenzar*. En el segundo fotograma (*juego)* sólo veremos la nave y el planeta. En el tercer fotograma (*ganar)* y en el cuarto fotograma (*perder)*, veremos el botón *repetir*, así como una imagen ampliada del planeta en el centro del escenario, con diferentes textos según sea el caso de ganar o perder. Las estrellas serán el fondo común de los cuatro fotogramas.

Tutorial 9. Creación de un juego (I)
Paso 23 de 26

Ahora que tenemos los botones creados, necesitamos programar su funcionamiento. Volvemos al fotograma 1 de la capa *acciones* y pulsamos **F9** para abrir el **panel Acciones**.

En primer lugar, al iniciar nuestra película queremos que se detenga en el primer fotograma, por lo que incluiremos la instrucción `stop();`. Podemos colocar esta instrucción por ejemplo en la primera línea de nuestra programación.

El movimiento de la nave ahora no debe permitirse directamente, sino que debe ser activado cuando pulsemos el botón *comenzar_btn.* Por otro lado, el botón *comenzar_btn* también debe llevarnos al fotograma 2, etiquetado como *juego*, en el que no se muestra ningún botón.

Añadimos por tanto un *listener* al botón *comenzar_btn*, de tal forma que ejecute una función a la que llamaremos *jugar* cuando clicemos sobre él.

```
comenzar_btn.addEventListener(MouseEvent.CLICK,jugar);
```

Las sentencias que habíamos creado para seleccionar (poner el foco) y añadir los *listeners* a la nave deben estar ahora dentro de la función *jugar.*

La función *jugar* quedará por tanto de la siguiente manera:

```
function jugar(e:MouseEvent):void
{
   gotoAndStop("juego");
   nave_mc.focusRect = false;
   stage.focus = nave_mc;
   nave_mc.addEventListener(KeyboardEvent.KEY_DOWN,flechas);
   nave_mc.addEventListener(Event.ENTER_FRAME,navegar);
}
```

Para el botón instrucciones no hay programación, ya que su función es mostrar la ayuda cuando situamos el puntero sobre el botón, y eso lo hemos conseguido mostrando la ayuda en el estado *Sobre* del botón.

Cambiamos ahora las sentencias *trace* de la función *navegar* por las sentencias siguientes:

```
gotoAndStop("ganar"); //si llegamos al planeta
gotoAndStop("perder"); //si nos salimos del escenario
```

Si no hubiéramos puesto etiquetas en los fotogramas, las instrucciones serían `gotoAndStop(3)` en un caso y `gotoAndStop(4)` en el otro caso.

Tutorial 9. Creación de un juego (I)
Paso 24 de 26

Aunque todavía nos falta por programar el botón para volver a jugar, si probamos ahora la película y llegamos al planeta, comprobaremos que después de mostrar el fotograma de que hemos ganado, al poco tiempo se mostrará el fotograma de que hemos perdido.

Esto se debe a que, pese a que no vemos la nave en los fotogramas 3 y 4, no hemos eliminado los *listeners*, por lo que se siguen calculando posiciones para la nave. Es decir, que después de llegar al planeta, la posición de la nave saldrá por un extremo del escenario, dando lugar a que veamos el fotograma de que hemos perdido.

Para solucionar este problema, antes del *gotoAndStop* que envía la cabeza lectora a los fotogramas *ganar* o *perder*, tenemos que eliminar los *listeners* que habíamos creado para la nave.

```
nave_mc.removeEventListener(KeyboardEvent.KEY_DOWN,flechas);
nave_mc.removeEventListener(Event.ENTER_FRAME,navegar);
```

Después de la sentencia del *gotoAndStop,* que enviará la cabeza lectora a los fotogramas del final, podemos añadir el *listener* para habilitar el botón *repetir*, ya que es entonces cuando el botón aparecerá en escena:

```
repetir_btn.addEventListener(MouseEvent.CLICK,repetir);
```

Por tanto, los pasos que ocurrirán tanto si ganamos como si perdemos, será remover en primer lugar los *listeners* de la nave, después ir al fotograma *ganar* o *perder*, según sea el caso, y por último añadir un *listener* al botón *repetir*.

Salvo el nombre del fotograma al que dirigirnos, las instrucciones que damos son las mismas en ambos casos (ganar o perder). Para no repetir la programación en dos lugares diferentes (en los dos condicionales *if* dentro de *navegar)*, lo más adecuado en estos casos es crear una función independiente que contenga estas sentencias.

Para solucionar el hecho de que haya que ir a un fotograma diferente en cada caso, podemos añadir un parámetro a la función, de tal forma que reciba el nombre del fotograma al que tiene que ir.

Así podríamos incluir en la llamada a la función, a la que llamaremos *juegoAcabado*, el nombre del fotograma de la siguiente manera (lo mismo pero con *perder* en el otro caso):

```
if (e.target.hitTestObject(planeta_mc))
{
    juegoAcabado("ganar");
}
```

Tutorial 9. Creación de un juego (I)
Paso 25 de 26

La función, que recibe un parámetro con el nombre del fotograma, tendrá esta definición:

```
function juegoAcabado(nombreFotograma:String):void
{
  nave_mc.removeEventListener(KeyboardEvent.KEY_DOWN,flechas);
  nave_mc.removeEventListener(Event.ENTER_FRAME,navegar);
  gotoAndStop(nombreFotograma);
  repetir_btn.addEventListener(MouseEvent.CLICK,repetir);
}
```

Como vemos, al parámetro que recibe la función le hemos llamado *nombreFotograma*, y a la sentencia *gotoAndStop* le indicamos que vaya a un fotograma con el valor del parámetro recibido (en nuestro caso los valores recibidos son *"ganar"* o *"perder"*).

La función *repetir*, que será llamada cuando pulsemos el botón *repetir_btn*, tan sólo indicará que volvamos al fotograma inicial:

```
function repetir(e:MouseEvent):void
{
   gotoAndStop("inicio");
}
```

Resumiendo la programación creada en la primera parte del juego, tenemos en primer lugar un *stop* y la creación de las variables *movimiento* y *velocidad*.

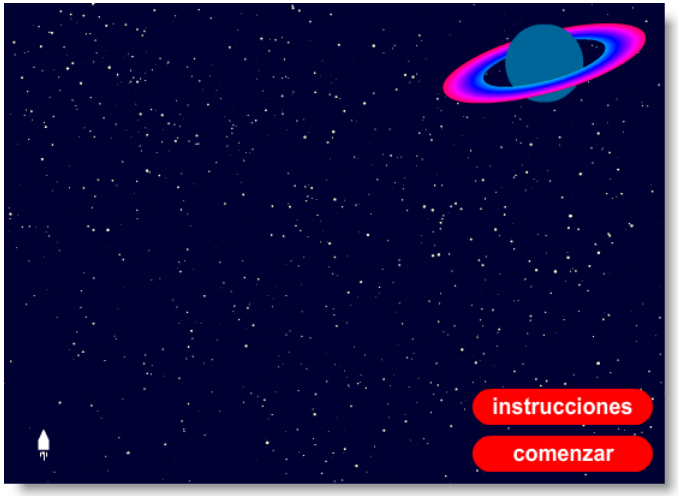
Después añadimos un *listener* al botón *comenzar_btn*.

Por último tenemos creadas cinco funciones, que serán llamadas en diferentes momentos. Las funciones que hemos creado son:
-*jugar* (activa el movimiento de la nave).
-*flechas* (rota la nave e indica la dirección del movimiento).
-*navegar* (desplaza la nave y evalúa si ha llegado al planeta o ha salido fuera del escenario).
-*juegoAcabado* (borra los *listeners,* va a la pantalla final y prepara el botón repetir).
-*repetir* (va al fotograma inicial del juego).

En el siguiente tutorial añadiremos complejidad al juego añadiendo obstáculos que se moverán de forma aleatoria por el escenario.

Tutorial 9. Creación de un juego (I)

Paso 26 de 26



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Cread en un nuevo documento un dibujo de un cuerpo humano utilizando botones, de tal forma que cuando situemos el puntero sobre cada área del cuerpo, el área quede resaltada y se muestre un recuadro con información sobre esa área.
2. Cread un nuevo juego con diferentes niveles, en el que podamos mover un objeto con el teclado y al llegar a un objetivo pasemos a otra pantalla. En cada pantalla nueva aumentamos la velocidad.

Tutorial 10. Creación de un juego (II)

Paso 1 de 22

Continuando con el juego que comenzamos en el anterior tutorial, vamos a completar su desarrollo añadiendo obstáculos que la nave tiene que evitar.

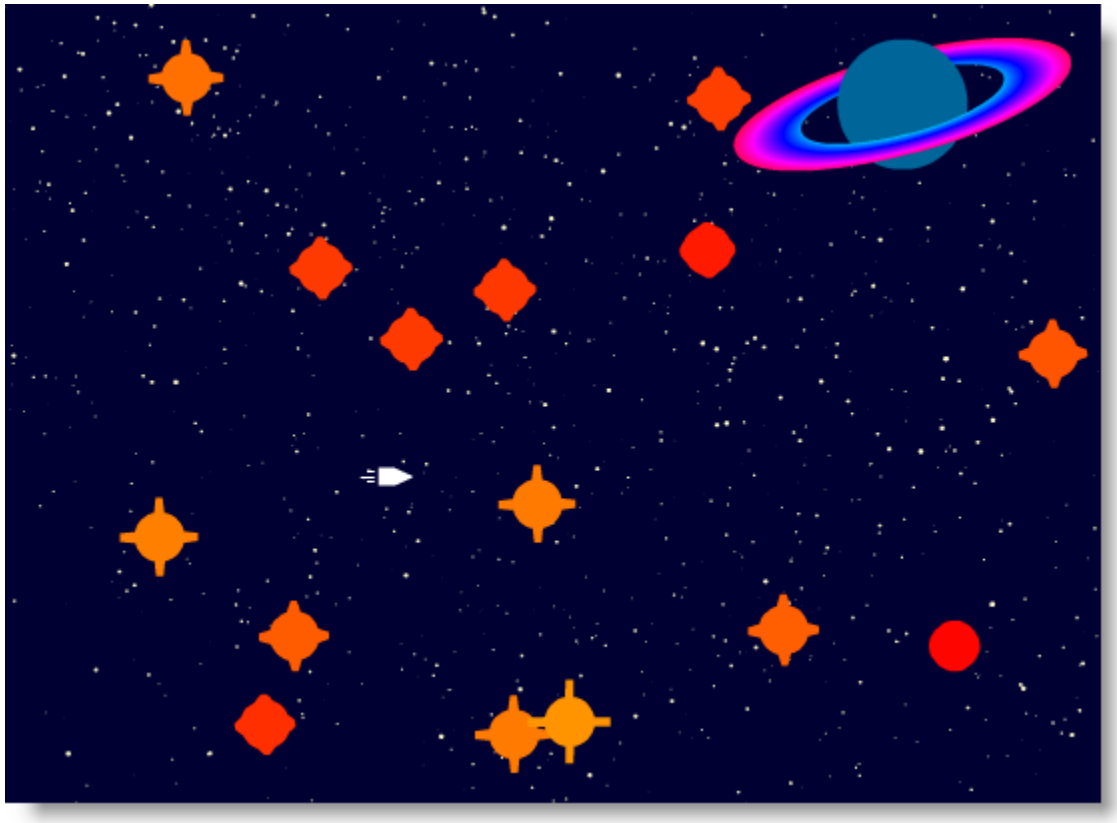
En primer lugar, crearemos para los obstáculos una animación utilizando una interpolación de forma.

Aprenderemos a añadir objetos dinámicamente desde la biblioteca, así como a posicionarlos en el escenario aleatoriamente.

También aprenderemos a generar movimiento aparentemente errático.

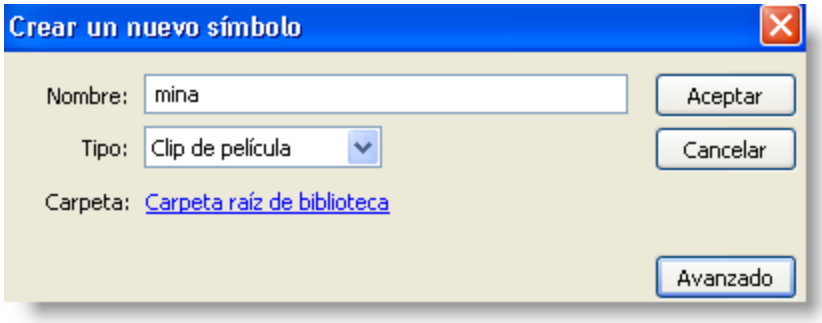
Utilizaremos sonidos desde la biblioteca sin necesidad de añadirlos al escenario.

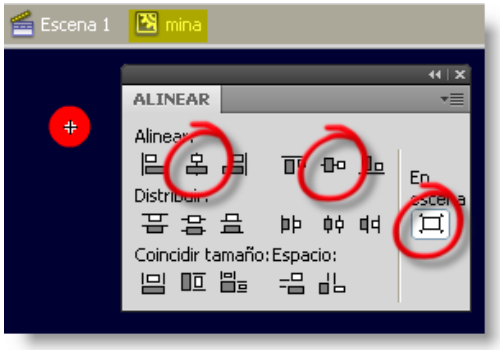
Finalmente, permitiremos al jugador elegir el número de obstáculos que aparecerán en el juego utilizando un campo de introducción de texto.



Tutorial 10. Creación de un juego (II)
Paso 2 de 22

Vamos a crear en primer lugar los obstáculos que tendrá que sortear la nave. Seleccionamos **Insertar > Nuevo símbolo**. Damos al símbolo el nombre *mina* y como tipo seleccionamos **Clip de película**.





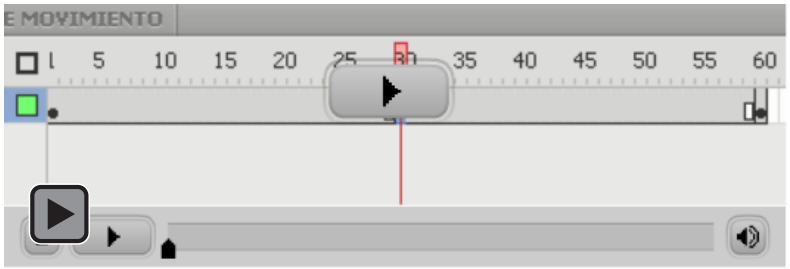
En la línea de tiempo de este nuevo clip, dibujaremos un pequeño círculo rojo de 25 x 25 px, sin trazo y con relleno rojo.

Con el **panel Alinear**, centramos este círculo en su escenario.

Este clip tendrá una sencilla animación en la que cambiaremos su forma y su color, para después volver a su forma original.

La animación durará 60 fotogramas. Insertamos un **fotograma clave** pulsando **F6** en el *fotograma 60*. De esta forma se copiará el contenido del fotograma 1 en el fotograma 60.

Insertamos otro **fotograma clave** en el *fotograma 30*. También se copiará el contenido del fotograma 1, pero la intención en este caso es modificar en este fotograma la forma del círculo original.

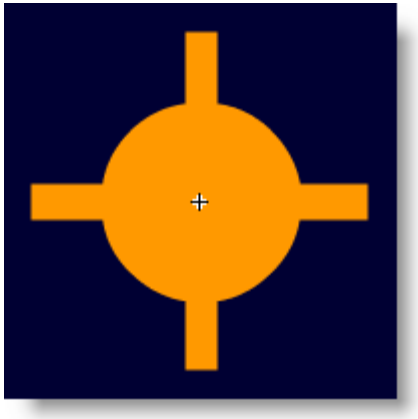


Tutorial 10. Creación de un juego (II)
Paso 3 de 22

Con el fotograma clave 30 seleccionado, dibujamos dos rectángulos, uno vertical y otro horizontal, que centraremos a medida que los dibujemos. Podemos trabajar con zoom para mayor comodidad.

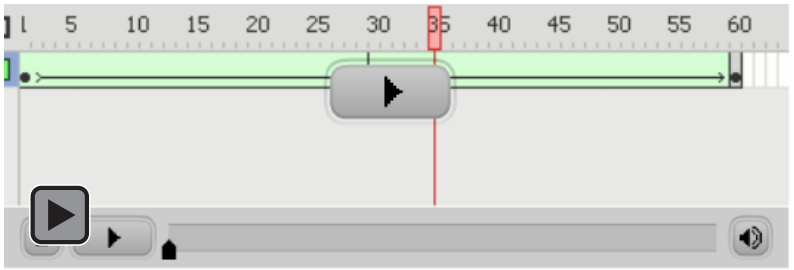
Estos rectángulos tampoco tendrán trazo y tendrán como relleno el mismo color rojo que el círculo. De esta manera, los rectángulos quedarán unidos al círculo en una sola forma.

Por último cambiamos el color de la forma compuesta a un tono anaranjado.



Para crear una transición desde la forma del círculo original al círculo con los rectángulos, y otra transición hasta volver de nuevo a la forma y color originales, tenemos que crear interpolaciones de forma.

Para ello clicamos con el **botón derecho** del ratón sobre el área gris que hay entre dos fotogramas clave, y seleccionamos en el menú contextual **Crear interpolación de forma**. Repetimos el proceso para la transición comprendida entre los otros dos fotogramas clave.



Podemos crear interpolaciones de forma entre dos formas cualesquiera. Para controlar cambios de forma complejos, podemos utilizar los consejos de forma, aunque en nuestro caso no va a ser necesario. Consulta la ayuda de Flash a este respecto si necesitas crear interpolaciones de forma complejas.

Tutorial 10. Creación de un juego (II)
Paso 4 de 22

Si pulsamos **Intro** o movemos la cabeza lectora, veremos que la forma cambia del círculo rojo inicial, a la forma compuesta anaranjada, para volver posteriormente al círculo rojo.

Podéis experimentar con diferentes formas o colores hasta crear una interpolación de forma que os guste más.

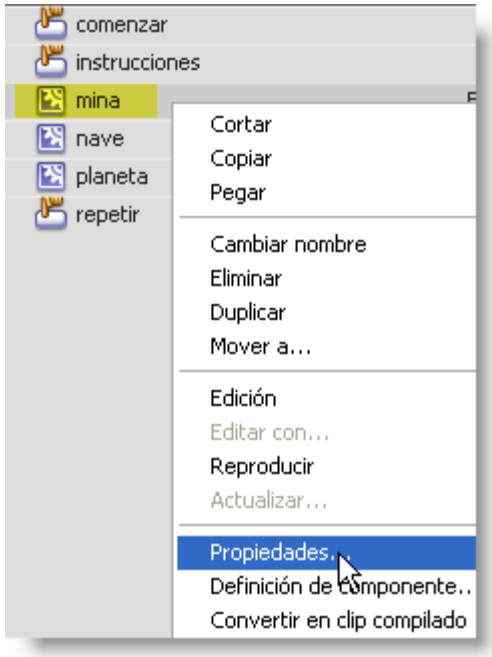
Con esto damos por finalizada la creación de la mina, por lo que volvemos a la escena principal.

Vamos a añadir copias al escenario del clip mina utilizando ActionScript.

Hasta ahora, para poder hacer referencia a un objeto utilizábamos en la programación su nombre de instancia. Para asignar un nombre de instancia a un clip, seleccionábamos el clip en el escenario y escribíamos su nombre de instancia en el inspector de Propiedades.

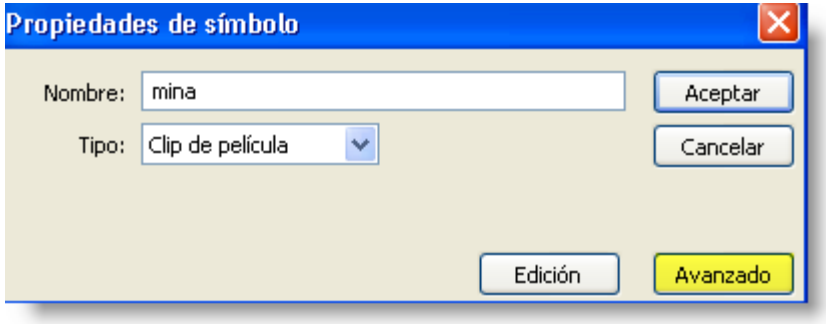
En el caso en el que el objeto no esté en el escenario, tenemos que exportarlo para ActionScript desde la biblioteca.

Hacemos clic con el **botón derecho** del ratón sobre el nombre del clip (*mina* en este caso), y seleccionamos **Propiedades** en el menú contextual.

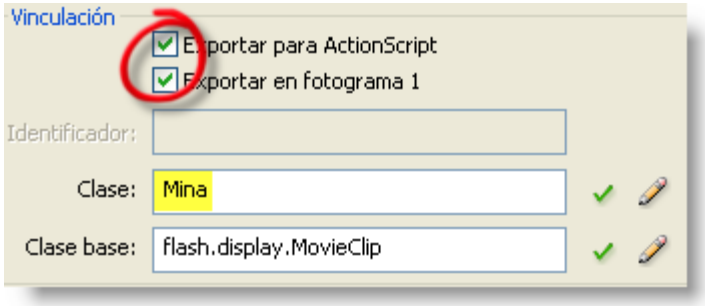


Tutorial 10. Creación de un juego (II)
Paso 5 de 22

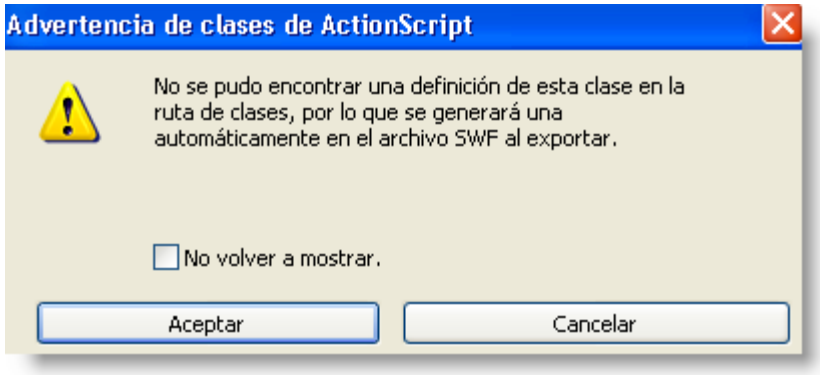
En la ventana **Propiedades de símbolo**, pulsamos sobre el botón **Avanzado**.



En el área **Vinculación** marcamos las casillas **Exportar para ActionScript** y **Exportar en fotograma 1**. Escribimos el nombre de clase ***Mina*** (los nombres de las clases comienzan con mayúscula).



Al pulsar en **Aceptar** nos aparecerá una ventana de advertencia. Pulsamos de nuevo en Aceptar para que se genere de forma automática la definición de la clase Mina.



Después de haber exportado el clip en una clase llamada Mina, ya podemos hacer referencia a esta clase en la programación

Tutorial 10. Creación de un juego (II)
Paso 6 de 22

Vamos a definir una nueva función a la que llamaremos *crearMinas*.

```
function crearMinas():void
{
      var mina:Mina = new Mina;
      mina.x = 275;
      mina.y = 200;
      stage.addChild(mina);
}
```

En la primera línea de esta función, creamos un nuevo objeto llamado *mina* que pertenece a la clase *Mina,* y que será una nueva copia del clip que tenemos en la biblioteca. Después de esta definición, ya podemos hacer referencia al nuevo objeto creado.

En las siguientes líneas, definimos que la posición del objeto sea el centro del escenario (*x:275* e *y:200*). Por último, añadimos ese objeto al escenario *(stage)* utilizando el método *addChild*.

Para que esta función se ejecute, en algún momento tenemos que llamarla con la instrucción `crearMinas();`. Escribiremos esta instrucción dentro de la función *jugar*, ya que será el momento en el que queremos que se añadan las minas al escenario.

Si probamos la película, comprobaremos que al pulsar el botón *comenzar* (que llama a la función *jugar*, que a su vez llama a la función *crearMinas*), aparece una mina en el centro del escenario.

Para que la posición en la que aparece la mina sea aleatoria, vamos a utilizar el método *random* de la clase *Math*. *Math.random()* devuelve un número aleatorio comprendido entre 0 y 1. Si multiplicamos ese valor aleatorio por 550 (la anchura del escenario), el número devuelto estará comprendido entre 0 y 550.

Por lo tanto, para crear una posición aleatoria de la mina dentro del escenario, que mide 550 x 400, las líneas que determinan la posición del clip dentro de la función *crearMinas* quedarán como sigue:

```
mina.x = Math.random() * 550;
mina.y = Math.random() * 400;
```

La posición *x* del clip será un número comprendido entre 0 y 550, mientras que la posición *y* será un número comprendido entre 0 y 400. Cada vez que se ejecuta la función *Math.random()*, el número devuelto será diferente.

Tutorial 10. Creación de un juego (II)
Paso 7 de 22

Por ahora sólo se añade una copia del clip al escenario. Para añadir varias copias, debemos hacer un bucle que cree varias minas y las añada al escenario.

Vamos a definir, junto con las variables *movimiento* y *velocidad*, una nueva variable llamada *numMinas*, cuyo valor será el número de minas que queremos crear (por ejemplo 10).

```
var numMinas:Number = 10;
```

Ahora utilizaremos un bucle *for* para crear estas 10 copias de la mina.

Los bucles *for* tienen la siguiente estructura:

```
for (valor inicial; condición; cambio en el valor)
{
   //sentencias a ejecutar
}
```

Por ejemplo, en nuestro caso:

```
for (var i:Number = 0; i < numMinas; i++)
{
   //sentencias a ejecutar
}
```

Este bucle funcionaría de la siguiente manera:

•Creamos una variable llamada *i* con un valor inicial de 0.

•Comprobamos si se cumple la condición, que en este caso es que el valor de *i* sea menor que el valor de *numMinas.*

•Al cumplirse la condición de que *i* < *numMinas*, ejecutaremos las sentencias que se encuentren entre las llaves del bucle *for.*

•Aumentamos el valor de *i* en una unidad *(i++ significa i = i + 1*).

•Volvemos a comprobar la condición. Ahora *i* vale 1, que sigue siendo menor que 10 (valor de *numMinas*).

•Como la condición se sigue cumpliendo, volvemos a ejecutar las sentencias, y sumamos otra unidad a *i*, que ahora valdrá 2.

•Cuando *i* valga 10, momento en el que no se cumplirá que *i* sea menor que *numMinas*, ya no se ejecutará el bucle. Al empezar con un valor de *i=0*, el bucle se habrá ejecutado un total de 10 veces.

Tutorial 10. Creación de un juego (II)
Paso 8 de 22

Por lo tanto, para crear la cantidad de minas que hayamos indicado en la variable *numMinas*, la función *crearMinas* quedará como sigue:

```
function crearMinas():void
{
    for (var i:Number = 0; i < numMinas; i++)
    {
        var mina:Mina = new Mina  ;
        mina.x = Math.random() * 550;
        mina.y = Math.random() * 400;
        stage.addChild(mina);
    }
}
```

Si probamos ahora la película, podemos comprobar que aparecen 10 minas en el escenario.

Todas las minas realizan su animación al mismo tiempo. Podemos hacer que cada mina comience su animación en un fotograma diferente, llevando su cabeza lectora a un fotograma aleatorio entre el 1 y el 60, que es el número de fotogramas que tiene la animación.

Esta vez necesitamos por tanto generar un número aleatorio entre 1 y 60. El número devuelto tiene en este caso que ser entero.

Si utilizamos *Math.random() * 60* obtendremos números decimales entre 0 y 60. Para asegurarnos de que el número resultante sea un entero comprendido entre 1 y 60 podemos utilizar el método *ceil*, que redondea al alza un número decimal.

Por lo tanto, dentro del bucle *for*, después de haber determinado una posición aleatoria para cada mina y antes de añadir la mina al escenario, escribiremos la instrucción:

```
mina.gotoAndPlay(Math.ceil(Math.random() * 60));
```

Probamos de nuevo la película. Ahora, al comenzar el juego, se crean 10 copias de la mina, comenzando la animación de cada una de ellas en momentos diferentes.

Tutorial 10. Creación de un juego (II)
Paso 9 de 22

Para convertir las minas en obstáculos contra los que debemos evitar chocar, añadiremos un detector a cada mina, para que evalúe en cada momento si está chocando con la nave.

Para ello añadimos en primer lugar *listener* para cada mina en el mismo bucle que utilizamos para crearlas, dentro por tanto de la función *crearMinas* y dentro del bucle *for,* y antes de añadir las minas al escenario con *addChild*.

```
mina.addEventListener(Event.ENTER_FRAME,enemigos);
```

La nueva función, a la que hemos llamado *enemigos*, comprobará si cada mina a la que hemos añadido el *listener* choca con la nave en algún momento. Si choca, entonces ejecutaremos la función juegoAcabado, con el parámetro *"perder"*.

```
function enemigos(e:Event):void
{
    if (e.target.hitTestObject(nave_mc))
    {
        juegoAcabado("perder");
    }
}
```

Si probamos ahora nuestro juego, tanto si ganamos como si perdemos, nos aparecerá un error debido a que, a pesar de que no hay ninguna nave, la función *enemigos* seguirá comprobando si cada mina choca con la nave. Al no encontrar ninguna nave en el escenario se mostrará el error.

Por tanto, lo primero que debemos hacer en la función *juegoAcabado*, antes de la orden de ir a otro fotograma en el que no esté la nave, es eliminar los *listeners* que hemos añadido a las minas. A la vez que eliminamos cada *listener*, aprovecharemos para eliminar las minas del escenario. En el siguiente paso mostraremos cómo hacerlo.

El escenario (*stage)* funciona como un contenedor que contiene en primer lugar la línea de tiempo, y contiene también cada mina que le hemos ido añadiendo con *addChild*. Después de crear todas las minas, el *stage* tendrá 11 elementos secundarios. El primero, en el nivel inferior de visualización (nivel 0), será la línea de tiempo. Después estarán todas las minas, cada una en un nivel superior a la anterior.

La cantidad de minas creadas depende del valor que hayamos dado a *numMinas*. La última mina creada se mostrará por tanto en un nivel de visualización que coincide con *numMinas*.

Tutorial 10. Creación de un juego (II)
Paso 10 de 22

Al comienzo de la función *juegoAcabado*, antes de las sentencias en las que eliminábamos los *listeners* de la nave, podemos eliminar el *listener* de cada mina, y después eliminar cada mina en sí, con el siguiente código:

```
for (var i:Number = numMinas; i > 0; i--)
{
   stage.getChildAt(i).removeEventListener(Event.ENTER_FRAME,enemigos);
   stage.removeChildAt(i);
}
```

Analicemos lo que realiza este bucle. El bucle comenzará con un valor de *i* igual a *numMinas*, que en nuestro caso es 10. Como la variable *i* es mayor que 0, se ejecutarán las sentencias que después explicaremos. Después reduciremos en una unidad el valor de *i* (*i*-- significa *i = i – 1*). Por tanto, *i* valdrá 10 en la primera ejecución del bucle, 9 en la siguiente vuelta, etc., hasta el último valor que cumple la condición, es decir, hasta que *i* valga 1 (cumple *i > 0*).

En las sentencias que se encuentran dentro del bucle, en primer lugar eliminamos el *listener* del elemento que se encuentra contenido dentro del *stage*, en el nivel de visualización *i (10, 9, 8, 7,…, 1),* que en el nuestro es cada mina que hemos creado. De esta forma ya no tratarán de detectar posibles colisiones con la nave. Con la siguiente línea eliminamos la mina a la que acabamos de eliminar su *listener*.

Ahora nuestro juego será funcional y no tendrá errores, pero vamos a introducir mejoras en los siguientes pasos:

•En primer lugar, nos aseguraremos de que la posición inicial de las minas no esté muy cerca de la nave, para que nos dé tiempo a empezar a jugar.

•Para hacer el juego más complejo, añadiremos un movimiento aleatorio de las minas por el escenario.

•Cambiaremos la imagen del planeta por la imagen de una mina en el fotograma *perder*.

•Después añadiremos un sonido para cuando ganemos y otro para cuando perdamos.

•Por último, permitiremos al usuario decidir el número de minas que quiere que aparezcan en el juego.

Tutorial 10. Creación de un juego (II)
Paso 11 de 22

Comencemos por la primera mejora.

Debido a la aleatoriedad de la posición de las minas, es posible que antes incluso de mover la nave, ya haya alguna mina que choque con ella.

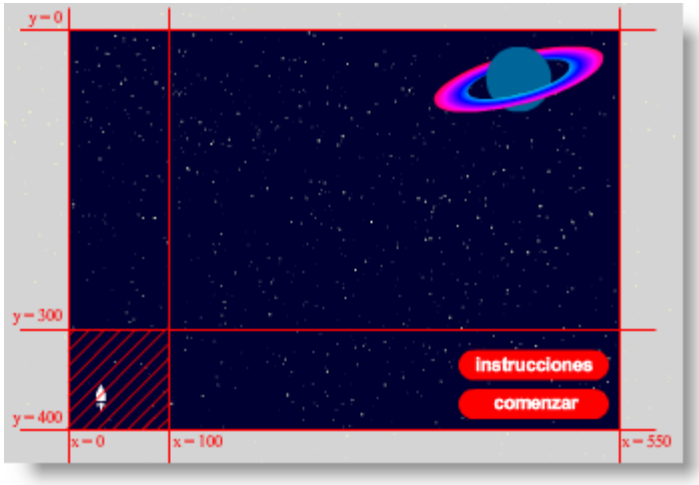
Para solucionar este problema, sustituiremos el código en el que creábamos la posición *x* e *y* de cada mina, dentro de la función *crearMinas,* por este código:

```
do
{
    mina.x = Math.random() * 550;
    mina.y = Math.random() * 400;
}
while (mina.x < 100 && mina.y > 300);
```

La sentencia *do…while* lo que hace es ejecutar en primer lugar lo que se encuentre entre llaves. Mientras se cumpla la condición que está entre los paréntesis del *while*, entonces lo que está entre las llaves del *do* volverá a ejecutarse.

En este caso, creamos una posición al azar para la mina. Si la posición se encuentra cerca de la esquina inferior izquierda (si la posición horizontal de la mina es menor que 100, y además su posición vertical es mayor que 300), entonces volverá a generarse una posición aleatoria para la mina. Este proceso se repetirá hasta que la condición no se cumpla, lo que significará que la posición generada para la mina ya no está cerca de la nave.

De un modo más gráfico, si la posición aleatoria de la mina está en la zona del cuadrado rayado, entonces cambiaremos la posición de la mina de forma aleatoria, hasta dar con una posición válida.



Tutorial 10. Creación de un juego (II)
Paso 12 de 22

Vamos a añadir el movimiento de vibración de las naves en el escenario. Creamos una nueva variable al inicio de la programación, junto con las variables *movimiento*, *velocidad* y *numMinas*, a la que llamaremos *vibracion*. Le asignamos un valor de 5.

```
var vibracion:Number = 5;
```

Vamos a programar que la posición de la nave varíe desde la posición en la que se encuentra en cada momento, a una nueva posición aleatoria que se encuentre a una distancia máxima de 5 píxeles de la posición actual, que es la cantidad que hemos asignado a la variable *vibracion*.

Es decir, si la posición actual de una nave fuera *x=200* e *y=300*, al instante siguiente el valor de *x* estaría entre *195* y *205*, mientras que el valor de *y* podría estar entre *295* y *305*. De esta forma, la nave vibrará un máximo de 5 píxeles en cada sentido (horizontal y vertical).

Como hemos aprendido anteriormente, la sentencia

```
Math.random() * vibracion
```

devolverá un valor que se encuentra entre 0 y 5 (que es el valor que hemos dado a la variable *vibracion*).

Por lo tanto, la sentencia

```
Math.random() * vibracion - Math.random() * vibracion
```

devolverá un número entre *-5* (0-5) y *5* (5-0).

Dentro de la función *enemigos* (que se ejecuta llamada por un *enter frame* de cada mina), pero fuera del condicional que evalúa si hay choque entre la mina y la nave, escribiremos el siguiente código:

```
e.target.x+=Math.random()*vibracion-Math.random()*vibracion;
e.target.y+=Math.random()*vibracion-Math.random()*vibracion;
```

Para evitar que con esta vibración aleatoria la nave pueda acabar fuera del escenario, añadiremos dentro de la misma función unas sentencias condicionales que evalúen si la mina está fuera de los límites, y de estarlo, la colocaremos de nuevo en el límite del escenario.

```
if (e.target.x < 0)
{
    e.target.x = 0;
}
```

Debemos añadir otros condicionales similares (si *x>550* entonces que *x=550*, si *y<0* entonces *y=0*, y si *y>400*.entonces *y=400*).

Tutorial 10. Creación de un juego (II)
Paso 13 de 22

Probamos la película para comprobar si se efectúa la vibración de las minas. Veremos que aunque las minas nunca desaparecen completamente del escenario, puede que veamos media forma de la mina fuera del escenario en algunos momentos, porque el punto de registro de la mina se encuentra en su centro.

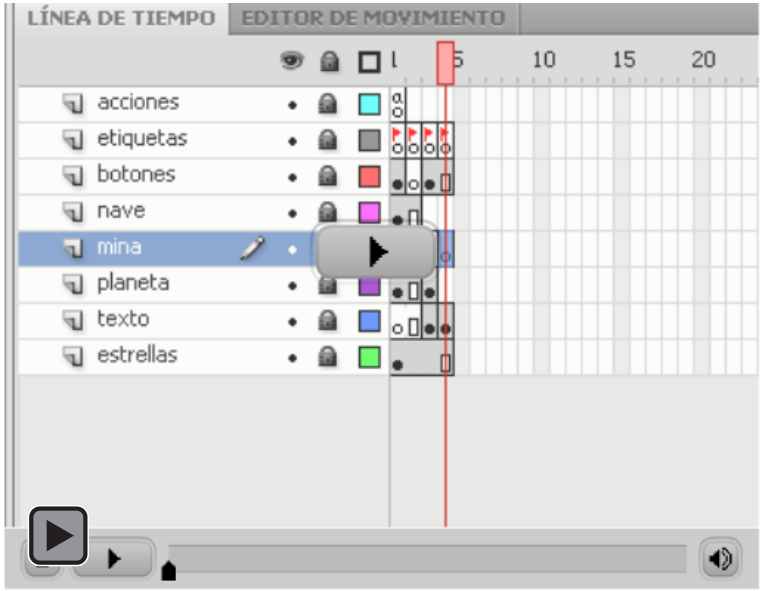
Si queremos, podemos modificar los límites de movimiento de la mina cambiando los valores de los condicionales del paso anterior. Por ejemplo, podemos reducir el espacio en el que pueden moverse las minas en 25 píxeles en cada lado.

La siguiente mejora va a ser eliminar el planeta en el cuarto fotograma (fotograma perder), y mostrar como sustitución una mina en el centro del escenario.

En primer lugar, añadimos una nueva capa a la que llamaremos *mina*. Situamos esta capa por encima de la capa *planeta*.

Eliminamos el cuarto fotograma de la capa planeta pulsando sobre él con el **botón derecho** del ratón, y seleccionando **Quitar fotogramas**.

Después pulsamos con el **botón derecho** del ratón el cuarto fotograma de la capa *mina*, y seleccionamos **Insertar fotograma clave**.



En este nuevo fotograma clave, arrastramos una instancia del clip *mina* desde la biblioteca al centro del escenario, y aumentamos su tamaño hasta por ejemplo W:100 y H:100.

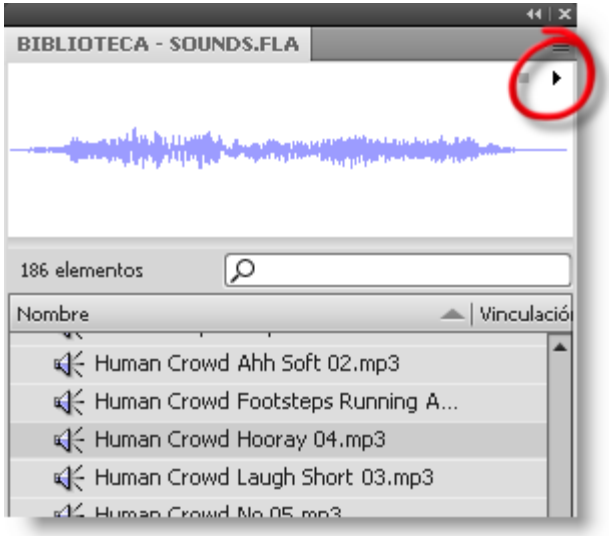
Tutorial 10. Creación de un juego (II)

Paso 14 de 22

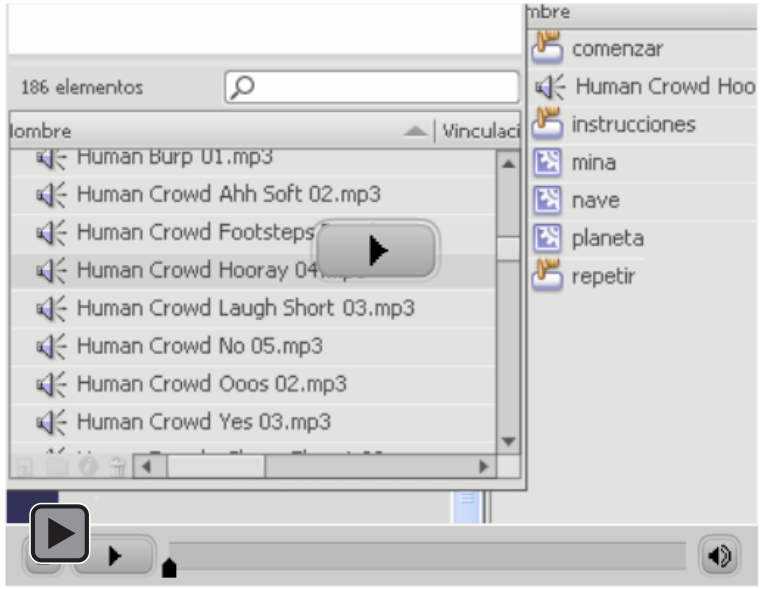
Vamos a buscar sonidos que asociaremos al momento de ganar o de perder la partida.

En **Ventana > Bibliotecas comunes > Sonidos** podemos encontrar algunos ejemplos de sonidos que podemos utilizar en nuestros proyectos.

Para escuchar los diferentes sonidos, seleccionamos el sonido en la biblioteca y pulsamos sobre el pequeño botón para reproducir el que se encuentra en la ventana donde se muestra la onda del sonido.



Utilizaremos los sonidos *Human Crowd Hooray 04.mp3* para el momento de ganar, y *Multimedia Internet CD-Rom Flash Blast 06.mp3* para el de perder. Si queréis podéis utilizar otros sonidos. Podemos ampliar el tamaño de la biblioteca de sonidos para leer los nombres de los sonidos con más comodidad. Para utilizar estos sonidos, los arrastraremos desde la biblioteca de sonidos hasta la biblioteca de nuestro juego. Una vez incorporados en nuestra biblioteca, podemos cerrar la biblioteca de sonidos.



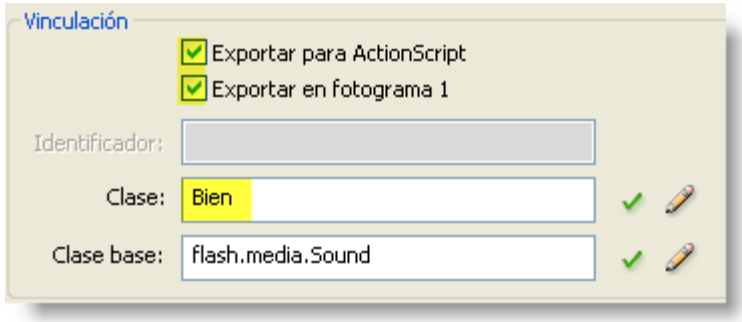
Tutorial 10. Creación de un juego (II)
Paso 15 de 22

Al igual que hicimos con el clip de la mina, vamos a exportar ambos sonidos para ActionScript.

Para ello pulsamos primero sobre el nombre del sonido *Human Crowd Hooray 04.mp3* en la biblioteca con el **botón derecho** del ratón y seleccionamos **Propiedades**.

En el área **Vinculación** de la ventana de **Propiedades de sonido** marcamos **Exportar para ActionScript** y **Exportar en fotograma 1**. Si no vemos esta área, pulsamos sobre el botón **Avanzado**.

Como nombre de clase escribimos simplemente *Bien*. Aceptamos la pantalla de advertencia final. Repetimos el proceso para el otro sonido, y como nombre de clase escribimos *Mal*.



En la biblioteca podremos ver los nombres de las clases que hemos dado a nuestros sonidos, así como el nombre que dimos al clip *mina*.

Para utilizar estos sonidos añadiremos, junto con la definición de otras variables al inicio de la programación, las siguientes líneas:

```
var sonidoGanar:Bien = new Bien();
var sonidoPerder:Mal = new Mal();
```

La variable *sonidoGanar* pertenecerá a la clase *Bien*, y será una nueva instancia de esa clase, cuya clase base es *Sound*, como pudimos ver en la ventana de vinculación. De la misma forma, *sonidoPerder* será un objeto de la clase *Mal*.

Tutorial 10. Creación de un juego (II)
Paso 16 de 22

Dentro de la función *juegoAcabado* reproduciremos un sonido u otro dependiendo de si hemos ganado o hemos perdido.

En esta función, la variable que nos indica si hemos ganado o perdido es *nombreFotograma*. Por lo tanto, lo que haremos es comprobar si *nombreFotograma* tiene el valor *"ganar"*. De ser así, reproduciremos *sonidoGanar*. Si no fuera así, reproduciremos *sonidoPerder*.

Por tanto, dentro de la función *juegoAcabado* incluiremos este código:

```
if (nombreFotograma == "ganar")
{
    sonidoGanar.play();
} else {
    sonidoPerder.play();
}
```

El doble signo de igual *(==)* compara si *nombreFotograma* es *"ganar"*. Este signo es para comparar, mientras que un solo signo de igual (=) es para asignar.

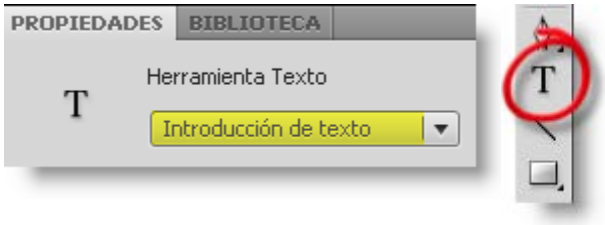
Si *nombreFotograma* es *"ganar"*, se reproducirá *sonidoGanar*. En caso de no serlo *(else)* se reproducirá *sonidoPerder*.

Ya tenemos el juego prácticamente finalizado. Podemos aumentar los valores a las variables *velocidad*, *numMinas* y/o *vibracion* para aumentar la dificultad del juego. Por ejemplo, una vibración de 15 complicaría considerablemente el juego, ya que el movimiento de las minas sería menos predictible.

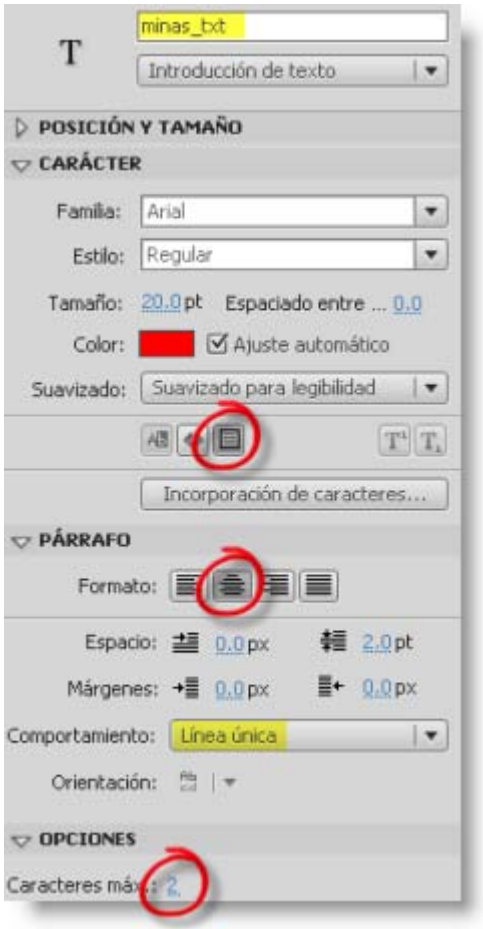
Podemos hacer que el propio jugador modifique estos valores mediante un campo de introducción de texto. Como ejemplo de ello, vamos a permitir que el jugador decida el número de minas con el que quiere jugar.

Para ello creamos en primer lugar una nueva capa llamada *num minas* y la situamos bajo la capa *botones*.

Tutorial 10. Creación de un juego (II)
Paso 17 de 22



Seleccionamos la **herramienta Texto**, y en el inspector de Propiedades seleccionamos **Introducción de texto**.



Clicamos sobre cualquier punto en el escenario para crear un campo de introducción de texto en la capa *num minas*.

Seleccionamos este campo de texto recién creado. En el **inspector de Propiedades,** le asignamos el nombre *minas_txt*.

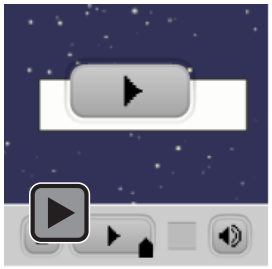
En el área **Carácter** seleccionamos el tipo de letra y el tamaño que queramos, el color **rojo** para que destaque sobre el fondo, y marcamos la casilla **Mostrar borde alrededor del texto**. Esto hará que aparezca un recuadro blanco con un marco negro. El marco negro apenas lo veremos, ya que nuestro juego tiene un fondo muy oscuro, pero el color blanco del recuadro nos mostrará claramente dónde se encuentra nuestro campo de texto.

En el área **Párrafo**, alineamos el texto en el centro y elegimos un **comportamiento** de **Línea única**.

Por último, en el área **Opciones** seleccionamos que se puedan escribir un máximo de 2 caracteres.

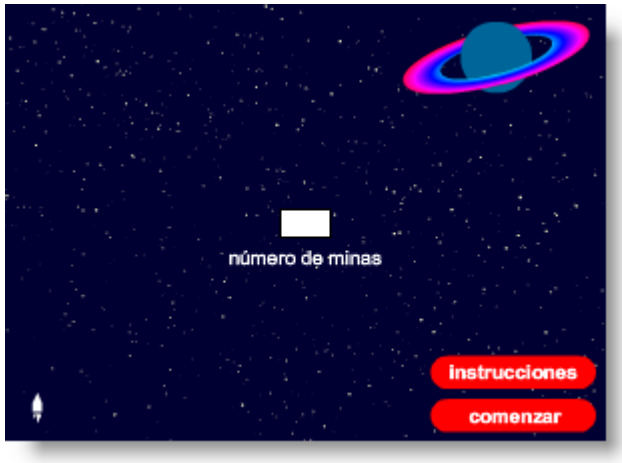
Tutorial 10. Creación de un juego (II)
Paso 18 de 22

Para que el bloque de texto no nos quede demasiado grande, podemos escribir en él dos números de prueba, y ajustar entonces el tamaño del campo.



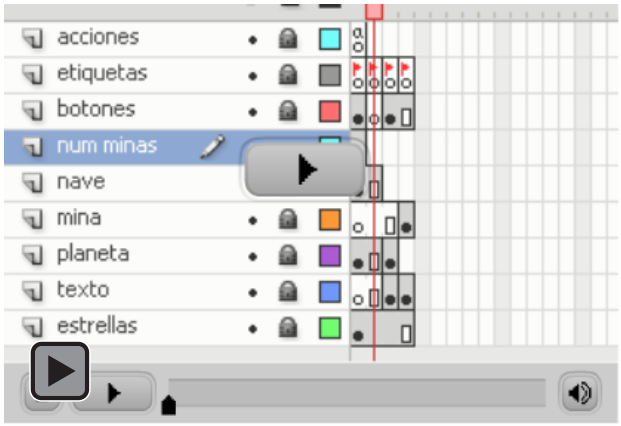
Una vez adaptado el tamaño, borramos los números que habíamos escrito como guía y centramos el campo de texto en el escenario.

En la misma capa, creamos un campo de **texto estático** bajo el campo de introducción de texto, y escribimos el texto *número de minas* en color blanco.



También podemos incluir en la misma capa una instancia de la mina, para que se muestre sobre el campo de texto. Para ello la arrastramos directamente desde la biblioteca.

Para que estos elementos sólo se muestren en el primer fotograma, seleccionamos el resto de fotogramas de esa capa, clicamos con el **botón derecho** del ratón, y seleccionamos **Quitar fotogramas** del menú contextual.



Tutorial 10. Creación de un juego (II)
Paso 19 de 22

Volviendo a la programación, podemos indicar en primer lugar que, antes de pulsar ningún botón, el foco del escenario se encuentre en el campo de introducción de texto, al que habíamos llamado *minas_txt* .

De esta forma podremos escribir directamente en el campo de texto, sin necesidad de clicar previamente sobre él.

Para ello añadiremos, junto a la definición inicial de las variables de nuestro juego, la siguiente instrucción:

```
stage.focus = minas_txt;
```

Después, cuando pulsemos sobre el botón *comenzar*, el foco pasará a la nave, ya que es lo que habíamos indicado en la función *jugar*. Por ello no hay ningún problema en establecer el foco inicial del juego en el campo de texto.

Si probamos ahora el juego, comprobaremos que si escribimos un número con el teclado, éste aparecerá en el campo de texto si necesidad de haberlo seleccionado previamente.

Cambiamos el valor inicial de la variable *numMinas* a 0 en vez de 10:

```
var numMinas:Number = 0;
```

Una vez que se pulse el botón *comenzar*, lo primero que se debe hacer es asignar a la variable *numMinas* el valor que hayamos escrito en el campo *minas_txt*.

Por lo tanto, la primera instrucción dentro de la función *jugar*, será:

```
numMinas = Number(minas_txt.text);
```

Para acceder al contenido de un campo de texto, tenemos que acceder a la propiedad *text* del campo. El contenido de un campo de texto es siempre de tipo *String*. Sin embargo, la variable *numMinas* es de tipo numérico. Por ello es necesario convertir el contenido textual del campo de texto *minas_txt* en número utilizando *Number*.

Probamos nuevamente nuestro juego. El número de minas se corresponderá al que indiquemos en el campo *minas_txt*. Sin embargo, el juego también comenzará si dejamos el campo en blanco o si introducimos letras. El siguiente paso será controlar estas posibles circunstancias.

Tutorial 10. Creación de un juego (II)
Paso 20 de 22

En la función jugar podemos indicar que, una vez que tengamos el *numMinas* a partir del texto introducido en el campo de texto, se evalúe el número de minas. Si se introduce un número de minas que se encuentre entre 1 (que haya al menos una mina) y 50 (número con el que muy difícil llegar al planeta), entonces que se ejecuten todas las demás instrucciones de la función *jugar*.

Por lo tanto, en la función *jugar* primero estará la línea

```
numMinas = Number(minas_txt.text)
```

Después habrá un condicional que evaluará si *numMinas* está dentro del margen que consideramos válido, y, si es así, se ejecutará el resto de la función y el juego comenzará.

```
if (numMinas > 0 && numMinas < 51)
{
    //ponemos aquí el resto de las sentencias que teníamos
    //en la función jugar, y comenzará el juego
} else
{
    //¿qué hacemos si numMinas está fuera de los límites?
    //haremos que aparezca un mensaje de error
}
```

Para el mensaje de error, podemos dibujar un rectángulo rojo similar al de los botones, y escribir sobre él un texto de aviso *(introduce un número de minas comprendido entre 1 y 50).*

Seleccionamos el rectángulo y el texto de aviso, y pulsamos **F8** para convertirlo en un **clip de película** llamado *aviso*. Le damos el nombre de instancia *aviso_mc.*

Al inicio de la película, este clip debe estar invisible por lo que, junto con la definición de variables al inicio de la programación, escribiremos

```
aviso_mc.visible = false;
```

Si *numMinas* no está dentro de los límites queremos que se muestre el aviso. Además podemos añadir instrucciones para borrar el número erróneo que hemos introducido, así como para devolver el foco al campo de texto, ya que se habrá perdido al pulsar el botón *comenzar*. Para todo ello escribiremos entre las llaves del *else*

```
aviso_mc.visible = true;
minas_txt.text = "";
stage.focus = minas_txt;
```

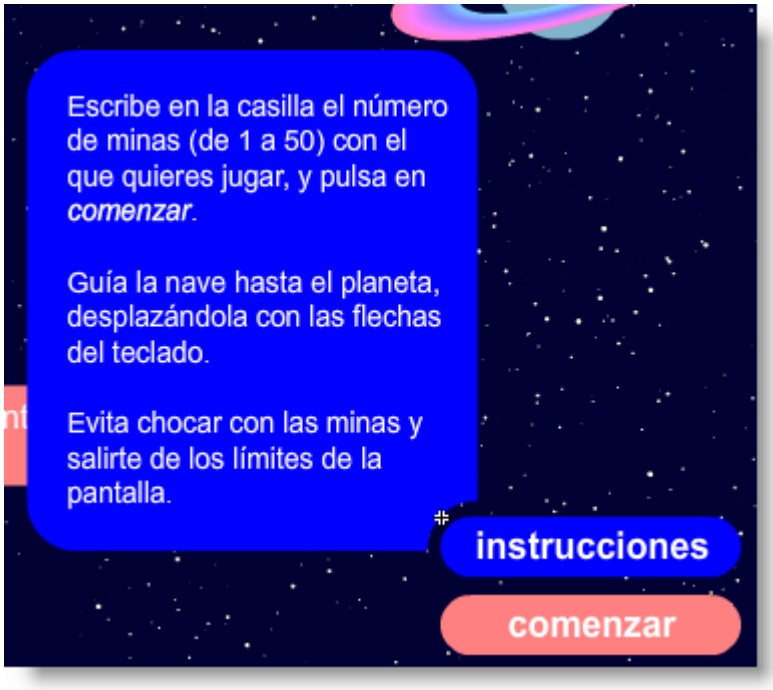
Tutorial 10. Creación de un juego (II)

Paso 21 de 22

En el caso de que *numMinas* esté dentro del margen válido, la película se desarrollará en el fotograma *juego* (fotograma 2), donde ya no se encuentra el aviso, por ello no es necesario indicar en este caso que *numMinas* no esté visible.

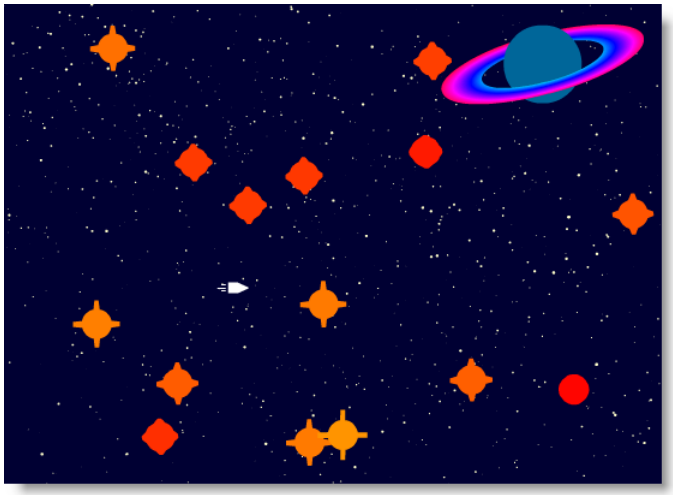
Por último nos queda modificar la información que muestra el botón instrucciones, para que aparezca también información sobre las minas. Posiblemente tendremos que aumentar el tamaño del rectángulo sobre el que aparecen las instrucciones.

Con esto ya habremos completado la programación de nuestro juego, con el que hemos aprendido a programar muchas tareas habituales en los juegos, tales como controlar objetos con el teclado, detectar colisiones, añadir elementos desde la biblioteca, generar azar, etc.



Tutorial 10. Creación de un juego (II)

Paso 22 de 22



Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer las siguientes actividades:

1. Permitid al jugador seleccionar la velocidad de la nave y la vibración de las minas.
2. Cread niveles para el juego, de tal forma que al llegar al planeta se comience un nuevo juego con más minas, mayor velocidad y mayor vibración
3. Añadid un nuevo control para el movimiento de la nave, haciendo que la barra espaciadora detenga su movimiento.

Tutorial 11. Aplicaciones dinámicas

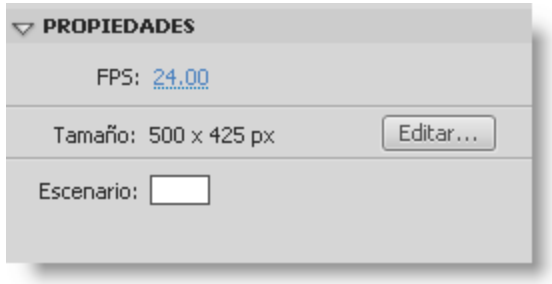
Paso 1 de 23

En este tutorial vamos a hacer una pequeña aplicación que mostrará imágenes, textos y vídeos externos.

Este tipo de aplicación es fácilmente actualizable, ya que el contenido de la película cambiará con tan sólo cambiar los archivos externos. Por tanto, es una buena opción utilizar cargas de datos externos en aplicaciones que deban actualizarse frecuentemente.

Aprenderemos también a incluir enlaces en nuestras películas.

Para comenzar, abrimos un nuevo archivo de Flash (AS 3.0). Cambiamos las dimensiones del escenario a 500 x 425 px. Guardamos el documento como *tutorial11.fla*.



Bar y habitación 3D
Imágenes realizadas con la herramienta 3D Studio.

Tutorial 11. Aplicaciones dinámicas
Paso 2 de 23

En esta película cargaremos imágenes con un tamaño de 400 x 300 px.

Para que las imágenes tengan un pequeño marco negro alrededor, dibujaremos un rectángulo negro ligeramente más grande que ese tamaño, y lo dejaremos como capa de fondo.

Seleccionamos la **herramienta Rectángulo**, y dibujamos un rectángulo negro de 410 x 310. Como ya vimos en el tutorial 9, recordemos que si sabemos con exactitud la medida de una forma antes de dibujarla, podemos pulsar la tecla **Alt** y hacer clic en el escenario para introducir manualmente la medida de la figura.

Situamos este rectángulo en la posición X:80 e Y:10.

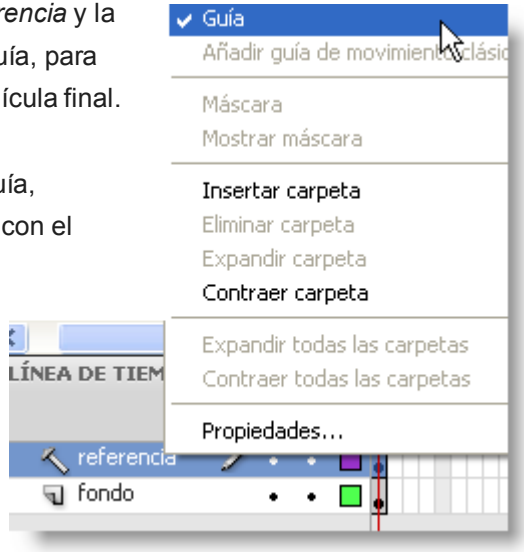
Llamamos *fondo* a esta capa, y la mantendremos siempre por debajo de todas las demás.

Creamos una nueva capa que nos sirva de guía para la posición donde irán las imágenes y los vídeos. Dibujamos un **rectángulo** rojo con el tamaño que tendrán las imágenes, esto es, 400 x 300.

Centramos este rectángulo dentro del fondo negro. Su posición final será X:85 e Y:15.

Llamamos a esta capa *referencia* y la convertimos en una capa guía, para que no se muestre en la película final.

Para convertir la capa en guía, pulsamos sobre su nombre con el **botón derecho** del ratón y seleccionamos **Guía**.



Tutorial 11. Aplicaciones dinámicas
Paso 3 de 23

Añadimos una nueva capa a la que llamaremos *botones*, en la que posicionaremos los tres botones para ir a las secciones *imagen*, *vídeo* y *enlaces* respectivamente.

Empezaremos creando el botón *imagen*. Escribimos la palabra *imagen* con la fuente *Calibri* (elige la fuente que prefieras), con un tamaño de 20 pt. Después añadimos un filtro de sombra con el 50% de intensidad, y a una distancia de 2 px.

Seleccionamos el campo de texto creado y pulsamos **F8** para convertir en un **símbolo de tipo Botón**, al que llamaremos imagen.

Dentro de la línea de tiempo del botón, **insertamos un fotograma clave** para el estado **Sobre**, y le cambiamos el color a *#006699*.

Volviendo a la escena principal, le damos a esta instancia el nombre *imagen_btn*.

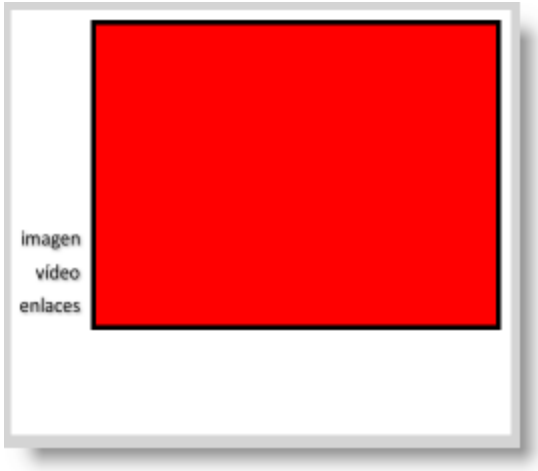
Tutorial 11. Aplicaciones dinámicas
Paso 4 de 23

Creamos los otros dos botones *(vídeo y enlaces)* de la misma manera.

Una forma rápida de crearlos puede ser duplicando el botón *imagen*, tal y como hicimos en el tutorial 8. De esta forma sólo tendríamos que sustituir el texto *imagen* por *vídeo* y *enlaces* respectivamente.

En el **inspector de Propiedades**, asignamos a estos dos nuevos botones los nombres de instancia *video_btn* y *enlaces_btn*.

Los situamos en la parte inferior izquierda del lugar reservado para las imágenes y los vídeos.

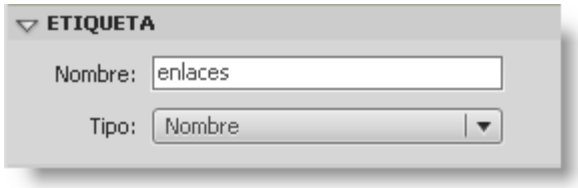


Cada botón irá a un fotograma diferente que mostrará la sección correspondiente. Vamos a crear estos fotogramas y añadirles una etiqueta con su nombre de sección, para facilitar de esa forma la programación de la navegación.

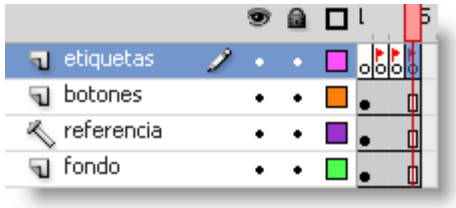
Insertamos una nueva capa a la que llamaremos *etiquetas*. Insertamos **fotogramas clave** en los fotogramas 2, 3 y 4. Podemos crear estos fotogramas clave haciendo clic sobre cada fotograma y pulsando la tecla **F6**.

Tutorial 11. Aplicaciones dinámicas
Paso 5 de 23

Pulsamos sobre los fotogramas clave 2, 3 y 4 que hemos creado, y les asignamos un nombre de etiqueta en el **inspector de Propiedades**. Al número 2 le llamaremos *imagen*, al 3 *vídeo*, y al 4 *enlaces*.



Insertamos fotogramas en las otras tres capas pulsando **F5**, para que todas las capas creadas hasta el momento se muestren en los cuatro fotogramas.



Añadimos una nueva capa llamada *texto* por encima de la capa *botones*. En esta capa añadiremos dos cajas de texto dinámico. En una de ellas mostraremos el título de la imagen o el vídeo, y en la otra incluiremos una pequeña descripción.

Seleccionamos la **herramienta Texto**, y en el desplegable superior del **inspector de Propiedades** seleccionamos **Texto dinámico**.

Clicamos con la **herramienta Texto** sobre el escenario, en la parte izquierda bajo la zona donde irán las imágenes y los vídeos. Después arrastramos su esquina para cambiar el tamaño del campo de texto.

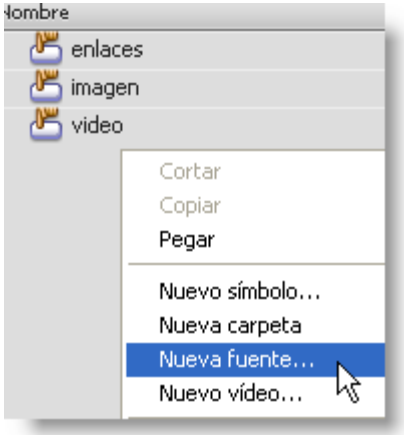
Una vez creado, podemos ajustar su posición arrastrándolo con la herramienta **Selección**..

Tutorial 11. Aplicaciones dinámicas
Paso 6 de 23

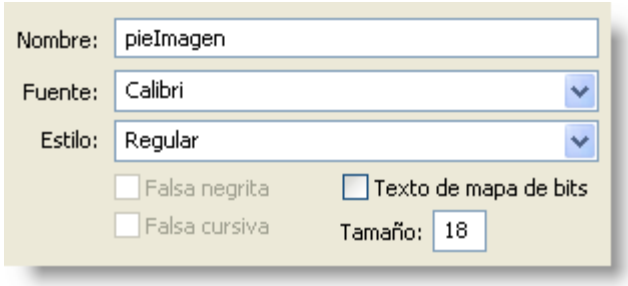
Para asegurarnos de una buena legibilidad, y de que el texto se visualiza con la fuente elegida aunque el usuario final no disponga de ella, incorporaremos la fuente en nuestra película.

Hay que tener en cuenta que esto aumenta el tamaño del archivo final. Para los textos estáticos no es necesario incorporar las fuentes.

En primer lugar clicamos con el **botón derecho** del ratón sobre un espacio vacío de la **biblioteca**, y seleccionamos **Nueva fuente** en el menú contextual.

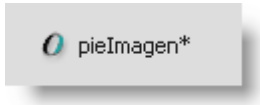


Como nombre de la fuente escribimos *pieImagen* como nombre, fuente *Calibri* y estilo regular (la misma que hemos utilizado en los botones).



Seleccionando de nuevo nuestro campo de texto en el escenario, seleccionamos como familia la fuente *pieImagen* que ahora aparecerá incluida en la parte superior del desplegable de las fuentes.

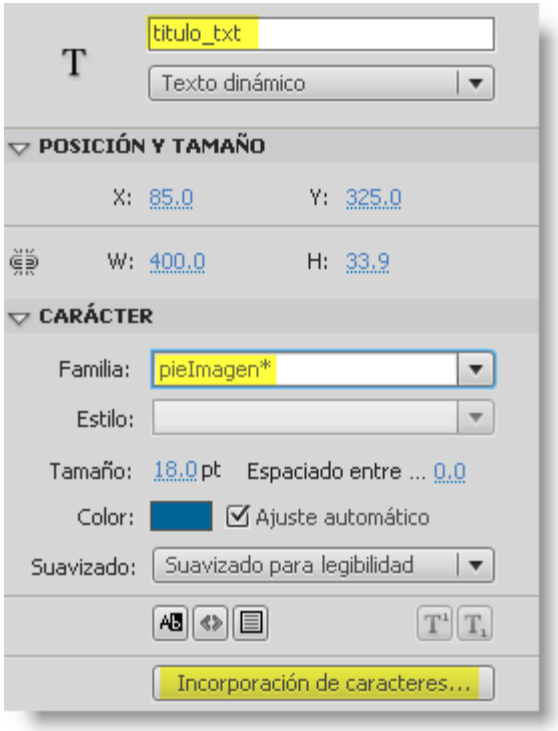
El nombre de la fuente tendrá un asterisco para indicar que es una fuente incluida en la película.



Tutorial 11. Aplicaciones dinámicas
Paso 7 de 23

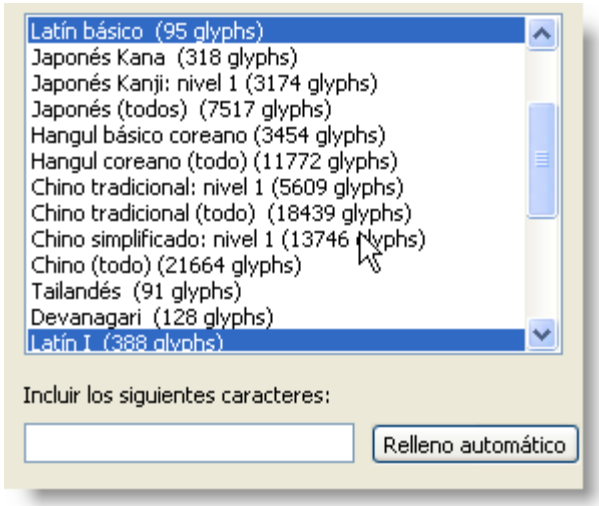
Elegimos también un tamaño de *18pt* y el color *#06699*. Le asignamos el nombre de instancia *titulo_txt*. Le añadimos también el mismo efecto de sombra que incluimos en los botones (paso 3).

Ahora nos queda incorporar los caracteres que necesitaremos de esta fuente. Para ello pulsamos en el botón **Incorporación de caracteres**.



Si supiéramos los caracteres exactos que necesitamos, podríamos escribir estos caracteres en la casilla *Incluir los siguientes caracteres,* y de esa forma disminuiría el tamaño del archivo final.

Como estos campos no tendrán un texto fijo, incluiremos los juegos de caracteres *Latín básico* y *Latín 1* (este último para acentos, ñ, ç, etc.).



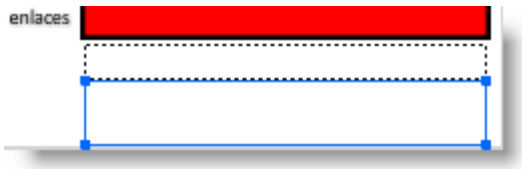
Aquí podemos ver la diferencia entre un campo de texto que no tiene caracteres incorporados, y el mismo texto con los caracteres incorporados.

Tutorial 11. Aplicaciones dinámicas
Paso 8 de 23

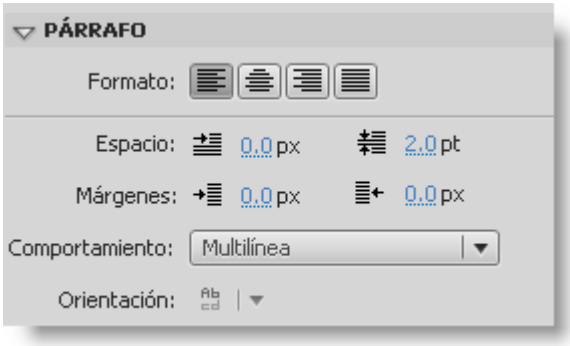
Siguiendo el mismo proceso, creamos otro campo de texto al que daremos el nombre de instancia *descripcion_txt*.

Seleccionamos la misma fuente incorporada *pieImagen* para mostrarlo. Cambiamos su color a negro, y el tamaño a 16pt. Esta vez no incluiremos ningún filtro como las sombras.

Situamos este campo alineado con el campo del título, y lo hacemos más ancho para que puedan caber varias líneas.



Para que puedan mostrarse varias líneas, debemos también seleccionar **Comportamiento Multilínea** en el inspector de propiedades.



Es interesante conocer el resto de opciones de párrafo, tales como alineación, sangría, espacio interlineal o márgenes, aunque en este caso no vamos a hacer cambios en estas propiedades.

Ahora que ya tenemos nuestros campos de texto preparados, vamos a crear con el bloc de notas un archivo de texto externo al que llamaremos *informacion.txt*.

Este archivo de texto contendrá los títulos y las descripciones de las tres secciones de nuestra película.

Tutorial 11. Aplicaciones dinámicas
Paso 9 de 23

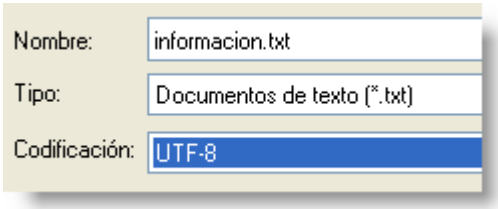
El contenido del archivo de texto tienen que ser pares de variable-valor, es decir, escribiremos el nombre de una variable, después el signo igual, y después el valor para esa variable.

Excepto en la primera variable, las demás deben ir precedidas por el signo *&*, que es la forma de indicar que comienza una nueva variable.

```
tituloImagen=Aquí escribimos el título de la imagen
&descripcionImagen=Esto es la descripción de la imagen.
&tituloVideo=Aquí pondremos el título del vídeo
&descripcionVideo=Esto es la descripción del vídeo.
&tituloEnlaces=Esto es el título de la sección enlaces
&descripcionEnlaces=Y esto es la descripción de la sección
enlaces. Probamos a escribir algo más largo para comprobar
si escribe varias líneas.
```

Como podemos ver, hemos llamado a las variables *tituloImagen*, *descripcionImagen*, *tituloVideo*, etc.

Estos serán los nombres de las variables a los que hagamos referencia en nuestra programación desde Flash.



Damos al archivo el nombre *informacion.txt* y seleccionamos codificación *UTF-8*. Lo guardamos en la misma carpeta en la que tengamos nuestro documento *tutorial11.fla.*

Ahora pasamos ya a la programación de nuestro documento de Flash.

Añadimos una nueva capa, por encima de las demás, a la que llamaremos *acciones*.

Con el primer fotograma de esta capa seleccionado, pulsamos F9 para acceder a la ventana Acciones.

Escribimos `gotoAndStop("imagen");` para que al probar la película vaya a la sección con la etiqueta *imagen* y se detenga. Sin un *stop()* o un *gotoAndStop()* la película se reproduciría en un bucle continuo.

Tutorial 11. Aplicaciones dinámicas
Paso 10 de 23

Vamos a explicar el proceso que hay que seguir para cargar un texto externo. En primer lugar necesitamos crear un cargador que esté preparado para recibir datos de tipo texto. Llamaremos a este cargador *cargadorTexto* (incluimos *var* la primera vez que lo definimos):

```
var cargadorTexto:URLLoader = new URLLoader();
```

Después le decimos que los datos que va a procesar tienen el formato de pares variable-valor, es decir, que son variables:

```
cargadorTexto.dataFormat = URLLoaderDataFormat.VARIABLES;
```

Le indicamos que cuando haya cargado los datos por completo, ejecute una función a la que hemos llamado *variablesLeidas*:

```
cargadorTexto.addEventListener(Event.COMPLETE,textoCargado);
```

Finalmente le decimos que cargue nuestro archivo *informacion.txt.* (Si el archivo lo tuviéramos en otra carpeta, aquí tendríamos que indicar la ruta relativa al archivo):

```
cargadorTexto.load(new URLRequest("informacion.txt"));
```

Creamos la función *textoCargado*, que se ejecutará cuando se genere el evento *complete*, es decir, cuando todo el archivo de texto se haya cargado.

Con esta función, le diremos que en el texto del campo de texto *titulo_txt* escriba el valor de la variable *tituloImagen* del archivo cargado. A su vez, damos la instrucción de que en el campo *descripcion_txt* se escriba el valor de la variable *descripcionImagen*.

Esos datos los tiene ahora almacenados el cargador del texto que habíamos creado, por lo que la función quedará como sigue:

```
function textoCargado(e:Event):void
{
  titulo_txt.text=cargadorTexto.data.tituloImagen;
  descripcion_txt.text=cargadorTexto.data.descripcionImagen;
}
```

Si probamos ahora la película, veremos que aparecen los textos en los campos correspondientes. A continuación programaremos que esos textos cambien en función del botón pulsado.

Tutorial 11. Aplicaciones dinámicas
Paso 11 de 23

Añadimos los *listeners* para cada botón de forma similar a como lo hemos hecho en tutoriales anteriores.

Las funciones a las que llamarán estos *listeners* las llamaremos *seccionImagen*, *seccionVideo* y *seccionEnlaces* respectivamente.

```
imagen_btn.addEventListener(MouseEvent.CLICK,seccionImagen);
video_btn.addEventListener(MouseEvent.CLICK,seccionVideo);
enlaces_btn.addEventListener(MouseEvent.CLICK,seccionEnlaces);
```

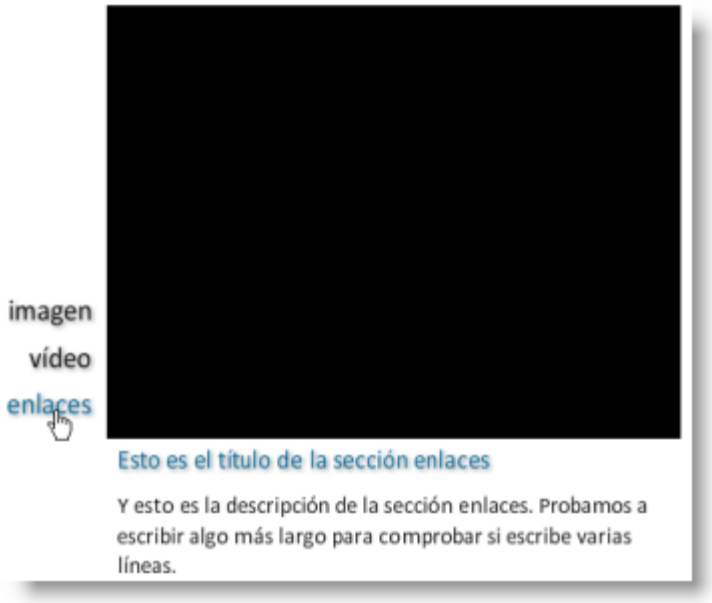
A continuación creamos las tres funciones. Por una parte, se encargarán de llevar la cabeza lectora al fotograma de la sección correspondiente (aunque por ahora todos los fotogramas son iguales). Por otro lado, cambiarán el valor de los campos de texto del título y de la descripción.

Este sería el ejemplo para la función *seccionVideo*:

```
function seccionVideo(e:MouseEvent):void
{
   gotoAndStop("video"); //nombre del fotograma
   titulo_txt.text = cargadorTexto.data.tituloVideo;
   descripcion_txt.text = cargadorTexto.data.descripcionVideo;
}
```

Completamos la programación de las otras dos funciones (*seccionImagen* y *seccionEnlaces*) y probamos de nuevo la película.

Los campos de texto mostrarán en primer lugar lo que habíamos indicado en la función *textoCargado* del paso anterior. Después cambiarán dependiendo del botón que pulsemos. Eso ocurre pese a cambiar de fotograma, ya que los campos de texto se mantienen visibles durante los fotogramas de todas las secciones.



Tutorial 11. Aplicaciones dinámicas
Paso 12 de 23

Vamos a cargar la imagen que aparecerá en la sección *imagen*, y que también será lo que se cargue por defecto al abrir el documento.

En primer lugar creamos un cargador para la imagen, que en este caso es un objeto *Loader*, en vez de un objeto *URLLoader*, que era el utilizado para cargar el texto. El objeto *Loader* se utiliza cuando necesitamos cargar de forma externa archivos jpg, png, gif y swf.

```
var cargadorImagen:Loader = new Loader();
```

Como veremos a continuación, en este caso, los *listeners* no se añaden al cargador, sino al contenido, por lo que tenemos que añadir *contentLoaderInfo* tras el nombre del cargador. Vamos a añadir dos *listeners*, uno que nos informe de cómo va el progreso de la carga de la imagen ejecutando una función llamada *cargandoImagen*, y otro que llame a la función *imagenCargada* cuando el contenido se haya cargado por completo.

```
cargadorImagen.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,cargandoImagen);
cargadorImagen.contentLoaderInfo.addEventListener(Event.COMPLETE,imagenCargada);
```

Por último cargamos el contenido, que en este caso es una imagen. Esta imagen se encuentra en la carpeta *tutorial11.* Tenéis que trasladarla, junto con el vídeo videodeldia.flv que también encontraréis en la misma carpeta, al mismo directorio donde tengáis el documento *tutorial11.fla*

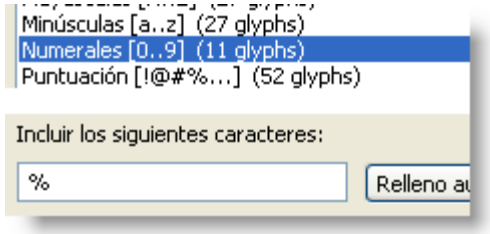
```
cargadorImagen.load(new URLRequest("imagendeldia.jpg"));
```

Si queréis podéis probar con cualquier otra imagen, siempre que tenga el tamaño de 400 x 300 px. En este caso habrá que poner el nombre de la imagen que queráis cargar en su lugar, o bien dar a vuestra imagen el nombre *imagendeldia.jpg*. Si la imagen la tuviérais dentro de una carpeta llamada img, la forma de hacer referencia a ella sería *img/imagendeldia.jpg*

Tutorial 11. Aplicaciones dinámicas
Paso 13 de 23

Queremos que la función *cargandoImagen* nos muestre el tanto por ciento cargado. Para mostrar este texto, en la capa *texto* creamos otro campo de texto.

Para este campo de texto sería suficiente con incluir los caracteres numerales y el carácter *%*.



Alineamos el párrafo en el centro, y situamos el campo de texto en un lugar central de la zona donde se cargará la imagen. Cambiamos el color a blanco, ya que este texto se verá sobre fondo negro. Por último le damos el nombre de instancia *porcentaje_txt*.

Calculamos el porcentaje de la carga, que será la cantidad de bytes cargados dividida por la cantidad total de bytes del objeto que se está cargando (el objeto que se ha llamado a esta función), multiplicada por 100. Para que no aparezca un número decimal, redondeamos a la baja con *Math.floor* la cifra obtenida. Después mostramos el resultado en el campo de texto que habíamos creado.

```
function cargandoImagen(e:ProgressEvent):void
{
   var porcentaje:Number = Math.floor(e.bytesLoaded/e.bytesTotal * 100);
   porcentaje_txt.text = porcentaje + "%";
}
```

Como vemos, primero hemos creado la variable *porcentaje*, que es un número. Después, en el campo de texto del porcentaje, mostramos el porcentaje seguido del símbolo *%*.

Cuando sumamos un número y una cadena de texto, el resultado es un texto, y por tanto se puede mostrar directamente en un campo de texto.

Si no hubiéramos añadido el símbolo *%*, tendríamos que haber convertido previamente el número del porcentaje en un texto con *String* para que no diera error:

```
porcentaje_txt.text = String(porcentaje);
```

Tutorial 11. Aplicaciones dinámicas
Paso 14 de 23

Ahora programaremos la función *imagenCargada*.

En primer lugar, queremos que cuando la imagen se haya cargado, ya no se muestre el texto del porcentaje de la carga. Una forma sencilla de hacerlo es asignar al texto un valor vacío, y esto se consigue abriendo y cerrando comillas sin nada en su interior.

Podríamos haber dicho que el campo de texto fuera invisible, pero esto tendría la desventaja de que tendríamos que recordar hacerlo visible de nuevo cuando otra imagen se fuera a cargar.

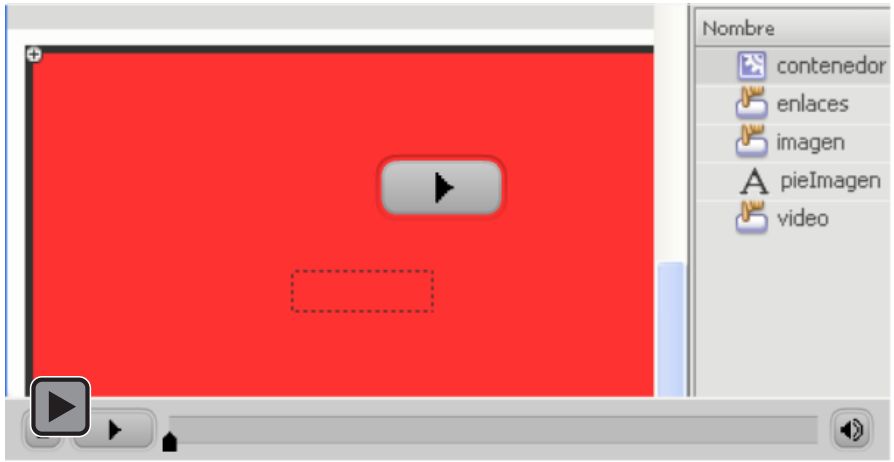
Por último, tenemos que añadir a la película la imagen que se ha cargado. Podríamos añadirla directamente a la escena principal. En este caso se posicionaría en la parte superior izquierda del escenario. Después podríamos darle las coordenadas de la posición correcta.

Una forma más práctica de posicionar y controlar un objeto que se añade a una película es crear un clip contenedor, y cargar el objeto dentro del contenedor. De esta forma, podemos posicionar el contenedor donde queramos, y automáticamente el objeto contenido dentro de él se posicionará en el mismo lugar.

Seleccionamos **Insertar > Nuevo Símbolo** y creamos un símbolo de tipo **Clip de película** con el nombre *contenedor*.

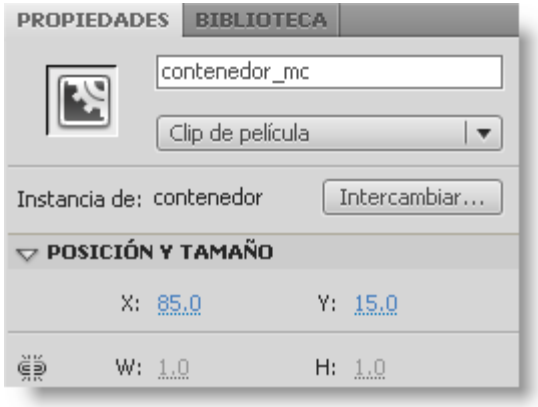
Creamos una capa a la que llamaremos también *contenedor*, y que situaremos por encima de la capa *texto*

Volvemos a la escena principal. Aunque el clip *contenedor* no tiene contenido, puede arrastrarse igualmente de la biblioteca al escenario para crear una instancia. Una pequeña cruz nos indicará su posición. Situamos el clip en la esquina superior izquierda del rectángulo de referencia.



Tutorial 11. Aplicaciones dinámicas
Paso 15 de 23

Damos al clip el nombre de instancia contenedor_mc en el inspector de Propiedades, y de paso nos aseguramos de que la posición del clip sea X:85 e Y:15.



Ahora ya podemos crear la función *imagenCargada*, y añadir la imagen que se ha cargado dentro del clip *contenedor_mc*.

```
function imagenCargada(e:Event):void
{
	porcentaje_txt.text = "";
	contenedor_mc.addChild(cargadorImagen);
}
```

La imagen cargada mostrará su esquina superior izquierda en la posición de registro del clip contenedor.

Probamos la película, y vemos que la imagen se carga directamente, sin ver el texto con el porcentaje. Esto ocurre porque al tener los archivos de forma local, la carga es instantánea.

Desde el menú del player en el que se muestra el swf cuando probamos la película, podemos simular una descarga para comprobar si nuestro cargador está funcionando correctamente.

En primer lugar, seleccionamos en el player **Ver > Configuración de descarga > DSL** (o 56K o el valor que queramos). Después seleccionamos **Ver > Visor de anchos de banda**. Por último, seleccionamos **Simular descarga**.

En la parte superior izquierda de la pantalla se irán mostrando varios detalles de la película, como su tamaño, FPS, en qué fotograma se encuentra, cómo va la carga, etc.

Veremos que la pantalla permanece un tiempo en blanco, que es el tiempo en el que se están cargando los elementos gráficos previos, y después podremos ver el porcentaje de la carga de nuestra imagen.

Tutorial 11. Aplicaciones dinámicas
Paso 16 de 23

La imagen de nuestra película pesa solo 20 KB, por lo que el porcentaje de su carga es muy rápido. Podemos hacer pruebas con fotografías de más peso para comprobar el funcionamiento del cargador.

De momento nuestro documento pesará unos 90 KB, que en realidad se cargaría más rápido de lo que muestra el simulador de descarga. De todas formas es una herramienta muy útil para comprobar el funcionamiento de cargadores, o hacernos una idea clara de qué partes de nuestra película tardan más en cargarse.

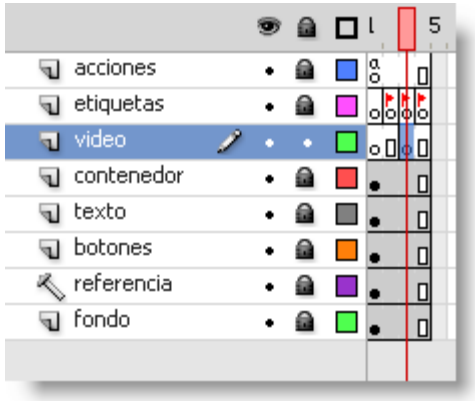
Continuemos con la programación de nuestra película. La imagen se ha cargado pero se muestra en los fotogramas de todas las secciones. Esta vez la ocultaremos asignándole *visible = false* cuando se pulse en la sección de vídeo o de enlaces, y *visible = true* cuando se pulse en la sección de imagen.

Podríamos también eliminar el fotograma que contiene el contenedor en las secciones de vídeo y enlaces, pero entonces tendríamos que volver a cargar la imagen cada vez que volviéramos al apartado imagen.

Por lo tanto añadimos el código `contenedor_mc.visible = false;` dentro de las funciones *seccionVideo* y *seccionEnlaces*, y `contenedor_mc.visible = true;` en la función *seccionImagen*.

Ahora que tenemos la sección imagen terminada, pasaremos a la sección vídeo.

Añadimos una nueva capa, y esta vez insertamos un fotograma clave en el fotograma 3, que es al que le corresponde la etiqueta *video*.

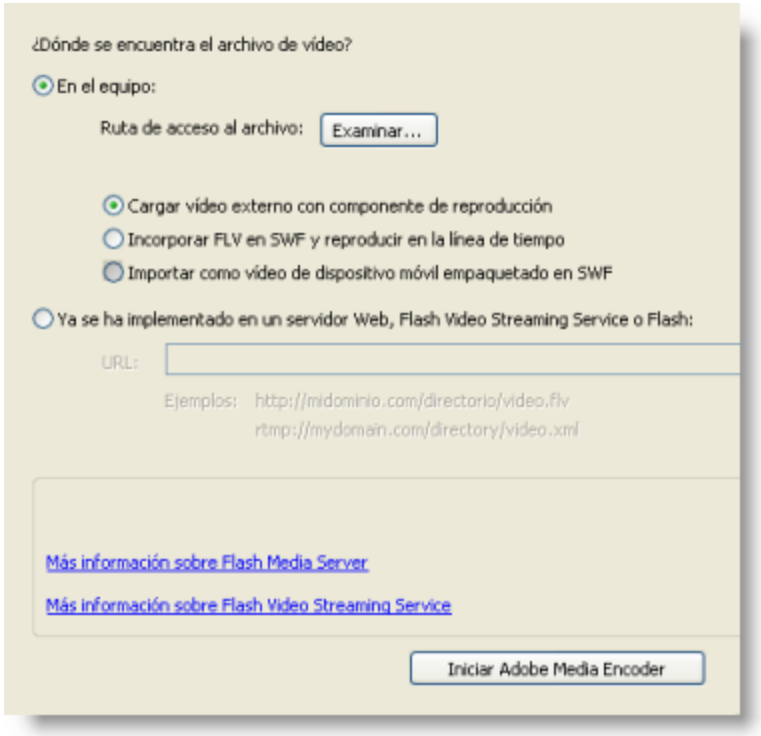


Estando en ese fotograma, seleccionamos **Archivo > Importar > Importar vídeo**. Se nos abrirá una nueva ventana con las opciones para importar vídeo.

Tutorial 11. Aplicaciones dinámicas
Paso 17 de 23

Si el vídeo estuviera implementado en un servidor, podríamos indicar directamente la URL donde se encuentra.

En nuestro caso, seleccionamos **En el equipo**, y pulsamos sobre **Examinar** la ruta de acceso.



Seleccionamos el archivo *videodeldia.flv*. Si tenemos algún otro archivo FLV también podemos utilizarlo.

La opción de incorporar FLV en SWF y reproducir en la línea de tiempo sólo sería recomendable para vídeos muy cortos (unos pocos segundos) y sin audio, ya que puede haber problemas de sincronización.

La opción de importar como vídeo de dispositivo móvil es para el caso de estar preparando contenido para su distribución en teléfonos móviles y otros dispositivos electrónicos.

Por lo tanto, seleccionaremos **Cargar vídeo externo con componente de reproducción** y clicamos en **Siguiente**.

Si el vídeo no fuera FLV, aparecería un aviso para que iniciáramos Adobe Media Encoder para convertir el vídeo en un archivo FLV compatible.

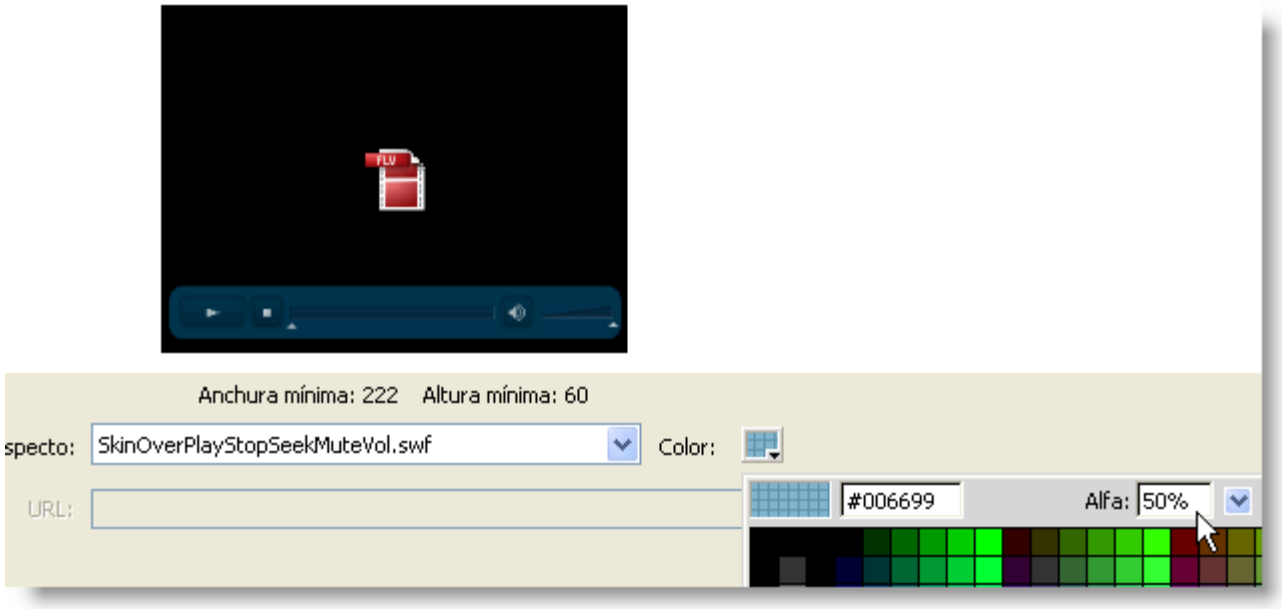
Tutorial 11. Aplicaciones dinámicas

Paso 18 de 23

En la siguiente pantalla podemos elegir un reproductor con más o menos controles, y que se muestre por debajo del vídeo o sobre él. También podremos personalizar el color.

Por ejemplo, podemos seleccionar el color *#006699* con una transparencia del *50%*, y que se muestre sobre el vídeo (*SkinOver…*).

Clicamos en **Siguiente** y después en **Finalizar** sin necesidad de marcar la casilla de ver la ayuda del vídeo.

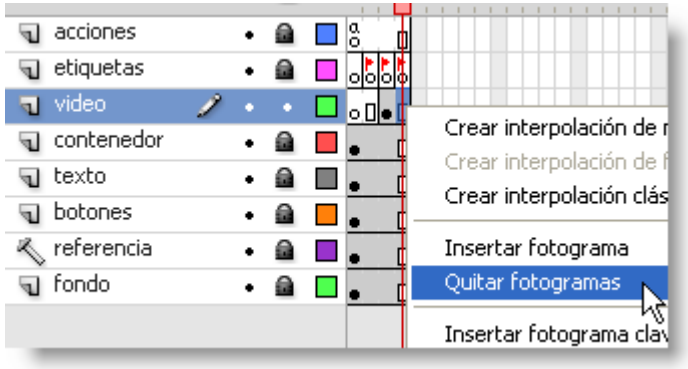


Tutorial 11. Aplicaciones dinámicas
Paso 19 de 23

En el escenario tendremos ahora una instancia del componente *FLVPlayback*. No es necesario dar un nombre a esta instancia.

La posicionamos en el mismo lugar que el contenedor de las imágenes, es decir, en X:85 e Y:15.

Podemos también ampliar su tamaño a W: 400 y H:300. No es recomendable cambiar el tamaño de los vídeos, porque pueden verse pixelados, pero en este caso cambiaremos el tamaño para que coincida con el área de la imagen.



Quitamos el último fotograma para eliminar el vídeo en la sección *enlaces*, es decir, en el fotograma 4.

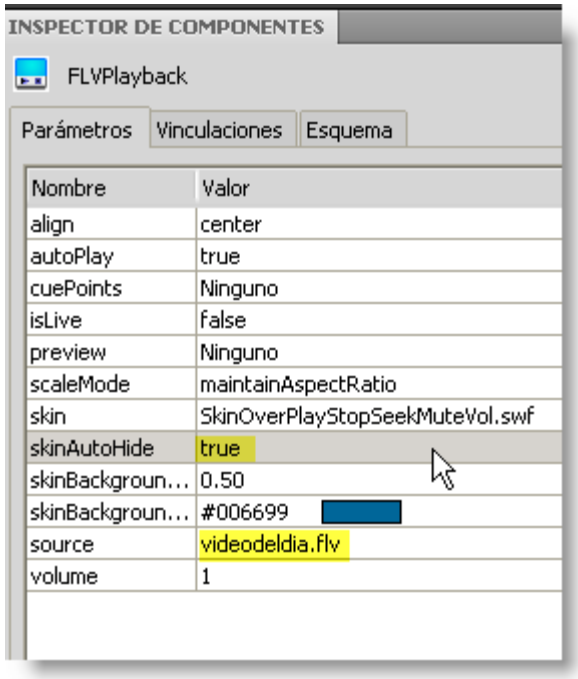
Para ello clicamos con el **botón derecho** del ratón sobre el cuarto fotograma de la capa vídeo, y seleccionamos **Quitar fotogramas**. También podríamos haber seleccionado *Insertar fotograma clave en blanco*.

Probamos la película para comprobar que nuestro vídeo se reproduce correctamente.

Tutorial 11. Aplicaciones dinámicas
Paso 20 de 23

En el caso de los vídeos, a diferencia de las imágenes o el texto, el contenido no se carga en nuestra película, sino que se lee directamente de forma externa.

Con el componente del vídeo seleccionado en el escenario, clicamos en el símbolo  en el **inspector de Propiedades** o seleccionamos **Ventana > Inspector de componentes**.



Desde esta ventana podemos configurar varios aspectos de nuestro vídeo.

Por ejemplo, ahora está seleccionado *autoPlay* como *true*, lo que significa que el vídeo comienza a reproducirse automáticamente. Lo mantendremos como *true*.

Desde aquí también podemos cambiar el reproductor (*skin*) utilizado, así como su visualización. Vamos a cambiar la propiedad **skinAutoHide** a **true**, para que los controles sólo se muestren si posicionamos el puntero sobre el vídeo.

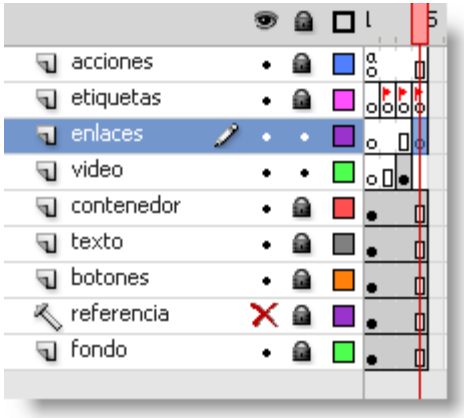
Desde aquí también podemos cambiar el vídeo que se va a reproducir. Así que en el caso de los vídeos, si quisiéramos actualizar el vídeo que se muestra, podríamos o bien cambiar desde esta pantalla el vídeo al que hace referencia, o bien, de forma externa, podríamos seleccionar otro vídeo y darle este mismo nombre.

Si probamos la película, podemos comprobar que pese a que el vídeo desaparece en las secciones *imagen* y *enlaces*, su audio sigue sonando. Para apagar el sonido incluiremos la instrucción `SoundMixer.stopAll();` en las funciones *seccionImagen* y *seccionEnlaces*.

Tutorial 11. Aplicaciones dinámicas
Paso 21 de 23

Para la última sección, creamos una nueva capa llamada *enlaces*, y creamos un fotograma clave en el fotograma 4, que es el correspondiente a la sección *enlaces.*

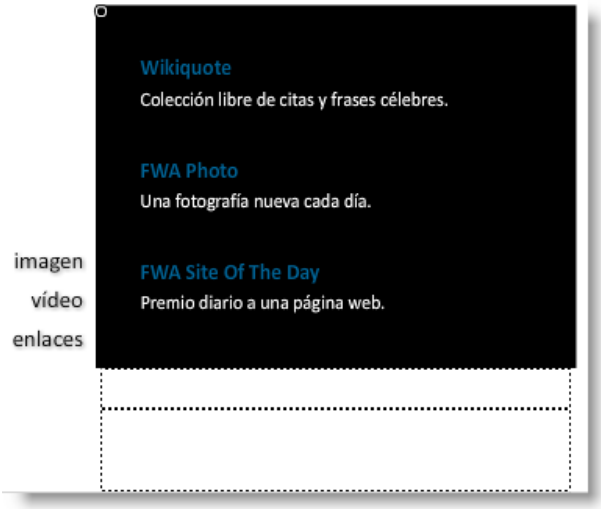
Ocultamos la capa *referencia* que nos había ayudado al posicionamiento del contenedor y del vídeo, para poder ver así el fondo negro real sobre el que se mostrarán los textos de los enlaces.



Esta vez crearemos los campos de texto como texto estático. Este apartado no va a ser por tanto dinámico.

Escribimos los nombres de las webs que queramos enlazar, y bajo ellas una pequeña descripción del sitio.

Seleccionamos tipos de texto diferentes para el título de la web, que se convertirá en el enlace, y la descripción. En este caso hemos utilizado el mismo tipo de letra que en el resto de la película. Hemos asignado a los títulos el estilo *Bold* (negrita) y un tamaño de letra mayor.



Tutorial 11. Aplicaciones dinámicas
Paso 22 de 23

La forma más rápida de crear un enlace es seleccionar una parte de un texto, o un campo de texto al completo y, en el **inspector de Propiedades**, en el apartado **Vínculo**, escribir la dirección web (URL)

Cuando creamos un vínculo de esta manera, el texto que se ha convertido en un vínculo aparecerá subrayado. Sin embargo, al probar la película, este subrayado no se mostrará.

Como destinos más habituales podemos elegir entre
-**_blank**, que abre el vínculo en una ventana nueva
-**_self**, que abre el vínculo en la ventana actual

Por defecto, si no seleccionamos un destino, el contenido se abrirá en una nueva ventana.

Si queremos que el vínculo sea una dirección de correo electrónico, escribiremos el vínculo con el formato *mailto:nombre@web.com.*

Si preferimos añadir el vínculo a un botón en vez de a un texto, tendríamos que añadir en la función correspondiente a ese botón el código siguiente:

```
navigateToURL(new URLRequest("http://www.direccion.com"));
```

Los vínculos funcionarán cuando probemos la película, pero sin embargo no funcionarán si abrimos directamente el swf por un problema de seguridad.

Para que un swf con vínculos sea totalmente funcional, debe estar subido a un servidor en internet.

En el caso de nuestra aplicación, para que todo funcionara tendríamos que subir junto con el swf generado, los archivos externos que hemos utilizado (texto, imágenes y vídeo), así como el swf del skin que hemos seleccionado para el vídeo, que podemos ver que se encuentra en la misma carpeta que nuestro *tutorial11.swf.*

Tutorial 11. Aplicaciones dinámicas

Paso 23 de 23

Bar y habitación 3D
Imágenes realizadas con la herramienta 3D Studio.

Para complementar los conceptos desarrollados en este tutorial, se recomienda hacer la siguiente actividad:

1. Modificad los títulos y las descripciones de cada sección en el archivo externo de texto.
2. Añadid al menú principal, bajo el botón *enlaces*, un botón llamado *contacto* que sea un vínculo a vuestro correo electrónico.
3. Sin modificar el documento fla, cambiad la imagen y el vídeo que se muestran en la película

Tutorial 12. Publicación y exportación

Paso 1 de 10

Una vez que hemos creado una película Flash, sea una animación, un juego, un recurso educativo, etc., el último paso es publicarla o exportarla a otro formato, según sea su finalidad y canal de distribución.

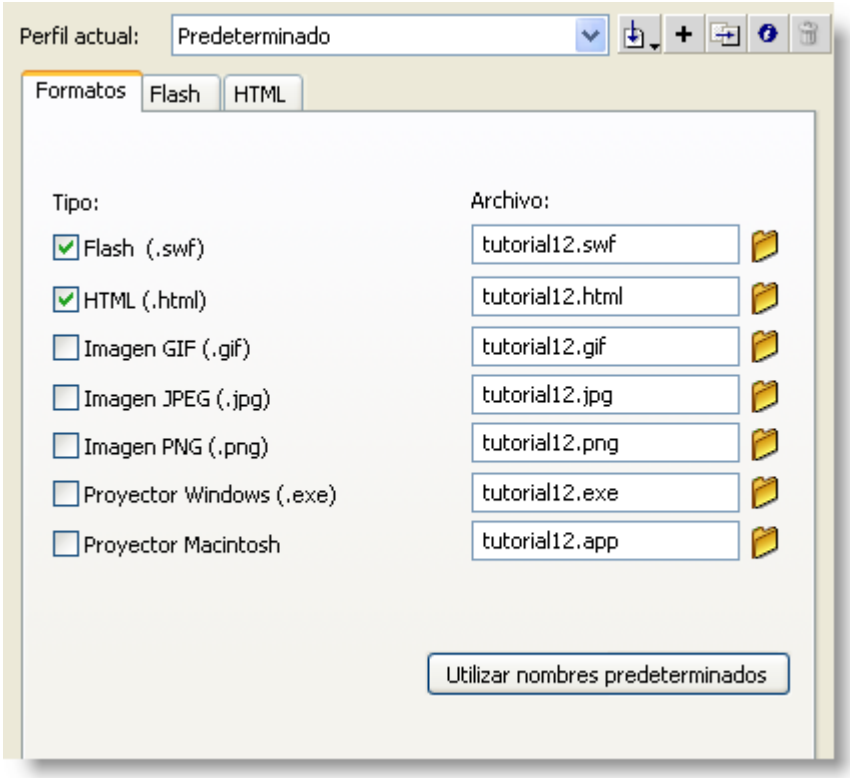
Una opción habitual es que nuestra película se visualice en una página web. La forma más rápida de generar un archivo *HTML* en el cual visualizar nuestro archivo *SWF* es mediante las opciones de publicación incorporadas en Flash.

Abrimos un documento fla con el contenido que queramos publicar. Seleccionamos **Archivo > Opciones de configuración**.

De forma predeterminada, aparecerán seleccionadas las casillas Flash y HTML, que son las necesarias para crear una página html que contenga el documento SWF.

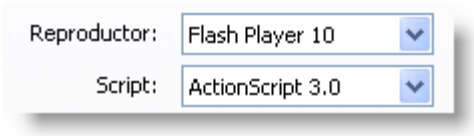
Los nombres de los archivos que se van a crear tendrán el nombre de nuestro archivo fla. Podemos cambiar estos nombres sin problemas.

Por cada tipo de formato que seleccionemos, excepto para los proyectores, se mostrará una pestaña en la que podremos configurar las características del archivo que queremos publicar.



Tutorial 12. Publicación y exportación
Paso 2 de 10

Comenzaremos por describir las opciones más importantes a tener en cuenta de la pestaña **Flash**.



En la opción **Reproductor** seleccionaremos la versión del Player necesaria para ver nuestra película. Salvo si vamos a crear un proyector, que lleva incorporado su propio player, los usuarios deberán tener una versión igual o mayor del plugin Flash Player que la que seleccionemos para ver correctamente el contenido de nuestra película.

Por ello es conveniente, si fuera posible, seleccionar una versión del reproductor lo más baja posible. Sin embargo, puede que muchas características que hayamos utilizado en nuestra película no sean compatibles con versiones anteriores del player.

Si seleccionamos una versión del reproductor que no sea compatible con algunas de las funciones que hemos utilizado en nuestra película, una ventana de alerta nos informará de esta incompatibilidad.

En la opción **Script** seleccionaremos la versión del lenguaje de script que hemos utilizado en nuestra película. En esa guía hemos utilizado *ActionScript 3.0*. En algunas películas, aunque no hayamos escrito código, puede que sea también necesario exportar con la versión 3.0.

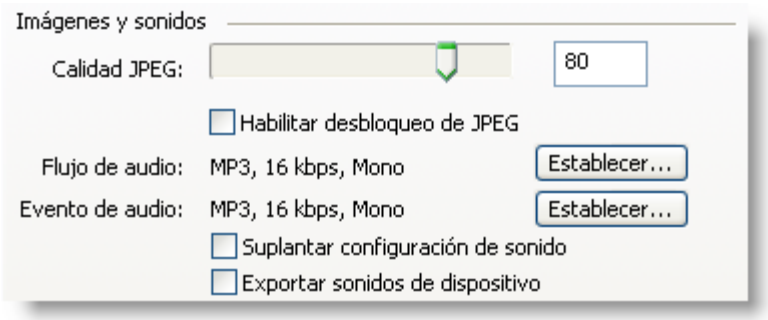
Si seleccionamos una versión del reproductor inferior a Flash Player 9, automáticamente la versión del script cambiará a 2.0 o 1.0, según la versión del reproductor seleccionada.

Si hemos seleccionado una versión del Script que sea incompatible con nuestro código o con algún elemento de la película, al publicar la película se mostrará una advertencia en el panel *Salida*, o aparecerán errores en el panel *Errores de compilador*.

Algunas de las funciones de Flash que necesitan la última versión tanto del reproductor como del script son las interpolaciones 3D o la herramienta huesos.

Tutorial 12. Publicación y exportación
Paso 3 de 10

En la parte superior del área de *Imágenes y sonidos* tenemos en primer lugar la **Calidad JPEG**. Aquí podemos seleccionar la calidad de los mapas de bits utilizados en nuestra película. Una calidad menor en la imagen significa una mayor compresión, y, por tanto, un peso menor en el archivo resultante.



Respecto al audio, en principio la configuración de la calidad de sonido será la que esté configurada desde las propiedades de cada sonido en la biblioteca en el caso de los eventos de audio, tal y como vimos en el tutorial 8.

Sin embargo, podemos seleccionar la casilla **Suplantar configuración de sonido** para establecer desde aquí la calidad de los sonidos en el archivo final.

En el caso de que hayamos utilizado en nuestra película sonidos como flujo de audio, es necesario establecer desde aquí la calidad de la compresión del sonido, incluso aunque no esté seleccionada la casilla de suplantar la configuración del sonido.

Cuando nuestra película esté publicada, una de las diferencias que podemos encontrar entre los sonidos de flujo y los de evento, además de las explicadas en el tutorial 8, es que un sonido de **flujo** comenzará a reproducirse junto con el resto de la película cuando se hayan descargado los primeros fotogramas, y podrá detenerse en algún punto si la reproducción avanza a más velocidad que la descarga.

Sin embargo, un sonido de tipo **evento** tendrá que haberse descargado completamente antes de empezar la reproducción, por lo que tardará más en iniciarse, aunque después no se detendrá hasta que no lo indiquemos de forma explícita.

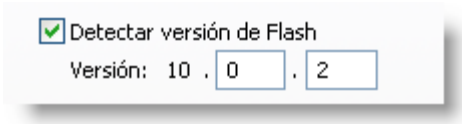
Tutorial 12. Publicación y exportación
Paso 4 de 10

Pasando a la pestaña **HTML**, una de las primeras opciones que nos podemos encontrar es si queremos que de forma automática se detecte la versión de Flash del navegador.

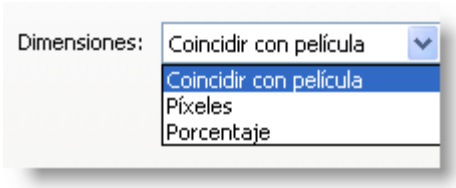
Si seleccionamos la casilla **Detectar versión de Flash** y el usuario tiene una versión inferior de Flash, se le enviará automáticamente a una página para actualizar su reproductor.

Si la casilla no está seleccionada, el archivo se reproducirá aunque el usuario no disponga de la versión adecuada, pero las funciones incompatibles con la versión del usuario se mostrarán con errores.

Por ello, siempre resulta conveniente asegurar que el usuario está visualizando la película con la versión adecuada del player.



En el apartado dimensiones, podemos seleccionar si la película debe mostrarse con el mismo tamaño que nuestro documento, o bien si debe ocupar un número concreto de píxeles o un porcentaje de la ventana del navegador.



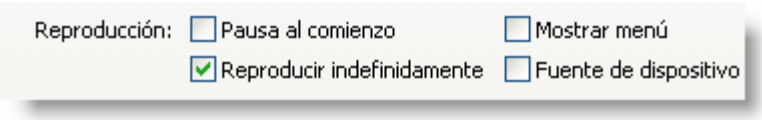
En el apartado dimensiones, podemos seleccionar si la película debe mostrarse con el mismo tamaño que nuestro documento, o bien si debe ocupar un número concreto de píxeles o un porcentaje de la ventana del navegador.

Si seleccionamos píxeles o porcentaje, es posible que el usuario visualice partes de la película que estaban fuera de los límites de nuestro documento.

En películas que incluyan mapas de bits o vídeos, no es conveniente redimensionar la película, ya que las imágenes y vídeos pueden mostrarse pixelados.

Tutorial 12. Publicación y exportación
Paso 5 de 10

En el área de Reproducción, podemos seleccionar, entre otras opciones, si la película deber **reproducirse indefinidamente** (como un bucle). De esta forma, sólo se detendrá cuando lo explicitemos con un stop. De no estar seleccionada la casilla, la película se reproducirá al completo, y se detendrá al llegar al último fotograma.



Si la casilla **Mostrar menú** está activada, se mostrará un menú contextual cuando pulsamos sobre la película con el botón derecho del ratón. El usuario podrá avanzar y retroceder en la película, cambiar la visualización, etc. Para desactivar este menú contextual, y que sólo aparezca información acerca de Flash, desactivamos la casilla Mostrar menú.

Consulta la ayuda de Flash para conocer las características del resto de opciones de publicación de HTML.

Seleccionamos **Archivo > Vista previa de publicación > HTML** para ver el resultado de nuestra película con la configuración actual. Podemos hacer varias pruebas con las opciones de HTML hasta conseguir que la película se muestre como queremos.

Para publicar la película seleccionamos **Archivo > Publicar**, o bien pulsamos sobre el botón **Publicar** de la ventana de Configuración de publicación.

Para poder visualizar la película en Internet, tendremos que subir tanto el archivo swf como el archivo html que hemos publicado.

Si observamos el código fuente del archivo html que Flash ha generado automáticamente, veremos que hay muchas líneas de código.

Si queremos incluir nuestra película como parte de una página web junto con otro contenido, el archivo HTML que crea Flash puede resultar confuso y difícil de compatibilizar.

Por ello recomendamos publicar desde Flash sólo nuestro archivo swf, y para integrar el contenido en un archivo HTML utilizar una librería javascript que detallaremos a continuación.

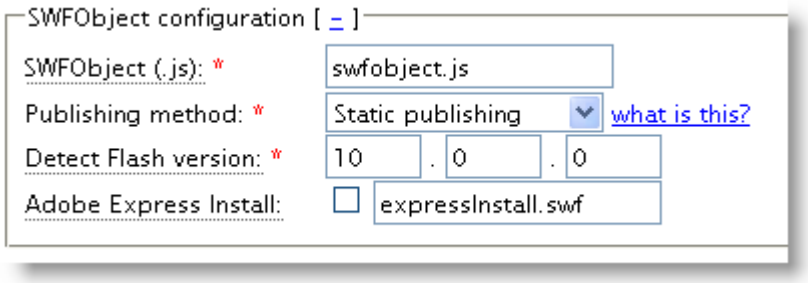
Tutorial 12. Publicación y exportación
Paso 6 de 10

Para la publicación de contenido flash incrustado en una página web, existe una librería Javascript llamada **swfobject.js** que permite incluir nuestra película en un archivo html respetando los estándares web.

Los archivos necesarios y la documentación sobre esta librería la podemos encontrar en http://code.google.com/p/swfobject/ .

En la misma página podemos encontrar un generador del código necesario para integrar nuestro SWF con esta librería, el cual facilita la tarea de crear el código.

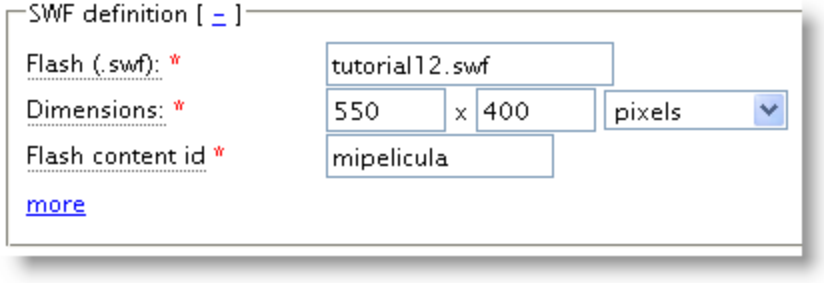
En la parte superior del generador del código tenemos que incluir en primer lugar la ubicación en la que tenemos nuestro archivo swfobject.js.



También tenemos que determinar la versión del Flash player necesaria para visualizar nuestra película.

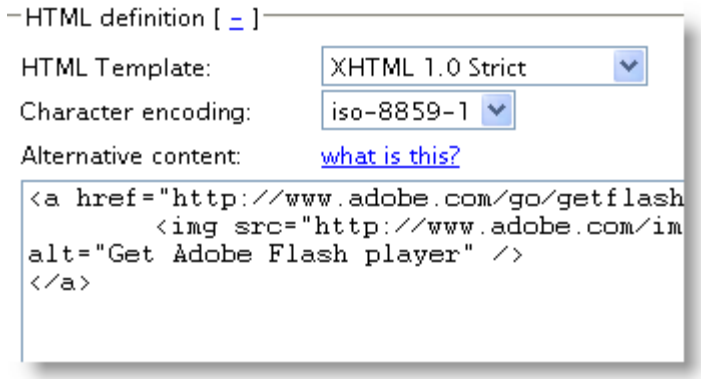
Podemos incluir el Adobe Express Install, un pequeño instalador que aparecerá en el caso de no disponer de la versión del plugin requerida, y que permitirá una instalación de una versión actual del player sin salir de nuestra página.

En el área de definición del SWF, tenemos que indicar el nombre y ubicación de nuestro archivo respecto al documento HTML. También debemos especificar sus dimensiones (en píxeles o en porcentaje). En el área *more* podremos definir otros parámetros, como por ejemplo si queremos que se muestre o no el menú contextual.



Tutorial 12. Publicación y exportación
Paso 7 de 10

En el área del HTML podemos definir el contenido alternativo que se mostrará a los usuarios que no dispongan de la versión del Flash player requerida.



Si por ejemplo se tratara de un banner animado que no es imprescindible para el funcionamiento de la página, podríamos mostrar simplemente una imagen fija en sustitución de la animación swf. En otros casos donde la película tenga mayor importancia, podríamos incluir una descripción del contenido, e indicar que para visualizar correctamente la aplicación se necesita tener una versión más actual de Flash player.

Por último pulsamos sobre **Generate** para generar el código html necesario para visualizar nuestra película.

Si queremos que nuestra película forme parte de otro documento html ya creado, será suficiente con copiar las líneas de script que se encuentran en la cabecera del código creado.

```
<script type="text/javascript" src="swfobject.js"></script>
<script type="text/javascript">
        swfobject.registerObject("mipelicula", "10.0.0");
</script>
```

También tenemos que pegar todo el contenido que se encuentra entre las etiquetas * * (incluidas) dentro del cuerpo de nuestra página.

De esta forma tendremos todo el contenido de nuestra película dentro de una etiqueta div. Para comprender las ventajas que esto supone, necesitamos tener conocimientos previos de HTML y CSS.

No debemos olvidar incluir el archivo *swfobject.js* junto con la publicación de nuestra página, así como *expressInstall.swf* si lo hubiéramos seleccionado.

Tutorial 12. Publicación y exportación
Paso 8 de 10

Volviendo a la ventana **Configuración de publicación**, también podemos publicar archivos GIF, JPEG y PNG para mostrar una imagen de nuestro documento.

Las últimas casillas son para crear **proyectores**, esto es, archivos ejecutables que contienen nuestra película, y que incluyen su propio reproductor independiente. De esta forma no es necesario que el usuario final disponga de la versión requerida del Flash player para visualizar el contenido.

Esta forma de publicación es la más habitual para crear CD interactivos con contenido realizado en Flash.

Para un ejecutable en Windows es suficiente con el archivo EXE que genera el proyector, por lo que no es necesario generar un archivo SWF.

Cuando se genera un ejecutable para un CD, es habitual incluir un archivo llamado *autorun.inf* que indique que se reproduzca de forma automática nuestro ejecutable. Para ello tendrá el siguiente contenido:

```
[autorun]
open=nombreArchivo.exe
```

Si necesitamos crear un proyector para Línux, podemos descargarnos un proyector en http://www.adobe.com/support/flashplayer/downloads.html , abrir con él nuestro swf, y crear un proyector nuevo desde Linux con nuestro contenido.

En el archivo que hemos creado con Flash y que está pensado para publicarse en un CD, es frecuente incluir algunas instrucciones para que el contenido se visualice a pantalla completa:

```
stage.displayState = StageDisplayState.FULL_SCREEN;
```

Dentro de la pantalla completa, podemos indicar si el contenido debe ampliarse o no escalarse, con una de las siguientes líneas:

```
stage.scaleMode = StageScaleMode.SHOW_ALL;
stage.scaleMode = StageScaleMode.NO_SCALE;
```

Tutorial 12. Publicación y exportación
Paso 9 de 10

Además de estas opciones de publicación que hemos visto, también podemos exportar el contenido de nuestra película a diversos formatos.

Seleccionamos **Archivo > Exportar > Imagen** para crear una imagen con el fotograma actual. Podemos elegir entre varios formatos además de los que ya aparecían en las opciones de publicación, como por ejemplo *Mapa de bits* o *Adobe Illustrator.*

Seleccionando **Archivo > Exportar > Película** podemos elegir también entre varios formatos de exportación.

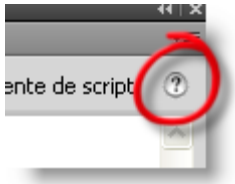
Por ejemplo, podemos exportar nuestra película como una secuencia de imágenes que luego podremos importar y modificar con programas de edición de vídeo. También podemos crear directamente un archivo de vídeo avi o mov (este último sólo funcionará correctamente si la publicación es para versiones del Flash player 5 o inferior).

Hay que tener en cuenta que, salvo el formato SWF, ninguna de las demás opciones de exportación de película ejecutará la programación de ActionScript. Por lo tanto, si tenemos alguna animación creada mediante código, ésta no se visualizará en el archivo exportado. Tampoco podrán visualizarse las líneas de tiempo anidadas

Ante cualquier duda sobre los temas tratados en este tutorial o en los anteriores, recomendamos consultar la ayuda de Flash.

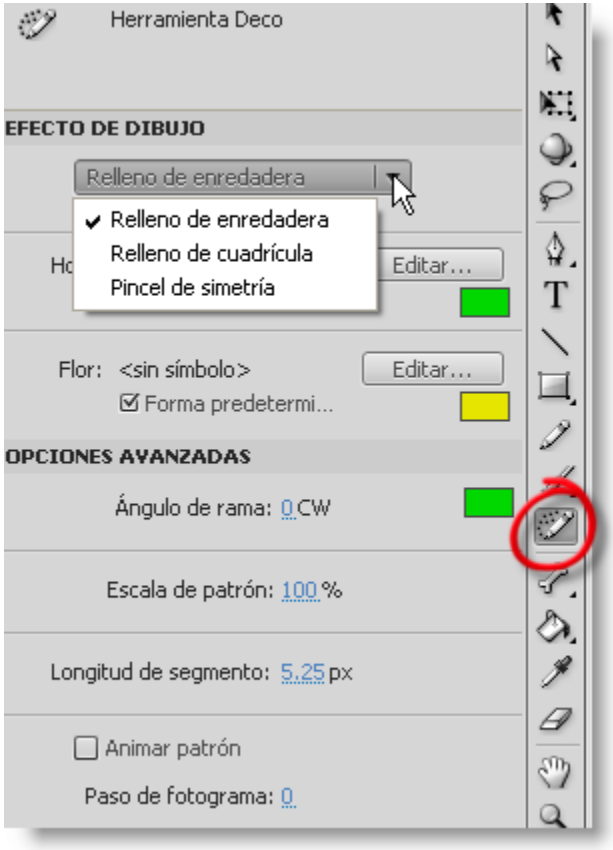
Recordemos que podemos acceder a la ayuda general de Flash desde el buscador de la barra de menú si se trata de dudas generales sobre el entorno de edición

o desde el icono de ayuda del panel Acciones si se trata de dudas sobre programación.



Tutorial 12. Publicación y exportación

Paso 10 de 10



Antes de finalizar, comentar que Flash dispone de algunas herramientas muy útiles que no hemos tratado a lo largo de estos tutoriales y que recomendamos probar, como por ejemplo:

•La **herramienta Deco**, que nos permite crear fácilmente rellenos complejos a partir de formas predeterminadas, o utilizando símbolos que tengamos en nuestra biblioteca.

•El **panel de kuler,** que podemos abrir seleccionando **Ventana > Extensiones > Kuler**, y con el cual podemos acceder a grupos de colores o temas creados por una comunidad en línea de diseñadores. También nos permite crear y guardar nuestros propios temas de colores para incluirlos fácilmente en nuestros proyectos.

Créditos

Grado y máster en multimedia (GMMD)
http://multimedia.uoc.edu/guias

Coordinación
Ferran Giménez Prado

Autoría
Raquel García Cabañas

Fecha de publicación
01.09.2009

Estudios de Informática, Multimedia y Telecomunicación