

## Guía de aprendizaje de Flash CS5

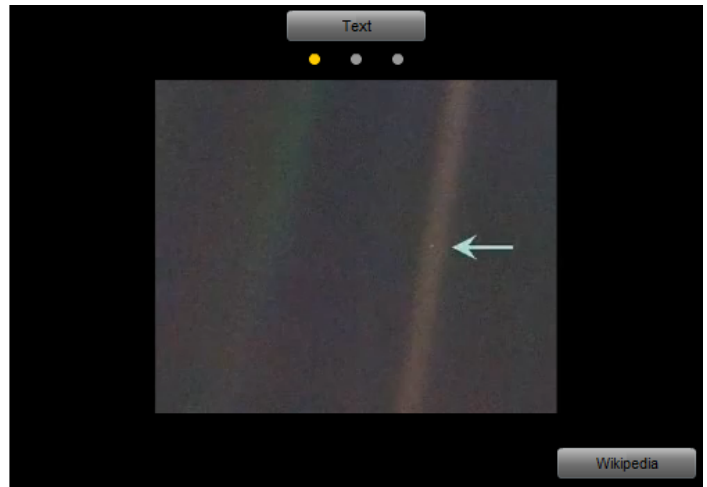
# Tutorial 12 - Vídeo y TLF

### Paso 1 de 18

En este tutorial vamos a experimentar con las nuevas posibilidades que ofrece el trabajo con vídeo en Flash y el nuevo motor de texto llamado Text Layout Framework.

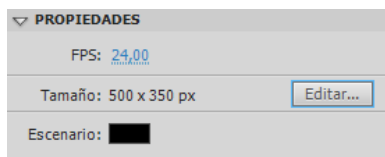
Continuando con el tutorial anterior, utilizaremos componentes para crear botones y aprenderemos a personalizarlos.

Por último, descubriremos cómo el nuevo panel Fragmentos de código puede facilitarnos la programación de nuestra película o aplicación.



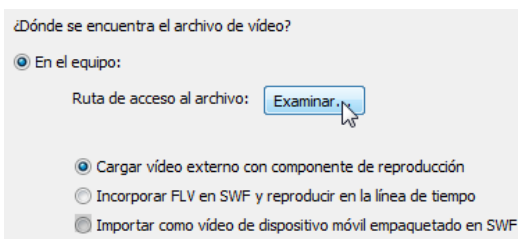
### Paso 2 de 18

Para comenzar crearemos un nuevo archivo Flash con fondo negro y dimensiones 500 x 350 px.



Comenzaremos con la importación del vídeo que vamos a utilizar en esta película. El vídeo, así como otros archivos que necesitaremos para la realización de este tutorial, lo encontraremos en el archivo [recursosTutorial12.zip](#).

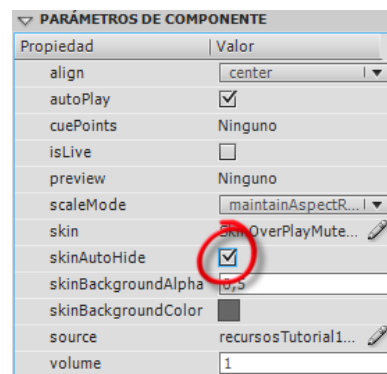
Seleccionamos **Archivo > Importar > Importar vídeo**, y en la pantalla de importación seleccionamos **En el equipo**. Buscamos la **Ruta de acceso al archivo** y seleccionamos **Cargar vídeo externo con componente de reproducción**.



En la siguiente pantalla seleccionaremos el aspecto del componente de vídeo que vamos a utilizar. Los nombres de los skins nos indican si se mostrarán sobre el vídeo o bajo él, y qué botones aparecerán.

En este caso utilizaremos **SkinOverPlayMute.swf**. Como color seleccionaremos un gris con una transparencia del 50%.

Una vez que tenemos el vídeo en el escenario, y con él seleccionado, vamos a marcar la casilla **skinAutoHide** para que el componente se oculte cuando no tengamos el cursor sobre el vídeo.



Como podemos ver, desde aquí podríamos modificar también el skin elegido, la ruta del vídeo, si se reproduce o no automáticamente, etc.

### Paso 3 de 18

Manteniendo el vídeo seleccionado en el escenario, veremos que en el inspector de Propiedades aparece también un área para marcar **Puntos de referencia** en nuestro vídeo.

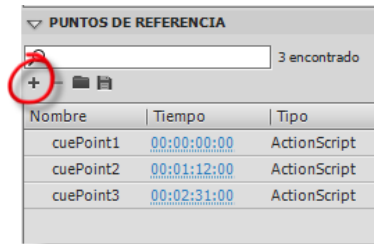
La utilización de puntos de referencia en un vídeo puede servir para navegar entre diferentes partes del vídeo, o bien para que ocurra algo cuando el vídeo llegue a determinado punto. En este tutorial vamos a utilizar ambas posibilidades.

Para añadir un punto de referencia en un vídeo, podemos reproducirlo desde el escenario y hacer clic sobre el símbolo + cuando alcance el momento de reproducción que queremos marcar. De esta forma se añadiría automáticamente un punto de referencia con el código de tiempo del momento en que hemos pulsado.

Otra manera de añadir un punto de referencia, que es la que vamos a utilizar en este caso, es hacer clic sobre el símbolo + directamente, y añadir manualmente los códigos de tiempo.

Añadimos 3 puntos de referencia, con los códigos de tiempo *00:00:00:00* (el inicio del vídeo), *00:01:12:00* y *00:02:31:00*.

Cambiamos sus nombres a *cuePoint1*, *cuePoint2* y *cuePoint3*, que son los nombres que utilizaremos en nuestro código.

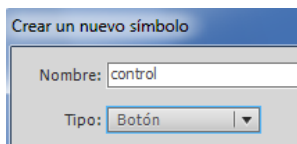


Damos a nuestro vídeo el nombre de instancia *sagan\_video*.

### Paso 4 de 18

Vamos a crear tres pequeños botones para navegar entre los diferentes puntos de referencia del vídeo.

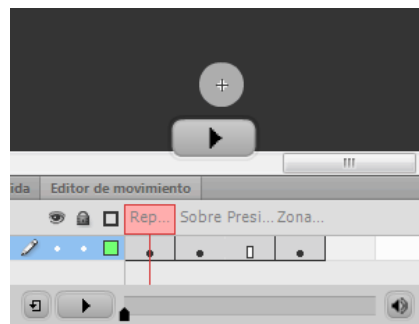
Seleccionamos **Insertar > Nuevo símbolo**. Le damos el nombre *control* y como tipo seleccionamos **Botón**.



En el fotograma *Reposo* dibujamos un círculo de 8 x 8 píxeles, sin trazo y con relleno gris, y lo centramos en su escenario.

Insertamos un fotograma clave (**F6**) en el fotograma *Sobre*, y cambiamos su color a blanco.

Como el botón va a ser muy pequeño, podemos aumentar el área de clic creando otro fotograma clave en el fotograma *Zona activa* y dando al círculo las dimensiones de 20 x 20 píxeles. No olvidemos centrar también en el escenario este último círculo.



## Paso 5 de 18

En una nueva capa a la que llamaremos *controls*, situaremos tres instancias del botón *control*. Situamos los botones en las coordenadas *x:220*, *x:250* y *x:280* y todas ellas en *y:40*.

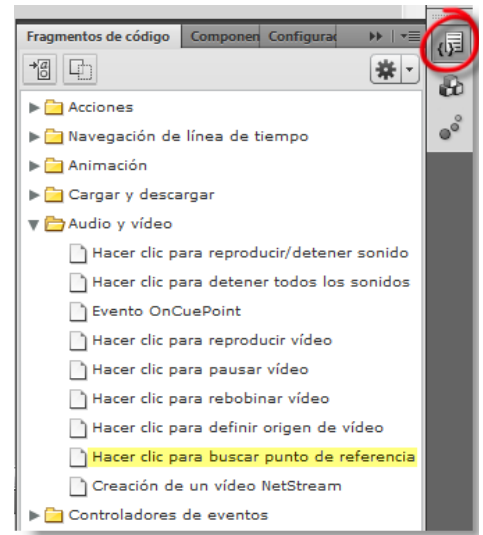
Damos a los botones los nombres de instancia *control1\_btn*, *control2\_btn* y *control3\_btn*.

Vamos a ayudarnos del nuevo panel **Fragmentos de código** para programar el funcionamiento de estos botones. Este nuevo panel puede facilitar la programación de algunas de las tareas más comunes de las películas Flash. También podemos crear y guardar fragmentos de código personalizados.

Seleccionamos el primer botón en el escenario. En el panel **Fragmentos de código** desplegamos la carpeta **Audio y vídeo** y seleccionamos **Hacer clic para buscar punto de referencia**, ya que queremos que, al hacer clic sobre el botón, el vídeo se desplace al punto de referencia correspondiente.

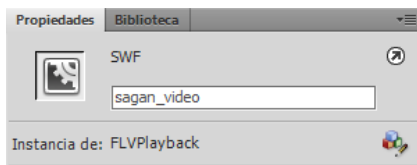
Se creará automáticamente una capa llamada *Actions*, en la que aparecerá un código y las instrucciones a seguir para su correcto funcionamiento.

En el siguiente paso modificaremos este código para que se adapte a nuestras necesidades.



## Paso 6 de 18

Las instrucciones del código que se ha generado nos indican que necesitamos un componente *FLVPlayback*. Si seleccionamos el vídeo, en el inspector de Propiedades podremos comprobar que efectivamente estamos utilizando ese componente.



En la primera línea de código tras los comentarios se ha añadido un listener al primer botón, que es el que teníamos seleccionado cuando añadimos el fragmento de código.

Como necesitaremos añadir los mismos listeners a los otros botones, copiamos esa línea y cambiamos los nombres de las instancias para que los tres botones también llamen a la misma función.

```
control1_btn.addEventListener(MouseEvent.CLICK,
fl_ClickToSeekToCuePoint);
control2_btn.addEventListener(MouseEvent.CLICK,
fl_ClickToSeekToCuePoint);
control3_btn.addEventListener(MouseEvent.CLICK,
fl_ClickToSeekToCuePoint);
```

Dentro de la función *fl\_ClickToSeekToCuePoint*, sustituimos *video\_instance\_name* por *sagan\_video* en las dos líneas, tal y como indican las instrucciones.

Dependiendo del botón pulsado, nos interesará ir a un punto de referencia o a otro, así que utilizaremos el nombre de la instancia que ha llamado a la función para saber el nombre del punto de referencia al que debemos ir.

Deberemos sustituir "Punto de referencia 1" por el nombre del punto de referencia correspondiente, que será "cuePoint" + `event.target.name.substr(7,1)`, ya que el carácter 7 del nombre de instancia de los botones corresponde con el número que hemos puesto en el nombre del punto de referencia.

Si tienes dudas sobre este paso puedes consultar el [paso 10 del tutorial 11](#).

La función *fl\_ClickToSeekToCuePoint* quedará por tanto como sigue:

```
function
fl_ClickToSeekToCuePoint(event:MouseEvent):void
{
    var cuePointInstance:Object =
sagan_video.findCuePoint("cuePoint" +
event.target.name.substr(7,1));
    sagan_video.seek(cuePointInstance.time);
}
```

Si probamos ahora nuestra película, podemos comprobar que los botones ya nos trasladan a los puntos de referencia que habíamos marcado en el vídeo.

## Paso 7 de 18

Para mejorar la usabilidad podemos utilizar un marcador que nos indique en qué punto nos encontramos. Por ejemplo, cuando pulsamos un botón podría verse un pequeño círculo amarillo en vez del botón gris, y de esta forma indicar visualmente que es ahí donde nos encontramos.

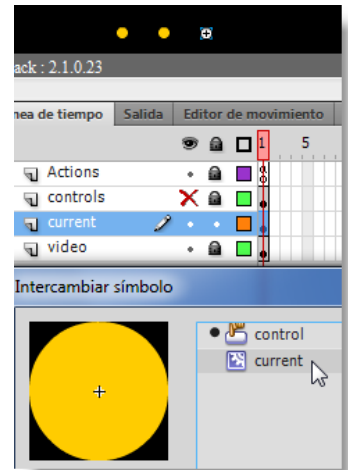
Seleccionamos **Insertar > Nuevo símbolo**, le asignamos el nombre *current* e indicamos que sea **Clip de película**. No es necesario en este caso que sea un botón, ya que no vamos a necesitar los diferentes estados de la línea de tiempo de los botones.

En este clip dibujaremos un círculo amarillo (#FFCC00, por ejemplo) de 8 x 8 píxeles y centrado en su escenario.

Creamos una nueva capa llamada *current* por debajo de la capa *controls*. Copiamos el fotograma de la capa *controls* y pegamos el fotograma en la capa *current*. Tendremos de esta forma los tres botones en el mismo lugar en ambas capas.

Bloqueamos y ocultamos la capa *controls*. Seleccionamos cada botón que hemos copiado en la capa *current* con el **botón derecho** del ratón y marcamos **Intercambiar símbolo** en el menú contextual.

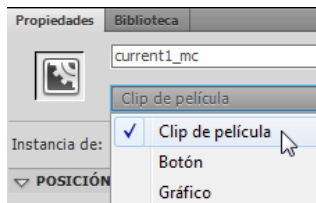
Seleccionamos *current* en vez de *control*. De esta forma tendremos los clips en el mismo lugar en el que se encontraban los botones, ya que hemos intercambiado unos símbolos por otros.



## Paso 8 de 18

Damos los nombres de instancia *current1\_mc*, *current2\_mc* y *current3\_mc* a estos clips.

Seleccionamos en el inspector de Propiedades un comportamiento de **Clip de película**, ya que al haber aparecido en el escenario como sustitutos de unos botones, adoptarán por defecto el mismo tipo que el de los símbolos a los que sustituyen.



Queremos que cuando se pulse un botón éste desaparezca, dejando ver de esta manera el marcador amarillo que se encuentra por debajo. El resto de botones deberán permanecer visibles para poder ser pulsados.

Una forma de solucionar este problema es hacer visibles todos los botones (no sabemos si habrá alguno invisible por haber sido previamente pulsado), y después hacer invisible el botón que acabamos de pulsar.

Para hacer todos los botones visibles crearemos una función a la que llamaremos *controlVisible*:

```
function controlVisible():void
{
    control1_btn.visible = true;
    control2_btn.visible = true;
    control3_btn.visible = true;
}
```

Añadimos dentro de la función *fl\_ClickToSeekToCuePoint* las siguientes líneas:

```
controlVisible();
event.target.visible=false;
```

De esta forma primero haremos todos los botones visibles, y después ocultaremos el que hemos pulsado.

Al inicio del vídeo el primer botón debe estar invisible, y para ello escribiremos fuera de la función la siguiente línea:

```
control1_btn.visible=false;
```

## Paso 9 de 18

Debido a que hay mucha distancia temporal entre los puntos de referencia que hemos marcado en el vídeo, sería conveniente que los pequeños círculos amarillos también pudieran ser pulsados por si el usuario quisiera volver al punto de referencia correspondiente.

Para ello añadimos a las tres instancias del clip *current* los mismos listeners que tienen las instancias de *control*, y de este modo ejecutarán la misma función al ser pulsadas.

```
current1_mc.addEventListener(MouseEvent.CLICK,
    fl_ClickToSeekToCuePoint);
current2_mc.addEventListener(MouseEvent.CLICK,
    fl_ClickToSeekToCuePoint);
current3_mc.addEventListener(MouseEvent.CLICK,
    fl_ClickToSeekToCuePoint);
```

Dentro de la función `fl_ClickToSeekToCuePoint`, ahora no nos interesa que el símbolo que haya llamado a la función sea invisible, ya que puede haber sido un clip *current* y no un *control*.

Para asegurar que es el botón y no el clip el que se hace invisible, y aprovechando que el número del nombre de instancia se encuentra en la misma posición en los clips *current* que en los botones *control*, sustituiremos la línea:

```
event.target.visible=false;
```

por la línea:

```
this["control" + event.target.name.substr(7,1) +
    "_btn"].visible = false;
```

Hasta ahora hemos programado que, al hacer clic sobre unas instancias, la cabeza lectora del vídeo se desplace hasta un punto de referencia. También nos interesa que, aunque no hagamos clic, cuando el vídeo se esté reproduciendo y alcance un punto de referencia, se muestre el *current* correspondiente.

---

## Paso 10 de 18

Como esta vez será el propio vídeo el que detecte el evento de haber llegado a determinado punto, lo seleccionamos en el escenario, abrimos el panel **Fragmentos de código** y seleccionamos **Audio y vídeo > Evento onCuePoint**.

Si no modificamos el nuevo código generado y probamos nuestra película, veremos que en el panel Salida aparecen los nombres de nuestros puntos de referencia cuando el cabezal pasa por ellos.

Por lo tanto, `event.info.name` devuelve el nombre de nuestros puntos de referencia.

Vamos a cambiar la función generada por otra en la que, en primer lugar, se hagan visibles todos los botones. Después, con un `switch`, analizaremos en qué punto nos encontramos, y en base a ello haremos invisible un botón u otro.

Crearemos un último punto de referencia al que llamaremos *cuePoint4* en la posición *00:03:31:00*, que es la que corresponde al final del vídeo. Cuando la cabeza lectora alcance esta posición, dejaremos marcado el inicio del vídeo. Así, si el usuario vuelve a reproducir el vídeo, encontrará marcado el primer punto.

Si tienes dudas sobre la utilización de la sentencia `switch`, puedes consultar el [paso 10 del tutorial 9](#).

```
function fl_CuePointHandler(event:MetadataEvent):void
{
    controlVisible();
    switch (event.info.name)
    {
        case "cuePoint1" :
            control1_btn.visible = false;
            break;
        case "cuePoint2" :
            control2_btn.visible = false;
            break;
        case "cuePoint3" :
            control3_btn.visible = false;
            break;
        case "cuePoint4" :
            control1_btn.visible = false;
            break;
        default :
            trace("Error: " + event.info.name);
    }
}
```

## Paso 11 de 18

En los siguientes pasos vamos a crear un clip que contenga el texto con la transcripción del vídeo, y lo repartiremos en tres fotogramas. En cada uno de los fotogramas mostraremos el texto correspondiente a cada punto de referencia. El usuario podrá elegir entre ver el vídeo o leer el texto mientras escucha el audio del vídeo.

La transcripción del vídeo se encuentra en el archivo *Pale-Blue-Dot.txt* que podemos encontrar en el mismo archivo de recursos donde se encontraba el vídeo.

Creamos una nueva capa llamada *text*. Bloqueamos y ocultamos la capa que contiene el vídeo para poder trabajar con más comodidad en esta capa.

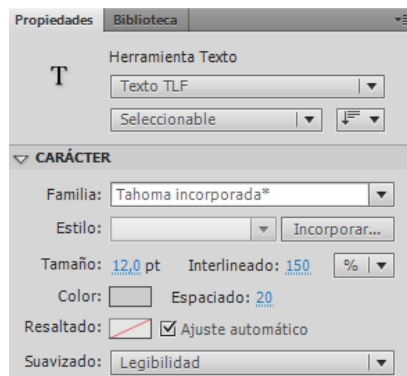
Seleccionamos **Archivo > Importar > Importar a escenario** y seleccionamos la imagen *Pale\_Blue\_Dot.png*. Posicionamos la imagen en X:310 e Y:51.

Para el texto vamos a utilizar el nuevo motor de texto llamado **Text Layout Framework (TLF)**. Vamos a utilizar algunas de las nuevas características de este motor de texto, pero para profundizar podemos consultar la [información de Adobe sobre el TLF](#).

Seleccionamos la herramienta **Texto**, **Texto TLF** y **Seleccionable**.

En el área **Carácter** elegimos la fuente *Tahoma Regular*, con un tamaño de 12 pt, color *gris claro*, y un interlineado del 150 %.

Seleccionamos como Suavizado: *Legibilidad* e incorporamos la fuente, a la que podemos llamar *Tahoma incorporada*, marcando como rangos de caracteres las mayúsculas, minúsculas y puntuación. De nuevo en el inspector de Propiedades, seleccionamos como *Familia* la fuente que hemos incorporado.



Si tenemos dudas sobre la incorporación de fuentes podemos consultar el [paso 17 del tutorial 10](#).

## Paso 12 de 18

Con la herramienta Texto hacemos clic sobre el escenario y arrastramos para crear un campo de texto en la parte izquierda de la imagen. Repetimos el mismo proceso para crear otro campo de texto en la parte inferior del escenario.

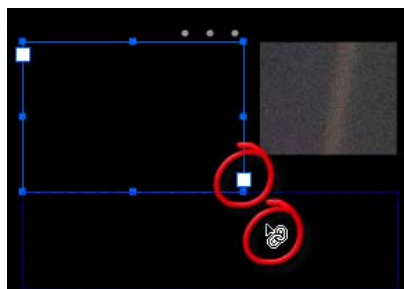
Seleccionamos el primer campo de texto. El inspector de Propiedades mostrará unas opciones u otras dependiendo de si hemos seleccionado la herramienta Texto, un bloque de texto en el escenario, o bien si estamos editando el campo de texto.

Damos al primer campo los valores X:25, Y:50, AN:265 y AL:180 en el inspector de Propiedades. Al otro campo de texto le damos los valores X:25, Y:230, AN:450 y AL:118.

Una de las novedades que supone el uso de TLF es que podemos unir dos campos de texto de tal forma que el texto fluya de un campo al otro.

Para unir los dos bloques hacemos clic sobre el primer cuadro, después hacemos clic sobre el puerto de salida del bloque (un recuadro que aparece en la parte inferior derecha), y después hacemos clic sobre el bloque de texto inferior.

De esta forma ambos campos de texto quedarán vinculados y el texto fluirá entre ellos.



Copiamos los dos primeros párrafos del texto del txt adjunto (los que se encuentran bajo el número 1). Pegamos el texto en cualquiera de los dos bloques, y veremos como fluye de un bloque a otro.

En este caso hemos utilizado la versión en inglés, pero podemos utilizar la versión en castellano o en catalán.

## Paso 13 de 18

Seleccionamos todo el contenido de la capa *text* (la imagen y los dos bloques de texto), y pulsamos **F8** para convertirlo en un símbolo de tipo **Clip de película**. Situamos el punto de registro en el extremo superior izquierdo. Le damos el nombre *text-img*.

Hacemos doble clic sobre este nuevo símbolo en el escenario para añadir los otros dos fotogramas que contendrá sin perder la perspectiva de su posición y tamaño respecto al escenario.

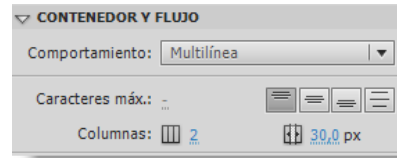
Vamos a añadir fotogramas en la línea de tiempo de este clip. Para ello seleccionamos el segundo fotograma con el **botón derecho** del ratón y seleccionamos **Insertar fotograma clave en blanco**. Repetimos el mismo proceso para añadir un tercer fotograma.

Nos situamos en el segundo fotograma y creamos un bloque de texto de las mismas características que los del primer fotograma (misma fuente, tamaño, etc.).

Posicionamos el bloque de texto en el punto *X:0* e *Y:0* de su línea de tiempo, y le damos el tamaño *AN:450* y *AL:270*.

Copiamos dentro del texto los cinco párrafos que se encuentran bajo el número 2 en el documento de texto adjunto.

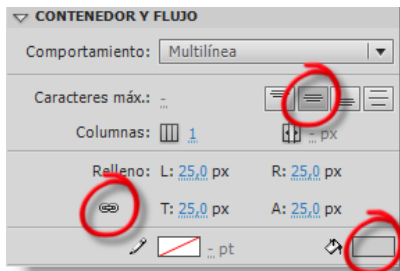
En el área de **Contenedor y flujo** del inspector de Propiedades, seleccionamos 2 columnas con una separación entre ellas de 30 px.



## Paso 14 de 18

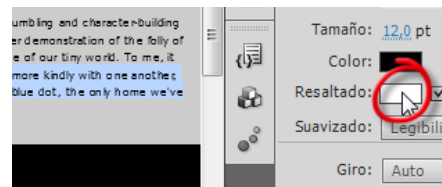
En el tercer fotograma vamos a crear otro campo de texto. En el inspector de Propiedades le damos los valores *X:0*, *Y:0*, *AN:450* y *AL:250*. Copiamos los dos últimos párrafos del archivo de texto.

En el área **Contenedor y flujo**, alineamos el texto verticalmente con el centro del contenedor, seleccionamos un padding para los cuatro lados de 25 px y seleccionamos un relleno *gris*.



En el área **Carácter** seleccionamos un color *negro* para que el texto se lea con más claridad.

Seleccionamos una parte del último párrafo, desde *our responsibility* hasta el final, y seleccionamos un **resaltado** blanco.



Con esto ya tendremos nuestros tres fotogramas con el texto del vídeo, y hemos explorado parte de las nuevas posibilidades del Text Layout Framework.

Volvemos a la escena principal y damos a nuestro clip el nombre de instancia *tlf\_mc*.

## Paso 15 de 18

Vamos a programar que el texto vaya sincronizado con el vídeo. Después crearemos el botón que utilizaremos para alternar las vistas entre vídeo y texto.

Cuando la película se inicie, deberemos en primer lugar detener el clip que contiene el texto en el primer fotograma, así como hacerlo invisible. Para ello escribiremos en la capa *Actions* el siguiente código:

```
tlf_mc.gotoAndStop(1);
tlf_mc.visible = false;
```

Para que el texto vaya al fotograma que coincida con el nombre del botón pulsado, dentro de la función `fl_ClickToSeekToCuePoint` añadiremos la línea:

```
tlf_mc.gotoAndStop(event.target.name.substr(7,1));
```

Dentro de la función `fl_CuePointHandler` deberemos hacer que el clip con el texto vaya al fotograma cuyo número coincida con el número del nombre del botón que hemos hecho invisible, por lo que la deberemos añadir líneas en el sentencia `switch`:

```
case "cuePoint1" :
    controll_btn.visible = false;
    tlf_mc.gotoAndStop(1);
    break;
```

Deberemos ir al fotograma 2 en el caso *cuePoint2*, al fotograma 3 en el caso *cuePoint3*, y de nuevo al 1 en el caso *cuePoint4*.

Ahora el vídeo y el texto funcionarán de la misma manera respecto a los puntos de referencia.

El paso siguiente será añadir un botón para alternar entre ambas vistas.

## Paso 16 de 18

Añadimos una capa llamada *switch*, donde colocaremos el botón que servirá de conmutador.

Seleccionamos **Componentes > User interface > Button**. Posicionamos el botón en *X:200* e *Y:5*. Le damos el nombre de instancia *switch\_btn*.

Añadimos este código, que explicaremos a continuación:

```
switch_btn.label = "Text";
var videoVisible:Boolean = true;
switch_btn.addEventListener(MouseEvent.CLICK, view);

function view(event:MouseEvent)
{
    if (videoVisible)
    {
        switch_btn.label = "Video";
        tlf_mc.visible = true;
        sagan_video.visible = false;
        videoVisible = false;
    }
    else
    {
        switch_btn.label = "Text";
        tlf_mc.visible = false;
        sagan_video.visible = true;
        videoVisible = true;
    }
}
```

La propiedad `label` de un componente de tipo *Button* asigna el texto que aparecerá en el botón. Comenzaremos asignando a `label` el valor `Text`, ya que al inicio es el vídeo el que está visible, y el botón debe servir para ver el texto.

Creamos una variable llamada `videoVisible` de tipo booleano (puede tener el valor `true` o `false`). Al inicio el vídeo está visible, por lo que le damos el valor inicial `true`.

Añadimos un listener al botón `switch_btn`, para que al ser pulsado ejecute la función `view`.

La función `view` evalúa en primer lugar si el vídeo está visible. De estarlo, cambia la etiqueta del botón a `Video`, muestra el clip del texto, esconde el vídeo, y guarda el dato de que el vídeo ya no está visible.

Si al pulsar el botón el vídeo no estaba visible se ejecutará lo que se encuentra entre las llaves del `else`, es decir, cambiará la etiqueta del botón a `Text`, el clip con el texto se ocultará, se mostrará el vídeo, y se almacenará el dato de que el vídeo está visible.

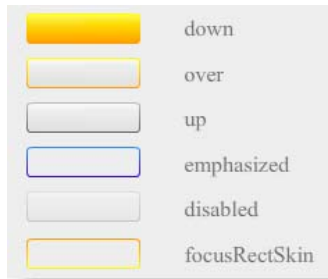
Podemos probar la película para comprobar si nuestro botón conmutador funciona correctamente.



## Paso 17 de 18

Podemos personalizar el aspecto del botón haciendo doble clic sobre él en el escenario. De esta forma entraremos en la línea del tiempo del componente, y podemos cambiar los gráficos.

Por ejemplo, para que el estilo sea similar al de los clips *current*, podríamos seleccionar colores amarillos o anaranjados para los bordes o el relleno del botón dependiendo del estado.



Añadimos una nueva capa llamada *web*, donde pondremos un botón con un enlace a una web. Vamos a crear el botón partiendo también de un componente *Button*, como en el caso anterior. Esta vez lo posicionamos en X:395 e Y:320.

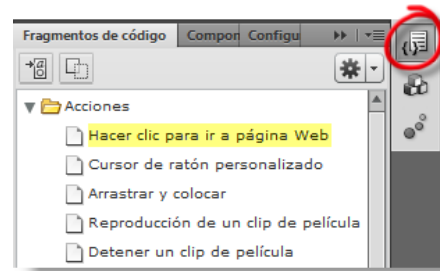
Al haber ya un botón en la biblioteca y no reemplazarlo, tendrá las mismas características de personalización que hayamos puesto en el otro botón.

Como no vamos a cambiar la propiedad *label* de forma dinámica, podemos asignar el valor del *label* en el área **Parámetros del componente** del inspector de Propiedades. Asignamos *Wikipedia* como valor del *label*.

PARÁMETROS DE COMPONENTE	
Propiedad	Valor
emphasized	<input type="checkbox"/>
enabled	<input checked="" type="checkbox"/>
label	<input type="text" value="Wikipedia"/>
labelPlacement	<input type="text" value="right"/>
selected	<input type="checkbox"/>
toggle	<input type="checkbox"/>
visible	<input checked="" type="checkbox"/>

Damos a este botón el nombre de instancia *web\_btn*.

Con el botón seleccionado, seleccionamos **Fragmentos de código > Acciones > Hacer clic para ir a página web**.



En el código generado, sustituimos *http://www.adobe.com* por *http://en.wikipedia.org/wiki/Pale\_Blue\_Dot* manteniendo las comillas, como indica la ayuda.

## Paso 18 de 18

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

1. Uniendo los conocimientos de este tutorial y el anterior, haz una minigalería de vídeos (sin puntos de referencia), utilizando un componente de reproducción de vídeo con más opciones, como por ejemplo la barra de desplazamiento.
2. Crea una nueva película que contenga un vídeo y cuya imagen de fondo cambie según el lugar en que se encuentre el vídeo.

