Universidad Técnica Federico Santa Maria.
Departamento de Electrónica.

macromedia®
FLASH

Trabajo de Programación de Sistemas

Profesor: Agustín González.

Nombre Autor: Darwin Valderas.

Fecha de Entrega: 30 de Octubre de 2002.

Resumen

En el siguiente trabajo se abordaran algunas ideas generales de Flash y de su lenguaje de programación ActionScript. Flash es un programa que permite crear animaciones muy novedosas y versátiles ricas en movimiento y color, la forma en que esto se lleva acabo es por medio de una combinación entre imágenes creadas y/o importadas y el trato que se le puede dar estas, en flash cada imagen o texto tiene una propia identidad, lo cual implica que pueden ser manipuladas como objetos de hecho se les llama objetos, lo interesante de Flash es la manera en que puede reunir y sacarle provecho a todos sus atributos de texto, color, películas, sonido, etc. El trabajar con capas y escenas le permite manejar muchos elementos diversos sin entrar en conflicto o en un terrible desorden, una vez creados los objetos ActionScript es el encargado de darle dinamismo, movimiento y acción a las películas Flash.

ActionScript es un lenguaje que desciende de JavaScript, por lo tanto ha heredado muchos de sus conceptos y sintaxis. ActionScript es un lenguaje interpretado por lo que es independiente a la plataforma de trabajo.

Introducción

El objetivo de es trabajo es dar a conocer aspectos generales de Flash pero en particular de ActionScript, se mostrara a grandes rasgos su sintaxis, sus propiedades orientadas a objeto y como crear Scripts, seria quizás conveniente si se desconoce el programa Flash familiarizarse con este antes, ya que hay detalles que se han dejado de lado como son los de crear imágenes, texto, etc, funciones del área de trabajo de Flash para poder dar más detalles sobre ActionScript.

¿Qué es Flash?

Programa creado por Macromedia para elaborar animaciones interactivas para la Web que ofrece posibilidades versátiles.

Los gráficos que crea Flash son del tipo vectorial, lo cual implica que se podrán redimensionar sin producir perdida de calidad; esto hace que su adaptación al tamaño de la pantalla del usuario se produzca con suma facilidad, además lo archivos resultantes son pequeños lo cual permite una rápida descarga de estos a través de la red.

ActionScript el lenguaje de programación.

Todas las acciones de flash se basan en el lenguaje ActionScript, este trabaja con un código interpretado similar a *JavaScript* o *VBScript*, cada fragmento del programa es leído por un *motor/interprete*- incluido en el visor de Flash o en los conectores de Flash de los Navegadores de Internet- que ejecuta las acciones traduciéndolas a un lenguaje propio de la plataforma.

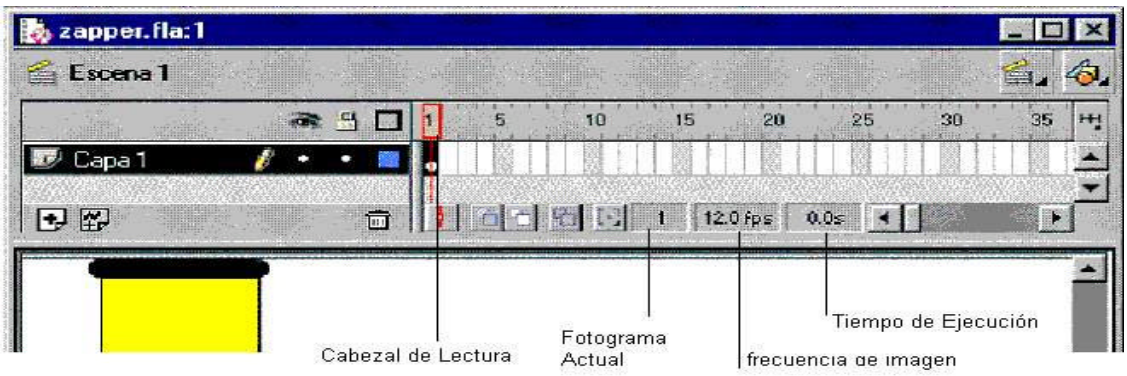
Los lenguajes compilados generan el código máquina al crearse la aplicación y se distribuyen sin necesidad de un interprete, lo cual les da la ventaja de ser más rápidos, pero dependen de la plataforma para la cual hayan sido creados. El código ActionScript es independiente de la máquina y el sistema operativo sobre el que se vaya a ejecutar.

La línea de tiempo

La *línea de tiempo* nos permite organizar el contenido de la película y controlar su ejecución a través del tiempo. Una película esta compuesta por una o mas capas las cuales pueden tener uno o mas fotogramas (espacio tiempo), lo que vemos en un instante dado depende de los fotogramas existentes y del orden en que se hayan dispuesto la ejecución.

El escenario

El *Escenario* es la parte de la ventana de la aplicación en la que se componen los fotogramas individuales de la película, bien dibujándolos con las herramientas del propio flash o bien insertando imágenes creadas con otra aplicación y que nosotros importemos.



Sintaxis general de ActionScript.

Normas sintácticas de carácter general :

- Nombres de variables, constantes, funciones, objetos y otros, deben ser de una única palabra y sin espacios.
- El uso de mayúsculas y/o minúsculas, es indiferente, es decir `Variable` es igual que `VARIABLE`.
- Al final de cada sentencia se debe poner punto y coma.

```
- Valor = 5*valor;
```

- Se pueden escribir varias sentencias en una misma línea.

```
- Valor = 5*valor; hola = 5;
```

? Caracteres admisibles como variables son las letras, números y cualquier otro que no tenga un significado específico, como el signo más, coma, punto, paréntesis, similares y además existen palabras reservadas.

break	for	new	var
continue	function	return	void
delete	if	this	while
else	in	typeof	with

? Comentarios, para comentar sobre una línea se puede utilizar `//comentario` y para un comentario de más de una línea `/*comentario*/`

Sintaxis de punto .

? La sintaxis de punto es similar a la POO de *Java* o *C++*, permite relacionar objetos predefinidos que llevan asociados propiedades, acciones, métodos y/o constantes-, que se referencian mediante la sintaxis:

- `ángulo = Math.PI/2;`
- `Seno = Math.sin(ángulo);`

Tipos de datos. Variables y Operadores.

Las variables nos permite almacenar valores diversos, datos numéricos o cadenas de caracteres; son la base de la creación de códigos interactivos, permiten controlar la ejecución de películas, por ejemplo actuando como contadores, tomando entradas de usuario, etc. Mediante operadores podremos calcular y evaluar datos, con el anexo viene una lista de Operadores.

? Una cadena es una secuencia de caracteres tales como letras, números y signosde puntuación. Las cadenas se introducen en una sentencia de ActionScript incluyéndolas entre comillas simples o dobles. A las cadenas se les trata como caracteres en lugar de como variables. Por ejemplo:

- `nombre=”juanito”;`

Puede utilizar el operador de suma (+) para concatenar o unir dos cadenas. ActionScript trata los espacios del comienzo o del final de una cadena como parte literal de la cadena. La siguiente expresión incluye un espacio después de la coma:

```
- greeting = "Welcome, " + firstName;
```

? El tipo de dato numérico es un número flotante de doble precisión. Puede manipular los números utilizando los operadores aritméticos de suma (+), resta (-), multiplicación, (*), división (/), módulo (%), incremento (++) y decremento (--). También puede utilizar los métodos del objeto predefinido Math para manipular los números. El siguiente ejemplo utiliza el método sqrt (raíz cuadrada) para devolver la raíz cuadrada del número 100:

```
- valor = Math.sqrt(100);
```

? Un valor Booleano es aquel que es verdadero o falso. ActionScript también convierte los valores true y false en 1 y 0 cuando sea adecuado. Los valores booleanos se usan con mayor frecuencia con los operadores lógicos en sentencias de ActionScript que realizan comparaciones para controlar el flujo de un script. Por ejemplo, en el siguiente script, la película se reproduce si la variable userName y password son true:

```
- if ((userName == true) && (password == true)){
  play();
 }
```

Scripts orientados a objetos

En los scripts orientados a objetos la información se organiza en grupos denominados clases. Pueden crearse varias instancias de una clase; a éstas se les denomina objetos, y los objetos pueden emplearse en los scripts. Un objeto es un conjunto de propiedades. Cada propiedad tiene un nombre y un valor. El valor de la propiedad puede ser de cualquier tipo de dato Flash, incluso el tipo de datos de objeto. Esto permite organizar a los objetos uno dentro de otro (anidarlos). Para especificar objetos y sus propiedades, debe utilizar el operador punto (.). Por ejemplo: `hoursWorked` es una propiedad de `weeklyStats`, que a su vez es una propiedad de `employee`: `employee.weeklyStats.hoursWorked`.

Puede utilizar los objetos predefinidos de ActionScript para acceder y manipular tipos de información específicos. Por ejemplo, el objeto `Math` tiene métodos que realizan operaciones matemáticas con los números que le pasan, ejemplo, `sqrt`:

```
- squareRoot = Math.sqrt(100);
```

Se pueden utilizar clases predefinidas de ActionScript y/o crear propias. Cuando se crea una clase, es necesario definir todas las propiedades y métodos de cada objeto que ésta creando, de modo análogo a como se definen los objetos en el mundo real. Una persona, por ejemplo, tiene ciertas propiedades como el género, la altura y el color de pelo, así como ciertos métodos (comportamientos) como son hablar, caminar y correr. En este ejemplo "persona" es, genéricamente, una clase y cada persona individual es un objeto, o una instancia de dicha clase. Los objetos de ActionScript pueden contener datos o pueden representarse gráficamente en el Escenario como clips de película. Todos los clips de película son instancias de la clase predefinida MovieClip. Cada instancia de clip de película contiene todas las propiedades `(por ejemplo, _height, _rotation,_totalframes)` y todos los métodos `(por ejemplo, gotoAndPlay, loadMovie,startDrag)` de la clase `MovieClip.` Para definir una clase, debe crearse una función especial, denominada función constructora; las clases predefinidas tienen funciones constructoras previamente definidas. Por ejemplo, si se desea obtener información acerca de un ciclista que aparece en la película, puede crearse una función constructora, `ciclista,` con las propiedades `time` y `distance` y el método `rate,` que indique la velocidad a la que se desplaza el ciclista:

```
- function ciclista(t, d) {
   this.time = t;
   this.distance = d;
   }
- function Speed() {
   return this.time / this.distance;
  }
- ciclista.prototype.rate = Speed;
```

Se pueden crearse copias, es decir, instancias, de la clase. El siguiente código crea instancias del objeto `ciclista` denominadas `pepito` y `juanito.`

```
- pepito = new ciclista(30, 5);
- juanito = new ciclista (40, 5);
```

En la creación de scripts orientados a objetos, las clases pueden recibir propiedades y métodos unas de otras, de acuerdo a un orden determinado; esta funcionalidad se denomina herencia. La herencia puede utilizarse para ampliar o redefinir las propiedades y métodos de una clase. Una clase que hereda propiedades y métodos de otra clase recibe el nombre de subclase. Una clase que pasa propiedades y métodos a otra clase recibe el nombre de superclase.

Declaración de variables

Para declarar variables globales, utilice la acción `setVariables` o el operador (=). Ambos métodos logran el mismo resultado. Para declarar una variable local, utilice la sentencia `var` dentro de una función. El ámbito de las variables locales es el contexto del bloque de código y expiran al final del mismo. Las variables locales que no han sido declaradas dentro de un bloque de código expiran cuando finaliza el script en que se utilizan.

```
- function zeroArray (array){
   var i;
   for (i=0; i < array.length; i++) {
   array[i] = 0;
  }
  }
```

Condicionales y Recurrentes

ActionScript dispone de las acciones `if, else, for, for in, while y do while.` Las cuales nos permitirán controlar acciones repetitivas y realizar bifurcaciones en la ejecución del código.

```
- if (n<10){              //si n menor que diez
     gotoAndplay(1);    //ir al fotograma 1
  }
```

Acciones

Las acciones son sentencias o comandos de ActionScript. Si se asignan varias acciones al mismo fotograma u objeto se crea un script. Las acciones pueden actuar independientemente unas de otras, como se muestra en las siguientes sentencias:

```
- swapDepths("mc1", "mc2");
- gotoAndPlay(15);
```

También puede anidar acciones utilizando una acción dentro de otra, esto permite que las acciones afecten unas a las otras. Las acciones pueden mover la cabeza lectora en la Línea de tiempo (`gotoAndPlay`), controlar el flujo de un script creando recurrencia (`do while`) o lógica condicional (`if`), o crear nuevas funciones y variables (`function`, `setVariable`). Ver la tabla de acciones de ActionScriptque acompaña en el anexo.

Funciones predefinidas

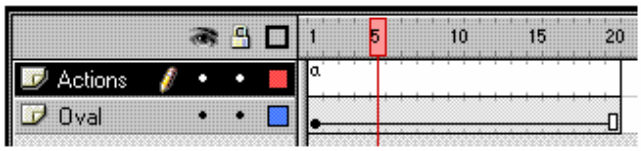
Flash dispone de funciones predeterminadas que le permiten acceder a cierta información y realizar ciertas tareas, como la detección de choque (`hitTest`), la obtención del valor de la última tecla presionada (`keycode`), `etc.` Una función de ActionScript se puede volver a utilizar en cualquier parte de una película. Ver la tabla de funciones en el anexo.

Llamada función

Se Puede llamar a una función desde cualquier Línea de tiempo, incluida una película que ha sido cargada. Cada función tiene sus propias características y algunas necesitan que se le pasen ciertos valores. Si se le pasan más argumentos de los que necesita la función, los valores sobrantes se ignorarán. Si no se le pasa un argumento necesario, los argumentos vacíos se asignarán al tipo de datos undefined, que puede causar errores cuando exporte un script. Para llamar a una función, ésta debe encontrase en un fotograma al que haya llegado la cabeza lectora.

Asignación de acciones a fotogramas

Si se desea que una película realice un acción específica en un fotograma clave, se le debe asignar una acción al fotograma clave. Por ejemplo, para crear una función recurrente entre los fotogramas 20 y 10 de la Línea de tiempo, se puede agregar la acción `gotoAndPlay (10)` al fotograma 20.



Una "a" en un fotograma clave indica una acción de fotograma.

Para asignar una acción a un fotograma: Se debe Seleccionar un fotograma clave en la Línea de tiempo y elegir Ventana > Acciones, y luego editar una en el panel de acciones el cual es un editor de ActionScript que ligara el script con el fotograma.

Programación de una ruta de destino

Para utilizar una acción con el fin de controlar un clip de película o una película que ha sido cargada, debe especificar su nombre y su dirección. Acciones que toman una o más rutas de destino como argumentos:

- loadMovie
- loadVariables
- unloadMovie
- setProperty
- startDrag
- duplicateMovieClip
- removeMovieClip
- print
- printAsBitmap
- tellTarget

Por ejemplo, la acción loadMovie toma los argumentos URL, Location, y Variables. La dirección *URL* es la ubicación en la Web de la película que desea cargar. *Location* es la ruta de destino en la que se cargará la película.

```
- loadMovie(URL, Location, Variables);
```

La siguiente sentencia carga la dirección *URL* http://www.valderas.com/miPeli.swf en la instancia bar en la Línea de tiempo principal, `_root; _root.bar` es la ruta de destino;

```
-loadMovie("http://www.mySite.com/myMovie.swf",_root.bar);
```

En ActionScript se identifica a un clip de película por su nombre de instancia. Por ejemplo, la

propiedad `_alpha` del clip de película llamado `estrella` está establecida a un 50% de visibilidad: `estrella._alpha = 50;`

Como dar un nombre de instancia a un clip de película:

1 Seleccionar el clip de película en el Escenario.

2 Seleccionar Ventana > Paneles > Instancia.

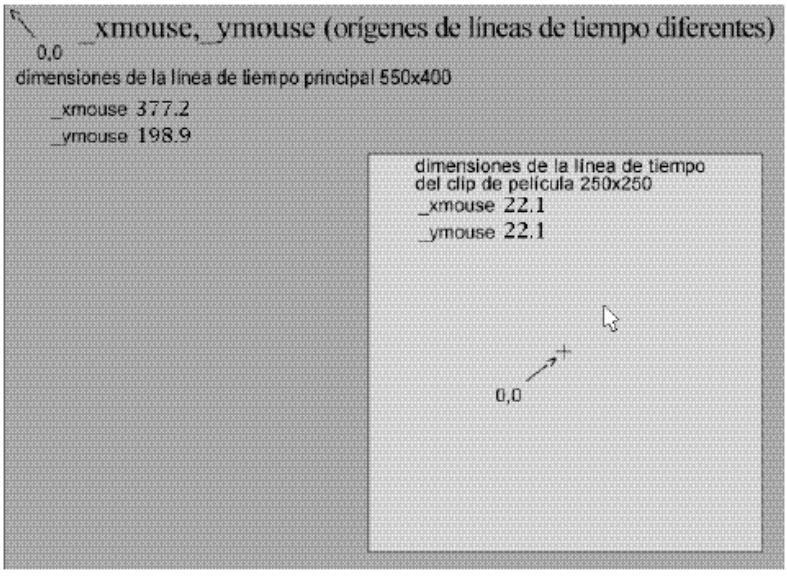
3 Introducir un nombre de instancia en el campo Nombre.

Análisis de unos script de ejemplo

Analizaremos 2 películas sencillas en las cuales se muestran algunos de los conceptos aprendidos. La idea de los ejemplos es concretizar más lo antes mencionado y dar ideas reales de uso de ActionScript.

Obtención de la posición del ratón

Las propiedades _xmouse e _ymouse se pueden usar para encontrar la ubicación del puntero del ratón (cursor) en una película. Cada Línea de tiempo tiene una propiedad _xmouse e _ymouse que devuelven la ubicación del ratón dentro de su sistema de coordenadas.



La siguiente sentencia podría colocarse en cualquier Línea de tiempo de la película para que devuelva la posición `_xmouse` en la Línea de tiempo principal:

```
- x_pos = _root._xmouse;
```

Para determinar la posición del ratón dentro de un clip de película, puede utilizar el nombre de instancia del clip de película. Por ejemplo, la siguiente sentencia podría colocarse en cualquier Línea de tiempo de la película para que devuelva la posición `_ymouse` en la instancia miClip:

```
- y_pos = _root.miClip._ymouse
```

También puede determinar la posición del ratón en un clip de película utilizando las propiedades `_xmouse` e `_ymouse` en una acción de clip, como se muestra a continuación:

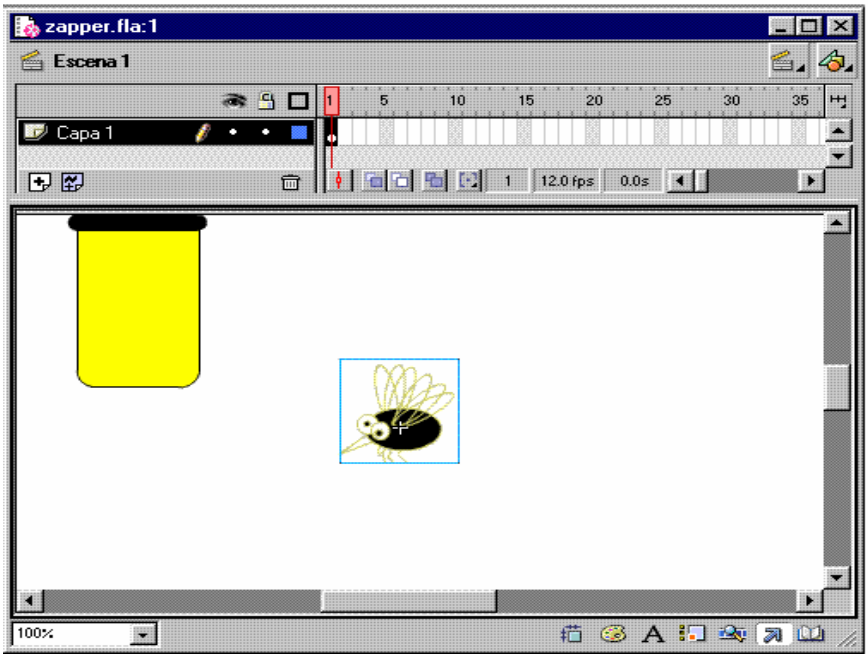
```
- onClipEvent(enterFrame){
  xmousePosition = _xmouse;
  ymousePosition = _ymouse;
  }
```

Las variables `x_pos` e `y_pos` se utilizan como contenedores para guardar los valores de las posiciones del ratón y la vez son instancias de textos dinámicos en la película lo que implica que los scripts están ligados a un texto dinámico, lo que permite que se puedan reflejar los cambios en las variables `x_pos` e `y_pos` por pantalla . Puede utilizar estas variables en cualquier script de su película. En el siguiente ejemplo, los valores de `x_pos` e `y_pos` se actualizan cada vez que el usuario mueve el ratón.

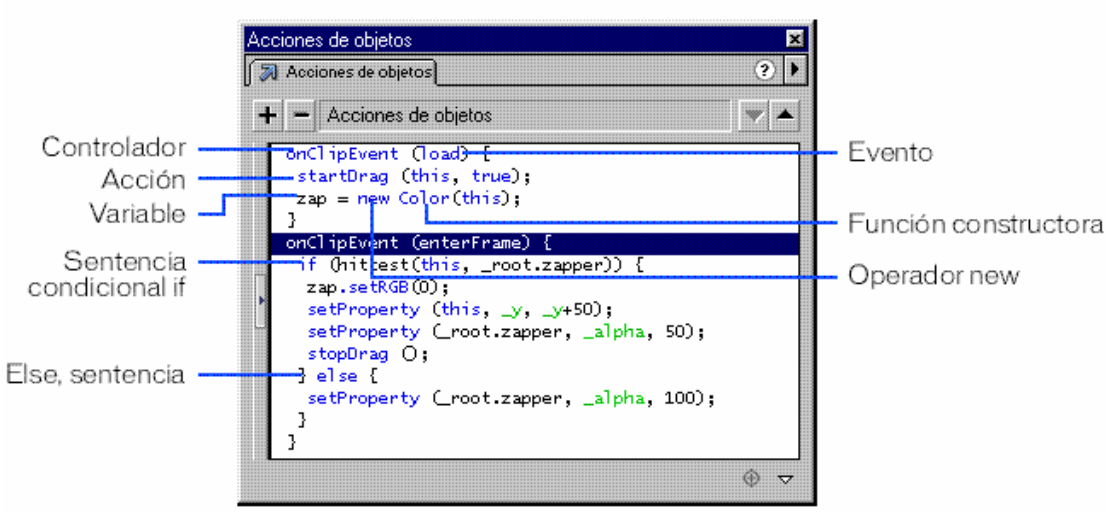
```
- onClipEvent(mouseMove){
  x_pos = _root._xmouse;
  y_pos = _root._ymouse;
  }
```

El mosco y el matamoscas

La película tiene una longitud de un fotograma y contiene dos objetos, la instancia del clip de película del insecto y la instancia del clip de película del matamoscas. Cada clip de película contiene un fotograma. Cuando un usuario arrastra el insecto hacia el matamoscas, el insecto comienza a cambiar de color hasta llegar a negro y caer, y el matamoscas parece parpadear.



Sólo hay un script en la película y está asociado a la instancia del insecto:



Ambos objetos deben ser clips de película, de forma que sea posible asignarles nombres de instancia en el panel Instancia y manipularlos mediante ActionScript. El nombre de instancia del insecto es bug y el del matamoscas es zapper. En el script, se hace referencia al insecto como this porque el script está anexado al insecto y la palabra reservada `this` hace referencia al objeto que lo llama. Hay dos controladores `onClipEvent` con dos eventos diferentes: `load` y `enterFrame`. Las acciones de la sentencia `onClipEvent(load)` se ejecutan sólo una vez, al cargarse la película. Las acciones de

la sentencia `onClipEvent(enterFrame)` se ejecutan cada vez que se accede a un fotograma. Incluso aunque se trate de películas de un solo fotograma, se accederá al fotograma reiterativamente y el script se ejecutará de la misma forma. Dentro de cada controlador `onClipEvent` se llevan a cabo las siguientes acciones:

`onClipEvent(load)`, la acción `startDrag` hace posible que se pueda arrastrar el clip de película del insecto. El operador `new` y la función constructora de color, `Color`, crean una instancia del objeto Color y la asignan a la variable zap:

```
- onClipEvent (load) {
  startDrag (this, true);
  zap = new Color(this);
  }
```

`onClipEvent(enterFrame)` Una sentencia condicional `if` evalúa una acción `hitTest` para comprobar si la instancia del insecto `(this)` está tocando a la instancia del matamoscas `(_root.zapper)`. La evaluación puede generar dos posibles resultados, verdadero o falso:

```
onClipEvent (enterFrame) {
if (hitTest(_target, _root.zapper)) {
zap.setRGB(0);
setProperty (_target, _y, _y+50);
setProperty (_root.zapper, _alpha, 50);
stopDrag ();}
else {
setProperty (_root.zapper, _alpha, 100);
}
}
```

Si la acción `hitTest` devuelve `true`, el objeto `zap` creado por el evento `load` se utilizará para cambiar el color del insecto a negro. La propiedad *y* `(_y)` del insecto se establece a `50`, de modo que el insecto caiga. La transparencia del matamoscas `(_alpha)` se establece al valor 50, de modo que se atenúe. La acción `stopDrag` impide que se arrastre el objeto. Si la acción `hitTest` devuelve el valor `false`, se ejecutará la acción que sigue ala sentencia `else` y el valor `_alpha` del matamoscas se establecerá al

valor 100. Esto provocará que el matamoscas aparezca intermitentemente, ya que su valor `_alpha` pasará de un estado inicial (100) a un estado activo (50), volviendo posteriormente al estado inicial. La acción `hitTest` arroja como resultado el valor `false` y las sentencias `else` se ejecutan una vez que se ha golpeado el insecto.

Publicación de una película

Una vez terminada la película se puede proceder a publicar ésta, es decir convertirla en una página Web, los pasos a seguir son sumamente simples basta con guardar el archivo con extensión .swf y luego ir a `Archivo > Publicar` y ya.

Conclusion

Este trabajo a sido una experiencia muy enriquecedora me ha permitido conocer en mayor profundidad un tema que para mi antes resultaba ser un poco enredado la verdad es que si no se lee un manual, libro o guía es imposible lograr algo (entonces ni hablar del milenario método de prueba y error).

Es importante resaltar la tremenda efectividad que tiene flash para crear animaciones, se pueden lograr un sin numero de trabajos con resultados de gran calidad y detalle, que además resultan pequeños en tamaño lo cual permite una rápida descarga de estos desde cualquier sitio es por esto el gran auge de Flash.

Referencias

[1] Jaime Peña, Maria Carmen Vidal, ”Guia de Aprendizaje, Flash”, McGraw-Hill.

[2] Macromedia, “Guia de Consultas ActionScript”, Macromedia.

[3] http://www.actionscript.org

Anexo

1. Lista de Acciones de AcionScript:

Acciones				
break	evaluate	include	print	stopDrag
call	for	loadMovie	printAsBitmap	swapDepths
comment	for...in	loadVariables	removeMovieClip	tellTarget
continue	fsCommand	nextFrame nextScene	return	toggleHighQuality
delete	function	on	setVariable	stopDrag
do...while	getURL	onClipEvent	setProperty	trace
duplicateMovieClip	gotoAndPlay gotoAndStop	play	startDrag	unloadMovie
else	if	prevFrame	stop	var
else if	ifFrameLoaded	prevScene	stopAllSounds	while

2. Esta tabla enumera todos los operadores de Action Script y su asociatividad, desde la precedencia más alta a la más baja.

Operador	Descripción	Asociatividad
	Precedencia más alta	
+	Más unario	De derecha a izquierda
-	Menos unario	De derecha a izquierda
~	Complemento a uno como bit	De derecha a izquierda
!	NOT lógico	De derecha a izquierda
not	NOT lógico (estilo de Flash 4)	De derecha a izquierda
++	Incremento posterior	De izquierda a derecha
--	Decremento posterior	De izquierda a derecha
( )	Llamada a función	De izquierda a derecha

Operador	Descripción	Asociatividad
[ ]	Elemento de matriz	De izquierda a derecha
.	Miembro de una estructura	De izquierda a derecha
++	Incremento previo	De derecha a izquierda
--	Decremento previo	De derecha a izquierda
new	Asignar objeto	De derecha a izquierda
delete	Anular la asignación de objeto	De derecha a izquierda
typeof	Tipo de objeto	De derecha a izquierda
void	Devuelve un valor no definido	De derecha a izquierda
*	Multiplicación	De izquierda a derecha
/	División	De izquierda a derecha
%	Módulo	De izquierda a derecha
+	Suma	De izquierda a derecha
add	Concatenación de cadenas (anteriormente &)	De izquierda a derecha
-	Resta	De izquierda a derecha
<<	Desplazamiento a la izquierda como bit	De izquierda a derecha
>>	Desplazamiento a la derecha como bit	De izquierda a derecha

Operador	Descripción	Asociatividad
>>>	Desplazamiento a la derecha como bit (sin signo)	De izquierda a derecha
<	Menor que	De izquierda a derecha
<=	Menor o igual que	De izquierda a derecha
>	Mayor que	De izquierda a derecha
>=	Mayor o igual que	De izquierda a derecha
lt	Menor que (versión para cadenas)	De izquierda a derecha
le	Menor o igual que (versión para cadenas)	De izquierda a derecha
gt	Mayor que (versión para cadenas)	De izquierda a derecha
ge	Mayor o igual que (versión para cadenas)	De izquierda a derecha
==	Igual	De izquierda a derecha
!=	Distinto	De izquierda a derecha
eq	Igual (versión para cadenas)	De izquierda a derecha
ne	Distinto (versión para cadenas)	De izquierda a derecha
&	AND como bit	De izquierda a derecha
^	XOR como bit	De izquierda a derecha
|	OR como bit	De izquierda a derecha

Operador	Descripción	Asociatividad
&&	AND lógico	De izquierda a derecha
and	AND lógico (estilo de Flash 4)	De izquierda a derecha
||	OR lógico	De izquierda a derecha
or	OR lógico (estilo de Flash 4)	De izquierda a derecha
?:	Condicional	De derecha a izquierda
=	Asignación	De derecha a izquierda

3. Funciones predeterminadas por Flash.

Boolean	getTimer	isFinite	newline	scroll
escape	getVersion	isNaN	number	String
eval	globalToLocal	keycode	parseFloat	targetPath
false	hitTest	localToGlobal	parseInt	true
getProperty	int	maxscroll	random	unescape