

ACTION SCRIPT 2.0

PARA FLASH

José Lavado Guzmán
j_antonio76@hotmail.com

INTRODUCCION

En este capítulo aprenderemos a conseguir que Flash analice la situación y decida que hacer, hasta el momento, hemos visto algunas formas de controlar Flash usando acciones, eventos y variables. Las acciones son como órdenes, que le dicen a Flash lo que tiene que hacer. Los eventos permiten reaccionar en tiempo real a cosas como presionar un botón o pasar a un nuevo fotograma y finalmente las variables dotan a Flash de memoria de manera que podemos almacenar información para usarla cuando la necesitemos.

Con todo esto, tenemos mucho con lo que jugar, pero no está exento de problemas, recuerde la clase numero dos, cuando tuvimos que desactivar la casilla de verificación de Ajuste Automático para los cuadros de entradas de datos.

El array de la clase anterior funcionaba bien, pero que sucedería si alguien colocaba en vez de 1 2 ó 3; 100 ,200 ó 1000. No creen que debería aparecer una nota o mensaje diciendo que esos valores no son permitidos.

Muy bien, aquí viene entonces, lo que seguramente ya te has dado cuenta, es que necesitamos una manera de saber si los datos introducidos son o no correctos, y así poder generar una respuesta. Dicho claramente Flash necesita tomar decisiones.

CAPITULO IV

Condicionales

ActionScript dispone de una estructura condicional que permite que la películas que creamos actúen de una forma o de otra dependiendo de que una determinada condición se cumpla o no.

Sintaxis 1	Sintaxis En ActionScript 1
<pre>Si (Condición){ realizarAcciones; realizarAcciones; } }</pre>	<pre>if (Condición){ realizarAcciones; } }</pre>
Sintaxis 2:	Sintaxis En ActionScript 2
<pre>Si (Condición){ realizarAcciones; realizarAcciones; }Si No { realizarOtrasAcciones; realizarOtrasAcciones; } }</pre>	<pre>if (Condición){ realizarAcciones; realizarAcciones; }else{ realizarOtrasAcciones; realizarOtrasAcciones; } }</pre>
Sintaxis 3:	Sintaxis En ActionScript 3
<pre>Si (Condición 1){ realizarAcciones 1; realizarAcciones 1; }Si No Si (Condición 2){ realizarAcciones 2; realizarAcciones 2; } Si No Si (Condición 3){ realizarAcciones 3; realizarAcciones 3; }Si No{ realizarOtrasAcciones; realizarOtrasAcciones; } }</pre>	<pre>if (Condición 1){ realizarAcciones 1; realizarAcciones 1; }else if (Condición 2){ realizarAcciones 2; realizarAcciones 2; }else if (Condición 3){ realizarAcciones 3; realizarAcciones 3; }else{ realizarOtrasAcciones; realizarOtrasAcciones; } }</pre>

Ejemplos:

Declararemos una variable con un valor numérico, por ejemplo 15.

Luego, mediante el uso de las condicionales IF, sabremos si esa variable es mayor a 10, de ser así deberá enviar un mensaje.

Observemos el código.

```
a= 15;

If (a > 10) {
    trace ("A es mayor a que 10");
}
```

Ahora lo haremos un poquito mas interesante, en este caso especificamos que **si** A es mayor a 10 , se muestre el mensaje "A es mayor a 10" en la ventana de Salida; y **si no** lo es, que se muestre "A es menor o igual que 10", ojo que aquí cambiaré el valor de A.

```
a= 6;

If (a > 10) {
    trace ("A es mayor a que 10");
} else {
    trace ("A es menor o igual a 10");
}
```

Y en este ultimo código utilizaremos la tercera sintaxis, donde colocaremos el valor de A en 45, y se establecerán distintos rangos de valores en los que pueda encontrarse el valor de A, dependiendo del cual se mostrará uno de los cuatro mensajes.

```
a= 45;

If (a <= 25) {
    trace ("A es menor o igual a 25");
} else if (a <= 50){
    trace ("A es menor o igual a 50 y mayor que 25");
} else if (a <= 75){
    trace ("A es menor o igual a 75 y mayor que 50");
} else {
    trace(" A es mayor a 75");
}
```

Practica:

Realizar un ejercicio donde se determine el mayor de dos números.

Realizar un ejercicio donde se determine el mayor de tres números.

Realizar un ejercicio donde se controle el acceso de un usuario mediante su clave.

Eventos Internos

Ha sido fácil entender la relación *Evento – Acciones dentro del evento*, en eventos externos como es pulsar un botón. Sin embargo, los eventos internos no son tan obvios ya que las películas los generan automáticamente.

Probablemente el evento más fácil sea *onEnterFrame*, que se genera cada vez que la cabeza lectora de la línea de tiempo entra en un nuevo fotograma. Bien, pero *¿Qué significa en términos prácticos?* Significa que es extremadamente fácil escribir un trozo de script para que Flash lo ejecute en cada fotograma.

Imagine que la línea de tiempo vaya al doble de velocidad:

_currentframe: se refiere al número del fotograma en el que estamos, así que podríamos escribir un código para *onEnterFrame* que contenga la acción *gotoAndStop(_currentframe + 2)*.

Explicación:

- La película empieza y se genera un evento *onEnterFrame*.
- Se acciona *gotoAndStop(1 + 2)*, yendo al fotograma número 3.
- Esto genera otro evento *onEnterFrame*.
- Se acciona *gotoAndStop(3 + 2)*, yendo al fotograma 5.
- Y así sucesivamente....

Básicamente nos saltamos la mitad de los fotogramas, por lo que parece que la película va el doble de rápido.

Ejercicio:

En una película haga que la línea de tiempo vaya a la mitad de su velocidad.

Ejemplo:

Escena de 400 * 200

Dibuja un cuadrado y conviértelo a clip de película, asigna un nombre de instancia *clip1_mc*. Coloca el siguiente código en el primer fotograma.

```
onEnterFrame = function(){
    clip1_mc._rotation = clip1_mc._rotation + ((_xmouse - 200)/10);
}
```