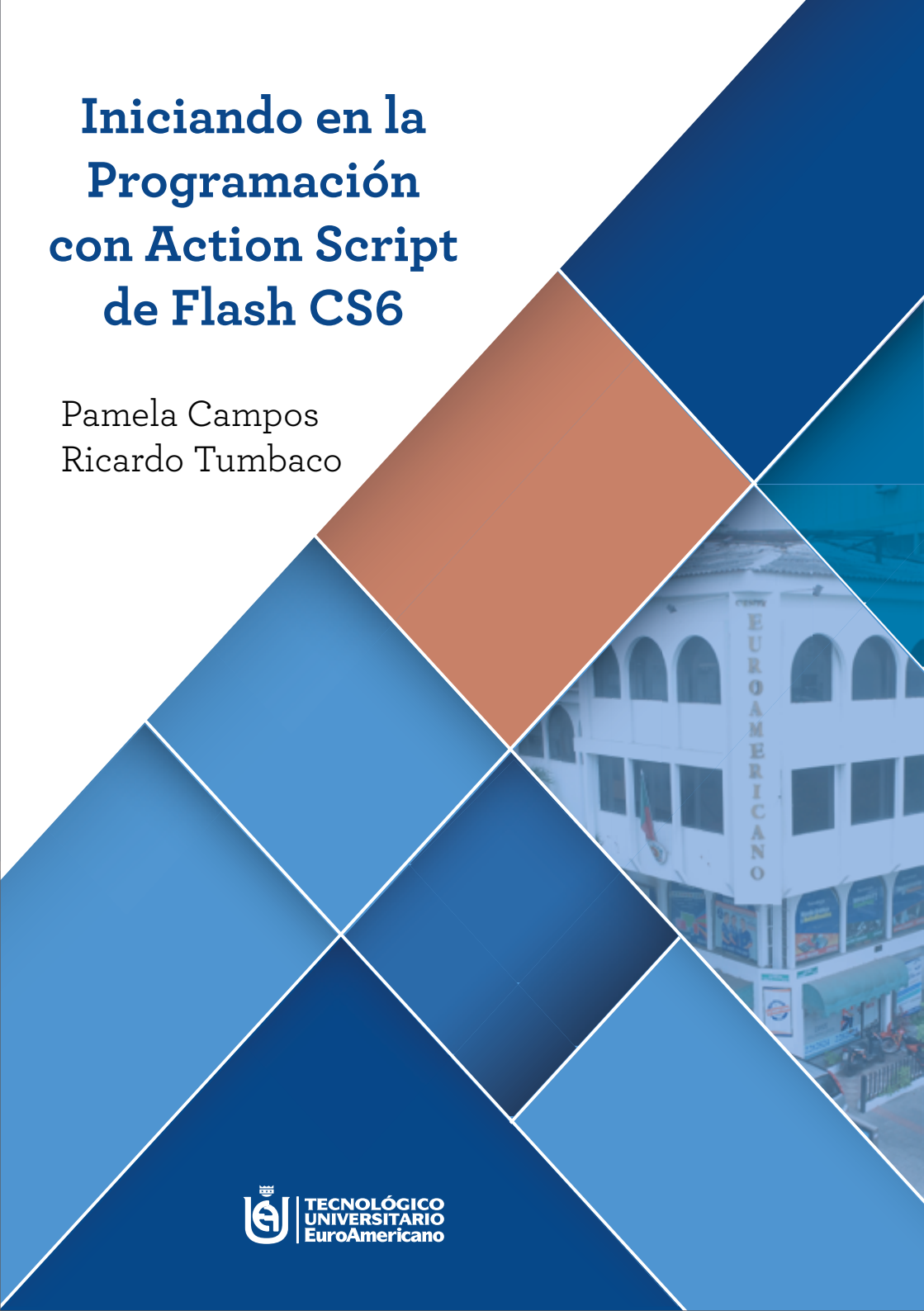
Iniciando en la Programación con Action Script de Flash CS6

Pamela Campos
Ricardo Tumbaco



TECNOLÓGICO UNIVERSITARIO EuroAmericano

Iniciando en la Programación con Action Script de Flash CS6

Tnlga. Pamela Campos Guillén
Lcdo. Ricardo Tumbaco Orellana

Tecnológico Universitario EuroAmericano

DIRECCIÓN:
Quisquís 1317 y Los Ríos
Guayaquil – Guayas - Ecuador
(+593) 04-2288-440
www.euroamericano.edu.ec

RECTOR:
Mgtr. Antonio Manuel Marques Gutiérrez

AUTOR:
Tnlga. Pamela Campos Guillén
COAUTOR:
Lcdo. Ricardo Tumbaco Orellana

CORREO:
pcampos@euroamericano.edu.ec
rtumbaco@euroamericano.edu.ec

Primera Edición – marzo 2023
Editorial "R2ICS" | Pichincha | Quito | Ecuador

Datos de catalogación bibliográfica

CAMPOS, P. & TUMBACO, R.

Iniciando en la Programación con Actionscript de Flash CS6
Primera Edición
Quito, Ecuador, 2023
Editorial: Red Internacional de Investigación en Ciencias Sociales y Humanidades "R2ICS"

ISBN: 978-9942-7125-8-5
Área: Tecnologías
Formato A5: 148 x 210 mm
Páginas: 110

ISBN: 978-9942-7125-8-5
9789942712585

Diseño y maquetación R2ICS

Índice

Contenido

Prólogo

Prólogo

Mediante programas como Adobe Flash o Adobe Anímate podemos combinar un lenguaje de programación y motor de gráficos para la creación de programas y productos interactivos como cuentos interactivos o juegos, es decir, entornos multimedia programables.

Para ello el lenguaje que nos permitirá realizar dichas interacciones es Action Script. Este lenguaje basado en C permite crear videojuegos, presentaciones especializadas, cuentos interactivos, formularios, etc.

Action Script es considerado como un lenguaje de programación dirigido a objetos. Esto quiere decir que todo lo que se llegue a programar con este es la interacción del usuario con cualquier elemento o animación.

Introducción

Introducción

ActionScript es el lenguaje de programación que Flash ha utilizado desde sus inicios y por supuesto utiliza Flash CS6. En general podemos decir que ActionScript nos permitirá hacer lo que queramos con Flash, ya que nos da un control absoluto sobre todo lo que rodea a la animación.

Sin embargo, en estos temas, solo veremos una breve introducción a ActionScript 3 para sentar las bases para que comience a trabajar con ActionScript.

ActionScript como su nombre indica es un lenguaje de scripting, esto significa que no es necesario crear un programa completo para lograr el resultado, es común aplicar fragmentos de código ActionScript a objetos existentes en la película que nos permitirá lograr nuestra meta.

La sintaxis de ActionScript tiene muchas similitudes con Javascript o PHP; Si conocemos estos lenguajes, la sintaxis y el estilo de ActionScript nos resultarán muy familiares.

Capítulo 1

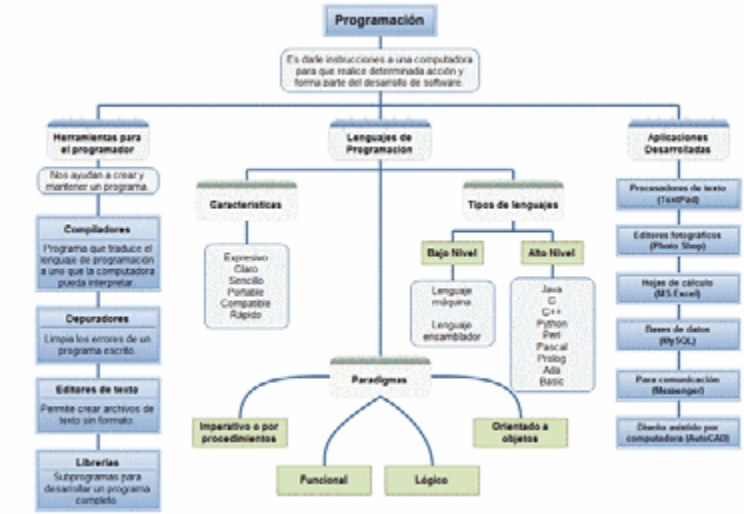
Introducción a Lenguajes de Programación

¿Qué es un lenguaje de programación?

Un lenguaje de programación es un lenguaje formal (es decir, un lenguaje con reglas gramaticales bien definidas) que, con la ayuda de una serie de instrucciones, ayuda al programador escribir o programar una serie de órdenes, acciones consecutivas, datos o algoritmos con la finalidad de dirigir el comportamiento físico y/o lógico de una máquina, computadora o dispositivo. A dichos grupo de órdenes escritas mediante algún lenguaje de programación le llamamos programa.

Figura 1

Diagrama de programación sobre los diversos tipos de lenguajes



Nota: Tomado de Barzanallana, 2014. Tipos de lenguaje de programación y clases.

Mediante este tipo de lenguajes se comunican el programador y la máquina. Logrando obtener de forma precisa aspectos como:

- Los datos que van a utilizarse mediante un software particular.
- Como dichos datos deben ser guardados y/o transmitidos.
- Las acciones se deben tomar según las circunstancias

o variables presentes.

Por ello programar es el proceso de crear un software preciso a través de la escritura, la prueba y depuración, la compilación y mantenimiento del código fuente de tal programa informático. Este proceso es llevado a cabo aplicando los siguientes pasos:

- El desarrollo lógico del software para resolver problemas específicos.
- Escritura lógica del software utilizando un lenguaje de programación en particular.
- Compilación del programa y el código para convertirlo en lenguaje de máquina.
- Prueba y depuraciones del software.
- Desarrollo de documentación del programa.

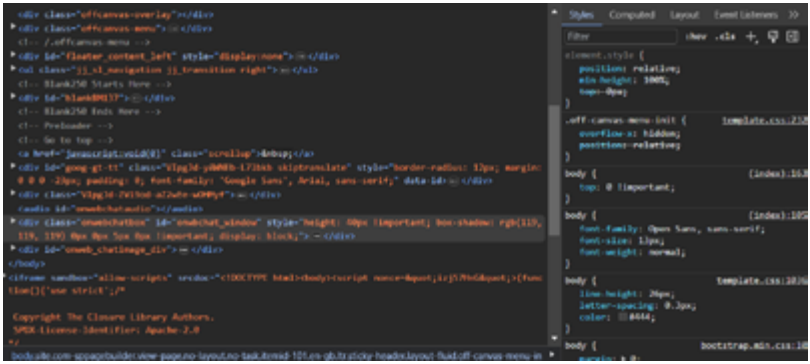
Los lenguajes de programación se constituyen por varios conjuntos de símbolos llamados alfabeto, estos crean reglas gramaticales de carácter léxico/morfológicas, semánticas y sintácticas. Lo cual define dichas estructuras como válidas para el lenguaje y su significado la programación sigue a estos pasos o conjuntos de reglas de manera estricta para que nos permitan expresar las instrucciones que serán interpretadas por el programador.

Los lenguajes informáticos comprenden otros lenguajes que propician una estructura para dar formato a un texto, pero estos no son lenguajes de programación por sí mismos.

Cabe recalcar que no todos los lenguajes informáticos se les debe considerar de programación, pero todos los lenguajes de programación si son informáticos. Por ello existe el error común de tratarlos como sinónimos. Los lenguajes informáticos abarcan a los lenguajes de programación. Tenemos el ejemplo de HTML, que es un lenguaje de marcado de páginas web el cual no es propiamente un lenguaje de programación, sin embargo, si es un conjunto de instrucciones.

Figura 2

Imagen Html, un "lenguaje" bastante conocido



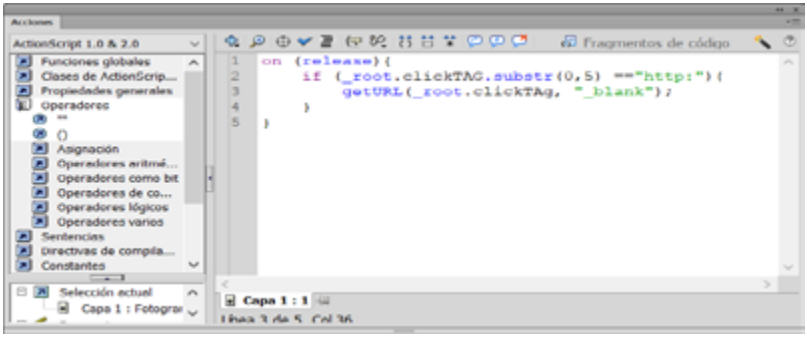
Nota: Tomado de (iStock, 2015). Diferencias entre lenguajes de programación y lenguajes informáticos.

Un lenguaje de programación nos da paso para especificar de forma clara sobre qué tipo de información va a operar algún programa o software específico. A la vez de saber cómo deben estos datos ser guardados y/o compartidos; aparte de las acciones a tomar de parte del software según una lista de posibles eventos o circunstancias. Todo esto último expresado a través de comandos que intentan estar casi próximos al lenguaje humano o natural.

Una característica importante de estos tipos de lenguajes es que más de un programador lo pueda emplear y conocer siguiendo un conjunto establecido de instrucciones. Eso ayuda a que el lenguaje sea comprensible al momento de realizar un programa de manera colaborativa o grupal.

Figura 3

ActionScript es un lenguaje dirigido a objeto



Clasificación de los lenguajes de programación

Los lenguajes de programación se clasifican según distintos criterios como:

Clasificación histórica

Gracias a que aparecieron los nuevos lenguajes que permiten crear nuevas formas de programación mucho más expresivas en su estilo estos se dividieron en una serie de generaciones que representan lenguajes de programación que se hicieron basados en tendencias en una época particular o específica y por ende llevan una similitud o características genéricas en común.

Lenguajes de alto y de bajo nivel

Los lenguajes de programación recaen también en dos categorías que tratan sobre su "nivel de abstracción". Con esto se evalúa si van de lo específico y concreto o de lo general en referencia a la arquitectura que presenta el sistema en el que se programa.

Los ordenadores y dispositivos solo entienden un lenguaje conocido como código binario o código máquina, el cual consiste en ceros y unos que representan pulsaciones electromagnéticas. Es decir, sólo utilizan ceros y unos al momento de codificar cualquier acción. Por esta razón se les denomina de bajo nivel.

Por su parte los lenguajes más cercanos a los programadores manejan funciones, métodos, semánticas e interfaces. Por ser más claros y entendibles se denominan lenguajes de alto nivel.

Clasificación por paradigmas

Los paradigmas en programación pueden distinguir varios patrones de cómputo y modelos de estilos al momento de estructurar y organizar las tareas a realizarse en un programa. Un lenguaje de programación suele ofrecer soporte para de uno a muchos paradigmas de programación y estos pueden darse de forma total o parcial.

Estos métodos sistemáticos son empelados en todos los pasos de la programación para solucionar problemas computacionales. Se dividen en: Imperativo (sucesión de comandos o grupos de sentencias concretas), declarativo (no requieren que se definan algoritmos pues es descrito el problema en lugar de encontrar una solución al mismo), reactiva (basado en escuchar lo que emite un evento o cambios en el flujo de datos) y orientada a objetos (estructuras que representan elementos del problema que se debe resolver, estos gozan de características y funciones propias).

Clasificación por propósito

Otra clasificación existente entre lenguajes de programación se encuentra en como basamos el propósito de este. Así a breves rasgos podemos identificarlos entre los que van de propósitos generales y aquellos de propósitos específicos.

En varias ocasiones a estos también se les clasifican en familias. Pues estos suelen compartir ciertas características en

común; al igual que pueden compartir el estilo de sintaxis usado. En general se da esta similitud en características pues muchos lenguajes heredan cosas de otros, que suelen ser anteriores a este y les sirvieron de inspiración o guía a los creadores en dicho lenguaje o lenguajes mejorados.

Capítulo 2

Lenguaje Action Script

Conceptos básicos de Action Script

"Adobe ActionScript (lanzado en 1997) es el lenguaje de programación de la plataforma o software Adobe Flash." (Wikimedia, 2021) Este fue ideado originalmente como una herramienta para que desarrolladores programen de un modo completamente interactivo. La programación con ActionScript busca obtener más eficiencia en Flash a la hora de crear animaciones de diversos tipos, tanto simples o complejas, diversas en análisis y envío de datos como para aplicar interfaces interactivas.

La versión establecida actualmente es ActionScript 3.0 en las últimas actualizaciones de Flash (Anímate actualmente), esta versión es una mejora para la creación de programas orientados a objetos al ajustarse al estándar ECMA-262.

ActionScript es empelada aparte de Adobe Flash, también en Adobe Anímate y Flex. Desde la versión 2 de Flex esta incluye la versión 3.0 de ActionScript, lo cual es una mejoría para su rendimiento comparado con sus versiones anteriores. Esto ayuda agregando nuevas funciones como usar expresiones regulares y la forma que se empaquetan o encapsulan las clases y sus métodos.

Características generales

• ActionScript es el lenguaje de programación exclusivo del software Flash. El ActionScript está basado en la especificación ECMA-262, lo que significa que se basa en un estándar de JavaScript que está pensado en garantizar la interoperabilidad de los elementos para web en diferentes navegadores. Además, que especifica la sintaxis y la semántica para ActionScript de matrices, función, etc.

• Como lo indica su nombre es un lenguaje de script. Lo que significa que no hace es necesario crear un programa entero para poder tener un ejecutable funcional. La aplicación de varios fragmentos o funciones de código en los elementos existentes en nuestras mesas de trabajos o proyectos permite alcanzar nuestras metas fácilmente.

• ActionScript se le considera desde un inicio como un lenguaje orientado a objetos. Y tiene semejanzas con lenguajes como son el Microsoft Visual Basic, el Borland Delphi, etc. Aunque no tenga en si la potencia que gozan otros lenguajes que son completamente orientados a objetos, en cada versión y actualización se acerca más a un lenguaje de este tipo. En resumen, la versión 2.0 es más potente y orientada a objetos que la versión 1.0.

• El ActionScript demasiadas coincidencias en semántica y sintaxis con Javascript pues de este último se basa ActionScript. Si se sabe de Javascript la sintaxis y el estilo de ActionScript resultan muy familiares y fáciles de entender.

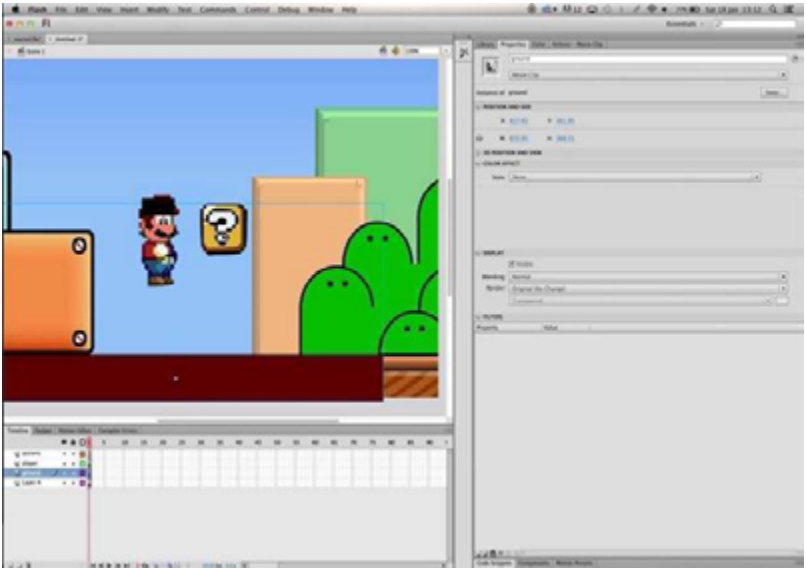
• En la mayor parte de situaciones no nos será necesario programar desde cero pues Flash pone a disposición

ante el usuario una librería de “funciones” ya adecuadas para realizar lo que estamos buscando, solo basta con colocarlas en las líneas del código adecuadas para poder probarlas y ejecutarlas.

En resumen, se puede decir que la estructura de ActionScript se base en propiedades de elementos aplicando diversos métodos y funciones orientado a estos objetos.

Figura 4

Proyectos simples o hasta complejos se pueden realizar con ActionScript



Nota: ActionScript se basa en propiedades de elementos aplicando diversos métodos y funciones orientado a estos objetos simples o complejos.

Sentencias – Sintaxis básica.

https://www.youtube.com/watch?v=IsoRjPkMAeo&ab_channel=JorgeLLanten

La sentencia básica tiene en su estructura el nombre del objeto, ya sea este un clip de película o botón, seguido de una acción o procedimiento. Esta acción puede estar declarada por reacciones de usuario interactuando por los periféricos de mouse o teclado. La manera de la estructura de sintaxis no cambia casi nunca, encontramos "function", y después un par de paréntesis seguidos por una llave que se abre y cierra. Dentro de estas llaves van las declaraciones.

Se escribe la acción correspondiente, condicionantes como:

if

else

else if

Switch

for.

Y para cerrar sólo se cierra la llave.

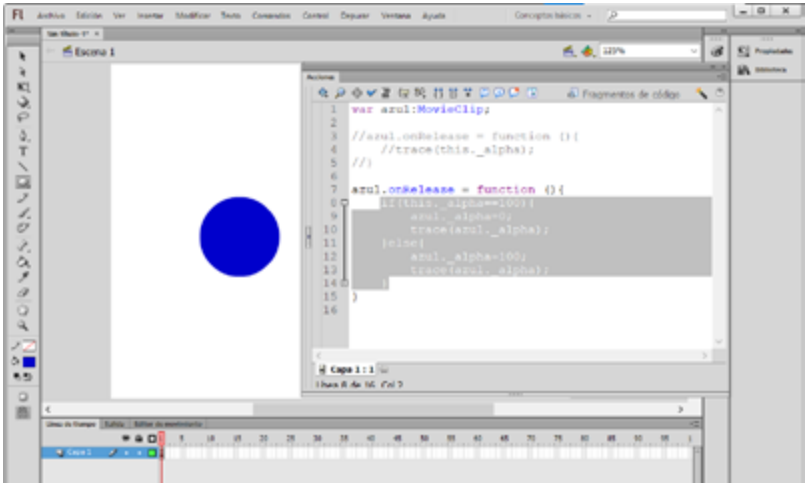
Por ejemplo, el elemento rojo seguido de un punto y la acción será igual a los procesos declarados en la *function ()*, entre llaves:

Código:

```
azul.onRelease = function () {
//trace(this._alpha) ;
}
azul.onRelease= function () {
  if (this._alpha==100);
      azul._alpha=0;
      trace(azul._alpha) ;
} else {
azul._alpha=100;
trace(azul._alpha) ;
}
}
```

Figura 5

Ejemplo de sintaxis en ActionScript



Capítulo 3

Nociones Básicas de Programación

Introducción: Organizando mi código en Actionscript

Para empezar a crear código y comandos en Flash es necesario saber que vamos a emplear las opciones de Acciones. La herramienta o ventana Acciones se encuentra en Ventana y luego Acciones o dando clic en la tecla F9 como atajo.

Debemos entonces situarnos sobre un fotograma de la línea de tiempo o un elemento grafico en la mesa del trabajo y luego abrir la ventana de acciones para poder escribir nuestros códigos.

Cuando el código lo colocamos a algún elemento en nuestra mesa de trabajo debemos darle clic a este primero y dar clic a F9, por ejemplo. A este tipo de organización se le conoce como código especifico o local pues afecta a botones o clips de película en particular en referencia al resto.

Por otro lado, tenemos los códigos generales, este tipo de código se emplean seleccionando el o los fotogramas donde queremos pasen las acciones. Se recomienda organizar la línea de tiempo para crear una capa de fotogramas en blancos destinados para los códigos y aparte las capas para elementos gráficos.

El empleo de estas dos técnicas siempre dependerá de lo que se busca crear. Se recomienda el uso de códigos locales

cuando queremos que ese elemento u objeto en Flash tenga atributos, variables, contadores, acumuladores, etc. Específicos que solo sirvan para este por ejemplo algún puntaje, vidas, característica mediante este ejecutándose. Por otro lado, los códigos generales podrían servir para establecer datos que afecten a todos los elementos en escena, por ejemplo.

Es importante saber que a los elementos tipo grafico no se les puede agregar códigos mediante la ventana de Acciones.

Manejo de condicionales múltiples en Actionscript

Cuando necesitamos tomar una decisión basada en una condición, usamos una condición if, pero a veces necesitamos verificar muchos datos diferentes para saber qué hacer.

Aquí es donde entran en juego algunas condiciones, en este consejo veremos cómo utilizarlas. Los condicionales son:

AND (&&)

OR (||)

NOT (!)

AND (&&)

Este operador ejecutará el código solo si ambas condiciones son verdaderas.

Código:

```
if((1 == 1) && (1 < 2)) //Si uno es igual a uno y uno es menor que dos
{
  trace("True"); //True
}
```

OR (||)

Ejecuta el código si alguna de las condiciones es verdadera.

Código:

```
if((1 == 4) || (1 < 2)) //Si uno es igual a cuatro o uno es menor que dos
{
  trace("True"); //True, la condicion de la derecha es verdadera
}
```

NOT (!)

Ejecuta el código si la condición es contraria.

Código:

```
var lloviendo:Boolean = false;
if(!(lloviendo == true))
{
  trace("No está lloviendo"); //Si no se hubiera colocado el
```

[b]![/b], no se ejecutaría el trace

}

Con estas condiciones implementadas, podemos estructurar mejor nuestro código y reducir la aparición de if innecesarios.

Condicionales – if

La sentencia *if* ayuda a controlar cuales sentencias de una regla o función se ejecutan mediante la evaluación del valor que declaremos como condicionante. Dicho valor debe ser una expresión para comprobar si es verdadera o falsa. La sentencia *if,* en el lenguaje de políticas de Impact es la misma que la empelada en lenguajes como C/C++ y Java.

Por ello la sintaxis de una sentencia *if* consta de la palabra primordial "if" seguida de una expresión booleana (verdadero o falso) o numérica con un respectivo rango encerrada entre paréntesis. Dicha expresión le sigue un bloque de sentencias o líneas de código delimitado o encerrado por llaves: {}.

Opcionalmente, a la sentencia *if* le puede proseguir las palabras claves "else" o "else if" que también deben tener un bloque de sentencias con llaves.

Código:

```
If (this._alpha == 100){
                rojo._alpha = 0;
```

```
            trace(rojo._alpha);
        } else {rojo._alpha = 100;
        trace(rojo._alpha);
}
```

"Caso contrario" – else

En programación se define a una sentencia condicional como la instrucción o grupo de pasos que se van a ejecutar o no en función al valor ya designado de una condicionante.

Entre los tipos más reconocidos de las sentencias condicionales hallamos *Si… Entonces…* (if... then), Si… *Entonces... SI NO* (if... then... else…) y *Según* (case o switch).

Se puede llamar al uso de este tipo de condiciones como alternativas más modernas para prevenir en nuestro código el "anidamiento" de varias sentencias condicionales.

Estas sentencias se conforman como los pilares en la programación planificada y estructurada. Estas en conjunto con los bucles.

Por ejemplo, tenemos la siguiente sentencia de códigos que indica que el objeto se desplazará hacia arriba siempre y cuando no supere 10 unidades en el eje Y:

Código:

```
arriba.onrelease=function(){
        if(objeto._y >= 10){
```

```
        objeto._y -= 5;
} else {
        objeto._x = 275;
        objeto._y = 200;
}

}
```

"En caso de" – ELSE IF

En este caso encontramos condiciones en cadena anidadas para limitar varios rangos y condicionantes en los cuales el objeto grafico puede verse afectado.

Código:

```
arriba.onRelease = function(){
        if(objeto._y >= 10){
        objeto._y -= 5;
}
else if (objeto._x >= 300){
        objeto._x = 275;
        objeto._y = 200;
}
else if(objeto._x <= 20){
        objeto._visible = false;
}

}
```

Test personalizado – Usando condicionales

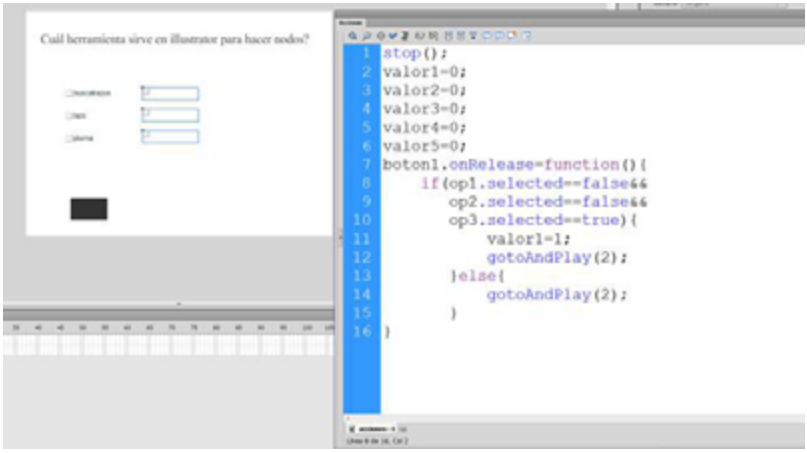
Un ejercicio de test es un proyecto sencillo, utilizando diferentes opciones de condicionales y elementos propios de flash, así los "CheckBox" y su forma de declarar verdad o falso.

```
top();
valor1=0;
valor2=0;
valor3=0;
valor4=0;
valor5=0;
boton1.onRelease=function(){
        if(op1.selected==false&&
          op2.selected==false&&
          op3.selected==true){
                valor1=1;
                gotoAndPlay(2);
          }else{
                gotoAndPlay(2);
          }
}
stop();
valor_total=valor1+valor2+valor3+valor4+valor5;
if(valor_total>=4){
        cuadro.loadMovie("img1.jpg");
```

```
}else if (valor_total>=3){
        cuadro.loadMovie("img2.jpg");
}else if (valor_total>=1){
        cuadro.loadMovie("img3.jpg");
}
```

Figura 6

Test usando diferentes checkbox



Nota: Aquí podemos observar código con diferentes opciones de condicionales y elementos: "CheckBox".

Suma de valores – Concepto de concatenar

Concatenación es el método usado para sumar valores dentro del código de programación que están definimos como texto o variables que albergan caracteres. En ActionScript 2.0

como en otros tantos lenguajes de programación deberemos llamar primero a la variable, o tomar la cadena de caracteres, la cual se le define como **String** o también, para convertirlo en número. Se utiliza para uno o más valores el signo + para concatenarlos o sumarlos. En este caso ejemplo se lo establece como *suma.onRelease = function()*, y su sentencia sería:

Código:

```
suma.onRelease=function(){
        numberresultado=Number(number1.text) +
Number(number2.text);
        }
```

Además, podríamos utilizar este modelo de sumar variables para aplicarle otras operaciones como resta, división y multiplicación.

SUMA:

```
resultado = 0;
suma.onRelease=function(){
        resultado=Number(caja1.text)+Number(caja2.
text)+Number(caja3.text);
}
```

RESTA

```
resultado=0;
resta.onRelease=function(){
        resultado=Number(caja1.text)-Number(caja2.text)-
Number(caja3.text);
```

MULTIPLICACIÓN:

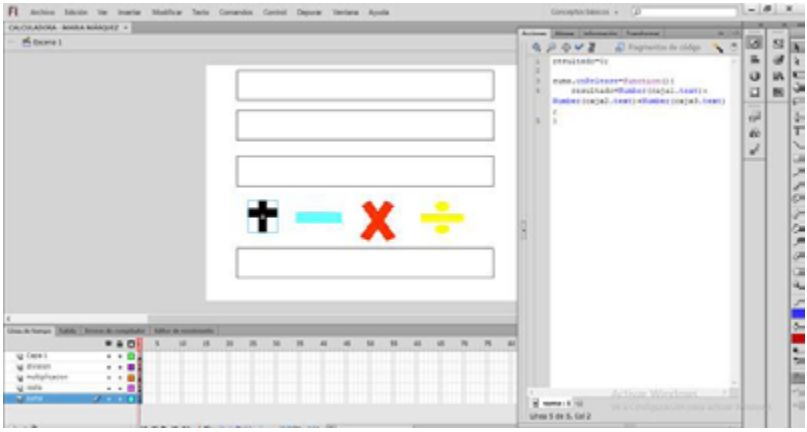
```
resultado=0;
multiplicacion.onRelease=function(){
        resultado=Number(caja1.text)*Number(caja2.
text)*Number(caja3.text);
}
```

DIVISIÓN:

```
resultado=0;
division.onRelease=function(){
        resultado=Number(caja1.text)/Number(caja2.text)/
Number(caja3.text);
}
```

Figura 7

Ejemplo de funciones de suma en ActionScript



Nota: Aquí vemos un ejercicio de calculadora.

Posición de objetos

Las propiedades de los objetos están determinadas para que siempre se detalle con la misma sintaxis:

Nombre del objeto, seguido de la barra inferior y la propiedad a ser afectada, esto permite saber la ubicación de un objeto, su transparencia y demás propiedades.

Por ejemplo, aquí tenemos valores que aplicaremos a los ejes de las X y Y del plano cartesiano.

Código:

```
arriba.onRelease=function(){
        objeto._y-=5;
}
abajo.onRelease=function(){
        objeto._y+=5;
}
derecha.onRelease=function(){
        objeto._x+=5;
}
izquierda.onRelease=function(){

        izquierda._x-=5;
```

Hay que recordar que en ActionScript utilizamos coordenadas del plano cartesiano. Estos formando Vectores de dirección y posicionamiento en R2 (ejes X y Y).

Esto es aplicable a toda programación básica para visualizar elementos. Teniendo en cuenta que el punto 0 del plano se encuentra en la esquina superior izquierda de la mesa de trabajo de Flash.

Acciones de controladores con condicionales

A estos controladores se lo puede combinar con condicionales para crear eventos específicos con los elementos en el escenario.

Código:

```
arriba.onRelease=function(){
        if(objeto._y>=10){
        objeto._y-=5;
}
}
abajo.onRelease=function(){
        if(objeto._y<=400){
        objeto._y+=5;
}
derecha.onRelease=function(){
        if(objeto._X<=550){
        objeto._x+=5;
}
}
```

```
izquierda.onRelease=function(){
        if(objeto._X>=10){
        izquierda._x-=5;
}
}
```

Capítulo 4

Interacción de Objetos en Flash con Action Script

Movimiento de objetos clip de película

Para mover un elemento por el escenario al pulsar un botón debemos hacerlo clip de película y emplear el siguiente código. Hay que recordar que para ello se utiliza la función. *onRelease* y anteponer el nombre del botón o clip de película que presionaremos. Dentro de esta función declararemos lo que queremos sea afectado, en este caso los pasos en el eje que el objeto o personaje queremos que se traslade. Recordar poner nombre a todos los clips de película o botones según corresponda como arriba, abajo, derecha, izquierda, centro (botones que pulsaremos para que se mueva el objeto).

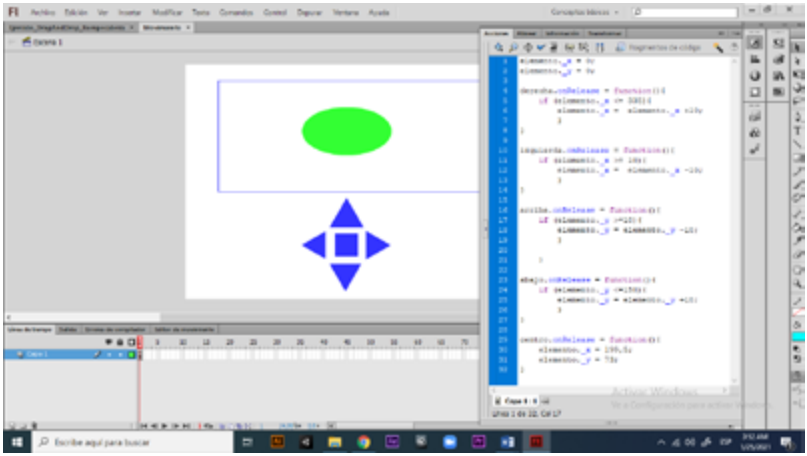
Código:

```
elemento._x = 0;
elemento._y = 0;
derecha.onRelease = function(){
        if (elemento._x <= 335){
                elemento._x =  elemento._x +10;
                }
}
izquierda.onRelease = function(){
        if (elemento._x >= 10){
                elemento._x =  elemento._x -10;
                }
}
```

```
arriba.onRelease = function(){
        if (elemento._y >=10){
                elemento._y = elemento._y -10;
                }
        }
abajo.onRelease = function(){
        if (elemento._y <=150){
                elemento._y = elemento._y +10;
                }
}
centro.onRelease = function(){
        elemento._x = 199,5;
        elemento._y = 73; }
```

Figura 8

Ejemplo de movimiento de elementos en ActionScript 2



Nota: Dentro de la función (onRelease), declaramos el

nombre del clip de película que vaya a ser afectado por dicha función, en este caso los pasos en el eje que el objeto o personaje queremos que se traslade

Escala de objetos

Para rotar objetos utilizamos el mismo procedimiento usado al mover los objetos. De igual manera creamos nuestros clips de película y botones con los nombres correspondientes: *elemento, más y menos*. Y la instrucción o controlador a usar sobre este elemento sería: ._*xscale* o ._*yscale*.

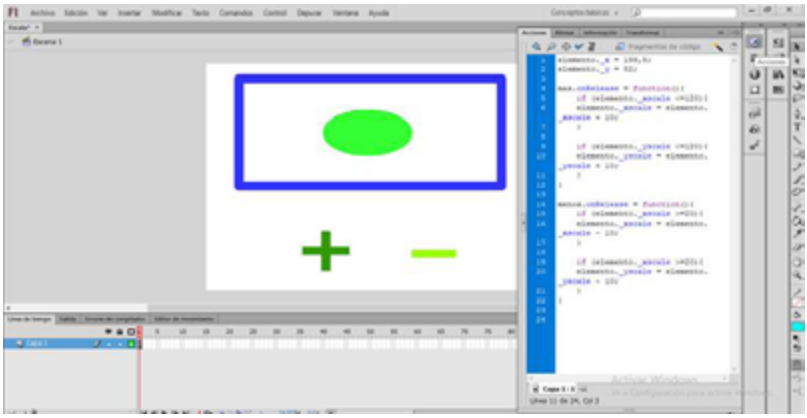
Código:

```
elemento._x = 199,5;
elemento._y = 82;
mas.onRelease = function(){
        if (elemento._xscale <=120){
        elemento._xscale = elemento._xscale + 10;
        }
        if (elemento._yscale <=120){
        elemento._yscale = elemento._yscale + 10;
        }
}
menos.onRelease = function(){
        if (elemento._xscale >=20){
        elemento._xscale = elemento._xscale - 10;
        }
```

```
        if (elemento._xscale >=20){
        elemento._yscale = elemento._yscale - 10;
        }
}
```

Figura 9

Ejemplo de escala de elementos en ActionScript 2



Nota: Aquí modificaremos la instrucción o controlador a usar sobre este elemento (_xscale – _yscale).

Transparencia de objetos

Para cambiar la transparencia de clips de película u objetos en Flash se emplea el controlador. ***_alpha.*** Para ello repetimos el proceso de crear el clip de película para el elemento que deseamos darle transparencia y botones que sumen o resten la transparencia en sus funciones hacia más

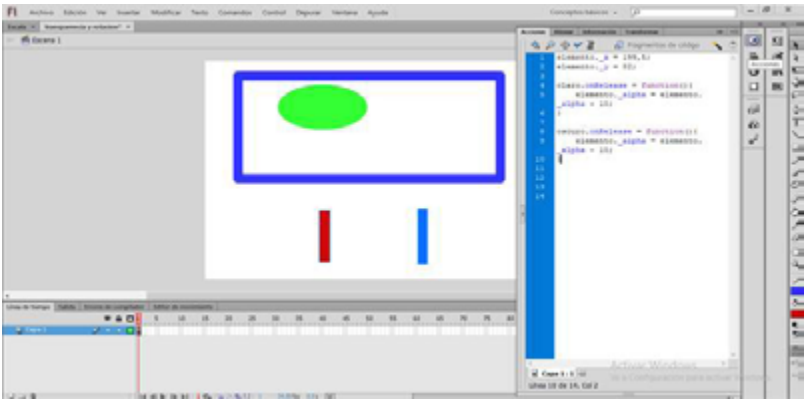
claro o transparente a más oscuro o sólido. También podemos agregarle condicionantes con un ***if*** dentro de la función si queremos para que cuando llegue al valor 0 o 100 no sume más valores, lo que permite ahorrar clics al volver a otro estado.

Código:

```
elemento._x = 199,5;
elemento._y = 82;
claro.onRelease = function(){
        elemento._alpha = elemento._alpha + 10;
}
oscuro.onRelease = function(){
        elemento._alpha = elemento._alpha - 10;
}
```

Figura 10

Ejemplo de transparencia de elementos en ActionScript 2



Nota: Aquí modificaremos la instrucción o controlador a usar

sobre este elemento (_alpha).

Rotación de objetos

Para este caso emplearemos la instrucción. ***_rotation*** para agregarle valores que indican los grados en los que va a ir rotando siempre que aplastemos el botón. En este caso el botón amarillo es el encargado de hacer rotar el elemento en la parte superior si le damos clic. Para ello hacemos al elemento principal clip de película y al otro botón con clic derecho y convertir en símbolo. Elegir la opción de la ventana emergente el **tipo** sea botón o clip de película en el menú desplegable y aceptar. Con esto podemos dar nombre a estos objetos desde la ventana de propiedades en el primer recuadro que se llama nombre de instancia. Luego de eso llamar a los elementos de la misma manera se llaman en el código. Por ejemplo, elemento para el personaje y rotar para el botón.

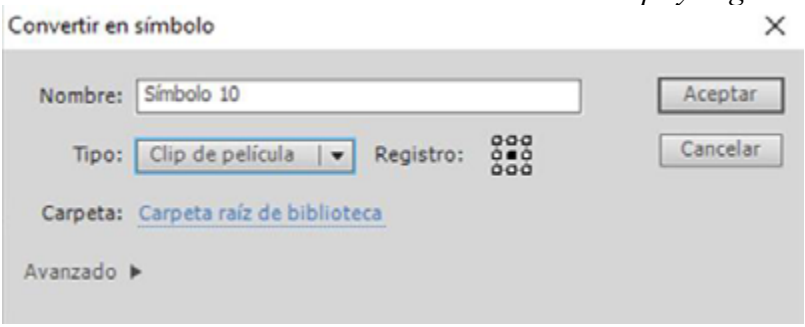
Código:

```
elemento._x = 199,5;
elemento._y = 82;
rotar.onRelease = function(){
        elemento._rotation = elemento._rotation - 10;
}
```

Nota: Para hacer rotar desde el centro recordar seleccionar el *pivot* o punto de referencia al centro cuando creemos el botón o clip de película en la ventana emergente de convertir en símbolo debemos en el registro seleccionar el botón central de la matriz mostrada.

Figura 11

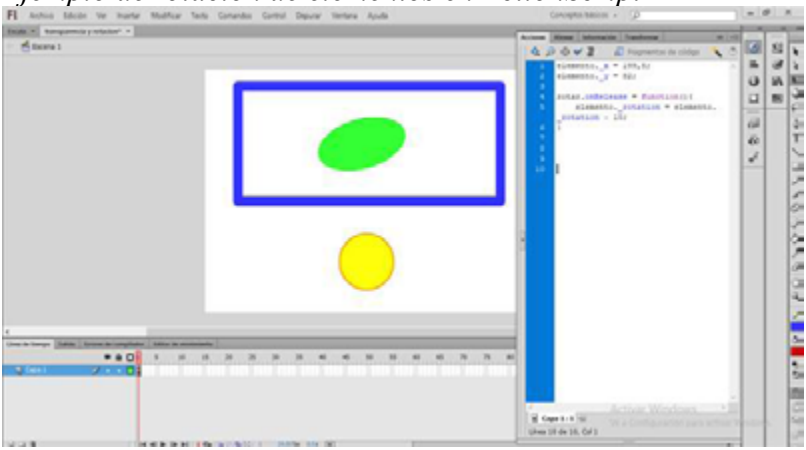
Ventana Convertir en Símbolo con la selección de Tipo y Registro



Nota: Aquí se da el nombre del Símbolo y se elige un tipo de Símbolo.

Figura 12

Ejemplo de rotación de elementos en ActionScript 2



Nota: Aquí modificaremos la instrucción o controlador a usar sobre este elemento (_rotation).

Arrastre de objeto – start drag / stop drag

Para el uso de arrastres de elementos gráficos creados en Flash se utilizan las funciones onPress y onRelease. El primero para mantener sostenido el cursor y elemento gráfico; usamos la segunda función para detener el drag o arrastre en el punto cuando soltemos el botón del mouse. Recordar que para hacerlo funcionar nuestro elemento grafico debe ser convertido antes a clip de película y sus nombre de instancia debe ser el mismo que acompañe a la función por ejemplo *pieza.onPress = function(){}...* donde pieza es el nombre del clip de película.

Código:

```
pieza.onPress = function(){
        this.startDrag();
}
pieza.onRelease = function(){
        this.stopDrag();

        }}
```

Colisión entre objetos – hit test

Las colisiones es una forma de detectar cuando dos objetos o clip de películas chocan y ocupan el mismo sitio. Hoy usamos bounding boxes o límites de elementos, fácil para empezar a hablar de físicas.

Para este ejemplo partiremos del drag and drop y

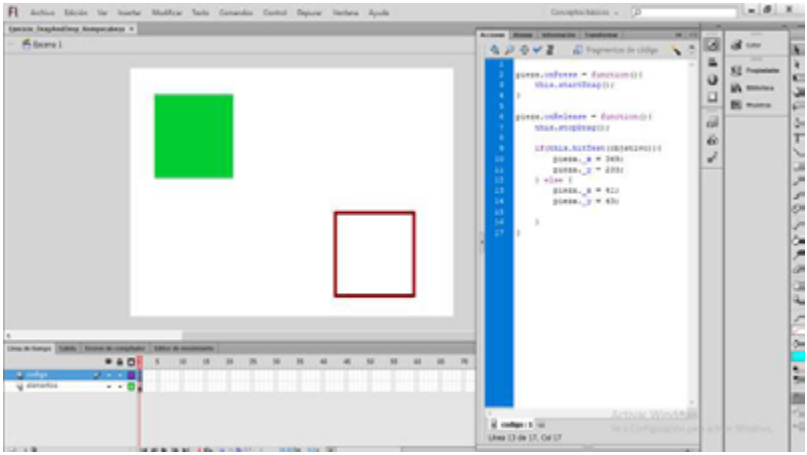
agregaremos una condicionante al momento de tocar o colocar un objeto sobre otro. Para ello empleamos la instrucción HitTest. (*"nombre de instancia del objeto con que colisionamos");* debemos recordar siempre que entre paréntesis debe ir el nombre del objeto B con el cual nuestro objeto A (el que arrastramos) va a colisionar:

Código:

```
pieza.onPress = function(){
        this.startDrag();
}
pieza.onRelease = function(){
        this.stopDrag();
        //Aquí empieza segmento del código de colisión de objetos o HitTest
        if(this.hitTest(objetivo)){
                pieza._x = 349;
                pieza._y = 233;
        } else {
                pieza._x = 41;
                pieza._y = 43;
                }
}
```

Figura 13

Ejemplo de colisión de elementos en ActionScript 2



Nota: Empleamos la instrucción HitTest.(“nombre de instancia del objeto con que colisionamos”);

Instancias de botones

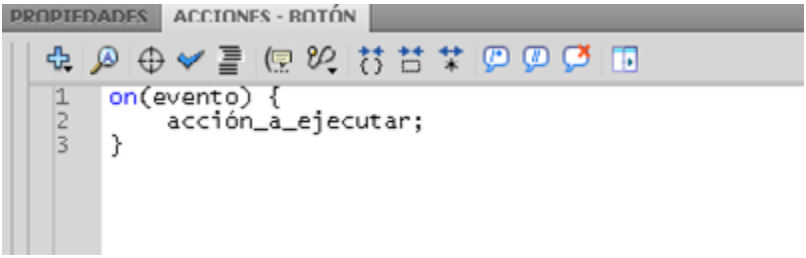
Botones en Flash: Actionscript 2.0

“La estructura básica para programar con as2 es relativamente sencilla. Podemos programar sobre el botón o sobre un fotograma en blanco utilizando nombres de instancia.” (Jimena, 2011)

Se escribe el código sobre el botón, la siguiente estructura

Figura 14

Estructura de código Actionscript 2.0 sobre el botón.



Nota: Aquí observamos la acción que se lleva a cabo sobre un botón.

Evento:

• release: Al soltar el botón del ratón mientras el puntero se encuentra sobre la instancia de botón.

• rollOver: Se produce al desplazar el puntero del ratón sobre el botón.

• rollOut: Se produce al desplazar el puntero del ratón fuera del botón.

• press: Al mantener presionado el botón.

• acción_a_ejecutar: En otras palabras, qué sucede si ocurre lo que se especifica en el evento. Por ejemplo:

• getURL ("url_completa", "_blank"): Redirigir a un sitio web o archivo HTML en su computadora.

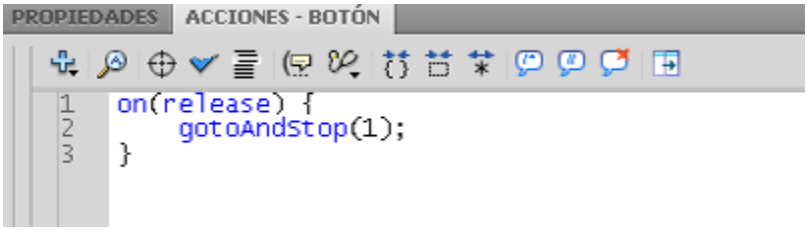
• gotoAndStop ("nombre_de_escena", fotograma):

Dirige a un fotograma de la línea de tiempo.

• loadMovie ("url_completa_del_video"): Carga una película swf o flv.

Figura 15

Ejemplo de evento Release



Nota: Nos lleva a un fotograma específico dentro de la línea de tiempo.

Cuando presionamos el botón, nos llevara automáticamente al fotograma 1.

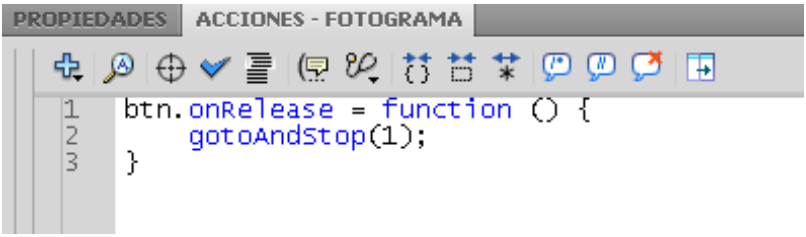
Al tener un clip de película con una animación dentro, y le damos un nombre de instancia a este clip ("película"). Si queremos que la animación en el clip vaya al fotograma 1, debemos agregar el nombre de la versión del clip delante de "gotoAndStop", por lo que la línea de código se verá así:

pelicula.gotoAndStop(1);

Para escribir el código en el fotograma, debemos cambiar la estructura. Primero que nada, hay que ponerle un nombre de instancia al botón en cuestión, por ejemplo "btn".

Figura 16

Ejemplo de una estructura con evento onRelease



Nota: Aquí la animación dentro del clip se va dirigir al fotograma 1.

Capítulo 5

Visualización y Exportación de Proyectos con Action Script

fscommand – Todas sus funciones

Al trabajar en Flash nos importa que la película se muestre a pantalla completa, bien porque sea más fácil o porque se ve mejor. La programación varía según la versión de ActionScript que utilicemos.

Si estamos utilizando ActionScript 2, debemos utilizar la función "fscommand" con el comando "fullscreen" y el parámetro "true". Si queremos que el proyector o visor vea la pantalla completa cuando abrimos la película o proyecto editado, debemos ir al primer fotograma del archivo flash y escribir lo siguiente.

Código:

```
on (release) {
	fscommand("exec","file.bat");
}
on (release) {
	fscommand("fullscreen","true");
}
on (release) {
	fscommand("quit");

}
```

Combinación de fscommand y condicionales

Aparte también podemos combinar la función de fullscreen con condicionales. Por ejemplo:

Código:

```
on (release) {
        if(nombre_de_variable==5){
        fscommand("exec","file.bat");
   }
}
```

Importar texto externo al proyecto

Podemos importar texto externo guardado en un bloc de notas al proyecto a través del loadVariablesNum.

Es importante saber que para poder editar un texto a través de código estos deben ser tipo dinámico. Los de tipo estático solo sirven para estar presentes como decoración, pero no para funcionar como variables.

Código:

```
stop();
loadVariablesNum ("archivo.txt", 0);
```

Acción constante – onEnterFrame

OnEnterFrame nos ayuda a leer acción de los clips de películas o frames internos de la animación, por ejemplo:

Código:

```
stop();
var velocidad:Number = 15;
        avanzar_btn. onPress = function () {
        _root .onEnterFrame = function () {
                if (scroll_img ._x > -1018) {
                        scroll_img ._x -= velocidad;
                }
        }
}
avanzar_btn. onRelease = function () {
        delete _root .onEnterFrame;
}
retroceder_btn. onPress = function () {
        _root .onEnterFrame = function () {
                if (scroll_img ._x < 2046) {
                        scroll_img ._x += velocidad;
                }
}
}
retroceder_btn. onRelease = function () {
        delete _root.onEnterFrame;
}
```

Creación de proyecto ejecutable - .EXE

Básicamente, un archivo con extensión ".exe" (del inglés executable), es un formato de archivo informático que contiene una serie de instrucciones para ejecutar el software.

Originalmente introducidos en 1983 con el sistema operativo MS-DOS, dicho tipo de archivos contienen programas que empleamos para ejecutar aplicaciones en los sistemas, incluyendo instaladores de otros programas, aplicativos como Excel, Word y muchos más.

¿Qué es un archivo Exe?

Existen varios tipos de archivos ejecutables tales como BAT, COM o BIN; EXE es uno de ellos y estos tipos de archivos son los que actualmente tienen mayor preponderancia y auge debido fundamentalmente a sus características, extensión y compatibilidad.

Actualmente para nuestros ordenadores se llega a encontrar dos tipos de archivos "exe": Los de 32 Bits y de 64 Bits. Ambos fueron diseñados específicamente para ser compatible con dicha arquitectura para sistemas operativos.

Una de las mayores características de estos archivos ejecutables es que no se puede leer o editar de parte del usuario final, ya que estos pasan por el proceso informático conocido como compilación. El cual es básicamente hacer de este código fuente que lleva el programa a una conversión de código

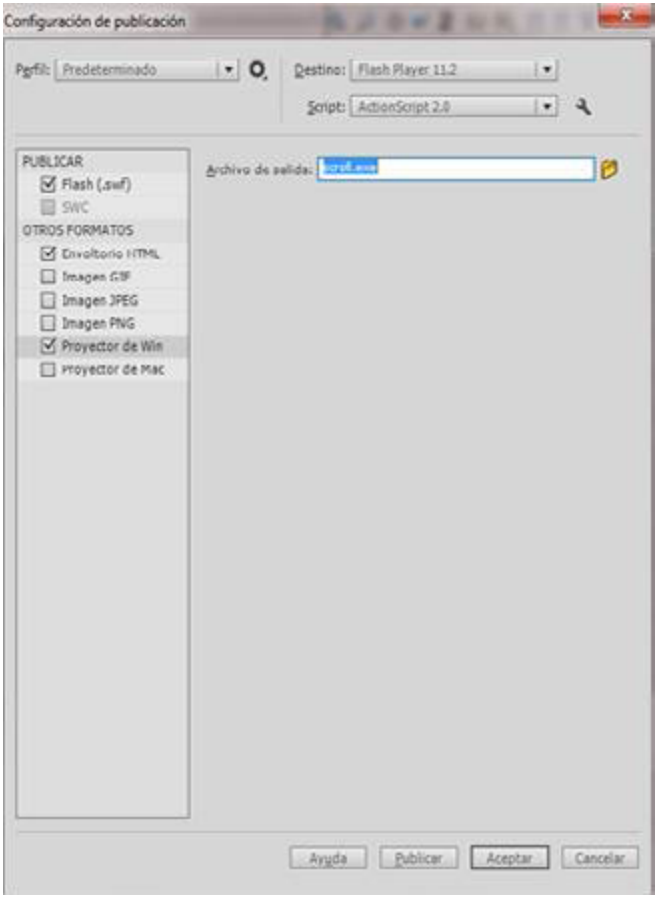
entendible por el procesador.

Por otra parte, el acto de editar un archivo .EXE mediante herramientas tecnológicas de manera inversa es considerado como algo ilegal en la mayoría de las legislaciones a nivel mundial. Estos archivos están protegidos por leyes de derechos de autor.

Sin embargo, no pasa de la misma forma con archivos en el ámbito del código libre u Open Source. Pues estos ejecutables no mantienen este tipo de restricciones legales. Además de ser softwares o programas que dan al usuario el código fuente de este para editar, lo que libera a este de utilizar técnicas de hacking o alteración de código para modificar funciones finales de dicha aplicación o programa.

Figura 17

Ventana de Configuración de Aplicación .exe



Nota: En el software de Adobe flash lo hallamos en Archivo y luego Configuración de Publicación.

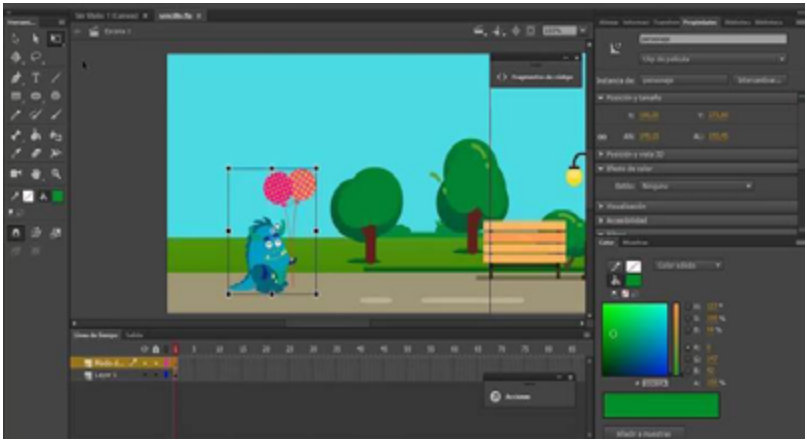
Cuento interactivo o videojuegos educativos

Los proyectos multimedia con fines educativos y su uso en el ámbito educativo destacan por sus importantes características interactivas y visuales que ayudan a un producto de calidad, que responda a las necesidades e intereses de usuarios objetivos.

Se debe recalcar lo importante de conocer sobre crear dichos proyectos multimedia con fines pedagógicos. Puesto que, en lo que refiere a la educación se han presentado cambios significativos gracias al empleo de las tecnologías de la información y comunicación en actividades formativas, educativas y lúdicas en conjunto.

Figura 18

Ejemplo de proyecto Cuento o Videojuego con gráficas animadas



Nota: Proyecto multimedia con fines educativos con características interactivas y visuales.

Capítulo 6

Carga de Datos Externos con Action Script

Carga de textos externos

"LoadMovie es obviamente más fácil de usar, pero no proporciona ninguna información sobre el estado de carga, por lo que, si queremos realizar una precarga, debemos usar la instancia de MovieClipLoader. Este ejemplo muestra cómo utilizar este tipo de objeto y sus eventos asociados." (ESDEERRE, 2023)

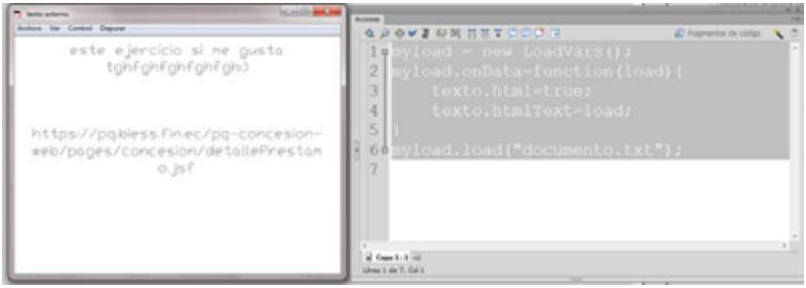
No es muy diferente cuando se trata de textos, presentan una forma sencilla de trabajar

Código:

```
.myload = new LoadVars();
myload.onData=function(load){
texto.html=true;
texto.htmlText=load;
}
myload.load("documento.txt");
```

Figura 19

Carga de texto externo acelera a un proyecto general



Nota: Aquí en este ejemplo se muestra cómo utilizar este tipo de objeto y sus eventos asociados.

Carga de imágenes externas

"Esta clase le permite implementar devoluciones de llamada de escucha que proporcionan información de estado mientras se cargan archivos SWF, JPEG, GIF y PNG en clips de película. Para usar las características de MovieClipLoader, use MovieClipLoader.loadClip () en lugar de loadMovie () o MovieClip.loadMovie () para cargar archivos SWF." (Freddie, 2004)

Código:

```
var swfLoader:Loader = new Loader();
var swfFile:URLRequest = new URLRequest("file.swf");
var container:MovieClip= new MovieClip();
swfLoader.contentLoaderInfo.addEventListener(Event.
```

```
COMPLETE,
swfLoadedHandler);
var currentSWF:MovieClip = new MovieClip();
swfLoader.load(swfFile);
container.addChild(swfLoader);
addChild(container);
function swfLoadedHandler(e:Event):void {
 currentSWF = MovieClip(swfLoader.content);
currentSWF.addEventListener(Event.ENTER_FRAME,
checkLastFrame);
 function checkLastFrame(e:Event):void {
 if (currentSWF.currentFrame == currentSWF.totalFrames) {
 currentSWF.stop();
 bob.play();
 if (bob.currentFrame == 2) {
 bob.stop();
        }
      }
   }
}
```

Carga de archivos externos XML

"El tipo de archivo XML, creado como formato de datos de texto, no sólo es legible a nivel de usuario sino también comprensible a nivel de máquina y, por lo tanto, es cómodo de usar en servicios web. XML es un lenguaje multiplataforma diseñado para almacenar diferentes tipos de datos." (Jennifer, 2022) Destaca por su simplicidad, facilidad de uso y generalidad, es tan popular como HTML y se distribuye ampliamente en Internet.

La notoriedad de este formato se debe en gran medida al hecho de que los archivos XML se pueden modificar con un editor de texto.

Detalles técnicos sobre los archivos XML

Un documento XML contiene una cadena de caracteres Unicode, cada uno de los cuales puede residir en un documento individual. La codificación Unicode estándar forma un documento XML dividido en etiquetas y contenido según reglas de sintaxis simples. Tiene una ventaja sobre los documentos HTML porque permite agregar etiquetas personalizadas a los objetos para identificar y diferenciar los datos que contienen.

Código:

```
var carga:XML=new XML();
carga.load("datos.xml");
carga.ignoreWhite=true
```

```
carga.onLoad=function(){
trace(carga.firstChild.childNodes[0].firstChild);
trace(carga.firstChild.childNodes[1].firstChild);
trace(carga.firstChild.childNodes[2].firstChild);
trace(carga.firstChild.childNodes[3].firstChild);
}
```

Comunicación con scripts externos

"Además de cargar archivos de datos externos, la clase URLVariables se puede utilizar para enviar variables al script del servidor y procesar la respuesta del servidor. Esto es útil, por ejemplo, si está programando un juego y desea enviar la puntuación del jugador al servidor para calcular si desea agregarlo a la lista de puntuación más alta o enviar la información de inicio de sesión del usuario al servidor, sujeto a autenticación o no. El script del servidor puede procesar el nombre y la contraseña del usuario y devolver la confirmación de que las credenciales proporcionadas por el usuario son válidas." (Platform, 2023)

El siguiente código crea un objeto URLVariables llamado variable, que crea una nueva variable llamada nombre. Ahora, se crea un objeto URLRequest para enviar las variables y detallar la URL del script del servidor. A continuación, establezca la propiedad del método del objeto URLRequest para enviar las variables como una solicitud HTTP POST. "Para añadir un objeto URLVariables a una solicitud de URL, creamos la propiedad de datos del objeto URLRequest en

el objeto URLVariables creado anteriormente.” (Platform, 2023). Al Final iniciamos la solicitud, creando la instancia de URLLoader y se llama al método URLLoader.load().

Código:

```
var variables:URLVariables = new
URLVariables(“name=Julio”);

var request:URLRequest = new URLRequest();

request.url = “http://www.[tudominio].com/greeting.cfm”;

request.method = URLRequestMethod.POST;

request.data = variables;

var loader:URLLoader = new URLLoader();

loader.dataFormat = URLLoaderDataFormat.VARIABLES;

loader.addEventListener(Event.COMPLETE,
completeHandler);

try
{
   loader.load(request);
}
catch (error:Error)
{
   trace(“No se puede cargar la URL”);
}
 function completeHandler(event:Event):void
{
   trace(event.target.data.welcomeMessage);
}
```

Capítulo 7

Propiedades de Objetos

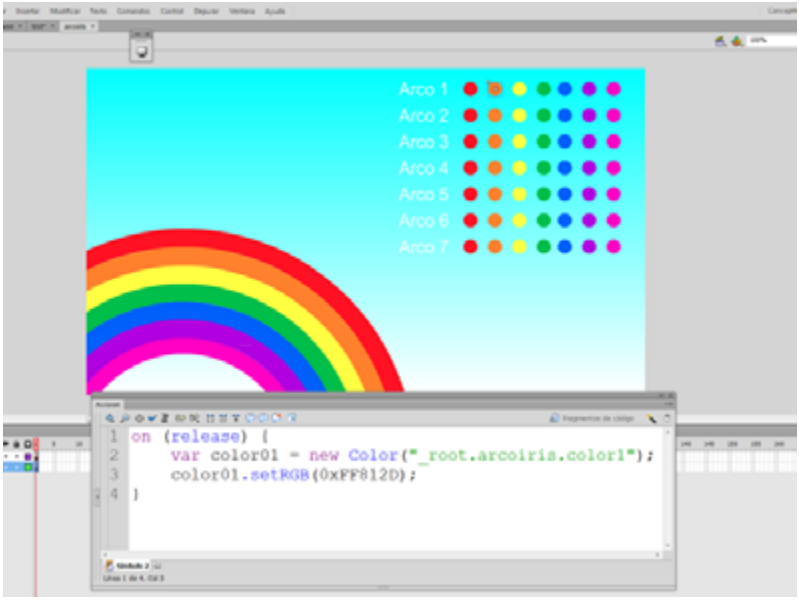
Valor set (color)

El método setRGB () asigna nuevas compensaciones de transformación para los componentes RGB de un clip de película. El nuevo desplazamiento se especifica más fácilmente como un número hexadecimal de seis dígitos de la forma 0xRRGGBB, donde RR, GG y BB son números de dos dígitos entre 00 y FF que representan los componentes rojo, verde y azul. Por ejemplo, el triplete RGB (51, 51, 102) es equivalente al valor hexadecimal.

```
on (release) {
    var color01 = new Color("_root.arcoiris.color1");

    color01.setRGB(0xFF812D); }
```

Figura 20

Ejercicio de cambio de colores del arcoíris



Nota: El método setRGB () asigna nuevas compensaciones de transformación para los componentes RGB de un clip de película.

Capítulo 8

Carga de Variables Externos

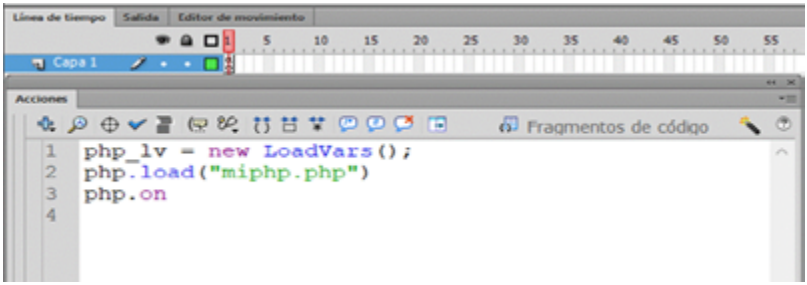
Load Vars

La clase LoadVars le permite enviar todas las variables en un objeto a una URL especificada y le permite cargar todas las variables en una URL especificada en un objeto. También le permite enviar variables específicas, en lugar de todas las variables, lo que puede hacer que su aplicación sea más eficiente. Puede usar el controlador LoadVars.onLoad () para asegurarse de que su aplicación se ejecute cuando se carguen los datos, y no antes.

La clase LoadVars funciona de manera muy similar a la clase XML; utiliza los métodos load (), send () y sendAndLoad () para comunicarse con un servidor. La principal diferencia entre la clase LoadVars y la clase XML es que LoadVars transfiere pares de nombre-valor de ActionScript, en lugar de un árbol de Modelo de Objeto de Documento XML (DOM) almacenado en el objeto XML. (Matemática, 2023)

Figura 21

LoadVars en este ejemplo está leyendo un archivo php.



Nota: La clase LoadVars sigue las mismas restricciones de seguridad que la clase XML.

LoadVars.send()

Función pública *send (url:String, target:String, [method:String]):Booleano*

Envía las variables en el objeto my_lv a la URL especificada. Todas las variables enumerables en my_lv se concatenan en una cadena en el formato application/x-www-form-urlencoded de forma predeterminada, y la cadena se publica en la URL mediante el método HTTP POST. Este es el mismo formato utilizado por loadVariables(). El tipo de contenido MIME enviado en los encabezados de solicitud HTTP es el valor de my_lv.contentType o el valor predeterminado application/x-www-form-urlencoded. Se utiliza el método POST a menos que se especifique GET.

Debe especificar el parámetro de destino para garantizar que se ejecutará el script o la aplicación en la URL especificada. Si omite el parámetro de destino, la función devolverá verdadero, pero el script o la aplicación no se ejecutará.

El método send() es útil si desea que la respuesta del servidor sea:

Reemplace el contenido SWF (use "_self" como parámetro de destino);

Aparecer en una nueva ventana (use "_blank" como parámetro de destino);

Aparecer en el marco principal o de nivel superior (use "_parent" o "_top" como parámetro de destino);

Aparece en un marco con nombre (use el nombre del marco como una cadena para el parámetro de destino).

Una llamada exitosa al método send() siempre abrirá una nueva ventana del navegador o reemplazará el contenido de una ventana o marco existente. Si prefiere enviar información a un servidor y continuar reproduciendo su archivo SWF sin abrir una nueva ventana o reemplazar el contenido en una ventana o marco, entonces debe usar *LoadVars.sendAndLoad()*.

Este método es similar a XML.send().

El entorno de prueba Flash siempre utiliza el método GET. Para probar usando el método POST, asegúrese de intentar usarlo desde un navegador.

El uso de este método no está permitido si el archivo SWF que realiza la llamada se encuentra en un entorno limitado local que no es de confianza.

No puede conectarse a puertos reservados comúnmente. Para obtener una lista completa de puertos bloqueados, consulte la entrada system.Security.loadPolicyFile().

En Flash Player 10 y versiones posteriores, si utiliza un tipo de contenido de varias partes (por ejemplo, “datos de formulario/multiparte”) que contiene una carga (indicada por un parámetro “nombre de archivo” en un encabezado “disposición de contenido” dentro del cuerpo de la POST), la operación POST está sujeta a las reglas de seguridad aplicadas a las cargas:

La operación POST debe realizarse en respuesta a una acción iniciada por el usuario, como un clic del mouse o presionar una tecla.

Si la operación POST es entre dominios (el destino POST no está en el mismo servidor que el archivo SWF que envía la solicitud POST), el servidor de destino debe proporcionar un archivo de política URL que permita el acceso entre dominios.

Además, para cualquier tipo de contenido de varias partes, la sintaxis debe ser válida (según el estándar RFC2046). Si la sintaxis parece no ser válida, la operación POST está sujeta a las reglas de seguridad aplicadas a las cargas.

Para obtener más información relacionada con la seguridad, consulte lo siguiente:

"Comprensión de la seguridad" en Aprendizaje de ActionScript 2.0 en Flash

Tema del Centro de desarrolladores de Flash Player: Seguridad

Parámetros:

url:String — Un string; la URL a la que cargamos las variables.

target:String — Un string; la ventana o marco del navegador en el que aparecerá cualquier respuesta. Puede ingresar el nombre de una ventana específica o seleccionar entre los siguientes nombres de destino reservados:

"_self" especifica el fotograma actual en la ventana actual.

"_blank" especifica una nueva ventana.

"_parent" especifica el padre del fotograma actual.

"_top" especifica el marco de nivel superior en la ventana actual.

method:String [opcional] — Una cadena; el método GET o POST del protocolo HTTP. El valor predeterminado es POST.

Retorno:

Booleano: un booleano value;false si no se especifica ningún parámetro value;true en caso contrario.

LoadVars.sendAndLoad()

Función pública sendAndLoad(url:String, target:Object, [method:String]):Boolean

Variables públicas en el objeto my_lv en la URL especificada. La respuesta del servidor se descarga, se analiza como datos variables y las variables resultantes se colocan en el objeto de destino.

Las variables se publican de la misma manera que LoadVars.send(). Las variables se descargan en el destino de la misma manera que LoadVars.load().

Al utilizar este método, considere el modelo de seguridad de Flash Player:

No se permite la carga de datos si el archivo SWF que realiza la llamada se encuentra en el entorno limitado local con sistema de archivos y el recurso de destino proviene de un entorno limitado de red.

Tampoco se permite la carga de datos si el archivo SWF que realiza la llamada proviene de un entorno limitado de red y el recurso de destino es local.

La carga de datos desde un dominio diferente requiere un archivo de política en el recurso de destino.

No puede conectarse a puertos reservados comúnmente. Para obtener una lista completa de puertos bloqueados, consulte la entrada system.Security.loadPolicyFile().

En Flash Player 10 y versiones posteriores, si utiliza un tipo de contenido de varias partes (por ejemplo, “datos de formulario/multiparte”) que contiene una carga (indicada por un parámetro “nombre de archivo” en un encabezado “disposición de contenido” dentro del cuerpo de la POST), la operación POST está sujeta a las reglas de seguridad aplicadas a las cargas:

La operación POST debe realizarse en respuesta a una acción iniciada por el usuario, como un clic del mouse o presionar una tecla.

Si la operación POST es entre dominios (el destino POST no está en el mismo servidor que el archivo SWF que envía la solicitud POST), el servidor de destino debe proporcionar un archivo de política URL que permita el acceso entre dominios.

Además, para cualquier tipo de contenido de varias partes, la sintaxis debe ser válida (según el estándar RFC2046). Si la sintaxis parece no ser válida, la operación POST está sujeta a las reglas de seguridad aplicadas a las cargas.

Para obtener más información relacionada con la seguridad, consulte lo siguiente:

“Comprensión de la seguridad” en Aprendizaje de ActionScript 2.0 en Flash

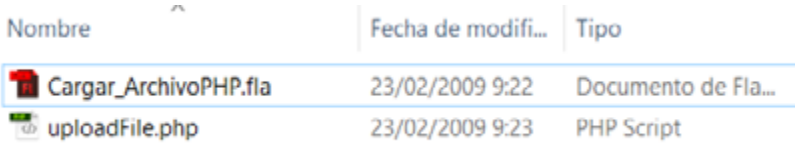
Tema del Centro de desarrolladores de Flash Player: Seguridad

Cargar variables php

"La clase File Reference de AS3 es bastante parecida a la de AS2. Además el ejemplo de la ayuda de Flash es bastante sencillo de entender. El código que pongo a continuación este hecho a partir de dicho ejemplo, al que he añadido algunos comentarios para explicarlo." (ESDEERRE, 2023)

Figura 22

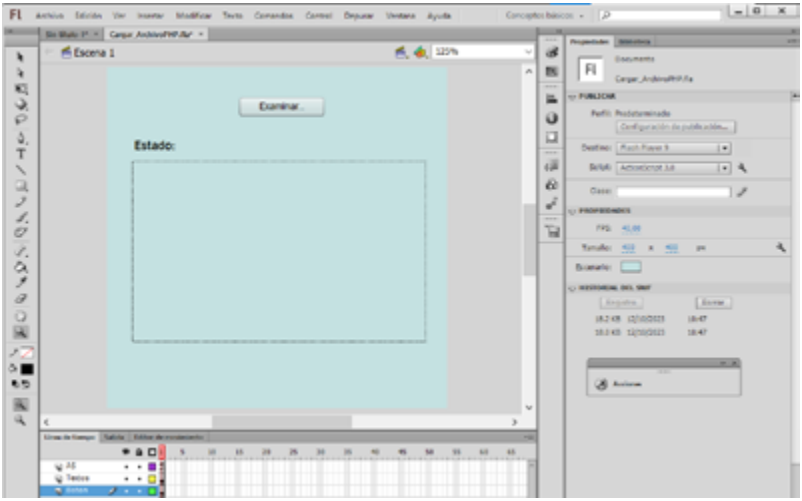
Archivo de llamado y carga de Código PHP



Nota: Aquí se muestra el archivo PHP que cargaremos a nuestro archivo de Actionscript.

Figura 23

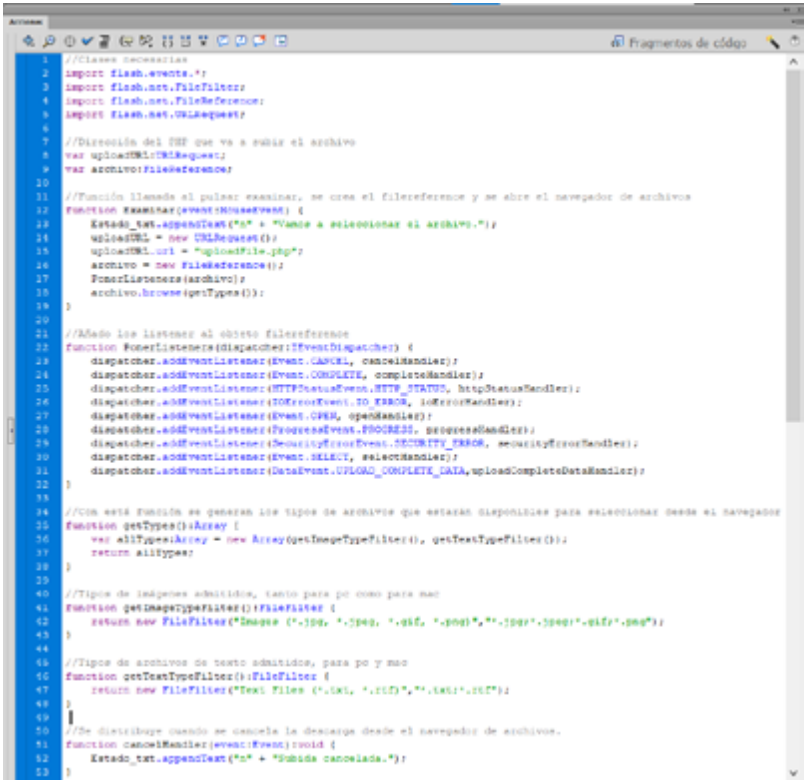
Escenario con un botón y un campo de texto dinámico



Nota: Escenario con un botón (Examinar_bt) y un texto dinámico (Estado_txt).

Figura 24

Llamado desde Actionscript a un archivo PHP.

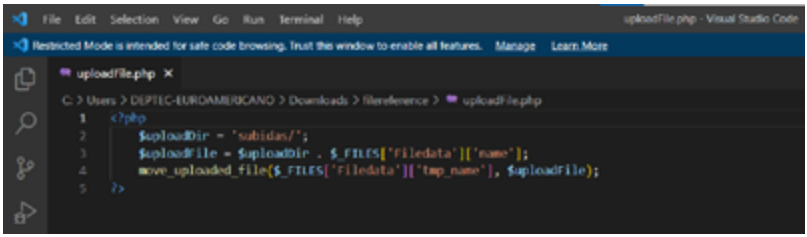


```
//Se distribuye cuando se finaliza la descarga o la carga
function completeHandler(event:Event):void {
    Estado_txt.appendText("\n" + "Subida completada.");
}

//Se reciben datos del servidor tras completar la carga
function uploadCompleteDataHandler(event:Event):void {
    Estado_txt.appendText("\n" + "Subida confirmada por el servidor.");
}

//Se produce cuando falla la carga y hay un código http de error.
//Por ejemplo si no se encuentra el PHP, se generará un error 404.
function httpStatusHandler(event:HTTPStatusEvent):void {
    Estado_txt.appendText("\n" + "Se ha producido el siguiente error: " + event.status);
}

//Se produce cuando falla la carga o descarga
function ioErrorHandler(event:IOErrorEvent):void {
    Estado_txt.appendText("\n" + event.text);
}

//Se inicia la carga o descarga
function openHandler(event:Event):void {
    Estado_txt.appendText("\nComienza la subida");
}

//Se distribuye periodicamente durante la carga o la descarga, mostrando el progreso de la misma.
function progressHandler(event:ProgressEvent):void {
    var file:FileReference = FileReference(event.target);
    Estado_txt.appendText("\n" + event.bytesLoaded + " bytes de " + event.bytesTotal + " bytes subidos.");
}

//Se distribuye al intentar descargar o cargar un archivo de un servidor fuera del entorno de seguridad de la pe
function securityErrorHandler(event:SecurityErrorEvent):void {
    Estado_txt.appendText("\nEl servidor no permitió la carga del archivo.");
}

//Se distribuye al elegir el archivo para carga o descarga desde el navegador de archivos.
function selectHandler(event:Event):void {
    Estado_txt.text = "";
    var archivo:FileReference = FileReference(event.target);
    Estado_txt.appendText("\n" + "Archivo elegido: " + archivo.name + "\n" + "Tamaño = " + archivo.size + " bytes.
    if(archivo.size > 20000){
        Estado_txt.appendText("\nNo se pueden subir archivos de más de 20 KB.");
    }else{
        archivo.upload(uploadURL);
    }
}

Examinar_bt.addEventListener(MouseEvent.CLICK,Examinar);
```

Nota: Código llamando desde Actionscript a un archivo PHP

Figura 25

Código del archivo PHP



Nota: Aquí observamos el código del archivo (uploadFile.php)

Referencias Bibliográficas

Referencias Bibliográficas

ESDEERRE. (2023). *ESDEERRE: Ejemplos de Wordpress, Html y Javascript*. Obtenido de ESDEERRE: Ejemplos de Wordpress, Html y Javascript: https://www.esedeerre.com/page/10/

Jennifer, L. y. (2022). *Hardware, Software, Redes de ordenadores*. Obtenido de Hardware, Software, Redes de ordenadores: https://jennyylika.jimdofree.com/software/tipo-de-archivos/

Jimena. (2011). *TODO TUTORIALES: Código para botones en Flash*. Obtenido de https://todotutoriales-jime.blogspot.com/2011/05/codigo-para-botones-en-flash.html

Matemática, D. d. (2023). *Departamento de Ingeniería Matemática*. Obtenido de https://www.dim.uchile.cl/~jorge/flash/ActionScriptDictionary/12_ASD292.html

Platform, A. F. (2023). *Adobe Flash Platform*. Obtenido de https://help.adobe.com/es_ES/as3/dev/WS5b3ccc516d4fbf351e63e3d118a9b90204-7cfd.html

Wikimedia, F. (Julio de 2021). *WIKIPEDIA La Enciclopedia libre*. Obtenido de https://es.wikipedia.org/wiki/ActionScript


Evaluación
de pares

Quito, 31 de octubre de 2023

Señor Magister
Renato Esteban Revelo Oña
Presidente "R2ICS"
Presente,

De mi consideración:

Una vez realizada la valoración metodológica para la **evaluación par** de la obra **Iniciando en la Programación con Action Script de Flash CS6,** de los autores: Pamela Campos Guillén y Ricardo Tumbaco Orellana; se certifica que la obra cumple con los criterios de relevancia y pertinencia que especifica el respectivo reglamento de Educación Superior, por lo tanto, la misma es recomendable como una **OBRA RELEVANTE**.

Cabe indicar que los contenidos cumplen con estándares de calidad para los procesos de enseñanza y aprendizaje, es inédita y contribuyen al conocimiento y formación de los estudiantes universitarios, de tal manera que resultan fundamentales y sustanciales en la Educación Superior.

Informe que pongo a vuestra consideración para los fines pertinentes.

Atentamente,

Mgtr. Fabián Pazmiño Linzan
CC: 1707992259
Docente de investigación
UNIVERSIDAD DE LAS FUERZAS ARMADAS

r2ics

Red Internacional de Investigación en Ciencias Sociales y Humanidades

CERTIFICADO DE REVISIÓN METODOLÓGICA (PAR EVALUADOR)

Quito, 31 de octubre de 2023

Una vez realizada la valoración metodológica para la **evaluación par** de la obra **Iniciando en la Programación con Action Script de Flash CS6,** de los autores: Pamela Campos Guillén y Ricardo Tumbaco Orellana; se certifica que la obra cumple con los criterios de relevancia y pertinencia que especifica el respectivo reglamento de Educación Superior, por lo tanto, la misma es recomendable como una **OBRA RELEVANTE**.

Cabe indicar que los contenidos cumplen con estándares de calidad para los procesos de enseñanza y aprendizaje, es inédita y contribuyen al conocimiento y formación de los estudiantes universitarios, de tal manera que resultan fundamentales y sustanciales en la Educación Superior.

Atentamente,

Mgtr. Anita Lucía Mata Velasteguí
CC: 1712685831
Revisor Metodológico

www.euroamericano.edu.ec