Programação Multimédia

Flash MX ActionScript

Pedro Costa / 2004

Flash MX ActionScript

Introdução

Neste momento, o ActionScript é provavelmente a ferramenta mais poderosa integrada no Flash.

Esta linguagem foi redesenhada no Flash5 e aperfeiçoada no Flash_MX e tornou-se neste momento uma linguagem completa de programação, baseada no JavaScript.

Devido à generalização do JavaScript, hoje em dia é muito raro que uma aplicação feita em Flash não seja programada em ActionScript.

Flash MX ActionScript

Introdução

O Flash_Mx avançou ainda mais no sentido de tornar esta linguagem num suporte potente e autónoma no desenvolvimento de aplicações para a Internet e outras plataformas, integrando uma linguagem mais poderosa.

Foram melhorados alguns objectos e o mais importante, foram criados novos métodos que conferem à linguagem uma forma de programar distinta e orientada ao objecto.

Flash MX ActionScript

Variáveis

Em qualquer aplicação Flash, tornou-se necessário guardar um valor, seja o número de vidas ainda disponíveis num jogo, o valor das nossas compras numa loja virtual ou mesmo a quantidade de visitas ao nosso site.

Podemos ver uma variável como pequenas caixas onde guardamos informação, ficando disponível para consulta quando assim o entendermos.

Estas “pequenas caixas” estão *sempre* identificadas com um nome, possibilitando a sua identificação e consulta.

Flash MX ActionScript

Variáveis

Declarar Variáveis:

Criar uma variável em ActionScript é idêntico a criar uma variável em JavaScript, é um processo simples de inicialização onde atribuímos um nome para a variável.

Quando se declarar uma variável, esta passa a existir, seja num timeline de um stage , de um movieclip ou integrada num objecto, mas com um valor indefinido. Esta só passa a ter um valor definido quando lhe for atribuído um valor.

Este valor pode ser um número inteiro, uma string ou um boleano(True, False).

Flash MX ActionScript

Regras a cumprir nos nomes das variáveis:

Têm de ser compostas por letras, números e underscore (“_”), não sendo permitido a utilização de pontuação ou espaços entre as letras.

Têm sempre de começar por uma letra ou underscore.

Flash MX ActionScript

Variáveis

Para declarar uma variável utiliza-se o comando "*var*":

```
var ano;
var hora;
var minutos;
```

Também se pode declarar variáveis da seguinte forma:

```
var  ano, hora, minutos;
```

Nota: entenda-se que "var" é o comando e "ano, hora, minutos" são os nomes das variáveis.

Flash MX ActionScript

Variáveis

Atribuir valores:

nome_variavel = valor;

```
var ano = 2005;   // variável cujo valor é um inteiro;
var horas = 8;    // variável cujo valor é um inteiro;
var minutos = 30;          // variável cujo valor é um inteiro;

var primeiro_nome = “Pedro”;     // variável cujo valor é uma string;

var teste1 = TRUE;                    // variável cujo valor é um boleano;
var teste2 = FALSE;        // variável cujo valor é um boleano;
```

Flash MX ActionScript

Variáveis

Alguns operadores:

+	soma;
-	subtracção;
*	multiplicação;
/	divisão;
%	resto de uma divisão;

Flash MX ActionScript

Arrays

Os arrays são utilizados para guardar e manipular dados em forma ordenada, fundamentais para a consulta e o armazenamento sequencial de dados.

Um array é um conjunto estruturado de dados que compreende mais do que um valor, sendo um tipo composto de dados. Pode conter vários tipos de dados, variáveis numéricas, texto ou ainda outros arrays.

Flash MX ActionScript

Arrays

Podemos ver um array como um móvel o número de gavetas que desejarmos. Quando nós queremos guardar algo, colocamos dentro das gavetas.

Se por exemplo o móvel tiver 5 gavetas e se na primeira colocarmos meias, na segunda roupa interior, na terceira t-shirt's, na quarta camisas e na quinta gravatas temos as nossas peças de roupa organizadas por gavetas.

Assim quando queremos meias, sabemos que estão na primeira gaveta e só necessitamos de abrir a primeira gaveta.

Flash MX ActionScript

Arrays

Com os arrays o funcionamento é muito parecido, nós guardamos os dados que necessitamos nas várias posições do array e quando necessitamos acedemos á posição do array que queremos e observamos o valor que se encontra nessa posição.

Aceder a uma posição do array chama-se indexar. Quando indexamos um array podemos, ler o valor dessa posição do array, podemos colocar um novo valor e podemos retirar o valor que se encontra nessa posição.

Flash MX ActionScript

Arrays

Declarar um array:

var meuArray = new Array();

neste caso “meuArray” será o nosso array.

Array contendo texto:

meuArray = [“flash”,”actionscript”,”workshop”];

Flash MX ActionScript

Arrays

meuArray = [“flash”,”actionscript”,”workshop”];

Um array é constituído por elementos, e mais uma vez, a sua indexação é feita com base em zero, por exemplo, para acedermos ao conteúdo “flash” teremos de fazer :

meuArray[0], desta forma estamos a pedir ao array que nos dê o conteúdo que ele tiver na posição zero.

Flash MX ActionScript

Arrays

Assim neste caso na posição zero do array temos “flash”, na segunda temos “actionscript” e por fim na terceira temos “workshop”.

É muito importante não confundir as posições do array com a sua dimensão, neste caso a dimensão do array é quatro, mas para acedermos á ultima posição teremos de fazer “meuArray[3];”, temos de indexar com o valor três porque as posições começam em zero.

Um array pode conter não só texto como também números e também outros array’s.

Flash MX ActionScript

Arrays

Podemos definir as posições do array independentemente:

```
var carros = new Array();

carros[0] = ["BMW"];
carros[1] = ["CITROEN"];
carros[2] = ["FIAT"];
carros[3] = ["VW"];
```

Flash MX ActionScript

Arrays

Aqui temos um array multidimensional:

```
carros[0] = ["BMW","S3","M3"];
carros[1] = ["CITROEN","C3"];
carros[2] = ["FIAT","PUNTO"];
carros[3] = ["VW","POLO"];
```

Seguindo o exemplo das gavetas é fácil perceber um array multidimensional, podendo estar subdivididas.

carros[0][2], desta forma estamos a pedir ao array que nos dê o conteúdo que ele tiver na posição zero, e na subposição dois, devolvendo o valor “M3”.

Flash MX ActionScript

Expressões

==	igualdade;
!=	diferente;
<	menor;
>	maior;
<=	menor ou igual;
>=	maior ou igual;
&&	e lógico;
||	ou lógico;
!	não lógico;

Flash MX ActionScript

Condições

```
if(expressão){
        //código a realizar se a expressão for verdadeira;
}
```

Na condição if é analisada a expressão que estiver dentro dos parêntesis, e se o resultado for verdadeiro ou seja se o retorno da analise da expressão for TRUE, é realizado o código que estiver dentro das chavetas.

Se o resultado for falso ou seja se o retorno da analise da expressão for FALSE, o código que se encontra dentro das chavetas não é realizado.

Flash MX ActionScript

Condições

Existe uma condição que se pode juntar á condição *if* que é a condição *else*. Esta condição permite, em junção com if, decidir se é realizado um troço de código ou outro.

```
if(expressão){
        //código a realizar se a expressão for verdadeira;
}
else{
        //código a realizar se a expressão for falsa;
}
```

Neste caso ao contrário de quando só temos a instrução *if*, se a avaliação da expressão for falsa é realizado o código que está dentro das chavetas da instrução *else*.

Flash MX ActionScript

Condições

```
switch(expressão){
case 1:
//instruções1
break;

case 2:
//instruções2
break;

default:
//instruções3
break;
}
```

Flash MX ActionScript

Condições / Loops

Muitas vezes sentimos necessidade de repetir um troço de código ou uma expressão, um determinado número de vezes. Para tal utilizamos vários tipos de loops diferentes:

while:

Este loop avalia uma condição, se esta for verdadeira o bloco incluído no loop é executado. Quando o resultado da condição for falso, o loop termina. A sua sintaxe é a seguinte:

```
while(condição){
        //código a realizar;
}
```

Flash MX ActionScript

Condições / Loops

Exemplo:

```
var x =0;
while(x < 10){
trace(x);
x++;
}
```

Flash MX ActionScript

Condições / Loops

do-while:

Enquanto o loop while executa o código apenas se a condição for verdadeira, o do-while executa o código pelo menos uma vez antes de avaliar a condição. A sua sintaxe é a seguinte:

```
do{
        //código a realizar;
}while(condição);
```

Flash MX ActionScript

Condições / Loops

Exemplo:

```
var x =0;
do{
trace(x);
x++;
} while (x < 10);
```

Flash MX ActionScript

Condições / Loops

for:

O loop for é semelhante ao while em termos de resultados, mas utiliza uma sintaxe diferente e mais compacta no seu funcionamento.

A sua sintaxe é a seguinte:

```
For(condição inicialização; condição teste; condição actualização){
//código a realizar;
}
```

Neste caso a condição de inicialização é sempre executada em primeiro lugar e uma única vez, seguindo-se a condição de teste e em quanto esta for verdadeira realiza o código que está entra as chavetas. Antes de voltar a repetir o código é sempre realizada a condição de actualização e a condição de teste e só se esta ultima for verdadeira é que volta a repetir o código.

Flash MX ActionScript

Condições / Loops

Exemplo:

```
for(var i=0; i<10; i++){
trace(i);
}
```

Flash MX ActionScript

Funções

As funções são troços de código que podem ser reutilizados ao longo da programação através da sua invocação.

Não só permitem um controlo sobre os objectos que compõem a aplicação de Flash como conferem uma enorme flexibilidade na programação bem como permitem o design modular.

As funções têm a seguinte sintaxe:

```
function nome (parametros) {
código a realizar;
}
```

Flash MX ActionScript

Funções

1)

```
function escreverNome1(){
        trace("o meu nome é Pedro");
}
```

---

2)

```
function escreverNome2(name){
        trace(name);
}
```

Flash MX ActionScript

Funções

As funções podem ou não receber parâmetros.

Em “1“ está representada uma função que não recebe parâmetros, e em “2” uma que recebe um parâmetro.

Se as chamadas ás funções forem:

escreverNome1();

escreverNome2(“o meu nome é Pedro”);

Neste caso vamos obter o mesmo resultado.

Flash MX ActionScript

Funções

Mas se as chamadas forem:

escreverNome1();

escreverNome2(“Costa”);

Agora a primeira função irá escrever na janela de output do Flash “o meu nome é Pedro”.;

A segunda função irá escrever “Costa”;

Em resumo, a primeira função sempre que for chamada irá escrever “o meu nome é Pedro”, a segunda função irá escrever a string que lhe for passada na sua chamada.

Flash MX ActionScript

Slash vs Dot sintax

Em HTML a (/) representa a raiz (_root) e quando se possui um caminho relativo para, por exemplo, subir um nível na hierarquia este é representado por (../).

Em Flash a (/) simboliza o palco e para atribuir uma *action* a um botão que afecte um *movie clip* chamado bola, no palco, basta digitar */bola*. No caso inverso, se possuir um *movie* que vai accionar uma ordem para o palco, então no *movie* deverá aparecer apenas /

Flash MX ActionScript

Dot sintax

A *dot syntax* foi introduzida no actionscripting do Flash 5. É o método preferencial para identificar o caminho para um *movie clip* ou para uma variável.

A *dot sintax* é muito semelhante à *slash sintax*, mas as barras foram substituídas por pontos.

Os caminhos podem ser absolutos e relativos e existem duas referências especiais, **_root** e **_parent**.

A **_root** num caminho absoluto refere-se à *timeline* principal (raiz), da mesma maneira que a (/) na *slash sintax*. A **_parent** refere-se à *timeline* onde o filme (*movie clip*) está inserido, e é semelhante a utilizar (../) numa estrutura HTML, quando especificada em termos relativos.

Flash MX ActionScript

Dot sintax

O topo da árvore é a *timeline* principal do filme, referida como *root* no Flash.

Todos os *movie clips* são instâncias localizadas na *timeline* principal e são dependentes da *root*. Existem dois métodos para referenciar o *target path* de um *movie clip*: Absoluto e relativo.

Flash MX ActionScript

Dot sintax

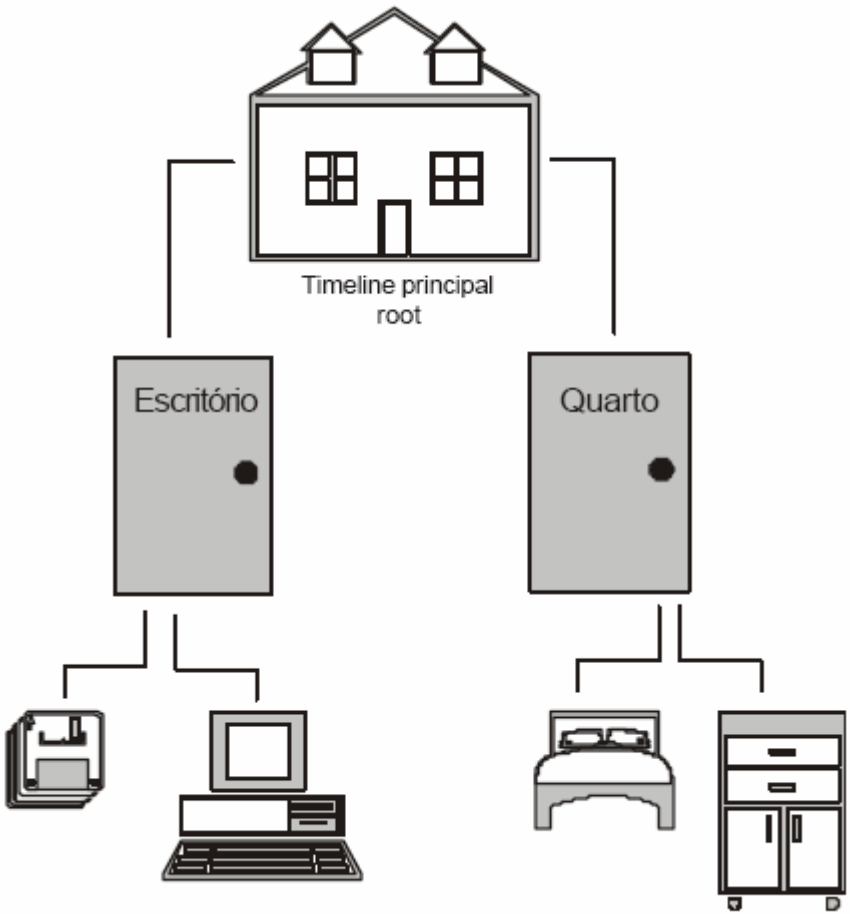

Figura001 - Target Paths

No esquema da figura 001, o caminho absoluto para o *movie* computador será:

_root.escritório.computador,

e o caminho relativo para o *movie clip* quarto a partir do *movie clip* cama será **_parent**.