# Introdução

Apesar do actionScript já existir nas versões anteriores à versão 5 do Flash, a partir desta versão o actionScript foi totalmente re-desenvolvido para que o Flash desfrutasse de uma linguagem de programação completa, moderna e orientada à objetos.

O actionScript foi desenvolvido seguindo as especificações da ECMA-262, que também serviu de base para outras linguagens como o javaScript e o Jscript.

Um site ou uma animação no Flash é composta de elementos gráficos e de texto que normalmente são encapsulados em instâncias que podem ser : graphic, movieClip ou button. E assim, a princípio pode parecer um pouco confuso programar em actionScript, pois o que acontece em termos de codificação é que cada elemento pode possuir um trecho específico de código e como cada elemento possui uma timeline própria, sua timeline ainda pode receber actionScript. Fora a timeline principal que também pode conter trechos de código.

Como qualquer outra linguagem de programação, é necessário dar tempo ao tempo, ie, não é de um dia para o outro que se aprende todos estes conceitos e macetes de programação em actionScript.

# Elementos Básicos do ActionScript

A programação em actionScript se faz com base em conceitos pre-definidos sobre sua estrutura e seus componentes, sendo que os principais são:

## Variáveis

Uma variável é simplesmente um local onde armazenamos um determinado tipo de informação ou dado. Este local é o endereço físico na memória do computador e a função da variável é permitir o acesso e o armazenamento de dados neste local em uma linguagem de alto nível (quase humana), caso contrário teríamos que utilizar a linguagem de máquina para fazê-lo, o que tornaria o processo de programação uma tarefa tediosa, lenta e propícia a erros constantes.

As suas principais características são :

***Nome*** : a variável recebe um nome para que possa ser referenciada via o actionScript

***Valor*** : é o seu conteúdo armazenado

Exemplo : price = 250;

Porquê usar variáveis?

A melhor resposta seria a palavra **Reusabilidade**, que seria o processo de se utilizar o mesmo código, porém com valores diferentes.

Exemplificando : suponha um parágrafo em que o preço de determinado produto apareça n vezes :

*"O mouse que custa R$ 250 reais está disponível nos estabelecimentos x, y e z; porém para o dono do estabelecimento x R$ 250 reais é considerado um preço muito elevado e para o dono de y R$ 250 é um preço razoável. Já para o dono de z, R$ 250 é considerado um preço fora dos padrões admissíveis..."*

Imagine que ao longo de todos os parágrafos o preço '250' a parecesse 500 vezes e por um motivo fosse necessário mudar o preço para R$ 120 reais. O processo seria uma repetição de 500 vezes a troca de 250 por 120, porém se uma variável tivesse sido usada, somente esta necessitaria ser alterada:

price = 120;
*"O mouse que custa R$ price reais está disponível nos estabelecimentos x, y e z; porém para o dono do estabelecimento x R$ price reais é considerado um preço muito elevado e para o dono de y R$ price é um preço razoável. Já para o dono de z, R$ price é considerado um preço fora dos padrões admissíveis..."*

## Tipos de Variáveis

Vimos que ao declarar uma variável, estamos alocando um espaço na memória do computador e armazenando uma

'quantidade' de informação neste local. Portanto quanto mais variáveis utilizarmos, mais memória será necessária. Para que possamos gerenciar melhor a memória utilizada, o Flash possui dois tipos básicos de variáveis : global e local. Sendo que este tipo de declaração se refere ao seu scopo, ie, até aonde a variável está acessível e por quanto tempo.

### *Globais*

Por default, as variáveis declaradas no Flash são globais, ie : estão acessíveis de qualquer local ou objeto no movie principal e o tempo todo. Ao declararmos uma variável como por exemplo : price = 120; estamos implicitamente declarando uma variável do tipo global.

### *Locais*

Se o número de variáveis utilizadas no movie for intensamente grande e se todas forem do tipo global, o excesso de memória alocada pode prejudicar o desempenho do movie, então faz-se necessário fazer com que as variáveis não utilizadas sejam eliminadas para a liberação de memória no computador, e é aí que utilizamos as variáveis locais, que serão utilizadas durante um determinado bloco de código e eliminadas automaticamente após o seu uso. As variáveis locais devem ser explicitamente declaradas da seguinte maneira utilizando-se a palavra reservada 'var'. Ex. var i = 0;

## Tipos de Dados

Note que nos exemplos acima, a variável recebeu um número e outras vezes uma string. Na verdade uma variável pode receber vários tipos de dados, porém por questões de organização, O actionScript bem como qualquer outra linguagem de programação, define seus tipos de dados, cada um com suas características e modos de serem manipulados. Os principais tipos de dados são :

***Number*** : define um valor numérico à variável. Exemplo : x = 10;

***String*** : define um grupo de caracteres à variável, que deve estar entre aspas duplas : nome = "josé";

***Booleans*** : este tipo de variável admite somente 2 valores : true ou false. Normalmente usado para avaliação do status de outras variáveis ou sentenças.  Exemplo :

```
calor = true;
if(calor == true)
{
   trace("Está quente hoje.");
}...
```

Note que, números podem ser somados, strings podem ser concatenadas e booleans são avaliadas. Se declararmos x = "10"; e y = "20", ao fazermos x+y, como x e y são strings e não números, o resultado seria "1020" e não 30.

## Operadores

Utilizamos operadores quando queremos executar algum processo sobre as variáveis. Existem vários operadores no actionscript, porém os principais são :

***Operadores Aritméticos*** : *, /, +, - ; nesta ordem de precedência.

Exemplo : 2+3*3 = 11. Se o objetivo for fazer com que 2+3 seja executado antes, devemos 'quebrar' a ordem de precedência com a utilização de parênteses : (2+3)*3 = 15

***Operadores de Comparação*** : <, >, <=, >=

## Objetos do Flash e suas Propriedades

O Flash possui objetos pré-definidos e prontos para serem utilizados, sendo que um dos mais utilizados é o ***movieClip***. Este possui propriedades default que podem ser acessadas a qualquer momento, como : _width, _scale, _x, _y, etc. Cada tipo de objeto possui suas propriedades pré-definidas e no momento de sua criação o Flash automaticamente seleciona as propriedades disponíveis para o determinado objeto. Por exemplo, o objeto sound possui propriedades como : _volume, _pan, etc. Mas não possui a propriedade _scale, pois isto não faz sentido.

A utilização de Objetos é a principal característica da POO - Programação Orientada a Objetos, que se difere muito das antigas linguagens de programação onde os programas começavam a sua execução na primeira linha e seguiam rigidamente em um sentido top-down a execução do código até que se atingisse a última linha do programa. Isto funcionava relativamente bem em programas em modo DOS ou Unix, porém com a introdução de interfaces gráficas mais avançadas e do mouse, cada parte da tela do computador estaria sujeita ao click do mouse e em cada uma dessas partes um trecho de código deveria ser executado, ficando claro que um programa que deveria executar o seu código do começo ao fim não poderia ser utilizado de modo adequado em um sistema com uma inteface gráfica dessas. E foi a partir desta necessidade e dificuldade a ser resolvida que surgiu a POO.

Podemos fazer uma analogia a este fato para entendermos como programar no Flash com o actionScript. Cada um de seus objetos incluindo as timelines são pontos onde trechos de código podem ser inseridos e serão executados em momento oportuno, como por exemplo no click do mouse sobre o mc ou no frame número n de sua timeline. E é através da combinação desses pequenos trechos de códigos e de seu posicionamento correto que o Flash pode se tornar uma poderosa ferramenta de desenvolvimento.

## Dot Sintax : sintaxe de pontos

O actionScript utiliza o que nós chamamos de dotSintax que é usada para se acessar os objetos a partir de outros objetos. Dot Sintax é característico das linguagens orientadas à objeto como : C++, Java ou até mesmo o javaScript.

Ex. : considere 2 mcs sendo que o primeiro se chama 'ext' e o segundo 'int', e considerando que o segundo mc está dentro do primeiro mc. Se a partir do stage quiséssemos saber qual a posição x do mc int (int._x), teríamos a seguinte notação :

xpos = _root.ext.int._x

onde _x é a propriedade que retorna a posição horizontal do objeto.

## ESTRUTURAS DO ACTIONSCRIPT

O modo com que combinamos os elementos do actionScript formarão as estruturas do actionscript. As principais são:

***Expressões***

A combinação das variáveis com os operadores formam as expressões e consequentemente estas geram um novo valor. Ex. : 3 + 5 = 8

***Actions***

Quando executadas geram uma ação expecífica. As actions devem ser finalizadas com o ';'.
Ex. : gotoAndStop(5);

São 2 os tipos de actions : frameActions e objectActions

frameActions : são porções de códigos inseridos em um determinado keyFrame e são executados no momento em que o movie atinja o frame em questão.

objectActions : são porções de código atachadas ao objeto que pode ser um button ou um movieClip.

## *Métodos*

Um método é algo que um objeto possa fazer ou algo que se possa fazer com o objeto.Ex. : myMc.stop();

***Blocos***

São sentenças agrupadas e localizadas entre parênteses : {}

```
Ex. : on(release)
      {
         size = 50;
         ball._scale = size;
      }
```

## Estruturas de Decisão

São usadas para determinar qual bloco devemos executar.

```
Ex. : if( i < 10)
      {
          trace("i="+i);
      }
      else
      {
         stop();
      }
```

## Loops

São utilizados para que determinados blocos sejam executados repetidamente. São 3 os tipos de loops :

```
i=10;
While(i<5)
{
   trace(i);
}
```

---

```
i=10;
do
{
   trace(i);
}While(i<5)
```

---

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

1. o loop ***while*** testa primeiramente a condição e depois caso a satisfaça, o bloco é executado. No exemplo acima, trace(i) não será executado!

2. o loop ***doWhile*** executa o bloco e depois testa a condição. No exemplo acima, trace(i) será executado 1 vez!

3. o loop ***for*** possui condição pre-estabelecida para terminar, pois i++ fará o incremento em uma unidade da variável i até que ao atingir o valor 10 e o loop será parado.

## Funções

Uma função é um bloco de código com um nome determinado que pode ser chamado de qualquer lugar do seu movie por um outro trecho de código.

Usamos funções quando possuímos um determinado bloco que precise ser utilizado várias vezes. Isto ajuda também a otimizar a quantidade de código a ser digitada.

Ex. : suponha 3 botões no stage de seu movie sendo que cada um deles precise abrir uma URL de acordo com a variável de controle i.

O código do botão 1 seria :

```
On(release)
{
   i=1;
   if(i=1)
```

```
   {
      getURL("http://www.flash.com", "_blank");
   {
   else if(i=2)
   {
       getURL("http://www.livemotion.com", "_blank");
   {
   else if(i=3)
   {
       getURL("http://www.afterFXs.com", "_blank");
   {
}
```

Os códigos dos botões 2 e 3 seriam os mesmos com a diferença que cada um deles forneceria à variável i os valores 2 e 3 respectivamente.

Para otimizarmos o processo poderíamos criar uma função no frame 1 da timeline principal da seguinte maneira:

```
Function abreURL(i)
{
   if(i=1)
   {
      getURL("http://www.flash.com", "_blank");
   {
   else if(i=2)
   {
      getURL("http://www.livemotion.com", "_blank");
   {
   else if(i=3)
   {
      getURL("http://www.afterFXs.com", "_blank");
   {
}
```

E os botões teriam a seguinte estrutura:

```
On(release)
{
   abreURL(1);
}

On(release)
{
   abreURL(2);
}

On(release)
{
   abreURL(3);
}
```

Deste modo cada botão chamaria a função abreURL(i) e passaria o parâmetro i = 1,2 ou 3, otimizando assim a quantidade de código utilizado bem como facilitando o entendimento e a leitura deste código. Via de regra, uma função deve executar somente uma tarefa e devemos nomear funções com nomes significativos. Caso uma função possua sub-tarefas, devemos criar novas funções para estas sub-tarefas

Estes são os conceitos básicos para se programar em actionScript. Para um maior aprofundamento no assunto deve-se consultar o Dicionário ActionScript que está no Help do Flash e que possui uma versão atualizada à disposição para download no site da Macromedia : http://www.macromedia.com/

---