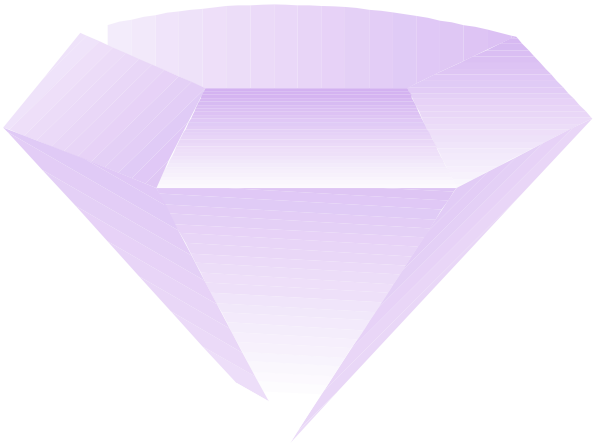
Diamond´s Pit



-TUTORIAIS FLASH -

www.diamante.websolutions.com.br

ACTION SCRIPT

Belo Horizonte – MG
Brasil
30-06-2001

ACTIONS

Action: Drag Movie Clip

Sintaxe:

Start Drag ("[Target]", L=[Boundary], T=[Boundary], R=[Boundary], B=[Boundary], [Lockcenter])
Start Drag

Parâmetros:

Target- Identifica o movie clip a ser arrastado.

Boundary - se este parâmetro for definido em L (Left), T (Top), R (Right), B (Bottom) é definido então um rectângulo por onde é possível arrastar o movie cilp. O valor de Boundary é a distância máxima possível de arrastar em cada uma das direcções e o centro do movie clip onde o arrastar acontece.

Lockcenter - Se esta opção for checada então o centro do movie clip estará sempre junto do ponteiro do rato. Caso contrário o movie clip terá sempre uma distância em relação ao rato.

Stop Drag - Associado a um evento do rato pára de arrastar um determinado movie clip.

Exemplos de uso:

Jogos, carrinhos de compras ou qualquer outra situação onde seja necessário a função drag and drop.

Descrição:

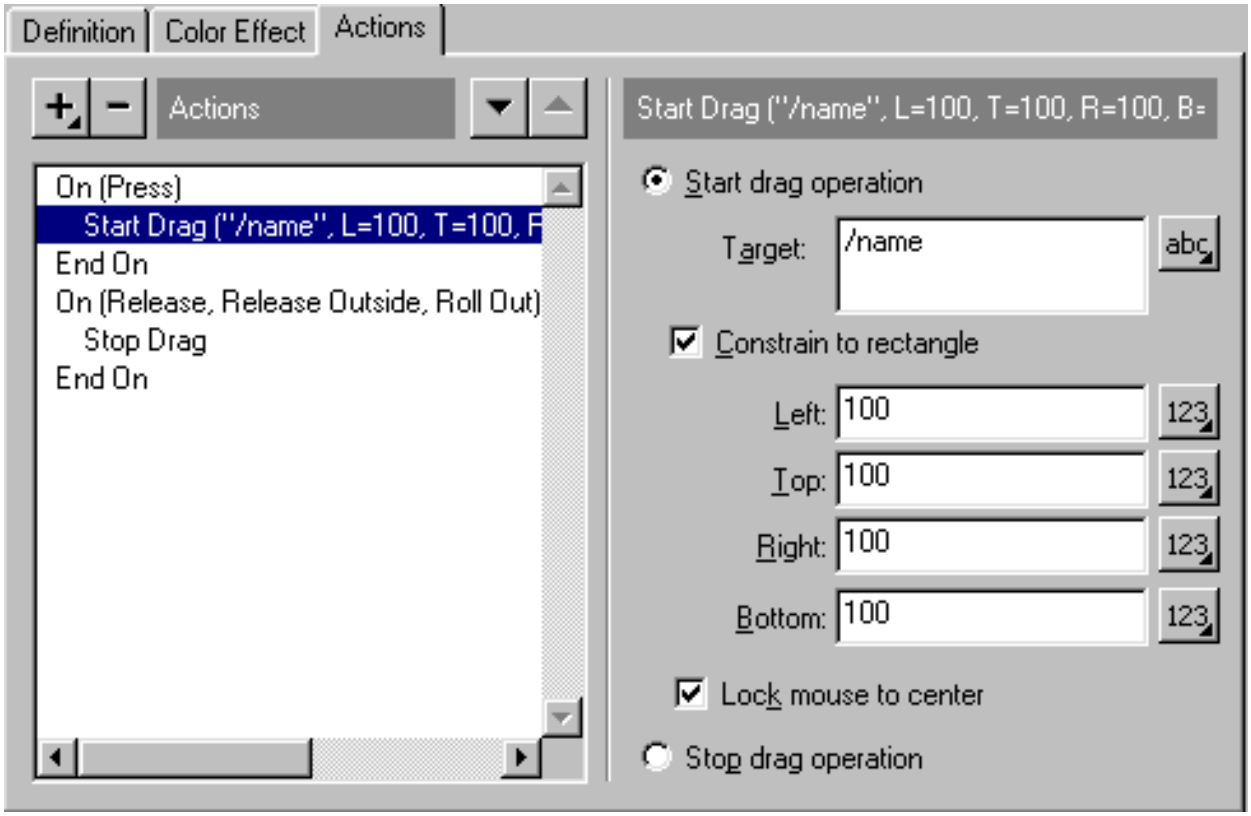
Esta acção deve ser usada com o evento **On (Press)**, quer isto dizer, enquanto o botão do rato se mantém pressionado. Para usar a função é necessário primeiro definir o **target** (movie clip a ser arrastado). A seguir, se a opção **Constrain rectangle** não for checada o filme será arrastavel livremente por todo a janela do flashplayer. Se a opção anterior for checada será necessário definir os valores boundary que irão restringir a área arrastavel. A opção final da função é **Lockcenter** esta função terá o seu comportamento sujeito ás opções anteriores.

Todo start tem um stop e Stop Dragging requere um novo evento do rato. Para activar a acção basta checar a opção Stop Drag . (ver exemplo junto)

Exemplo básico:

```
On (Press)
    Start Drag ("/name", L=100, T=100, R=100, B=100, lockcenter)
End On
On (Release, Release Outside, Roll Out)
```

Stop Drag
End On



Action: Duplicate/Remove Movie Clip

Sintaxe:

Duplicate Movie Clip (["Target"], ["New Name"], [Depth])
Remove Movie Clip (["Target"])

Parâmetros:

Action - Duplicar ou Remover movie clip duplicado.

Target - Caminho do movie clip original a ser duplicado ou caminho do movie clip a ser removido

New Name - Nome do duplicado

Depth - posição em relação a outros movie clip duplicados. (0 - nível mais abaixo, >= 1 aparece mais acima, quanto maior for o valor mais acima aparece o nosso objecto.

Exemplos de uso:

Em tudo o que requer múltiplas copias do mesmo objecto tal como jogos e sistemas de menu.

Descrição:

Com esta acção está-se apto para duplicar e remover movies em tempo real. Basta colocar o nosso movie original na primeira frame da timeline e depois é só duplicar. Os movies duplicados ficam com as características do original.

Exemplo Básico:

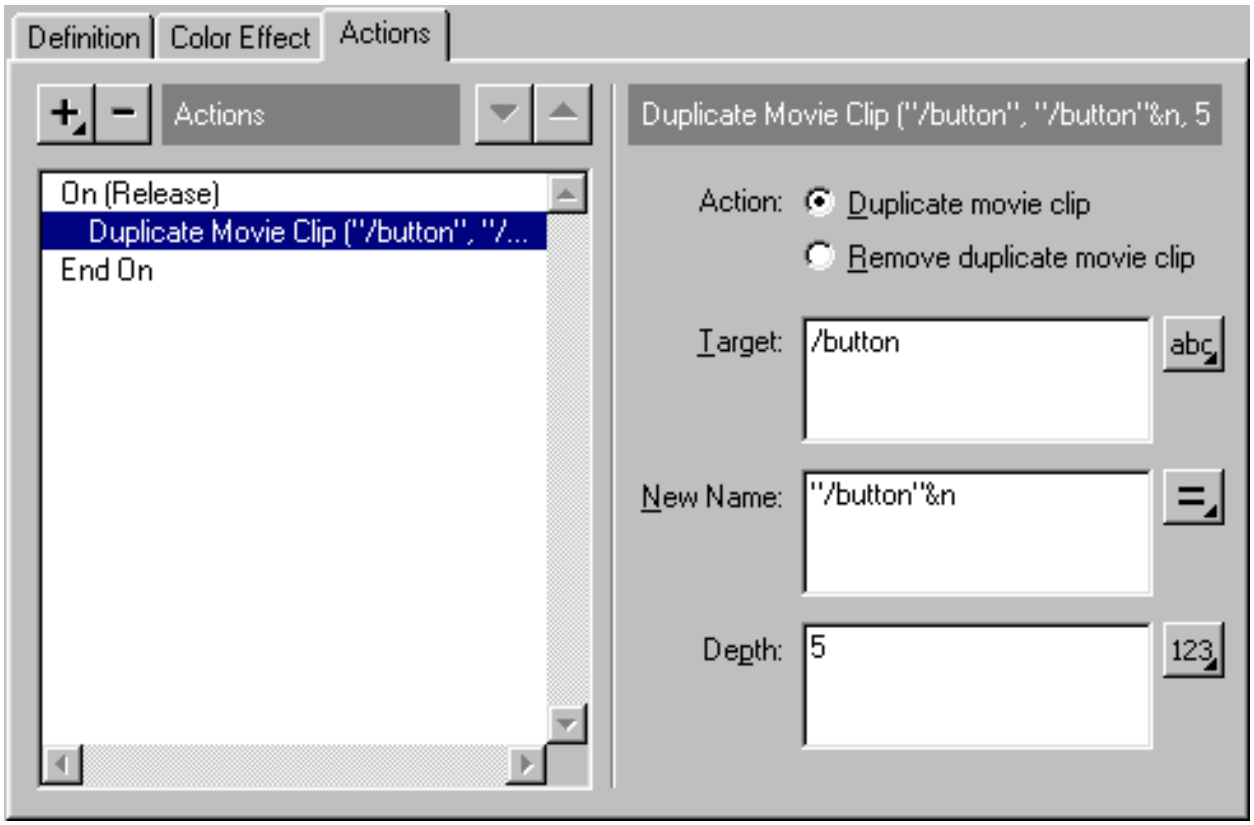
```
On (Release)
    Duplicate Movie Clip ("/person", clone, 5)
End On
```

Exemplo Intermédio:

```
On (Release)
    Remove Movie Clip ("/clone")
End On
```

Exemplo Avançado:

```
Set Variable: "n" = 1
Loop While (n < 10)
      Duplicate/Remove Movie Clip ("/button", button&amp;mp;n, n)
      Set Variable: "n" = n + 1
End Loop
```



Action: Get URL

Sintaxe:

Get URL ("URL", Window/Variables)

Parâmetros:

URL - define o URL para a action, poderá ser relativo ou absoluto.

Window - Identifica como deve ser carregado o URL

1. _self - carrega o URL no filme no filme corrente.
2. _blank - carrega o URL numa nova janela de browser.
3. _parent - carrega o URL dentro de uma frame parente da frame corrente.
4. _top - remove todas as frames e carrega o URL na janela do browser.

Variables - define como as variáveis são tratadas.

1. Don't Send - Não envia variáveis. Opção por defeito.
2. Send using GET - envia um pequeno numero de variáveis utilizando a barra do URL.
3. Send using Post - aconselhado quando se usam forms de flash e envia as variáveis separadamente para o URL. Quando usamos CGI ou algo similar é este o método a usar.

Exemplos de uso:

Navegação HTML dentro de filmes flash, abrir janelas de browser, enviar variaveis como uma form html.

Descrição:

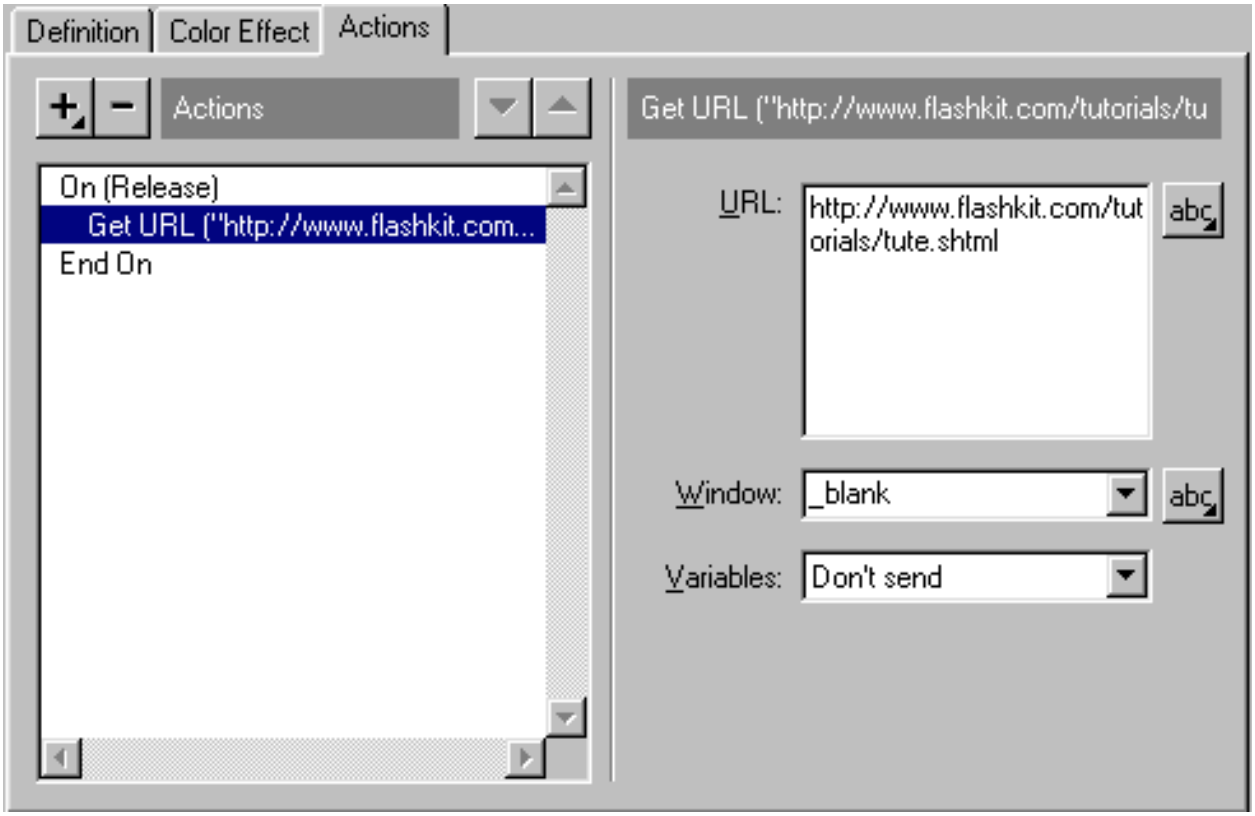
Get URL pode executar dois papéis. Primeiro, carregar numa janela de browser um URL, segundo, enviar variáveis do filme actual para um URL ou para um CGI. O primeiro papel será a principal utilidade. Cria "hyperlinks" para outros filmes de flash aceitando caminhos relativos e absolutos.

Exemplo básico:

```
On (Release)
    Get URL ("http://www.truquesedicas.com/tutoriais/00003.html", window="_blank")
End On
```

Exemplo avançado:

```
On (Release)
    Get URL ("http://www.truquesedicas.com/cgi-bin/truques.cgi", vars="POST")
End On
```



Action: Goto

Sintaxe:

Go to and [Play/Stop] ([Scene],[Frame/"Frame Label"/Expression])

Exemplos de uso:

Em qualquer situação! É um dos comandos fundamentais em actionscripting.

Descrição:

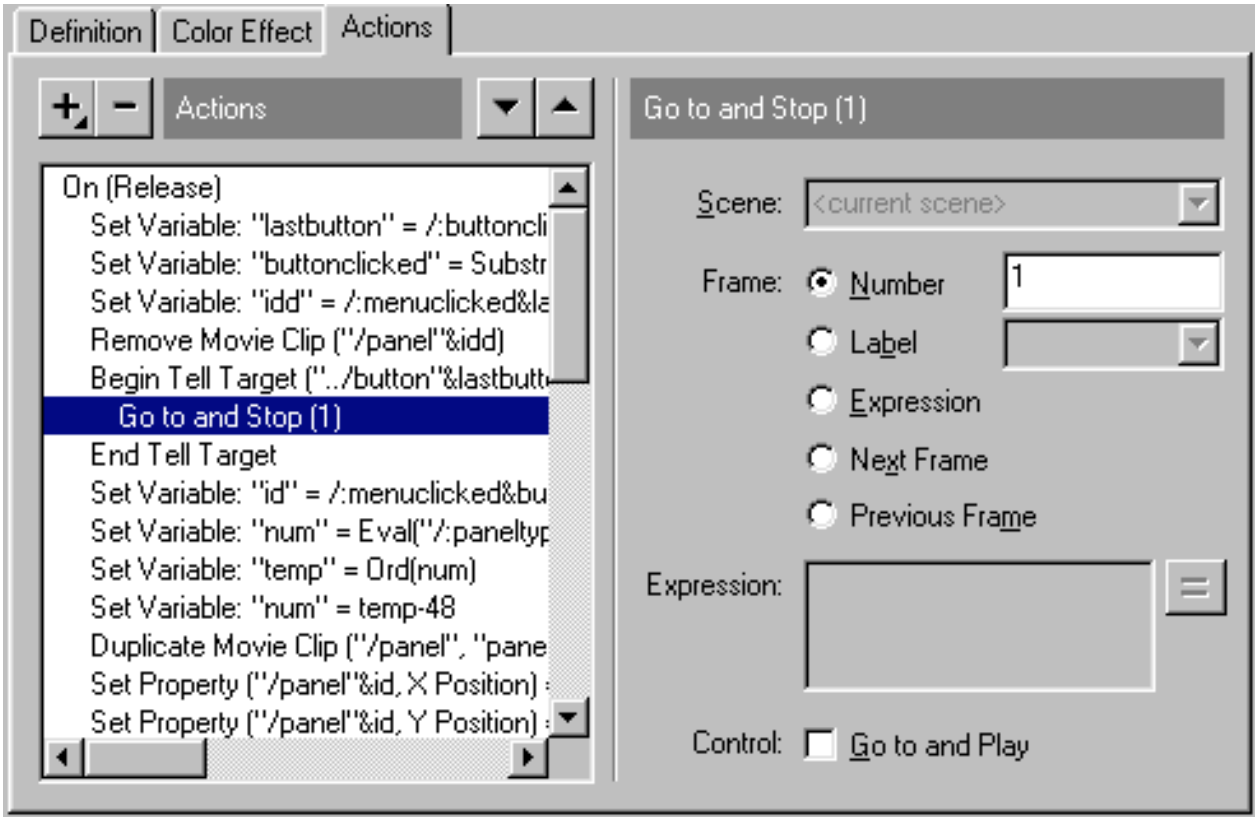
Chama um movie clip e fá-lo correr (**Play**) ou parar (**Stop**) a partir de um numero ou de um nome (**Label**) de uma frame. É possível também dizer em que **Scene** se pretende que isto aconteça bem como adicionar interactividade usando uma expressão.

Exemplo básico:

```
On (Release)
    Go to and Stop(mainmovie, 1)
End On
```

Exemplo avançado:

```
On (Release)
    Begin Tell Target ("mymovie")
        Go to and Play ("start")
    End Tell Target
End On
```



Action: Call

Sintaxe:

Call (["Frame Label"])

Parâmetros:

Frame Label - identifica a frame que contém a actionscript a ser invocada.

Descrição:

Possibilita chamar qualquer actionscript em qualquer timeline invocando o nome (**label**) de uma frame. Isto mantém o código limpo e ajuda a minimizar o tempo de programação. Variaveis usadas na frame invocada têm que ser chamadas usando a actionscript Set Variable action**.**

No exemplo avançado abaixo a variável "**exemplo**" é criada assumindo o valor da TextField "**var_textfield**". A seguir a frame "**decide**" é chamada e a variável "**exemplo**" é usada para cálculos na frame "**decide**" o output é a variável "**decisao**".
Finalmente o filme vai para a frame "**decisao**" e corre o seu conteúdo dando o feedback ao utilizador. (Não é tão complicado quanto parece!)

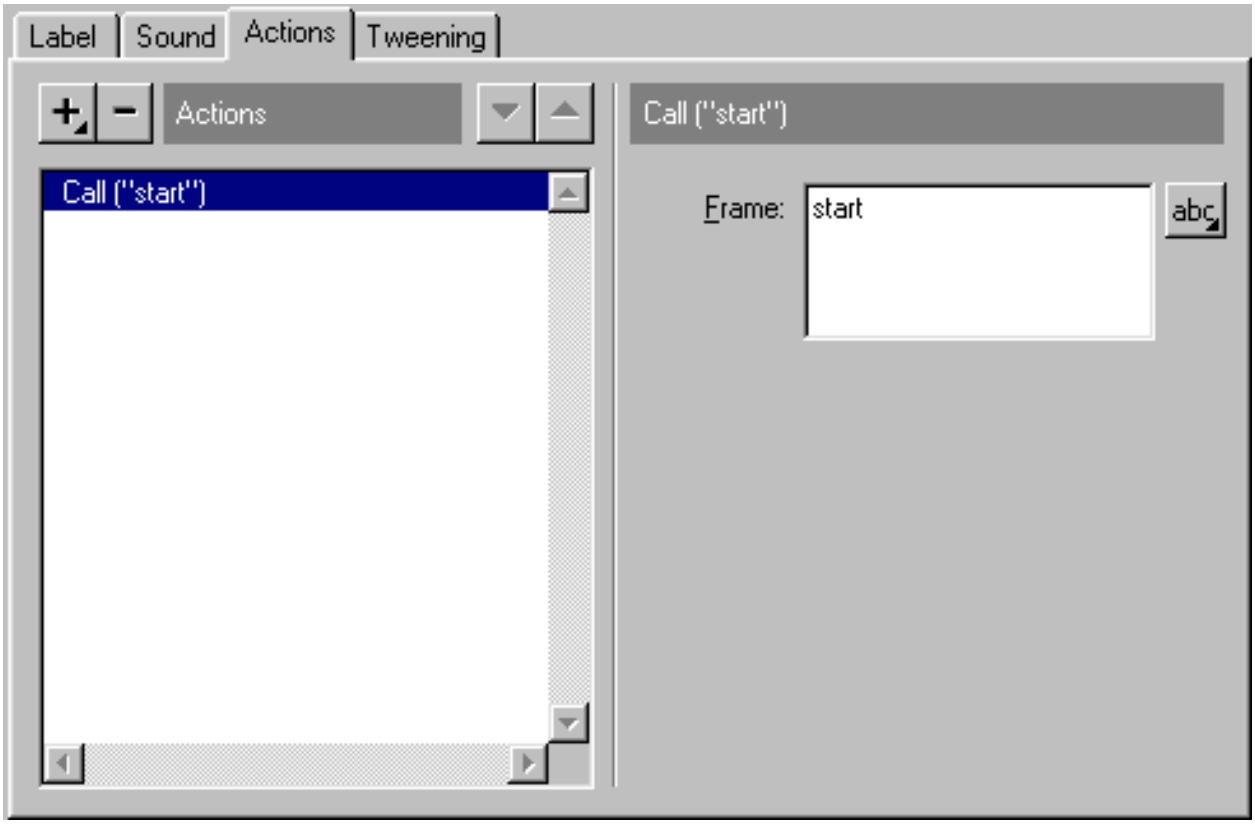
É necessário garantir que a frame que é chamada foi carregada, caso contrário será ignorada.

Exemplo básico:

Call ("Start")

Exemplo Avançado:

```
On(Release)
    Set Variable: "exemplo" = var_textfield
    Call ("decide")
    Go to and Play (decisao)
End On
```



Action: Comment

Sintaxe:

Comment [Message]

Descrição:

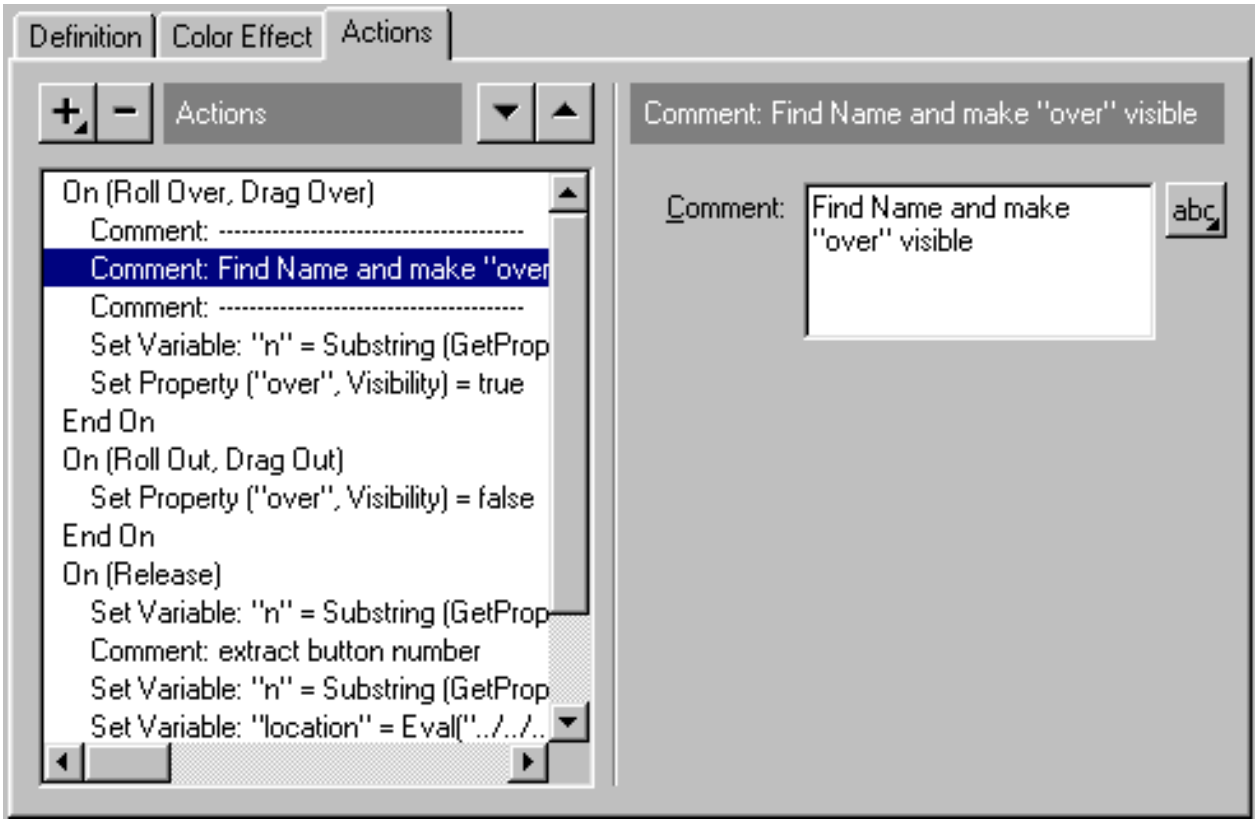
Todos gostamos de ter o código bem documentado mas muitos detestam fazê-lo. Esta acção ajuda a comentar código facilmente e assim, após algum tempo, quando se volta ao código tudo se torna mais fácil de entender. É a melhor forma de deixar informações para que outros entendam o nosso código.

Exemplo básico:

Comment: Code for Menu Button

Exemplo básico:

```
Comment: ----------------------------------------------------------------------------
Comment:          Good Clear Documentation is important
Comment: ----------------------------------------------------------------------------
```



Action: If Frame Is Loaded

Sintaxe:

If Frame Is Loaded ([Scene], Frame/"Frame Label"])
    [Action]
End Frame Loaded

Parâmetros:

Scene - Cena que contém a frame que vai ser testada se foi carregada.

Frame - Numero da frame a ser testada.

Frame Label - Nome da frame a ser testada.

Actions - O que é executado quando a frame já tiver sido carregada.

Exemplos de uso:

Em preloaders e quando é necessário ter a certeza que uma informação já foi carregada.

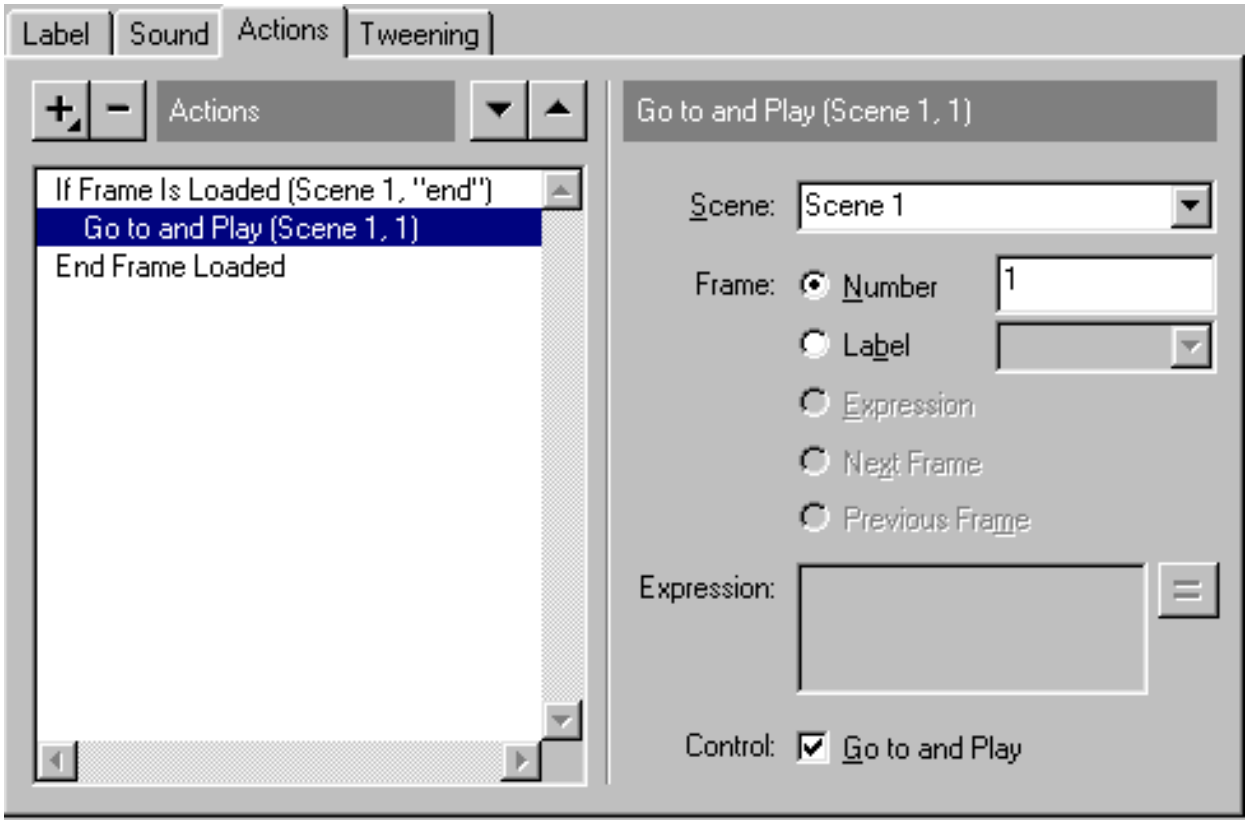
Descrição:

Uso desta função em preloaders. Com esta função é possível criar uma barra de progressão colocando no inicio do filme esta função a garantir que a primeira parte da animação foi carregada (por exemplo frame 1: **if** (frame 20) **is loaded goto** frame 10; frame 2: **goto** frame 1; frame 10 **play** filme. Pode-se colocar isto num movie instance para exibir uma barra de progressão no background enquanto está a carregar um som. Outro utilidade é o uso desta função para certificar se o conteúdo de uma frame já foi carregado antes de um comando

chamar esta frame.

Exemplo:

```
If  Frame Is Loaded (Scene 1, "end")
    Go to and Play (Scene 1, 1)
End Frame Loaded
```



Action: If

Sintaxe:

```
If ([Condition])
   [Actions]
Else If ([Condition]) - Optional
   [Actions] - Optional
Else - Optional
   [Actions] - Optional
End If
```

Parâmetros:

Condition - condição que necessita ser verdadeira para a acção (**action**) ser executada.

Actions - comandos que se executam quando a condição é válida

Exemplos de uso:

Em qualquer situação! É um dos comandos fundamentais em actionscripting.

Descrição:

Esta action é de extrema importância, é ela que permite decidir qual a continuação de um filme. De uma maneira fácil é uma instrução que caso seja verdadeira é executada uma acção. Por exemplo, se está sol então o céu é azul (ver exemplo básico). Em alternativa é possível dar uma instrução diferente se não estiver sol (ver exemplo intermédio). O exemplo final (avançado), é possível ver-se o uso da instrução **Else if**, serve para testar condições a seguir á primeira ter falhado e então executar acções apropriadas. É possivel adicionar-se o número de **Else if** que se pretender.

Exemplo básico:

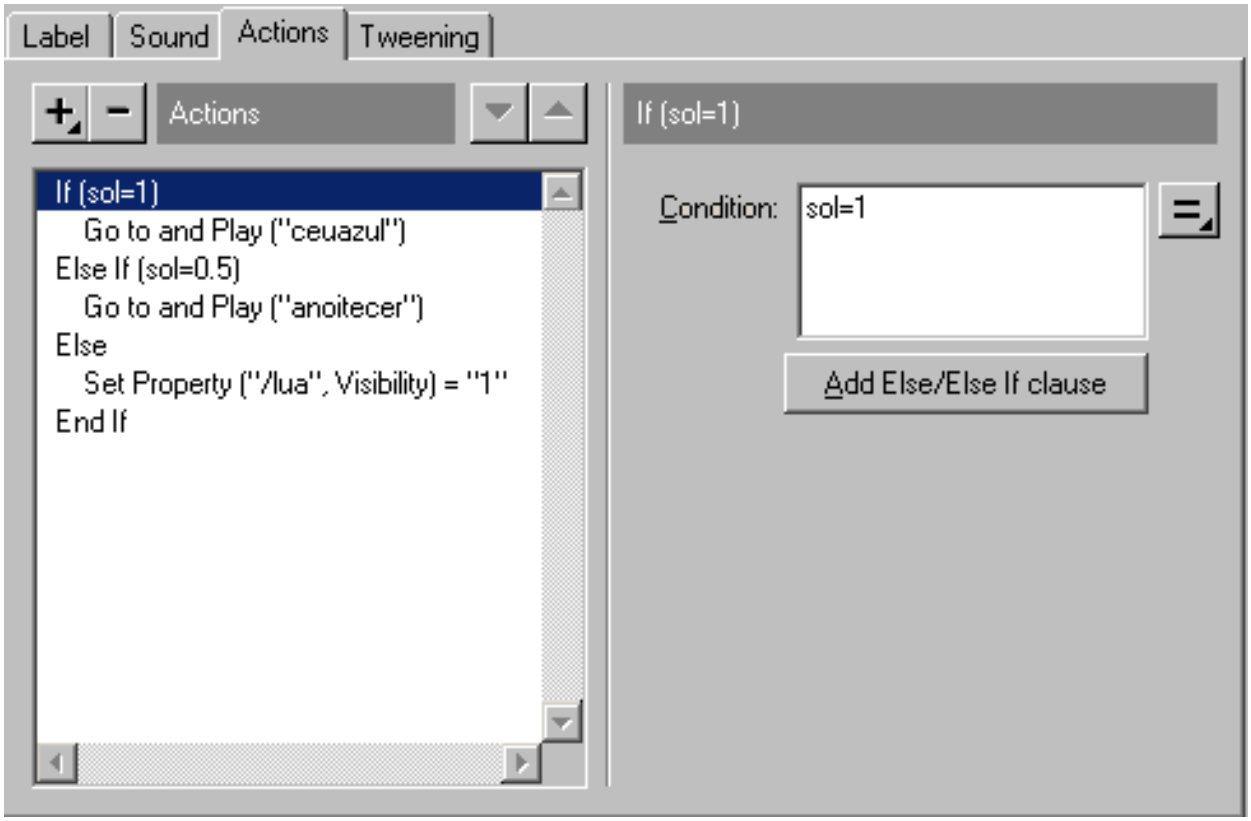
```
If (sol = 1)
    Go to and Play ("ceuazul")
End If
```

Exemplo intermédio:

```
If (sol = 1)
    Go to and Play ("ceuazul")
Else
    Go to and Play ("lua")
End if
```

Exemplo avançado:

```
If (sol = 1)
    Go to and Play ("ceuazul")
Else if (sol = 0.5)
    Go to and Play ("anoitecer")
Else
    Set Property ("/lua", Visibility) = "1"
End if
```



Action: Load/Unload Movie

Sintaxe:

Load Movie ([Target], [Location])
Unload Movie ([Location])
Load Variables ([Datafile], [Location])
Load Variables ([Script], "/" ,[Variable])

Parâmetros:

Target - Identifica o movie clip a ser carregado

Location - Define o level(nível) ou a localização onde o movie clip deve ser carragado

Datafile - Endereço absoluto ou relativo do arquivo que contém as variáveis. ie "http://www.truquesedicas.com/data.txt"

Script - Endereço absoluto ou relativo do um CGI para o qual são enviadas as variáveis.

Variables - Opção **Don't Send** não são enviadas variáveis, **GET** envia variáveis junto com o URL, **POST** útil para enviar um largo número de variáveis.

Exemplos de uso:

Jogos, aplicações flash e musica.

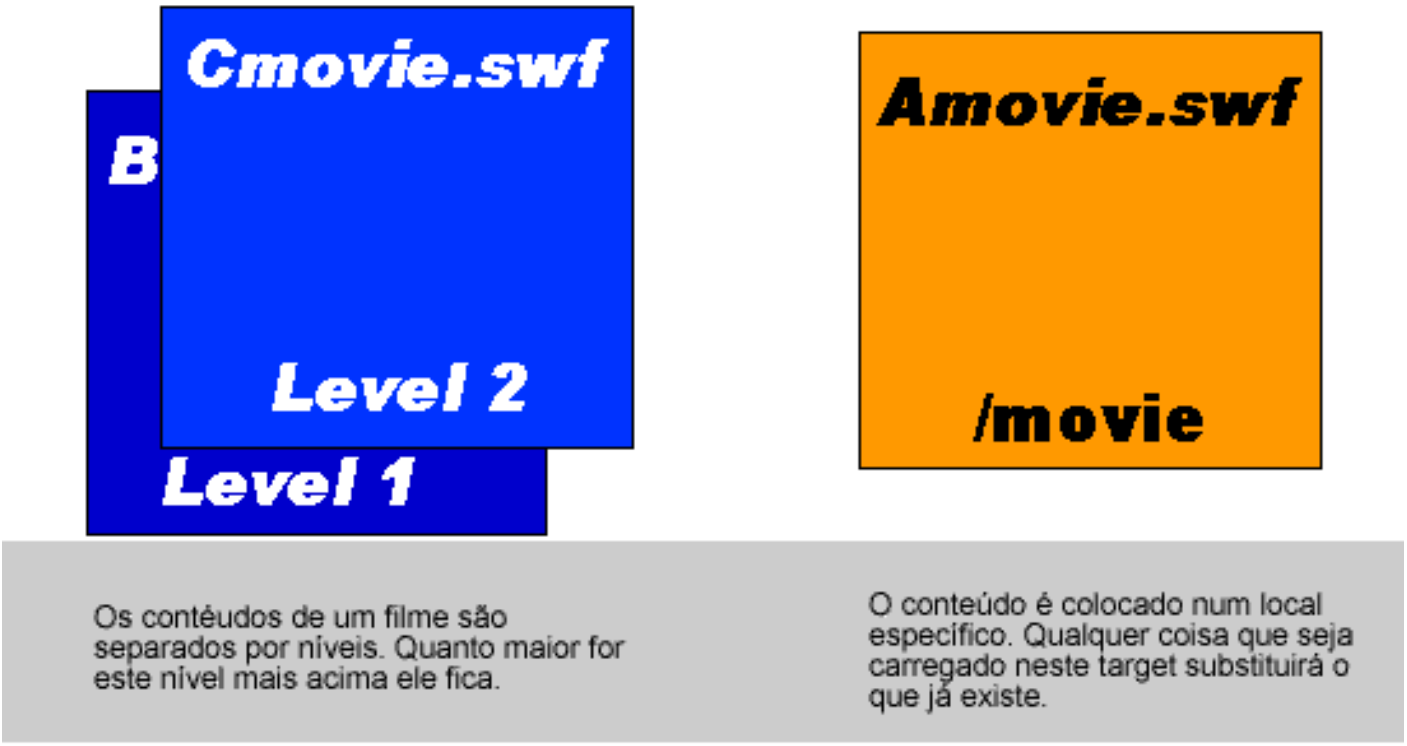
Descrição:

Esta acção é na verdade uma serie de acções, como se pode ver na lista a seguir. Load movie

action permite fazer o seguinte:

1.Carregar um Flash movie dentro de um filme que está a correr.
2.Carregar um movie para substituir um movie já carregado.
3.Carregar variáveis de um arquivo de texto.
4.Retirar um filme previamente carregado dentro do filme corrente.
5.Enviar Variáveis para um script (CGI) para ser processado.

Para usar a acção Load movie será necessário entender o conceito de level e target. Ver a imagem abaixo:



Carregar um movie num level é como trabalhar com layers. Um movie carregado num level 5 é exibido sempre atrás dos conteúdos que estão em níveis superiores 6, 7, etc. A timeline mãe está no nível 0.

Em termos de targets, ao carregar um filme num target que já contém um movie carregado o resultado é a substituição do movie antigo pelo agora carregado. A acção load movie pode ser usada também para carregar e enviar variáveis para scripts.

Load Movie exemplo (level):

```
On (Release)
    Load Movie ("/movie", 1)
End On
```

Load Movie exemplo (target):

On (Release)

```
    Load Movie ("/movie", "/")
End On
```

Unload Movie exemplo:

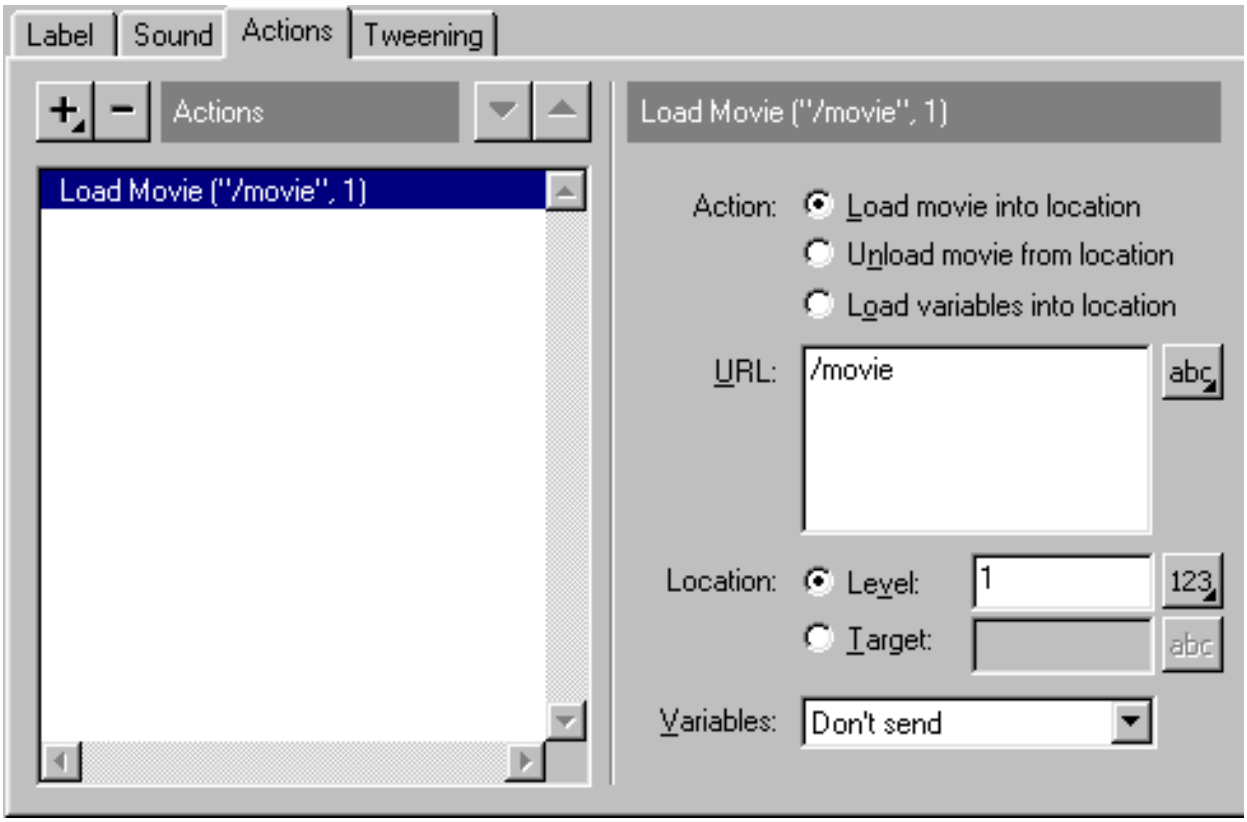
```
On (Release)
   Unload Movie (1)
End On
```

Load Variables exemplo:

```
On (Release)
    Load Variables ("data.txt", "/")
End On
```

Send Variables exemplo:

```
On (Release)
    Load Variables ("http://www.flashkit.com/cgi-bin/myscript.cgi", "/", vars=POST)
End On
```



Action: Loop

Sintaxe:

```
Loop While ([Condition])
    [Actions]
End Loop
```

Parâmetros:

Condition- Condição que tem que ser verdadeira para que a acção seja executada.

Actions - comandos que são executados enquanto a condição é verdadeira.

Exemplos de uso:

Em qualquer local que seja necessário que uma acção corra continuamente enquanto a condição é verdadeira.

Descrição:

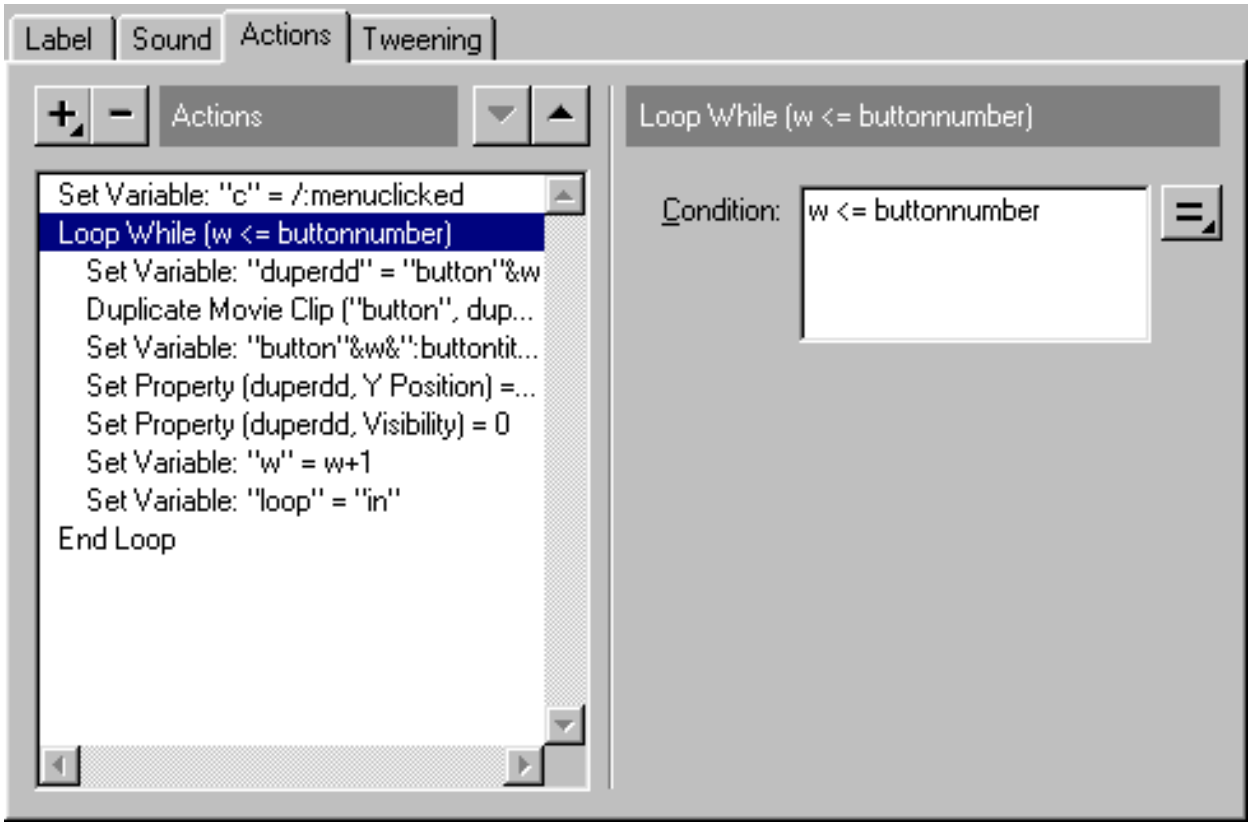
A função Loop permite fazer acções repetidas numa única frame. Pegando na analogia sol/céu/lua vamos ver como funciona o loop. O tempo do sol é mais ou menos 12 horas assim, enquanto o sol for menor ou igual a 12 o céu é azul. Quando o sol for maior que 12, sai do loop e continua as acções seguintes. (ver exemplo básico)

Exemplo Básico:

```
Loop While (sol <=12)
    Go to and Play (ceuazul)
    Set Variable: "sol " = sol +1
End Loop
Go to and Play (lua)
```

Exemplo Avançado:

```
Set Variable: "c" = /:number
Loop While (w <= buttonnumber)
      Set Variable: "dup" = "button"&w
      Duplicate Movie Clip ("button", dup, w)
      Set Variable: "dup"&w&":buttontitle" = Eval ("/:buttontitle"&c & w)
      Set Property (dup, Y Position) = w+100
      Set Property (dup, Visibility) = 0
      Set Variable: "w" = w+1
End Loop
```



Action: Play

Sintaxe:

Play

Exemplos de uso:

Em qualquer local onde se pretenda que comece o progresso de uma timeline.

Descrição:

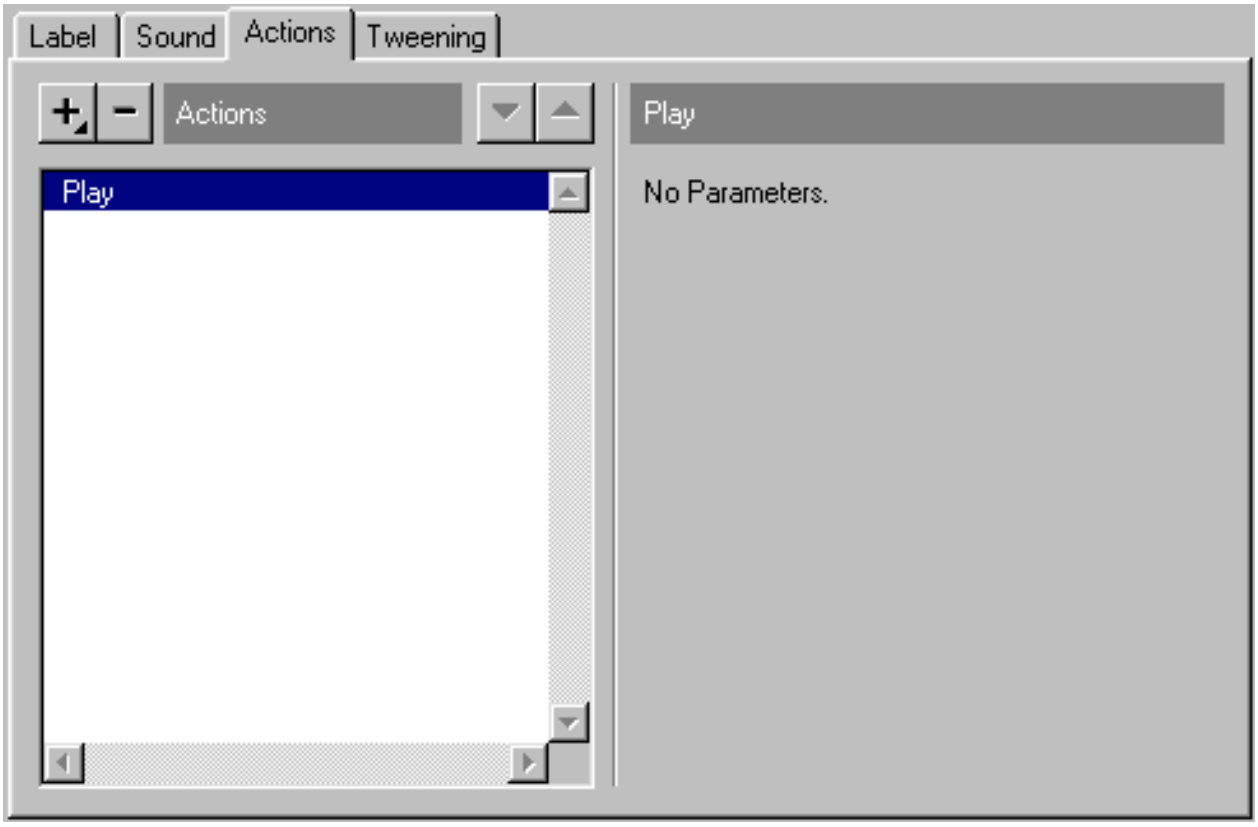
Chama um movie para que comece o progresso da sua timeline a partir da frame que está. Para parar esta acção será necessário a acção Stop ou Goto and Stop.

Exemplo Básico:

```
On (Press)
      Play
End On
```

Exemplo Avançado:

```
On (Press)
      Begin Tell Target ("mymovie")
            Play
      End Tell Target
End On
```



Action: Set Property

Sintaxe:

Set Property(["Target"],[Set])

Parâmetros:

Set - O utilizador define através de uma listbox a propriedade desejada.

Target - Identifica o objecto que vai ser alterado pela propriedade seleccionada em Set.

Value - Valor da propriedade.

Exemplos de uso:

Interactividade na posição dos objectos, transparência, visibilidade, rotação, movimentos aleatórios, etc.

Descrição:

A acção Set Property abre as portas á interactividade. Através desta acção é possível em tempo real alterar as propriedades do nosso flash movie bem como dos seus objectos.

- **X Position** - posição horizontal de um movie em relação ao centro do filme onde está inserido. (pixels)
- **Y Position** - posição vertical de um movie em relação ao centro do filme onde está inserido. (pixels)
- **X Scale** - escala um movie horizontalmente, mais estreito se o valor for inferior a 100%

e mais largo se for maior que 100%. (percentagem)

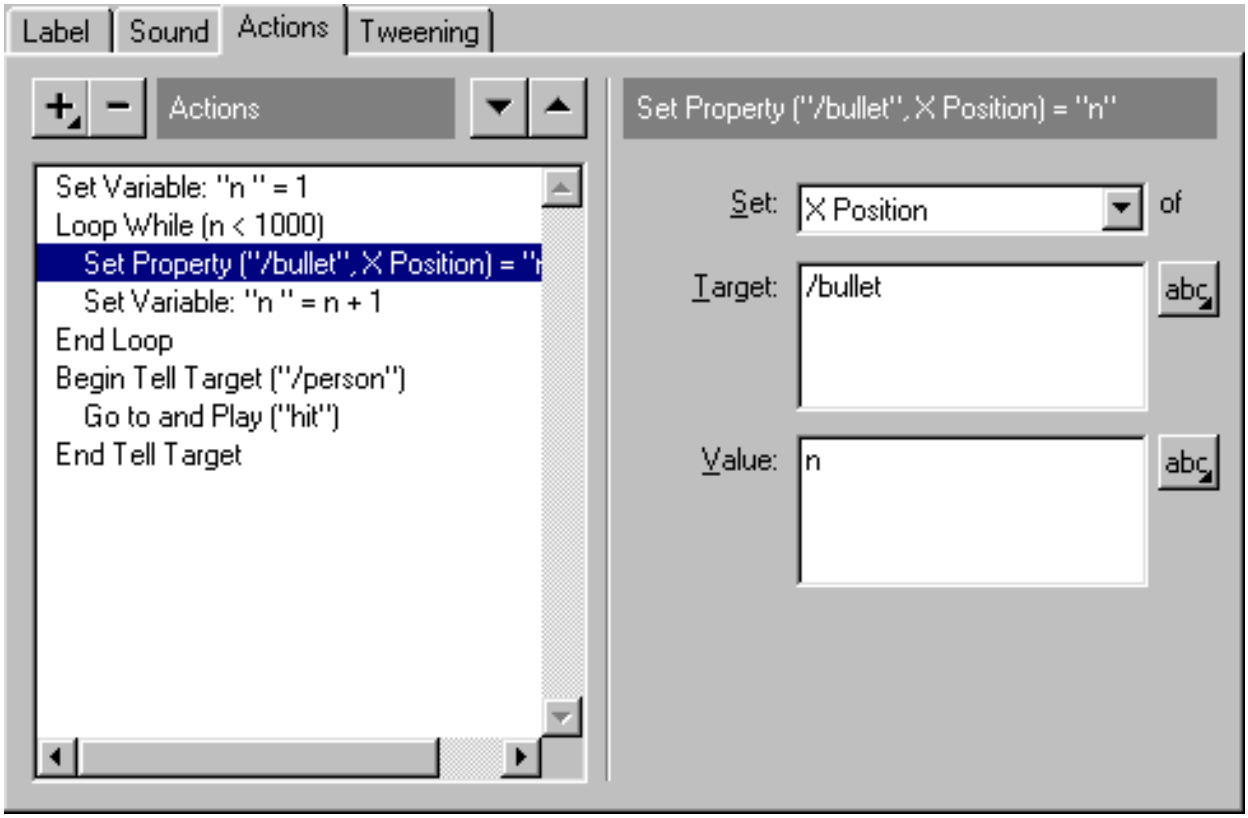
- **Y Scale** - escala um movie verticalmente, mais baixo se o valor for inferior a 100% e mais alto se for maior que 100%. (percentagem)
- **Alpha** - define a transparência, invisível (0%), opaco (100%). Mantém a interactividade (percentagem)
- **Visibility** - tem 2 estados on ou off (1 ou 0). Quando 0 o objecto perde a interactividade.
- **Rotation** - em graus, se negativos a rotação é contrária ao sentido dos ponteiros do relógio.
- **Name** - permite nomear e renomear um movie clip dinamicamente.
- **High Quality** - define a qualidade de um filme. (0 - baixa, 1 - boa, 2 - melhor)
- **Show Focus Rectangle** - Mostra um rectângulo amarelo a volta dos botões, permite a navegação através da tecla tab. (0 - invisível, >=1 - visível)
- **Sound Buffer Time** - Numero de segundos desde que um som começa a carregar até que começa a tocar.

Exemplo Básico:

Set Property ("/lightbulb", visibility) = 1

Exemplo Avançado:

```
Set Variable: "n" = 1
Loop While (n < 1000)
    Set Property ("/bullet", X Position) = n
    Set Variable: "n" = n + 1
End Loop
Begin Tell Target ("/person")
    Goto and Play ("hit")
End Tell Target
```



Action: Set Variable

Sintaxe:

Set Variable: "[VariableName]"=[Value]

Parâmetros:

VariableName - Nome da nova variável.

Value - Valor da variável, pode ser uma string, ou uma expressão matemática.

Exemplos de uso:

Passar variáveis globais para locais ou vice-versa.

Descrição:

Esta é a acção fundamental em actionscripting. Permite manipular informação entre variáveis. O valor de uma variável pode ser exibido e esta função permite atribuir-lhe qualquer conteúdo.

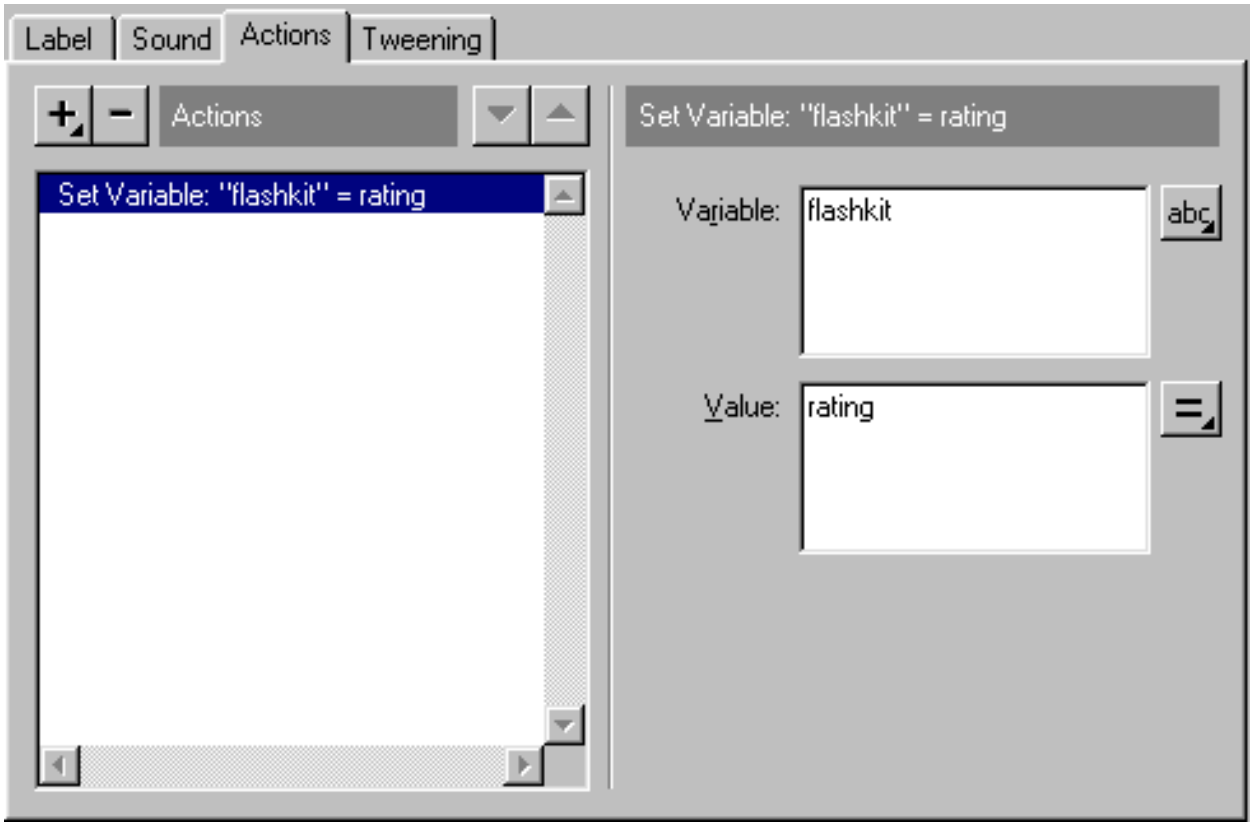
Exemplo Básico:

Set Variable: "number" = 1

Exemplo Avançado:

```
Set Variable: "n" = 1
Loop While (n < 5)
    Set Variable: "textfield" = "Flashkit is cool"
```

```
    Set Variable: "n" = n + 1
End Loop
```



Action: Stop

Sintaxe:

Stop

Exemplos de uso:

Usado em botões start/stop, quando se pretende parar um som, parar um movie num ponto qualquer, em suma sempre que se pretende parar uma timeline.

Descrição:

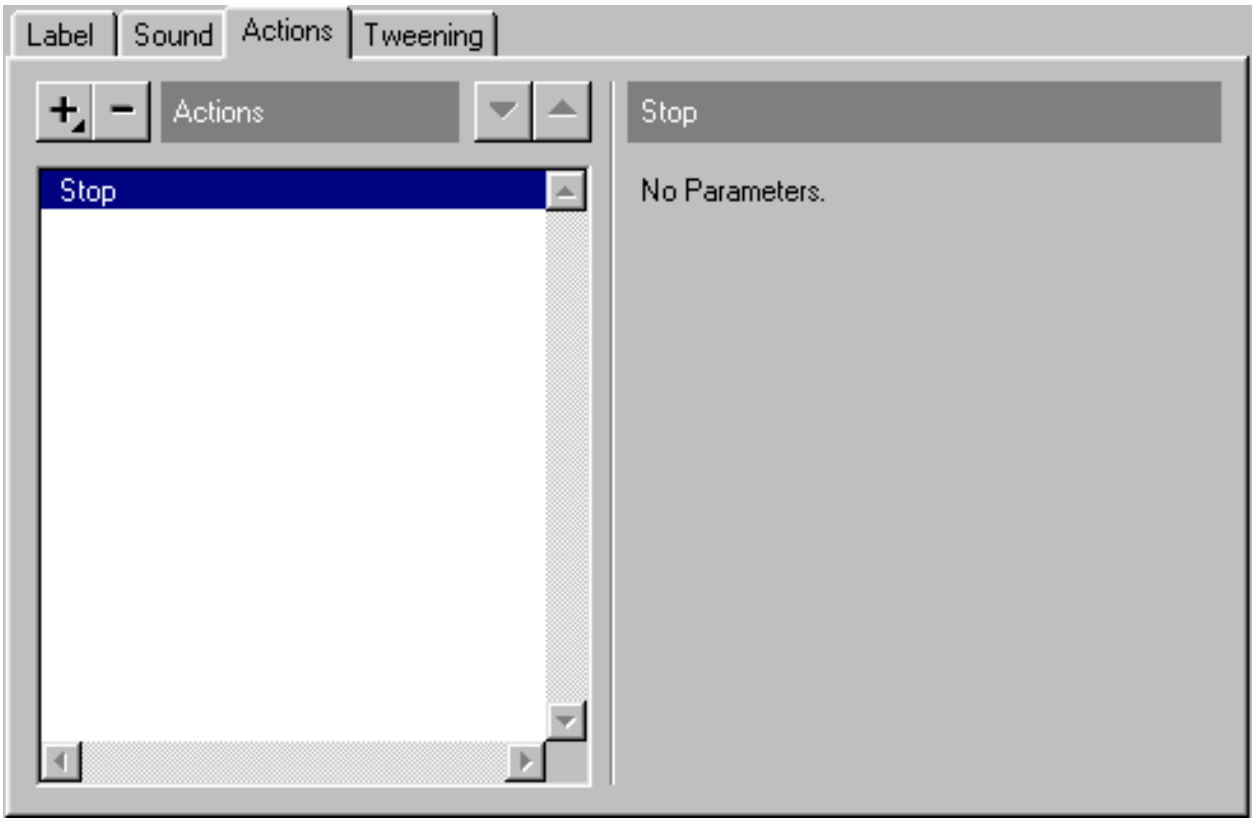
Chama um movie e pára o seu progresso no ponto em que estiver. Assim o movie terá que esperar por uma das acções Play ou Goto and Play para continuar a timeline.

Exemplo Básico:

```
On (Press)
    Stop
End On
```

Exemplo Avançado:

```
On (Press)
    Begin Tell Target ("car")
        Stop
    End Tell Target
End On
```



Action: Tell Target

Sintaxe:

```
Begin Tell Target (["Target"])
    [Actions]
End Tell Target
```

Parâmetros:

Target - o movie que se pretende controlar.

Actions - as acções a serem executadas no target.

Exemplos de uso:

Em qualquer parte!!

Descrição:

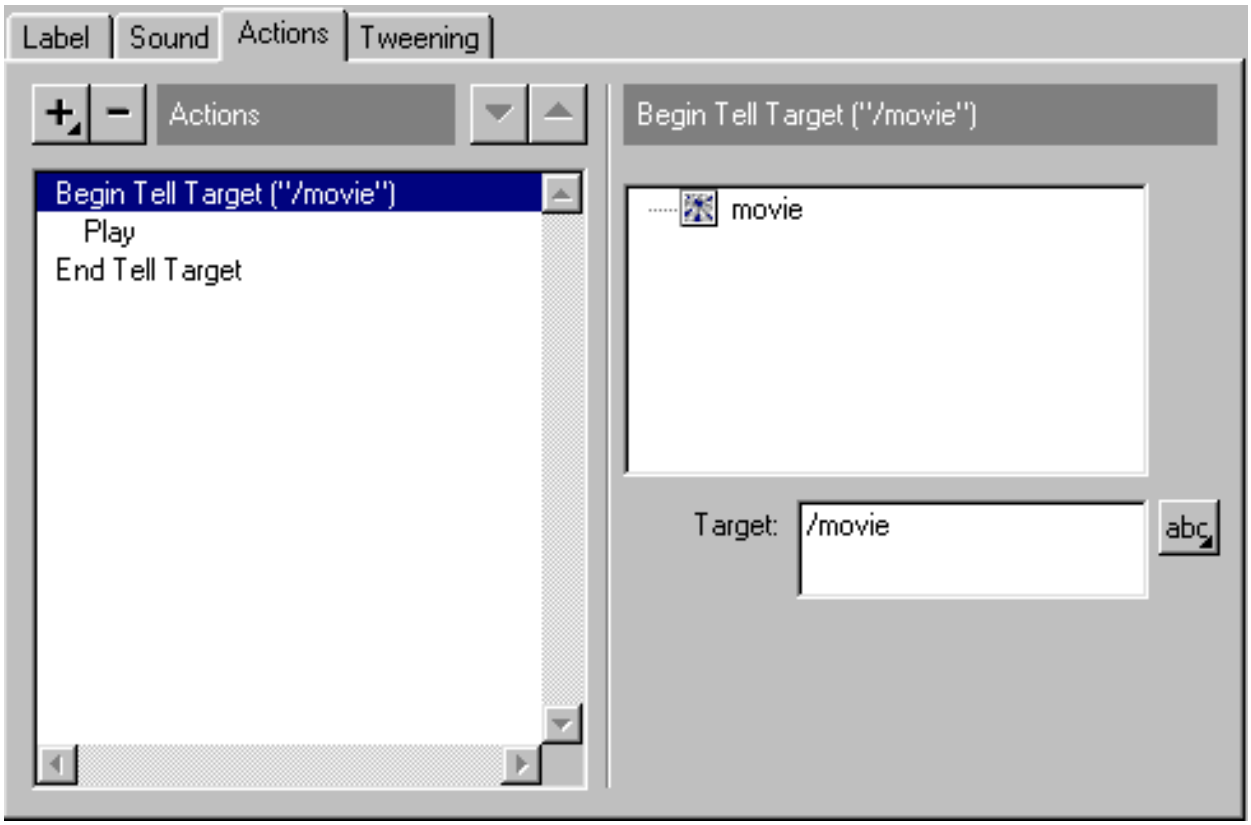
A acção Tell Target serve para chamar outra timeline (o target) e nela operar as acções que escrevemos dentro do Tell Target. Esta acção é de extrema importância, pois permite controlar uma timeline apartir de outra.

Exemplo Básico:

```
On (Release)
      Begin Tell Target ("/song")
            Play
      End Tell Target
End On
```

Exemplo Avançado:

```
On (Release)
     Begin Tell Target ("/song"&n)
           Play
     End Tell Target
End On
```



Action: Toggle High Quality

Sintaxe:

Toggle High Quality

Exemplos de uso:

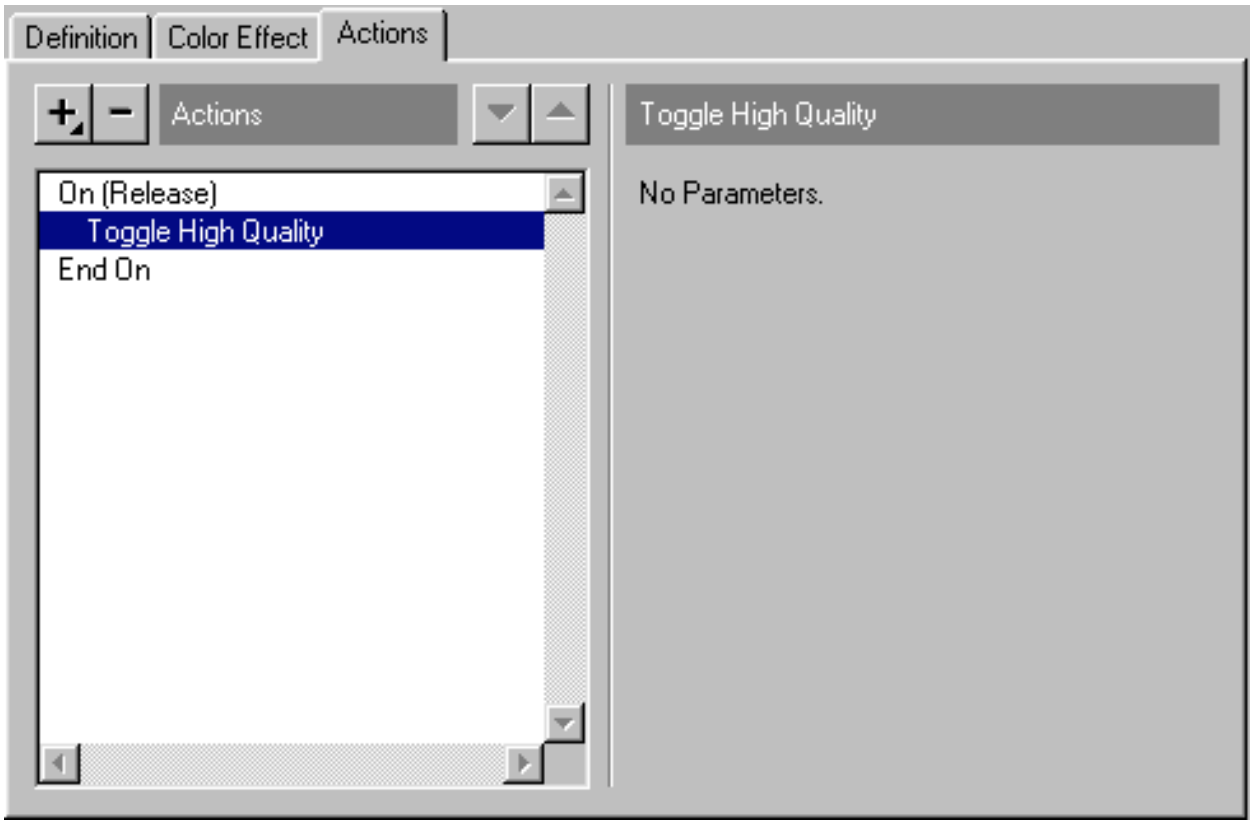
Minorar o impacto de animações pesadas no CPU.

Descrição:

Esta acção afecta todo o movie e dá a possibilidade de controlar a relação qualidade e velocidade. O que no fundo faz é ligar o antialiasing se antes estiver ligado e vice versa, o que faz com que a qualidade e a velocidade do filme sejam alterados. Quando desligado a qualidade visual é menor mas em computadores antigos a fluídez do filme é melhor. Quando ligado a qualidade é maior mas as animações são mais lentas.

Exemplo Básico:

```
On (Release)
    Toggle High Quality
End On
```



Action: Trace

Sintaxe:

Trace ("[Message]" & [Variable])

Parâmetros:

Message - Qualquer texto. Por vezes serve para dar significado à variável que vai ser exibida.

Variable - Variável que vai ser testada.

Exemplos de uso:

Testar filmes flash.

Descrição:

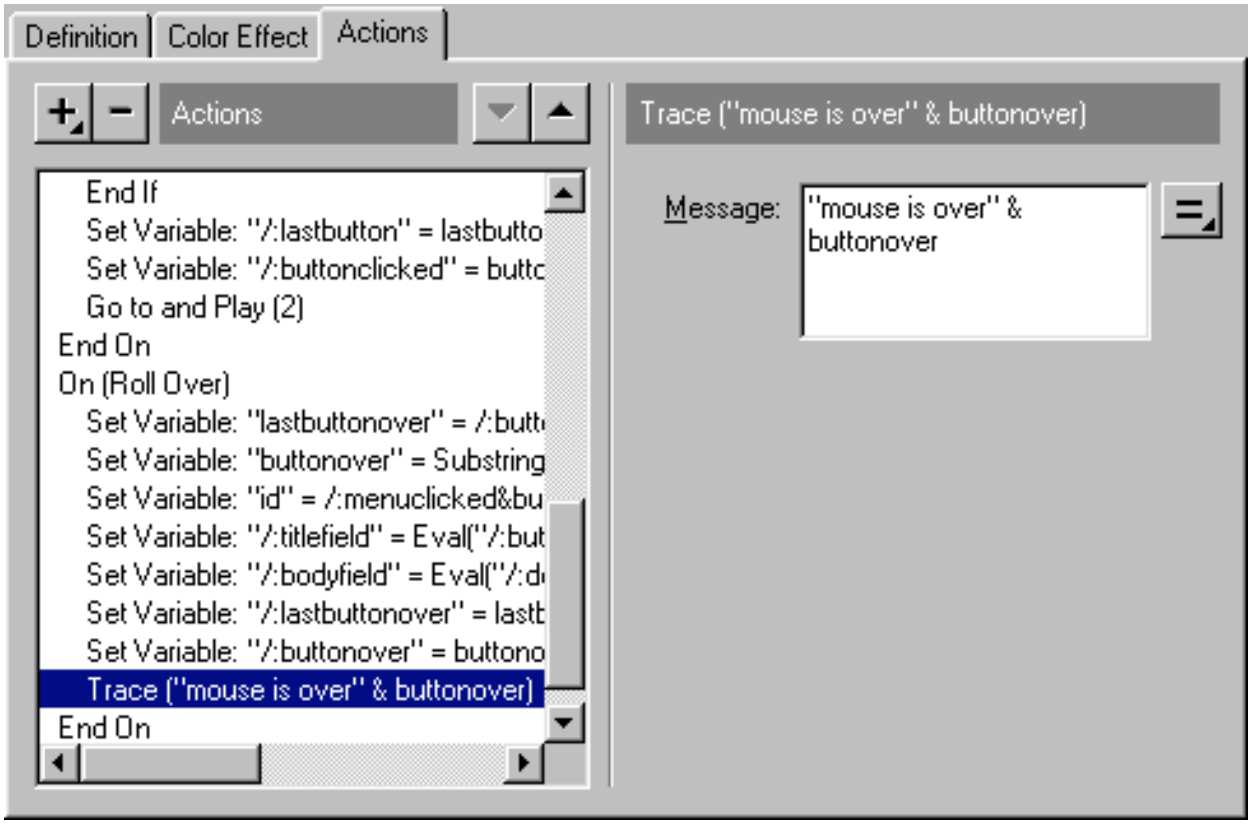
Quando testamos um filme e esta acção é lida, é lançada uma janela de output com o texto e ou as variáveis que foram introduzidas na acção. Assim se não aparecer nenhum texto é porque a nossa acção trace não foi lida. Podemos colocar quantos traces quisermos pois nem o tamanho do filme aumenta nem aparece nada no filme final.

Exemplo Básico:

Trace ("Now inside Loop")

Exemplo Básico:

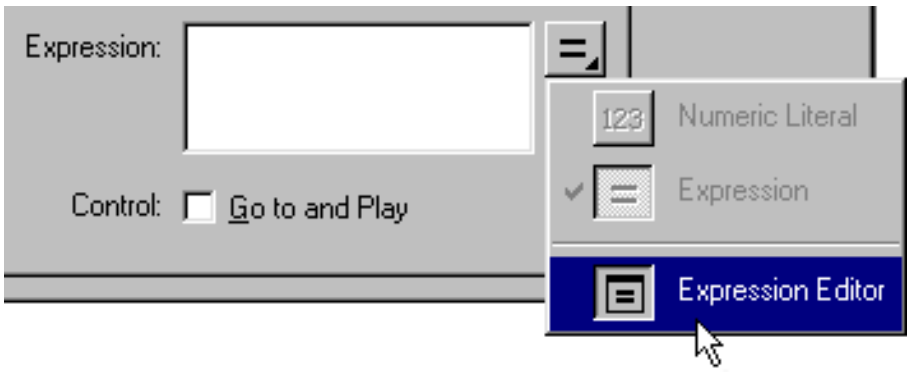
Trace ("mouse is over button" & buttonnumber)



Programação para Designers - Editor de Expressões

1. Introdução

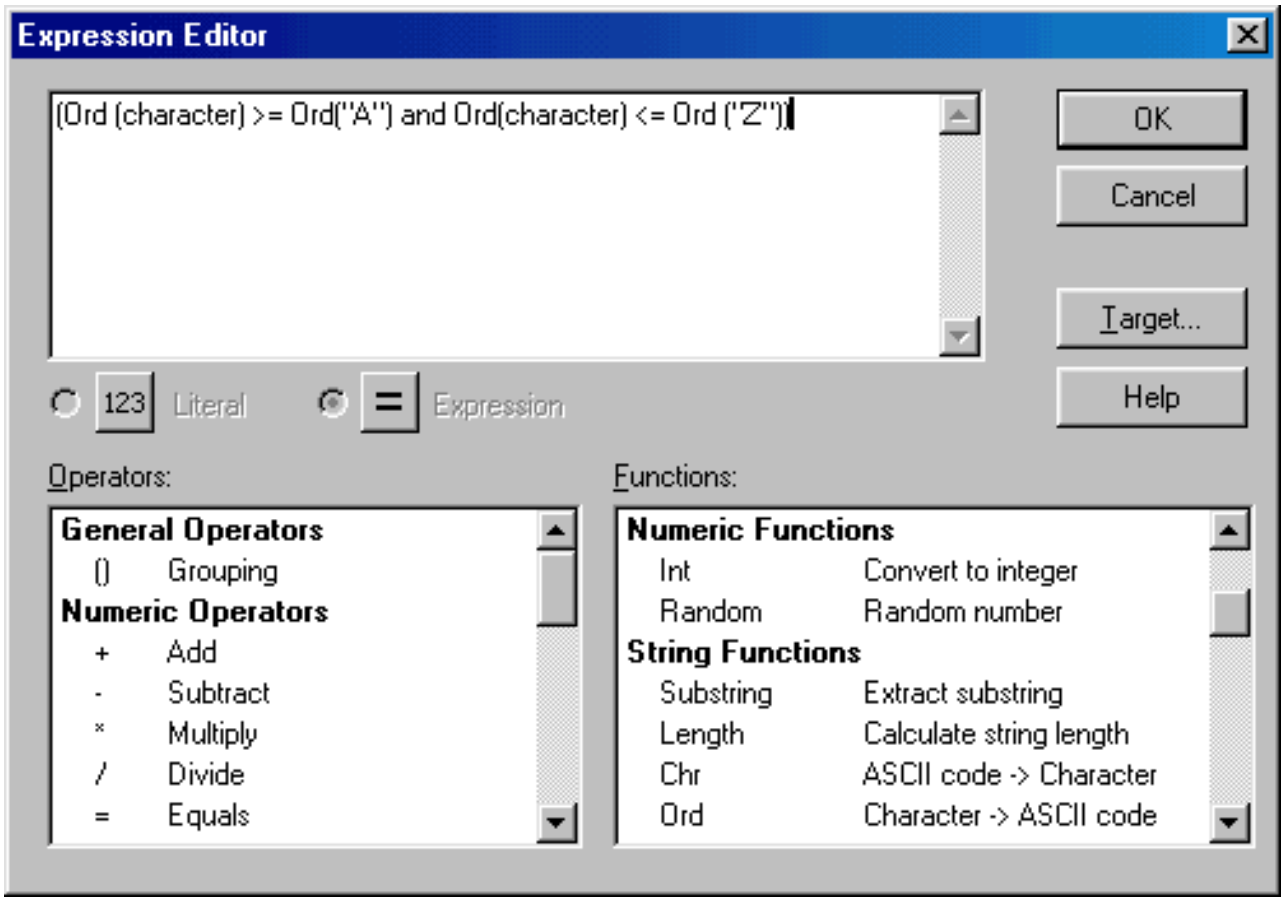
O editor de expressões é uma ferramenta simples que permite produzir complexas expressões e ou funções com a sintaxe correcta. Para abrir o editor clique no botão do lado direito da caixa das expressões.



Seleccionar uma das opções na janela que aparece abaixo do botão, para definir o tipo de informação da caixa texto. "String Literal" ou "Numeric Literal" definem se a informação na caixa vai ser string ou numero. Seleccionar a opção expressão para definir que a informação na caixa vai ser uma expressão. É aqui que se define o que tem aspas (string) e o que não tem (expressão).

2. O editor

Se seleccionar a opção de editor de expressões



É aqui onde se podem criar expressões. Volta a ter aqui as opções Literal e Expression. Para criar uma expressão basta um duplo clique numa das opções das listas dos Operators/Functions.A janela da esquerda contém todos os operadores e a da direita todas as funções.

Se for pedido o Target, basta um clique no botão do target para procurar por ele. Finalmente clicar OK e o target será introduzido na sua expressão, onde o rato estiver posicionado.

Agora é só pôr em prática tudo o que foi aprendido ao longo desta serie de tutorias denominados Programação para Designers. No tutorial final Pensar como o Inimigo vamos aprender a pensar como um programador.

!

Programação para Designers - Pensar como o Inimigo

1. Introdução

"Para derrotar o inimigo é preciso pensar como ele"

Actionscripting na perspectiva de um designer não é muito fácil porque vêm o mundo em função de assuntos, formas, conteúdos. Para ter sucesso como programador é necessário apanhar muitos calafrios (problemas de resolução complicada na perspectiva de um programador). É necessário identificar o problema (calafrio) para a seguir criar a solução passo a passo. Tudo isto é actionscript.

Lista de instruções passo a passo, o computador executa as instruções pela ordem que estão escritas, uma a seguir á outra. Esta lista pode ser simples ou muito complicado, como quisermos. Para uma instance o comando,

Go to and Play (1)

é uma instrução muito simples. Quando o player chega á frame onde a instrução está, executa a instrução e salta para a primeira frame do movie. Programar é encontrar um método que resolva o nosso problema, o melhor é quebrá-lo em subprogramas e resolve-los individualmente, após isto temos o nosso problema principal resolvido.

> **Definição:** Programadores chamam ás instruções passo a passo "Algoritmos". Os algoritmos são usados para criar as bases para um programa de computador.

Neste tutorial iremos trabalhar com um exemplo da vida real para conseguir-mos entender como um programador pensa. Então, vamos resolver um problema de actionscript. No fim deste tutorial de certo entendemos melhor como se desenvolve actionscript! Nas próximas páginas vamos "

Download

2. Exemplo da Vida Real

Pegar num exemplo da vida real - Comprar Leite.

1.Encontrar as chaves e a carteira
2.Conduzir até ao supermercado
3.Encontrar o leite
4.Comprar o leite
5.Conduzir para casa

Agora que foram encontrados os maiores subproblemas, refinar e resolver cada um deles. Quando cada um deles estiver resolvido o problema inteiro está resolvido. Por exemplo passo 4:

4. Comprar o leite
4.1 Procurar o preço
4.2 Subtrair o valor á carteira

Vamos resolver um problema em actionscript - Converter uma string em minúsculas

3. Arquivo SWF

Isto é o aspecto final do actionscript que vamos criar
4. Identificação do Algoritmo

Antes de fazer qualquer há que identificar o ambiente ou contexto em que o script vai operar. Neste exemplo há uma text field "input" que permite introduzir a "string". Há outra text field "output" para mostrar o resultado. Agora olhar para o problema e partir em subproblemas.

Converter em minúsculas
1. Inicializar variáveis (e text fields)
2. Encontrar (primeira) letra
3. Converter em minúsculas se (IF) necessário
4. Adicionar a letra convertida á variável "output"
5. Preparar para procurar a próxima letra

Uma consideração que possivelmente já questionou, "Como vamos preparar a pesquisa da próxima letra?" Temos que introduzir uma string e a tarefa é converter cada letra, se (IF) necessário, em minúsculas. Significa isto que cada letra tem que ser corrida uma a seguir a outra ordenadamente. O pouco de senso comum e experiência diz esta é a técnica para fazer isto é usando uma variável contadora. Esta variável pode ser incrementada para permitir extrair e mover o caracter #1 para #2 para #3 e assim por diante.

Outro aspecto que poderá já ter reparado, é que o passo 2 e 5 são praticamente a mesma coisa para cada letra. A única coisa que muda é a letra a ser convertida. Assim a experiência indica que este é o lugar perfeito para usar a acção Loop. Então adiciona mais este passo para o fundo da tua lista.

5. Refinar Algoritmo

Agora olhar para cada sub problema e verificar se é possível refinar mais. Um desses é o ponto 3, converter em minúsculas se (IF) necessário. Mas antes de avançar é necessário entender um pouco sobre caracteres. Cada caracter no teclado está associado a um valor que é conhecido como código ASCII. Por exemplo o código para a letra "a" é 97 e "A" é 65 (a diferença é 32, memorizar isto para mais tarde). Este código será usado para auxiliar a conversão. Vejamos agora o ponto 3 refinado subproblemas resolvidos.

3. Converter em minúsculas se (IF) necessário
3.1 Verificar se (IF) a letra está entre "A" e "Z"
3.2 se (IF) sim converter a letra em código ASCII
3.3 adiciona (add) 32 ao código (diferença numérica entre os códigos "a" e "A")
3.4 Converte o valor de ASCII para caracter

Agora que não é possível dividir em mais subprogramas o próximo passo é converter em código a lista de instruções.

6. Criar o Código

(a numeração seguinte é a do algoritmo)

#1.Inicialização de Variáveis

OK, há uma text field para o Output que tem que estar em branco e uma variável contadora que começa na primeira letra (#1) quando este código é executado. Então o código terá este aspecto:

Set Variable: "r" = 1
Set Variable: "output" = ""

#2.Encontrar (primeira) letra

Do nosso conhecimento de **Funções** sabemos que o modo de extrair caracteres de uma string é através da função **substring**. Também será melhor guardar o valor numa variável para usar mais tarde.

Set Variable: "character" = Substring (input, r, 1)

Os argumentos para esta função, da esquerda para a direita, o primeiro é a string onde se vai extrair os caracteres. Neste caso é a variável "input" que contém a string de entrada. O segundo argumento é posição onde se começa a extracção da subtring. O último argumento é o número de caracteres a extrair. Neste caso só precisamos de um.

#3.Converter em minúscula se (If) necessário

#3.1 Verificar se (IF) a letra está entre "A" e "Z"

Tudo o que é necessário é sacar o código ASCII da letra e ter a certeza que se encontra entre "A" e "Z". Se não estiver não se faz nenhuma conversão e o programa continua na letra seguinte. A função **Ord** converte uma letra num número para permitir uma comparação numérica.

If (Ord(character) >= Ord("A") and Ord(character) <= Ord ("Z"))

#3.2 Se (IF) sim converter a letra em código ASCII

Aqui utiliza-se outra vez a função **Ord** mas desta vez para a transformar código ASCII e guardar numa variável temporária.

Set Variable: "upperCode" = Ord (character)

#3.3 Adiciona (add) 32 ao código (diferença numérica entre os códigos "a" e "A")

O próximo passo é converter o número ("upperCode") em código ASCII equivalente á letra minúscula somando 32 e armazenar este valor na variável temporária "lowerCode".
((a letra "A" = 65) + 32) = (97 = a letra "a")

Set Variable: "lowerCode" = upperCode + 32

#3.4 Converte o valor de ASCII para caracter

Neste passo tudo o que se vai fazer é converter o valor do código ASCII da minúscula (armazenado na variável "lowercase") para caracter com a função **Chr**.

Set Variable: "lowerCaseLetter" = Chr (lowerCode)

#4. Adicionar a letra convertida á variável "output"

Concatenar a variável output com a letra convertida.

Set Variable: "output" = output & lowerCaseLetter

#5. Preparar para procurar a próxima letra

Agora temos que preparar a próximo letra. Para isso usar a variável **r** utilizada na função substring. Lembrar que esta variável controla a posição a partir de onde a extracção da letra a avaliar é feita. Assim, após a primeira letra ser avaliada incrementar-mos 1 valor ao **r** ,função substring vai agora procurar a 2 letra

Set Variable: "r" = r+1

#6. Salta para o passo #2 até toda a string estar convertida

Enquanto no algoritmo este passo vem no fim, na actionscript vem no início. Esta função vai repetir todo o processo anterior letra á letra até elas acabarem. Se a variável contadora "**r**" for maior que o numero de letras da string armazenada na variável "input" então o programa acaba.

Loop While (r <= Length (input))

7. Colocar tudo junto

Agora, colocar tudo junto! Se nos lembrar-mos da lista de instruções, temos algo como isto.
(as linhas estão coloridas para ser possível relacionar o algoritmo e a actionscript)

1. Inicialização de Variáveis (e text fields)
2. Encontrar (primeira) letra
3. Converter em minúscula se (**If**) necessário
    3.1 Verificar se (IF) a letra está entre A e Z
    3.2 Se (IF) sim converter a letra em código ASCII
    3.3 Adiciona (add) 32 ao código
        (diferença numérica entre os códigos "a" e "A")
    3.4 Converte o valor de ASCII para caracter
4. Adicionar a letra convertida á variável "output"
5. Preparar para procurar a próxima letra
6. Salta para o passo #2 até toda a string estar convertida

Agora quando se junta todas as actionscripts deverá ter algo como isto.

```
Set Variable: "r" = 1
Set Variable: "output" = ""
Loop While (r <= Length (input))
   Set Variable: "character" = Substring (input,r, 1)
   If (Ord (character)>=Ord("A") and Ord(character)<=Ord ("Z"))
      Set Variable: "upperNumber" = Ord (character)
      Set Variable: "lowerNumber" = upperNumber + 32
      Set Variable: "character" = Chr (lowerNumber)
   End If
   Set Variable: "output" = output & character
   Set Variable: "r" = r+1
End Loop
```

Programação para Designers - Variáveis

Este tutorial faz parte de um grupo de tutoriais (Programação para Designers) cujo objectivo é ensinar actionscripting a quem não tem bases de programação.

1. O que é uma Variável?

Numa linguagem de programação é necessário várias coisas:

1. Dados
2. Modos de avaliar dados
3. Modos de manipular dados

Isto existe em linguagem de programação com:

1. Variáveis
2. Operadores
3. Funções / Expressões

Neste tutorial vamos falar de variáveis, o restante fica para mais tarde. Uma variável é um simples recipiente, um balde, se quisermos. No entanto este balde só consegue suportar uma porção de informação de cada vez, não interessa o tamanho, só uma porção. Para entender vejamos as seguintes variáveis:

Nome = "Zé"
Idade = 25
Renda = "insuficiente"
Amigos = 0

Cada linha pode ser quebrada em 3 partes, o nome da variável, o operador e o valor.

Nome da Variável

O nome da variável pode ser qualquer letra, numero ou underscore e não é case sensitive mas tem que começar com um caracter. Estes nomes devem ter algum significado para nós. Enquanto o valor da variável muda o seu nome nunca.

Download

2. Atribuição de valores

O valor atribuído a uma variável pode ser de quatro tipos, number (numero), boolean (valor lógico), string (texto) ou nada.

Number - Qualquer numero (ex: 12, -54, 52387)
Set Variable: "myvariable" = 12

Boolean - True ou False
Set Variable: "myvariable" = True

String - Qualquer numero ou letra apresentado dentro de aspas (ex: "Olá" ou "12")
Set Variable: "myvariable" = "Avenida da Liberdade n.º23"

Nothing - Usado pelos actionscripters para definir uma variavel vazia, sem string. (ex: "")
Set Variable: "myvariable" = ""

Estes tipos são facilmente entendidos, mas temos que ter em atenção que cada um deles tem tratamentos diferentes em actionscripting. Um número é fácil, é um simples número. Uma string é a maneira dos programadores descrever texto. De notar que no exemplo acima a string contém um numero, quando dentro de aspas um numero é uma string.

Uma variável do tipo Boolean é utilizado em actions que contém condições tal como if ou loop. Quando se usa o tipo boolean o resultado só pode ser true ou false. Alguns programadores não utilizam estas palavras mas sim os números 0 para indicar false e qualquer número diferente de 0, normalmente o 1, para representar true.

Nothing representa o vazio de texto (string). Em flash muitos usam isto para testar se uma variável foi alterada numa função loop ou para ver se a ultima variável num arquivo de texto foi carregada.

Loop While (eof < 1)
...
If ("button"&n = "")

Set Variable: "eof" = 1
End If
...
End Loop

3. Preencher Variáveis

Um dos principais erros em actionscripting são as aspas " ". Porquê? Porque "myvariable" é completamente diferente disto - myvariable. Porque com aspas é uma string e sem é uma expressão.

Quando igualamos uma variável a uma string o que estamos a fazer é simplesmente guardar na variável o que está dentro das aspas, por exemplo

Set Variable: "myvariable" = "counter"

Isto não é passar o valor da variável counter para a variável myvariable, mas sim passar o texto (string) counter. O grande poder do actionscripting é a capacidade de associar expressões a variáveis. Uma expressão pode ser uma ou mais variáveis, números ou texto (string). Agora vamos observar os seguintes exemplos com variáveis e expressões: (variáveis a verde e expressões a vermelho - não esquecer que as expressões podem conter variáveis)

Set Variable: "myvariable" = othervariable

Set Variable: "Counter" = Counter + 1

Set Variable: "subtotal_one" = 20
Set Variable: "subtotal_two" = 45
Set Variable: "total" = subtotal_one + subtotal_two

Set Variable: "city" = "Sydney"
Set Variable: "country" = "Australia"
Set Variable: "home" = city & country

Nota: O símbolo & representa concatenação, ou seja junta o que estiver antes e depois do símbolo.

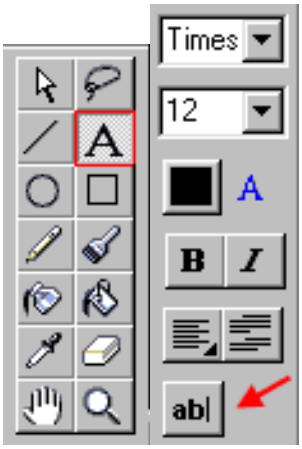
4. Variável Text Field

É outro tipo de variável que deverá ser bem aprendida - Text Field. Isto é uma variável quer acreditemos ou não, e devemos tratá-la como qualquer outra variável. A única diferença é que através da text field podemos inserir ou visualizar o valor da variável. Vejamos:

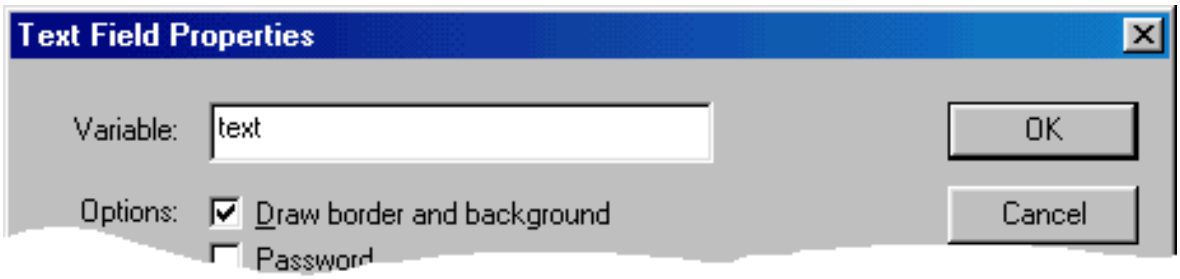
Set Variable: "text" = "Hello"

A variável "text" é nome da Text Field.Vamos ver como se configura e se usa isto.

Primeiro é necessário criar uma Text Field. Isto faz-se clicando na ferramenta texto na barra de ferramentas.



Agora desenhar a text field. No fim clicar com o botão direito em cima da text field e seleccionar "properties". É exibido uma caixa de diálogo, aqui pode-se alterar as propriedades da text field. O comando Set Variable no exemplo acima usava a variável "text", colocar este nome no espaço reservado á variável



Agora, sempre que se altera o valor da variável "text" aparecerá na text field.

Já sabemos trabalhar com variáveis, para continuar a aprender actionscripting aconselhamos o tutorial - **Expressões**

Programação para Designers - Expressões

Este tutorial faz parte de um grupo de tutoriais (Programação para Designers) cujo objectivo é ensinar actionscripting a quem não tem bases de programação.

Escrever Expressões

Uma expressão pode ser uma variável, texto, numero e operador ou alguma combinação dos itens anteriores. O mais importante a reter é que de uma expressão o resultado é um sempre valor, podendo este ser string, numero ou boleano.

Em actionscript uma expressão tem muitas utilizações. Uma das mais obvias é a atribuição de um valor a uma variável. No exemplo a seguir estamos a atribuir o valor 2 à variável "Contador":

(expressões a vermelho)

Set Variable: "Contador" = 2

Outro exemplo é o uso de expressão para evoluir uma variable. Temos uma variável "primeiro_nome" com o valor "Zé". A variável "parabens" evolui assim para "Bem Vindo Zé,

ao meu site":

```
Set Variable: "primeiro_nome" = "Zé"
Set Variable: "parabéns" = "Bem Vindo" & primeiro_nome & ", ao meu site."
```

Aqui usamos uma expressão para calcular a frame para onde o filme deve saltar:

```
Set Variable: "frame" = 3
Go To and Play (frame + 6)
```

Finalmente este exemplo é um script de acesso. Este script tem como output "Accesso Rejeitado" porque a variável "lastname" é igual a "Fagunde" e não "Fagundes".

```
Set Variable: "firstname" = "Zé"
Set Variable: "lastname" = "Fagunde"
If ((firstname = "Zé") and (lastname = "Fagundes"))
    Set Variable: "output" = "Olá Sr" & lastname
    Begin Tell Target ("/membersarea")
        Go to and play("enter")
    End Tell Target
Else
    Set Variable: "output" = "Accesso Rejeitado"
    Go to and play("start")
End If
```

Quando trabalhamos com expressões temos sempre que saber qual o tipo de valor que queremos, uma string, um número ou um valor booleano.

Relembrar o que foi referido no tutorial **Variáveis** acerca das aspas, porque as aspas ajudam a determinar qual o tipo da expressão. Observar os exemplos seguintes e descobrir as diferenças:

```
Set Variable: "myvariable" = "450"    [a string]
Set Variable: "myvariable" = 450      [a number]
Set Variable: "myvariable" = True     [a boolean value]
Set Variable: "myvariable" = "True"   [a string]
Go to and Play(myvariable)            [go to and play frame 450]
Go to and Play("myvariable")          [go to and play label myvariable]
```

Já sabemos trabalhar com expressões, para continuar a aprender actionscripting aconselhamos o tutorial - **Operadores**

Programação para Designers - Operadores

Este tutorial faz parte de um grupo de tutoriais (Programação para Designers) cujo objectivo é ensinar actionscripting a quem não tem bases de programação.

1. Introdução

Operadores determinam como a informação de uma variável deve ser avaliada numa expressão.

Há vários tipos de operadores diferentes - aritméticos, comparativos e lógicos. Cada um tem o seu objectivo. Operadores aritméticos são usados para operações matemáticas. Operadores Comparativos usam-se para comparar expressões. Finalmente os Operadores Lógicos permitem criar complexas expressões lógicas.

O tipo de dados (numérico, string ou boleano) determinam o tipo de operador a usar. Seguem-se alguns pequenos exemplos.

```
Set Variable: "fname" = "Fred"
Set Variable: "lname" = "Wahid"
Set Variable: "salary" = 500
Set Variable: "salary" = 500
Set Variable: "overtime" = 120

Set Variable: "pay" = salary + overtime

If (overtime > 300)
    Go to and Play("toomuch")
End If

If (fname ne "")
    Set Variable: "fullname" = fname & " " & lname
End If
```

2. Operadores Numéricos

Operadores Numéricos são muito simples. Há dois tipos - Aritméticos e Comparativos que só podem ser usados com Números. Portanto se forem usadas strings ocorrerão alguns erros.

Operadores Aritméticos

+	Soma dois números
-	Subtrai um número a outro
/	Divido um número por outro
*	Multiplica dois números

Exemplos:

```
Set Variable: "units" = 5
Set Variable: "cost" = 10
Set Variable: "price" = 20

Set Variable: "profit" = (price - cost)*5
```

Operadores Comparativos

=	Igual

<>	Diferente
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a

Exemplos:

```
If (input < number)
    Set Variable: "status" = "guess a higher number"
Else If (input > number)
    Set Variable: "status" = "guess a lower number"
Else If (input = number)
    Set Variable: "status" = "Correct!!"
    Go to and Play("end")
End If
```

3. Operadores String

Operadores string são usados para avaliar e comparar strings. A aplicação prática para estes operadores é procurar e exibir nomes. Cuidado, não misturar tipos de dados quando se avaliam expressões, esta é uma das razões de muitos erros.

Operadores String

""	String
&	Concatenar
eq	Igual
ne	Diferente
gt	Maior que
lt	Menor que
le	Menor ou igual a
ge	Maior ou igual a

O primeiro operador **""** (string) sempre que usado numa expressão indica que o que está dentro de aspas é uma string de caracteres.

O símbolo **&** é a concatenação. Usado quando se pretende adicionar uma string, variável ou função no fim de outra string, variável ou função. É importante salientar que muitos números numa expressão string são automaticamente convertidos em string. Por ex:

```
Set Variable: "idade" = 5
Set Variable: "mensagem" = "Eu tenho " & idade & " anos de idade"
```

O output da variável mensagem é "Eu tenho 5 anos de idade".

O resto dos operadores (**eq**, **ne**, **gt**, **lt**, **le**, **ge**) são chamados operadores comparativos porque são usados para comparar strings ou variáveis numa string. A sua função principal é determinar a ordem alfabética a partir da primeira letra da string. Não quer isto dizer que não é possível comparar as segundas e terceiras letras, só requer um pouco mais de código (ver substring function).

Nota Importante: Strings são case sensitive, "Hi" não é igual a "hi" e as letras minúsculas são maiores que a maiúsculas. Isto é um pouco confuso, mas tem tudo a ver com a tabela ASCII, o código da letra 'A' é 65 e o código para a letra 'a' é 97.

Aqui estão alguns exemplos de operadores string:

```
Set Variable: "input" = "john"
Set Variable: "accountname" = "joe"

If (input eq accountname)
    Set Variable: "output" = "Hello, " & accountname & " welcome back"
Else
    Set Variable: "output" = "User not recognised"
End If
```

4. Operador Lógico AND

Operadores Lógicos permitem aumentar a complexidade das expressões.

Operadores Lógicos

and	e
not	não
or	ou

Operador And

O operador lógico AND é usado quando se quer que um evento ocorra quando duas condições são simultaneamente verdadeiras. Vejamos um exemplo:

```
On (Release)
    If ((primeiro_nome eq "ze") and (password eq "qwerty"))
        Set Variable: "status" = "Acesso aceite"
        Go to and Play ("membersonly")
    Else
        Set Variable: "status" = "Acesso rejeitado"
    End If
End On
```

Neste script o acesso só é aceite quando as duas condições (primeiro_nome eq "ze") e (password eq "qwerty") são verdadeiras.
Abaixo está criada uma tabela de verdade que mostra todas as hipóteses existentes. Observe-se que apenas com as duas condições verdadeiras o resultado com o operador AND é verdadeiro.

Condição 1	**Condição 2**	**1 AND 2**
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Falso	Falso
Falso	Verdadeiro	Falso
Falso	Falso	Falso

5. Operador Lógico OR

O operador lógico OR é usado quando se quer que um evento ocorra quando pelo menos uma das condições é verdadeira. Vejamos um exemplo:

```
On (Release)
   If ((cargo eq "webmaster") or (username eq "truques e dicas"))
      Set Variable: "saudacao" = "Olá Zé"
   Else
      Set Variable: "saudacao" = "Tu não és o Zé"
   End If
```

Analisemos a actionscipt. Enquanto o operador AND só uma hipótese era verdadeira com o operador OR há três possibilidades. Primeiro, se o valor da variável "cargo" for igual a "webmaster". Segundo, se a variável username for igual a "truques e dicas" e terceiro se o "cargo" for igual "webmaster" e "username" igual "truques e dicas". A única vez que este operador avalia a condição como falsa é quando as variáveis não são iguais nem a "webmaster" nem a "truques e dicas" respectivamente. Ver a tabela de verdade.

Condição 1	**Condição 2**	**1 OR 2**
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Falso	Verdadeiro
Falso	Verdadeiro	Verdadeiro
Falso	Falso	Falso

6. Operador Lógico OR

O uso do operador NOT trabalha de uma forma diferente dos anteriores. Em vez do seu resultado ser em função de duas condições aqui é apenas uma condição utilizada. O resultado da aplicação deste operador é a negação do resultado da condição associada. Quer isto dizaer que se temos o operador associado a uma condição falsa o resulta é verdadeiro.

Por exemplo, se a variável "on" tiver o valor true (ou 1) a declaração NOT(on) será avaliada como false (ou 0). No exemplo abaixo se o valor de "on" for 1 (Set Variable: "on"=1) o output do script é "not on" e vice-versa.

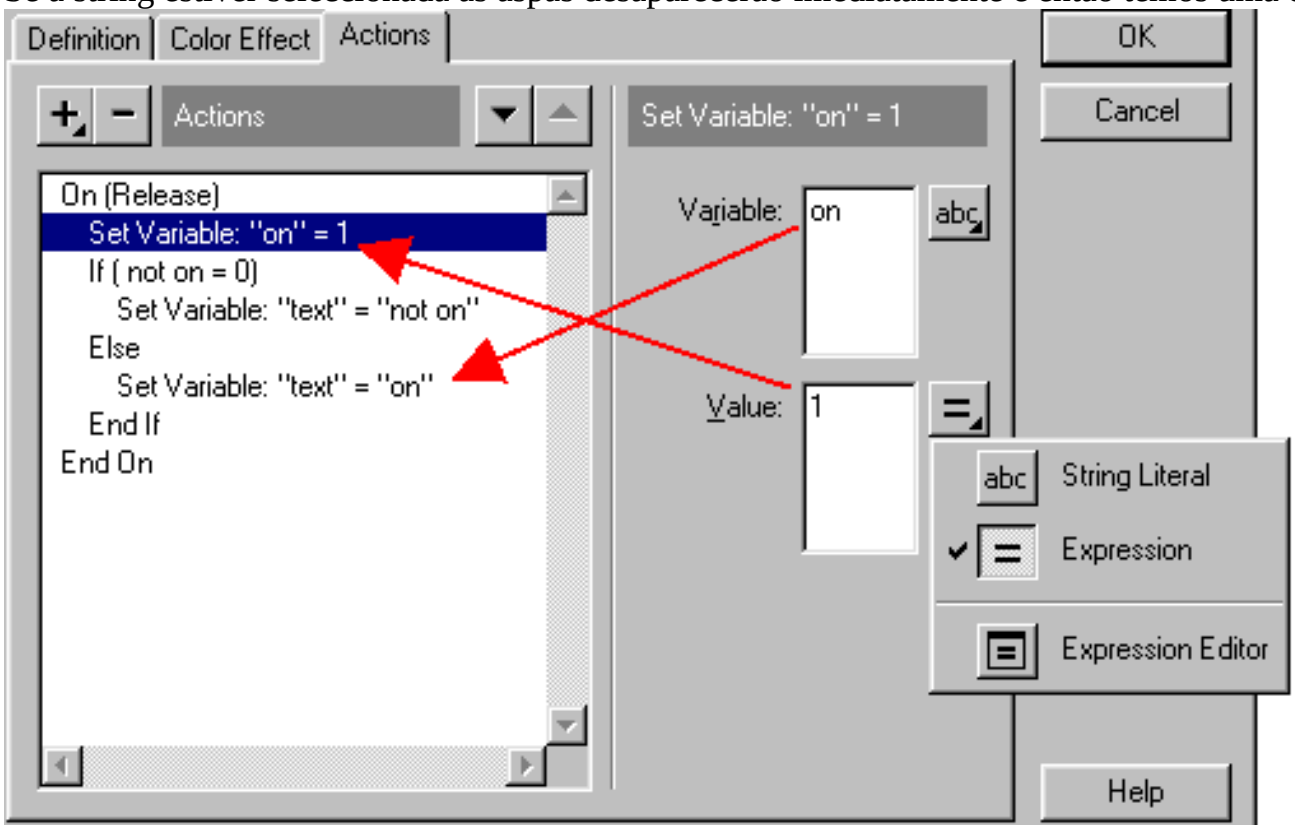
```
On (Release)
   If ( not on = 0)
     Set Variable: "output" = "not on"
   Else
     Set Variable: "output" = "on"
   End If
End On
```

Ver tabela de verdade:

Condição 1	**NOT Condição 1**
Verdadeiro	Falso
Falso	Verdadeiro

7. Apontamentos Finais

Se um valor tiver entre aspas então temos uma string e então é exigido um operador de strings. Para mudar entre uma string e uma expressão clicar no botão ao lado da textfield Value e escolher Expression. Se a string estiver seleccionada as aspas desaparecerão imediatamente e então temos uma expressão.



Já sabemos trabalhar com operadores, para continuar a aprender actionscripting aconselhamos o tutorial - **Funções**

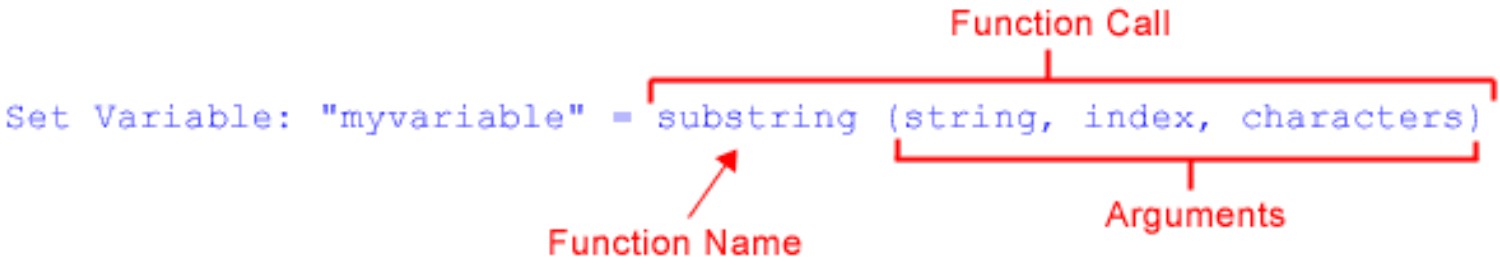
Programação para Designers - Funções

Este tutorial faz parte de um grupo de tutoriais (Programação para Designers) cujo objectivo é ensinar actionscripting a quem não tem bases de programação.

1. Definição da Função

As funções de actionscript do Flash 4 são pré-definidas e têm como utilidade executar tarefas de programação e criar complexos scripts. A melhor maneira de ver uma função é pensarmos numa máquina em que há uma entrada, um processamento e uma saída. Muitas funções permitem interagir com variáveis e seus conteúdos. Por exemplo a função "Lenght" permite contar quantos caracteres tem uma string. Isto pode parecer simplista mas esta função quando usada num script pode por exemplo validar o correcto numero de dígitos de um numero de telefone que está a ser submetido. A animação seguinte ajuda a perceber o funcionamento das funções, neste caso é a função lenght.

Uma função tem várias partes, the function call (a chamada da função), the function name (nome da função) e the arguments (os argumentos). A chamada da função é fácil, é a função toda. O nome da função diz ao computador o que tem a fazer e quais os argumentos que deve procurar. Os argumentos são a chave, eles é que tem a informação necessária para o computador correr a função. A seguir está um esquema que mostra a sintaxe da função substring.



2. Exemplo de Trabalho

Considerar novamente a função "Lenght". A Sintaxe correcta é a seguinte:

Length (string)

O nome da função é "Lenght". Quando o computador lê isto já sabe que só há uma informação (o argumento) que será o que está dentro de parênteses. O argumento pode ser uma de duas coisas, uma string ou uma variável contendo uma string. Como aprendemos anteriormente uma string é representada entre aspas e o nome de uma variável não.

Vejamos agora a função "Length" usando os dois tipos de argumento:

Set Variable: "stringlength" = Length ("myvariable")

O resultado é o numero 10

Set Variable: "othervariable" = "Hello"
Set Variable: "stringlength" = Length (othervariable)

O resultado do segundo exemplo é o numero 5. É assim que uma função trabalha, primeiro identifica qual a é função para saber como deve agir, neste caso "Lenght". Então vai tirar informação que foi dada no campo (string), conta o numero dos caracteres na string e o resultado da função é um valor numérico que vai ficar armazenado na variável "stringlenght".

3. Funções e Expressões

Uma função pode ser usada em qualquer sitio numa actionscript e não deve ser limitado por este tipo de exemplo. Uma Função é por vezes uma expressão porque avalia um valor o qual pode ser utilizado mais tarde numa actionscript. Vejamos agora uma função a ser usada como expressão.

```
If (Substring (firstname, 1, 1) eq "E")
   Set Variable: "output" = "Your name begins with E"
Else If (Substring (firstname, 1, 1) eq "e")
   Set Variable: "output" = "Your name begins with E"
Else
   Set Variable: "output" = "Your name does not begin with E"
End If
```

Agora vamos ver o arsenal de funções que a Macromedia criou para nós no Flash4

4. Arsenal de Funções - Eval

Sintaxe: Eval (Variavel)

Permite avaliar variáveis e expressões em tempo real. Por exemplo:

Set Variable: "name6" = "George"
Set Variable: "counter" = 6
Set Variable: "outputName" = Eval ("name" & counter)

Se a variável "counter" tiver o valor 6 então a função eval avalia a variável "name6". Assim a variável "outputName" fica com o valor "George". Este método de avaliar variáveis evita o uso de ciclos e avalia múltiplas variáveis rapidamente.

5. Arsenal de Funções - True

Sintaxe: True

Esta função atribui o valor True (verdade) a uma variável ou expressão.

Set Variable: "on" = "1"
If (on)
Set Variable: "output" = "Power On"
End If.

A expressão (on) tem o valor 1 ou True (verdadeiro) mas se a variável "on" tiver o valor 0 então a expressão tem como valor False (falso), então não entra dentro do ciclo if. Há uma serie de maneiras para escrever funções booleanas. No exemplo anterior é utilizado uma expressão simples (on) e num ciclo IF statement pode ser avaliado por 1 ou 0, neste caso é 1 ou True (verdadeiro). Esta expressão pode ser escrita de diversas maneiras tudo depende da maneira que acharmos mais confortável. If (on=1) ou If (on = True) estas maneiras também são aceitáveis.

6. Arsenal de Funções - False

Sintaxe: True

Esta função atribui o valor False (falso) a uma variável ou expressão.

Set Variable: "on" = "1"
If (on = False)
Set Variable: "output" = "Power Off"
End If.

A expressão (on = false) tem o valor 0 ou False (porque on=1) mas a variável "on" tem o valor 1 ou True. Não confundir as duas coisas, a primeira é uma expressão para o Ciclo If statement. A segunda é a variável "on", neste caso a variável "on" tem o valor 0 ou False. Esta expressão pode ser escrita de diversas maneiras tudo depende da maneira que acharmos mais confortável. If (not on) ou If (on = 0) estas maneiras também são aceitáveis.

7. Arsenal de Funções - Newline

Sintaxe: Newline

Esta função cria uma linha de caracter vazia.

Set Variable: "nome" = "Fagundes"
Set Variable: "carta" = "Querido" & nome & "," & Newline & "Como estás?"

O output é:

Querido Fagundes,
Como estás?

Este exemplo utiliza o operador &, estudado na secção operadores, e a função Newline para conseguir o output e o layout do texto anterior. A função Newline faz o mesmo que a tecla enter num processador de texto. Quando o computador lê este comando salta para a linha de baixo e continua com o outup da expressão.

8. Arsenal de Funções - GetTimer

Sintaxe: GetTimer

Esta é uma função global de medida de tempo, em milisegundos, passado desde que um filme começou a correr. Não é afectado pela frame rate do filme, o tempo é lido no relógio do sistema do computador. O resultado apenas pode ser utilizado pela timeline mãe.

Set Variable: "tempodesdeinicio" = GetTimer
Set Variable: "tempoemsegundos" = tempodesdeinicio/100

Se um filme correu 3000 milisegundos quando a função é invocada, a variável "tempodesdeinicio" tem o valor 3000. O tipo deste valor é numérico quer isto dizer que se pode aplicar operadores aritméticos. No exemplo a variável "tempodesdeinicio" é dividido por 100 para obter em segundos o tempo desde que o filme começou.

9. Arsenal de Funções - Int

Sintaxe: Int(Number)

A função Int é usada para passar a inteiro um numero decimal. Esta função pode ser aplicada a uma variável que contenha um valor numérico.

Set Variable: "meuNumero" = 105.345
Set Variable: "NumeroInteiro" = Int (meuNumero)

O valor inserido na variável "NumeroInteiro" é 105, o ".345" foi removido pela função Int. Isto é particularmente usado quando se fazem divisões e se pretende ficar só com números inteiros.

10. Arsenal de Funções - Random

Sintaxe: Random(Number)

Enquanto o computador não consegue gerar um número verdadeiramente aleatório, esta função faz esse serviço e permite criar imprevisíveis efeitos nos nossos filmes. O argumento da função especifica o numero máximo que a função pode gerar, o limite minimo é o 0. Quer isto dizer que se o argumento for 100 os números gerados estarão entre o intervalo de 0 a 99.

Set Variable: "randomNumber" = Random (100)

O output do exemplo anterior é um numero aleatório entre 0 e 99. Á primeira vista esta função pode não ser muito excitante mas com alguma imaginação pode fazer coisas muito interessantes pelos nossos filmes.

11. Arsenal de Funções - Substring

Sintaxe: Substring(String,Index,Characters)

Esta função é muito usada e permite extrair uma letra ou grupo de letras de dentro de uma string. Há três argumentos que afectam o output desta função. O primeiro é uma string ou uma expressão com o valor de do tipo string. O argumento Index define onde começa na string, ou seja quantos caracteres a partir da esquerda. O ultimo argumento indica quantos caracteres se extrai. Se deixar- mos este argumento em branco são extraídos todos caracteres até ao fim da string. O espaço é considerado um caracter.

Set Variable: "myString" = "supercalafragalisticexpialodocious"
Set Variable: "stringPortion" = Substring (myString, 3, 6)

Nesta função, a variável "stringPortion" vai ficar com o valor "percal". A extracção da palavra começa na terceira letra da string contida na variável "myString" e acaba na sexta letra. A aplicação prática desta função pode ser num script que ordena alfabeticamente uma lista de palavras.

12. Arsenal de Funções - Lenght

Sintaxe: Length(String)

Esta função conta o numero de caracteres de uma string ou uma variável que contenha uma string.

Set Variable: "myString" = "Hello"
Set Variable: "charCount" = Length (myString)

O valor inserido na variável "charCount" é 5, o numero de caracteres da palavra "Hello" que está na variável "myString". O espaço é considerado caracter.

13. Arsenal de Funções - Chr

Sintaxe: Chr (String)

Esta função converte um caracter no seu código ASCII. Todos os teclados têm um único código ASCII. Por exemplo a letra "a" é o 97 e a letra maiúscula "A" é o 65. A função Chr converte o numero 97 na letra "a". A aplicação prática desta função é de conversão de letras maiúsculas para minúsculas. Isto consegue-se convertendo a letra maiúscula num numero e subtrai-se a esse valor 32, ao converter o resultado para uma letra, usando a função Chr, obtém-se a minúscula da letra inicial.

Set Variable: "temp" = 65
Set Variable: "myLetter" = Chr (temp)

A variável "myLetter" assume a string "A"

14. Arsenal de Funções - Ord

Sintaxe: Ord (character)

A função Ord converte um caracter em código ASCII. Esta função pode ser utilizada para converter letras maiúsculas em letras minúsculas. Vejamos como a conversão funciona.

Set Variable: "upperCaseLetter" = "A"	upperCaseLetter = "A"
Set Variable: "upperNumber" = Ord (upperCaseLetter)	upperNumber = 65
Set Variable: "lowerNumber" = upperNumber + 32	lowerNumber = 97
Set Variable: "lowerCaseLetter" = Chr (lowerNumber) upperCaseLetter = "A"	lowerCaseLetter = "a"

O código converte letras maiúsculas em minúsculas.

15. Arsenal de Funções - GetProperty

Sintaxe: GetProperty(target,property)

Get Property é provavelmente a função mais usada no Flash 4. É inacreditável a habilidade que a função tem para extrair valores de um movie clip que não é onde ela está a ser usada. Vamos ver os diferentes usos da função GetProperty:

_x)	Extraí a posição horizontal do movie clip. (pixels)
_y)	A posição vertical do movie clip (pixels)
_width)	A largura do movie clip (pixels)
_height)	A altura do movie clip (pixels)
_rotation)	Quanto um movie clip foi rodado (degrees)
_target)	O nome e o caminho completo do movieclip (string)
_name)	O instance name do movie clip (string)
_url)	O URL completo de um arquivo .SWF ou qualquer filme filho (string)
_xscale)	A percentagem escalada na direcção x (percentagem)
_yscale)	A percentagem escalada na direcção y (percentagem)
_currentframe)	Em que frame está actualmente um movie clip (number)
_totalframes)	Numero total de frames de um movie ou movie clip (number)
_framesloaded)	Numero de frames que foram carregadas (number)
_alpha)	Indica o valor de transparência de um movie clip (percentagem)
_visible)	Quando o movie clip é visivel ou não, true (1) ou false (0) (boleano)
_droptarget)	Constantemente actualizado, devolve o caminho do movie que está a ser arrastado

Aqui estão dois exemplos para ajudar a imaginar o que esta função faz.

O seguinte insere na variável "movieName" o nome (string) do movie clip "/myMovie"

]

Set Variable: "movieName" = GetProperty ("/myMovie", _name)

O próximo exemplo dá a localização vertical do movie clip quando se carrega no botão que está no próprio movie clip.

```
On (Release)
    Set Variable: "output" = _y
End On
```

Há outras funções que não foram mencionadas aqui, mas este tutorial foi elaborado apenas para dar uma introdução ás funções.

Agora é só começar a experimentar, que só assim é que se aprende!