Capítulo 7
A linguagem Actionscript

Índice

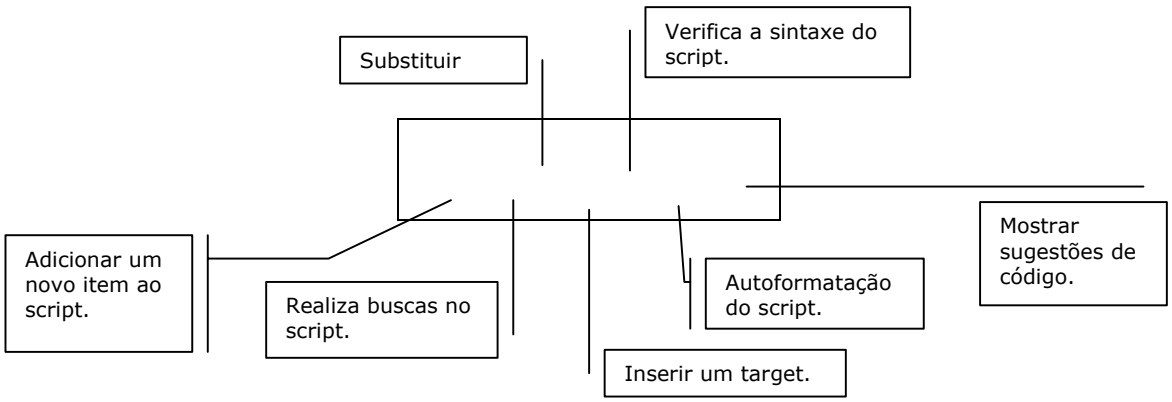
1. Painel Actions

Como já vimos, é no Painel Actions que iremos trabalhar com todo o código Actionscript que devemos gerar em nosso documento Flash. Para facilitar a programação, nesse painel, você encontra algumas ferramentas, como você pode observar na imagem abaixo:

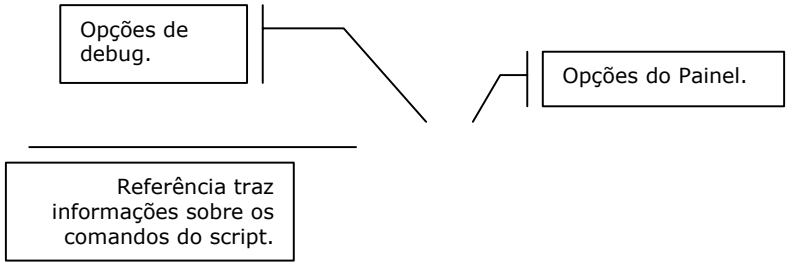
Abra o painel de actions em **Window > Development Panels > Actions**, ou pressione a tecla **F9**.

Exemplo do Painel Action do Flash.

- Canto superior esquerdo.



- Canto superior direito.



1.1 Referência

Durante a criação, muitas vezes, desejamos obter maiores informações sobre os recursos do Actionscript, para isso, selecione a palavra desejada e, então, clique no botão de referência. Se a palavra selecionada for encontrada na base de dados do Flash, uma janela será aberta, com informações sobre utilização, sintaxe etc.

As informações exibidas sobre uma determinada palavra são recuperadas do sistema de Ajuda da Macromedia. Por isso, para obter informações recentes, lembre-se sempre de atualizar o painel de ajuda.

1.2 Verificar a sintaxe

O Actionscript, como toda linguagem de programação, depende de uma sintaxe correta. Se estiver incorreta, a ação não será executada corretamente e um erro será reportado. O Flash oferece ferramentas que testam a sintaxe do script escrito.

Se a sintaxe estiver correta, será exibida uma mensagem informando que o script não contém erros.

Se estiver incorreta, será exibida uma mensagem informando que o script contém erros e a janela de output (saída) exibirá informações sobre o erro encontrado.

1.3 Auto Formatação

Ferramenta para auxiliar a formatação do script, se o código estiver correto o Flash se encarrega de formatar a sua disposição (identação do código).

1.4 Sugestão de código

Funções, geralmente, levam parâmetros e retornam resultados. Muitas vezes, não sabemos quais são os parâmetros devem ser passado a uma determinada função. Essa ferramenta nos ajuda a descobrir que tipo de dados deve ser inserido nos parâmetros de funções.

1.5 Adicionar um novo item ao script

Mesmo para os profissionais mais experientes, nem sempre é possível lembrar de todos os recursos e sintaxe de uma linguagem. Utilize essa ferramenta para adicionar ações ou eventos no código, de maneira mais rápida e eficiente.

1.6 Opções de Debug

Assim como em qualquer ambiente de programação, o Flash também disponibiliza ferramentas para inserir breakpoints e analisar valores de variáveis em tempo de execução.

1.7 Opções do Painel

Apresenta as opções de configuração do Painel Action. Por exemplo: podemos mostrar o número das linhas no script digitado.

1.8 Inserir um target

Quando utilizamos movie clips aninhados, o que significa um movie clip dentro de outro, é muito trabalhoso saber o número de níveis que devemos descer ou subir para chegar até a variável ou movie clip desejado. A ferramenta **Inserir um target** nos poupa tempo. Para utilizá-la, selecione o movie clip, que o target é inserido automaticamente.

1.9 Substituir

Quando temos um código muito extenso e precisamos modificar o nome de uma variável, algum trecho incorreto ou sintaxe; podemos utilizar o recurso da ferramenta Replace (Substituir) para nos poupar tempo. Ela substituirá todos os trechos desejados de uma só vez.

1.10 Buscar

Códigos podem ser extensos e confusos. Utilize a ferramenta Buscar, para auxiliá-lo na busca de palavras ou trechos de códigos no actionscript.

2. Actionscript

A utilização da linguagem Actionscript expande o potencial do Flash. Criação de ações básicas até programas complexos, sites dinâmicos para a Web, jogos, controle de som e multimídia abusam da interatividade com o Actionscript.

No Flash, cada objeto possui um conjunto específico de propriedades (características) e métodos que estão embutidos em sua estrutura. Cuidado ao se tratar do termo "objetos", já que objetos de Actionscript não representam os objetos que nos referimos no ambiente de desenvolvimento do Flash.

A partir do Flash MX 2004, o actionscript em sua versão 2.0 se tornou Case Sensitive, ou seja, ele diferencia letras maiúsculas e minúsculas, por exemplo, duas variáveis com o mesmo nome: **minhaVar** e **minhavar**, são diferentes, pois possuem letras maiúsculas e minúsculas diferentes.

No ambiente de desenvolvimento, um objeto é definido por um desenho que pode ser selecionado ou que tenha um traço ou preenchimento associado a ele. Porém, em Actionscript, um objeto é um símbolo específico ou um tipo de dados pré-definido residente, com propriedades específicas e métodos únicos para aquele tipo de objeto.

O objeto mais utilizado em Flash é o objeto movie clip, uma linha de tempo independente que pode ser inserida na linha de tempo principal. Outros objetos pré-definidos no Flash incluem mouse, key (Tecla), math, entre outros.

2.1 Trace()

A função **trace()** é utilizada para verificar o valor de variáveis ou verificar se um procedimento está sendo chamado.

O painel de output somente é exibido em tempo de desenvolvimento, quando você publica seu trabalho, ele não será mais visualizado.

Essa chamada abrirá uma caixa de output (saída) mostrando os valores requeridos.

1. Crie um novo documento Flash.
2. Pressione a tecla **F9** no primeiro frame para abrir o painel de actions e digite o seguinte

```
var minhaVar = 40;
trace("O valor da variável minhaVar é igual a " + minhaVar);
```

3. Repare que o caractere + é utilizado para concatenar a string e o valor da variável **minhaVar**, que é convertida para String automaticamente. Agora, pressione **Ctrl + Enter** para testar. Uma janela parecida com a exemplificada pela imagem abaixo deve ser aberta:

2.2 Operações Aritméticas

Vamos, agora, estudar os principais operadores utilizados em Actionscript:

= (atribuição de valores a variáveis)
– (subtração)
+ (soma)
* (multiplicação)
/ (divisão)

2.2.1 Exemplo

1. Crie um novo documento Flash.
2. Clique no primeiro frame do palco e pressione a tecla **F9**, o painel Actions será aberto.
3. Digite o seguinte código abaixo:

```
////////////////////////////////////////////////
//início do código
var meuNumero1 = 7;
var meuNumero2 = 3;
var resultado = 0;
resultado = meuNumero1+meuNumero2;
trace("Resultado da Soma = "+resultado);
```

```
resultado = meuNumero1-meuNumero2;
trace("Resultado da Subtração = "+resultado);
resultado = meuNumero1*meuNumero2;
trace("Resultado da Multiplicação = "+resultado);
resultado = meuNumero1/meuNumero2;
trace("Resultado da Divisão = "+resultado);
//fim do código
////////////////////////////////////////////////
```

Pressione **<Ctrl + Enter>** para testar. O resultado a ser mostrado na caixa de saída é o exemplificado abaixo:

2.3 Delimitadores de atribuições

// (delimitador de comentário de uma linha)
/* (delimitador de comentário de uma linha ou mais linhas). Para fechar o comentário use:*/
+= (atribuição de soma)
-= (atribuição de subtração)
/= (atribuição de divisão)
*= (atribuição de multiplicação)

```
////////////////////////////////////////////////
//início do código
var meuNumero = 7;
meuNumero += 5;
/*
meuNumero+=5 é o mesmo que
meuNumero = meuNumero + 5;
*/
trace("meuNumero = "+meuNumero);
//fim do código
////////////////////////////////////////////////
```

Resultado:

O mesmo serve para todas as outras atribuições.

2.3.1 Incremento e Decremento

++ (incremento)
-- (decremento)

Forma Normal de Uso

```
////////////////////////////////////////////////
//início do código
var meuNumero = 7;
meuNumero++;
/*
meuNumero++ é o mesmo que
meuNumero = meuNumero + 1;
*/
trace("meuNumero = "+meuNumero);
//fim do código
////////////////////////////////////////////////
```

Mensagem que aparecerá na janela de output: meuNumero = 8

2.3.2 Pré-incremento e Pós-incremento

++expressão (pré-incremento)
expressão++ (pós-incremento)

A forma pré-incremento:

```
//////////////
x = 1;
y = ++x;
trace("x é igual a "+x);
trace("y é igual a "+y);
//////////////
```

A forma pré-incremento primeiro incrementa x para 2 (x +1 = 2) e retorna o resultado para y.

Resultado (janela de output):
x é igual a 2
y é igual a 2

A forma pós-incremento:

```
//////////////
x = 1;
y = x++;
trace("x é igual a "+x);
trace("y é igual a "+y);
//////////////
```

A forma pós-incremento, primeiro adiciona o valor de x para y e depois incrementa o valor de x.

Resultado (janela de output):
x é igual a 2
y é igual a 1

2.4 Outros operadores

Como em qualquer outra linguagem, o Flash possui

|| (ou lógico)
&& (AND)
! (NOT lógico)
!= (diferença)
< (menor que)
<= (menor ou igual a)
<> (diferença)
= (atribuição)
== (igualdade)
> (maior que)
>= (maior ou igual a)

Consulte o guia de referencia: FL_ActionScript_Ref.pdf ou entre no próprio site da Macromedia: http://www.macromedia.com/support/flash/action_scripts/actionscript_dictionary/index.html, para obter maiores informações sobre os operadores acima.

2.5 Estruturas de condição

Estruturas de condição podem ser feitas de três formas em Actionscript. A primeira forma, e mais usual é:

2.5.1 IF ELSE

Sintaxe

```
if(condicao){
   comandos;
else{
  comandos;
}
```

Vamos criar um exemplo simples no Flash. Siga os passos abaixo:

1. Crie um novo documento Flash. Clique no primeiro frame da timeline e pressione a tecla **F9**.
2. Digite o código a seguir:

```
////////////////////////////
var x = 5;
var y = 10;
if(x<6) {
```

```
    z = x;
}else{
    z = y;
}
trace("Valor de x é igual a "+z);
////////////////////////////
```

3. Pressione Ctrl + Enter para testar seu documento. O resultado a ser impresso na caixa de saída deve ser:
   Valor de x é igual a 5

2.5.2 ? : (condicional)

Sintaxe:

condição **?** expressão1 **:** expressão2

Se a condição for verdadeira, a expressão1 é executada. Caso contrário, a expressão2 é que será executada.

```
////////////////////////////////////////////
x = 7;
y = 10;
z = (x < 6) ? x : y;
trace("Valor de x é igual a "+z);
////////////////////////////////////////////
```

Nesse exemplo, como a variável x é maior que 6, o segundo bloco (depois dos dois pontos) é que será executado e atribuído à variável z.

2.5.3 Switch

Algumas vezes precisaríamos de muitos if's e else's pra montar uma estrutura com muitas condições, o código pode ficar confuso. Podemos utilizar o **switch** para otimizar nosso código.

Sintaxe:

```
switch (expressão){
        case valor:
        [break;]
        [default:]
}
```

O switch cria uma estrutura ramificada para comandos, testa condições e executa comandos se uma das condições for verdadeira.

Exemplo

No exemplo a seguir, se o parâmetro meuTexto for avaliado como "A", a ação seguinte ao case será executada; se o parâmetro meuTexto for avaliado como "B", a ação trace seguinte ao case 2 será executada e assim por diante. Se

nenhuma expressão corresponder ao parâmetro meuTexto, a ação trace seguinte à palavra-chave default (padrão) será executada.

```
//////////////////////////////////////////
var meuTexto = "A";
switch (meuTexto) {
case "A":
     trace("case 1 verdadeiro ");
     break;
case "B":
     trace("case 2 verdadeiro");
     break;
case "C":
     trace("case 3 verdadeiro");
     break;
default :
     trace("nenhum caso verdadeiro");
}
//////////////////////////////////////////
```

Resultado na caixa de saída:
case 1 verdadeiro

Observe que, no exemplo anterior, sempre depois de um case, quebramos (break) a execução do Switch. Isso faz com que a sua execução seja interrompida naquele momento. No exemplo a seguir não iremos colocar a quebra no primeiro case, portanto, se a letra "A" for atribuída à variável meuTexto, a janela de saída mostrará os traces do primeiro e do segundo case.

```
//////////////////////////////////////////
var meuTexto = "A";
switch (meuTexto) {
case "A":
     trace("case 1 verdadeiro ");
case "B":
     trace("case 2 verdadeiro");
     break;
case "C":
     trace("case 3 verdadeiro");
     break;
default :
     trace("nenhum caso verdadeiro");
}
//////////////////////////////////////////
```

Resultado na caixa de saída:
case 1 verdadeiro
case 2 verdadeiro

2.6 Repetição e Laço

2.6.1 For

O **for** é a estrutura mais utilizada para repetições.

Sintaxe

for(início; condição; próxima) {
comandos;
}

início - Uma expressão a ser executada antes do início da seqüência de loop, geralmente uma expressão de atribuição para variáveis.

condição - Expressão verificada como true ou false. A condição é avaliada antes de cada iteração do loop. O loop termina quando a condição é avaliada como false.

próxima - Uma expressão executada após cada iteração do loop, geralmente uma expressão de incremento.

```
////////////////////////////
for (i = 3; i <=6 ; i++){
     trace(i);
}
////////////////////////////
```

Esse exemplo mostrará uma seqüência de números na janela de output.
3
4
5
6

2.6.2 While

A estrutura de repetição **while** é normalmente utilizada para executar uma ação até que uma condição seja satisfeita.

No final de cada iteração, um contador pode ser incrementado até que o valor especificado seja obtido. Tome cuidado para não permitir loop infinito, o que travaria o programa.

Sintaxe

while (condição) {
comandos;
}

condição é a expressão que será avaliada sempre que a ação while for executada. Se a condição for verdadeira, os comandos definidos dentro da estrutura serão executados.

A cada iteração a condição é testada novamente, se o teste retornar verdadeiro, o bloco de comando será executado. Se a condição for false, o bloco de comando será ignorado e o primeiro comando após o bloco de comando da ação while será executado.

```
//////////////////////////
i = 3;
while (i<=7) {
     trace(i);
     i++;
}
//////////////////////////
```

2.6.3 do while

Esse tipo de laço é um pouco diferente do while convencional, pois ele sempre executa os comandos pelo menos uma vez e somente no final do bloco, é que a condição será testada.

Sintaxe

do {
comandos
} while (condição**)**

condição: A condição a ser avaliada.

```
//////////////////////////
i = 3
do {
     trace(i)
     i ++;
} while (i<=7);
//////////////////////////
```

2.7 Funções

Uma função pode ser entendida como um conjunto de comandos pré-definidos para a realização de uma determinada tarefa, pode receber parâmetros e retornar valores. Você pode declarar uma função em um local e chamá-la de diferentes locais.

Declaração da função

function nomeDaFuncao (parametro1, parametro2, parametroN):tipoDoRetorno{
comando(s);
}

Exemplo

```
////////////////////////////
  function multiplica(n1,n2):Number{
    return n1*n2;
  }
  total = multiplica(5,4);

  trace("resultado da multiplicação = "+ total);
////////////////////////////
```

resposta
resultado da multiplicação = 20

2.8 Eventos

Muitas vezes, é preciso que um movie clip reaja a um "acontecimento", como por exemplo, ao clique do mouse. Para isso, utilizamos os eventos. Os eventos são acionados a partir de uma ação do usuário ou do próprio programa.

Alguns dos eventos mais utilizados para um movie clip são:

press	Acionado quando o Movie Clip é clicado pelo mouse.
release	Acionado quando liberamos o botão do mouse sobre a área do movie clip.
releaseOutside	Acionado quando liberamos o botão do mouse fora da área do movie clip.
rollOver	Acionado quando passamos o mouse em cima do Movie Clip.
rollOut	Acionado quando o mouse sai da área do Movie Clip.
load	Acionado quando o Movie Clip é carregado.
enterFrame	Acionado a cada frame do palco.
mouseMove	Acionado quando o mouse é movimentado.
keyPress	Acionado quando pressionamos uma tecla pré-definida do teclado.

Exemplo:

Vamos experimentar alguns dos eventos:

1. Abra um novo documento Flash.
2. Crie um Movie Clip qualquer.
3. Selecione-o e pressione a tecla **F9** para abrir o Painel Actions.
4. Insira o código abaixo:

```
//////////////
on (press) {
      trace("Movie Clip pressionado (press)");
}
on (release) {
      trace("Clique liberado sobre o Movie Clip (release)");
}
on (releaseOutside) {
      trace("Clique liberado fora do Movie Clip (releaseOutside)");
```

```
}
on (rollOver) {
      trace("Mouse em cima do Movie Clip (rollOver)");
}
on (keyPress "<Space>") {
      trace("Tecla de espaço pressionada");
}
onClipEvent (load) {
      trace("Movie Clip carregado");
}
//////////////
```

Ao terminar de inserir o código acima, pressione **<Ctrl + Enter>** e teste os eventos referentes ao Movie Clip.

2.9 String

Métodos mais utilizados para uma variável do tipo string:

Método	Descrição
String.**concat**	Combina (concatena) duas seqüências de caracteres retornando uma nova seqüência.
String.**indexOf**	Pesquisa os caracteres em uma string retornando o índice da primeira ocorrência dos caracteres especificados no parâmetro. Se o valor não for encontrado, -1 é retornado.
String.**lastIndexOf**	Pesquisa os caracteres em uma string retornando o índice da última ocorrência dos caracteres especificados no parâmetro. Retorna -1 se a busca não for bem sucedida.
String.**substr**	Retorna um número especificado de caracteres em uma string.
String.**substring**	Retorna os caracteres entre dois índices passados por parâmetro.
String.**toLowerCase**	Converte a string em minúsculas e retorna o resultado.
String.**toUpperCase**	Converte a string de caracteres em maiúsculas e retorna o resultado.
String.**length**	Retorna o número de caracteres da String.

Exemplo:

```
//////////////////////////////
var minhaVariavel = new String();
minhaVariavel = "Projeto Rived";

var minhaVariavelMinuscula = minhaVariavel.toLowerCase();
var t = new Number();
t = minhaVariavelMinuscula.indexOf("rived",0);
if(t!=-1){
      trace("A palavra rived foi encontrada na posição "+t);
}
//////////////////////////////
```

Teste o script, pressionando **<Ctrl+Enter>.**

2.10 Vetores (Array)

O objeto Array nos permite manipular e acessar matrizes identificadas por números que representam suas posições na matriz. Esse número é chamado de índice. Todas as matrizes são iniciadas pelo índice 0.

Para criar um vetor, use o construtor new Array ou o operador de acesso de matriz ([]).

Para acessar os elementos de uma matriz, use o operador de acesso de matriz ([ ]).

Resumo de métodos do objeto Array

Método	Descrição
objVetor.concat	Concatena dois vetores.
objVetor.join	Separa cada elemento do vetor por um caractere pré-definido.
objVetor.length	Retorna o número de elementos do vetor.
objVetor.pop	Retira elementos do topo do array (pilha).
objVetor.push	Coloca elementos no topo do array (pilha).
objVetor.reverse	Inverte os elementos do array.
objVetor.shift	Retira elementos do começo do array (pilha).
objVetor.sort	Ordena os elementos do array.
objVetor.toString	Transforma a array em uma string.

Exemplo:

No exemplo abaixo, criamos um vetor, com quatro posições, sendo que elas devem ser numeradas de 0 a 3:

```
//////////////////////
//new Array(); -> declara um vetor sem um limite de posições.
var nomes  = new Array();
 nomes[0] = "João";
 nomes[1] = "Carlos";
 nomes[2] = "Filomena";
 nomes[3] = "Daniel";

for(i=0;i<nomes.length;i++){
 trace(nomes[i]);
}
//////////////////////
```

Resultado na caixa de saída (output):
João
Carlos
Filomena
Daniel

2.10.1 Exercício

Crie uma função que receba como parâmetro um vetor de nomes de pessoas, essa função deverá mostrar na caixa de output todos os nomes que tenham a letra "a" (maiúscula ou minúscula).

Utilize o vetor abaixo para os testes:

```
var nomes  = new Array();
 nomes[0] = "Filomena";
 nomes[1] = "Carlos";
 nomes[2] = "Rodrigo";
 nomes[3] = "Daniel";
```

Use o método **toUpperCase** na resolução.

Caso necessite de mais informações sobre os métodos da linguagem Actionscript, utilize o Help do Flash pressionando <F1>.

> Ao terminar o exercício, envie os arquivos gerados pelo Flash para o seu tutor.
>
> Qualquer dúvida que tenha durante a produção do exercício acima, envie um e-mail ao seu tutor ou insira uma mensagem no Fórum FAQ de dúvida, no e-Proinfo.

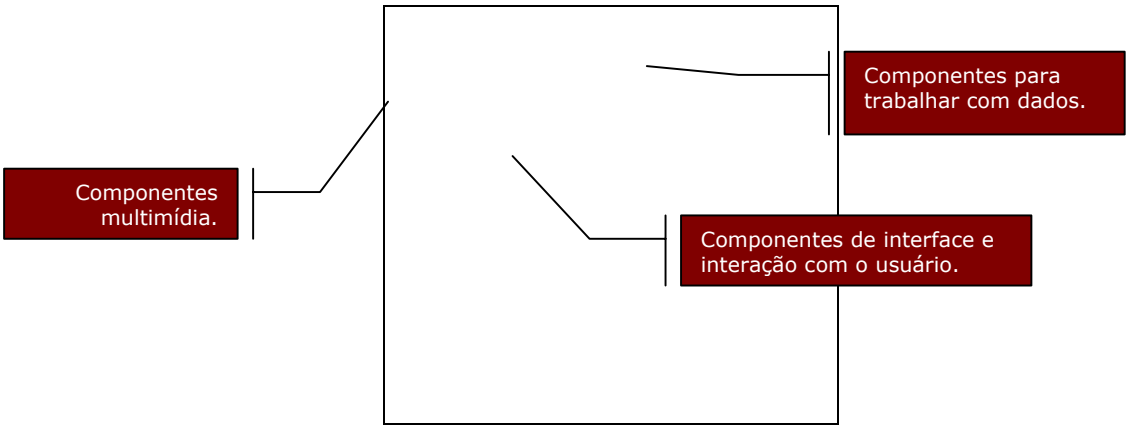
3. Utilizando Componentes Prontos

Um componente é um movie clip com parâmetros que só pode ser alterado em pontos específicos, definidos pelo criador do componente. Podemos utilizar APIs que nos permitem customizar o componente em tempo de execução. Componentes permitem que os desenvolvedores reutilizem e compartilhem código e, ainda, possam encapsular funcionalidades complexas que possam ser utilizadas por designers, sem experiência em Actionscript.

Você pode criar componentes, fazer download de componentes de outros desenvolvedores ou utilizar os componentes da Macromedia, já embutidos no Flash.

No Studio MX, o Flash foi dividido em duas partes, o Flash MX 2004 e o Flash MX 2004 Professional. Embora os dois tenham sofrido várias modificações, o segundo traz atrativos muito mais interessantes.

Se o painel de componentes não estiver aberto, abra-o, selecionando **Window > Development Panels > Components** ou pressione <Ctrl + F7>.



A versão Professional é uma ferramenta completa, que utiliza a tecnologia de formulários como no Delphi e no Visual Basic. O ambiente da aplicação é feito através de Forms facilitando o trabalho do desenvolvedor.

A maior novidade presente nas duas versões é a forma de programar, que mudou bastante da versão MX, podemos desenvolver um código muito parecido ao código do JAVA. O código pode ser todo trabalhado externamente em classes e possui alguns novos componentes que permitem a conexão direta com Web Services e XML (são os Data Connection), além de um suporte bem rígido em Web Services.

Há duas maneiras de trabalhar com componentes, a primeira é arrastando o componente para o palco e configurando seus parâmetros no Painel Property Inspector ou no Components Inspector. A outra maneira é criar a instância do componente dinamicamente e utilizar APIs para acessar as propriedades e métodos em tempo de execução.

3.1 Descrição dos Componentes

Componente	Descrição
Accordion component	Um grupo de telas sobrepostas que permite trocas da janela atual pelo usuário.
Alert component	Uma janela configurável que apresenta uma mensagem e botões para capturar a resposta do usuário.
Button component	Botão redimensionável que pode ser customizado com um ícone.
CheckBox component	Um caixa de escolha (verdadeiro ou falso).
ComboBox component	Permite aos usuários selecionarem uma opção de uma lista de escolhas.
DataGrid component	Permite mostrar e manipular múltiplas colunas de dados.
DataHolder component	O componente DataHolder é uma versão simplificada do componente DataSet que tem propósitos voltados para reter dados. Ele também pode ser usado como um conector entre

	componentes.
DataSet component	Esse componente trabalha com dados como coleções de objetos que podem ser indexados, classificados, buscados, filtrados e modificados.
DateChooser component	Permite que o usuário escolha uma data a partir de um calendário.
DateField component	Campo de texto não editável com um ícone de calendário, O usuário clica no campo de texto e o componente DataChooser é mostrado.
Label component	Um campo de texto de única linha, não editável.
List component	Permite aos usuários selecionarem uma ou mais opções de uma lista de rolagem.
Loader	Componente que pode carregar e mostrar um SWF ou um JPG.
MediaController component	Controla streaming de mídia em uma aplicação.
MediaDisplay component	Mostra o streaming de mídia na aplicação.
MediaPlayback component	Uma combinação dos componentes MediaController e MediaDisplay.
Menu component	Permite que usuários selecionem um comando de uma lista, um menu padrão desktop.
NumericStepper component	Permite aumentarmos ou diminuirmos o valor de um número por meio de cliques em botões, aumentando e diminuindo o valor.
ProgressBar component	Mostra o progresso de um processo de carregamento.
RadioButton component	Permite aos usuários escolher entre opções mutuamente exclusivas.
RDBMSResolver component	Permite combinarmos mudanças feitas para um DataSet em criar um pacote XML atualizado, pode ser analisado por um código fonte externo como um Web Service, Servlet ou ASP, que por sua vez tem a possibilidade de atualizar um banco de dados relacional.
ScrollBar component	Permite ao usuário ter a possibilidade de rolar o texto quando o mesmo não cabe na área delimitada.
TextArea component	Campo de texto editável ou não, de múltiplas linhas.
TextInput component	Campo de texto editável ou não, de linha única.
Tree component	Permite ao usuário manipular informações hierárquicas.
WebServiceConnector	Esse componente nos permite acessar

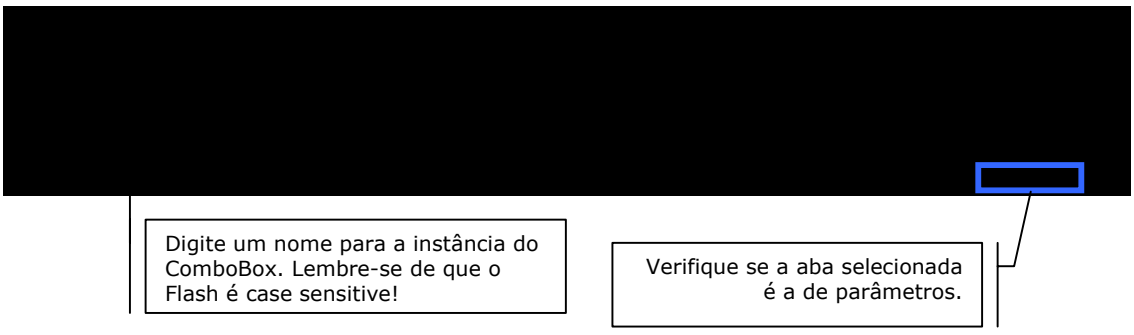
	métodos remotos publicados por um servidor usando o protocolo SOAP, via XML. Um Web Service pode aceitar parâmetros e retornar valores.
Window component	Esse componente mostra o conteúdo de um Movie Clip dentro de uma janela com uma barra de título, bordas e um opcional botão de fechar.
XMLConnector	Permite prover aplicações com acesso a qualquer fonte de dados que retorne ou receba um XML através de HTTP.
XUpdateResolver	É um padrão para descrever mudanças que são feitas a um documento XML e é suportado por uma variedade de banco de dados XML.

Quando adicionamos um componente ao documento, ele é inserido na Library (biblioteca). A partir desse momento, o Flash está pronto para utilizá-lo, caso você não queira mais, exclua-o do palco. Lembre-se de que mesmo excluindo um componente do palco, ele ainda existe na biblioteca. Um componente na biblioteca sempre será inserido no SWF, independentemente de ser instanciado ou não no palco, portanto, se não for utilizar mais o componente, retire-o da biblioteca.

Vamos fazer um exemplo?

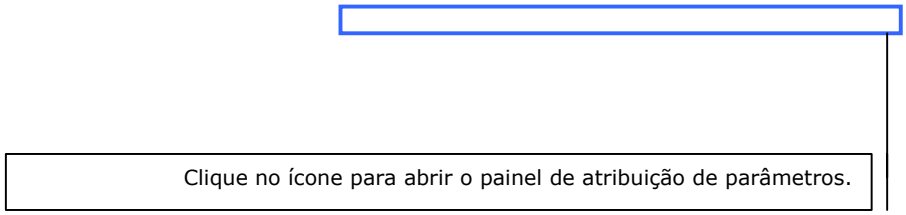
1. Crie um novo documento Flash.
2. Redimensione o palco para 350 px de largura por 60 px de altura. Abra o painel de componentes. Para isso, selecione **Window > Development Panels > Components** ou pressione **<Ctrl + F7>**.
3. Arraste os componentes: ComboBox, Button e Label, para o palco. Veja a figura abaixo:

4. Abra o Painel de Propriedades, se não estiver aberto (Window > Properties ou <Ctrl + F3>), clique sobre a instância do ComboBox, no palco.

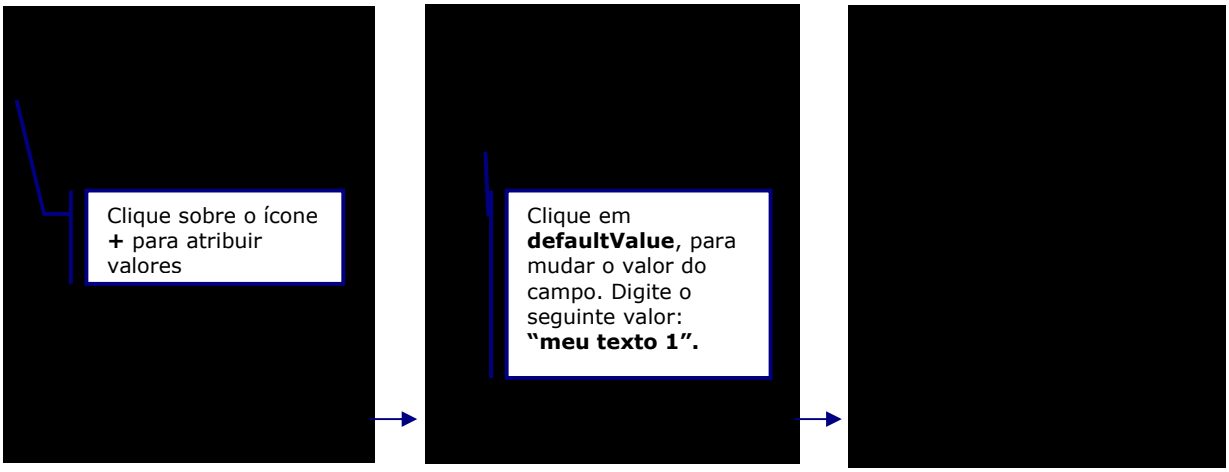


IMPORTANTE: não coloque acentuação em nome de instâncias.

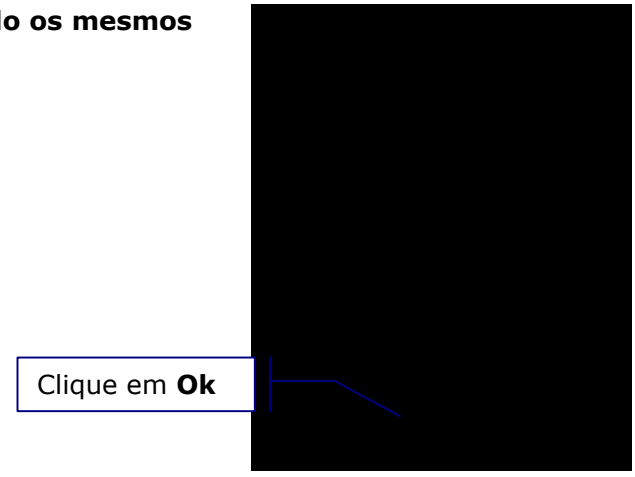
Na aba Parameters, encontre a propriedade labels e siga os passos abaixo:



Para inserir novos itens à caixa combo, faça o seguinte:



Crie mais três itens seguindo os mesmos passos:



5. Precisamos atribuir uma funcionalidade para o botão, que criamos. Ainda não nomeamos as instâncias dos componentes Button e Label. Para isso, abra o Painel de Propriedades, caso não esteja aberto.
    a. Selecione o botão e dê o nome de "meuBotao".
    b. Selecione o componente label e o nomeie para **"meuLabel"**.

6. Selecione novamente o botão e pressione **<F9>**.Certifique-se de que o Painel Action está fazendo referência ao botão meuBotao.

No Painel Action, digite o código abaixo:

```
on(click){
	var textoCombo = _root.meuComboBox.selectedItem.label;
	_root.meuLabel.text = textoCombo;
}
```

Entendendo o código

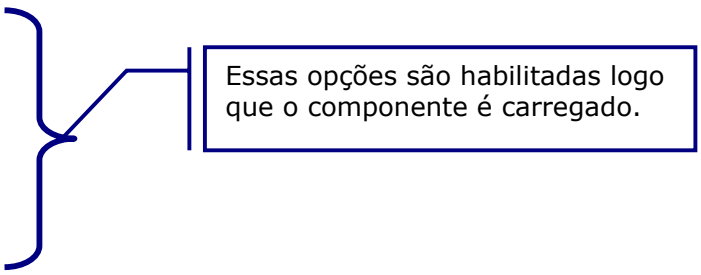
```
//Essa linha é chamada no evento onClick do botão:
on(click){
	//Atribuímos o label selecionado do ComboBox a uma nova
	variável:
	var textoCombo = _root.meuComboBox.selectedItem.label;

	//Atribuímos o valor da nossa nova variável (textoCombo)
	para a propriedade text do nosso componente Label
	(meuLabel):
	_root.meuLabel.text = textoCombo;
}
```

Pronto! Pressione **<Ctrl + Enter>** para testar a aplicação.

Para poder alterar outras características de um componente, utilize o painel **Component Inspector**. Para isso, selecione **Window > Development Panels > Component Inspector** ou pressione <Alt + F7>.

A figura abaixo é o painel **Component Inspector** de um componente tipo Button. Note que temos inúmeras opções para configurar o componente.



3.2 Exercício

Nesse exercício, você deverá criar uma pergunta de múltiplas respostas. Cada resposta deverá ser representada por um componente do tipo checkBox. Com isso, você aprenderá sozinho a utiliza o componente CheckBox. Lembrando que qualquer dificuldade que encontre, mande um e-mail para seu tutor! Abra o Help do Flash, teclando <F1>, para procurar informações sobre o componente.

Crie um Movie Clip para informar o acerto ou erro do aluno. Ele deve conter as seguintes mensagens:

Primeiro frame do Movie Clip de aviso:
"Parabéns, você acertou!".

Segundo frame do Movie Clip de aviso
"Você errou, tente novamente".

IMPORTANTE: utilize a função stop() nos dois frames.

Também deve existir um botão, que quando clicado, mostre o Movie Clip de aviso no frame correto da timeline.

Exemplo:

```
_root.meuAviso.gotoAndStop(1); //caso acertou
ou
_root.meuAviso.gotoAndStop(2); //caso errou
```

Ao finalizar seu exercício, envie os arquivos que o Flash gerou para o seu tutor.

Qualquer dúvida durante a produção do exercício, mande um e-mail ao tutor ou insira uma mensagem no fórum de dúvidas.

4. Publicação

Ao desenvolvermos um programa no ambiente do Flash, estamos alterando um arquivo de extensão FLA. Para que nosso conteúdo seja visualizado por outras pessoas, temos que publicá-lo em um formato reconhecido pelo player do Flash, em um navegador, ou seja, precisamos publicar nosso trabalho com a extensão SWF.

Algumas vezes, não podemos depender do player do Flash instalado nas máquinas, por isso, podemos publicar no formato .EXE (formato executável para windows), porém, o programa ficará muito maior já que o player do Flash é embutido no arquivo .EXE.

Se você utilizar publicar seu trabalho com as configurações padrão do Flash, ele preparará o seu arquivo para a Web, publicando o SWF, e criará um arquivo HTML com o código necessário para a sua exibição no navegador.

Para testar, crie um novo documento Flash e salve-o com o nome **teste.fla** ou utilize um de seus exercícios.
Publique o arquivo utilizando as opções padrão para publicação. Para isso, entre em **File > Publish** ou **<Shift + F12>**.

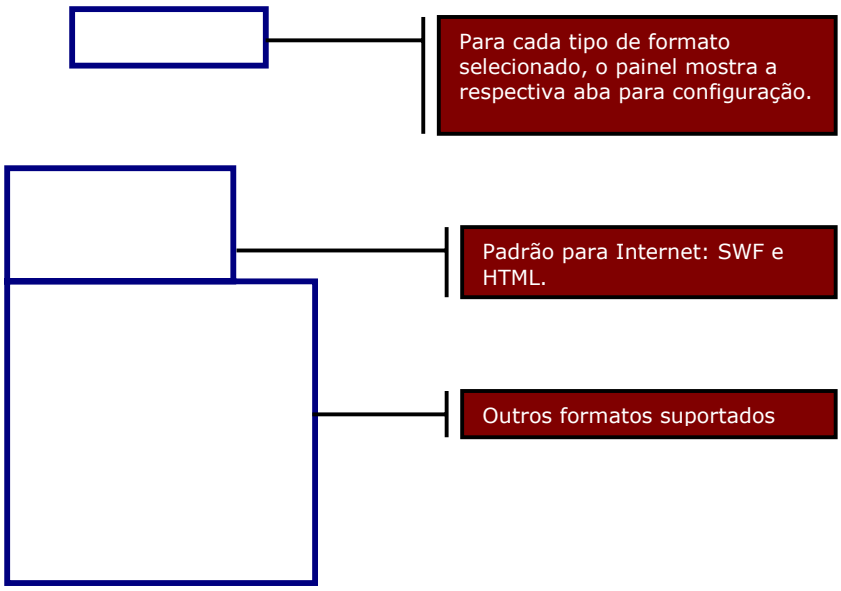
Dois arquivos são criados automaticamente na pasta onde se encontra o arquivo FLA, chamados de: teste.html e teste.swf.

Temos diversas formas de publicar as animações em Flash, seja para web, para um sistema operacional, um CD multimídia ou em imagens seqüenciais.

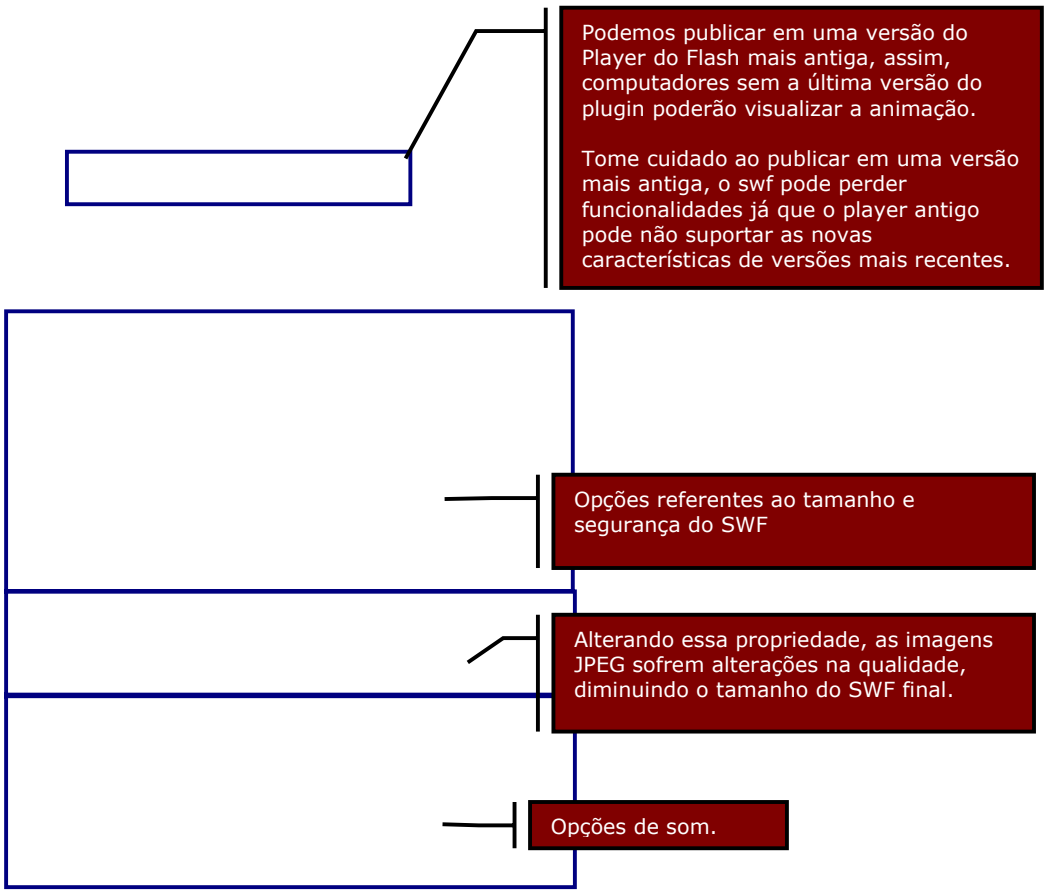
Para publicação em SWF, podemos alterar propriedades como a qualidade de imagem, compressão da animação e até a proteção, para que não seja aberta por usuários mal intencionados. Isso evita que possam capturar seu filme na Internet e utilizar imagens, filmes e sons sem sua autorização.

Quanto às animações seqüenciais, podemos também publicá-las em formato GIF. E, ainda, existe a opção para criar um vídeo em QuickTime.

Para verificar as opções de publicação do Flash, selecione **File > Publish Settings**. A janela abaixo será exibida:



A aba abaixo é uma das mais importantes, aqui mudamos as características da exportação no formato SWF.



Uma boa publicação trará enormes benefícios. Programas menores e compatíveis com navegadores e sistemas operacionais diferentes.