PROJECTO MULTIMÉDIA

AULA TÉCNICA (01). FLASH CS4

CONTEÚDOS + FLASH CS4 E AS3
+ SÍMBOLOS E LIVRARIA + VÍDEO

FBAUL: 22.MAI'09. A&M2

CRIAR BOTOES INTERACTIVOS COM ACTIONSCRIPT3.0

BOTAO > URL

MAKING A BUTTON WORK IN FLASH CS3 OR FLASH CS4 WITH ACTIONSCRIPT 3.0

http://flashthusiast.com/2008/02/25/making-a-button-work-in-flash-cs3-with-actionscript-30-its-not-too-bad/

CREATING BUTTONS THAT LINK TO DIFFERENT SCENES, AND WITHIN A SCENE, USING ACTIONSCRIPT 3.0

http://flashthusiast.com/2008/07/31/creating-buttons-that-link-to-different-scenes-using-actionscript-30/

ADDING MORE THAN ONE BUTTON TO A FLA FILE...

http://flashthusiast.com/2008/03/13/adding-more-than-one-button-to-a-fla-file-while-rocking-it-in-actionscript-30/

CONSTRUÇÃO

+ CARREGAR CONTEUDOS VISUAIS EXTERNOS AO SWF
+ LIGAR FILMES SWF
+ PRELOADING MOVIE

CARREGAR CONTEÚDOS

MODO DE CARREGAMENTO DOS FILMES EM FLASH:
O Flash carrega os conteúdos de cada frame.
Quando um primeiro fotograma é pesado, o resto do filme sofre com isso.

OPTIMIZAÇÃO DO FILME /
ESSENCIAL PARA A PUBLICAÇÃO NA WEB:
Retirar os conteúdos mais pesados (imagens, audio e ficheiros video) e usar ActionScript para carregar os ficheiros.
Desta forma, os conteúdos mais leves, carregam-se quase instantaneamente enquanto os conteúdos mais pesados são colocados no SWF a partir do servidor.

LOADING DINÂMICO

+ Utiliza a classe LOADER
+ introduzida no Flash CS3 e ActionScript 3.0

+ classe LOADER:

_carrega ficheiros externos .swf ou ficheiros de imagem (.jpg, .gif ou .png) para um .swf "colector" através do MÉTODO LOAD()

EXERCÍCIOS:

+ **CARREGAR FICHEIROS EXTERNOS .SWF OU IMAGENS**
+ **CARREGAR MÚLTIPLOS .SWF/IMGS COM UM ÚNICO LOADER // LOADER CLASS**
+ **CRIAR UM PRELOADER**

EXERCICIO:
“LOADING.FLA”. CHAMAR FICHEIROS EXTERNOS .SWF OU IMGS

ABRIR O FICHEIRO:
_veremos um único movieclip no stage. É o unico objecto na biblioteca.
_Demos o nome de “clip” à instância.

ABRIR **“LOADING_SG.FLA”**

1. seleccionar o primeiro fotograma do layer “script”, abrir o painel ACTIONS e escrever o seguinte código:
(ver ficheiro .fla c/ comentários)

```
var loader:Loader = new Loader();
//esta linha de código cria uma instancia da Loader class e armazena-a numa variável com o nome “loader”
//como esta instancia só vai carregar conteudos visuais é suficiente apenas criar o objecto
addChild(loader);
//clip.addChild(loader);
//estas 2 linhas de cima adicionam o “loader” numa lista de apresentação
//a primeira adiciona a variavel “loader” para q esta seja apresentada na timelina principal
//a segunda (comentada) adiciona o “loader” à lista de apresentação do movieclip c/ instance name “clip”

//--------------------------------
loader.contentLoaderInfo.addEventListener(Event.COMPLETE, completeHandler);
function completeHandler(evt:Event):void {
//clip.x=50;
//clip.y=50;
};
loader.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,progressHandler);
function progressHandler(evt:ProgressEvent):void {
        trace(evt.bytesLoaded);
};
//apresentam um par de “event handlers”
//o primeiro, ouve e responde a um evento Event.COMPLETE; isto significa que o ficheiro requisitado vai ser totalmente
“loaded”
//o segundo gere o ProgressEvent.PROGRESS, que é carregado continuamente enquanto faz o “load” dos ficheiros
//tb colocámos uma função “trace” para q o painel output vá mostrando qtos bytes estão a ser carregados

//--------------------------------
var req:URLRequest=new URLRequest (“toBeLoaded.swf”);
//var req:URLRequest=new URLRequest (“toBeLoaded.png”);
//dizem ao “loader” que ficheiro deve carregar: SWF ou PNG

loader.load(req);
```

2. testar filme

aparece uma img no canto superior esquerdo
vemos números no painel de OUTPUT
esses numeros mostram qto. está a ser carregado do filme durante a ocorrencia do ProgressEvent.PROGRESS
vemos poucos numeros pq o swf carrega mto rapidamente (está na nossa hard-drive)

3. retornar ao painel ACTIONS

comentar linhas:

```
//addChild(loader);
...
//var req:URLRequest=new URLRequest ("toBeLoaded.swf");
```

e descomentar todas as linhas previamente comentadas

diferenças:
neste momento estamos a adicionar o LOADER à lista de apresentação do MOVIECLIP em vez de na timeline principal e a dizer para o MOVIECLIP se reposicionar qdo. os seus conteudos são carregados e estamos a fazer o pedido de um .png em vez do .swf.

4. testar o filme
com a funç completeHandler e c as linhas
clip.x=50;
clip.y=50;
podemos reposicionar os conteudos do movieclip

NOTA IMPORTANTE:

**PARA VERMOS UMA SIMULAÇÃO DO LOADING DO FILME ATRAVÉS DE UM MODEM,
O SWF TEM A POSSIBILIDD DE PERCEBERMOS
O FLUXO DA LIGAÇÃO NO CARREGAMENTO DOS FICHEIROS.**

Dp de fazer “test movie” usar o “File” menu do swf e ver VIEW>SIMULATE DOWNLOAD. A partir daqui o Flash simula o pedido como se viesse de um servidor web remoto (lado esquerdo do painel). Na parte direita mostra o numero de fotogramas no swf (apenas um neste caso) e deixa-nos ver em q frame estamos.
assim podemos analisar as secções do nosso filme mais complicadas para o carregamento do filme.
Se quisermos simular c outro modem (c velocidds de download diferentes), seleccionar VIEW>DOWNLOAD SETTINGs do File menu do swf.

CARREGAR MULTIPLOS SWF OU IMAGENS COM UM UNICO LOADER // LOADER CLASS

Só precisamos de uma instância para carregar os .swf ou imagens que quisermos, desde que se visualize apenas um fotograma de cada vez (se apresentamos mais do que um frame precisamos de colocar uma instância “loader” por .swf ou imagem—precisamos de declarar o nº necessário de variaveis e dar a cada uma delas o seu nome.

COMO REUTILIZAR UMA ÚNICA INSTÂNCIA "LOADER" NUMEROSAS VEZES:

1. abrir "BEACHTRIP.FLA" da pasta "preloader"

Analisar filme:

+ O .fla tem 5 layers na timeline principal: scripts, container (para o movieclip que conterá as imgs carregadas dinamicamente), text, buttons e bg (para a img de background q está carregada na biblioteca).
+ Os botões têm nomes de instância "BTNPREV" e "BTNNEXT".
+ O texto no centro é dinâmico e tem o nome de instancia TFLOADPROGRESS.
+ O movieclip na layer "CONTAINER" está vazio como mostra o pequeno circulo no palco. ao clicar nesse circulo veremos q tem tb um nome de instancia "CONTAINER".

2. carregar no primeiro fotograma de scripts e começar a escrever AS:

+ este é um filme com multiplos frames
+ queremos q pare no 1º fotograma

abrir "BEACHTRIP_SG.FLA"

/* v. coments no action script do ficheiro "BeachTrip_SGcoments.fla"

```
stop();
//para q o filme pare no primeiro fotograma
//na realidd isto quer dizer “movieClip.stop()” mas como não usamos o prefixo percebemos q se trata da timeline principal

var loader:Loader=new Loader();
var req:URLRequest=new URLRequest();
container.addChild(loader);
//1ª linha: declaramos a instancia Loader, armazenada numa variavel “loader”
//1ª linha: a instancia URLRequest é declarada “req” e tb instanciada “new URLRequest()” mas sem q lhe atribuamos um nome de ficheiro
//o pedido do filme acontecerá nos fotogramas de 2 a 6
//para nos assegurarmos q as imgs q vão ser carregadas serão colocadas na display list de “container”, o “loader” é colocado como uma child de container

//2 Loader events c EventListeners
loader.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,progressHandler);
function progressHandler(evt:ProgressEvent):void {
        var percent:int=Math.round(evt.bytesLoaded/evt.bytesTotal*100);
//calcula qtos bytes estão a ser carregados, divididos pela totalidd de bytes e multiplicados por 100.
        //Esta expressão é arredondada para o numero inteiro mais aproximado pelo método Math.round()
        //se existirem 20.000 bytes no ficheiro q é carregado e 10.000 tiverem sido carregados, então a expressao diz:
        //10,000/20,000*100
        //ou seja 50%, metade do filme

        tfLoadProgress.text=percent.toString() + “%”;
        }
        //como percent é um numero inteiro, o metodo “toString()” é invocado como percentagem
        //o simbolo “+” agrupa a informação q será carregada dinamicamente pelo campo de texto
        //desta forma os utilizadores verão qto deste ficheiro já foi carregado como percentagem
```

```
//Este ProgressEvent Handler está a enviar info para a area de txt dinamico “tfLoadProgress”

loader.contentLoaderInfo.addEventListener (Event.COMPLETE,completeHandler);
function completeHandler(evt:Event):void {
	if (currentFrame !=totalFrames) {
		enableButton(btnNext, true);
	}
}
//o evento completo está a chamar uma função customizada “enableButton ()”

//utilizamos um “if” para q a afirmação avalie toda esta expressao para q assim determine se esta é falsa ou verdadeira
//desta forma, duas propriedds de movieclip sao comparadas com o operador de desigualdade “(!=)”
//”(!=)” significa “não é igual a”
//a expressao dentro do statement “if” é verdaeira qdo o play está “on” em qq fotograma da timeline excepto no ultimo fotograma
//porqUê?
//neste filme, o utilizador verá uma série de jpgs q são carregados dinamicamente qdo o utilizador carrega
//...nos botões
//não queremos q prossiga enqto a img q pediu não for carregada, então o botao “btnNext” será desligado em cada frame até q...
//...o evento Event.COMPLETE handler” seja accionado em todos os frames menos no ultimo (apenas pq não existe um proximo frame para além do frame

//----------------------
btnPrev.addEventListener(MouseEvent.CLICK, movePrev);
btnNext.addEventListener(MouseEvent.CLICK, moveNext);

function movePrev(evt:MouseEvent):void {
	prevFrame();
}
```

```
function moveNext(evt:MouseEvent):void {
	nextFrame();
}
//2 botões com os nomes de instancia btnPrev e btnNext associados a funçs customizadas “EventHandlers”...
//...accionados por MouseEvent.CLICK.
//as funçs sao nomeadas “MovePrev()” e “moveNext()”
//para q a timeline se mova uma frame antes e uma frame depois “prevFrame()” ou “nextFrame()”
//este codigo trata dos botoes no seu estado activo (por defeito é o q acontece c botoes)...
//agora só teremos que criar a funç enableButton() q faz c q os botoes fiquem activos ou desactivos como desejarmos

function enableButton(btn:SimpleButton, isActive:Boolean):void {
	btn.mouseEnabled=isActive;
	if (isActive) {
		btn.alpha=1;
	}
	else {
		btn.alpha=0.5;
	}
}
enableButton(btnPrev, false);

//esta funç aceita dois parametros, nomeados arbitrariamente “btn” e “isActive”
//o primeiro parametro refere-se a uma instancia SimpleButton (i.e. um botão como simbolo)
//o segundo é um valor Booleano (verdadeiro ou falso)
//dentro da funç, o botao é referenciado pelo parametro “btn”
//”btn.mouseEnabled=isActive;” faz c q o botao ou participe nos eventos relacionados c o rato (true) ou os ignore (false)
//”if (isActive)” verifica o valor de “isActive”
//se fôr verdade, o propriedade alfa do botão será colocada a 100% (ou seja 1) ou seja totalmente opaca
//se “false” seá colocada a 50% ou 0.5

//na ultima linha esta função é utilizada
//coloca o botão desactivado “btnPrev” sem eventos e semitransparente, q faz sentido no frame 1, pq não existe frame
anterior
```

3. dp de escrever o codigo na primeira frame, colocar keyframes no frame 2 e usar o menu actions para escrever:

```
req.url=”beach01.jpg”;
loader.load(req);

enableButton(btnPrev, false);
enableButton(btnNext, false);

//a variavel “req” declarada na frame 1 tem a sua propriedade URLRequest.url para chamar um jpg particular
//dp o método “loader.load()” é invocado pela instancia “loader” com “req” como parametro
//isto carrega uma img para cada frame como se pode ver pelo AS escrito nas frames seguintes
//como o “loader” foi adicionado à display list do “container” na frame 1, cada jpg vai ser carregado no movieclip container

//na frame 2, “btnPrev” é desactivado “enableButton(btnPrev, false)”...
//...pq esta frame é o inicio do noso slideshow
//o utilizador ainda nao pode saltar para a frame anterior
```

4.clicar no frame 3 escrever o seguinte codigo (repetir nos seguintes c ligeiras alteraçs):

```
req.url=”beach02.jpg”;
loader.load(req);

enableButton(btnPrev, true);
enableButton(btnNext, false);

//aqui o btnPrev é permitido pq a partir deste pto poderemos andar para tras e para a frente atraves dos botoes
//em todos os scripts nos frames, btnNext é desactivado, obrigando o utilizador a proceder até ao evento...
//...Event.COMPLETE handler for accionado
//de recordar q este event handler evoca enableButton() e o passa para os parametros btnNext e “true”
```

_
frames seguintes (alteramos nome ficheiro):

```
req.url=”beach03.jpg”;
loader.load(req);

enableButton(btnNext, false);

____
req.url=”beach04.jpg”;
loader.load(req);

enableButton(btnNext, false);

____
req.url=”beach05.jpg”;
loader.load(req);

enableButton(btnNext, false);
```

CRIAR UM PRELOADER // PROGRESSEVENT.PROGRESS

PRELOADER:

+ Informa o utilizador sobre o estado de carregamento do filme .swf
+ criação de uma animação enquanto o filme carrega
+ quando o filme é carregado, a animação desaparece,

_i.e. cria-se um cache de memória do .swf nos ficheiros temporários do utilizador;

_cria-se uma comparação entre LOADED BYTES e TOTAL BYTES (avaliados pelo IF statement nas primeiras 3 linhas de actionscript na frame 1)

_quando o igualamos os valores, o AS manda o filme parar com a animação e prosseguir com o filme principal

EXERCÍCIO
NASATRIP_SG.FLA

1. abrir **NASATRIP.FLA** na pasta Exercise/Preloader

2. analisar o filme:
_carregar na biblioteca no movieclip “PARROT”.
_adicionámos um txt dinamico numa caixa por baixo do papagaio. esta caixa manterá o utilizador informado do estado de progressão do carregamento do ficheiro

3. voltar à scene 1.
clicar na 1ª frame do layer script e escrever

```
if (loaderInfo.bytesLoaded >= loaderInfo.bytesTotal){
        //coloca o filme a interrogar se os bytesLoaded são superiores ou iguais aos bytesTotal
        gotoAndPlay(26);
}

loaderInfo.addEventListener(Event.COMPLETE, completeHandler);
//coloca o filmer a “ouvir” um evento “event.COMPLETE”,
//mas desta vez o evento é despachado por um evento loaderInfo
function completeHandler(evt:Event):void{
        if(currentFrame==15){
                //a frame onde se encontra o preloader
                play();
        }
        else {
                gotoAndPlay(26);
                //se o filme já estiver carregado passa para os conteudos
        }
};

//desta vez como o swf é interno ao filme, não precisamos de uma instancia “Loader”
```

mas precisamos de criar um stop ao ficheiro

4. adicionar Keyframe ao frame 15 da timeline principal

```
stop();
//para q a timeline espere até q o evento Event.COMPLETE seja expedido
```

5.para q seja colocado txt dinamico no PRE-LOAD q mostre ao utilizador qto é q já está carregado, colocar no action script da frame 15 ainda o seguinte txt (segue-se ao stop( );):

```
loaderInfo.addEventListener(ProgressEvent.PROGRESS, progressHandler);
function progressHandler(evt:ProgressEvent): void {
        var percent:int = Math.round(evt.bytesLoaded / evt.bytesTotal*100);
        parrot.percentage.text = percent.toString() + “%”;
};

//este evento associa-se à propriedade loaderInfo
//como o txt vai estar incluido no movieclip “parrot” e na instancia de txt dinamico “percentage”, temos q colocar:
//parrot.percentage.text = percent.toString() + “%”;
```

6. no script layer colocar keyframes em 39, 49, 59 e 69
colocar o seguinte código no frame 39:

```
stop();

btnNext.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(evt:Event): void {
        play();
}

//para o filme no primeira fotografia e coloca uma função customizada clickHandler no evento...
//MouseEvent.CLICK
//q acciona o botao c a instancia “btnNext”
```

7. colocar um METODO STOP(); nas frames 49, 59 e 69

```
stop();
```

8.testar o filme
9. simular o modem seleccionando VIEW>SIMULATE DOWNLOAD do menu de swf “FILE”
10. Colocar filme num servidor: para ver melhor a necessidade do pre-loader. ver:
http://areas.fba.ul.pt/411/AEM2/