Charger une image

Pour charger une image ou un SWF sur la scène d'une animation, vous devez faire appel à deux classes :

- URLRequest() : Pour stocker l'adresse de l'image à charger.
- Loader() : Pour définir un conteneur d'objet d'affichage dans lequel l'image va être chargée.

```
var chargeur:Loader = new Loader();

var adresseImage:URLRequest = new URLRequest("demo.jpg");

chargeur.load(adresseImage);

addChild(chargeur);
```

Note : Vous ne devez pas oublier d'ajouter l'objet d'affichage à la displayList à l'aide de la méthode addChild(), sinon vous ne verrez pas l'image sur la scène. Vous pouvez préciser le nom d'un conteneur d'objet d'affichage (displayObjet container) devant la méthode addChild().

Lorsque vous chargerez une image, vous aurez parfois besoin de connaitre l'instant où l'image est vraiment chargée (le chargement est terminé). Durant ce même chargement, vous aurez

peut-être besoin d'informer l'utilisateur de votre animation sur le niveau de charge. Vous devez donc faire appel aux deux évènements Event.COMPLETE et ProgressEvent.PROGRESS.

Exécuter une ligne d'instruction à la fin du chargement

Pour pouvoir exécuter une ligne d'instruction à la fin du chargement d'une image, vous devez faire appel à l'objet contentLoaderInfo car l'événement Event.COMPLETE n'est pas directement géré par la classe Loader().

```
chargeur.contentLoaderInfo.addEventListener(Event.COMPLETE,chargementTermine)

function chargementTermine (evt:Event) {

        trace("Chargement terminé")

}
```

Attention : Nous attirons votre attention sur l'ajout de l'objet contentLoaderInfo entre l'instance chargeur et la méthode addEventListener(). Sans cet ajout, vous ne pourrez pas exécuter le code correctement.

Exécuter une ligne d'instruction durant la phase de chargement

Nous devons faire appel à l'objet contentLoaderInfo, mais la classe d'événement à utiliser n'est pas Event, mais ProgressEvent.

```
chargeur.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,chargementEnCours)
```

```
function chargementEnCours (evt:ProgressEvent) {

        jauge.scaleX=evt.currentTarget.bytesLoaded/evt.currentTarget.bytesTotal;

}
```

Réaliser un preload d'images

Pour informer l'utilisateur du niveau de chargement d'une image, vous aurez besoin d'utiliser les points abordés ci-dessus, mais pour gagner du temps, Rudy (celui pour qui aucun immeuble ne lui résiste !!!) a préparé pour Yazo.net ce fichier tout préparé pour vous.

Télécharger l'exemple

Note : Dans les deux derniers cas ci-dessus (Exécuter une ligne d'instruction durant la phase de chargement et Réaliser un preload d'images), n'oubliez pas de faire un double Commande-Entrée (Mac) ou CTRL-Entrée (PC) pour tester votre animation dans Flash.

Charger un swf

Remarque : N'oubliez pas de publier vos animations au format SWF (un simple commande-Entrée (Mac) ou CTRL-Entrée (PC)) avant d'essayer de les charger.

Avant d'apprendre à charger un SWF sur la scène, essayons de comprendre à quoi cela peut servir.

Lorsqu'un Flasheur ne maitrise pas très bien la programmation en AS3, et qu'il ne peut donc pas programmer la construction dynamique d'une animation uniquement à base de code, il lui est toujours possible de charger un fichier au format SWF par dessus un autre et cela présente plusieurs avantages, en voici quelques-uns :

- Il devient plus facile de construire des écrans à partir de plusieurs fichiers, sous forme de

couches multiples.

- Le fait de travailler sur plusieurs fichiers facilite le travail en groupe.
- La gestions d'affichage de plusieurs parties sur la scène s'en trouve simplifiée.
- Cette technique peut permettre de réaliser des diaporamas ou projections de type PowerPoint © assez simplement.

Arrêtons ici notre énumération et passons à la procédure :

Elle est aussi simple que celle qui permet de charger une image car ce sont les mêmes lignes d'instructions, seules le nom du fichier à charger diffère.

```
var chargeur:Loader = new Loader();

var adresse:URLRequest = new URLRequest("animation1.swf");

chargeur.load(adresse);

addChild(chargeur)
```

ou

```
var chargeur = new Loader();

var adresse = new URLRequest("animation1.swf");

chargeur.load(adresse);

addChild(chargeur)
```

Comme nous l'évoquions au début de nos explications sur le chargement d'une animation dans une autre, vous pouvez utiliser cette technique, à travers la méthode load() de la classe Load() pour charger différentes animations les unes après les autres, simulant ainsi une navigation

d'écrans en écrans (tel un diaporama). Voici un exemple de script où deux animations peuvent être chargées dans une autre, l'une (des deux animations chargées) se substituant toujours à l'autre.

```
var chargeur:Loader = new Loader();

var adresse:URLRequest = new URLRequest("animation1.swf");

chargeur.load(adresse);

addChild(chargeur);
rubrique1_bt.addEventListener(MouseEvent.MOUSE_DOWN,charger1);

function charger1(evt:MouseEvent) {

    adresse = new URLRequest("animation1.swf");

    chargeur.load(adresse);

}
rubrique2_bt.addEventListener(MouseEvent.MOUSE_DOWN,charger2);

function charger2(evt:MouseEvent) {
```

```
    adresse = new URLRequest("animation2.swf");

    chargeur.load(adresse);

}
```

Remarque : Vous devez posséder trois animations intitulées accueil.swf, animation1.swf et animation2.swf. Le script ci-dessus se trouve dans l'animation accueil.swf qui doit également contenir 2 occurrences intitulées rubrique1_bt et rubrique2_bt.

Voilà, c'est aussi simple que ça, mais si vous souhaitez à présent faire communiquer les deux animations, cela se complique un peu. Nous allons aborder cette problématique dans le dernier développement de cette ressource, mais en attendant, voyons comment décharger une animation préalablement chargée sur la scène.

Décharger un swf

Nous avons deux animations intitulées accueil.swf et animation1.swf. Cette dernière est chargée dans la première grâce au premier script des explications relatives au chargement d'une animation (ci-dessus). Pour pouvoir décharger l'animation aniation1.swf, ajoutez le script ci-dessous dans l'animation accueil.swf.

```
carreParent.addEventListener(MouseEvent.MOUSE_DOWN,tourner);

function tourner(evt:MouseEvent) {

    chargeur.unload()
```

```
}
```

N.B. : N'oubliez pas de placer un clip sur la scène, que vous nommez carreParent.

Précisons que si vous aviez cherché à charger une autre animation à la place de la première (animation1.swf), sans la décharger préalablement, l'animation animation1.swf aurait tout naturellement été remplacée, comme le démontre le dernier exemple de l'explication sur le chargement d'une animation (l'avant dernier script ci-dessus).

Décharger un SWF à partir d'un clic sur une occurrence contenue dans l'animation à décharger

Attention : Avant d'essayer de comprendre le script ci-dessous, nous vous invitons à lire le dernier dévelopement ("Faire communiquer deux swf imbriqués) de cette ressource sur le chargement des images/swf.

```
var racine = root.parent.root;

carreEnfant.addEventListener(MouseEvent.MOUSE_DOWN,tourner);



function tourner(evt:MouseEvent) {



        MovieClip(racine).chargeur.unload();



}
```

Faire communiquer deux swf imbriqués (l'un chargé dans l'autre)

Comme nous l'évoquions ci-dessus, il n'est pas très facile de faire communiquer deux SWF imbriqués l'un dans l'autre car la syntaxe des deux script nécessaires pour faire dialoguer une animation à avec l'autre (dans les deux sens) est légèrement complexe. Afin de mieux comprendre le mécanisme, essayez par vous même de mettre en pratique l'exemple suivant :

1. Créez deux animations intitulées accueil.fla et animation1.swf. Toutes les deux contiennent une occurrence sur la scène intitulées carreParent(accueil.swf) et carreEnfant(animation1.swf).
2. Placez le script 1 dans l'animation accueil.fla.
3. Placez le script 2 dans l'animation animation1.fla.

Script 1 : Clic sur une occurrence contenue dans l'animation qui en charge une autre pour contrôler une occurrence contenue dans l'animation chargée.

```
var chargeur:Loader = new Loader();

var adresse:URLRequest = new URLRequest("animation1.swf");

chargeur.load(adresse);

addChild(chargeur)

carreParent.addEventListener(MouseEvent.MOUSE_DOWN,tourner);

function tourner(evt:MouseEvent) {

    MovieClip(chargeur.content).carreEnfant.rotation+=3;

}
```

Script 2 : Clic sur une occurrence contenue dans l'animation chargée pour contrôler une occurrence contenue dans l'animation qui a effectué le chargement.

```
var racine = root.parent.root

carreEnfant.addEventListener(MouseEvent.MOUSE_DOWN,tourner);



function tourner(evt:MouseEvent) {



    MovieClip(racine).carreParent.rotation+=3;



}
```

Dans le premier script, nous découvrons qu'il existe la propriété content qui permet de faire référence au contenu d'une animation chargée. Pour pouvoir faire appel à la propriété rotation, nous devons d'abord "convertir l'expression" chargeur.content en un symbole de type MovieClip, c'est pourquoi nous l'imbriquons dans le "mot" MovieClip().

Dans le deuxième script, le premier constat est étrange car nous stockons un chemin dans une variable. Une fois encore, comme nous venons de le voir ci-dessous, nous avons besoin de "convertir l'expression" en un MovieClip, c'est pourquoi nous faisons appel à la classe MovieClip(). Mais pourquoi root.parent.root ? Au moment où nous allons cliquer sur l'occurrence carreEnfant, nous allons avoir besoin de contrôler l'occurrence qui se trouve dans l'animation dans laquelle nous avons chargé l'animation animation1.swf. Nous devons donc chercher à remonter la hiérarchie comme nous l'avons fait. Afin que vous comprenniez peut-être un peu mieux la hiérarchie d'une telle syntaxe, voici ce que donne l'exécution des lignes d'instructions ci-dessous à partir de l'animation animation1.swf (animation chargée) et à partir de l'animation accueil.swf (animation qui reçoit le chargement).

```
trace(root)
trace(root.parent)
trace(parent)
trace(root.parent.root)
```

A partir du fichier animations1.swf (qui contient les lignes d'instructions ci-dessus) :

```
[object MainTimeline]
[object Stage]
```

```
[object Stage]
[object Stage]
```

A partir du fichier accueil.swf (le fichier animation1.swf est chargé avant d'exécuter le code) :

```
[object MainTimeline]
[object Loader]
[object Loader]
[object MainTimeline]
```

Cela vous démontre que la racine de l'animation chargée est bien une instance de type Loader et que vous devez chercher à remonter encore un niveau au dessus pour dialoguer avec l'occurrence de type Clip à partir de la timeline.