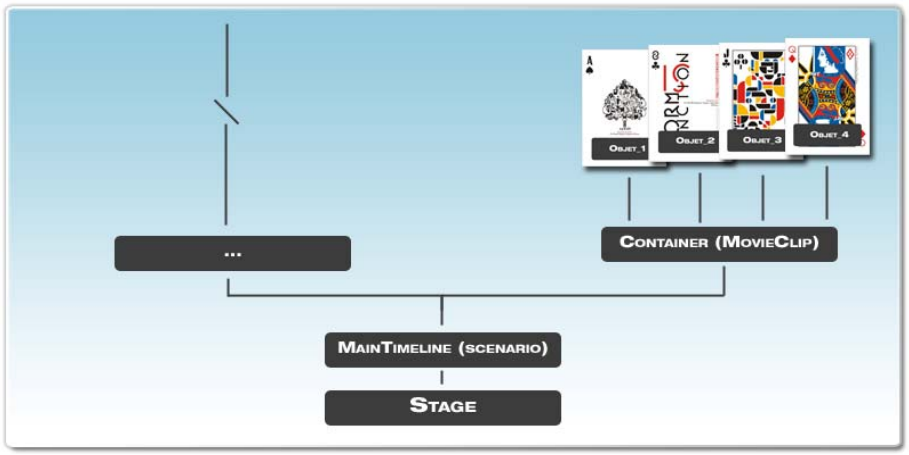
Ajouter et retirer du contenu graphique de la liste d'affichage.

En actionscript 2 tout objet graphique créé était aussitôt affiché. La nouvelle version d'actionScript différencie l'instanciation et l'affichage. Cette nouvelle structure est tout à notre avantage car elle nous permet d'effectuer toutes opération avant même que le clip soit sur la scène. Il est devenu maintenant plus simple et fluide de redimensionner un graphique chargé en le comparant avant insertion à la taille de la fenêtre du navigateur.

La liste d'affichage.



Propriétés :

```
getChildAt(index:int) : DisplayObject
getChildByName(nom_occurence:String): DisplayObject
addChild(obj_grpahique:DisplayObject): DisplayObject
addChildAt(index:int): DisplayObject
removeChild(obj_grpahique:DisplayObject): DisplayObject
removeChildAt(index:int): DisplayObject
getChildIndex(obj_grpahique:DisplayObject) :int
getNumChildren():int
```

Ajouter,enlever des objets sur la scène

Pour afficher un objet héritant de DisplayObject il doit être ajouté à la liste d'affichage à l'aide de la commande :
container.addChild(objetGraphique) ;

ou si l'on veut placer l'objet à un niveau determiné :
container.addChildAt(objetGraphique , index) ;

Dans ce cas tous les objets placés à un index superieur seront relevés d'un niveau.

Pour retirer cet objet de la liste on utilisera :
container.removeChild(objetGraphique) ;

ou si l'on veut retirer un objet à un niveau determiné :
container.removeChildAt(objetGraphique , index) ;
Dans ce cas tous les objets placés à un index supérieur seront descendus d'un niveau.

Suppression des objets.

Attention ! Contrairement à l'ancienne version d'actionscript, enlever un objet de la liste d'affichage ne le supprime absolument pas. Il est toujours présent dans le cache de l'application. (Essayez de replacer sur la scène un objet qui n'y figure plus en rappelant sa variable… Il réapparait). Pour « supprimer » un objet de l'application Il faut passer sa valeur à « null » :
removeChild(mon_clip) ;
mon_clip = null ;

Niveaux – profondeurs.

La notion de niveau fixe à disparu, dans la nouvelle version d'actionscript, au profit d'une notion d'empilement.
Les objets graphiques de la scène sont gérés par ce que l'on peut représenter comme un tableau indexé. Ainsi le niveau le plus faible représente toujours l'objet qui se trouve le plus en dessous.

Lorsqu'on place un symbole de la bibliothèque sur la scène dans l'environnement auteur, sa profondeur sera négative. Cela signifie qu'un élément instancié dynamiquement même si il est placé à l'index 0 à l'aide de addChildAt(0), figurera toujours au dessus. Il y aura lieu alors de prévoir dans le code un addChild() pour l'élément dans le script.

Les methodes getChildAt(index :int), addChildAt(objet :DisplayObject) et removeChildAt(objet :DisplayObject) traduisent la nouvelle gestion de niveau en as3.

```
container.getChildAt(0) : objet_1 ;
container.getChildAt(1) : objet_2 ;
container.getChildAt(2) : objet_3 ;
container.getChildAt(3) : objet_4 ;
```

Ici l'objet 4 se trouve au dessus de tous les autres sur la scène.

Si l'on retire l'un des éléments intermédiaires l'emplacement qu'il occupait est ecrasé et le « tableau » est réindexé.
Si nous retirons objet_2 :

container.removeChild(objet_2) ou container.removeChildAt(1) ;

le nouvel index s'echelonne comme suit :

```
container.getChildAt(0)  : objet_1 ;
container.getChildAt(1)  : objet_3 ;
container.getChildAt(2)  : objet_4 ;
```

Pour autant si l'objet 2 à bien été retiré de la scène, il existe toujours dans la mémoire de l'application. Si nous le replaçons :

```
container.addChild(objet_2) ;
```

l'index renouvelé montre que l'objet 2 est maintenant placé au dessus des autres ;

```
container.getChildAt(0) : objet_1 ;
container.getChildAt(1) : objet_3 ;
container.getChildAt(2) : objet_4 ;
container.getChildAt(3) : objet_2 ;
```

Note : utiliser la commande addChild() pour un élément déjà présent sur la scène aura pour conséquence de le placer au dessus des autres dans la liste.
Nous pouvons connaitre l'index d'un objet donné en interrogeant l'objet container à l'aide de la méthode getChildIndex(symbole_enfant :DisplayObject) :int

La méthode numChildren() :int permet de connaitre le nombre d'objets graphiques présents dans un objet héritant de DisplayObject. On pourra donc effectuer une boucle en se servant de getChildAt() pour récupérer chacun des objets du container.

```
function parcourirEnfants() :void
{

	var nb_enfants :int=container_parent.numChildren
	for(var i :int =0 ; i<nb_enfants ;i++)
	{
		var enfant:DisplayObject=container_parent.getChildAt(i);
	}
}
```

Astuce : l'utilisation d'une boucle assortie d'un getChildAt pour lister les éléments placés dans l'environnement auteur peut s'avérer problématique si celle-ci subit des changements comme l'ajout et le retrait de symbole. Le lecteur flash réagit mal et donne un index erroné. On préfèrera alors placer une fonction renvoyant un tableau des noms d'occurrences des objets pré-placés puis nous parcourrons le tableau en utilisant la fonction getChildByName() du container graphique.

```
function listeEnfant() :Array
{
	var tab_enfants :Array=new Array("nom_objet_1"," nom_objet_2"," nom_objet_3") ;
	return tab_enfants ;
}
```