MEMENTO

#2

AS3

David
TARDIVEAU

Memento publié en octobre 2008
Première édition - Version non relue
Deuxième édition : Mai 2010 (et toujours non relue)
(Pour celles et ceux qui veulent me renvoyer les corrections de relecture : david@yazo.net ☺)

Table des matières

PREAMBULE

Nature de cette publication

Attention, cette publication n'est pas un ouvrage pour découvrir ou apprendre l'ActionScript 3, mais un support, une aide, un mémento, pour vous accompagner dans vos premières productions d'animations Flash interactives.

En pré requis à la lecture de ce mémento, vous devez donc avoir reçu un enseignement ou être passé par une phase d'apprentissage en autodidacte du langage. Il ne peut en aucun cas être utilisé comme un ouvrage de référence en ActionScript 3.

Différence entre un objet d'affichage et une occurrence

Depuis Flash CS3 (Flash 9), la gestion des différentes occurrences sur la scène se fait différemment car la notion de "liste d'affichage" est l'une des bases fondamentales de l'ActionScript 3. Cette dernière est comparable à un registre, c'est-à-dire à un listing, qui fait un état des lieux permanent des "occurrences présentes" sur la scène. En anglais, la notion d'affichage" est souvent traduit par le terme "display", c'est pourquoi, dans la mesure où nous manipulons des occurrences comme nous le ferions avec des objets (un verre, un stylo, une bouteille, un vêtement, etc.), on parle "d'objet d'affichage", c'est-à-dire un "display object". Cette explication tend principalement à expliquer aux développeurs AS1/AS2, la différence entre une occurrence et un objet d'affichage. En conclusion, il s'agit de la même notion, c'est-à-dire la "représentation graphique" d'un symbole (de la bibliothèque ou créé dynamiquement) sur la scène.

LES BASES DE L'ACTIONSCRIPT 3

1. HIERARCHIE D'UNE ANIMATION

A - Schéma de la liste d'affichage (display list)

Lorsqu'un symbole, un texte ou une forme (etc.) sont placés/crées sur la scène, ils sont qualifiés d'objets d'affichage (anciennement des "occurrences"). Certains d'entre eux (voir le schéma de la Figure 1) peuvent contenir d'autres objets d'affichage, on les appelle alors des conteneurs d'objets d'affichage.

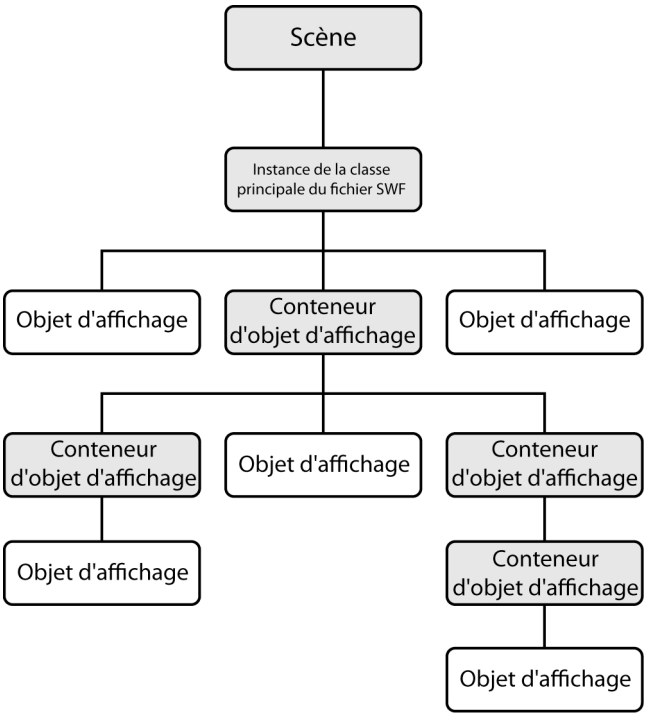

Figure 1 : La liste d'affichage d'une animation.

Lorsque vous placez un symbole de type Clip sur la scène, sa représentation graphique (sur la scène) est alors qualifiée d'objet d'affichage. Si vous éditez ce symbole pour lui placer un autre symbole à l'intérieur, dans ce cas, vous n'avez plus un objet d'affichage sur la scène, mais un conteneur d'objet d'affichage. Si vous placez sur la scène, un symbole qui en contient déjà d'autres (des objets d'affichage), vous obtenez directement un conteneur d'objet d'affichage.

B - Schéma des différents types d'objets d'affichage (display objects)

Les objets d'affichage qui se trouvent par définition sur la scène peuvent être de différents types, en voici une liste exhaustive :

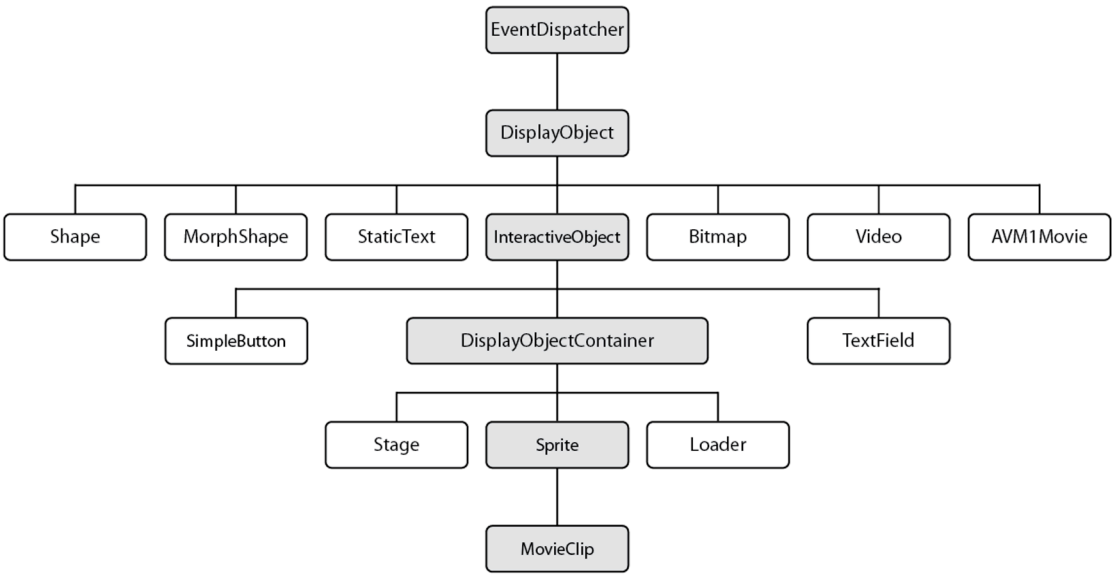

Figure 2 : Les différents types d'objets d'affichage

D'après ce schéma, on comprend pourquoi une forme ou un texte statique ne pourront jamais être des conteneurs d'objets d'affichage à la différence d'un objet d'affichage de type **Sprite**.

***Remarque : AVM1 Movie** signifie qu'il s'agit d'une animation Flash (SWF) aux formats Flash 8 ou antérieurs.*

C - Schéma de lecture d'une animation Flash au sein du lecteur Player

Lorsque le lecteur Flash lit une animation au format SWF, voici ce qu'il se passe à travers cette analogie :

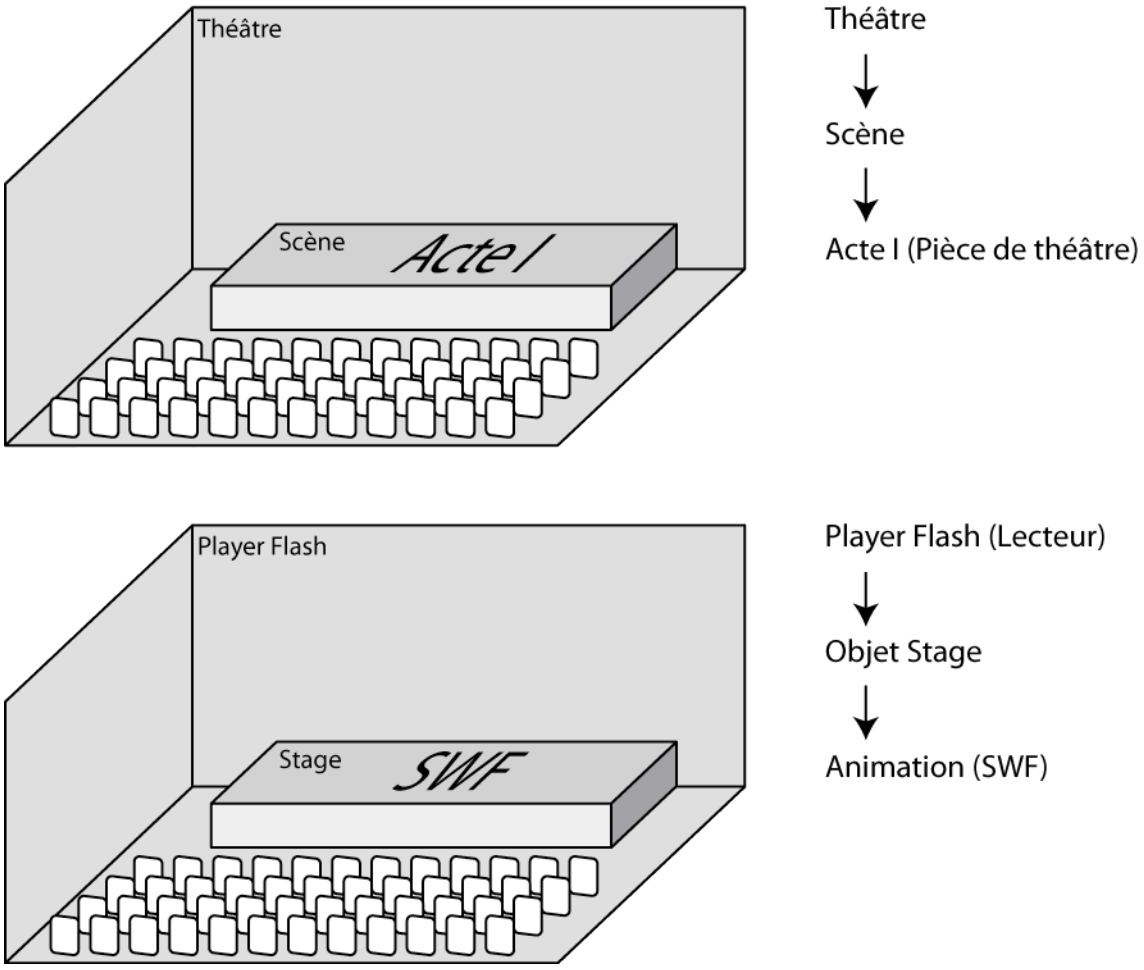

Figure 3 : Logique de lecture d'une animation Flash via le lecteur (player) Flash

Le lecteur Flash que vous téléchargez (le plug-in Flash de votre navigateur ou le lecteur autonome) est comparable à un théâtre qui contient une scène (l'objet Stage) sur laquelle se joue une pièce de théâtre (l'animation au format SWF).

Pour preuve et traduction de ce schéma en code, voici ce que renvoie l'animation lorsque les lignes d'instruction ci-dessous sont exécutées :

```
trace(this)
trace(parent)
trace(root)
```

Résultats dans la fenêtre Sortie (de l'IDE) de Flash.

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

2. LES ECOUTEURS

A - Structure d'un gestionnaire d'événement

1. Code exécuté au clic sur un objet d'affichage (dont le nom est **monBouton**).

```
monBouton.addEventListener(MouseEvent.MOUSE_DOWN,deplacer);
function deplacer(evt:MouseEvent) {
  personnage.x=150;
  personnage.y=230;
}
```

***Remarque** : Le nom de "l'objet" **evt** est choisi librement, mais il ne doit pas contenir de caractères spéciaux, accentués et ne commence pas par une majuscule. Si le terme "objet" de cette explication vous gêne, remplacez le maladroitement par le terme "variable". N'utilisez pas le nom **event** si vous démarrez en programmation afin qu'il n'y ait aucune ambiguïté avec le mot **Event** qui a un tout autre sens.*

Tableau des différentes constantes de la classe **MouseEvent**

Nom de la constante	**Déclenchement de l'évènement**
MOUSE_DOWN	Au clic sur l'objet d'affichage
MOUSE_UP	Lorsque le bouton de la souris est relâché après un clic
MOUSE_MOVE	Lorsque le curseur de la souris bouge sur (ou avec) l'objet d'affichage
MOUSE_OVER	Lorsque le curseur de la souris survol l'objet d'affichage
MOUSE_OUT	Lorsque le curseur ressort d'un objet d'affichage

Il existe également les constantes ci-dessous qui fonctionnent selon la même équivalence

Constante…	**…équivalente à cette constante**
CLICK	MOUSE_DOWN
ROLL_OVER	MOUSE_OVER
ROLL_OUT	MOUSE_OUT

2. Code exécuté en continu

```
etoile.addEventListener(Event.ENTER_FRAME,tourner);
function tourner(evt:Event) {
  etoile.rotation+=3;
}
```

***Remarque** : Le nombre d'images par seconde de l'animation influe directement sur la vitesse d'exécution des lignes d'instructions.*

Pour arrêter un mouvement en continu, exécutez la ligne d'instruction ci-dessous :

```
etoile.removeEventListener(Event.ENTER_FRAME,tourner);
```

B - Les événements MouseEvent.DOUBLE_CLICK et MouseEvent.MOUSE_WHEEL

Pour activer la reconnaissance du double-clic sur un objet d'affichage, réglez la propriété **doubleClickEnabled** à **true**.

```
elle.doubleClickEnabled = true
elle.addEventListener(MouseEvent.DOUBLE_CLICK,tourner);
function tourner(evt:MouseEvent) {
  elle.rotation+=3;
}
```

Pour désactiver la reconnaissance de l'utilisation de la molette de votre souris, réglez la propriété **mouseWheelEnabled** à **false** (par défaut, elle est réglée à **true**).

3. LES PROPRIETES D'UN OBJET D'AFFICHAGE

Toutes les propriétés ci-dessous peuvent être modifiées (exceptées les 5 dernières) et lues.

Propriétés	Unité	Valeurs maxi/mini	Spécificités
x	pixels	Pas de limites	Indique la position d'un objet d'affichage par rapport au bord gauche de la scène.
y	pixels	Pas de limites	Indique la position d'un objet d'affichage par rapport au haut de la scène.
alpha	pourcentage	Entre 0 et 1 (0.6 pour obtenir 60%)	Indique le degré de transparence d'un objet d'affichage .
height	pixels	Pas de limites	Indique la hauteur d'un objet d'affichage .
width	pixels	Pas de limites	Indique la largeur d'un objet d'affichage .
scaleX	pourcentage	Pas de limites	Indique l'échelle horizontale d'un objet d'affichage par rapport à sa taille d'origine (au moment où il a été placé sur la scène la première fois).
scaleY	pourcentage	Pas de limites	Indique l'échelle verticale d'un objet d'affichage par rapport à sa taille d'origine (au moment où il a été placé

			sur la scène la première fois).
visible	booléen	true ou false	Affiche ou masque un objet d'affichage .
rotation	degrés	Pas de limites	Règle l'angle de rotation d'un objet d'affichage .
rotationX	degrés	Pas de limites (A partir du player Flash 10 uniquement)	Règle l'angle de rotation d'un objet d'affichage par rapport à un axe horizontal (de gauche à droite, pour un mouvement vertical, de bas en haut).
rotationY	degrés	Pas de limites (A partir du player Flash 10 uniquement)	Règle l'angle de rotation d'un objet d'affichage par rapport à un axe vertical (de haut en bas, pour un mouvement horizontal, de gauche à droite).
rotationZ	degrés	Pas de limites (A partir du player Flash 10 uniquement)	Règle l'angle de rotation d'un objet d'affichage comme le fait la propriété **rotation**.
name	chaine de caractères	Pas de caractères spéciaux, accentués et ne doit pas commencer par une majuscule.	Permet de définir ou lire le nom d'occurrence d'un objet d'affichage .
mouseX	pixels	Propriété en lecture seule.	La valeur renvoyée est relative au côté gauche de la scène ou relative au point d'alignement de l'objet d'affichage .
mouseY	pixels	Propriété en lecture seule.	La valeur renvoyée est relative au haut de la scène ou relative au point d'alignement de l'objet d'affichage .
stage	n/a	Propriété en lecture seule.	Tous les objets d'affichage possèdent la propriété commune **stage**.
parent	n/a	Propriété en lecture seule.	Fait référence à l'objet d'affichage qui contient celui auquel vous associez la propriété parent.
root	n/a	Propriété en lecture seule.	Fait référence à l'objet d'affichage racine.

La propriété **mouseChildren**

Elle sera indispensable dans le cas où vous aurez besoin de désactiver ou d'activer le clic sur les objets d'affichage contenus dans un autre. (Dans l'exemple ci-dessous, un objet d'affichage intitulé **tableauBord** en contient deux autres intitulés **bt1** et **bt2**).

```
tableauBord.addEventListener(MouseEvent.MOUSE_DOWN,clicTableau)
tableauBord.mouseChildren = false
bt1.addEventListener(MouseEvent.MOUSE_DOWN,clicBoutons)
bt2.addEventListener(MouseEvent.MOUSE_DOWN,clicBoutons)
function clicTableau (evt:MouseEvent) {
```

```
    trace("Tableau de bord");
}
function clicBoutons (evt:MouseEvent) {
    trace("Bouton");
}
```

Changez le paramètre **false** par **true** pour évaluer la différence entre ces deux cas de figure.

Remarque : *Le script ci-dessus sous-entend que les objets d'affichage intitulés* ***bt1*** *et* ***bt2*** *ont été ajoutés dans l'objet d'affichage intitulé* ***tableauBord*** *à l'aide de la méthode* ***addChild()****. Si vous construisez vous-même un symbole contenant deux autres objets d'affichage (occurrences), utilisez la ligne d'instruction ci-dessous :*

```
tableauBord.bt1.addEventListener(MouseEvent.MOUSE_DOWN,clicBoutons)
```

4. TRAITEMENTS GRAPHIQUES D'UN OBJET D'AFFICHAGE

A - La propriété blendMode

Pour appliquer un mode de surimpression à un objet d'affichage par rapport à son arrière plan, utilisez la propriété **blendMode** de la façon suivante :

```
nomDeLobjetdAffichage.blendMode=BlendMode.DIFFERENCE;
```

La figure ci-dessous vous présente le résultat de l'application des différents modes entre un carré vert dont le nom d'objet d'affichage est **nomDeLobjetdAffichage** (au premier plan) et un carré rouge (en arrière plan).

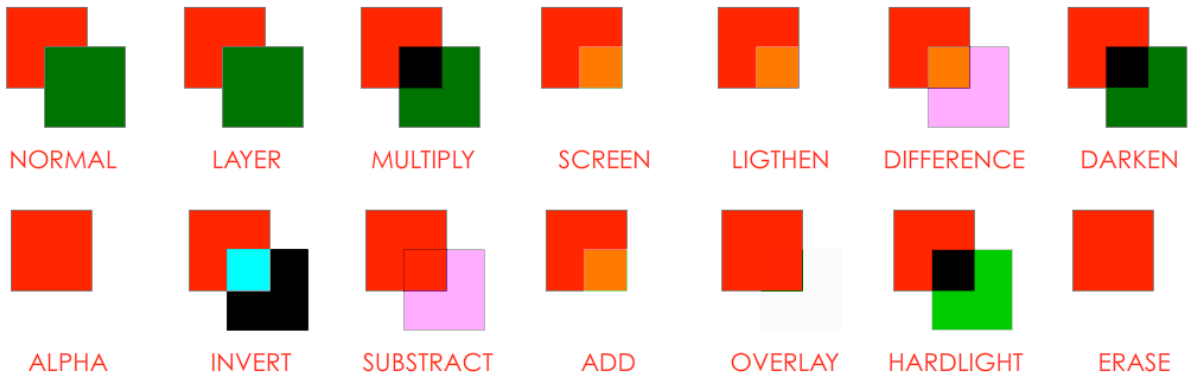

Figure 4: Les différents modes de surimpression (exemple 1)

Voici les mêmes calculs avec des carrés contenant des dégradés (Bleu-Rouge-Vert-Noir-Blanc).

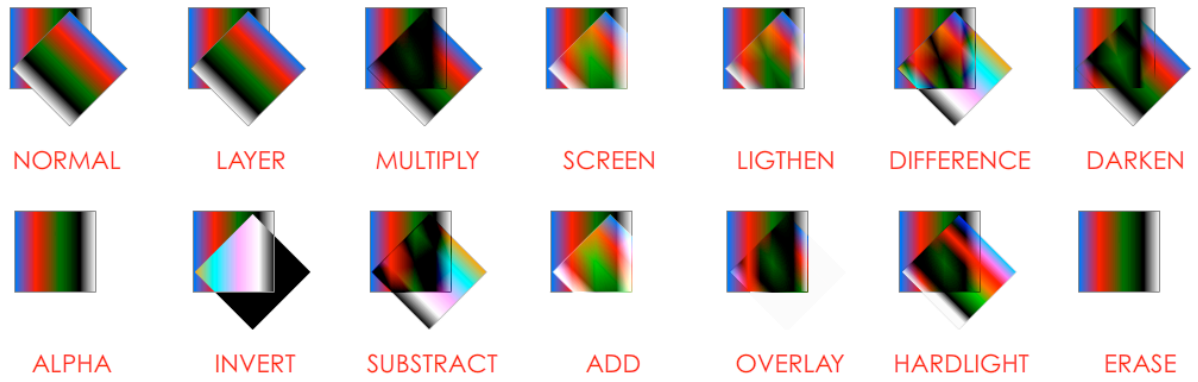

Figure 5: Les différents modes de surimpression (exemple 2)

B - La propriété buttonMode

Pour afficher le curseur de type doigt sur un objet d'affichage de type Sprite (ou MovieClip), utilisez la propriété **buttonMode** de la façon suivante :

```
boutonEcranSuivant.buttonMode=true;
```

C - La propriété mouseEnabled

Pour désactiver le clic sur un objet d'affichage, utilisez cette propriété de la façon suivante :

```
boutonEcranSuivant.mouseEnabled = false;
```

Dans le cas où vous souhaiteriez désactiver le clic de tous les objets d'affichages contenus dans un conteneur d'objets d'affichage, utilisez la propriété **mouseChildren**. (voir ci-dessus, "Les propriétés d'un objet d'affichage")

D - Les filtres

Pour appliquer une ombre portée (ou tout autre filtre) à un objet d'affichage, utilisez le script ci-dessous :

```
import flash.filters.DropShadowFilter;
var ombre:DropShadowFilter = new DropShadowFilter();
var filtres:Array = new Array();
ombre.blurX = 7;
ombre.blurY = 7;
ombre.distance = 3;
ombre.angle = 45;
ombre.alpha=0.3;
ombre.strength = 2;
ombre.quality = BitmapFilterQuality.HIGH;
ombre.color = 0xAA3333
filtres.push(ombre);
etiquette.filters=filtres;
```

Pour appliquer d'autres filtres, remplacez la fin de la première ligne du script ci-dessus (**DropShadowFilter**) par l'un des "mots" suivants : **BevelFilter, BlurFilter, DropShadowFilter, GlowFilter, GradientBevelFilter, GradientGlowFilter**. N'oubliez pas d'instancier la bonne classe (le deuxième et le troisième mot "DropShadowFilter" de l'exemple ci-dessus).

Liste des propriétés disponibles par type de filtre

	Bevel	**Blur**	**DropShadow**	**Glow**	**GradientBevel**	**GradientGlow**
alpha	-	-	Oui	Oui	-	-
alphas	-	-	-	-	Oui	Oui
angle	Oui	-	Oui	-	Oui	Oui
blurX	Oui	Oui	Oui	Oui	Oui	Oui
blurY	Oui	Oui	Oui	Oui	Oui	Oui
distance	Oui	-	Oui	-	Oui	Oui
quality	Oui	Oui	Oui	Oui	Oui	Oui
color	-	-	Oui	Oui		-
colors					Oui	Oui
strength	Oui	-	Oui	Oui	Oui	Oui
hideObject	-	-	Oui	-	-	-
inner	-	-	Oui	Oui	-	-
knockout	Oui	-	Oui	Oui	Oui	Oui
type	Oui	-	-	-	Oui	Oui
shadowAlpha	Oui	-	-	-	-	-
shadowColor	Oui	-	-	-	-	-
highlightAlpha	Oui	-	-	-	-	-
highlightColor	Oui	-	-	-	-	-
ratios	-	-	-	-	Oui	Oui

*Remarques : Les valeurs en pourcentage sont exprimées par une valeur comprise entre 0 et 1 (Ex. : 0.3 pour 30 %). Les valeurs pour les couleurs sont exprimées sous la forme 0x000000. Les valeurs de blur (blurX et blurY) possèdent généralement des valeurs comprises entre 1 et 10 (sauf dans le cas d'effets particuliers telles qu'une ombre fortement décalée). Faites un trace de la propriété que vous souhaitez modifier si vous ne connaissez pas la plage des valeurs applicables (Ex. : **trace(ombre.alpha)**).*

E - Les masques

Pour ne voir qu'une partie d'image, utilisez la propriété **mask** comme dans l'exemple ci-dessous :

```
visuel.mask=leMasque;
```

Dans l'exemple ci-dessous, le nom de l'objet d'affichage contenant la photo est **visuel**, le nom de l'objet d'affichage contenant la forme géométrique qui est au premier plan (les deux rectangles gris qui se chevauchent) est **leMasque**.

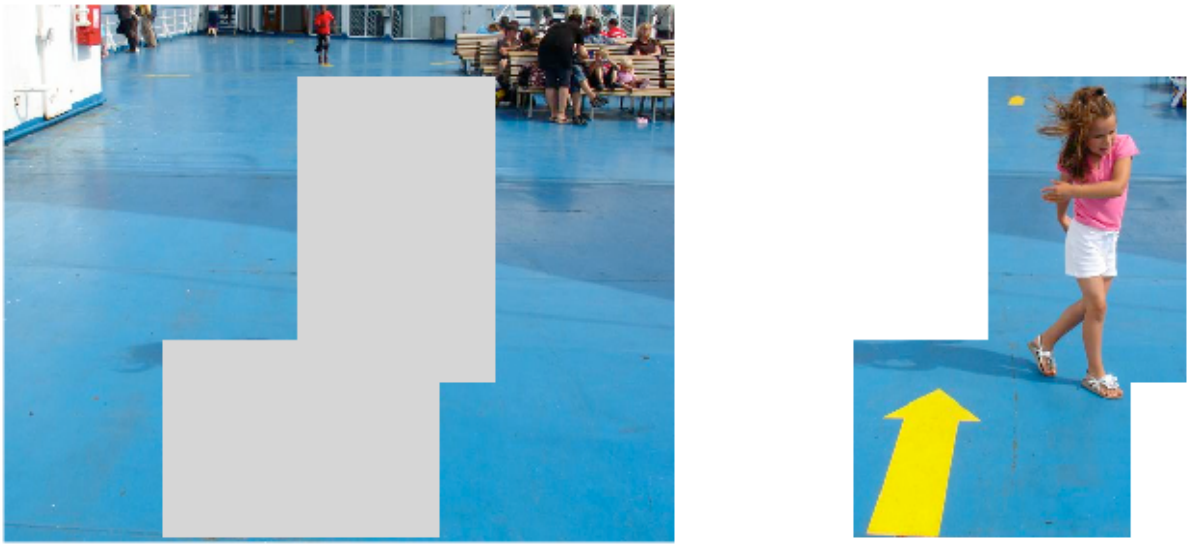

Figure 6: Technique du masque appliqué à un objet d'affichage

MANIPULER UNE OCCURRENCE

5. RENDRE UNE OCCURRENCE MOBILE

A - Automatiquement (ex. : curseur personnalisé)

1. Nommez un objet d'affichage (ex. : **curseur**)

```
curseur.startDrag(true);
```

Remarque : *Pour masquer le curseur (de la souris) d'origine, ajoutez la ligne ci-dessous :*

```
Mouse.hide();
```

B - Manuellement (ex. : glisser-déplacer un objet d'affichage)

1. Nommez un objet d'affichage (ex. : **carte**)

```
carte.addEventListener(MouseEvent.MOUSE_DOWN,deplacer);
carte.addEventListener(MouseEvent.MOUSE_UP,relacher);
function deplacer(evt:MouseEvent) {
  evt.currentTarget.startDrag(false);
  stage.addEventListener(MouseEvent.MOUSE_UP,relacher);
}
function relacher(evt:MouseEvent) {
  stage.removeEventListener(MouseEvent.MOUSE_UP,relacher);
  stopDrag();
}
```

Remarque : *Les deux lignes d'instructions ci-dessus qui commencent par le mot* ***stage*** *permettent de gérer les cas ou l'objet d'affichage reste collé au curseur de la souris (lorsque l'utilisateur relâche le bouton de la souris alors que le curseur n'est plus sur l'objet d'affichage).*

C - Définir une zone de contrainte

Instanciez le classe **Rectangle()** et ajoutez un deuxième paramètre à la méthode **startDrag()**.

```
var zoneDeContrainte:Rectangle = new Rectangle (30,40,300,50);
carte.addEventListener(MouseEvent.MOUSE_DOWN,deplacer);
carte.addEventListener(MouseEvent.MOUSE_UP,relacher);
function deplacer(evt:MouseEvent) {
  evt.currentTarget.startDrag(false,zoneDeContrainte);
  stage.addEventListener(MouseEvent.MOUSE_UP,relacher);
}
```

```
function relacher(evt:MouseEvent) {
  stage.removeEventListener(MouseEvent.MOUSE_UP,relacher);
  stopDrag();
}
```

Remarque : *La première ligne du script ci-dessus et l'adjonction du mot* ***zoneDeContrainte*** *à la méthode* ***startDrag()*** *suffisent à contraindre le déplacement. Les valeurs 30 et 40 correspondent au coin supérieur gauche de la zone mesurant 300 pixels de largeur et 50 pixels de hauteur dans laquelle l'occurrence peut être déplacée. Précisons que c'est le point d'alignement d'un symbole qui ne peut sortir de la zone définie.*

6. CHANGER LE NIVEAU (LA PROFONDEUR, LE PLAN) D'UN OBJET D'AFFICHAGE

Dans certains cas, vous aurez besoin de placer des objets d'affichage au premier plan (ou en arrière plan, ou sur des plans indexés (précis)), vous aurez alors besoin d'utiliser la méthode **setChildIndex()**.

```
setChildIndex(carte1,this.numChildren-1)
```

Le premier paramètre doit être un nom d'objet d'affichage (**carte1** dans notre exemple). Le deuxième paramètre doit être un nombre dont la valeur est inférieure au nombre d'objets d'affichage contenus dans le conteneur ciblé (il n'y en a pas dans le script ci-dessus).

Dans notre exemple, nous cherchons à placer un objet d'affichage au premier plan, c'est pourquoi, nous utilisons l'expression **numChildren-1**.

La propriété **numChildren** permet en effet de connaître le nombre d'objets d'affichage contenus dans un conteneur d'objet(s) d'affichage. Dans l'exemple ci-dessus, le mot clé **this** fait référence à l'animation. Rappelons également que le premier objet d'affichage placé sur la scène porte l'index 0 et non 1.

Remarque : *Si vous avez placé un objet d'affichage dans un autre à l'aide de la méthode* ***addChild()****, vous devez alors préfixer la méthode* ***setChildIndex()*** *par le nom du conteneur d'objet d'affichage parent comme nous le démontre l'exemple ci-dessous.*

```
setChildIndex(carte1,this.numChildren-1)
```

Dans certains cas, vous aurez besoin de convertir une expression en une instance de type **DisplayObject()** afin que la méthode **setChildIndex()** accepte le premier paramètre (la référence à l'objet d'affichage lui-même). Vous aurez donc besoin de procéder à une conversion telle que celle-ci :

```
setChildIndex(DisplayObject(carte1),this.numChildren-1);
```

7. INTERPOLATION DYNAMIQUE D'UN OBJET D'AFFICHAGE

*Rappel : Evitez d'utiliser la classe **Tween()** native de l'AS3, faites appel aux classes **TweenMax()** ou **Tweener()** (Caurina) (ou bien même une autre, il en existe plus d'une dizaine). Dans ce mémento, nous utiliserons le package **greensocl** qui contient les classes principales **TweenLite()**, **TweenFilterLite()** et **TweenMax()**.*

1. Télécharger le package **greensock** à l'adresse ci-dessous :
   - http://www.greensock.com/tweenmax/ (Lien en haut à droite de la page).
2. Placez le dossier **com** dans le même dossier que celui qui contient votre fichier .fla.
3. Placez un objet d'affichage sur la scène et nommez le **balle**.
4. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:250,y:150,rotation:45});
```

Vous pouvez utiliser les propriétés énoncées dans les premières pages de ce mémento en lieu et place des propriétés **x**, **y** et **rotation** de l'exemple ci-dessus. La durée de l'interpolation est toujours exprimée en secondes et non en millisecondes (2 secondes dans notre exemple).

A - Ajouter un effet de type rebond, élastique ou retour de force

1. Importez l'ensemble des classes du package **easing** (2eme ligne du script ci-dessous).
2. Dans la méthode **to()**, Ajoutez la propriété **ease** en précisant l'une des classes suivantes : **Linear**, **Elastic**, **Back**, **Bounce**
3. Spécifiez l'emplacement de l'effet : **easeIn** (au début), **easeOut** (à la fin) ou **easeInOut** (au début et à la fin de l'interpolation).

```
import com.greensock.*;
import com.greensock.easing.*;
TweenMax.to(balle, 2, {scaleX:5,scaleY:5, ease:Bounce.easeOut});
```

B - Retarder le lancement de l'interpolation

1. Ajoutez la propriété **delay** en spécifiant une valeur exprimée en secondes.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:350,delay:2});
```

C - Exécuter du code lorsque l'interpolation est terminée

1. Ajoutez la propriété **onComplete** suivie du nom de la fonction appelée à la fin de l'interpolation.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:350,onComplete:finInterpolation});
function finInterpolation() {
  balle.scaleX=0.3;
  balle.scaleY=0.3;
}
```

D - Exécuter du code au lancement de l'interpolation

1. Ajoutez la propriété **onStart** suivie du nom de la fonction appelée à la fin de l'interpolation.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:350,delay:3,onStart:finInterpolation});
function finInterpolation() {
  balle.scaleX=0.3;
  balle.scaleY=0.3;
}
```

Remarque : *Une telle propriété (ci-dessus) est principalement utile lorsque vous faites appel à la propriété* ***delay****.*

E - Exécuter du code durant l'exécution de l'interpolation

1. Ajoutez la propriété **onUpdate** suivie du nom de la fonction appelée à la fin de l'interpolation.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:350,onUpdate:finInterpolation});
function finInterpolation() {
  balle.rotation++;
}
```

F – Interpoler plusieurs occurrences en même temps

1. Utilisez la méthode **allTo()** en passant en paramètre de nom d'occurrences, un tableau.

```
import com.greensock.*;
TweenMax.allTo([balle1,balle2,balle3], 2, {x:350});
```

G – Réexecuter une interpolation dans le sens inverse

1. Utilisez les propriétés **yoyo, repeat et repeatDelay. yoyo** indique si vous souhaitez obtenir un retour d'interpolation, et **repeatDelay** le temps avant le retour.

```
import com.greensock.*;
TweenMax.to(balle, 2, {x:350, yoyo:true,repeat:1,repeatDelay:5});
```

CONSTRUCTION DYNAMIQUE

8. MANIPULER PLUSIEURS OCCURRENCES SUR LA SCENE

A – Boucle for() et mot clé this avec une paire de crochets (this[])

Lorsque vous aurez placé plusieurs occurrences sur la scène et que vous souhaiterez les manipuler (changer une de leurs propriétés, leur assigner un écouteur, etc.), vous pourrez alors utiliser la syntaxe suivante :

```
for (var i=1; i<=5; i++) {
  this["carte" + i].rotation = 45;
}
```

Dans cet exemple, nous avons 5 occurrences sur la scène qui s'intitulent carte1, carte2…carte5. Grace à **this** et la **paire de crochets**, nous pouvons cibler nos occurrences par concaténation de la chaine "carte" qui constitue le préfixe commun à chaque occurrence et de la variable i qui va prendre successivement les valeurs 1 à 5.

*Remarque : Dans le cas où vous auriez une occurrence intitulée **jeu** qui contiendrait les occurrences carte1…carte5, il faudrait alors écrire jeu["carte"+i].*

Nous allons à présent rédiger un script qui va permettre de rendre cliquable chaque occurrence tout en évitant de copier-coller vos lignes de type :

```
carte1.addEventListener(MouseEvent.MOUSE_DOWN,nomFonction)
```

Repartons simplement du script ci-dessus :

```
for (var i=1; i<=5; i++) {
  this["carte" + i].numero = i;
  this["carte" + i].addEventListener(MouseEvent.MOUSE_DOWN,clic);
}

function clic(evt:MouseEvent) {
  trace(evt.currentTarget.numero);
}
```

Vous observerez que nous avons ajouté une ligne sur laquelle nous assignons une valeur unique à chaque occurrence (…numero=i). Cette dernière va nous permettre d'identifier et distinguer chaque occurrence.

Remarque : Dans le cas où vous auriez besoin d'utiliser des tableaux, pensez à nommer votre première occurrence avec le suffixe 0 et non pas 1, ce qui donnerait carte0.

B – Utilisation de la propriété target

Lorsque plusieurs occurrences sur trouvent à l'intérieur d'une autre, il est possible d'utiliser le mot clé **target** en ne plaçant un écouteur que sur cette dernière.

Exemple : Afin que vous puissiez comprendre plus facilement le principe nous suivez la procédure suivante :

1. Placez un symbole sur la scène et ne nommez pas obligatoirement l'occurrence obtenue.
2. À l'aide de l'outil Flèche noire et de la touche Alt maintenue, déplacez plusieurs fois votre occurrence. Vous devriez obtenir plusieurs occurrences sur la scène (elles pourraient toutes être issues de symboles différents, cela reviendrait au même).
3. Sélectionnez toutes les occurrences sur la scène et transformez votre sélection en un symbole (à l'aide de la touche de fonction F8).
4. Nommez l'occurrence obtenue (ex. **tribu**).
5. Cliquez sur la scène, puis dans la fenêtre Actions avant de saisir le script ci-dessous :

```
tribu.addEventListener(MouseEvent.MOUSE_DOWN,masquer);
function masquer(evt:MouseEvent) {
  evt.target.visible = false;
}
```

Vous pouvez constater que chaque occurrence cliquée disparaît alors que l'écouteur a été défini pour la seule et unique qui se trouve sur la scène et s'intitule **tribu**. C'est grâce au mot clé **target** qui permet de cibler l'occurrence visée à l'aide du curseur de la souris lorsqu'elle se trouve à l'intérieur d'une autre.

9. PLACER UN SYMBOLE DYNAMIQUEMENT SUR LA SCENE

1. Créer un symbole en cliquant sur le bouton Avancé (figure 7).
2. Cliquez sur la case à cocher "Exporter pour ActionScript" (figure 8).
3. Donnez un nom de Classe (nom qui commence par une majuscule).
4. Utilisez le script ci-dessous

```
var maCarte:Carte = new Carte();
addChild(maCarte);
maCarte.x=100;
maCarte.y=120;
```

Après l'exécution du script ci-dessus, vous obtenez un objet d'affichage.

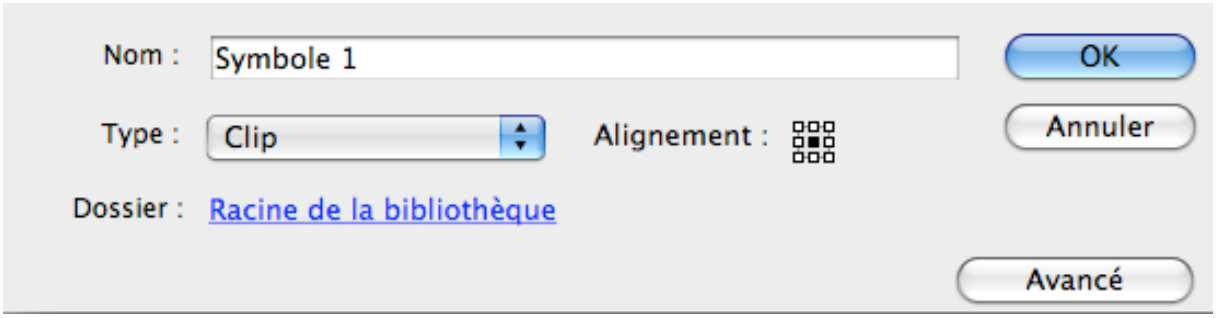

Figure 7 : Cliquez sur le bouton Avancé afin d'afficher la fenêtre dans son intégralité.

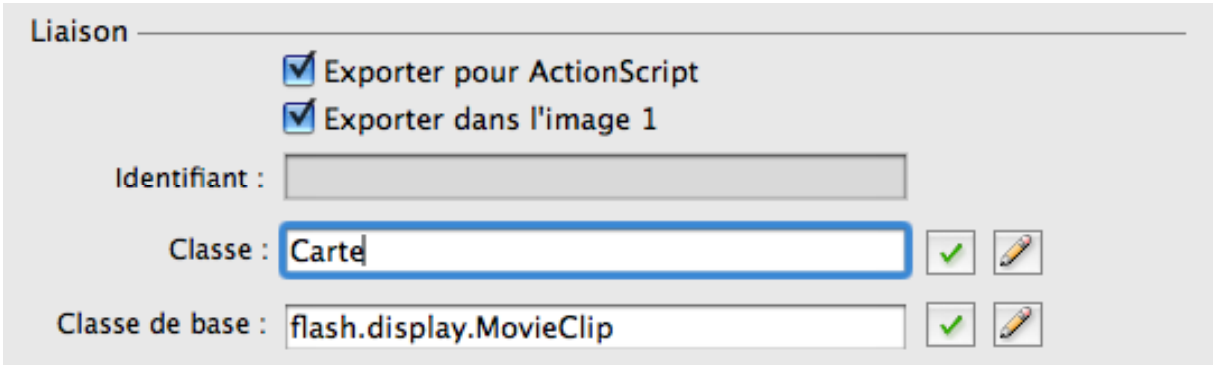

Figure 8 : Cliquez sur la case à cocher "Exporter pour ActionScript" afin de pouvoir donner un nom de classe.

Si vous avez besoin d'effectuer ce réglage pour un symbole déjà existant, effectuez un clic droit sur le symbole de votre choix dans la Bibliothèque puis cliquez sur la commande "Liaison" (si vous travaillez sur Flash CS3) ou "Propriétés" (si vous travaillez sur Flash CS4). Vous accéderez alors à la fenêtre de la figure 8.0.

Remarque : *Gardez toujours à l'esprit que les symboles avec liaison se chargent sur la première image, avant même qu'une jauge de chargement d'animation ne puisse s'afficher. Cela explique pourquoi vos jauges de préchargement d'animation ne commencent pas toujours à partir 0.*

10. UTILISATION DE LA CLASSE SPRITE() POUR REGROUPER DES OBJETS D'AFFICHAGE

Lorsque vous placerez dynamiquement des objets d'affichage sur la scène, vous aurez tout intérêt à les regrouper par fonctionnalités au sein de conteneurs d'objets d'affichage de type **Sprite**. Pour ce faire, créez préalablement sur la scène, des instances de la classe **Sprite()**.

```
var tapis:Sprite = new Sprite();
addChild(tapis);
var carte1:Carte = new Carte();
var carte2:Carte = new Carte();
var carte3:Carte = new Carte();
tapis.addChild(carte1);
tapis.addChild(carte2);
tapis.addChild(carte3);
carte1.x=100;
carte2.x=200;
carte3.x=300;
carte1.y=carte2.y=carte3.y=120;
```

Si vous manipulez à présent l'objet d'affichage **tapis**, vous manipulez globalement alors les objets d'affichage **carte1**, **carte2** et **carte3**. Pour supprimer toutes les cartes, il vous suffit de supprimer le conteneur d'objets d'affichage **tapis** (voir le point 10 ci-dessous).

11. SUPPRIMER UN OBJET D'AFFICHAGE

Faites simplement appel à la méthode **removeChild()** en précisant comme paramètre (à l'intérieur des parenthèses), le nom de l'objet d'affichage.

```
removeChild(tapis);
```

Si vous avez placé un objet d'affichage dans un conteneur d'objet d'affichage, vous devez alors préciser le nom du parent devant la méthode **removeChild()**.

```
tapis.removeChild(carte1);
```

Remarque : *Si vous supprimez un conteneur d'objet d'affichage, les éventuels enfants qu'il contient disparaissent également.*

12. CREER UNE FORME GEOMETRIQUE ELEMENTAIRE

Créez une instance de type **Sprite** et définissez lui des attributs graphiques attendus en fonction de la forme que vous souhaitez obtenir.

A - Créer un rond

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.drawCircle(150,100,30);
addChild(cible);
```

La méthode **lineStyle()** attend les paramètres suivants : L'épaisseur, la couleur et l'opacité du trait.

La méthode **drawCircle()** attend les paramètres suivants : Les positions horizontales et verticales du centre du rond, ainsi que le rayon de la forme.

Pour remplir une forme, vous devez utiliser la méthode **beginFill()** comme le démontre l'exemple ci-dessous :

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.beginFill(0xFF0000,0.5);
cible.graphics.drawCircle(150,100,30);
cible.graphics.endFill();
addChild(cible);
```

Remarque : *La valeur 0.5 (soit 50%) de la méthode **beginFill()** correspond à l'opacité du remplissage.*

B - Créer un carré ou un rectangle

Pour créer un carré ou un rectangle, remplacez la méthode **drawCircle()** de l'exemple ci-dessus, par **drawRect()**. Vous préciserez alors les 4 paramètres suivants :

```
cible.graphics.drawRect(50,100,100,150);
```

Les 4 valeurs de la méthode **drawRect()** correspondent respectivement aux coordonnées x et y du rectangle, accompagnées de la largeur et la hauteur de la forme.

Remarque : *Si vous avez besoin de faire tourner ou mettre à l'échelle une forme de type carré ou rectangle à partir de son centre, vous devrez alors utiliser des valeurs négatives pour placer le quadrilatère. Pour les connaître, vous devez utiliser la moitié de la largeur et la moitié de la hauteur de forme forme. Terminez votre script en replaçant votre forme à l'aide des propriété **x** et **y**.*

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.drawRect(-50,-75,100,150);
cible.x = 230
cible.y = 180
addChild(cible);
```

13. CREER UN TRACE A BASE DE SEGMENTS

Comme nous l'avons fait pour la création de formes fermées telles que le carré ou le rond, commencez par créer une instance de type **Sprite**.

Ensuite, vous devez utilisez les méthodes **moveTo()** pour indiquer le point de départ de votre tracé et **lineTo()** pour tracer une droite jusqu'à un point précis ou bien même **curveTo()** pour tracer une courbe.

A - Tracer une droite

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.moveTo(50,50);
cible.graphics.lineTo(150,50);
addChild(cible);
```

Vous pouvez combiner plusieurs tracés afin de générer une forme.

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.moveTo(50,50);
cible.graphics.lineTo(150,75);
cible.graphics.lineTo(150,150);
cible.graphics.lineTo(100,150);
cible.graphics.lineTo(100,80);
cible.graphics.lineTo(50,65);
cible.graphics.lineTo(50,50);
addChild(cible);
```

Figure 9 : La forme ci-dessous est le résultat de l'exécution des lignes d'instruction ci-dessus.

B - Tracer une courbe

Les paramètres de la méthode **curveTo()** indiquent les coordonnées du point directeur et celles du point d'arrivée de la courbe.

```
var cible:Sprite = new Sprite();
cible.graphics.lineStyle(5,0xFF0000,1);
cible.graphics.moveTo(50,100);
cible.graphics.lineTo(150,100);
cible.graphics.curveTo(175,150,150,200);
cible.graphics.lineTo(50,200);
addChild(cible);
```

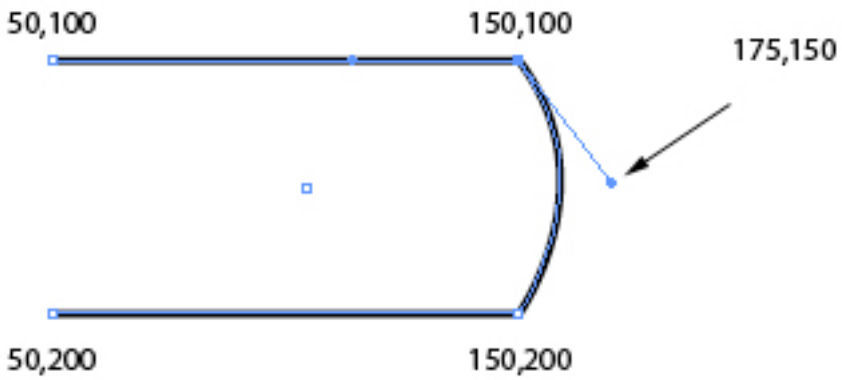

Figure 10 : La forme ci-dessous est le résultat de l'exécution des lignes d'instruction ci-dessus.

Remarque : *Comme vous pouvez le remarquer, il est tout à fait possible de mélanger courbes et droites.*

TRAITEMENT DES MEDIAS

14. LE TEXTE

A - Créer dynamiquement un texte sur la scène

1. Faites appel à la classe **TextField()** et ajoutez votre objet d'affichage sur la scène.

```
var titreDuFilm:TextField = new TextField()
titreDuFilm.text = "Les enfants du paradis";
addChild(titreDuFilm)
```

Vous utiliserez ensuite une ou plusieurs des propriétés suivantes pour affiner la nature et le type de votre texte :

- **type** (ex. : **titreDuFilm.type=TextFieldType.INPUT**)
- **border** (ex. : **titreDuFilm.border=true**)
- **borderColor** (ex. : **titreDuFilm.borderColor=0xFF0000**)
- **background** (ex. : **titreDuFilm.background=true**)
- **backGroundColor** (ex. : **titreDuFilm.backgroundColor=0x00FF00**)
- **autoSize** (ex. : **titreDuFilm.autoSize=TextFieldAutoSize.LEFT**)
- **maxChars** (ex. : **titreDuFilm.maxChars=5);**

Pour régler l'alignement, vous devez utiliser la classe **TextFormat()**.

Remarque : *La propriété **maxChars** sous entend que vous proposez à l'utilisateur de saisir un texte (Cela n'a pas de sens avec un texte dynamique).*

B - Surveiller la saisie de l'utilisateur

1. Placez un texte de type Saisie de texte sur la scène et nommez le **nomUtilisateur**.
2. Saisissez le script ci-dessous dans la fenêtre Actions

```
nomUtilisateur_txt.addEventListener(TextEvent.TEXT_INPUT,enCoursDeModification);
function enCoursDeModification(evt:TextEvent) {
  if (nomUtilisateur_txt.text.indexOf("David")>=0) {
        nomUtilisateur_txt.text="Gagné";
  }
}
```

C - Exécuter une ligne d'instruction lorsqu'un texte de saisie reçoit ou perd le focus

1. Placez un texte de type Saisie de texte sur la scène et nommez le **nomUtilisateur**.
2. Saisissez le script ci-dessous dans la fenêtre Actions

```
nomUtilisateur_txt.addEventListener(FocusEvent.FOCUS_IN,focusActif);
nomUtilisateur_txt.addEventListener(FocusEvent.FOCUS_OUT,focusPerdu);
function focusActif(evt:FocusEvent) {
  nomUtilisateur_txt.borderColor=0xFF0000;
  nomUtilisateur_txt.backgroundColor=0xFFCCCC;
}
function focusPerdu(evt:FocusEvent) {
  nomUtilisateur_txt.borderColor=0x000000;
  nomUtilisateur_txt.backgroundColor=0xFFFFFF;
}
```

D - Faire défiler un texte

Si vous possédez déjà un texte dynamique sur la scène, supprimez les 3 premières lignes du script ci-dessous.

```
var repertoire:TextField = new TextField();
repertoire.height=40;
addChild(repertoire);
repertoire.appendText("Marjorie\n");
repertoire.appendText("Marine\n");
repertoire.appendText("Luna\n");
repertoire.appendText("Mathilde\n");
repertoire.appendText("Melissa\n");
repertoire.appendText("Lucie\n");
repertoire.appendText("Eva");
btLigneApres.addEventListener(MouseEvent.MOUSE_DOWN,abaisser);
btLigneAvant.addEventListener(MouseEvent.MOUSE_DOWN,lever);
function abaisser(evt:MouseEvent) {
  repertoire.scrollV++;
}
function lever(evt:MouseEvent) {
  repertoire.scrollV--;
}
```

Remarque : *L'exemple ci-dessus vous propose le défilement d'un texte verticalement. Si vous utilisez la propriété* ***scrollH****, vous pouvez obtenir un défilement de texte horizontal avec un décalage lettre par lettre.*

E - Rendre les lignes d'un texte dynamique cliquable

1. Placez un texte dynamique sur la scène à l'aide de l'outil Texte ou créez le dynamiquement. Nommez l'objet d'affichage : **repertoire**.

```
repertoire.appendText("Marjorie\n");
repertoire.appendText("Marine\n");
repertoire.appendText("Luna\n");
```

```
repertoire.appendText("Mathilde\n");
repertoire.appendText("Melissa\n");
repertoire.selectable = false
repertoire.addEventListener(MouseEvent.MOUSE_DOWN,numeroLigne);
function numeroLigne(evt:MouseEvent) {
  trace(repertoire.getLineIndexAtPoint(repertoire.mouseX,repertoire.mouseY));
}
```

F - Formater un texte sur la scène

Vous devez faire appel à la classe **TextFormat()** et la méthode **setTextFormat()** de la classe **TextField()**.

```
var titreLivre:TextField = new TextField();
titreLivre.text="Les arbres dans le vent";
titreLivre.autoSize=TextFieldAutoSize.LEFT;
addChild(titreLivre);
var styleNo1:TextFormat = new TextFormat();
styleNo1.size=16;
styleNo1.color=0xEE442F;
titreLivre.setTextFormat(styleNo1);
```

G - Assigner une police de caractères à un texte dynamique

1. Commencez par importer une police dans la bibliothèque de votre animation.
2. Définissez lui une liaison en lui donnant un nom de classe

```
var titreLivre:TextField = new TextField();
titreLivre.text = "Les arbres dans le vent";
titreLivre.embedFonts = true;
titreLivre.autoSize = TextFieldAutoSize.LEFT
addChild(titreLivre);
titreLivre.x=20;
titreLivre.y=20;
var styleNo1:TextFormat = new TextFormat();
styleNo1.size = 16;
styleNo1.color = 0xEE442F;
styleNo1.font="Almagro";
titreLivre.setTextFormat(styleNo1);
```

15. L'IMAGE

A - Charger une image sur la scène

1. Placez une image intitulée image1.png dans le même dossier que votre fichier .fla.

```
var chargeur:Loader = new Loader();
var adresseImage:URLRequest = new URLRequest("image1.png");
chargeur.load(adresseImage);
addChild(chargeur);
```

B - Charger une image et la rendre cliquable

1. Placez une image intitulée image1.png dans le même dossier que votre fichier .fla.

```
var chargeur:Loader = new Loader();
var adresseImage:URLRequest = new URLRequest("image1.png");
chargeur.load(adresseImage);
addChild(chargeur);
chargeur.addEventListener(MouseEvent.MOUSE_DOWN,reduireTaille);
function reduireTaille (evt:MouseEvent) {
  chargeur.scaleX = chargeur.scaleY = 0.5;
  chargeur.x=150;
  chargeur.y=80;
}
```

C - Charger plusieurs images sur la scène

1. Créez un dossier intitulé images que vous placez dans le même dossier que votre fichier .fla.

2. Placez des images intitulées image1.png, image2.png, image3.png, image4.png dans ce dossier intitulé "images".

```
var chargeur:Loader;
var adresseImage:URLRequest;
for (var i:Number=0; i<=3; i++) {
  chargeur = new Loader();
  adresseImage = new URLRequest("images/image"+i+".png");
  chargeur.load(adresseImage);
  chargeur.x=(i*150);
  chargeur.y=120;
  chargeur.alpha = 0.7;
  addChild(chargeur);
  chargeur.addEventListener(MouseEvent.MOUSE_OVER,opacite100);
  function opacite100(evt:MouseEvent) {
        evt.currentTarget.alpha = 1;
  }
  chargeur.addEventListener(MouseEvent.MOUSE_OUT,opacite50);
```

```
  function opacite50(evt:MouseEvent) {
        evt.currentTarget.alpha = 0.5;
  }
}
```

D - Vérifier la fin du chargement d'une image

1. Placez une image intitulée image1.png dans le même dossier que votre fichier .fla.

```
var chargeur:Loader;
var adresseImage:URLRequest = new URLRequest("image1.jpg");
chargeur = new Loader();
chargeur.load(adresseImage);
chargeur.x=150;
addChild(chargeur);
chargeur.contentLoaderInfo.addEventListener(Event.COMPLETE, imageChargee);
function imageChargee(event:Event) {
  trace("Fin du chargement");
}
```

E - Réaliser une jauge de chargement d'une image

1. Placez une image intitulée image1.png dans le même dossier que votre fichier .fla.
2. Placez un texte dynamique sur la scène que vous intitulez **infoCharge**.
3. Placez un clip sur la scène (un rectangle au point d'alignement à gauche) que vous intitulez **barreProgression**.

```
var chargeur:Loader;
var adresseImage:URLRequest = new URLRequest("image1.jpg");
chargeur = new Loader();
chargeur.load(adresseImage);
chargeur.x=150;
addChild(chargeur);
chargeur.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,animerPreloade
r);
function animerPreloader(evt:ProgressEvent) {
  barreProgression.scaleX = evt.bytesLoaded / evt.bytesTotal;
  infoCharge.text = Math.round(evt.bytesLoaded / evt.bytesTotal*100).toString()+" %";
}
```

16. LA VIDEO

A - Définir la source d'une vidéo

1. Faites glisser le composant de type **FLVPlayback** de la palette Composants (Fenêtre/Composants) dans la bibliothèque de votre animation.
2. Placez ce symbole sur la scène et nommez l'objet d'affichage obtenu, exemple : **ecran**
3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
ecran.source = "Luna.flv";
```

Remarque : *Le chemin indiqué ("luna.flv" dans l'exemple ci-dessus) est toujours relatif à la page HTML qui contient le SWF dans lequel se trouve la vidéo que vous placez sur la scène.*

B - Affecter un contrôleur à une vidéo

Pour commencer, ouvrez le dossier qui se trouve à l'adresse suivante :

Adobe Flash CS3/Configuration/FLVPlayback Skins/ActionScript 3.0

Choisissez un fichier que vous placez ensuite dans le dossier qui contient votre animation .fla. Ajoutez les lignes d'instructions ci-dessous à votre script principal.

```
ecran.skin = "ma_skin.swf";
ecran.skinBackgroundColor = 0xAB99CB;
ecran.skinBackgroundAlpha = 0.8;
ecran.skinAutoHide = true;
```

C - Contrôler la lecture et la pause d'une vidéo

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.
2. Placez deux symboles sur la scène et nommez les objets d'affichage obtenus sous les noms **boutonLecture** et **boutonPause**.
3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
ecran.source = "Luna.flv";
boutonLecture.addEventListener(MouseEvent.MOUSE_DOWN,lireVideo);
boutonPause.addEventListener(MouseEvent.MOUSE_DOWN,mettreEnPause);
function lireVideo(evt:MouseEvent) {
   ecran.play();
}
function mettreEnPause(evt:MouseEvent) {
   ecran.pause();
}
```

D - Réaliser un bouton Lecture/Pause

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.

2. Créez un symbole de type clip qui contient deux images-clés. Sur la première des deux images, dessinez l'apparence d'un bouton symbolisant la lecture d'un média. Sur la deuxième, dessinez l'apparence d'un bouton symbolisant la mise en pause d'un média. Placez la fonction **stop()** sur la première image.

3. Placez ce symbole sur la scène et nommez l'objet d'affichage obtenu, ex. : **btLecturePause**.

4. Saisissez le script ci-dessous dans la fenêtre Actions.

```
ecran.source = "Luna.flv";
ecran.autoPlay = true;
btLecturePause.gotoAndStop(2);
btLecturePause.buttonMode = true
btLecturePause.addEventListener(MouseEvent.MOUSE_DOWN,lireVideo);
function lireVideo(evt:MouseEvent) {
   if (ecran.state=="playing") {
      ecran.pause();
      boutonLecturePause.gotoAndStop(1);
   } else {
      ecran.play();
      boutonLecturePause.gotoAndStop(2);
   }
}
```

E - Placer dynamiquement une vidéo sur la scène

1. Faites glisser le composant FLVPlayback de la palette Composants (Fenêtre/Composants) dans la bibliothèque de votre animation.

2. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import fl.video.FLVPlayback;
var ecran:FLVPlayback = new FLVPlayback();
addChild(ecran);
ecran.source = "Luna.flv";
ecran.setSize(384,216);
ecran.x = 8;
ecran.y = 8;
```

F - Déplacer la tête de lecture de la vidéo

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.

2. Placez trois symboles sur la scène et nommez les objets d'affichage obtenus, **btInstant1**, **btInstant2** et **btInstant3**.

3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
ecran.source = "Luna.flv";
btInstant1.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant1);
btInstant2.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant2);
btInstant3.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant3);
function allerInstant1(evt:MouseEvent) {
   ecran.seek(2);
}
function allerInstant2(evt:MouseEvent) {
   ecran.seek(5);
}
function allerInstant3(evt:MouseEvent) {
   ecran.seek(7);
}
```

Autre script possible :

```
btInstant1.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant);
btInstant2.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant);
btInstant3.addEventListener(MouseEvent.MOUSE_DOWN,allerInstant);
btInstant1.temps=2;
btInstant2.temps=5;
btInstant3.temps=7;
function allerInstant(evt:MouseEvent) {
   ecran.seek(evt.currentTarget.temps);
}
```

G - Gérer les repères dans une vidéo

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.
2. Placez sur la scène, un texte dynamique et nommez l'objet d'affichage obtenu, ex. : **messageVideo**
3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import fl.video.MetadataEvent;
ecran.source = "Luna.flv";
ecran.addASCuePoint(2,"Oui, oui");
ecran.addASCuePoint(6,"C'est qui ?");
ecran.addASCuePoint(10,"A quelle heure ?");
ecran.addEventListener(MetadataEvent.CUE_POINT,repereAtteint);
function repereAtteint(evt:MetadataEvent) {
   messageVideo.text = evt.info. name;
}
```

H - Détecter la fin d'une vidéo

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.
2. Placez sur la scène, un texte dynamique et nommez l'objet d'affichage obtenu, ex. : **messageInformation**

3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import fl.video.VideoEvent;
ecran.addEventListener(VideoEvent.COMPLETE,lectureTerminee);
function lectureTerminee(evt:VideoEvent) {
  messageInformation.text = "Fin";
}
```

I - Réaliser une jauge de lecture

1. Placez sur la scène, un symbole de type clip qui représente un rectangle dans le sens de la largeur (alignement du point d'ancrage à gauche) et nommez l'objet d'affichage obtenu, ex. : **jaugeLecture**.

2. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import fl.video.VideoEvent;
ecran.source = "Luna.flv";
ecran.playheadUpdateInterval = 100;
ecran.addEventListener(VideoEvent.PLAYHEAD_UPDATE,miseAJourJauge);
ecran.addEventListener(VideoEvent.COMPLETE,lectureTerminee);
function miseAJourJauge(evt:VideoEvent) {
   jaugeLecture.scaleX = ecran.playheadTime/ecran.totalTime;
}
function lectureTerminee(evt:VideoEvent) {
   jaugeLecture.scaleX = 0;
}
```

J - Connaître l'état de lecture d'une vidéo

1. Nommez un objet d'affichage de type FLVPlayback sur la scène, ex. : **ecran**.

2. Affectez un contrôleur à l'objet d'affichage.

3. Placez sur la scène, un texte dynamique et nommez l'objet d'affichage obtenu, ex. : **messageInfo**

4. Saisissez le script ci-dessous dans la fenêtre Actions.

```
import fl.video.VideoEvent;
ecran.source = "Luna.flv";
ecran.addEventListener(VideoEvent. PAUSED_STATE_ENTERED,pauseActivee);
ecran.addEventListener(VideoEvent.PLAYING_STATE_ENTERED,lectureActivee);
function pauseActivee(evt:VideoEvent) {
   messageInfo.text = "Pause";
}
function lectureActivee(evt:VideoEvent) {
   messageInfo.text = "";
}
```

K - Afficher le temps écoulé d'une vidéo

1. Placez simplement sur la scène, un texte dynamique et nommez l'objet d'affichage obtenu, ex. : **messageInfo**.

2. Saisissez le script ci-dessous dans la fenêtre Actions

```
import fl.video.VideoEvent;
ecran.source = "Luna.flv";
var positionTete:Number;
var heures:*;
var minutes:*;
var secondes:*;
ecran.addEventListener(VideoEvent.PLAYHEAD_UPDATE,lectureEnCours);
ecran.playheadUpdateInterval = 500;
function lectureEnCours(evt:VideoEvent) {
   positionTete = ecran.playheadTime;
   heures = Math.floor(positionTete/3600);
   minutes = Math.floor(positionTete/60);
   secondes = Math.floor(positionTete%60);
   heures= heures<=9 ? "0"+heures : heures;
   minutes= minutes<=9 ? "0"+minutes : minutes;
   secondes= secondes<=9 ? "0"+secondes : secondes;
   messageInfo.text = heures+":"+minutes+":"+secondes;
}
```

17. LE SON

A - Lecture automatique d'un son

1. Placez un son intitulé Chopin.mp3 dans le même dossier que votre fichier .fla.

```
var ecoute:Sound = new Sound();
var adresseFichierSon:URLRequest = new URLRequest("Chopin.mp3");
ecoute.load(adresseFichierSon);
ecoute.play();
```

Remarque : *Le chemin indiqué ("Chopin.mp3" dans l'exemple ci-dessus) est toujours relatif à la page HTML qui contient le SWF dans lequel se trouve la vidéo que vous placez sur la scène.*

B - Lire et arrêter un son à partir de deux boutons

1. Placez un son intitulé Chopin.mp3 dans le même dossier que votre fichier .fla.

2. Placer deux objets d'affichage de type Clip sur la scène que vous intitulez **boutonLecture** et **boutonArret**.

```
var ecoute:Sound = new Sound();
var adresseFichierSon:URLRequest = new URLRequest("Chopin.mp3");
var piste1:SoundChannel;
ecoute.load(adresseFichierSon);
boutonLecture.addEventListener(MouseEvent.MOUSE_DOWN,lireSon);
boutonArret.addEventListener(MouseEvent.MOUSE_DOWN,arreterSon);
function lireSon(evt:MouseEvent) {
  piste1 = ecoute.play();
}
function arreterSon(evt:MouseEvent) {
  piste1.stop();
}
```

C - Réaliser un bouton Lecture/Pause

1. Placez un son intitulé Chopin.mp3 dans le même dossier que votre fichier .fla.

2. Placez un objet d'affichage de type Clip sur la scène que vous nommez **boutonLecturePause**.

3. Placez également un texte dynamique que vous nommez **etiquetteBouton**.

```
var ecoute:Sound = new Sound();
var adresseFichierSon:URLRequest = new URLRequest("Chopin.mp3");
var piste1:SoundChannel;
ecoute.load(adresseFichierSon);
etiquetteBouton.text="Lecture";
var positionTete:Number = 0;
```

```
boutonLecturePause.addEventListener(MouseEvent.MOUSE_DOWN,lecturePauseSon);
function lecturePauseSon(evt:MouseEvent) {
  if (etiquetteBouton.text=="Lecture") {
        piste1 = ecoute.play(positionTete);
        etiquetteBouton.text="Pause";
  } else {
        positionTete=piste1.position;
        etiquetteBouton.text="Lecture";
        piste1.stop();
  }
}
```

D - Détecter la fin d'un son

1. Placez un son intitulé Chopin.mp3 dans le même dossier que votre fichier .fla.
2. Placez un objet d'affichage de type Clip sur la scène que vous nommez **boutonLecture**.

```
var ecoute:Sound = new Sound();
var adresseFichierSon:URLRequest = new URLRequest("Chopin.mp3");
ecoute.load(adresseFichierSon);
var piste1:SoundChannel;
boutonLecture.addEventListener(MouseEvent.MOUSE_DOWN,lectureSon);
function lectureSon(evt:MouseEvent) {
  piste1 = ecoute.play();
  boutonLecture.mouseEnabled = false;
  boutonLecture.alpha=0.7;
  piste1.addEventListener(Event.SOUND_COMPLETE,finDuSon);
}
function finDuSon(evt:Event) {
  boutonLecture.mouseEnabled = true;
  boutonLecture.alpha=1;
  piste1.removeEventListener(Event.SOUND_COMPLETE,finDuSon);
}
```

E - Exécuter une action lors de l'écoute d'un son

1. Placez un son intitulé Chopin.mp3 dans le même dossier que votre fichier .fla.
2. Placez un objet d'affichage de type Clip sur la scène que vous nommez **boutonLecture**.
3. Placez également un texte dynamique que vous nommez **affichageTemps**.

```
var ecoute:Sound = new Sound();
var adresseFichierSon:URLRequest = new URLRequest("Piano1.mp3");
ecoute.load(adresseFichierSon);
var piste1:SoundChannel;
var positionTete:Number = 0;
var heures:*;
var minutes:*;
```

```
var secondes:*;
boutonLecture.addEventListener(MouseEvent.MOUSE_DOWN,lectureSon);
function lectureSon(evt:MouseEvent) {
  piste1 = ecoute.play();
  boutonLecture.addEventListener(Event.ENTER_FRAME,sonEnCours);
}
function sonEnCours (evt:Event) {

  positionTete = piste1.position/1000;
  heures = Math.floor(positionTete/3600);
  minutes = Math.floor(positionTete/60);
  secondes = Math.floor(positionTete%60);
  heures= heures<=9 ? "0"+heures : heures;
  minutes= minutes<=9 ? "0"+minutes : minutes;
  secondes= secondes<=9 ? "0"+secondes : secondes;
  affichageTemps.text = heures+":"+minutes+":"+secondes;
}
```

TRAITEMENT DES DONNEES

18. MANIPULER UN FICHIER XML

A - Charger un fichier XML et lire la valeur d'un nœud

Contenu du fichier XML utilisé dans l'exemple de script ci-dessous :

```
<Etudiants>
  <Etudiant>Marc</Etudiant>
  <Etudiant>Lucie</Etudiant>
  <Etudiant>Jean</Etudiant>
  <Etudiant>Luna</Etudiant>
  <Etudiant>Eva</Etudiant>
  <Etudiant>Laurent</Etudiant>
</Etudiants>
```

1. Placez un fichier XML intitulé **etudiants.xml** dans le même dossier que votre fichier fla.

2. Placez un texte dynamique sur la scène que vous intitulez **affichage**.

```
var chargeur:URLLoader = new URLLoader();
var adresseFichier:URLRequest = new URLRequest("etudiants.xml");
var lesDonneesXML:XML;
chargeur.load(adresseFichier);
chargeur.addEventListener(Event.COMPLETE, chargementTermine);
function chargementTermine(evt:Event) {
  lesDonneesXML = new XML(chargeur.data);
  affichage.text = lesDonneesXML.Etudiant[3];
}
```

B - Charger un fichier XML et lire l'attribut d'un nœud

Contenu du fichier XML utilisé dans l'exemple de script ci-dessous :

```
<Etudiants>
  <Etudiant nom="Luna"/>
  <Etudiant nom="Marine"/>
  <Etudiant nom="Marjorie"/>
  <Etudiant nom="David"/>
  <Etudiant nom="Olivier"/>
  <Etudiant nom="Fabienne"/>
</Etudiants>
```

1. Placez un fichier XML intitulé **etudiants.xml** dans le même dossier que votre fichier .fla.

2. Placez un texte dynamique sur la scène que vous intitulez **affichage**.

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

```
var adresseFichier:URLRequest = new URLRequest("etudiants.xml");
var lesDonneesXML:XML;
chargeur.load(adresseFichier);
chargeur.addEventListener(Event.COMPLETE, chargementTermine);
function chargementTermine(evt:Event) {
  lesDonneesXML = new XML(chargeur.data);
  affichage.text = lesDonneesXML.Etudiant[3].@nom;
}
```

Vous remarquerez qu'en comparaison de l'exemple précédent, seule la ligne suivante change :

```
affichage.text = lesDonneesXML.Etudiant[3].@nom;
```

C - Rechercher une information dans un arbre XML

Utilisez des parenthèses (en guise de chemin) qui contiennent un opérateur.

Contenu du fichier XML utilisé dans l'exemple de script ci-dessous :

```
<Cinema>
<Film nom="Le Papillon">
  <Realisateur>Philippe MUYL</Realisateur>
  <ActeurPrincipal sexe="M">Michel SERRAULT</ActeurPrincipal>
  <Annee>2002</Annee>
  <Genre>Comédie dramatique</Genre>
  <Pays>France</Pays>
</Film>
<Film nom="Erin Brockovich">
  <Realisateur>Steven SODERBERGH</Realisateur>
  <ActeurPrincipal sexe="F">Julia ROBERTS</ActeurPrincipal>
  <Annee>2000</Annee>
  <Genre>Dramatique</Genre>
  <Pays>US</Pays>
</Film>
<Film nom="The Bourne Ultimatum">
  <Realisateur>Paul GREENGRASS</Realisateur>
  <ActeurPrincipal sexe="M">Matt DAMON</ActeurPrincipal>
  <Annee>2007</Annee>
  <Genre>Action</Genre>
  <Pays>US</Pays>
</Film>
</Cinema>
```

Exemples ponctuels :

Trouver le nom du ou des réalisateurs qui ont sorti un film en 2007.

```
trace(lesDonneesXML.Film.(Annee==2007).Realisateur);
```

Trouver le nom de l'acteur principal qui joue dans le film intitulé "Le Papillon".

```
trace(lesDonneesXML.Film.(@nom=="Le Papillon").ActeurPrincipal);
```

Remarque : *Pour rechercher la valeur d'un nœud, effectuez une simple comparaison entre (dans) les parenthèses qui caractérisent une recherche, appelée également "filtre". Pour rechercher la valeur d'un attribut, ajoutez le signe arobase devant votre recherche.*

Exemple complet :

```
var chargeur:URLLoader = new URLLoader();
var adresseFichier:URLRequest = new URLRequest("cinema.xml");
var lesDonneesXML:XML;
chargeur.load(adresseFichier);
chargeur.addEventListener(Event.COMPLETE, chargementTermine);
function chargementTermine(evt:Event) {
  lesDonneesXML=new XML(chargeur.data);
  trace(lesDonneesXML.Film.(Annee==2007));
  trace(lesDonneesXML.Film.(Annee==2007).Realisateur);
  trace(lesDonneesXML.Film.(@nom=="Le Papillon").ActeurPrincipal.@sexe);
  trace(lesDonneesXML.Film.(@nom=="Le Papillon").ActeurPrincipal);
}
```

Remarque *: Vous pouvez utiliser les opérateurs de comparaison suivants :* ***==*** *(égalité)* ***!=*** *(différent de)* ***>*** *(supérieur à)* ***<*** *(inférieur à). Vous pouvez aussi utiliser la méthode* ***indexOf()****.*

Exemple :

```
trace(lesDonneesXML.Film.(Realisateur.indexOf("Philippe")>=0).Realisateur);
```

D - Manipuler le résultat d'une requête via la classe XMLList

Placez le résultat d'une requête dans une instance de type **XMLList**.

```
var requete:XMLList;
requete=new XMLList(lesDonneesXML.Film.@nom);
```

Exemple complet :

```
var chargeur:URLLoader = new URLLoader();
var adresseFichier:URLRequest = new URLRequest("cinema.xml");
var lesDonneesXML:XML;
var requete:XMLList;
chargeur.load(adresseFichier);
chargeur.addEventListener(Event.COMPLETE, chargementTermine);
function chargementTermine(evt:Event) {
  lesDonneesXML=new XML(chargeur.data);
```

```
  requete=new XMLList(lesDonneesXML.Film.@nom);
  for each (var individu:String in requete) {
        trace(individu);
  }
}
```

Remarque : *Le contenu du fichier* ***cinema.xml*** *est présenté sur la page précédente.*

```
trace(lesDonneesXML.children().length());
```

Remarque : *Le contenu du fichier* ***cinema.xml*** *est présenté sur la page précédente.*

19. CHARGER DES DONNEES PROVENANT D'UNE BASE DE TYPE MYSQL

A - Charger des données

1. La page PHP de l'adresse ci-dessous doit renvoyer des valeurs formatées comme suit : **nomVariable=une valeur&nomVariable=une valeur&** etc…

2. Trois textes dynamiques intitulés **prenom_inst**, **nom_inst** et **adresseMail_inst** doivent être placés sur la scène.

```
var chargeur:URLLoader = new URLLoader();
var groupeVariables:URLVariables;
var adresse:URLRequest = new
URLRequest("http://www.yazo.net/racine/Tests/requete.php");
chargeur.load(adresse);
chargeur.addEventListener(Event.COMPLETE, donneesLoadees);
function donneesLoadees(event:Event):void {
  var groupeVariables:URLVariables = new URLVariables(chargeur.data);
  prenom_inst.text = groupeVariables.prenom;
  nom_inst.text=groupeVariables.nom;
  adresseMail_inst.text=groupeVariables.adressemail;
}
```

B - Envoyer des données

1. La page PHP de l'adresse ci-dessous ne doit pas obligatoirement renvoyer des valeurs formatées comme suit : **nomVariable=une valeur&nomVariable=une valeur&** etc… En revanche, vous devez prévoir un traitement et/ou un enregistrement des données que vous allez envoyer depuis Flash vers une page PHP.

N.B. : Cet exemple n'affiche aucun résultat dans la fenêtre Sortie de Flash.

```
var groupeVariables:URLVariables = new URLVariables();
groupeVariables.prenom="david";
groupeVariables.nom="TARDIVEAU";
```

```
groupeVariables.age="38";
var chargeur:URLLoader = new URLLoader();
var adresse:URLRequest = new URLRequest(adresse d'une page PHP);
adresse.data = groupeVariables;
chargeur.load(adresse);
```

C - Envoyer et charger des données

1. La page PHP de l'adresse ci-dessous doit renvoyer des valeurs formatées comme suit : **nomVariable=une valeur&nomVariable=une valeur&** etc…

N.B. : Cet exemple affiche le résultat dans la fenêtre Sortie de Flash. Référez-vous au script précédent pour de plus amples informations.

```
var groupeVariables:URLVariables = new URLVariables();
groupeVariables.prenom="david";
var chargeur:URLLoader = new URLLoader();
var adresse:URLRequest = new
URLRequest("http://www.yazo.net/racine/Tests/requete2.php");
adresse.data = groupeVariables;
chargeur.load(adresse);
chargeur.addEventListener(Event.COMPLETE, donneesLoadees);
function donneesLoadees(event:Event):void {
  trace(chargeur.data);
}
```

20. ENVOYER UN MAIL A PARTIR D'UNE ANIMATION FLASH

A - À partir du logiciel de messagerie de l'utilisateur

Remplacez l'adresse **david@yazo.net** de l'exemple ci-dessous par celle de votre choix.

```
btSendMail.addEventListener(MouseEvent.MOUSE_DOWN,envoyerMail);
var adresseDuMail:URLRequest = new URLRequest("mailto:
david@yazo.net?subject=Objet du message");
function envoyerMail(evt:MouseEvent) {
  navigateToURL(adresseDuMail);
}
```

B - À partir de l'animation sans quitter la fenêtre du navigateur

Code de la page PHP appelée :

```
<?php
mail(" david@yazo.net", "Objet de votre mail", $messagemail, "From:$prenom  $nom
($adressemail)");
?>
```

Dans une nouvelle animation Flash, suivez la procédure suivante :

1. Sur la scène, placez 4 textes de saisie que vous nommez **nomExpediteur**, **prenomExpediteur**, **adressemailExpediteur**, **messageExpediteur**)

2. Créez un bouton servant à valider l'envoi du message.

3. Saisissez le script ci-dessous dans la fenêtre Actions.

```
var chargeur:URLLoader = new URLLoader;
var variablesAtransmettre:URLVariables = new URLVariables();
var adresseMail:URLRequest = new URLRequest(adresse de la page PHP);
btEnvoyer.buttonMode = true;
btEnvoyer.addEventListener(MouseEvent.MOUSE_DOWN,envoyerMail);
function envoyerMail(evt:MouseEvent) {
   variablesAtransmettre.nom = nomExpediteur.text.toUpperCase();
   variablesAtransmettre.prenom = prenomExpediteur.text;
   variablesAtransmettre.adressemail = adressemailExpediteur.text;
   variablesAtransmettre.messagemail = messageExpediteur.text;
```

```
   adresseMail.data = variablesAtransmettre;
   chargeur.load(adresseMail);

   messageExpediteur.text = "Message envoyé"
}
```

Remarque : *L'expression "adresse de la page PHP" de l'exemple ci-dessus doit être remplacée par une adresse valide.*

NOTIONS ELEMENTAIRES A LA CONSTRUCTION D'UN ALGORITHME

21. LES VARIABLES

A - Déclarer une variable

Pour déclarer une variable, utilisez le mot clé **var** suivi du nom de la variable (un mot que vous choisissez librement). Il est indispensable de déclarer une variable avant de pouvoir y faire référence (l'initialiser ou la modifier) dans un programme.

```
var nomDeLaVariable;
```

Remarque : *Le nom d'une variable doit être représentatif (ex. : **nombreJoueurs**, **saisieUtilisateur**). N'utilisez donc pas d'abréviations (ex. : **nj** pour **nombreJoueurs**) et ne commencez pas le nom d'une variable par une majuscule. Il ne doit pas contenir de caractères spéciaux (àçï€ñ, etc.), ni d'espace (le caractère underscore ou tiret bas "_" est utilisable).*

Vous devez déclarer vos variables en dehors d'une fonction afin qu'elles puissent être accessibles de n'importe où (de n'importe quelle ligne) dans votre programme. Consultez le point "D - Portée d'une variable" pour comprendre ce mécanisme.

B - Typer une variable

Le principe est simple : Vous allez adjoindre un "mot" précis au nom de votre variable afin de faciliter le débogage de vos programmes et obtenir également l'autocomplétion. Ajoutez simplement deux points derrière le nom de la variable accompagnés du type.

```
var nomDeLaVariable:Number;
```

Voici une liste des types de variables les plus fréquents :

Types	Description de la valeur
int	Un nombre sans chiffres après la virgule (nombre entier).
uint	Un nombre sans chiffres après la virgule (nombre entier non signé).
Number	Un nombre qui peut contenir des virgules.
String	Une chaîne de caractères.
Boolean	Les "valeurs" **true** ou **false**.
Array	Un tableau (une série de valeurs séparées par des virgules).
XML	Une "variable" contenant des données de type XML.
Object	Un bloc d'accolades contenant des paires propriété:"valeur"

Remarque : *Il existe autant de types que de classes, nous vous présentons ici, les types les plus fréquents. Para ailleurs, dans le cas de boucles* ***for()****, privilégiez le type* ***int*** *sur le type* ***Number*** *lorsque c'est possible Le traitement des variables de type* ***int*** *se fait en effet plus rapidement.*

Si vous souhaitez typer une variable afin qu'elle puisse changer de type de contenu, utilisez l'étoile. Si vous faites un tel choix, vous rendez plus difficile la détection de bug.

```
var heure:*=3;
heure="0"+heure;
```

C - Initialiser une variable

On initialise une variable lorsqu'on lui attribue une valeur de départ (initiale). Une variable doit toujours être déclarée avant d'être initialisée. On peut aussi déclarer une variable et l'initialiser en même temps.

```
var nomDuJoeur:String = "David";
```

Remarque : *Dans la mesure ou le typage d'une variable se fait lors de sa déclaration, il est impossible de la typer au moment de son initialisation si vous l'avez déjà préalablement déclarée.*

Si une variable contient un texte, il doit être saisi entre guillemets, un nombre ne doit pas l'être.

```
var nomUtilisateur:String = "David";
var ageUtilisateur:int = 25;
```

En fonction du type choisi, la valeur d'une variable varie, vous ne placerez donc pas uniquement un texte ou un nombre derrière le signe "=", voici quelques exemples complémentaires :

```
var etatMajuscules:Boolean = true;
var gagnants:Array = new Array("Marine","Luna","Marjorie","Luna");
var nomUtilisateur:Object ={prenom:"David",age:"38"};
```

D - Portée d'une variable

Une variable doit être déclarée à l'aide du mot clé **var**. Cela se fait généralement sur les premières lignes d'un programme (en dehors de toutes fonctions). Il arrive parfois qu'une variable soit déclarée au sein d'une fonction, elle sera alors qualifiée de "variable locale". Cette dernière n'est disponible que dans le bloc de code (délimité par des accolades) où elle a été déclarée. Dans l'exemple ci-dessous, le compilateur (Flash) signale une erreur.

```
function preuve() {
  var prenom:String = "David";
  trace(prenom);
}
preuve();
trace(prenom);
```

*Message erreur obtenu : 1120: Accès à la propriété non définie **prenom**.*

La ligne d'instruction sur laquelle figure la commande **trace()** peut s'exécuter, mais pas la dernière ligne. La variable **prenom** n'est pas reconnue ! Normal, elle a été déclarée dans une fonction et non en dehors.

Dans l'exemple ci-dessous, la variable année est accessible depuis n'importe où dans le programme.

```
var annee:int = 2013;
function preuve() {
  var annee:int = 2009;
  trace(annee);
  trace(this.annee);
}
preuve();
trace(annee);
```

Dans la fenêtre Sortie de Flash, on obtient l'affichage des valeurs suivantes : 2009, 2013 et 2013.

***Remarque :** Le mot clé **this** permet de faire référence à l'animation dans son ensemble.*

E - Le type booléen (Boolean)

Il vous arrivera parfois d'avoir besoin d'affecter une valeur à une variable qui ne peut proposer que deux choix. Prenons l'exemple de la propriété **visible** d'un objet d'affichage, nous constatons qu'il n'existe que deux possibilités :

```
carte.visible = false;
```

ou

```
carte.visible = true;
```

Cet "état" est applicable à une variable dès lors que vous l'aurez typé avec le type Boolean.

```
var ecouteEnCours:Boolean = true;
```

Remarque : *Vous pouvez changer/inverser la valeur d'un booléen en utilisant la syntaxe suivante :* ***ecouteEnCours = !ecouteEnCours;***

22. LES TABLEAUX

Un tableau est une "sorte de variable" à plusieurs valeurs. Comme nous l'avons fait pour une variable, nous devons le déclarer puis l'initialiser. Cela peut se faire de deux façons :

A - Déclarer et initialiser un tableau

```
var annees:Array = new Array(1998,1999,2000,2001,2002);
```

ou aussi :

```
var annees:Array = [1998,1999,2000,2001,2002];
```

Remarque : *Pour saisir du texte, vous devez le faire entre guillemets comme le démontre l'exemple ci-dessous. Il est également possible de mélanger des nombres et du texte.*

```
var mois:Array = new Array("Janvier",31,"Février",28,"Mars",31,"Avril",30);
```

Parfois, vous déclarerez un tableau tout en l'initialisant sans aucune valeur (lorsque vous collecterez des informations) :

```
var annees:Array = [];
var annees:Array = new Array();
```

Vous utiliserez alors la méthode **push()** pour injecter une ou des entrées dans le tableau.

B - Lire une entrée de tableau

Faites simplement référence au nom du tableau suivi de l'index de l'entrée entre crochets :

```
trace(annees[2]);
```

Rappelons que la première entrée d'un tableau porte l'index 0. De ce fait si vous souhaitez lire la première entrée du tableau ci-dessous, vous utiliserez la syntaxe suivante :

```
var infoPays:Array = ["Paris","Français",22,"Entre 60 et 65 millions"];
var capitale:String = infoPays[0];
trace(capitale);
```

Résultat de l'affichage

Paris

C- Ajouter des entrées dans un tableau

Faites simplement appel à la méthode **push()** comme dans l'exemple ci-dessous :

```
var annees:Array = new Array();
annees.push(2005,2006,2008)
```

Remarque : *La méthode **push()** peut contenir une seul ou plusieurs valeurs comme le montre l'exemple ci-dessus.*

D - Nombre d'entrées dans un tableau

Faites appel à la propriété **length** comme dans l'exemple ci-dessous :

```
var infosPays:Array = ["Paris","Londres","Milan","Madrid"];
trace(infosPays.length);
```

Résultat de l'affichage

```
4
```

Remarque : *Si vous cherchez donc à lire la dernière entrée d'un tableau, pensez à utiliser la valeur **length-1** et non **length**. Un tableau contenant 5 entrée contiendra 5 index allant de 0 à 4).*

D - Supprimer l'entrée d'un tableau

Faites appel à la méthode **splice()** comme dans l'exemple ci-dessous :

```
var infosPays:Array = ["Paris","Londres","Milan","Madrid"];
trace(infosPays);
infosPays.splice(1,2);
trace(infosPays);
```

Résultat de l'affichage

```
Paris,Londres,Milan,Madrid
Paris,Madrid
```

La première valeur de la méthode **splice()** indique le numéro d'index à partir duquel la suppression doit commencer. Le deuxième paramètre précise le nombre d'entrées à supprimer.

E - Rechercher au sein d'un tableau

Utilisez la méthode **indexOf()**comme vous le feriez avec une chaîne de caractères.

```
var infosPays:Array = ["Paris","Londres","Milan","Madrid"];
```

```
trace(infosPays.indexOf("Milan"));
```

Résultat de l'affichage

```
2
```

En exécutant le script ci-dessus, vous obtenez la valeur 2 (dans la fenêtre Sortie) car la valeur "Milan" se trouve à l'index 2 du tableau. Si la valeur recherchée ("Milan" dans notre exemple) n'est pas trouvée, le programme vous renvoie alors la valeur -1.

F - Parcourir tout un tableau

Utilisez une boucle **for each()** comme le démontre l'exemple ci-dessous :

```
var infosPays:Array = ["Paris","Londres","Milan","Madrid"];
for each (var capitale:String in infosPays) {
  trace(capitale);
}
```

Résultat de l'affichage

```
Paris
Londres
Milan
Madrid
```

Remarque : *Si le tableau que vous parcourez contient des valeurs de différents types, adaptez alors la deuxième ligne du script ci-dessous de la façon suivante :*

```
for each (var capitale:* in infosPays) {
```

G - Parcourir tout un tableau : Méthode forEach() de la classe Array()

La technique ci-dessous fait non seulement appel à une méthode de la classe **Array**, mais également appel à une fonction avec 3 paramètres très précis : La valeur de l'entrée, l'index de l'entrée, la référence à l'ensemble des données du tableau.

```
var famille:Array = [ "Marine", "Luna", "Marjorie","David" ];
function parseuse( element:*, index:int, tableau:Array ) {
  trace("Entrée N°"+index+" : "+ element);
}
famille.forEach( parseuse );
```

Résultat de l'affichage

```
Entrée N°0 : Marine
Entrée N°1 : Luna
Entrée N°2 : Marjorie
Entrée N°3 : David
```

Comme el démontre l'exemple ci-dessus, vous n'êtes pas obligés de faire référence à tous les paramètres de la fonction, mais en revanche, vous devez toutes les déclarer (les citer entre les parenthèses). Cette méthode présente l'avantage de pouvoir effectuer un traitement spécifique avec toutes les entrées d'un tableau.

H - Dupliquer un tableau en le traitant : Méthode map() de la classe Array()

Vous aurez parfois besoin de créer un tableau à partir de données existantes dans un autre tableau. Vous utiliserez alors la méthode **map()** de la classe **Array**.

```
var famille:Array =  [ "Marine", "Luna", "Marjorie","David" ];
function parseuse( element:*, index:int, tableau:Array ):String {
  return element+".jpg";
}
var tableauFichiers:Array = famille.map ( parseuse );
trace( tableauFichiers );
```

Résultat de l'affichage

Marine.jpg,Luna.jpg,Marjorie.jpg,David.jpg

Au début du script, nous avons un tableau intitulé **famille** et nous terminons l'exécution du programme par la création d'un autre tableau intitulé **tableauFichiers**. Ce dernier contient des valeurs d'entrées ayant pour préfixe, celle du tableau **famille**.

I - Les tableaux de propriétés ou associatifs

1. Créez puis typez une variable en **Object**.
2. Utilisez les accolades pour englober les paires **propriété:"Valeur"**

```
var france:Object = {capitale:"Paris",regions:"22",langue:"Français"};
```

Pour lire la valeur d'une propriété, utilisez la syntaxe pointée **nomdelobjet.nomdelapropriete**.

```
trace(france.langue);
```

Résultat de l'affichage

Français

J - Filtrer les données d'un tableau : Méthode filter() de la classe Array()

Pour créer un tableau à partir de quelques entrées provenant d'un autre, vous utiliserez la méthode **filter()** de la classe **Array()**. Cette procédure est très ressemblante de la méthode **map()**, mais elle est différente dans la mesure où le nouveau tableau ne contient pas obligatoirement autant d'entrées que dans le premier tableau.

```
var famille:Array = [ { statut : "Parent", prenom : "David" },{ statut : "Parent", prenom : "Marjorie" },{ statut : "Enfant", prenom : "Marine" },{ statut : "Enfant", prenom : "Luna" } ];
```

```
function connaitreNomsEnfants( element:*, index:int, tableau:Array ):Boolean {
  return ( element.statut=="Enfant" );
}
var enfants:Array = famille.filter ( connaitreNomsEnfants );
function resultatFiltre( element:*, index:int, tableau:Array ):void {
  trace( element.prenom);
}
enfants.forEach( resultatFiltre );
```

Résultat de l'affichage

```
Marine
Luna
```

23. LES STRUCTURES CONDITIONNELLES AVEC IF() ET SWITCH()

Vous disposez de deux techniques pour effectuer un test au sein d'un programme :

- La structure avec la fonction i**f()**
- La structure avec la fonction **switch()**

Quelle que soit la méthode, vous allez pouvoir tester une égalité (**==**), une inégalité (**!=**) ou un dépassement **>** ou **>+** ou **<** ou **<=**).

A - Structure de type if()

Pour que les lignes d'instructions contenues entre les accolades de la structure **if(){}** s'exécutent, il faut que le test renvoie la valeur **true** (le test doit se vérifier, il doit être vrai). Dans l'exemple ci-dessous, nous avons qu'une seule ligne d'instruction contenant la fonction **trace()**.

```
var instant:Date = new Date();
var heure:Number = instant.getHours();
if (heure<=11) {
  trace("Bonjour");
}
```

S'il est moins de 11 heures du matin, le message "Bonjour" s'affichera dans la fenêtre Sortie de Flash. Dans le cas contraire, il ne se passera rien. Pour gérer le cas inverse, vous devez utilisez une structure **if() else**.

Rappelons que vous pouvez bien entendu saisir plusieurs lignes d'instructions à exécuter entre les accolades.

Dans le cas où vous n'avez qu'une ligne d'instruction à exécuter, vous pouvez alors faire appel à la syntaxe suivante :

```
if (heure<=11) trace("Bonjour");
```

B - Structure de type if() else

```
var instant:Date = new Date();
var heure:Number = instant.getHours();
if (heure<=11) {
  trace("Bonjour");
} else {
  trace("Bon après-midi");
}
```

En fonction de l'heure, avant ou après 11 heures, les textes "Bonjour" ou "Bon après-midi" vont s'afficher

C - Les tests multiples

Vous pouvez effectuer plusieurs tests au sein d'une structure conditionnelle, pour cela, vous devez utiliser les paires de signes **&&** (qui signifie *et*) ou **||** (qui signifie *ou*).

```
var instant:Date = new Date();
var heure:Number = instant.getHours();
if (heure>=11 && heure<17) {
  trace("Bon après-midi");
}
```

Dans l'exemple ci-dessus, s'il est plus de 11 heures du matin, mais aussi moins de 17 heures, le message "Bon après-midi" s'affiche dans la fenêtre Sortie de Flash.

```
var instant:Date = new Date();
var heure:Number = instant.getHours();
if (heure>=22 || heure<5) {
  trace("Il fait nuit dehors");
}
```

Dans ce dernier exemple (ci-dessus), s'il est plus de 22 heures ou moins de 5 heures du matin, le texte "Il fait nuit dehors" s'affiche dans la fenêtre Sortie de Flash.

Vous pouvez imbriquer plusieurs tests et mélanger les && et ||.

Remarque : *Sur Mac, le signe | s'obtient à l'aide du raccourci clavier Shift-Alt-L.*

D - La structure switch() case

A la première lecture d'un exemple faisant appel à cette technique de vérification, l'analyse du code n'est pas toujours évidente, voici donc un découpage de sa structure :

```
switch (heure) {
}
```

Vous saisissez le mot **switch** accompagné d'une paire de parenthèses et suivi d'une paire d'accolades. Entre les parenthèses, vous devez saisir le nom de la variable que vous souhaitez comparer. Ensuite, vous ajoutez autant de fois les lignes ci-dessous entre vos accolades.

```
case 12 :
  trace("Heure du déjeuner");
  break;
```

Dans le cas ci-dessous, le message "Heure du déjeuner" va s'afficher si la variable **heure** possède la valeur 12.

Remarque : *Le mot **break** est nécessaire pour indiquer au programme de ne pas continuer les vérifications dans le cas où le test s'est avéré.*

Cette structure de test est utile dans le cas où vous souhaitez exécuter une (ou plusieurs) ligne(s) d'instruction(s) en fonction de cas précis. Ainsi dans l'exemple suivant, il doit être 8 heures, midi, 16 heures ou 17 heures pour qu'un texte s'affiche. Dans le cas contraire, un texte par défaut va s'afficher (dans la fenêtre Sortie de Flash).

```
var instant:Date = new Date();
var heure:Number = instant.getHours();
switch (heure) {

  case 8 :
  trace("Heure du petit déjeuner");
  break;

  case 12 :
  trace("Heure du déjeuner");
  break;

  case 16 :
  trace("Heure du 4 heures");
  break;

  case 19 :
  trace("Heure du repas");
  break;

  default :
  trace("Il est "+heure+" heures");
}
```

24. LES ITERATIONS (BOUCLES)

Remarque : *Dans les points qui vont suivre, vous pourrez dans tous les cas utiliser le mot clé break pour interrompre l'itération.*

Lorsque vous aurez besoin d'exécuter plusieurs fois la ou les mêmes lignes d'instructions, vous n'utiliserez pas un copier-coller pour reproduire plusieurs exemplaire des lignes de code, vous ferez appel à une structure **for()** ou **while()**.

A - La boucle for()

Imaginons que nous ayons besoin de placer 5 fois un symbole sur la scène, afin d'obtenir 5 objets d'affichage. Utilisez la procédure ci-dessous.

1. Référez vous au point 8 (*Placer un symbole dynamiquement sur la scène*) pour apprendre à placer un symbole sur la scène.
2. Saisissez le script ci-dessous dans la fenêtre Actions.

```
var carte:Carte;
for (var numero:int=0; numero<=4; numero++) {
  carte = new Carte();
  carte.x = 30+(numero*60);
  carte.y=80;
  addChild(carte);
}
```

Rappelons que la variable **numero** va prendre successivement 5 valeurs différentes (de 0 à 4) et que ces dernières vont être utilisées dans l'exécution des lignes d'instructions contenues entre les accolades. Si vous devez affecter un comportement à chacun de ces objets d'affichage, vous devez le faire entre ces accolades (après la ligne d'instanciation).

B - La boucle for each()

Cette technique est comparable à une boucle for() "traditionnelle", mais vous l'utiliserez principalement pour parcourir un tableau, une arborescence XML, un objet, etc.

Dans ce premier exemple, nous parcourons l'ensemble des entrées du tableau pour les afficher dans la fenêtre Sortie de Flash.

```
var prenoms:Array = new Array("Marine","Luna","Marjorie","David");
for each (var individu:String in prenoms) {
  trace(individu);
}
```

Résultat de l'affichage

Marine
Luna
Marjorie
David

Dans ce deuxième exemple, nous cherchons à afficher les valeurs de l'ensemble des propriétés contenues dans l'objet **client**.

```
var client:Object = {prenom:"David",nom:"Tardiveau",dn:"1970",sexe:"Masculin"};
for each (var individu:String in client) {
  trace(individu);
}
```

Résultat de l'affichage

Masculin
1970
David
Tardiveau

Remarque : Lorsqu'un tableau contiendra des valeurs de types différents, vous utiliserez alors l'étoile (*) en lieu et place du type String du script ci-dessus, vous obtiendrez alors :

```
for each (var individu:* in prenoms) {
```

C - Les boucles imbriquées

Vous aurez parfois besoin d'imbriquer des boucles **for(),** c'est-à-dire exécuter une boucle au sein d'une autre. Ainsi dans l'exemple ci-dessous, nous cherchons à disposer sur la scène, 5 lignes de 6 objets d'affichage, représentant ainsi un tableau de 30 cellules.

Remarque : Commencez par définir un nom de Liaison/Classe à un symbole (Pour vous aider, référez-vous au point 8 (Placer un symbole dynamiquement sur la scène)) puis saisissez le script ci-dessous :

```
var cellule:Cellule;

for (var i:int=0; i<5; i++) {
  for (var j:int=0; j<6; j++) {
        cellule = new Cellule();
        cellule.x = 50+(j*30);
        cellule.y = 50+(i*30);
        addChild(cellule);
  }
```

```
}
```

La boucle contenant la variable locale **j** est répétée 5 fois. Cette dernière va prendre successivement les 5 valeurs suivantes : 0, 1, 2, 3 et 4.

D - La boucle while()

Afin de mieux comprendre le processus d'exécution d'une boucle **while()**, essayons de comparer ces deux scripts qui aboutissent au même résultat :

Avec un boucle **for()** :

```
var cellule:Cellule;

for (var j:int=0; j<6; j++) {
  cellule = new Cellule();
  cellule.x=50;
  cellule.y = 50+(j*30);
  addChild(cellule);
}
```

Avec une boucle **while()** :

```
var cellule:Cellule;
var j:int = 0

while (j<6) {

  cellule = new Cellule();
  cellule.x=50;
  cellule.y = 50+(j*30);
  addChild(cellule);
  j++
}
```

Comme vous pouvez le constater, le code contenu entre les accolades est identique à l'exception de la variable qui est locale dans le cas de la boucle **for()** alors qu'elle est globale dans le cas de la boucle **while()**.

Dans le cas d'une boucle **for()**, l'initialisation et l'incrémentation de la variable locale ainsi que la condition de répétition (de l'exécution des lignes d'instructions contenues entre les accolades) se trouvent entre les parenthèses. Dans le cas d'une boucle **while()**, ces trois paramètres sont séparés.

25. LES FONCTIONS

Une fonction sert à simplifier la rédaction de programmes dans la mesure où vous saisissez une boite fois pour toute des lignes d'instructions dans une fonction (la phase de déclaration de la fonction), puis, vous faites appel à la fonction en faisant référence à son nom.

A - Déclaration d'une fonction

Utilisez la structure de code ci-dessous :

- Le mot **function** (avec un *u* et non un *o*) suivi d'une paire de parenthèses
- Une paire d'accolades pour contenir le code rattaché à la fonction.

Avant de vous montrer le premier exemple, il est important que vous sachiez saisir l'intégralité de la structure du code à l'aide du raccourci clavier **Esc-f-n**. Dans le cas contraire, vous perdrez beaucoup de temps à écrire une fonction.

```
function afficher() {
  trace("Bonjour tout le monde");
}
```

Rappelons une fois encore qu'un nom donné (de variable, d'objet d'affichage, de tableau, d'instance, etc.) dans un programme (c'est valable pour le nom de notre fonction) ne doit pas contenir de caractères spéciaux, accentués ni même d'espace. Ne commencez pas non plus le nom de la fonction par une majuscule.

Remarque : *Le raccourci **Esc-f-n** ne se fait pas comme un raccourci de type **Commande-C** (Mac) ou **CTRL-C** (Windows). Vous devez effectuer une pression sur la touche **Esc** SANS LA MAINTENIR, puis vous appuyez successivement sur les touches **f** puis **n** de votre clavier.*

B - Appel d'une fonction

Lorsqu'une fonction est créée, vous pouvez alors l'appeler, c'est-à-dire y faire référence dans un programme pour demander l'exécution des lignes d'instructions contenues dans la fonction. Saisissez simplement le nom de la fonction accompagnée de ses parenthèses.

```
afficher();
```

Résultat de l'affichage

```
Bonjour tout le monde
```

C - Fonctions avec variables locales

Vous aurez parfois besoin de faire "appel à du code" en appelant une fonction tout en spécifiant une information particulière (en passant une valeur à la fonction).

Déclarez alors une fonction en spécifiant un paramètre, appelé aussi "variable locale", entre les parenthèses de la fonction.

```
function afficher(nomPersonne:String) {
  trace("Bonjour "+nomPersonne+" !");
}
```

Vous devez ensuite appeler la fonction en spécifiant une information entre les parenthèses.

```
afficher("David");
```

Résultat de l'affichage

Bonjour David !

FONCTIONNALITES COMPLEMENTAIRES

26. AFFICHER UNE ANIMATION EN PLEIN ECRAN

1. Placez deux objets d'affichage sur la scène et nommez les (ex. : **btPleinEcran** et **btEnModeFenetre**).

```
btPleinEcran.addEventListener(MouseEvent.MOUSE_DOWN,basculerPleinEcran);
btEnModeFenetre.addEventListener(MouseEvent.MOUSE_DOWN,ModeFenetre);
function basculerPleinEcran(evt:MouseEvent) {
   stage.displayState=StageDisplayState.FULL_SCREEN;
   stage.scaleMode=StageScaleMode.NO_SCALE;
}
function ModeFenetre(evt:MouseEvent) {
   stage.displayState=StageDisplayState.NORMAL;
}
```

Remarque : *Dans le cas d'une animation diffusée on-line, vous ne pouvez pas exécuter les deux lignes d'instructions contenues dans la fonction* ***basculerPleinEcran*** *en dehors d'un écouteur qui fait appel à un événement de type* ***MouseEvent****.*

27. EXECUTER UNE FONCTION JAVASCRIPT A PARTIR D'UNE ANIMATION FLASH

Pour pouvoir déclencher une fonction contenue dans une page HTML depuis une animation Flash, vous devez suivre la procédure suivante :

1. Déclarer une fonction javascript dans votre page HTML (celle qui contient les balises d'encapsulation du swf) :

```
<script language="JavaScript" type="text/javascript">
function ouvrirFenetreAccueil ( variableMessage ) {
alert (variableMessage);
}
</script>
```

2. Utilisez la ligne d'instruction ci-dessous dans votre animation Flash :

```
ExternalInterface.call ("ouvrirFenetreAccueil", "Bonjour David");
```

Dans notre exemple, la fonction **call()**contient deux paramètres : Le premier correspond au nom de la fonction appelée et le deuxième est une variable locale transmise à la fonction depuis Flash. Vous pouvez exécuter une fonction sans pour autant passer de variables, vous n'utiliserez alors qu'un seul paramètre dans la méthode **call()** et ne saisirez pas de nom de variable locale dans la fonction javascript.

28. NAVIGUER VERS UNE PAGE HTML DEPUIS UNE ANIMATION FLASH

La technique est extrêmement simple et logique si vous savez déjà chargé une image, des données ou le contenu d'un fichier XML dans une animation Flash. Vous allez en effet devoir faire appel à la classe **URLRequest()**, puis vous utiliserez la méthode **navigateToURL()**.

```
var adressePage:URLRequest = new URLRequest("http://www.google.fr");
btLien.addEventListener(MouseEvent.MOUSE_DOWN,allerAuSite);
function allerAuSite(evt:MouseEvent) {
  navigateToURL(adressePage);
}
```

Précisons que vous pouvez ajouter un paramètre à la méthode **navigateToURL()**, en spécifiant la fenêtre cible. Séparez simplement l'adresse et le nom de la fenêtre visée, par une virgule.

```
navigateToURL(adressePage, "_blank");
```

*Remarque : Vous pouvez utiliser les cibles habituelles **_self**, **_blank**, **_parent** et **_top**. Par ailleurs, privilégiez la méthode **call()** de la classe **ExternalInterface()** (voir le point 26 précédent) pour appeler des fonctions Javascript.*

29. TEMPORISER L'EXECUTION D'UNE LIGNE D'INSTRUCTION AVEC LA CLASSE TIMER()

A - Lancer l'exécution

L'instanciation de la classe **Timer()** attend deux paramètres : La durée exprimée en seconde et le nombre d'itérations (0 pour obtenir une répétition infinie).

```
var compteRebours:Timer = new Timer(2000,5);
compteRebours.addEventListener(TimerEvent.TIMER,initialisation);
compteRebours.start();
function initialisation(evt:TimerEvent) {
  trace("Et de une seconde de plus…");
}
```

***Remarque :** Dans l'exemple ci-dessus, la fenêtre Sortie va afficher 5 fois, toutes les 2 secondes, le message "Et une seconde de plus…"*

B - Détecter la fin de la dernière répétition

```
var compteRebours:Timer = new Timer(2000,5);
compteRebours.addEventListener(TimerEvent.TIMER,initialisation);
compteRebours.addEventListener(TimerEvent.TIMER_COMPLETE,finIterations);
compteRebours.start();
```

```
function initialisation(evt:TimerEvent) {
  trace("Et de une seconde de plus...");
}
function finIterations(evt:TimerEvent) {
  trace("Fin !");
}
```

Remarque : *Vous pouvez aussi utiliser l'exemple ci-dessous pour exécuter une ligne d'instruction une seule fois au bout de 10 minutes.*

```
var compteRebours:Timer = new Timer(600000,5);
```

30. STOCKER DES INFORMATIONS SUR LA MACHINE DE L'UTILISATEUR

1. Faites appel à la classe **SharedObject()** que vous n'instanciassiez pas pour autant avec le mot clé **new**.

2. Testez éventuellement la valeur d'une variable contenue dans une instance de la classe **SharedObject()** spécifiée pour vérifier l'existence de cette dernière.

3. Faites appel à la propriété data pour lire et stocker des informations sur la machine de l'utilisateur.

Remarque *: La procédure pour créer et charger/lire un cookie est la même. Il s'agit de la première ligne d'instruction du script ci-dessous :*

```
var coordonnees:SharedObject = SharedObject.getLocal("positionsObjetAffichage");
if (coordonnees.data.motPasse!=undefined) {
  zoneDeTest.text=coordonnees.data.motPasse;
}
btMemoriser.addEventListener(MouseEvent.MOUSE_DOWN,memoriserInfos);
btLire.addEventListener(MouseEvent.MOUSE_DOWN,lireInfos);
function memoriserInfos(evt:MouseEvent) {
  coordonnees.data.motPasse=zoneDeTest.text;
  coordonnees.flush();
  zoneDeTest.text="";
}
function lireInfos(evt:MouseEvent) {
  zoneDeTest.text=coordonnees.data.motPasse;
}
```

31. IMPRIMER UNE ANIMATION FLASH

Vous ne pouvez imprimer que des objets d'affichage de type **Sprite** (en précisant son nom) ou la scène (en précisant le paramètre **this**).

A - Imprimer la scène

Faites appel à la classe **PrintJob()** puis les méthodes **start()**, **addPage()** et **send()**.

```
var gestionImpression:PrintJob = new PrintJob();
gestionImpression.start();
gestionImpression.addPage(this);
gestionImpression.send();
```

Remarque : *Vous pouvez ajouter plusieurs impressions en ajoutant une ou plusieurs lignes d'instructions contenant la méthode* ***addPage()****.*

B - Imprimer une partie de la scène

Délimitez la zone à imprimer en créant une instance de la classe **Rectangle()**,puis en la passant comme paramètre de la méthode **addPage()**.

```
var zoneImpression:Rectangle = new Rectangle(30,30,330,130)
var gestionImpression:PrintJob = new PrintJob();
gestionImpression.start();
gestionImpression.addPage(this,zoneImpression);
gestionImpression.send();
```

32. OBTENIR DES INFORMATIONS RELATIVES A LA MACHINE DE L'UTILISATEUR

Pour connaître le système d'exploitation de l'ordinateur qui lit une animation Flash, connaître, sa langue et la version du lecteur Flash, faites appel à la classe **Capabilities**, suivie des propriétés suivantes :

- **version**
- **language**
- **os**

Exemple :

```
var versionPlayer = Capabilities.version;
```

Il ne vous reste plus qu'à utiliser la méthode **substr()** de la classe **TextField()** pour parcourir la chaine de caractères obtenue.

```
var versionPlayer = Capabilities.version.substr(4,1);
```

33. LISTE DES INSERTIONS AUTOMATIQUES DES RACCOURCIS DE YAZO.NET

Pour saisir très rapidement les "mots" en AS3 les plus utilisés dans un programme, vous avez la possibilité d'utiliser le raccourci Esc (ou Echapp) puis une combinaison de 2 lettres. Yazo.net a créé une extension MXP qui étend les raccourcis standards en vous proposant même d'insérer directement une ou plusieurs lignes d'instructions et non plus uniquement une méthode ou une propriété.

Note : Vous ne devez pas maintenir enfoncée la touche Esc lorsque vous appuyez sur les touches "a" puis "i" (ni a et i en même temps non plus).

Après avoir téléchargé et installé le fichier MXP disponible au téléchargement via le lien « Raccourcis Yazo » (en haut à gauche de chaque page de yazo.net) et après avoir relancé Flash, effectuez le raccourci Esc (ou Echapp) puis "a" puis "i". Vous découvrirez la liste des raccourcis (présentés ci-dessous) que vous pourrez utiliser pour saisir plus rapidement du code ActionScript.

http://www.yazo.net/racine/download/RaccourcisYazo.mxp

Essayez par exemple les combinaisons suivantes pour…

Charger une image : Esc-a-d et Esc-l-i
Charger du XML : Esc-a-d et Esc-l-x et Esc-v-x et Esc-cx
Réaliser une interpolation : Esc-t-w
Ecouter un son : Esc-a-d et Esc-l-s

Remarques : Pour placer une image sur la scène, vous n'oublierez pas d'ajouter addChild(cadre). Lorsque vous avez déclaré et initialisé une instance de type URLRequest, il n'est pas nécessaire d'en redéclarer une autre. Il vous suffit de redéfinir la propriété url de votre première déclaration comme ceci : adresse.url = cheminVersUnAutreFichier.extension (ex. adresse.url = "image23.jpg"). Les raccourcis commençant par la lettre l signifient load (pour charger).

Raccourcis d'origine

```
fn : function () {}
fr : for () {}
if : if () {}
tr : trace()
```

Raccourcis de Yazo.net

```
md : Écouteur MouseEvent.MOUSE_DOWN
en : Écouteur Event.ENTER_FRAME
ad : Instancier la classe URLRequest() pour définir un chemin vers un fichier
li : Charger une image
lx : Charger fichier XML
```

cx : Créer une instance XML
ls : Charger un fichier son
vi : Vérifier le chargement d'une image
vx : Vérifier le chargement d'un fichier XML
tw : TweenMax()
pe : Plein écran (FULL_SCREEN et NO_SCALE)
ap : Aller à une page web
pr : Script de préchargement d'une animation
ti : Script d'un Timer (1000 vaut 1 seconde)

Raccourcis de commandes rétablis par Yazo.net

dr : startDrag()
sd : stopDrag()
st : stop()
pl : play()
gs : gotoAndStop()
gp : gotoAndPlay()
pf : prevFrame()
nf : nextFrame()

Index