

# Séquence 2 : Les Variables

## I. Introduction

- Une variable est comme une boîte (une case mémoire) qui contient une valeur
- `x = 7;`
- La ligne ci-dessus stocke la valeur 7 dans la variable nommée x
- Attention une variable doit toujours être déclarée et typée
- A chaque apparitions de x dans le code, le programme récupère la valeur de la boîte
- Lorsque l'on utilise = de cette manière, cela est appelé **"assignation de variable"**

Code : <i>//déclaration de la variable</i> <code>private var maVariable : Number ;</code>  <i>//On assigne la valeur 7.2 à maVariable</i> <code>maVariable = 7.2 ;</code>	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <b>7.2</b> </div> maVariable
--	---

Donc pour stocker une variable dans la mémoire d'un ordinateur, il faudra toujours raisonner de la même manière :

- La déclarer en lui donnant un nom (lettres et chiffres). C'est avec ce nom que l'emplacement mémoire (et surtout son contenu) est identifié par le programme.
- La typer : Il existe plusieurs types de variables. Le type correspond à la nature de la donnée qui sera enregistrée dans la variable. Pour simplifier, nous allons nous intéresser à celles que nous sommes les plus susceptibles de rencontrer :
  - Texte : String (chaînes de caractères)
  - Numérique :
    - Entier : int (integer)
    - Réel : Number
  - Booléen (valeurs vrai ou faux) : boolean (true ou false)

ActionScript 3 intègre désormais trois types afin de représenter les nombres :

- int : représente un nombre entier 32 bit (32 bit signed integer)
- uint : représente un nombre entier non signé 32 bit. (32 bit unsigned integer)
- Number : représente un nombre décimal 64 bit (64-bit IEEE 754 double-precision floating-point number)

Notons que les deux nouveaux types int et uint ne prennent pas de majuscule, contrairement au type Number déjà présent au sein d'ActionScript 2.

Une variable de type int peut contenir un nombre oscillant entre -2147483648 et 2147483648 :

Syntaxe pour la création d'une variable :

Action Script

```
var nomDeLaVariable:classeDeLaVariable;  
var nomDeLaVariable:classeDeLaVariable = valeur; //On peut directement définir la  
valeur avec un =.
```

Le nom d'une variable peut s'écrire avec des caractères alphabétiques, accents et majuscules compris, des chiffres, l'underscore (\_) ainsi que  $\mu$  et \$. Mais il ne faut pas que le premier caractère soit un chiffre, et les majuscules ont leurs importances (largeur  $\neq$  Largeur).

Il existe différentes classes (ou types) de variables :

Number : peut prendre comme valeur tout nombre réel;

int : pour les nombres entiers;

uint : pour les nombres entiers positifs uniquement (unsigned int);

Boolean : variable qui prend la valeur soit true soit false (vraie ou faux);

String : chaîne de caractères, se définit avec des guillemets " autour du texte.

Comme dit plus haut, on peut définir une variable directement avec un = après sa déclaration. On peut aussi changer cette valeur à tout moment de l'exécution du programme ainsi :

nomDeLaVariable = uneValeur;

Exemple :

Action Script

```
var largeur:uint = stage.stageWidth;
var hauteur:uint = stage.stageHeight;
```

"uneValeur" peut être une autre variable par exemple, ou tout autre chose (comme dans l'exemple précédent) contenant une même classe de valeur (numérique, booléen, ou chaîne de caractères).

Le signe égal utilisé n'a pas la même signification que celui des mathématiques. Il peut se traduire par « prend la valeur de ».

**Opérations**

Les opérations classiques :

+ : addition,

- : soustraction,

\* : multiplication,

/ : division,

% : modulo.

*Le modulo est le reste de la division entière. Par exemple  $17 \% 3 = 2$ .*

Pour les variables de types String, seul le + peut être utilisé. Les chaînes de caractères sont alors collées à la suite. On peut afficher la valeur d'une variable dans la fenêtre Sortie avec la fonction *trace* comme dans l'exemple qui suit (apparition après l'exécution faite avec Ctrl+Entrer ou Pomme+Entrer).

Exemple :

### Action Script

```
var a:Number = 12;
var b:Number = 6;
var largeur:Number = stage.stageWidth;
var texteLargeur:String = "La largeur de la scène est de : ";
texteLargeur = textelargeur + largeur;
trace(a + " multiplié par " + b + " font " +(a*b)); // Il faut inclure des espaces
dans les guillemets.
trace(texteLargeur);
```

Par contre si on entre :

### Action Script

```
var largeur:Number = stage.stageWidth;
var texteLargeur:String;
texteLargeur = largeur;
trace(texteLargeur);
```

Alors une erreur apparaîtra : *"1067: Contrainte implicite d'une valeur du type Number vers un type sans rapport String."*

Il faut alors remplacer largeur par String(largeur). Ainsi, le nombre contenu dans largeur est transformé en chaîne de caractères et remplace la valeur de texteLargeur par des chiffres.

En plus des opérateurs courants précédemment vus, on peut utiliser d'autres d'opérations mathématiques :

Math.round(a) : Pour arrondir le nombre à l'entier le plus proche.

Math.random() : Pour tirer un nombre aléatoire entre 0 et 1.

Math.cos(a) : Pour calculer le cosinus de *a*, en degrés. On peut remplacer *cos* par *sin*, *exp*, *log*, *tan*.

Math.acos(a) : Pour calculer le cosinus de *a*, en radians. On peut remplacer *acos* par *asin*, *atan*.

Math.pow(a,b) : Pour calculer *a* à la puissance *b*.

### Incrémentasson

On a vu précédemment que le signe = signifiait « prend la valeur de », il est donc possible d'écrire : *"a = a + 1;"*

Ce qui ajoute 1 à la valeur de *a* pour *a* numérique. Mais l'ActionScript propose une autre façon d'écrire l'incrémentasson donnée en exemple *"a++;"*

De même, quelque soit *n* :

### Action Script

```
a = a - 1; //équivalent à :
a--;
a = a + n; //équivalent à :
a += n;
```

isn.s

```
a = a - n; //équivalent à :  
a -= n;  
a = a * n; //équivalent à :  
a *= n;  
a = a / n; //équivalent à :  
a /= n;
```

### b) Nommer des variables

Il est très souvent nécessaire d'être explicite quant au choix de nom des variables, car le travail de programmation est un travail collaboratif, que d'autres programmeurs peuvent être amenés à utiliser ou corriger. Par exemple, monTableau sera un bon nom pour un tableau (à condition qu'il n'y en ait qu'un !!)

Il est toujours bon de respecter des conventions qui permettent de nommer des variables. Par exemple, ajoutez systématiquement la lettre g devant le nom d'une variable globale, ou débutez systématiquement chaque mot de la variable par une majuscule sauf la première lettre.

- ☐ gMaVariableGlobale
- ☐ maVariableDePropriete
- ☐ var maVariableLocale

Les règles pour nommer une variable sont :

- ☐ un nom de variable ne doit pas contenir de caractère ESPACE, de tiret (-) ni de caractère spécial. Par contre, le caractère de soulignement(\_) peut être utilisé
- ☐ un chiffre seul ne peut être un nom de variable
- ☐ un nom de variable ne peut être this, var for et in qui sont des mots clefs réservés d'ActionScript.

## 2. Types de variables

Une variable peut avoir plusieurs types de contenus (nombre entier, nombre relatif, chaîne de caractère, booléen). Jusqu'à ActionScript 2, celui-ci n'a pas à être défini. C'est la valeur qu'on lui assigne qui détermine son type. Par exemple :

- ☐ en écrivant var maVariable=5;, on associe à maVariable un type d'entier, soit int
- ☐ en écrivant var monTableau=["une première valeur", "une seconde valeur"];, on associe à monTableau un type de tableau
- ☐ etc.

Mais, depuis actionScript 3, les variables doivent avoir leur types définis dès leur création (qu'on appelle *déclaration*).

Le mot clef var devient obligatoire. Ainsi, on écrira plutôt :

- ☐ var maVariable:Number=5;
- ☐ var monTableau:Array=new Array();
- ☐ etc.

Remarque : Par la suite, il n'est plus obligé de continuer à utiliser cette syntaxe une fois la première déclaration de variable effectuée. Par exemple, suite aux lignes précédentes, on pourra écrire : var monTableau=["une première valeur", "une seconde valeur"];.

### a) Liste des types de variables

Cette partie n'est surtout valable que pour AS3. Mais elle est déjà opérationnelle pour AS2.

Il existe quatre types de variables "primaires" :

- ☐ :Number - nombre réel, comprenant ainsi les nombres d'ensembles différents comme les entiers, les relatifs, etc.
- ☐ :String - indique une chaîne de caractères
- ☐ :int - restreint aux nombres entiers
- ☐ :Boolean - représente un nombre qui ne peut prendre que deux valeurs, 0 ou 1, qu'on peut également écrire false ou true.

En fait, il existe un grand nombre de types différents :

- toute classe est un type référencé :
  - `var monTableau:Array = new Array();`
  - `var monXML:XML = new XML();`
  - `var monMouvicipe:MovieClip=new MovieClip();`
  - `var loader:movieClipLoader=new movieClipLoader();`
  - il existe des types particuliers :
    - les fonctions sont de type void. Celà s'écrit `function maFunction(argument):void {`
- Il existe des types dits multiples (mais qui utilisent un grand nombre de ressources : \*. Par exemple : `var valeuretTableau:*`;

### b) Tester le type des variables

Il est possible de tester le type d'une variable ou son contenu de plusieurs façons, par exemple en vérifiant qu'elle possède bien un contenu, de quel type est son contenu, etc. Vous pourrez par exemple utiliser `isNaN()` (n'est pas un nombre), `typeof()` (renvoie le type de la variable) pour cela.

### 3) Portée des variables

#### a) Variables sous AS2

Les variables locales sont utilisées lorsque le script a besoin d'elles, mais disparaissent immédiatement après, lorsque toutes les instructions du script ont été effectuées.

Le mot clef pour déclarer une variable locale est `var`. Par exemple :

```
var maChainedeCaracteres="Je suis là";<br/>
maChainedeCaracteres+=" Toi aussi ?"
```

donnera à la variable `maChainedeCaracteres` la valeur : "Je suis là : Toi aussi ?" mais tout disparaîtra à la fin de l'exécution du script.

Les variables de scénario sont attachées à l'occurrence. Tant que celle-ci est présente, la variable de scénario aura cette valeur. On peut également la déclarer par le mot clef `var`.

On peut ainsi la modifier à distance. Par exemple, un script attaché à une occurrence appelée `monOccurrence` déclare une variable nommée `maVariable`. Depuis un script d'image je peux modifier ou accéder à cette variable grâce à l'instruction `monOccurrence.maVariable`.

Les variables globales permettent d'avoir accès à des valeurs stockées dans tout le document flash. Pour ce faire, il s'agit qu'elle soit déclarée au moins une fois. Par exemple l'instruction `_global.maVariable=""`; assigne à une variable globale une chaîne de caractères vide.

Par la suite, vous pourrez avoir accès directement à cette valeur en écrivant `maVariable`.

#### b) Variables sous AS3

Un seul mot clef pour toutes les variables : `var`. Mais les variables attachées à des fonctions ne sont plus accessibles. Ainsi, il est fortement conseillé, lorsque des variables doivent être accessibles de "partout", de les déclarer et les initialiser dans le script initial.

Enventuellement,	on	pourra	écrire :
<input type="checkbox"/> <code>var</code>			<code>maVariable :int=5 ;</code>
<input type="checkbox"/> <code>var</code>			<code>monClip1.maVariable :int=5 ;</code>
<input type="checkbox"/> <code>var monClip2.maVariable :String="une chaine de caractères" ;</code>			

Ces trois variables sont différentes, car elles sont attachées à trois endroits différents de l'animation, et existent donc toutes trois.

La notion des mots clefs `public`, `private` et `static` sera vue dans le chapitre sur les classes AS3 (bientôt publié).

**II.Conclusion**

---

---

---

---