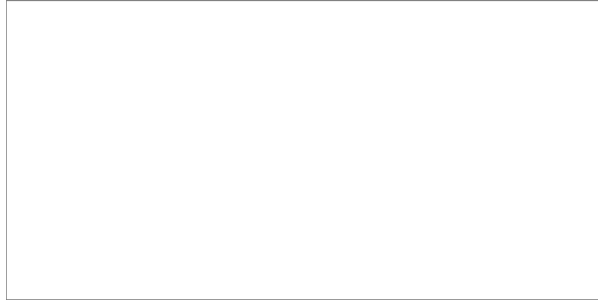


Séquence 5 : Les boucles

for do while

I. Généralités

Les boucles sont des techniques qui permettent de répéter les mêmes instructions plusieurs fois. Tout comme pour les conditions, il y a plusieurs façons de réaliser des boucles. Dans tous les cas, le schéma est toujours le même :



On répète n fois les mêmes instructions.

II. La boucle *while*

II.1. Structure

La boucle *while* s'écrit de la manière suivante :

```
while (condition)
{
    instruction 1;
    instruction 2;
    instruction 3;
    ...;
    instruction n;
}
```

Le terme *while* se traduit par tant que. La structure répétitive s'exécute selon les principes suivants :

- Tant que l'expression à l'intérieur des parenthèses reste vraie, la ou les instructions composant la boucle sont exécutées.
- Le programme sort de la boucle dès que l'expression à l'intérieur des parenthèses devient fausse.

II.2. Application

Vous allez réaliser une application qui permet de créer et positionner aléatoirement 10 billes sur la scène. Pour cela, vous allez créer un nouveau projet AIR AS3 Projector appelé lesBoucles. Ensuite recopier le code ci-dessous pour placer les 10 billes sur la scène.

```
public class Main extends Sprite
{
    //création de l'objet Sprite
    private var bille : Sprite ;
    /// variables
    private var nombreDeBille : int = 10 ;
    private var largeurEcran : int ;
    private var hauteurEcran : int ;
```

```

public function Main():void
{
    // entry point
    var i : int = 0 ;

    // récupérer la largeur et la hauteur de la scène
    largeurEcran = stage.stageWidth ;
    hauteurEcran = stage.stageHeight ;

    //dessin des billes
    while (i < nombreDeBille) //tant que i < nombreDeBille traite les
    {
        trace(i); //instructions entre les accolades
        i = i + 1 ;
        //créer un nouvelle objet bille
        bille = new Sprite();
        bille.graphics.beginFill(0xFF0000);
        bille.graphics.drawCircle(0, 0, 15);
        addChild(bille);
        //tester

        //place au hasard la bille sur la scène
        bille.x = largeurEcran * Math.random();
        bille.y = hauteurEcran * Math.random();
        //tester

        //associe à chacune des billes un évènement Click
        bille.addEventListener(MouseEvent.CLICK, billeCliquer);
    }
}

private function billeCliquer(e:MouseEvent):void
{
    //la bille cliquer n'est plus visible sur la scène
    e.target.visible = false;
    //tester
}
}

```

III. La boucle **do...while**

Le langage AS3 propose une autre structure répétitive, analogue à la boucle while, mais dont les instructions sont exécutées avant même d'avoir testé la condition d'exécution de la boucle. Il s'agit de la boucle do...while.

III.1. Structure

La boucle do...while se traduit par les termes faire...tant que. Cette structure s'écrit de la façon suivante.

```

do {
    instruction 1;
    instruction 2;
    instruction 3;
    ...;
    instruction n;
}

```

```
} while (condition);
```

La boucle *do...while* s'exécute selon les principes suivants :

- Les instructions situées à l'intérieur des accolades (boucle) sont exécutées tant que l'expression conditionnelle placée entre parenthèses () est vraie.
- Les instructions sont exécutées au moins une fois, puisque l'expression conditionnelle est examinée en fin de boucle, après exécution des instructions.

Attention, si la condition mentionnée entre parenthèses reste toujours vraie, les instructions de la boucle sont répétées à l'infini. On dit que le programme « boucle » et il plante !

Un point-virgule est placé à la fin de l'instruction *while* (expression);. N'oubliez pas d'en mettre un après le *while*, ou sinon votre programme plantera à la compilation.

III.2.Application

Vous allez ajouter aléatoirement sur la scène 8 carrés de couleur bleue (0x0000FF) en utilisant la boucle *do while*.

IV.La boucle *for*

IV.1.Structure

L'instruction *for* permet d'écrire des boucles dont on connaît à l'avance le nombre d'itérations (de boucles) à exécuter. Elle est équivalente à l'instruction *while* mais est plus simple à écrire.

```
for (initialisation; condition; incrémentation)
```

```
{
  instruction 1;
  instruction 2;
  instruction 3;
  ...;
  instruction n;
}
```

Les termes *Initialisation*, *Condition* et *Incrémentation* sont des instructions séparées obligatoirement par des points-virgules (;). Ces instructions définissent une variable, ou indice, qui contrôle le bon déroulement de la boucle. Ainsi :

- *Initialisation* permet d'initialiser la variable de la boucle (exemple : $i = 0$). Elle est la première instruction à être exécutée dans la boucle.
- *Condition* définit la condition à vérifier pour continuer à exécuter la boucle (exemple : $i < 10$). Elle est examinée avant chaque tour de boucle, y compris au premier.
- *Incrémentation* est l'instruction qui permet de modifier le résultat du test précédent en augmentant ou diminuant la valeur de la variable testée. L'incrément peut être augmenté ou diminué de N. N est appelé le « pas d'incrément » (exemple : $i = i + 2$). Cette instruction est exécutée à la fin de chaque tour de boucle.

Les boucles *for* réalisent un nombre précis de boucles dépendant de la valeur initiale, de la valeur finale et du pas d'incrément. Compléter les différents exemples pour voir comment ces boucles sont exécutées :

Int i	Valeur initiale	Valeur finale	Pas d'incrément	Nombre de boucles	Valeurs prises par i
for (i = 0; i < 5; i = i + 1)	0	4	1	5	0 1 2 3 4
for (i = 4; i <= 12; i = i + 2)					
for (i = 5; i > 0; i = i - 1)					

IV.2.Application

Vous allez ajouter aléatoirement sur la scène 12 ellipses de couleur verte (0x00FF00) en utilisant la boucle *for*.

V.Conclusion

Les trois boucles décrites ci-dessous ont toutes pour résultat l'affichage de la fenêtre de sortie suivante :

- La boucle ***while***

```
var i:int = 5;
while (i < 10) {
    trace("Boucle n° "+ i );
    i++;
}
```

Dans la boucle *while*, le test de continuation de boucle s'effectue à l'entrée de la boucle. La variable *i* (compteur) doit donc être déclarée et initialisée correctement pour être sûr que les instructions placées à l'intérieur de la boucle soient exécutées au moins une fois.

L'instruction *i++*, placée à l'intérieur de la boucle, garantit que le test (*i < 10*) ne reste pas toujours vrai et empêche le programme de boucler à l'infini.

- La boucle ***do...while***

```
var i:int = 5;
do {
    trace("Boucle n° "+ i );
    i++;
} while (i < 10);
```

Dans la boucle *do...while*, le test de continuation de boucle s'effectue en sortie de boucle. Ainsi, quelle que soit la condition, nous sommes assurés que la boucle est exécutée au moins une fois.

- La boucle ***for***

```
for (var i : int = 5; i < 10; i++) {
    trace("Boucle n° "+ i );
}
```

Dans la boucle *for*, la déclaration, l'initialisation, le test de continuation de boucle ainsi que l'incrémement du compteur sont placés à l'intérieur d'une seule et même expression. La boucle *for* est utilisée quand on connaît à l'avance le nombre d'itérations (boucles) à exécuter.

