Initiation à la programmation COFFEE. (partie 3.)

Par Tengaal (tengaal@libertysurf.fr) 19/03/2002.

Présentation

Entrons maintenant dans le vif du sujet, en apprenant à utiliser les objets de Cinema 4D par la programmation. Pour commencer, nous allons voir différentes manières de trouver un objet dans le projet 3D et comment accéder à ses propriétés. Ensuite, je vous propose la liste des objets de Cinema 4D.

Trouver et accéder à un objet

Il y a plusieurs manières d'accéder à un objet, on peut trouver un objet à partir de sa position dans une hiérarchie ou bien à partir de son nom dans le document.

Pour commencer, expliquons comment accéder aux propriétés et fonctions d'un objet. On a vu dans la partie 1 qu'on peut attribuer un objet à une variable, et c'est donc cette variable qui va nous servir de pointeur sur cet objet, pour accéder aux propriétés de cet objet il suffit de faire suivre la variable des caractères '->', cette « flèche » est ensuite suivie de la fonction de l'objet.

Exemple :

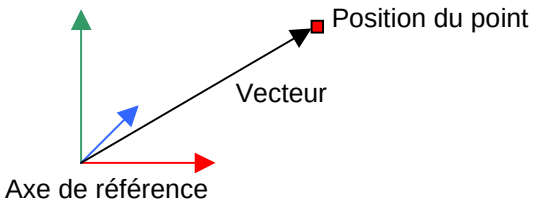
```
var position ;
var px,py,pz ;

main(doc,op)                    // op est l'objet associé à cette expression COFFEE
{
   position=op->GetPosition() ; // la position de l'axe de l'objet op est stocké dans position.
   px=position.x ;              // on prend la composante x du vecteur position.
   py=position.y ;              // puis la composante y.
   pz=position.z ;              // et enfin la composante z.
}
```

Explication : On créé une variable position puis trois variables px,py et pz, toutes indéfinies.
On a vu dans la partie 1 que la variable op représentait l'objet sur lequel se trouvait cette expression COFFEE, donc nous allons utiliser op et lire sa position.
La fonction GetPosition() est utilisable avec tous les objets de Cinema 4D, elle renvoie un vecteur de 3 composantes donnant la position d'un point dans l'espace, dans le cas d'un objet ce point est l'axe.
La variable position devient donc un objet vector et reçoit la position de op.
Les variables px,py et pz sont ensuite définies comme variable numérique et reçoivent chacune la composante du vecteur position pour les axes x,y et z.

Info : le caractère point '.' permet d'accéder aux composantes d'un objet de type vector, les trois valeurs contenues dans un vecteur ont pour nom x,y et z donc pour lire ces trois valeurs, on procède ainsi :

```
var MonVecteur=new (vector);  //est une variable de type vector.
MonVecteur.x=50;              //donne accès à la composante X du vecteur
MonVecteur.y=100;             //donne accès à la composante Y du vecteur
MonVecteur.z=0;               //donne accès à la composante Z du vecteur
```

Ce vecteur positionne un donc point en position (50,100,0)



Pointer sur un objet précis

La position dans une hiérarchie
Il existe plusieurs fonctions pour se déplacer dans une hiérarchie :

Syntaxe COFFEE	Hiérarchie dans Cinema 4D
ObjetPrécédent=Objet->GetPrev() ;	ObjetParent
ObjetSuivant=Objet->GetNext() ;	ObjetPrécédent
ObjetParent=Objet->GetUp() ;	Objet
ObjetEnfant=Objet->GetDown() ;	ObjetEnfant
	ObjetSuivant

L'accès à un objets peut aboutir à un échec (erreur dans le programme) dans le cas où l'objet demandé n'existe pas, on peut tester la réussite ou non du pointage sur un objet avec l'instruction if qui va tester si l'objet pointé n'existe pas (opérateur NOT, partie 2, !(condition)).

L'exemple suivant va essayer d'obtenir l'objet enfant de op :

```
var ObjetEnfant ;

main(doc,op)
{
   ObjetEnfant=op->GetDown() ;          // on pointe sur le sous-objet de op
   if ( !ObjetEnfant) return ;          // si ObjetEnfant n'existe pas on quitte le programme

   // suite du programme
}
```

Le pointage dans le document par le nom
Une fonction permet de trouver un objet dans le document à partir de son nom :
ObjetCible=Document->FindObject("Cible") ;

Exemple :

```
const var NomCible= "Je suis ici" ;          // constante texte (type string)
var ObjetCible ;                             // variable non définie

main(doc,op)                                 // doc représente le document (ou projet en cours)
{
   ObjetCible=doc->FindObject(NomCible) ;    // on cherche l'objet qui s'appelle "Je suis ici"
   if ( !ObjetCible) return ;                //si ObjetCible n'est pas trouvé on sort du programme

   // suite du programme
}
```

Info : Ce type de pointage est pratique pour trouver un objet dans tout le document quelque soit sa position dans une hiérarchie, par contre il faut que l'objet recherché porte un nom unique.

A savoir :

Le langage COFFEE fait la différence entre les majuscules et les minuscules, par conséquent il faut veiller à bien respecter l'orthographe des noms de variables et des fonctions.
Pour vérifier si votre programme contient des erreurs, fait « Exécuter » dans la fenêtre du script COFFEE et en bas de celle-ci, vous aurez l'information « Pas d'erreurs » si tout va bien sinon le type de l'erreur et sa position dans le programme seront mentionnés et le curseur sera placé à l'endroit où se trouve l'erreur dans le code.
Une expression COFFEE qui contient une erreur, « bloque » la manipulation des objets 3D dans la scène et envoie les messages d'erreur dans la fenêtre « console » de Cinema 4D.

Liste des objets de Cinema 4D

Exemple d'utilisation : var MonObjet=new(LightObject) ; //Créé un objet Lumière

Les objets :

ArrayObject
AttractorObject
BackgroundObject
BaseObject
BendObject
BezierObject
BoneObject
BooleObject
BulgeObject
CameraObject
ConPlaneObject
DeflectorObject
DestructorObject
EnvironmentObject
ExplosionObject
ExtrudeObject
FFDObject
FloorObject
ForegroundObject
FormulaObject
FrictionObject
GravitationObject
HeadphoneObject
HyperNURBSObject
InstanceObject
LatheObject
LightObject
LoftObject
LoudspeakerObject
MeltObject
MetaballObject
MicrophoneObject
NullObject
ParticleObject
PointObject
PolygonObject
PrimitiveObject
RotationObject
ShatterObject
ShearObject
SkyObject
SplineObject
SplinePrimitiveObject
StageObject
SweepObject
SymmetryObject
TaperObject
TurbulenceObject
TwistObject
WindDeformObject
WindObject
WrapObject

Les propriétés (Tags)

AnchorTag
BackupTags
BakeParticleTag
BaseTag
CoffeeExpressionTag
CompositingTag
DisplayTag
ExpressionPluginTag
FixExpressionTag
HermiteTag
IKExpressionTag
KinematicTag
MetaballTag
MotionblurTag
ParticleTag
PhongTag
PluginTag
PointSelectionTag
PointTag
PolygonSelectionTag
PolygonTag
ProtectionTag
RestrictionTag
StageObject
StickTextureTag
SunExpressionTag
TargetExpressionTag
TextureTag
UVWTag
VariableTag
VertexmapTag
WWWTag

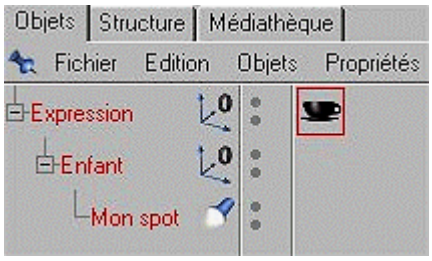
Bien sûr à cette liste il faut ajouter les objets qui concernent l'animation (Tracks, Sequences et Keys)
Nous les étudierons plus tard…

Création et insertion d'un objet dans le document

Nous allons terminer cette troisième partie avec un exemple de création et d'organisation d'objets dans un projet Cinema 4D.

Pour créer un objet et l'insérer dans notre projet, il faut d'abord définir la variable qui représente cet objet, puis une fois cette objet créé, il faut lui donner un nom et enfin l'insérer dans le document.
Une fois cette étape franchie, on peut placer les objets les uns par rapport aux autres en créant des hiérarchies.

Dans l'exemple suivant, nous allons créer un objet neutre (enfant de l'objet contenant l'expression) puis un objet Lumière (enfant de ce dernier objet neutre), on aura donc la hiérarchie suivante :



Le programme :

```
main(doc,op)
{
   var Enfant=op->GetDown() ;      //on créé la variable Enfant et on l'attribue au sous-objet de op.

   if (!Enfant)                    //si Enfant n'existe pas c'est que op n'a pas de sous-objets
   {                               //donc nous allons les créer.
         Enfant=new(NullObject) ;              //on définit Enfant comme objet neutre.
         Enfant->SetName("Enfant") ;           //on l'appelle « Enfant »
         doc->InsertObject(Enfant,op,NULL) ;   //on l'insert dans notre document, dans op.

         var MaLumière=new(LightObject) ;            // MaLumière est créé, c'est un objet Lumière.
         MaLumière->SetName("Mon spot") ;            // son nom est « Mon spot »
         doc->InsertObject(MaLumière,Enfant,NULL) ;  //on l'insert au le projet dans Enfant.
   }
}
```

Explications : La fonction objet->SetName(nom) permet d'attribuer un nom à un objet.
La fonction document->InsertObjet(nouvel_objet,objet_parent,objet_précédent) permet d'intégré l'objet créé (en mémoire) à notre scène 3D. La valeur NULL informe la fonction qu'il ne faut pas prendre en compte le paramètre qu'elle désigne.

A savoir :

Une expression COFFEE est exécutée en permanence, à chaque actualisation de la scène, c'est pourquoi dans notre exemple il est indispensable de créer nos objets uniquement si l'objet op n'a pas d'enfant, ainsi dès que ceux-ci sont créés, l'expression COFFEE les détecte et revient à Cinema 4D sans créer de nouveaux objets. Si il n'y avait pas cette sécurité, l'expression fabriquerait les objets en permanence et notre objet op verrait sa liste de sous-objets augmenter jusqu'à ce Cinema 4D soit saturé par ces milliers d'objets !

En ce qui concerne notre exemple, si vous effacez les enfants, ils seront instantanément recréés et si vous déplacez l'expression COFFEE sur un autre objet sans enfants, les enfants seront immédiatement créés.

Fin de cette troisième partie.