Forum: Moteur de jeu GameBlender et alternatives

Topic: Jeu de plateau : chance, probabilités et talent combinés

Subject: Re: Jeu de plateau : chance, probabilités et talent combinés

PostÃ© par: Bobibou

Contribution le : 6/1/2011 20:25:48

Citation :
Je sais programmer en python Mais j'arrive pas a digerer l'API de blender --' Et surtout, je ne trouve pas de tuto BIEN pour apprendre...
Enfin bref, c'est pas le sujet.
Allez, un petit résumé des essentiels juste pour toi (en fait non, je l'avais déjà posté ailleurs, mais je recycle.

) :
Ces deux lignes ont exactement le même rôle, mais ne sont pas comme ça par hasard. gl = GameLogic # 2.49 # La version 2.49, elle n'est pas obligatoire. Son seul rôle est de dire que 'gl' sera la même chose que GameLogic. # GameLogic est un ensemble de fonctions réservées au BGE. C'est un des modules propres à Blender, un module. # Cette ligne ne sert qu'à se simplifier la vie en utilisant dans le code 'gl' plutôt que 'GameLogic' qui est bien plus long. # Rien ne t'empêche de ne pas mettre cette ligne mais d'utiliser à la place 'GameLogic' qui est déjà importé par défaut au démarrage. from bge import logic as gl # 2.5 # Cette ligne importe le module logic situé dans le module bge. Comme pour la version 2.49, on utilise l'alias 'gl' grâce au mot clef 'as' qui simplifie le code # mais on aurait très bien pu conserver 'bge.logic' à la place. cont = gl.getCurrentController() # Ici, on récupère la brique controller qui a lancé le script et on l'enregistre dans la variable 'cont' (appelle la comme bon te semble. # Cette ligne est présente dans la plupart des scripts pour le bge car le controller est, avec la scène, le seul lien vers l'extérieur du code, vers le jeu et les objets. own = cont.owner # On enregistre dans la variable 'own' l'objet possesseur du controller ayant lancé le script. C'est extrêmement utile. sensor1 = cont.sensors["nom_du_sensor"] # Plus complexe : on récupère la liste des sensors attachés au controller 'cont' grâce à "cont.sensors" # puis en en extrait l'élément nommé "nom_du_sensor" à l'aide des crochets. # On enregistre le sensor ainsi récupéré dans la variable 'sensor1'. # Ce sensor à lui aussi un attribut "owner" pour récupérer l'objet le possédant, ce qui peut s'avérer très utile, mais c'est moins utilisé. if sensor1.positive : # On fait ici une condition testant si le sensor1 est activé ou non. # Le texte qui suit doit impérativement être indenté : c'est une règle de Python nom = own["property"] # On récupère la propriété de l'objet 'own' nommée 'property' et on l'enregistre dans la variable 'nom' scene = gl.getCurrentScene() # On récupère la scène courante dans laquelle s'est lancé le script. C'est avec gl.getCurrentController() # le seul lien entre Python et le jeu, l'espace 3D. objlist = scene.objects # On récupère la liste de tous les objets de la scène. # Cette liste se présente comme celle des sensors. Il faut donc utiliser les crochets pour récupérer un élément. obj = objlist[nom] # On récupère ici dans la variable 'obj' l'objet dont le nom est celui indiqué par la propriété nommé 'property' de l'objet own. # 'nom' est une variable, donc on ne met pas de guillemets autour. position = objet.position orientation = objet.orientation # On récupère toutes les infos que l'on veut sur l'objet (il y en a plein d'autres, cf la doc). objet.position = [0, 0, 0] # On place

l'objet à l'origine objet.orientation = [[1, 0, 0], [0, 1, 0], [0, 0, 1]] # On annule toute rotation dessus en le remettant à l'orientation [0, 0, 0] # L'orientation est une matrice. C'est pas trop dur à comprendre, et si t'es motivé que t'expliquerais.

Avec ça, il y a déjà de quoi s'amuser.