FICHE Blender 6 : premier contact avec Python

LOGICIELS

Il faut installer Python en version 3 : **https://www.python.org/**

On peut tester, **sans les enregistrer**, une suite de commande directement sur la console-Python

On peut **enregistrer** un code Python (fichier dans l'extension est .py) à travers :

- **IDLE Python** : ***Integrated Development Environment*** téléchargé avec Python.
- **Notepad++** (en oubliant pas de préciser le langage (Python !) et l'encodage UTF-8 des caractères.
- Il en existe d'autres que nous utiliserons par la suite.

Accès à la **console de l'ordinateur** sous Windows :

Sous Windows 10	**Sous Windows 7 ou moins**
Start Menu → All Apps → Windows System → Command Prompt Menu Démarrer → Système Windows → Invite de Commande	Menu Démarrer → Exécuter.

CODE PYTHON

Affichage de la console d'exécution directe	Changer le menu du haut à gauche de INFO vers PYTHON CONSOLE. On voit apparaitre 3 chevrons qui caractérise le fait qu'on puisse rentrer directement et en direct du code Python.
Affichage de la fenêtre Text Editor pour enregistrer un code Python	Changer le menu du haut à gauche de INFO vers TEXT EDITOR. Vous pouvez maintenant taper un code pour l'exécuter plus tard. Pour lancer le script : **RUN SCRIPT**. Pour enregistrer : **SAVE AS.** Pour convertir les 4 espaces en tabulation : **FORMAT - Convert Whitespace**
Affichage de la console de contrôle de Blender	**info - Window - Toggle System Console**
Module bpy	Le module BlenderPython (bpy) est directement intégrée à l'interface Python. Vous pouvez donc utiliser toutes les fonctions de ce module directement . Dans TEXT EDITOR, il faut impérativement importer le module : `import bpy`

Commentaires	Permet de renseigner le code sans l'afficher lors de l'exécution.
	`#Toute cette ligne est un commentaire.`
	`print ("Bonjour le monde") #Ceci est également un commentaire`
	Affiche juste : Bonjour le monde

CALCULS

Les 4 opérateurs standards :	**+ - * /**
La puissance :	**print(2**3)** affiche 8 car $2^3 = 8$
La division entière :	**print(9//4)** affiche 2 car 9 = 4*2 + 1
Le reste de la division entière :	**print(9%4)** affiche 1 car 9 = 4*2 + 1

ENCODAGE

Précision du langage utilisé : **Encodage des caractères :**	`#!/usr/bin/env python` `# -*-coding:Utf-8 -*`

VARIABLES (1er contact)

Définition	Espace de stockage d'informations accessible via un nom. La variable possède une adresse mémoire interne, un type et un contenu composée concrètement de 0 et de 1.

Exemples :	`monTexte = "Nouveauté"`
	`print(monTexte)` affiche `Nouveauté`
	
	`maNote = 3`
	`print(maNote)` affiche `3`
	
	`maNote = 3`
	`print(maNote*2)` affiche `6`

CREATION DE CUBE en manipulant l'interface avec du code Python

Création d'un cube	La ligne ci-dessous permet de créer un cube à la position actuelle du curseur 3D.
	`bpy.ops.mesh.primitive_cube_add()`
	
	Pour définir une autre taille que 1 à votre cube :
	`bpy.ops.mesh.primitive_cube_add( radius=2.0 )`
	
	Pour définir une coordonnée précise, il faut donner les coordonnées XYZ dans un tuple ( ) :
	`bpy.ops.mesh.primitive_cube_add( location=(5,5,0) )`

CODAGE DES BOUCLES FOR NUMERIQUE

Codage de la boucle POUR (FOR)

Exemple de boucle FOR numérique utilisant un range:

```
# Pour i partant de 5 et tant que i<30, en augmentant de 10.
for i in range(5,30,10):
    print("Cas ",i) # On obtient 5, 15 et 25.
    print (i," x2 =",i*2)
```

Remarque :
Si on tape range(30), on compte de un en un de 0 à 29, c'est à dire tant que i<30
Si on tape range(5,30), on compte de un en un de 5 jusqu'à 29, c'est à dire tant que i<30.

Avec une chaîne de caractère : **Affichage des caractères un à un**

```
y = "Hello Wordl!"   # On crée le string y
# x est ici un caractère
for x in y:          # Pour chaque caractère x contenu dans y
      print(x)       # Affiche x à l'écran
```

Avec une chaîne de caractère : **Affichage des caractères un à un**

```
y = "Hello Wordl!"      # On crée le string y
nLongueur = len(y)
# x est ici un integer qui correspond au numéro de case
for x in range(nLongueur):  # x est ici le numéro de la case
   print(y[x])
```

Lecture des **éléments** contenus dans une **liste** :

```
y = [1,2,9,8,"ah ah ah",45,(45,45,45)]
for x in y:
      print(x)
```

Attention : les instructions à traiter sont comprises par Python à l'aide de la **tabulation**.
Pensez à les faire afficher par Notepad++ dans le menu Affichage – Symboles spéciaux.

BOUCLE FOR et BLENDER

Cubes en ligne

```
import bpy # On importe la bibliothèque Blender Python
for x in range(-16,17,4):
    bpy.ops.mesh.primitive_cube_add(location = (x,0,0))
```

Cubes en cercle

```
import bpy # On importe la bibliothèque Blender Python
import math # On importe la bibliothèque math pour la trigo
rayon = 10
for angle_degre in range(-0,360,36):
    angle = math.radians(angle_degre) # convertir en radians
    x = rayon*math.cos(angle)
    y = rayon*math.sin(angle)
    bpy.ops.mesh.primitive_cube_add(location=(x,y,0))
```