勉強会: Python を使用してみる

2021-06-22

株式会社キングプリンターズ システム課

はじめに

- 今回の勉強会では Python をとにかく使ってみます 😤
- すでに Python バリバリやってますという方には内容が簡単過ぎるかもしれません (※すみません) 🙏

Python を学ぶ必要があるのか？

- 世の中には様々な言語が存在します
  - 絶対に **Python でないと解決出来ない問題** というのは少ないと思います
  - しかし、**Python だと解決出来る** という問題が存在するのも事実だと思います
- 適材適所で必要な言語を学び、問題解決に使えれば良いなと思いました 😄

Python とは何か？

- Pythonとは、組み込み開発、WEBアプリケーション、デスクトップアプリケーションなどで利用されるプログラム言語です

Python の特徴は

- 人工知能や機械学習の分野では、Python が使われているケースが多いそうです
- その他、何かのツールを操作する言語として使用されるケースがあります

Python インストール手順

- Mac の方は Homebrewを利用してpyenvをインストールします
  ※詳しくは説明出来ませんすみません
- Homebrewを利用してpyenvをインストールします

```
$ brew install pyenv
```

- Pythonをインストール

```
$ pyenv install --list
$ pyenv install 3.8.3
```

Python の確認 (Windows 10)

```
> python --version
Python 3.8.4
```

Python を対話モードで使用してみる

- 対話モードの開始

```
> python
```

- 四則計算してみる

```
>>> 1+1 [※Enter]
2
```

- 対話モードの終了

```
>>> [Ctrl + z] [※Enter]
```

Python をコードを実行してみる

※ここでは 'C:\tmp\hello_python' ディレクトリで作業しました

- 以下の内容の hello.py ファイルを作成します

```
print("ハローワールド")
```

- コマンドラインから実行します

```
> cd "C:\tmp\hello_python"
> python hello.py
ハローワールド
```

ここまでのまとめ

- 対話モードで Python を使うことが出来ました 😄
- コマンドラインから Python スクリプトを実行することが出来ました 😋

Python の基本的な書き方について

※全てを網羅するものではありません

- 変数定義

```
# 数値
count = 24

# 文字列
word = "Hello"

# リスト
list = ["red", "blue", "yellow"]
```

- if文

```
a = 15
if a > 10:
    print("aは10より大きいです")
elif a == 10:
    print("aは10です")
else:
    print("aは10未満です")
```

```
#結果
aは10より大きいです
```

- ループ文

```
words = ['Japanese', 'English', 'French']
for word in words:
    print (word)
```

```
#結果
Japanese
English
French
```

ここまでのまとめ

- Python は対話モードでも手軽に使うことが出来そうです 😋

次のステップとして タートルグラフィックス🐢という図形を描くツールを使ってみます

- タートルグラフィックスを使用してみる

```
> Python
>>> import turtle
>>> turtle.forward(100)
[Enter]
※GUIが立ち上がり矢印が右に進みます
>>> [Ctrl + z][※Enter]
```

- 四角形を描いてみる

```
> Python
>>> import turtle
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
>>> turtle.left(90)
>>> turtle.forward(100)
```

※Enter操作は省略

ここまでのまとめ

- Python はシンプルな言語なのでツールを操作する言語として使われることがあります 😋
- また、タートルグラフィックスは視覚的に確認出来るのでプログラミング教育に良さそうです

ここからが本番です 😤

- 実際に Python は様々なツールを操作する言語として採用されていると思います
- CGアニメやゲームで使用される3DCGを作成するツールとして**統合型3DCGソフト**というものがあります
  - **DCC(Digital Content Creation)ツール**と呼ばれます
  - 有名なのは Autodesk社の Maya、3dsMaxなどです ※年間サブスクが高額です(約29万円)😫
  - また、オープンソースで Blender というソフトもあります
- それらDCCツールの**操作言語としてPython**が広く使用されている現状があります
- そして3DCG業界ではDCCツールでPythonを使用するTA(テクニカルアーティスト)という役割が確立されつつあるそうです
  - エンジニアにとって素晴らしい夢のある話ですね 😋

次のステップとして Python で Blender を操作してみます

- Blender 2.8LTS こちらを使用しました

Blender オブジェクトの一覧を表示してみます

[Scripting]タブ → 左ペイン・中段[ビュー:コンソール]

```
for ob in bpy.data.objects:
    print(ob.name)
```

```
# 結果
Camera
Cube
Light
```

- カメラ、キューブ、ライトが存在することが分かります

立体(※タイプが「MESH」であるオブジェクト)の一覧表示

```
for ob in bpy.data.objects:
    if ob.type == "MESH":
        print(ob.name)
```

```
# 結果
Cube
```

- メッシュオブジェクトは Cube だけだと分かります

Blender 画面の一覧を表示してみます

```
for ob in bpy.data.screens:
    print(ob.name)
```

```
Animation
Compositing
Layout
Modeling
Rendering
Scripting
Sculpting
Shading
Texture Paint
UV Editing
```

- 画面の一覧が表示されます

Cube を操作してみます

- Cube を取得します

```
ob = bpy.data.objects['Cube']
```

- Cube を移動します

```
ob.location = (3, -2, 0)
```

- Cube を回転します

```
import math
ob.rotation_euler = (math.radians(45.0), 0, 0)
```

- Cube を拡縮します

```
ob.scale = (2.0, 3.0, 4.0)
```

- Cube の色を変えます

```
ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0].default_value = (0, 1, 0, 1)
```

ここまでのまとめ

- Blender という DCCツールを Python で操作することが出来ます
- このように Python は**他のツールを操作する言語**として非常に有用だと分かります 😋

次のステップとして Python で Blender のアニメーションを作成してみます

Blender ではアニメーションはキーフレームで表現されます

- デフォルトで 24FPS に設定されています
- 24FPS で1秒ですが Blender のビューポートでは環境によると処理落ちして表示されるので、ここでは深く考えません

Cube を Python から操作して内容をキーフレームに登録します

- 対話モードだと長いので Blender の組み込み Python エディタを使用します
  [Scripting]タブ → 中ペイン[エディタ] → [+新規] を押します
- 以下のコードをエディタに貼り付けて横の [▷]スクリプト実行ボタンを押します
- 成功するとアニメーションが登録されます
  - [Animation]タブに移動し下部バーの再生を操作します
  - 3Dビューのシェーディング(※中央ペイン右上)の地球儀アイコンの右端を選択します
  - 再生範囲：開始:1, 終了:48 に変更します
  - アニメーション再生ボタンを押します

```python
import math
import bpy
ob = bpy.data.objects['Cube']

# 0f
ob.location = (0, 0, 0)
ob.rotation_euler = (0, 0, 0)
ob.scale = (1, 1, 1)
mat = ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0]
mat.default_value = (1, 1, 1, 1) # white
ob.keyframe_insert('location', frame = 0)
ob.keyframe_insert('rotation_euler', frame = 0)
ob.keyframe_insert('scale', frame = 0)
mat.keyframe_insert('default_value', frame = 0)

# 12f
ob.location = (2, -3, 4)
ob.rotation_euler = (math.radians(45.0), 0, 0)
ob.scale = (1.5, 2, 3)
mat = ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0]
mat.default_value = (0, 1, 0, 1) # green
ob.keyframe_insert('location', frame = 12)
ob.keyframe_insert('rotation_euler', frame = 12)
ob.keyframe_insert('scale', frame = 12)
mat.keyframe_insert('default_value', frame = 12)

# 24f
ob.location = (-2, 2, -1)
ob.rotation_euler = (0, math.radians(45.0), 0)
ob.scale = (0.75, 1, 0.5)
mat = ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0]
mat.default_value = (1, 0, 0, 1) # red
ob.keyframe_insert('location', frame = 24)
ob.keyframe_insert('rotation_euler', frame = 24)
ob.keyframe_insert('scale', frame = 24)
mat.keyframe_insert('default_value', frame = 24)

# 36f
ob.location = (3, -2, 3)
ob.rotation_euler = (0, 0, math.radians(45.0))
ob.scale = (3, -2, 5)
mat = ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0]
mat.default_value = (0, 0, 1, 1) # blue
ob.keyframe_insert('location', frame = 36)
ob.keyframe_insert('rotation_euler', frame = 36)
ob.keyframe_insert('scale', frame = 36)
mat.keyframe_insert('default_value', frame = 36)

# 48f
ob.location = (0, 0, 0)
ob.rotation_euler = (0, 0, 0)
ob.scale = (1, 1, 1)
mat = ob.data.materials[0].node_tree.nodes["Principled BSDF"].inputs[0]
mat.default_value = (1, 1, 1, 1)
ob.keyframe_insert('location', frame = 48)
ob.keyframe_insert('rotation_euler', frame = 48)
ob.keyframe_insert('scale', frame = 48)
mat.keyframe_insert('default_value', frame = 48)
```

ここまでのまとめ

- 今回の Python で Blender を操作する内容はプログラミングスキルとしては大した事をしていません
    - しかし何かのツールを Python で操作することが出来れば、それは他の人にとって価値のある結果になる可能性があります

最後のまとめ 😋

- Python という道具が使えればエンジニアとして出来ることが増えそうです
  - 20年前、WEBという媒体が出現したことでプログラマ(エンジニア)という職種がより一般化したと思います
    - WEBというコンテンツを表現する手段として HTML, CSS, Javascript, バックエンド etc の技術が必要でした
  - 現在では今後需要が伸び続ける分野は AR/VR であると予想されています
    - そして、そこで使用される3DCGという分野でもプログラミングスキルが必要とされる時代になったと思いました

ご清聴ありがとうございました 🙇