

著者の新しいチュートリアルや 3D モデルの更新情報を知りたい場合は、彼の YouTube チャンネルを購読してください (<https://www.youtube.com/@FrancescoMilanese85>).

この PDF は無料です。気軽にシェアしてください！

こんにちは！

このチュートリアルでは、Blender の Python 関数、UV export layout を使用して、シーン内の全ての利用可能な MESH オブジェクトの UV maps レイアウトをエクスポートする方法を学びます。

このスクリプトは、3D ストックウェブサイトで 3D シーンを公開する際など、UV maps を画像として提供する必要がある場合に役立ちます。

このチュートリアルは Blender のバージョン 3.3 を使用して記録されました。時々、Blender の開発者は Blender Python APIs (BPY と略します) に変更を加えますが、このスクリプトはバージョン 3.0 以降で動作するはずです。

使用する `bpy.ops` 関数は非常にシンプルで、作成する画像のファイルパスと UV サイズ (ピクセル単位) の 2 つのパラメーターが必要です。しかし、シーン内の全ての利用可能な MESH オブジェクトに対してこれを使用する必要があるため、MESH オブジェクトに対して FOR ステートメントを使用します。

さらに、私はファイルの拡張子を指定します。これはデフォルトでは PNG です。

実際には、レイアウトを PNG としてエクスポートしますが、SVG や EPS ファイルとして UV レイアウトをエクスポートすることもできるので、指定方法を示すためにこれを行います。

多くのオブジェクト、それぞれに独自の UV レイアウトが含まれているため、私のアセットである Desk set 3 を使用しています。

スクリプトがすべてのレイアウトをエクスポートする方法を見ていきましょう。

もちろん、このチュートリアルをフォローし学ぶために、このアセットを持っている必要はありません！

さて、始めましょう！

BLEND ファイルを開きます。

新規のファイルから始めることもできますが、プロジェクトと同じフォルダに PNG ファイルを保存するため、どこかに保存する必要があります。

基本的なインポートをいくつか行いましょう：

```
import bpy  
from bpy import *
```

前述の通り、*export_layout* 関数は 2 つのパラメータを必要とするので、2 つの変数を定義しましょう。

関数自体に値を書くこともできますが、わかりやすさを重視してここで定義します：

```
UVpath = bpy.path.abspath("//") + "UV-LAYOUT---"  
UVsize = (2048, 2048)
```

ご覧の通り、*bpy path abspath* は BLEND ファイルのディスク上の絶対パスを返します。

すべての PNG ファイルに接頭辞、UV LAYOUT を追加しますが、これを変更したり削除したりすることも可能です。

UV size パラメータは、作成する画像のピクセル単位の寸法である 2 つの数値のタプルです。ここでは 2048 と書いているので、2k の画像を作成します。

シーン内の利用可能なすべての MESH オブジェクトで関数を実行するには、FOR 文を作成する必要があります。

したがって、以下のように記述することができます：

```
for o in bpy.data.objects:
```

そして、FOR 文のブロック内に（テキストをインデントするのを忘れないでください）、オブジェクトのタイプにフィルターを追加することができます：

```
if(o.type=="MESH"):
```

条件は明確です：現在の bpy data object のタイプは MESH でなければなりません。

以下の文は IF 内に配置する必要があるので、再度、テキストをインデントするのを忘れないでください！

```
bpy.ops.objects.select_all(action='DESELECT');
```

これは他のすべてのオブジェクトの選択を解除するために、毎回行う必要があります。そうしないと、重なった 2 つ以上の UV マップを持つ画像が得られてしまいます。

```
bpy.context.view_layer.objects.active = o
```

この代入は、シーンのアクティブなオブジェクトを設定し、現在のメッシュを `context.view_layer.objects.active` に割り当てます。

これは、次の OPS (Blender Python operation) が一つのアクティブなオブジェクトに対して実行されるためです。

複数のオブジェクトを選択できますが、一度にアクティブなオブジェクトは一つだけであることを覚えておいてください。

```
bpy.ops.uv.export_layout(filepath = UVpath + o.name, size = UVsize,  
                           mode='PNG')
```

ops 文は、アクティブなオブジェクトのレイアウトを、UV path 変数によって与えられたファイルパスにエクスポートします。これには、UV LAYOUT の接頭辞とアクティブなオブジェクトの名前が含まれ、ピクセル単位で指定されたサイズが適用されます。

```
1 import bpy
2 from bpy import *
3
4
5 UVpath = bpy.path.abspath("//") + "UV-LAYOUT---"
6
7 UVsize = (2048,2048)
8
9
10 for o in bpy.data.objects:
11     if(o.type=="MESH"):
12         bpy.ops.object.select_all(action='DESELECT')
13         bpy.context.view_layer.objects.active = o
14         bpy.ops.uv.export_layout(filepath = UVpath + o.name,size = UVsize, mode='PNG')
```

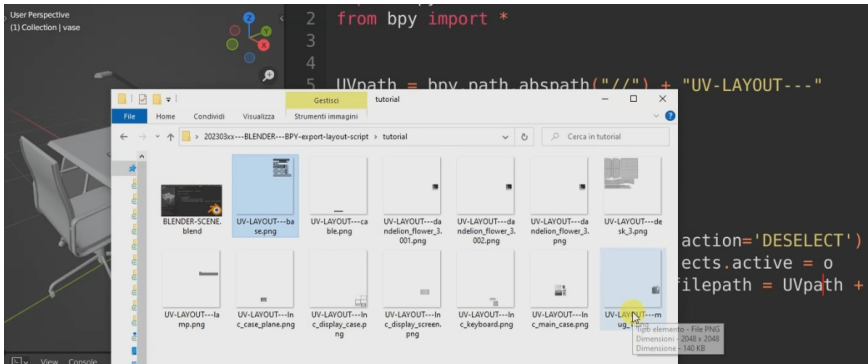
以前述べたように、UV レイアウトを PNG、EPS、または SVG ファイルとしてエクスポートすることができます。

出力フォーマットは、私が示したように、mode パラメータで指定できます。

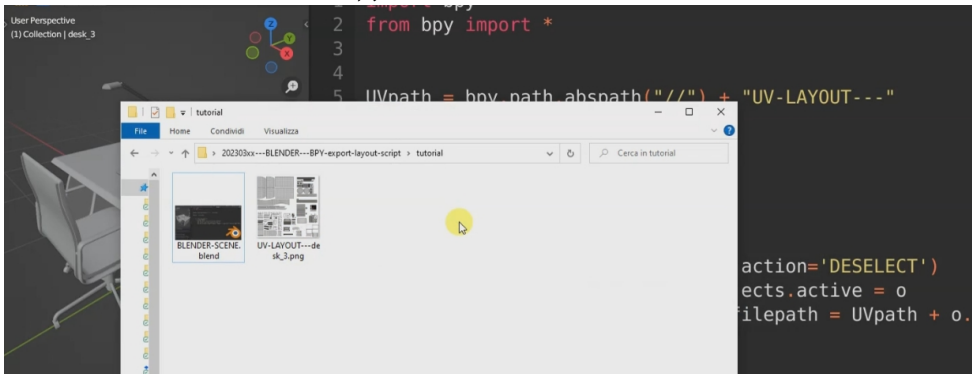
出力フォルダを見てみると、メッシュの数だけ画像が作成されていることがわかります。

BLENDER #004 - BPY 3 チュートリアル - uv.export_layout を使って UV Layouts を画像
としてエクスポートする方法

www.francescomilanese.com --- 2024 --- <https://youtu.be/Szn2rmi9LJU>



私の 3D モデル「Desk set 3」内のメッシュは、重複しない UV 島と
同じレイアウトを共有しているため、すべてのオブジェクトをまと
め、スクリプトを再度実行すると、アセットの重複しない UV レイ
アウトマップが1つだけ得られます。



このチュートリアルを終える前に、あなたに課題を出します：この
スクリプトを編集して、シーン内の利用可能なメッシュの 2k と 4k
の UV layouts をエクスポートし、画像に 2 つの異なる接頭辞（例え
ば、UV2K と UV4K）を提供してください。

このチュートリアルはこれで終わりです！ また近いうちにお会いしまし
よう！