BPY 3 (Blender 3.x und Python) Grundlagen: So exportieren Sie UV-Layouts als PNG-Bilder

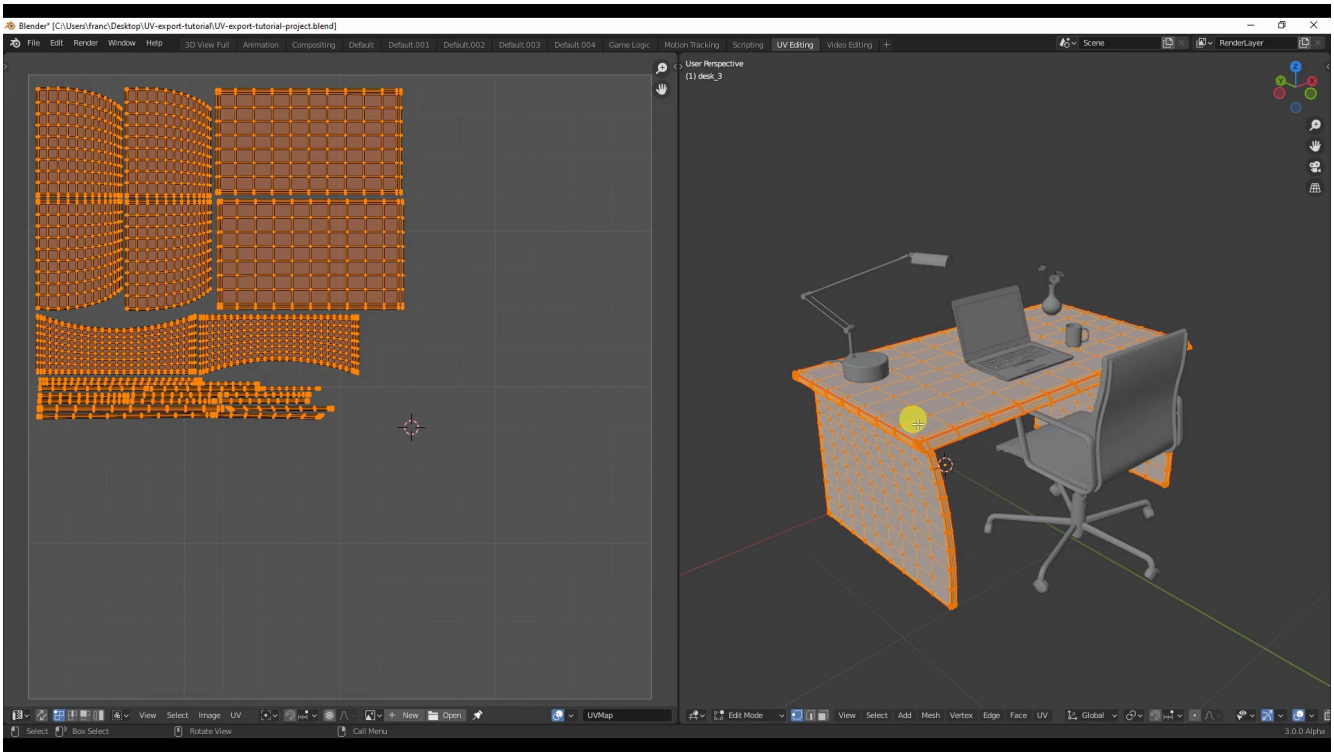
Hallo!

In diesem Tutorial erfahren Sie, wie Sie die funktion **export_layout** in **Blender 3.0 (Alpha)** verwenden, um die **UV**-Karten (oder: Layouts) aller in einer Szene verfügbaren **MESH**-Objekte zu exportieren.

Dieses Skript kann nützlich sein, wenn Sie die **UV**-Karten als **PNG**-Bilder bereitstellen müssen, z. B. wenn Sie Ihre 3D-Szene auf einigen 3D-Stock-Websites veröffentlichen möchten.

Die **bpy.ops**-Funktion, die wir verwenden werden, ist eigentlich ziemlich einfach: Sie erfordert zwei Parameter (den Dateipfad des zu erstellenden Bildes und die **UV**-Größe in Pixeln), aber wir müssen sie für alle in der Szene verfügbaren **MESH**-Objekte verwenden, also verwenden wir eine **FOR**-Anweisung für die **MESH**-Objekte.

Öffnen wir zunächst eine **BLEND**-Datei. Sie können auch mit einer brandneuen Datei beginnen, aber Sie müssen sie irgendwo speichern, da wir die **PNG**-Dateien im selben Ordner des Projekts ablegen.



Lassen Sie uns einige grundlegende Importe durchführen:

```
import bpy

from bpy import *
```

Wie ich bereits sagte, erfordert die **export_layout** Funktion zwei Parameter, also definieren wir zwei Variablen (wir könnten die Werte in die Funktion selbst schreiben, aber ich ziehe es vor, sie hier der Klarheit halber zu definieren)

```
UVpath = bpy.path.abspath(“//”) + “UV-LAYOUT---”

UVsize = (2048, 2048)
```

Wie Sie sehen können, liefert "**bpy.path.abspath**" den absoluten Pfad auf der Festplatte der BLEND-Datei. Wir werden allen Bilddateien ein Präfix ("**UV-LAYOUT---**") hinzufügen, aber Sie können das ändern oder entfernen.

Der Parameter **UVsize** ist ein **Tupel** aus zwei numerischen Werten, die die Abmessungen – in Pixeln – der zu erstellenden Bilder sind; Ich schreibe **2048** und erstelle damit ein **2k**-Bild.

Um die Funktion für alle in einer Szene verfügbaren **MESH**-Objekte auszuführen, müssen wir eine **FOR**-Anweisung erstellen, damit wir schreiben können:

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

Dann können wir in der **FOR**-Anweisung (also: vergessen Sie nicht, den Text einzurücken) einen "Filter" für den Typ des Objekts hinzufügen:

```
if(o.type==”MESH”):
```

Die Bedingung ist klar: Der Typ des aktuellen **bpy.data.object** muss **MESH** sein.

Die folgenden Aussagen müssen in das **IF** eingefügt werden, also vergessen Sie nicht, den Text einzurücken

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

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

Die Zuweisung legt das ACTIVE-Objekt der Szene fest und weist das aktuelle Netz **context.view_layer.objects.active** zu. Wir müssen dies tun, weil die folgende **OPS (Blender Python Operation)** für das einzige ACTIVE-Objekt ausgeführt wird (denken Sie daran: Sie können mehrere Objekte auswählen, aber es gibt nur ein aktives Objekt zu einem Zeitpunkt).

Die **OPS**-Anweisung exportiert das Layout des ACTIVE-Objekts im angegebenen Dateipfad (erstellt von der **UVpath**-Variablen mit dem Präfix "**UV-LAYOUT---**" plus dem Namen des ACTIVE-Objekts) und mit der angegebenen Größe in Pixeln.

Bevor ich dieses Tutorial beende, werde ich Ihnen eine Aufgabe geben: Bearbeiten Sie dieses Skript, um sowohl das **2k**- als auch das **4k-UV-Layout** der in Ihren Szenen verfügbaren Netze zu exportieren, wobei die Bilder mit zwei verschiedenen Präfixen ("**UV-2K---**" und "**UV-4k---**" zum Beispiel) bereitgestellt werden.

Das ist alles! Ich hoffe, Ihnen hat dieses Tutorial gefallen!

Bis bald!