Programando una órbita planetaria con Blender y Python

Osvaldo R. Salazar S.

NERDEARLA

Osvaldo R. Salazar S.

salazarysanchez.com

linktr.ee/osvaldosalazar

TODAY'S DISCUSSION

1. Instalar Blender
2. Controles básicos
3. ¡ A programar :-D !
4. Disfrutar
5. Preguntas

Osvaldo R. Salazar S.


blender


coffee
RUN
Blender Cloud

blender.org/download

Controles básicos

Texturas

solarsystemscope.com/textures


¡A programar :-D !


import bpy
from math import radians
#####
Crear Tierra
#Crear esfera
bpy.ops.mesh.primitive_uv_sphere_add()
so = bpy.context.active_object
#Cambiar nombre
bpy.context.object.name = "Tierra"


# Activar nuestra esfera
obj = bpy.context.active_object

Ubicar nuestro planeta
bpy.data.objects["Tierra"].location[0] = 0.0
bpy.data.objects["Tierra"].location[1] = 0.0
bpy.data.objects["Tierra"].location[2] = 0.0

#Modificar el tamaño
bpy.data.objects["Tierra"].scale[0] = 8.0
bpy.data.objects["Tierra"].scale[1] = 8.0
bpy.data.objects["Tierra"].scale[2] = 8.0

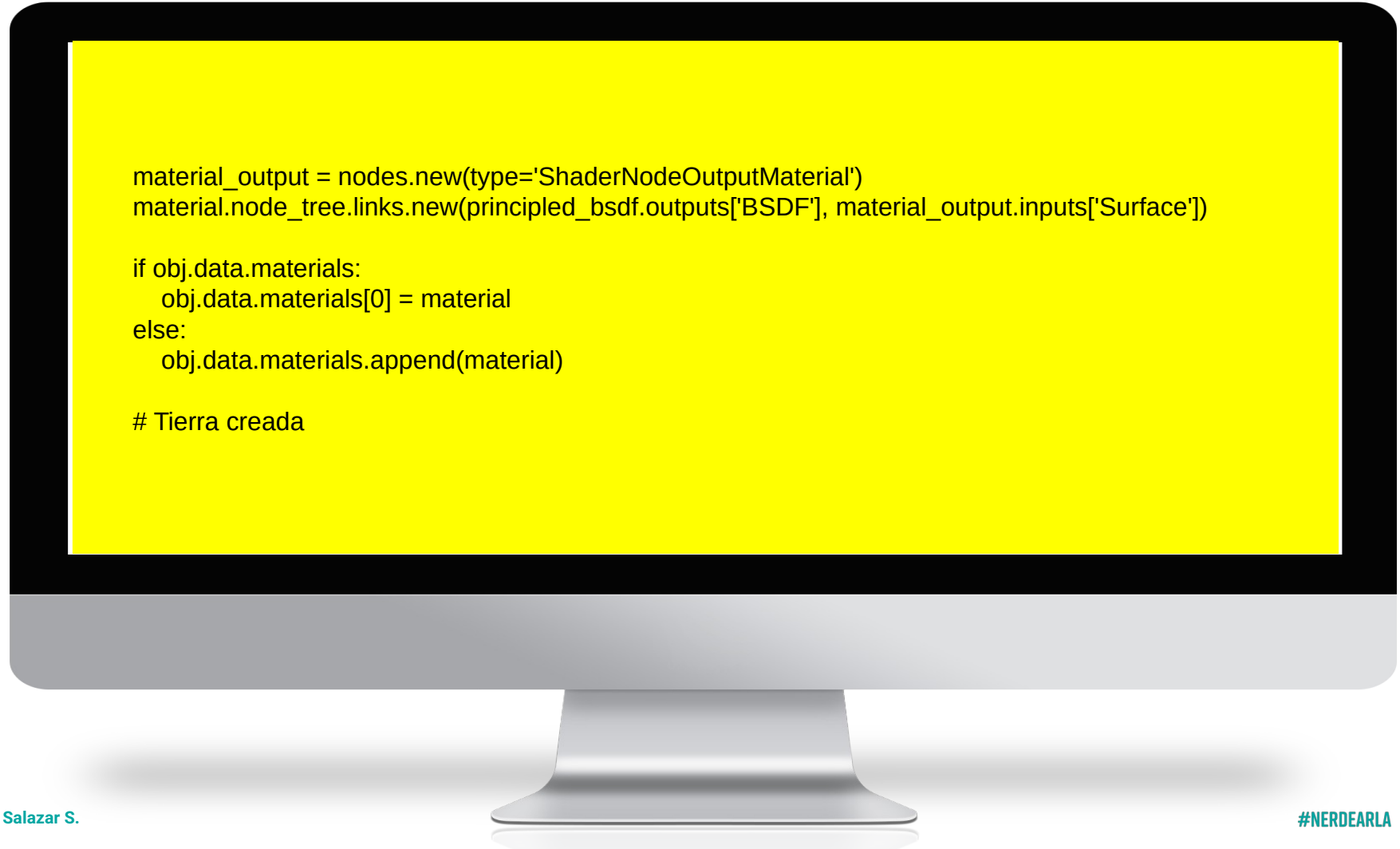

#Crear modificador
mod_subsurf = so.modifiers.new("Mi modificador", "SUBSURF")
mod_subsurf.levels = 3
#Suavizar
bpy.ops.object.shade_smooth()
# Textura de la Tierra
# Crear material nuevo
material = bpy.data.materials.new(name="TexturedMaterial")
material.use_nodes = True # Activar nodos
nodes = material.node_tree.nodes


# Limpiar nodos existentes
for node in nodes:
nodes.remove(node)

# Agregar BSDF
principled_bsdf = nodes.new(type='ShaderNodeBsdfPrincipled')

# Cargar la textura
image_texture = nodes.new(type='ShaderNodeTexImage')
image_path = "/home/chico/2k_earth_daymap.jpg"
image = bpy.data.images.load(image_path)
image_texture.image = image

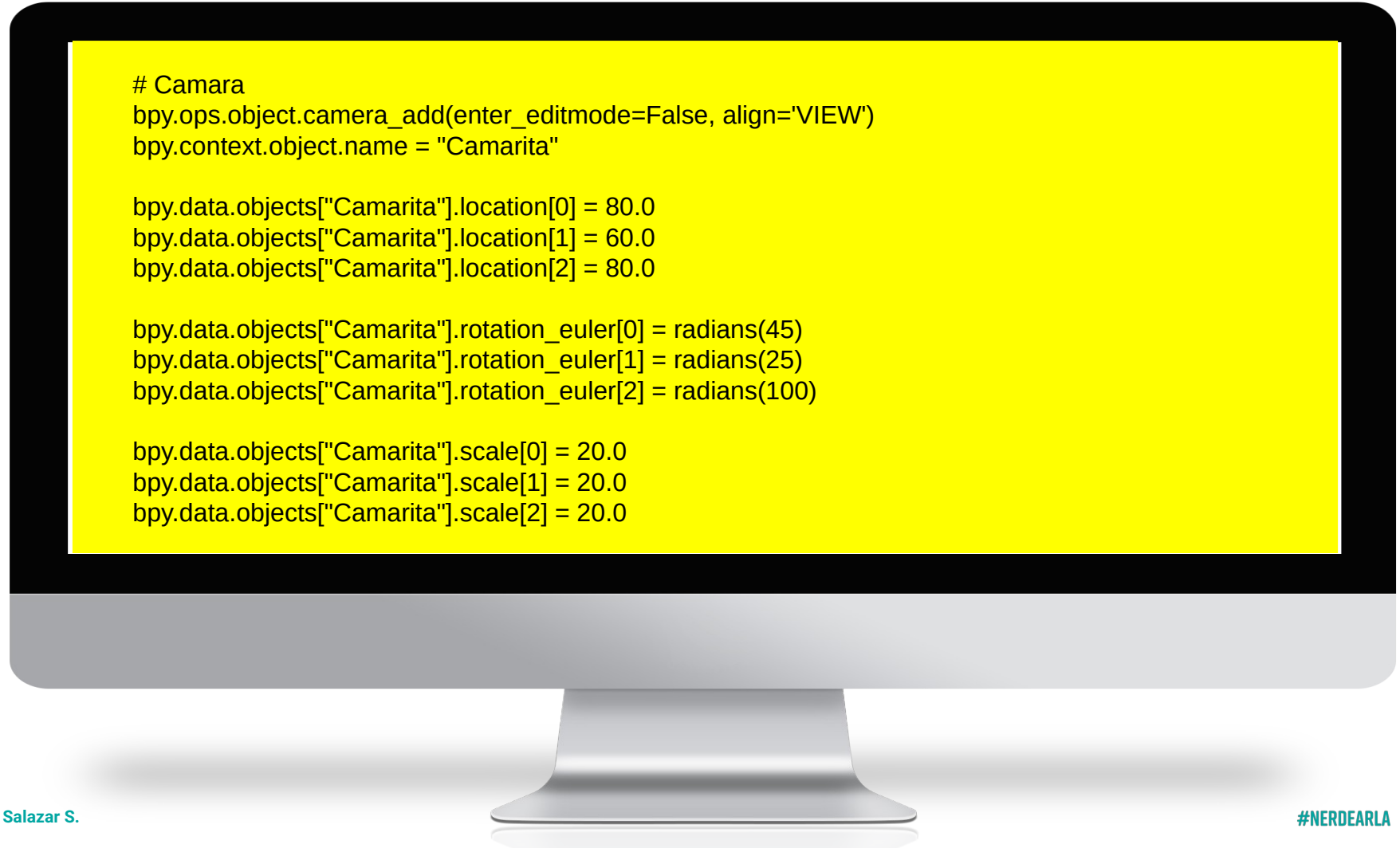
# Conectar la textura a el BSDF
material.node_tree.links.new(image_texture.outputs['Color'], principled_bsdf.inputs['Base Color'])


material_output = nodes.new(type='ShaderNodeOutputMaterial')
material.node_tree.links.new(principled_bsdf.outputs['BSDF'], material_output.inputs['Surface'])
if obj.data.materials:
obj.data.materials[0] = material
else:
obj.data.materials.append(material)
# Tierra creada

```
# Crear Luna
bpy.ops.mesh.primitive_uv_sphere_add()
so = bpy.context.active_object

bpy.context.object.name = "Luna"

obj = bpy.context.active_object

bpy.data.objects["Luna"].location[0] = 0.0
bpy.data.objects["Luna"].location[1] = 48.0
bpy.data.objects["Luna"].location[2] = 0.0

#Modificar el tamaño
bpy.data.objects["Luna"].scale[0] = 4.0
bpy.data.objects["Luna"].scale[1] = 4.0
bpy.data.objects["Luna"].scale[2] = 4.0
```

```
mod_subsurf = so.modifiers.new("Mi modificador", "SUBSURF")
mod_subsurf.levels = 3

bpy.ops.object.shade_smooth()

# Textura de la Luna

material = bpy.data.materials.new(name="TexturedMaterial")
material.use_nodes = True  # Enable nodes

nodes = material.node_tree.nodes

for node in nodes:
    nodes.remove(node)
```

```
principled_bsdf = nodes.new(type='ShaderNodeBsdfPrincipled')

image_texture = nodes.new(type='ShaderNodeTexImage')

image_path = "/home/chico/2k_moon.jpg"
image = bpy.data.images.load(image_path)
image_texture.image = image

material.node_tree.links.new(image_texture.outputs['Color'], principled_bsdf.inputs['Base Color'])

material_output = nodes.new(type='ShaderNodeOutputMaterial')
material.node_tree.links.new(principled_bsdf.outputs['BSDF'], material_output.inputs['Surface'])

if obj.data.materials:
    obj.data.materials[0] = material
else:
    obj.data.materials.append(material)
```


# Camara
bpy.ops.object.camera_add(enter_editmode=False, align='VIEW')
bpy.context.object.name = "Camarita"

bpy.data.objects["Camarita"].location[0] = 80.0
bpy.data.objects["Camarita"].location[1] = 60.0
bpy.data.objects["Camarita"].location[2] = 80.0

bpy.data.objects["Camarita"].rotation_euler[0] = radians(45)
bpy.data.objects["Camarita"].rotation_euler[1] = radians(25)
bpy.data.objects["Camarita"].rotation_euler[2] = radians(100)

bpy.data.objects["Camarita"].scale[0] = 20.0
bpy.data.objects["Camarita"].scale[1] = 20.0
bpy.data.objects["Camarita"].scale[2] = 20.0


# Agregar Luz solar
bpy.ops.object.light_add(type='SUN', align='WORLD', location=(0, 0, 0), scale=(1, 1, 1))
bpy.context.object.data.energy = 10

```
# animar Luna
import numpy as np
import mathutils
import math

Satelite = bpy.data.objects['Luna']
SateliteOrigin = np.array(Satelite.location)
theta = (2*math.pi)/250

def rotateSatelite(scene):
    newTheta = theta*scene.frame_current
    rotationMatrix = np.array([[math.cos(newTheta), math.sin(newTheta), 0],
                               [math.sin(newTheta), math.cos(newTheta), 0],
                               [0, 0, 1]])
    Satelite.location = np.dot(SateliteOrigin, rotationMatrix)
```


def setRotationSatelite():
# Clear old handlers
bpy.app.handlers.frame_change_pre.clear()
# register a new handler
bpy.app.handlers.frame_change_pre.append(rotateSatelite)
setRotationSatelite()


#Para render
bpy.context.scene.render.engine = 'CYCLES'
bpy.context.scene.cycles.samples = 10

Disfrutar

Consulta el código en:

¡GRACIAS :-D !

Grupo de Usuarios de GNU/Linux de La Laguna

Osvaldo R. Salazar S.