Blender Game Engine Python Manual

by

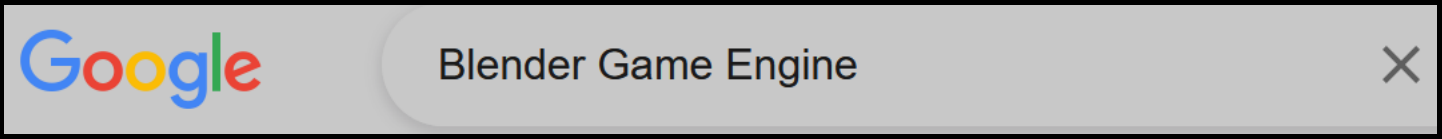
Christopher Andrew Topalian

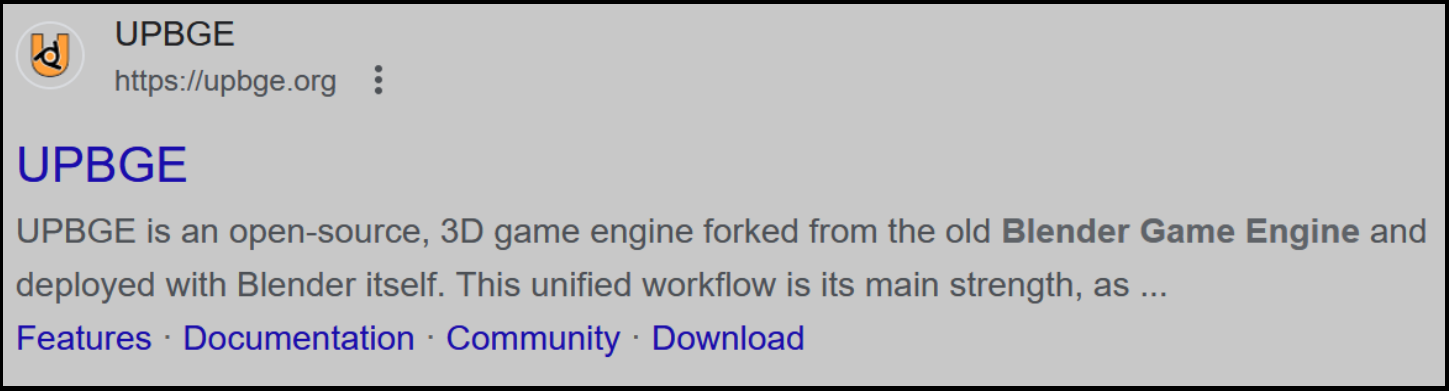
Dedicated
to
God the Father

Table of Contents

Search Google for: Blender Game Engine

Search for: Blender Game Engine




UPBGE
https://upbge.org

UPBGE

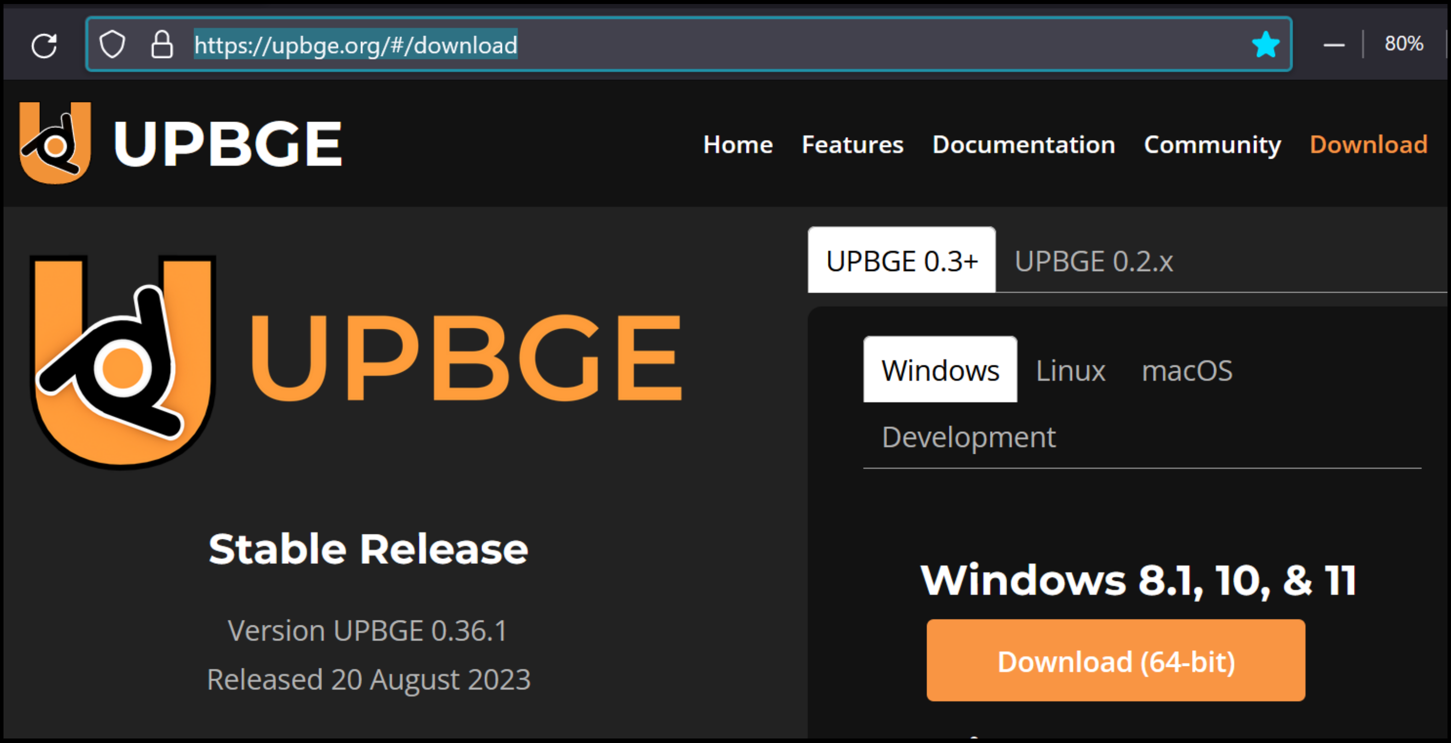
UPBGE is an open-source, 3D game engine forked from the old **Blender Game Engine** and deployed with Blender itself. This unified workflow is its main strength, as ...

Features · Documentation · Community · Download

Left Click: Download

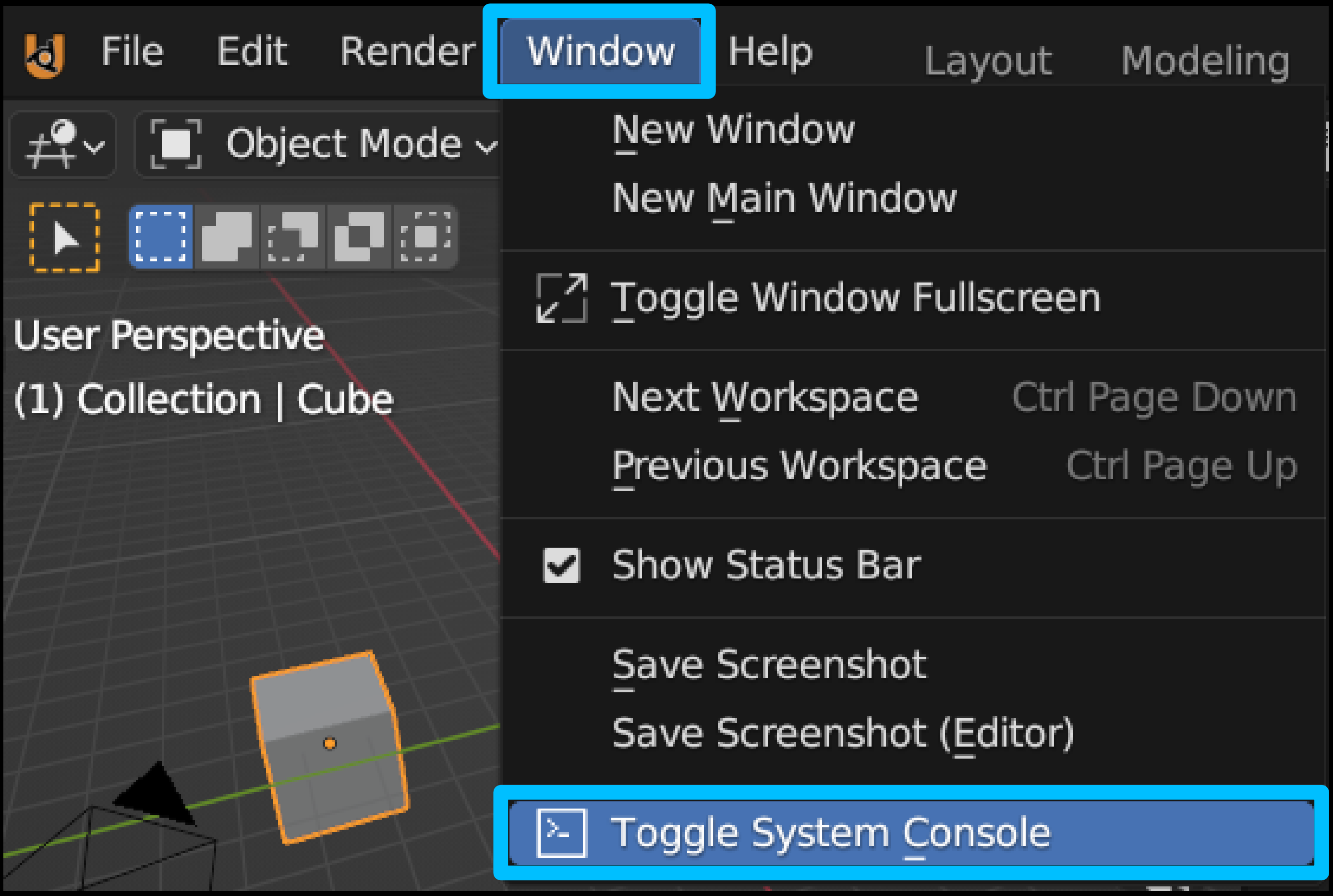
https://upbge.org/#/download

Download: Blender Game Engine



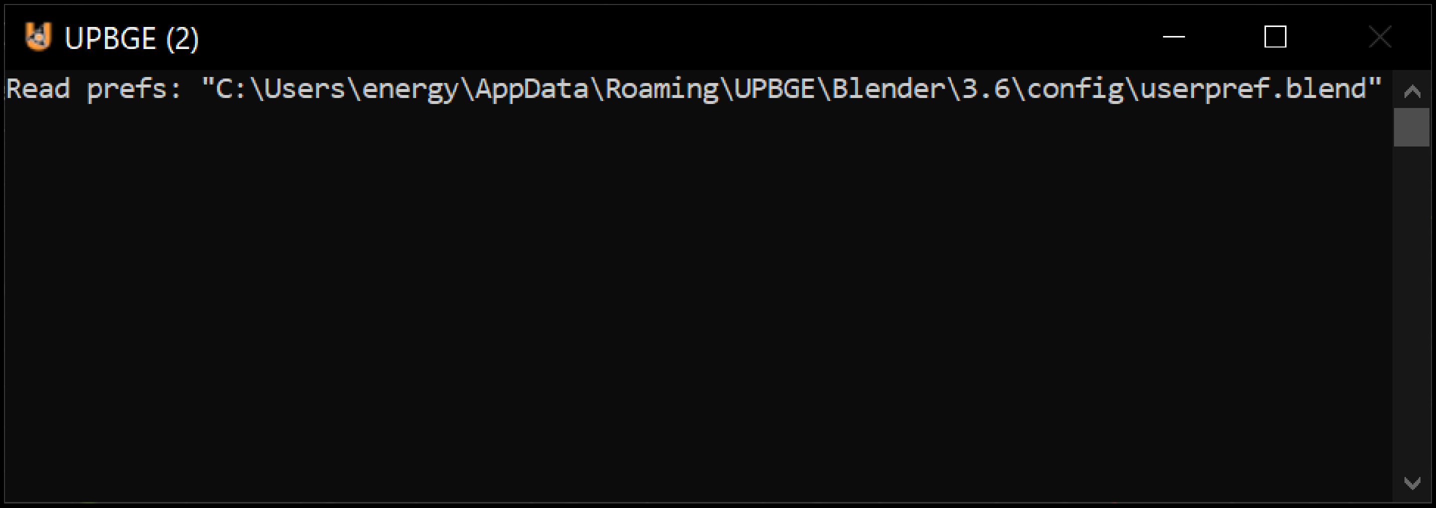
Left Click: Download (64-bit)

System Console



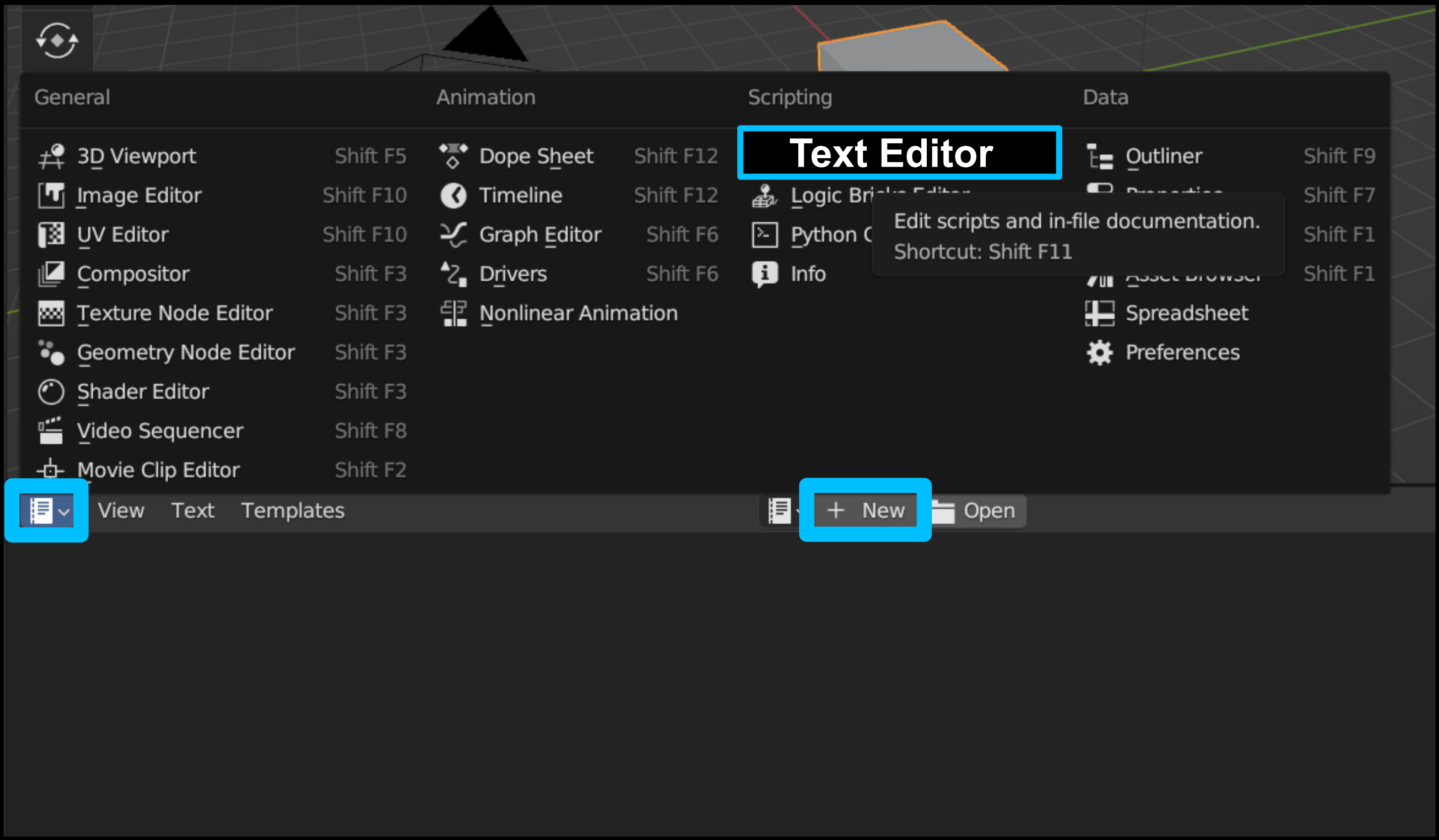
Left Click on: Window menu

Left Click on: Toggle System Console



```
Read prefs: "C:\Users\energy\AppData\Roaming\UPBGE\Blender\3.6\config\userpref.blend"
```

Open Text Editor to Edit Python Scripts

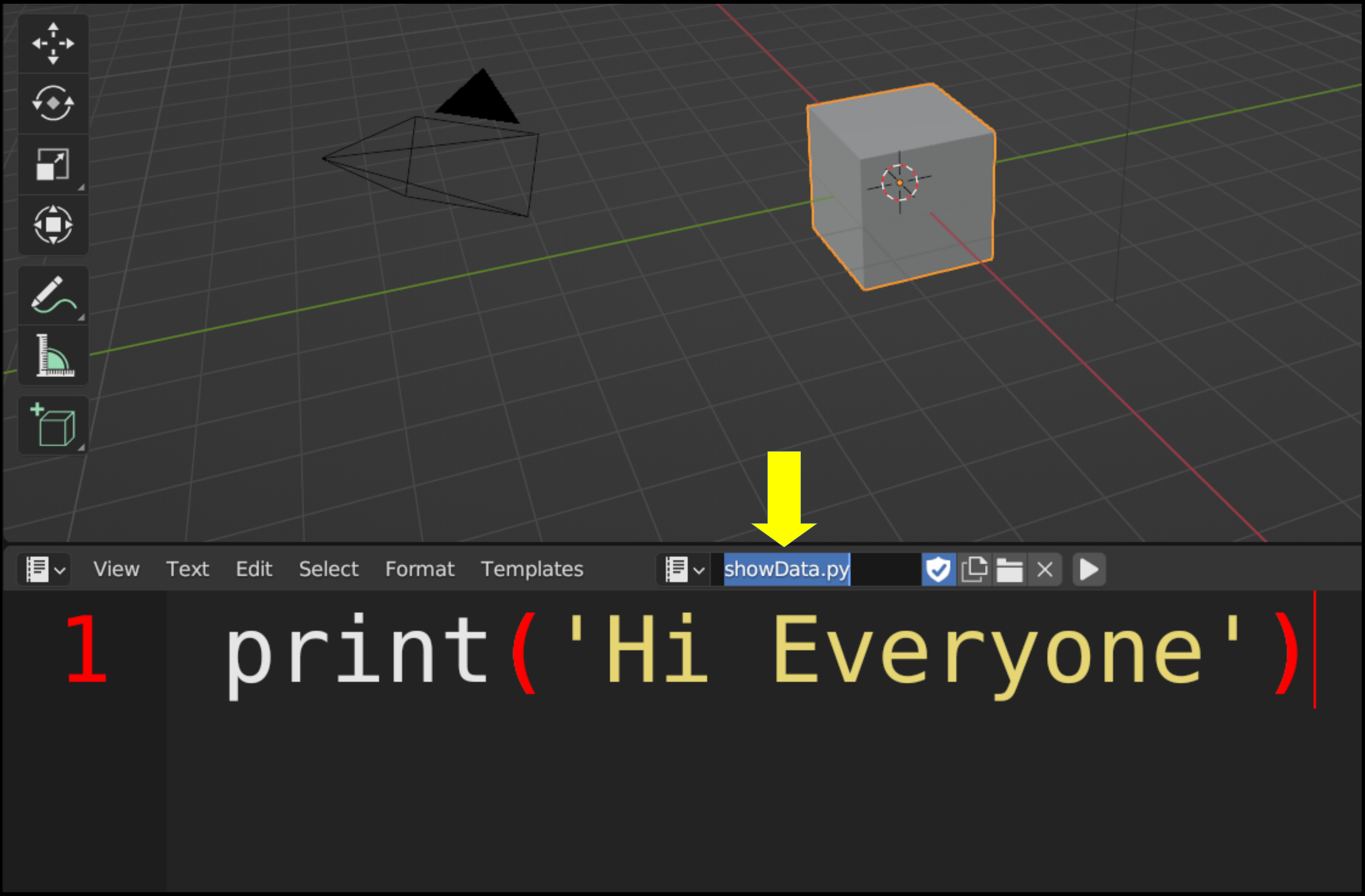


Left Click: Editor Type Button

Left Click: Text Editor

Left Click: New

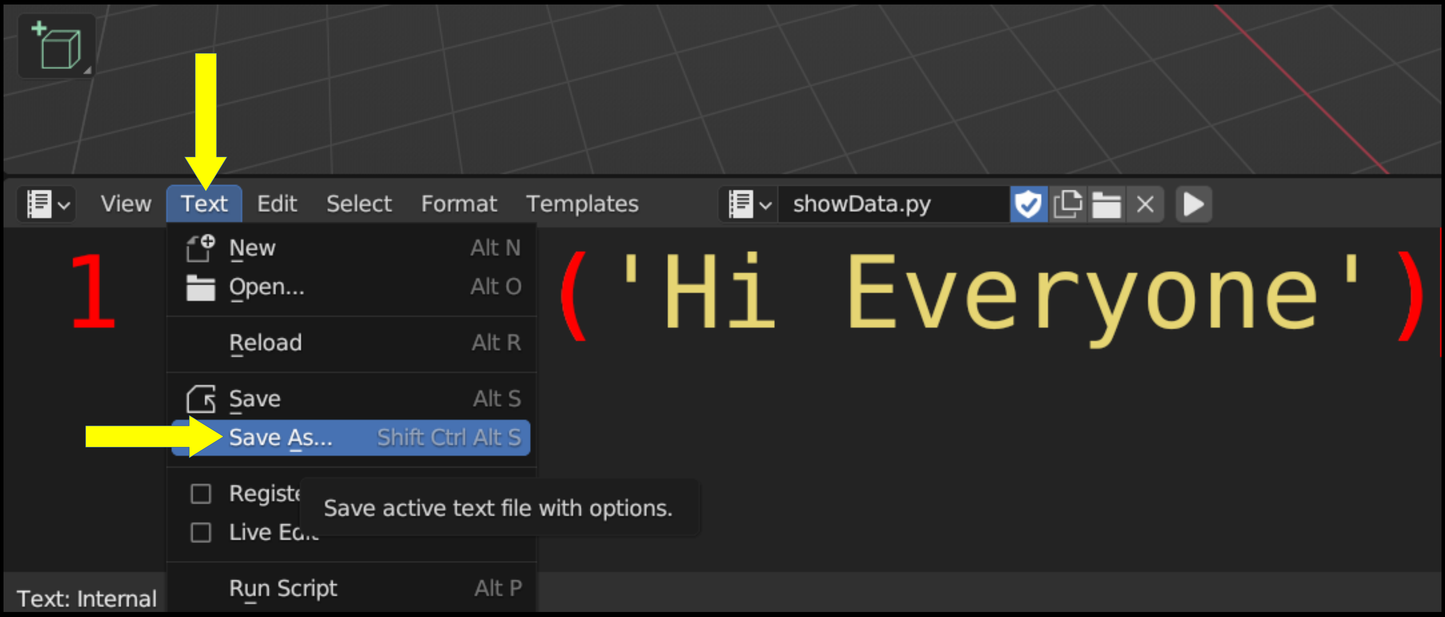
Name Script and Type our Python Code



```
print('Hi Everyone')
```

Save Python Script

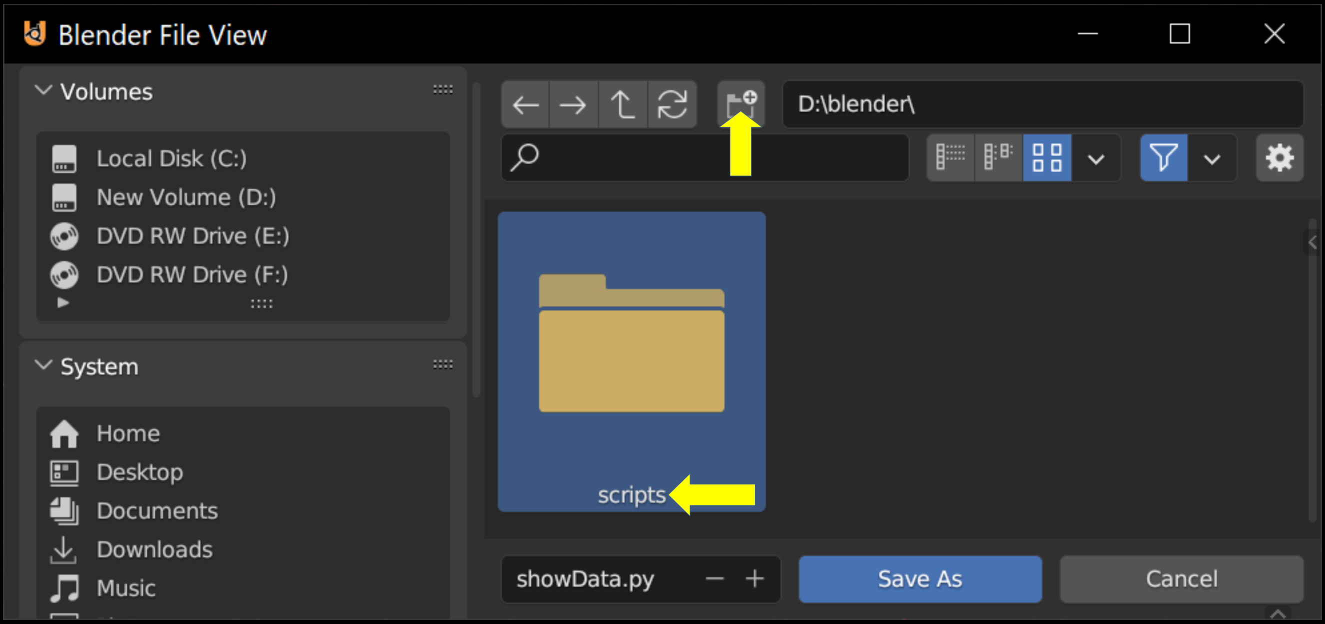
Notice that our Python script has no spaces in the name and that the Python script name ends with the .py extension type, showData.py



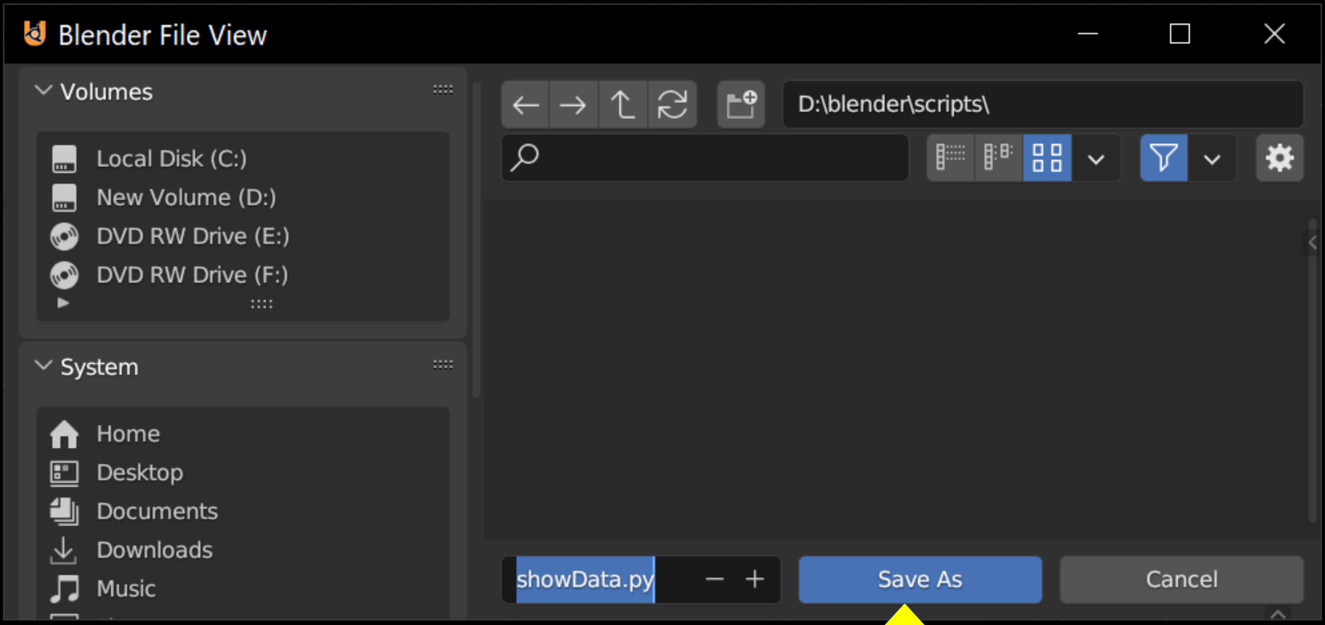
```
('Hi Everyone')
```

Save active text file with options.

We Make a New Folder and Name it, scripts



Double Left Click: scripts folder

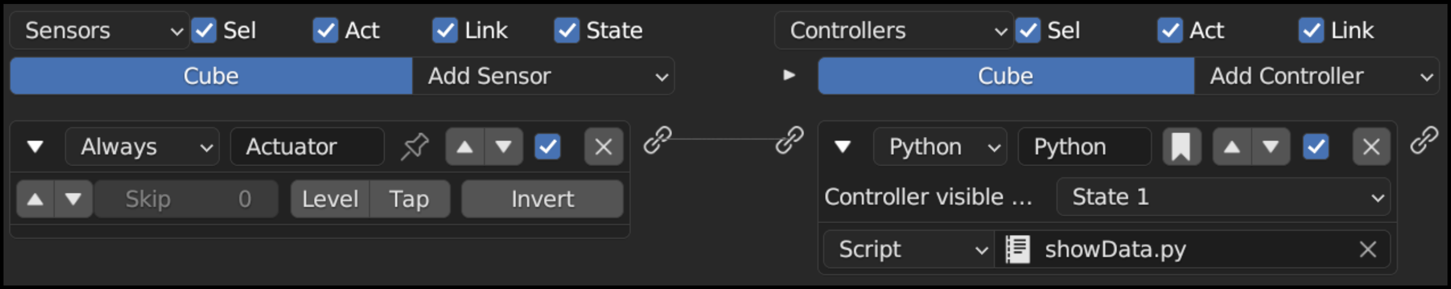


Left Click: Save As

Our Python script named scriptData.py is now saved in our scripts folder.

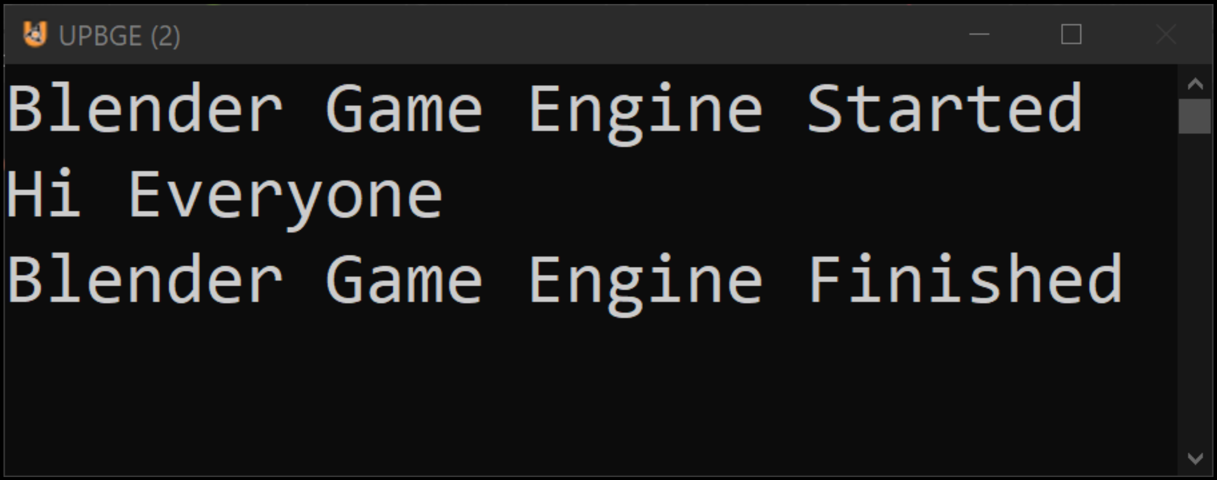
Connect our Python Script to our Cube

We add an: Always Actuator and Python Controller



We have set this **Always Actuator** to trigger only one time. We choose the script that will trigger by using the Script dropp down menu, of the **Python Controller.**

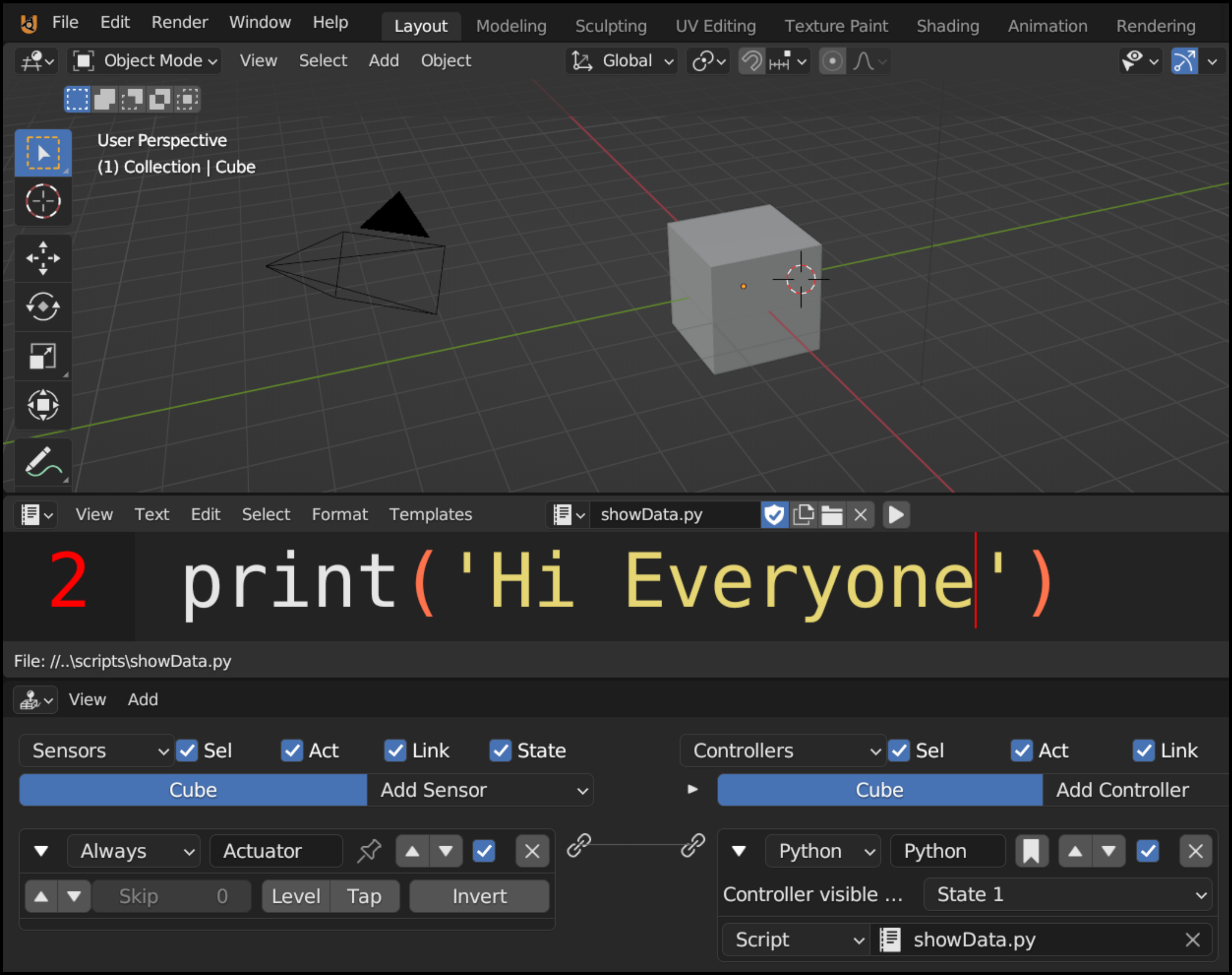
When we start the game engine it will state **Hi Everyone** in the Blender Console


UPBGE (2)

```
Blender Game Engine Started
Hi Everyone
Blender Game Engine Finished
```

How to Start the Blender Game Engine

We place our mouse arrow in: 3d Window



```
print('Hi Everyone')
```

File: //..\scripts\showData.py

Sensors ✓ Sel ✓ Act ✓ Link ✓ State
Cube | Add Sensor
Always | Actuator
Skip 0 | Level | Tap | Invert

Controllers ✓ Sel ✓ Act ✓ Link
Cube | Add Controller
Python | Python
Controller visible ... | State 1
Script | showData.py

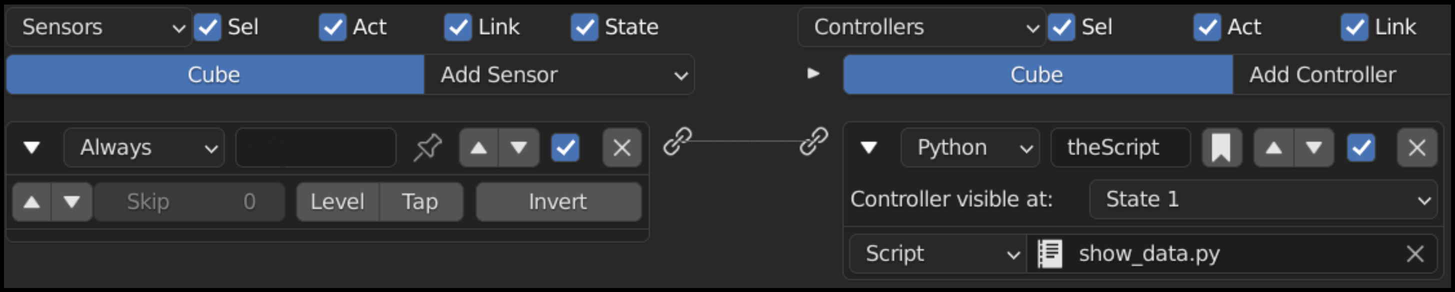
To Start the ENGINE, we press the letter: P

This begins the Blender Game Engine.

We make sure to have the Blender Console open to see the Hi Everyone Message and any other messages from our Game Making.

We End the ENGINE by pressing ESC

```
# show_object_name.py

# import blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the data of the object
print(obj)
```

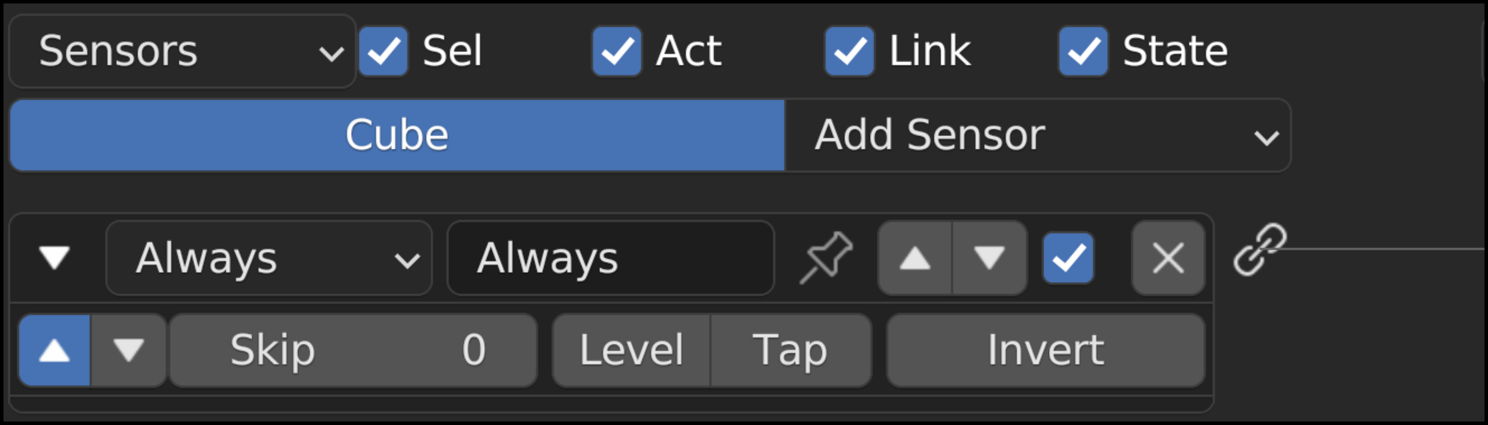


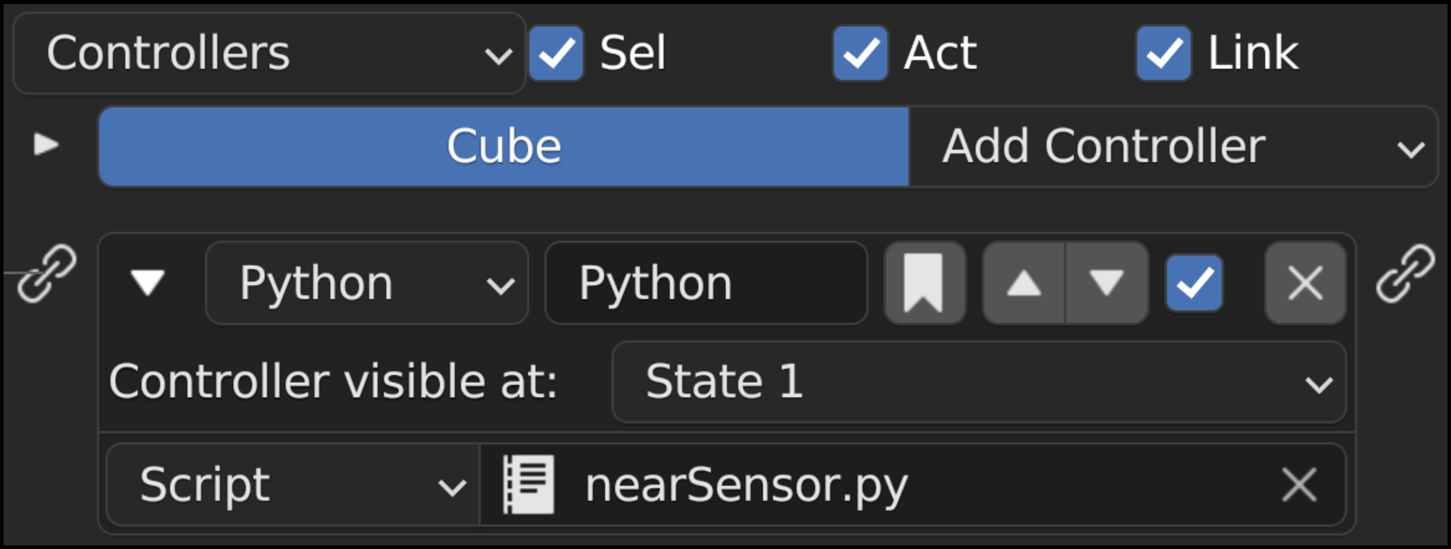
Controller visible at: State 1

Script show_data.py

```
# Result
# Cube
```

Zooming in, we see the Always Sensor





Controller visible at: State 1

Script nearSensor.py

```
# show_name.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the data of the object
print(obj.name)


#Result
#Cube
```

```
# show_name_with_label.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the data of the object
print("Name: ", obj.name)


#Result
#Name is: Cube
```

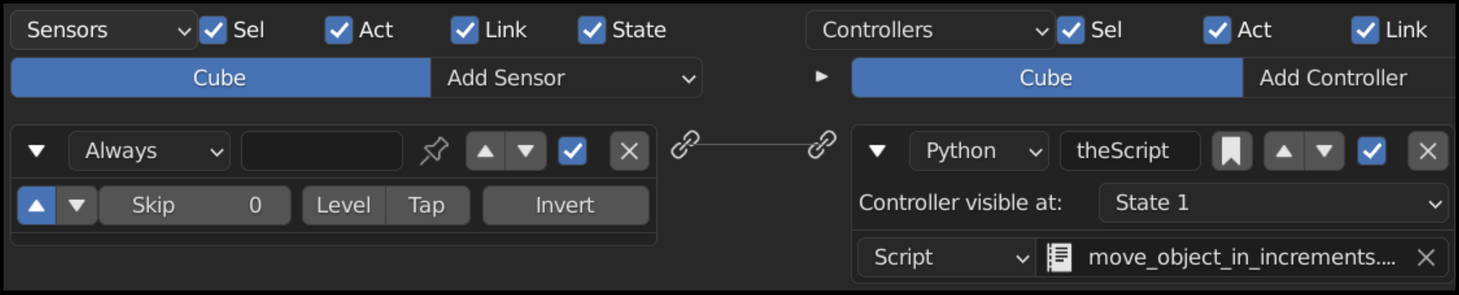
```
# get_position.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the location of the object
print(obj.position)


#Return
#<Vector (0.0000, 0.0000, 0.0000)>
```

```
# get_position_x.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the location of the object
print(obj.position.x)



# 0.0000
# get_position_y.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the location of the object
print(obj.position.y)



# 0.0000
# get_position_z.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the location of the object
print(obj.position.z)


# Result
# 0.0000
# get_position_x_y_z.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show the location of the object
print(obj.position.x)
print(obj.position.y)
print(obj.position.z)

'''
0.0000
0.0000
0.0000
'''
```

```
# move_object_to_position_x_y_z.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# move the object by xyz
obj.position.x = -3
obj.position.y = -3
obj.position.z = 3

print(obj.position)

'''
moves object to position of
<Vector (-3.0000, -3.0000, 3.0000)>
'''
```

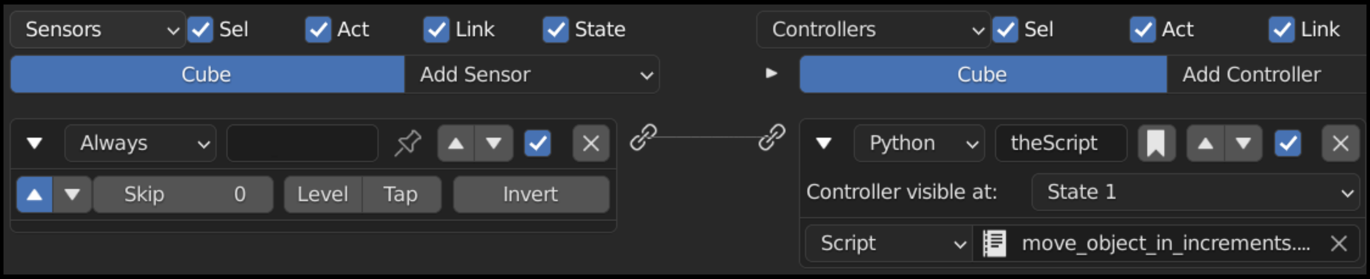
```
# move_object_to_position_vector_xyz.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# move the object to vector position
obj.position = (-3, -3, 3)

print(obj.position)

'''
moves object to position of
<Vector (-3.0000, -3.0000, 3.0000)>
'''
```

```
# move_object_in_increments.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# move the object by 0.1
obj.position.x += 0.1

# moves object 0.1 pixels in the x direction
```



```
# Always Sensor pulse is set to positive
```

The Always Sensor triggers repeatedly at a set Skip rate, or just one time, if no pulse set



Always

Skip 0

Python theScript

Controller visible at: State 1

Script move_object_in_increments....

Activates every TRUE pulse of the game engine. ▲ ▼

obj.position.x += 0.1

This means that every true pulse of the game engine, our Cube object will move on the x direction 0.1 meters.

If not selected, the always sensor will only activate one time upon start of the game.

▲ ▼

```
# get_sensor.py

import bge

def main():
    # get the current object
    obj = bge.logic.getCurrentController().owner

    # target object name for proximity check
(ensure it exists in the scene)
    target_name = "TargetObject"

    scene = bge.logic.getCurrentScene()

    target = scene.objects.get(target_name)

    # define a "near" range
    proximity_range = 25.0

    if target:
        # check if within range (similar to
KX_INPUT_ACTIVE for keys)
        is_near = obj.getDistanceTo(target) <=
proximity_range

        if is_near:
            # if within proximity, apply movement
(or any other action)
            obj.applyMovement((0.2, 0, 0), True)
            print("Target within range, moving
object.")

# run the main function
main()
```

```
# show_mouse_arrow.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show mouse arrow
bge.render.showMouse(True)
```

```
# show_mouse_arrow_at_position.py

# import the blender game engine
import bge

# get controller running this script
controller = bge.logic.getCurrentController()

# get game object controller is on
obj = controller.owner

# show mouse arrow
bge.render.showMouse(True)

# set mouse arrow position
bge.render.setMousePosition(100, 250)
```

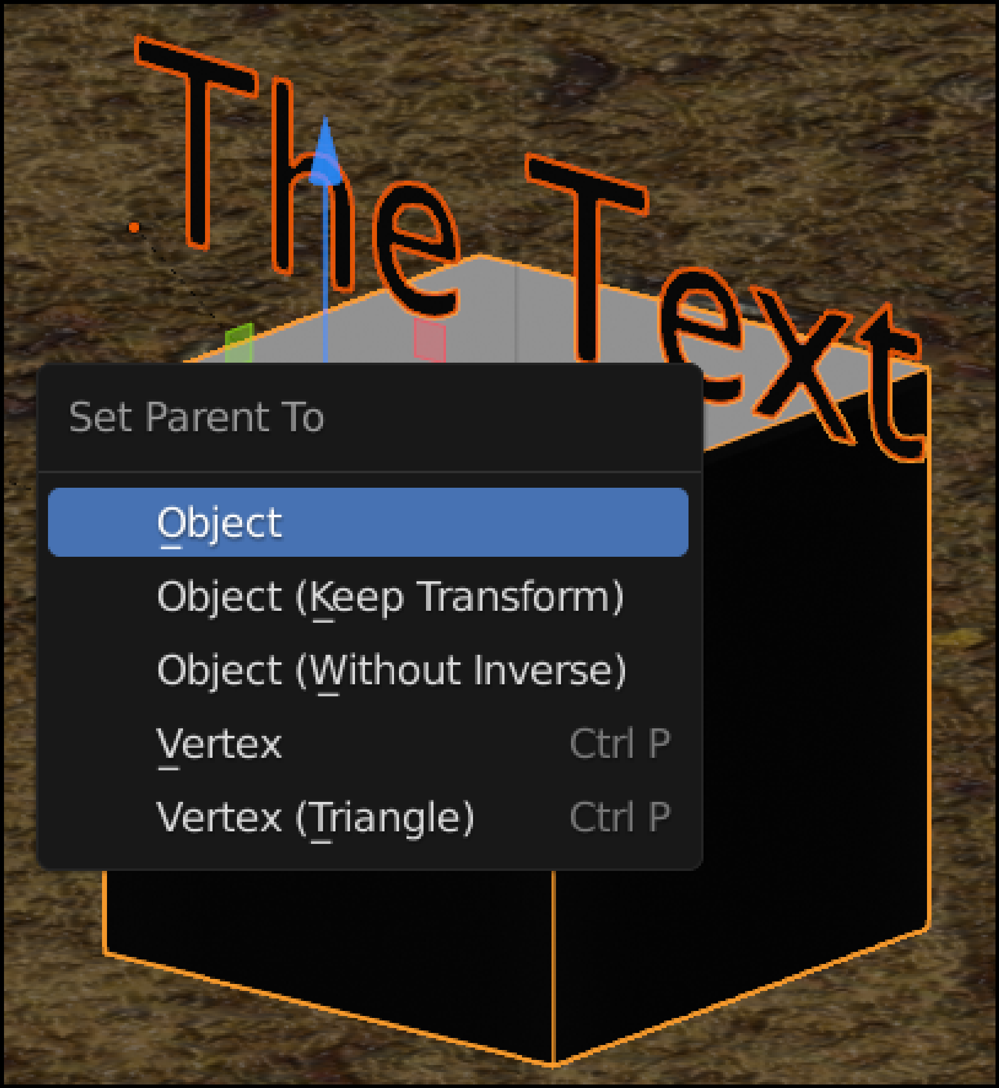
```
# world_position_rotation.py

import bge

controller = bge.logic.getCurrentController()

owner = controller.owner

owner.worldPosition.z += 0.1

# [x, y, z]
owner.applyRotation([0, 0, 0.1])
```

```
# text_change.py

import bge

controller = bge.logic.getCurrentController()

owner = controller.owner

owner['Text'] = 'Hi Everyone'
```

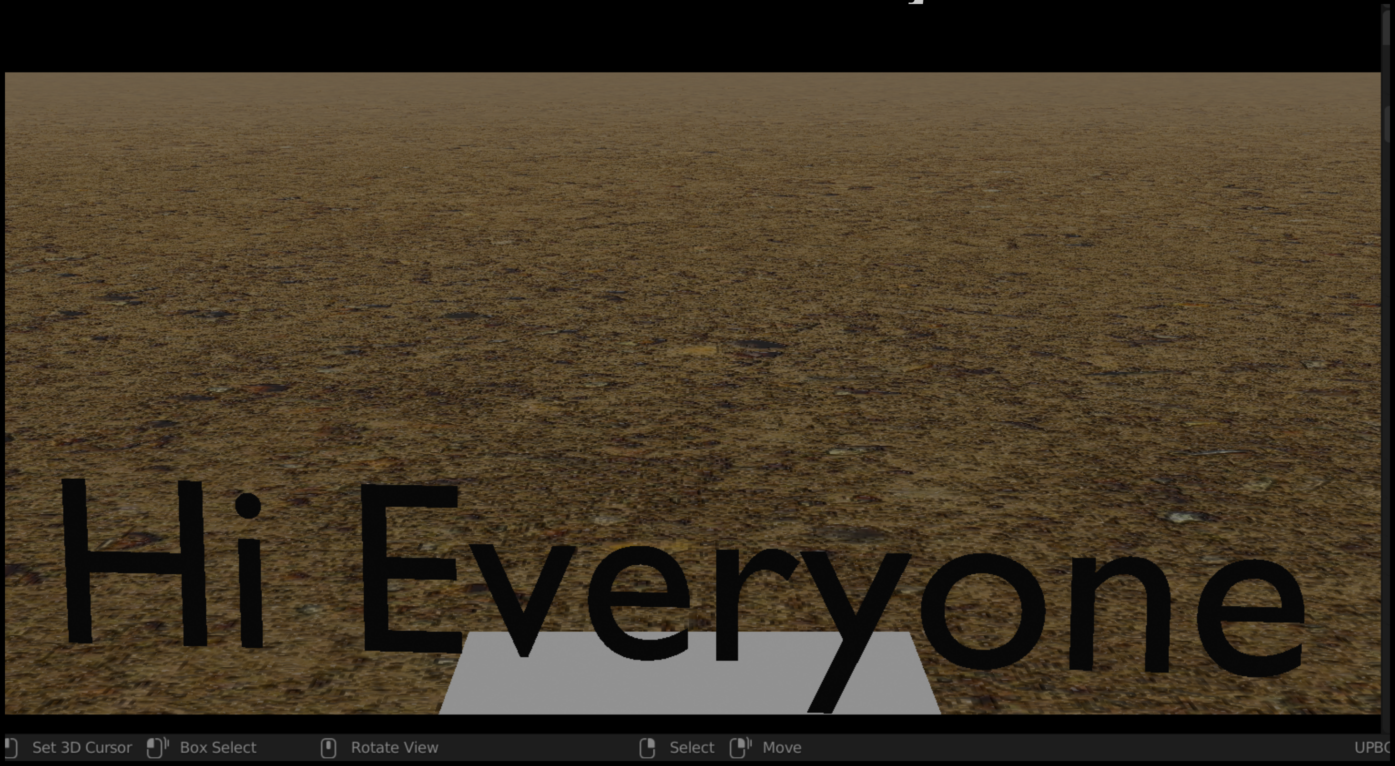
```
# text_change_other_object.py

import bge

theScene = bge.logic.getCurrentScene()

theScene.objects['theText']['Text'] = 'Howdy'
```

We connect the text object named theText to the object named ourPlayer
by Selecting first theText, holding shift and selecting ourPlayer, and then Holding Control + P to Set Parent To

-



In the Blender Game Engine mode, when we begin our game in camera mode, we can now see our text attached to our object.



In this example we simply made ourPlayer object be the parent of the text object. Now the text stays with ourPlayer.

```
# property_change.py

import bge

theScene = bge.logic.getCurrentScene()

# we reference the object named coin,
# and the property it has named counter
ourCounter = theScene.objects['coin']
['counter']

ourCounter = 5

theScene.objects['theText']['Text'] = ourCounter
```

The text **object** named theText will change to the value of the counter **property**.

Move Object Using Python Script

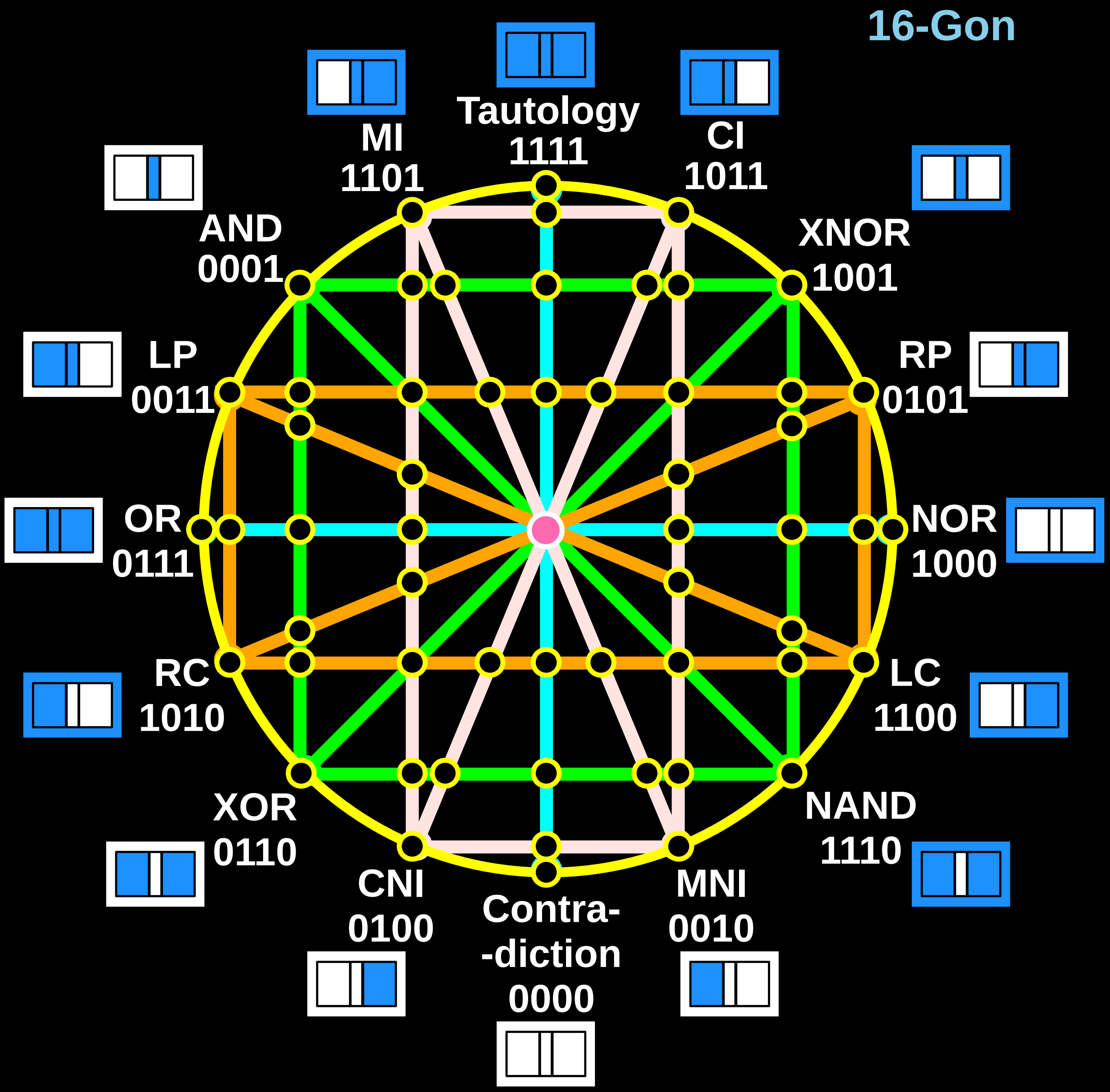
```
# Move Object Using Python Near Sensor
equivalent

import bge

def main():
    # get controller running this script
    controller = bge.logic.getCurrentController()

    # get game object this script is attached to
    obj = controller.owner

    # get all objects in scene
    scene = bge.logic.getCurrentScene()
    objects = scene.objects

    # check for proximity to another object
    target_object_name = "TargetObject"

    target_object =
objects.get(target_object_name)

    if target_object:
        # calculate distance between objects
        distance = (obj.worldPosition -
target_object.worldPosition).length

        # define threshold distance for triggering
movement
        threshold_distance = 10.0

        # is if target within threshold distance
        if distance < threshold_distance:
            obj.position.x += 0.2
        else:
            obj.position.z += 1.2

# run main function
main()
```

Player Motion Controls

```
# player_motion_controls.py

import bge

def main():
    cont = bge.logic.getCurrentController()

    own = cont.owner

    keyboard = bge.logic.keyboard

    wKey = bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.WKEY]

    sKey = bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.SKEY]

    aKey = bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.AKEY]

    dKey = bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.DKEY]

    if wKey:
        own.applyMovement([0,0.2,0], True)
    if sKey:
        own.applyMovement([0,-0.2,0], True)
    if aKey:
        own.applyMovement([-0.2,0,0], True)
    if dKey:
        own.applyMovement([0.2,0,0], True)

main()
```

Christopher Topalian

True Artificial Intelligence System



For More Tutorials:

CollegeOfScripting.weebly.com

CollegeOfScripting.wordpress.com

GitHub.com/ChristopherTopalian

GitHub.com/ChristopherAndrewTopalian

Youtube.com/ScriptingCollege

Twitter.com/CollegeOfScript

Rumble.com/user/CollegeOfScripting

Sites.google.com/view/CollegeOfScripting

Dedicated to God the Father

This book is created by the
College of Scripting Music & Science.

Always remember, that each time you write a script with a pencil and paper, it becomes imprinted so deeply in memory that the material and methods are learned extremely well. When you Type the scripts, the same is true.

The more you type and write out the scripts by keyboard or pencil and paper, the more you will learn programming!

Write & Type EVERY example that you find.
Keep all of your scripts organized.
Every script that you create increases your programming abilities.

SEEING CODE, is one thing,
but WRITING CODE is another.
Write it, Type it, Speak it, See it, Dream it.

www.CollegeOfScripting.weebly.com