

2020

BGE + PYTHON

```
177     ),
178     default='v',
179 )
180 global_scale_setting = FloatProperty(
181     name="Scale",
182     min=0.01, max=1000.0,
183     default=1.0,
184 )
185
186 def execute(self, context):
```

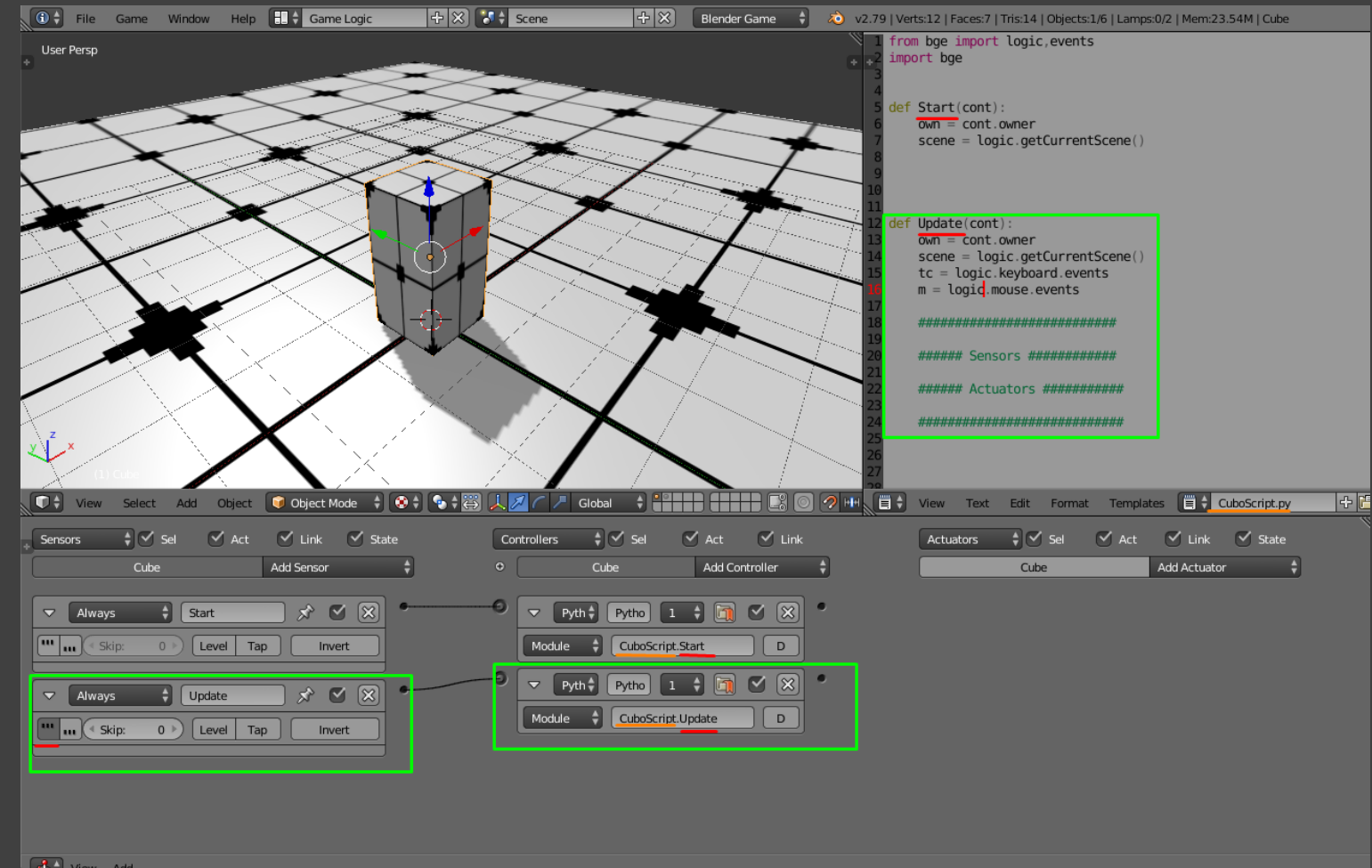
```
90 # get objects selected in the viewport
91 viewport_selection = bpy.context.selected_objects
92
93 # get export objects
94 obj_export_list = viewport_selection
95 if self.use_selection_setting == False:
96     obj_export_list = [i for i in bpy.context.scene.objects]
97
98 # deselect all objects
99 bpy.ops.object.select_all(action='DESELECT')
100
101 for item in obj_export_list:
102     item.select = True
103     if item.type == 'MESH':
104         file_path = os.path.join(folder_path, "{}.obj".format(item.name))
105         bpy.ops.export_scene.obj(filepath=file_path, use_selection=True,
106                                 axis_forward=self.axis_forward_setting,
107                                 axis_up=self.axis_up_setting,
108                                 use_animation=self.use_animation_setting,
109                                 use_mesh_modifiers=self.use_mesh_modifiers_setting,
110                                 use_edges=self.use_edges_setting,
111                                 use_smooth_groups=self.use_smooth_groups_setting,
112                                 use_smooth_groups_bitflags=self.use_smooth_groups_bitflags_setting,
113                                 use_normals=self.use_normals_setting,
114                                 use_uv=self.use_uv_setting,
115                                 use_material=self.use_material_setting)
```

Edinaldo Cicero
ÁtomoGames.
28/07/2020

BGE + PYTHON

Começando com Funções para Modulos :

Ao criar uma Função Start ou Update você há usa como Modulo no objeto que deseja usar no script ou seja o dono do script o próprio objeto.



Para chamar o script pelo Modulo vc pega o nome do (script . Nome da Função) como visto acima no controlador python .

Lá na função Update(cont) esse (cont) corresponde á (controller) para quer serve ,para chamar o logic bricks de controlador do objeto ou seja o que está em verde com isso vc tem acesso ao sensores e actuators do objetos que estiverem ligados nesse controlador .

Own ou (owner = cont.owner) aqui corresponde ao próprio objeto proprietário do script com isso em mão vc pode manipular como quiser o próprio objeto exemplo:

Para movimentar o objeto own:

`owner.applyMovement([x,y,z],True)`

veja como eu aplico o movimento para o dono do próprio script (owner.) depois de colocar o ponto vc deve chamar um função da biblioteca,da class KX_GameObjects que pertence á todos os objetos do blender com isso tem varias funções para todos os objetos.

KX_GameObject (SCA_IObject) ¶

classe base - SCA_IObject

classe bge.types.KX_GameObject(SCA_IObject)

Todos os objetos do jogo são derivados dessa classe.

As propriedades atribuídas aos objetos do jogo são acessíveis como atributos desta classe.

Nota: Chamar QUALQUER método ou atributo em um objeto que foi removido de uma cena gerará um SystemError, se um objeto puder ter sido removido desde o último acesso a ele, use o `invalid` atributo para verificar.

https://docs.blender.org/api/2.79/bge.types.KX_GameObject.html#bge.types.KX_GameObject

getAxisVect(vect)

Retorna o vetor do eixo que gira pela orientação do espaço no mundo do objeto. Isso é o equivalente a multiplicar o vetor pela matriz de orientação.

Parâmetros: **vect** (*vetor 3D*) - um vetor para alinhar o eixo.
Devoluções: O vetor em relação à rotação dos objetos.
Tipo de retorno: Vetor 3d

applyMovement(movimento , local = Falso)

Define o movimento do objeto do jogo.

Parâmetros:

- **movimento** (*vetor 3D*) - vetor de movimento.
- **local** -
 - Falso: você obtém o movimento "global", isto é, relativo à orientação mundial.
 - Verdadeiro: você obtém o movimento "local", isto é, relativo à orientação do objeto.
- **local** - booleano

applyRotation(rotação , local = Falso)

Define a rotação do objeto do jogo.

Parâmetros:

- **rotação** (*vetor 3D*) - vetor de rotação.
- **local** -
 - Falso: você obtém a rotação "global", isto é, relativa à orientação mundial.
 - Verdadeiro: você obtém a rotação "local", isto é, relativa à orientação do objeto.
- **local** - booleano

Tento isso em mente vc poderá perceber que o que estiver detro da classe Kx_GameObjects poderá ser utilizado após o (owner.) .

Uma coisa importante,se lembre que para cada função que for feita para um objeto e for chamada em Modulo no objeto que desejás manipular o (owner) se passa a ser o próprio objeto certo,ou seja , cada função poderá ser usada em cada objeto porém variáveis que estiverem dentro do escopo de uma função não poderá ser acessada dentro de outra !.

Como Movimentar e Rotacionar um obj e o que é Vetor2 E 3 :

Para movimentar ou Rotacionar um obj no universo do blender existe duas classes Vetoriais :

1 = owner.applyMovent([vetor3],Boo)

owner.applyMovent([X,Y,Z],Boo)

Para rotacionar:

2 = owner.applyRotation([vetor3],Bool)

owner.applyRotation([X,Y,Z] ,Bool)

Vetor2 ou Vetor3 nada mais é na realidade um tipo de variavel que guarda posições geograficas X,Y,Z se for vector2 seria X,Y

Com isso `own.applyMovement([x,y,z],True)` no lugar dos `([X,Y,Z])` coloque o valor que desejás que o obj ande ou rotacione , tenha em mente que se vc colocar (- ou +) na frente do valor desejado isso fará mudar de direção pois isso está baseado nos paramentos geométricos do mundo 3D ou seja positivo para frente,negativo para trás e os valores tende á ser do tipo (Float) números quebrados ou numero seguidos de virgulas certo ,pode sim usar números do tipo (Int) mais saiba que números interiros são números exatos e corresponde a mais espaços para o objeto se deslocar pois o obj não andar realmente claro! , mais sim se deslocar de acordo com o valor digitado , lembrou física muito bem !.

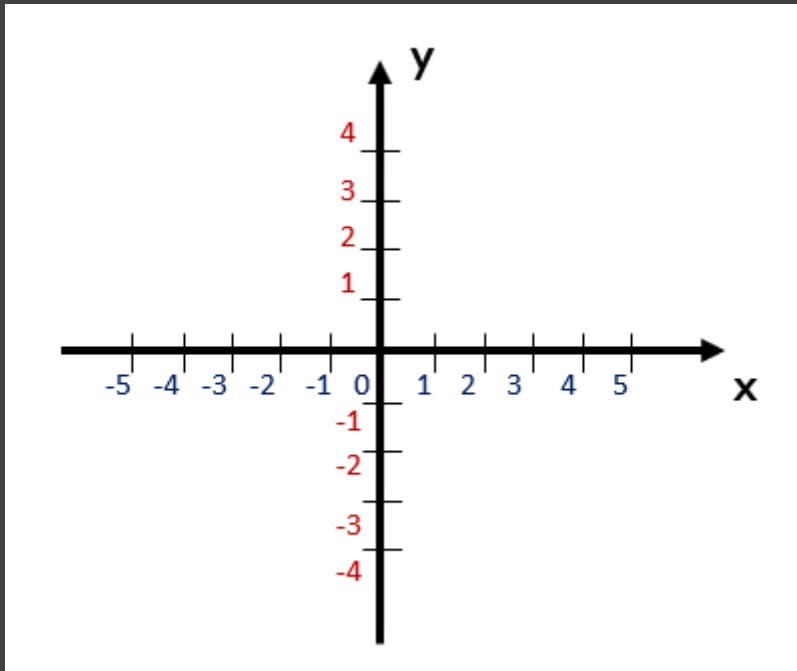
X = Largura

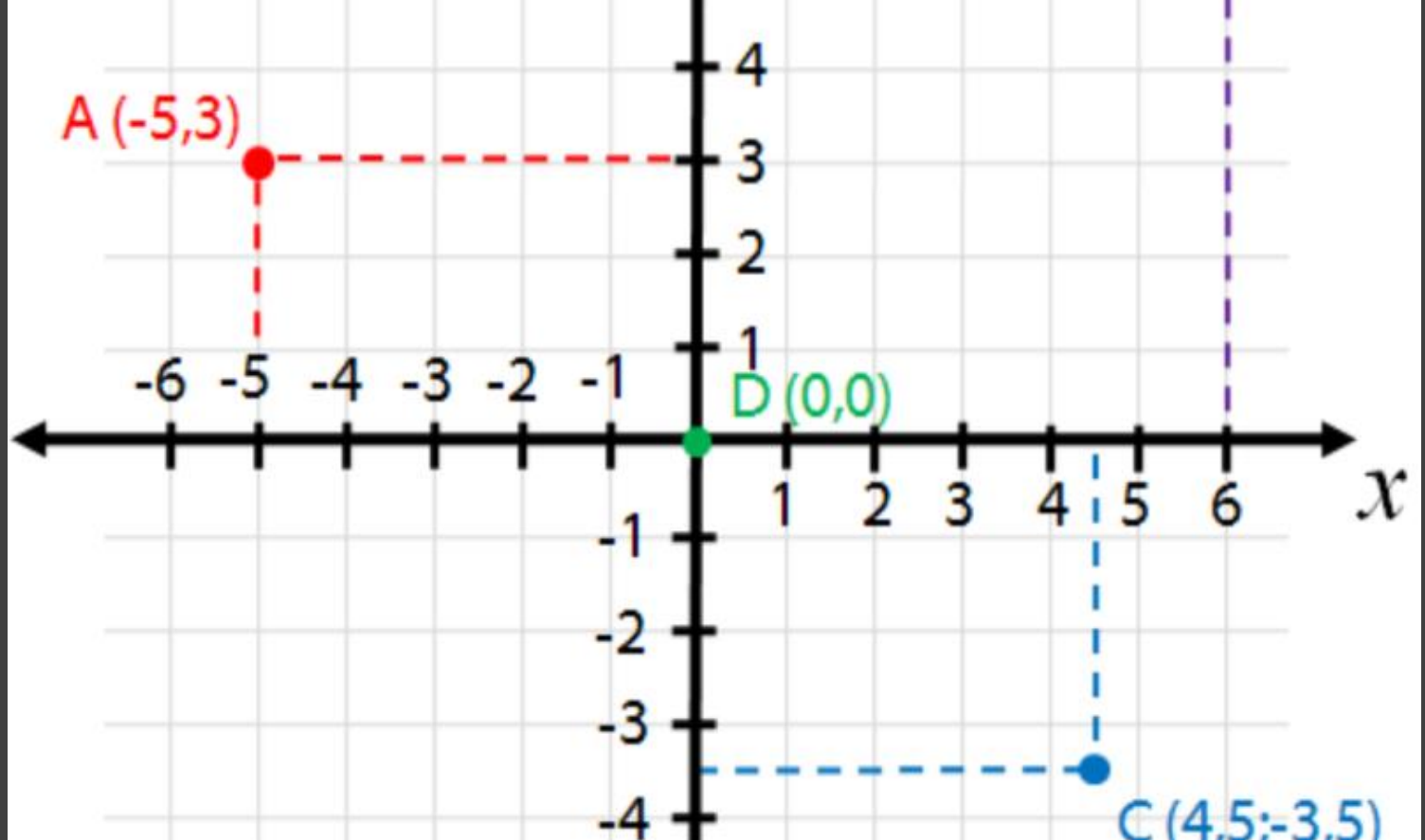
Y = Altura

Z = Profundidade

Vector2 pra Games 2D

Vector3 pra Games 3D





Como como usar Inputs de Teclado e Mouse:

Para acessar teclas do teclado se deve chamar Chaves de Constantes ,para isso chame todas as teclas do teclado :

Antes de chamar certifique-se de ter importado as bibliotecas de Logicas e eventos do blender assim :

```
From bge import logic, events
```

Feito isso agora crie uma variavel com o nome que quiser mais que se torne fácil de lembrar claro e faça desse jeito :

```
Teclado = logic.keyboard.events
```

Com isso pronto vamos para usar cada tecla ,e para isso vamos para a API com todas as teclas para usar :<https://docs.blender.org/api/2.79/bge.events.html>

E para usar a tecla escolhida basta criar uma condição de Input :

```

def Update(cont):
    own = cont.owner
    scene = logic.getCurrentScene()
    teclado = logic.keyboard.events
    m = logic.mouse.events
    #####
    ##### Sensors #####
    ##### Actuators #####
    #####

    if teclado[events.WKEY]:
        print("Tecla W precionada !!")

```

If teclado[biblioteca de eventos do BGE.Tecla escolhida terminando com KEY no final]:

Com isso temos uma condição que quando a tecla W for apertada após os dois pontos vc aperta enter e entrará dentro do escopo da condição que será abaixo com isso vc poderá fazer ações com o obj colocando até o próprio own.applyMovement([x,y,z],True) e no lugar dos ([X,Y,Z]) coloque o valor que deseja que o obj ande ou rotacione !..