Package **MiniGameEngine**

Classes

class GameObject (x: int, y: int, imagePath: str, tipo: str = 'undef', collisions: bool = False, layer: int = 1)

Constructor de la clase GameObject que inicializa un objeto en el mundo de juego.

Args

x : `int`
Coordenada x inicial del objeto.

y : `int`
Coordenada y inicial del objeto.

imagePath : `str`
Ruta de la imagen del objeto.

tipo : `str` , optional
Tipo del objeto (por defecto es "undef").

collisions : `bool` , optional
True si este objeto participara de las colisiones (por defecto es False)

layer : `int` , optional
capa en que se colocara este objeto (por defecto es 1)

Methods

def collidesWith(self, obj) -> bool

Determina si este GameObject colisiona con otro

Args

obj : `GameObject`
GameObject a detectar si colision con este

Returns

`bool`
True si colisiona. False en caso contrario

def destroy(self)

Elimina el objeto del mundo de juego.

def getHeight(self) -> int

Obtiene la altura del objeto.

Returns

`int`
: Altura del objeto.

def getTipo(self) -> str

Obtiene el tipo del objeto.

Returns

`str`
: Tipo del objeto.

def getWidth(self) -> int

Obtiene el ancho del objeto.

Returns

`int`
: Ancho del objeto.

def getWorldHeight(self) -> int

Obtiene la altura del mundo de juego.

Returns

`int`
: Altura del mundo de juego.

def getWorldWidth(self) -> int

Obtiene el ancho del mundo de juego.

Returns

`int`
: Ancho del mundo de juego.

def getX(self) -> int

Obtiene la coordenada x actual del objeto.

Returns

`int`
Coordenada x del objeto.

def getY(self) -> int

Obtiene la coordenada y actual del objeto.

Returns

`int`
Coordenada y del objeto.

def isPressed(self, key_name: str) -> bool

Verifica si una tecla específica está siendo presionada.

Args

key_name : `str`
Nombre de la tecla a verificar.

Returns

`bool`
True si la tecla está siendo presionada, False en caso contrario.

def loadImage(self, imagePath: str) -> tkinter.PhotoImage

Carga la imagen que se encuentra en la ruta especificada

Args

imagePath : `str`
Ruta de la imagen a cargar.

Returns

`binary`
La imagen a cargar.

def loadImages(self, imagesPaths: list) -> list

Carga las imagenes referenciadas por el arreglo de paths

Args

imagesPaths : `list`
Arreglo de imagenes cargadas

def onCollision(self, dt: float, gobj)

Llamado cuando el objeto colisiona con otro objeto.

Args

dt : `float`
Tiempo en segundos desde la ultima llamada.

gobj : `GameObject`
Objeto con el que colisiona.

def onUpdate(self, dt: float)

Llamado en cada actualización del juego para el objeto.

Args

dt : `float`
Tiempo en segundos desde la ultima llamada.

def setBgPic(self, bgPath: str)

Cambia la imagen de fondo

Args

bgPath : `str`
Ruta de la imagen a utilizar como fondo

def setCollisions(self, collisions: bool)

Habilita o deshabilita participar del procesamiento de colisiones

Args

collisions : `bool`
True para habilitar, False para deshabilitar

def setPosition(self, x: int, y: int)

Establece la posición del objeto en el mundo de juego.

Args

x : int
Nueva coordenada x del objeto.

y : int
Nueva coordenada y del objeto.

def setShape(self, imagePath: str)

Cambia la forma del objeto reemplazando su imagen.

Args

imagePath : str
Ruta de la nueva imagen del objeto.

class GameWorld (width: int, height: int, title: str = 'MiniGameEngine', bgColor: str = 'gray', bgPath: str = None, numLayers: int = 10)

Constructor de la clase GameWorld que inicializa una instancia del mundo de juego.

Args

width : int
Ancho de la ventana del juego.

height : int
Altura de la ventana del juego.

title : str , optional
Título de la ventana del juego (por defecto es "MiniGameEngine").

bgColor : str , optional
Color de fondo de la ventana del juego (por defecto es "gray").

bgPath : str , optional
Ruta de la imagen de fondo de la ventana del juego (por defecto es None).

numLayers : int , optional
Numero de capas a permitir en el juego (por defecto es 10).

Methods

def collide(self, o1, o2) -> bool

Detecta si dos GameObjects colisionan entre si

Args

o1 : `GameObject`
El GameObject a verificar si colisiona con o2

o2 : `GameObject`
El GameObject a verificar si colisiona con o1

Returns

`bool`
True si colisionan. False en caso contrario.

def exitGame(self)

Finaliza el loop principal del juego

def gameLoop(self, fps: int)

Inicia el loop principal del juego.

Args

fps : `int`
Número de cuadros por segundo del juego.

def getWorldHeight(self) -> int

Obtiene la altura del mundo de juego.

Returns

`int`
Altura del mundo de juego.

def getWorldWidth(self) -> int

Obtiene el ancho del mundo de juego.

Returns

`int`
Ancho del mundo de juego.

def isPressed(self, key_name: str) -> bool

Verifica si una tecla específica está siendo presionada.

Args

key_name : str
Nombre de la tecla a verificar.

Returns

bool
True si la tecla está presionada, False en caso contrario.

def loadImage(self, imagePath: str) -> tkinter.PhotoImage

Carga la imagen que se encuentra en la ruta especificada

Args

imagePath : str
Ruta de la imagen a cargar.

Returns

binary
La imagen cargada.

def loadImages(self, imagesPaths: list) -> list

Carga las imagenes referenciadas por el arreglo de rutas

Args

imagesPaths : list
Arreglo de rutas a las imagenes a cargar.

Returns

list
Arreglo con las imágenes cargadas.

def onUpdate(self, dt: float)

Llamada por cada ciclo dentro del loop (fps veces por segundo)

Args

dt : float
Tiempo en segundos desde la última llamada

```
def setBgPic(self, bgPath: str)
```

Cambia la imagen de fondo

Args

bgPath : `str`
Ruta a la imagen a utilizar como fondo

```
class TextObject (x: int, y: int, text: str, font: str = 'Arial', size: int = 10, bold: bool = False,
                  italic: bool = False, color: str = 'black')
```

Constructor de la clase TextObject que agrega un Texto al mundo del juego

Args

x : `int`
Coordenada x del texto

y : `int`
Coordenada y del texto

text : `str`
Texto para este objeto

font : `str` , optional
Font a utilizar para el texto (por defecto es "Arial").

size : `int` , optional
Tamano a utilizar para el texto (por defecto es 10).

bold : `bool` , optional
Especifica que el texto estara en bold (por defecto es False).

italic : `bool` , optional
Especifica que el texto estara en italic (por defecto es False).

color : `str` , optional
Color a utilizar para el texto (por defecto es "black").

Methods

```
def destroy(self)
```

Elimina este texto del mundo del juego

```
def setText(self, x: int = None, y: int = None, text: str = None, font: str = None, size: int = None,
            bold: bool = None, italic: bool = None, color: str = None)
```

Modifica el texto desplegado y sus atributos. Si no se especifican atributos se convservan los existentes

Args

x : `int` , optional
    Coordenada x del texto.

y : `int` , optional
    Coordenada y del texto

text : `str` , optional
    Texto para este objeto

font : `str` , optional
    Font a utilizar para el texto

size : `int` , optional
    Tamano a utilizar para el texto

bold : `bool` , optional
    Especifica que el texto estara en bold

italic : `bool` , optional
    Especifica que el texto estara en italic

color : `str` , optional
    Color a utilizar para el texto

Index

Classes