Curso Pygame Avanzado

ING informática

Librerías a Utilizar

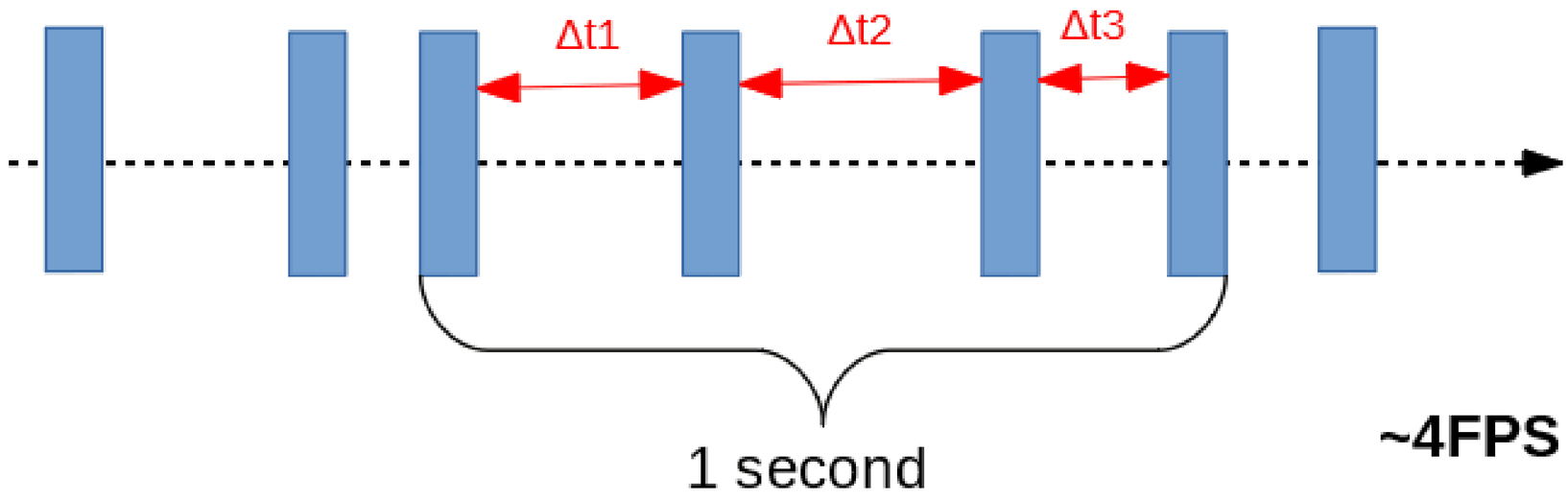
- *Pygame*
- *Pygame_gui*
- *Numpy*

UIManager

- manager = pygame_gui.UIManager((largo, alto))
- manager.process_events(event)
- manager.update(time_delta)
- manager.draw_ui(window_surface)

Delta Time

- clock = pygame.time.Clock()
- time_delta = clock.tick(60)/1000.0



Componentes

- **relative_rect**=py.Rect((0, 0), (100, 200)),
- **manager**=uiManager
- **anchors**={'centerx': 'centerx', 'top': 'top'},
- **container**=None,
- **parent_element**=None,
- **object_id**='#title',
- **visible**=True,

Componentes

uiManager.**set_visual_debug_mode**(True)

Componentes

tituloLabel**.kill()**

Label

```
tituloLabel = UILabel(
        relative_rect=py.Rect((0, 0), (100, 200)),
        manager=uiManager
        text='Página de Inicio',
)
```

Label

Button

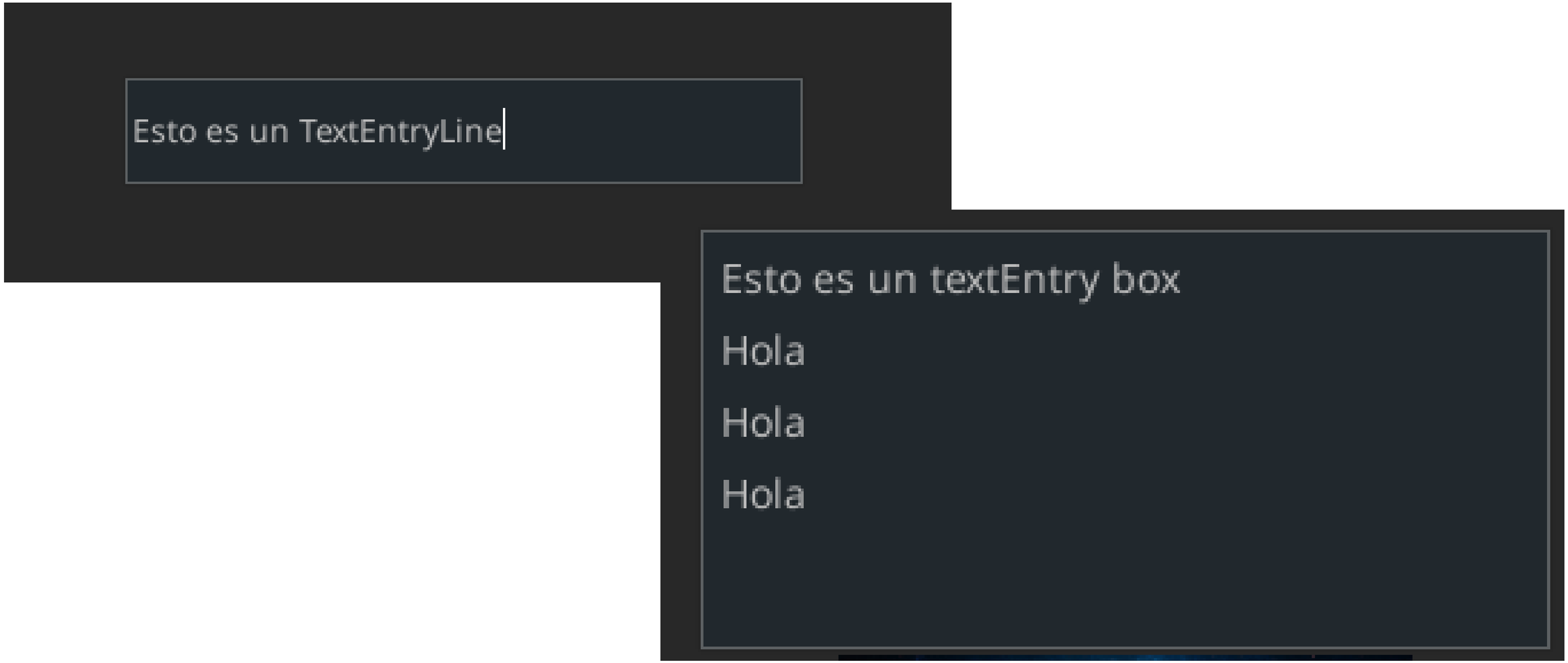
```
button = UIButton(
        relative_rect=py.Rect((0, 100), (400, 100)),
        command=lambda: print("ejecuta funcion"),
        text='Titulo de Boton',
        manager=uiManager
)
```

Button

TextEntryLine/Box

```
textEntryLine = UITextEntryLine(
        initial_text='Escribe algo aquí...',
        placeholder_text='Escribe algo...',
        relative_rect=py.Rect((0, 220), (300, 40)),
        manager=uiManager,
    )
```

TextEntryLine/Box



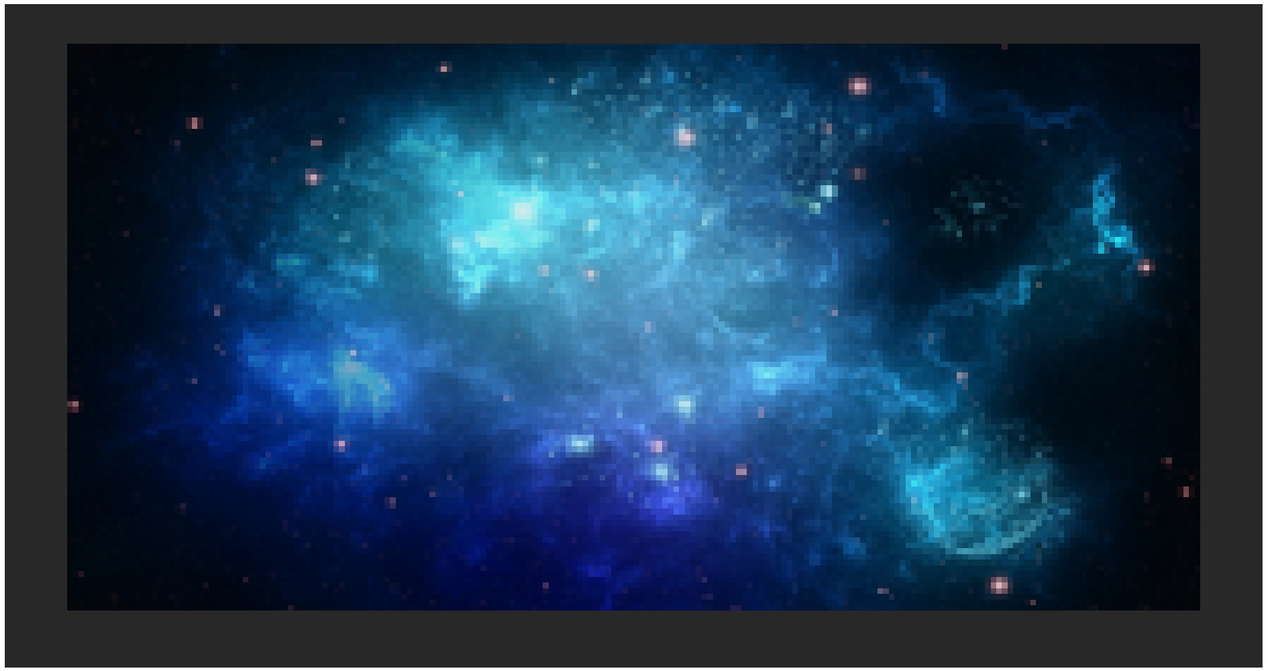
TextEntryLine/Box

textEntryLine.**get_text()**

Imagenes

```
image = UIImage(
        relative_rect=py.Rect((0, 400), (200, 100)),
        image_surface=py.image.load("url"),
        manager=uiManager,
    )
```

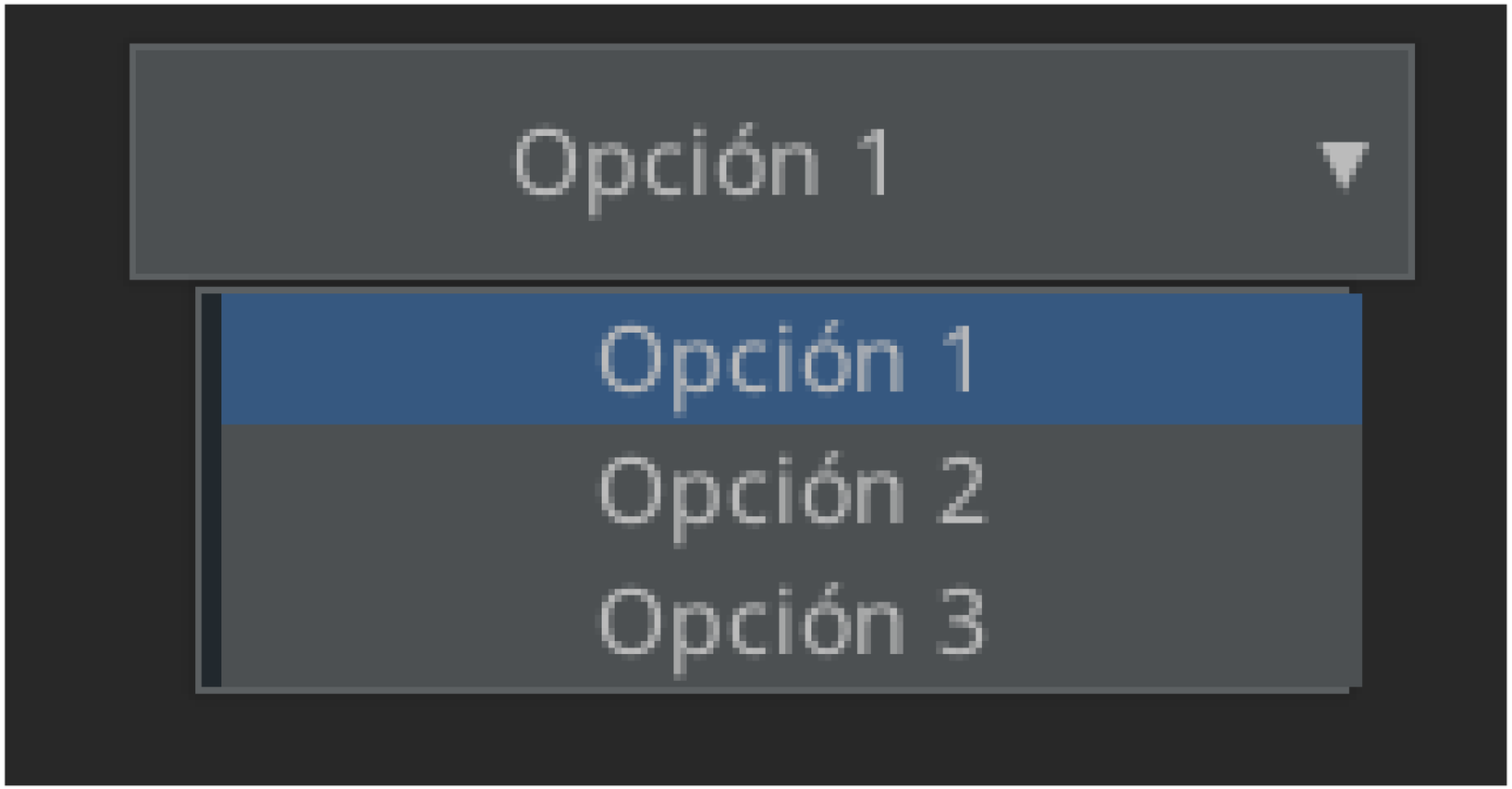
Imagenes



DropDownMenu

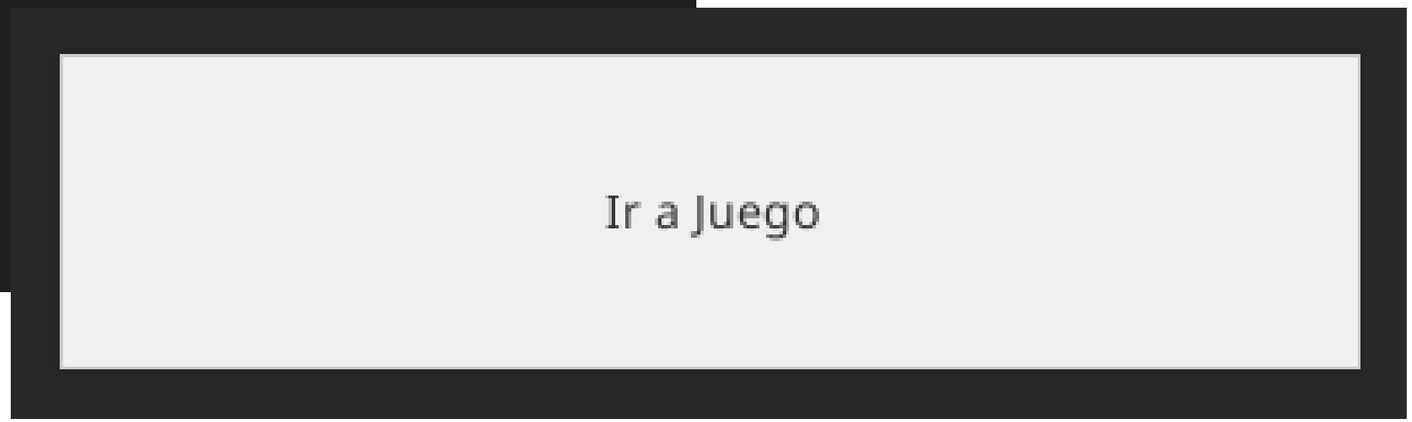
```
dropdown_menu = UIDropDownMenu(
    options_list=['Opción 1', 'Opción 2', 'Opción 3'],
    starting_option='Opción 1',
    relative_rect=py.Rect((0, 250), (200, 40)),
    manager=uiManager,
)
```

DropDownMenu



Temas

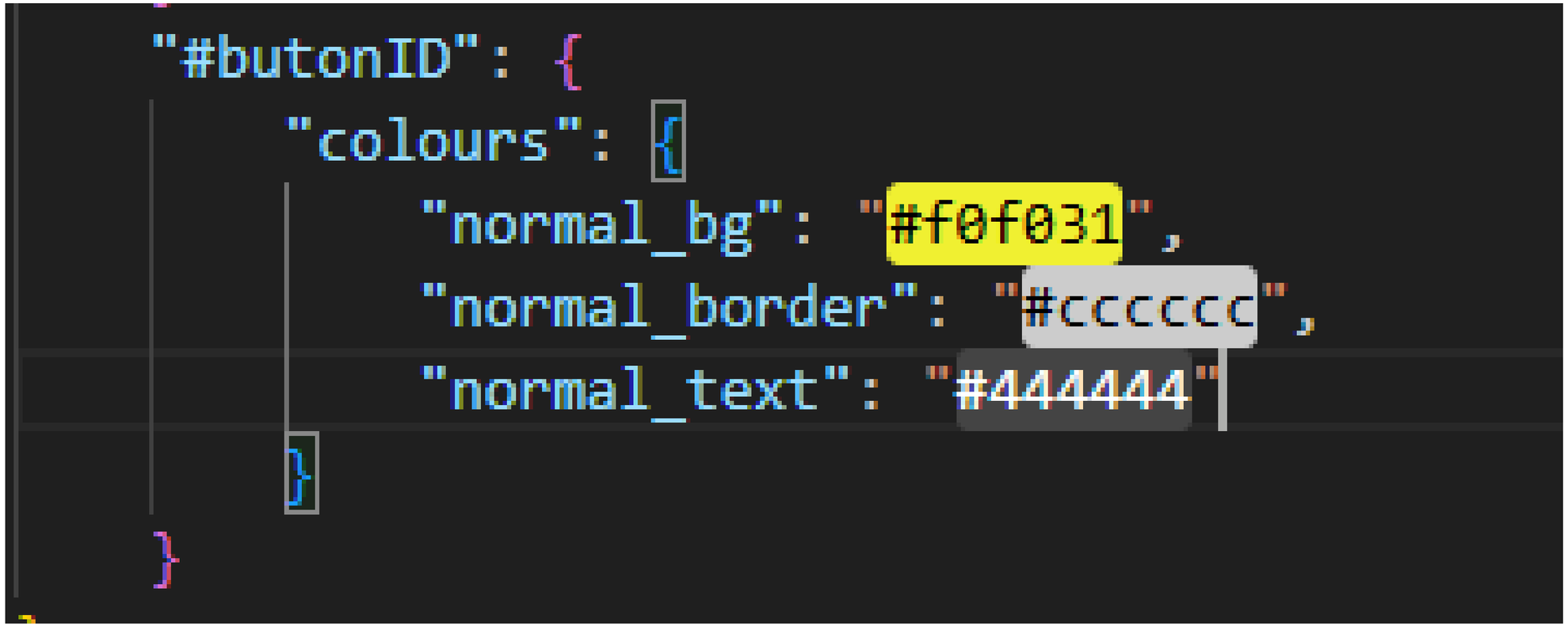
```
"button": {
    "colours": {
        "normal_bg": "#f0f0f0",
        "normal_border": "#cccccc",
        "normal_text": "#333333"
    }
}
```



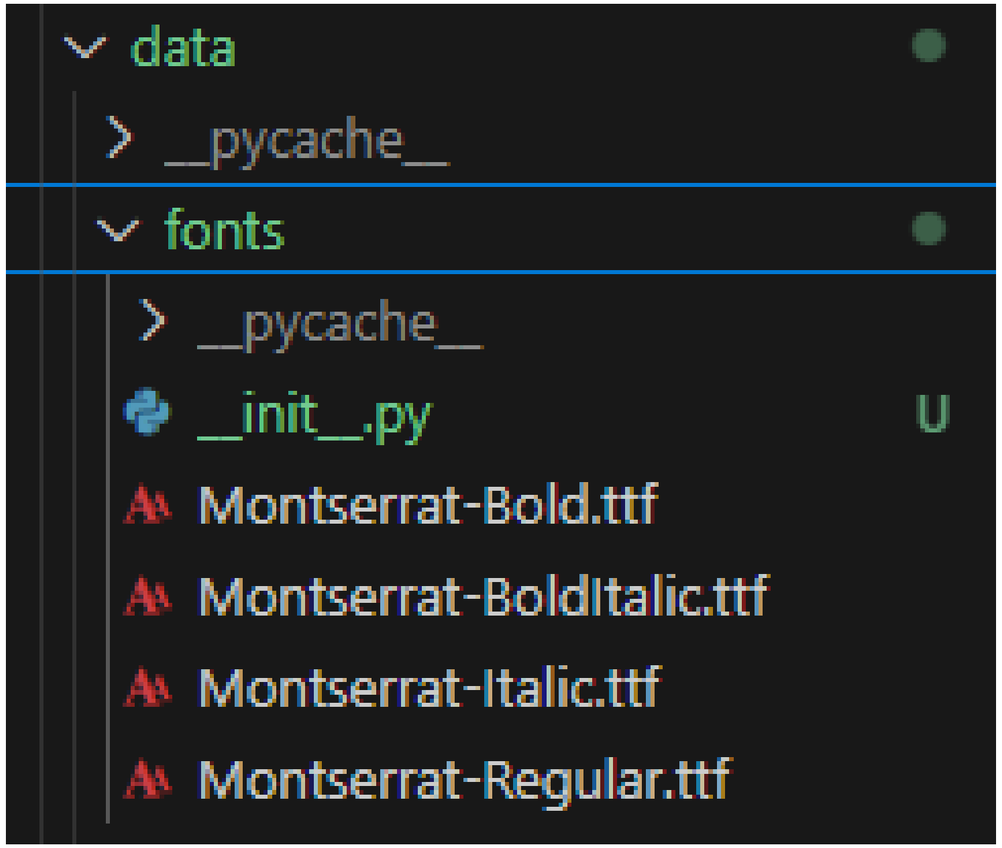
Object_id

Object_id = ObjectID(
class_id='@clase1',
object_id='#id1'
)

Object_id


"#butonID": {
"colours": {
"normal_bg": "#f0f031",
"normal_border": "#cccccc",
"normal_text": "#444444"
}
}

Fuentes




```
"button": {
    "font": {
        "name": "montserrat",
        "size": "12",
        "bold": "0",
        "italic": "0",
        "regular_resource": {
            "package": "data.fonts",
            "resource": "Montserrat-Regular.ttf"
        }
    },
    "colours": {
        "normal_bg": "#f0f0f0",
        "normal_border": "#cccccc",
        "normal_text": "#333333"
    }
}
```

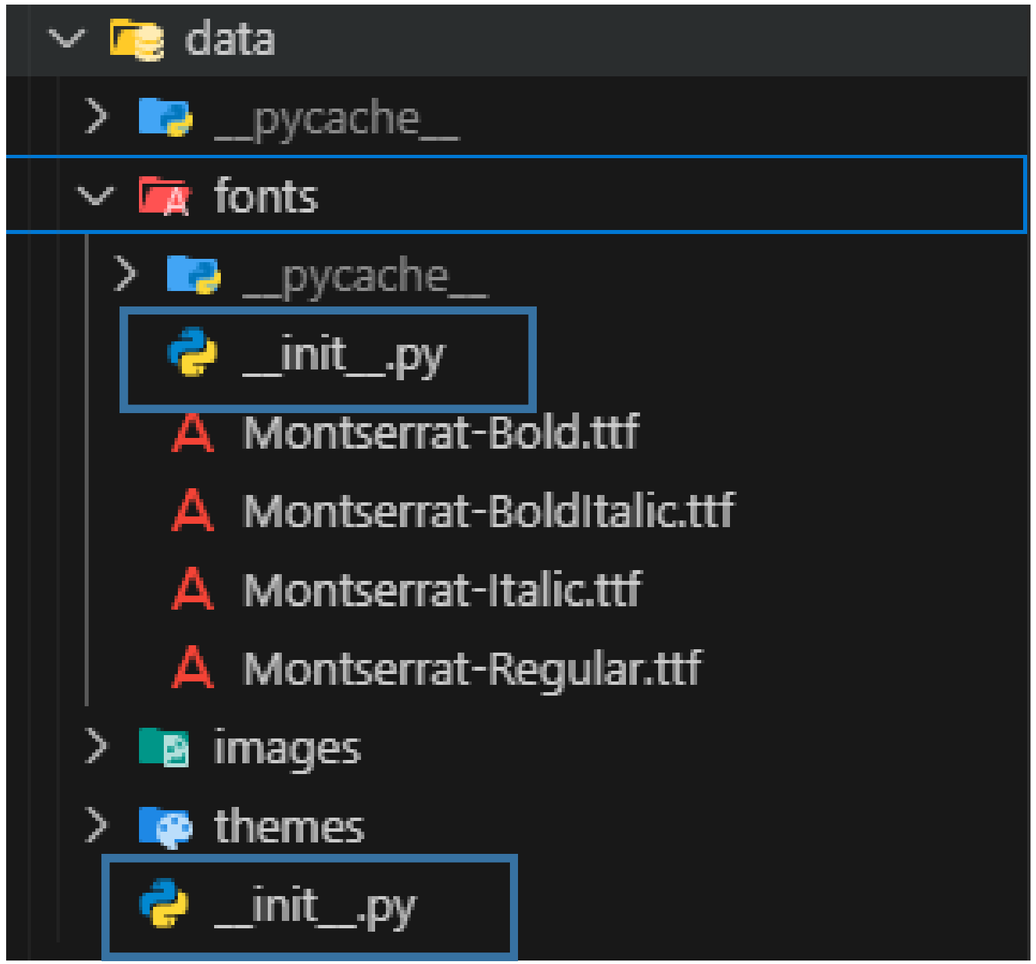
Fuentes

```
"font": {
        "name": "montserrat",
        "size": "12",
        "bold": "0",
        "italic": "0",
        "regular_resource": {
            "package": "data.fonts",
            "resource": "Montserrat-Regular.ttf"
        }
    },
```

Fuentes

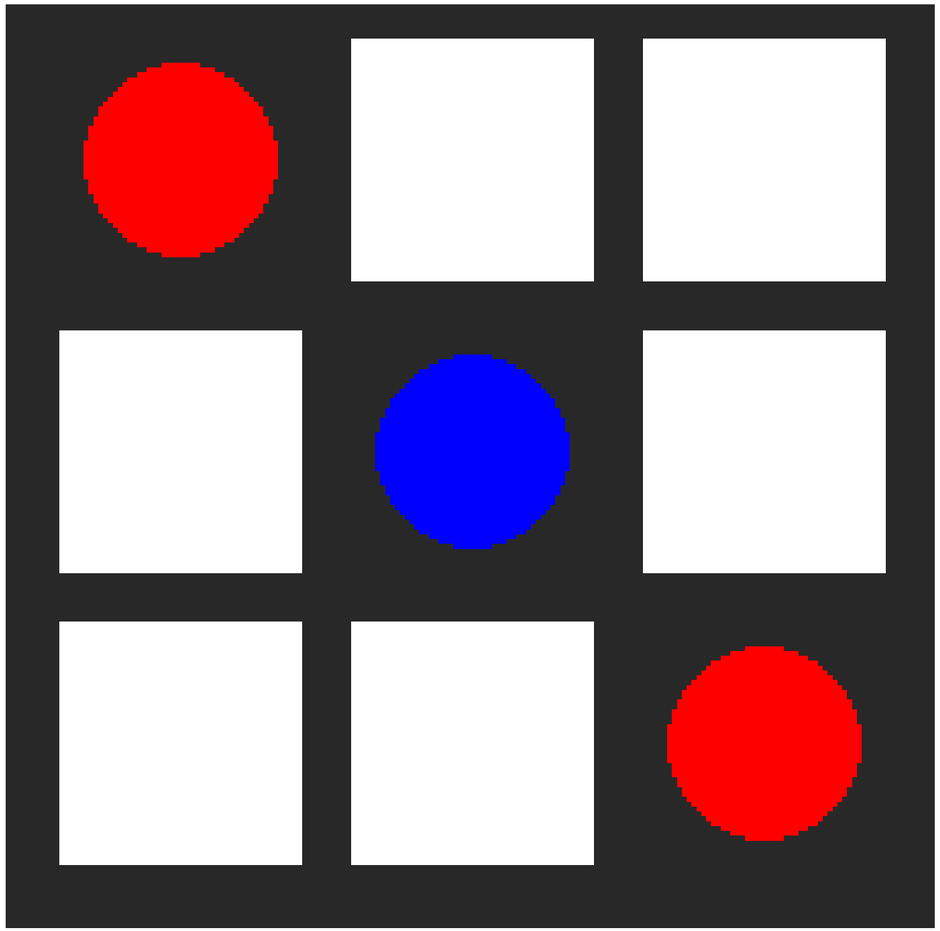
```
"bold_resource": {
        "package": "data.fonts",
        "resource": "Montserrat-Bold.ttf"
    },
"italic_resource": {
        "package": "data.fonts",
        "resource": "Montserrat-Italic.ttf"
    }
```

Paquetes __init__



Integración con matrices

```
[[1 0 0]
 [0 2 0]
 [0 0 1]]
```

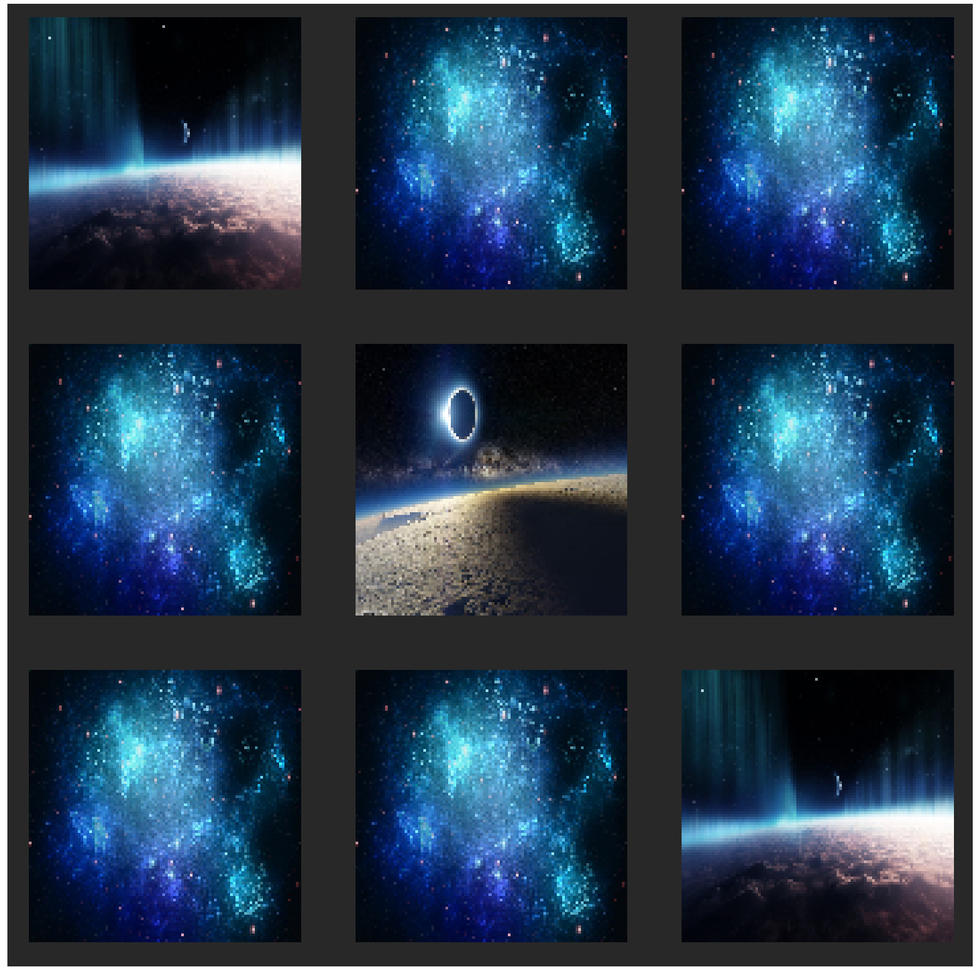


Integración con matrices

```
for i, row in enumerate(self.mat):
    for j, value in enumerate(row):
        x = 100 + j * 60
        y = 100 + i * 60
        if value == 0:
            py.draw.rect(surface, (255, 255, 255), (x, y, 50, 50))
        elif value == 1:
            py.draw.circle(surface, (255, 0, 0), (x + 25, y + 25), 20)
        else:
            py.draw.circle(surface, (0, 0, 255), (x + 25, y + 25), 20)
```

Integración con matrices

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Integración con matrices

```
# Cargar imágenes para las casillas
self.img_empty = py.image.load('data/images/space/space_1.jpg')
self.img_1 = py.image.load('data/images/space/space_2.jpg')
self.img_2 = py.image.load('data/images/space/space_3.jpg')
# Redimensionar imágenes a 50x50
self.img_empty = py.transform.scale(self.img_empty, (100, 100))
self.img_1 = py.transform.scale(self.img_1, (100, 100))
self.img_2 = py.transform.scale(self.img_2, (100, 100))
```

Integración con matrices

```
    for i, row in enumerate(self.mat):
        for j, value in enumerate(row):
            x = 100 + j * 120
            y = 100 + i * 120
            if value == 0:
                surface.blit(self.img_empty, (x, y))
            elif value == 1:
                surface.blit(self.img_1, (x, y))
            else:
                surface.blit(self.img_2, (x, y))
```

Paginación

0 Menu	**1** **Juego** **(Actual)**	2 Registro

Paginación

```python
from enum import Enum, auto


class ScreenList(Enum):
    MAIN_MENU = auto()
    INPUT_TEXT = auto()
    GAME = auto()
```

Paginación

1
Juego
(Actual)

- Kill todos los componentes de la pagina anterior
- Crear todo los componentes de la nueva pagina
- Empezar a ejecutar el bucle lógico de la siguiente pagina

0
Menu

Extensiones

- Color Highlight
- Error Lens
- Image preview
- Material Icon Theme

Gracias

github.com/DanielCarrenoMar