Documentación de la Clase Escenario

Gerstep

August 28, 2024

1 Introducción

La clase `Escenario` representa un conjunto de objetos 3D (`Objeto`) dentro de un entorno gráfico. Su propósito es organizar y gestionar múltiples objetos que pueden ser renderizados juntos en una escena. Un `Escenario` puede considerarse como el contenedor de todos los elementos visibles en una escena 3D, y es fundamental para la estructura y presentación visual de un proyecto de gráficos.

Esta clase pertenece al espacio de nombres `OpenTK_Tarea_3.Clases_Base` y es crucial para la construcción y gestión de escenas en aplicaciones que utilizan la biblioteca OpenTK.

2 Propiedades de la Clase

2.1 Objetos

- **Tipo de Dato:** `IReadOnlyList<Objeto>`
- **Descripción:** Esta propiedad expone una lista de solo lectura de los objetos (`Objeto`) contenidos en el `Escenario`. Cada `Objeto` representa un modelo 3D que forma parte de la escena.
- **Acceso:** Lectura. Esta propiedad permite acceder a la lista de objetos pero no permite modificarla directamente. Para agregar objetos a la lista, se debe utilizar el método `AgregarObjeto`.
- **Nota:** La lista subyacente es interna a la clase y se gestiona a través de la propiedad de solo lectura, lo que garantiza que la lista no pueda ser modificada accidentalmente desde fuera de la clase.

3 Métodos de la Clase

3.1 AgregarObjeto (Agregar un Objeto al Escenario)

```
public void AgregarObjeto(Objeto objeto)
```

3.1.1 Descripción

Este método se utiliza para agregar un nuevo objeto (`Objeto`) al escenario. Es esencial para construir la escena, ya que permite incluir diferentes modelos 3D que se mostrarán juntos en el entorno gráfico.

3.1.2 Parámetro

- `objeto`**:** Un objeto de tipo `Objeto` que se desea agregar al escenario. Este objeto debe estar previamente definido y no puede ser `null`.

3.1.3 Funcionamiento

El método `AgregarObjeto` sigue los siguientes pasos:

1. Verifica si el parámetro `objeto` es `null`.
2. Si `objeto` es `null`, se lanza una excepción de tipo `ArgumentNullException`, lo que indica que no se puede agregar un objeto nulo al escenario.
3. Si `objeto` no es `null`, se agrega a la lista interna de objetos (`_objetos`), que representa todos los elementos que forman parte de la escena.

3.1.4 Excepciones

- `ArgumentNullException`**:** Se lanza si el parámetro `objeto` es `null`. Esto es para asegurar que el escenario no intente manejar objetos no válidos.

3.1.5 Ejemplo de Uso

```
Escenario escenario = new Escenario();
Objeto objeto = new Objeto(new Vector3(0, 0, 0),
                           new Vector3(0, 0, 0),
                           new Vector3(1, 1, 1));

escenario.AgregarObjeto(objeto);

// Ahora el objeto forma parte del escenario y será
// considerado en la renderización de la escena.
```

3.2 DibujarEscenario (Renderizar el Escenario)

```
public void DibujarEscenario(Renderización renderizacion, Matrix4 view,
Matrix4 projection)
```

3.2.1 Descripción

Este método se encarga de renderizar todos los objetos contenidos en el escenario. Utiliza un objeto de tipo `Renderización` para dibujar cada `Objeto` en la escena, aplicando las matrices de vista y proyección para transformarlos correctamente en el espacio 3D.

3.2.2 Parámetros

- `renderizacion`: Un objeto de tipo `Renderización` que se utiliza para renderizar los objetos en el escenario.
- `view`: Una matriz (`Matrix4`) que representa la vista de la cámara, determinando desde qué punto de vista se observa la escena.
- `projection`: Una matriz (`Matrix4`) que representa la proyección, definiendo cómo los objetos se proyectan desde 3D a 2D en la pantalla.

3.2.3 Funcionamiento

El método `DibujarEscenario` sigue los siguientes pasos:

1. Itera a través de la lista interna de objetos (`_objetos`).
2. Para cada objeto en la lista, llama al método `RenderizarObjeto` del objeto `Renderización`, pasándole el objeto, la matriz de vista (`view`) y la matriz de proyección (`projection`).
3. Este proceso se repite para todos los objetos en la lista, asegurando que cada uno se dibuje correctamente en la escena.

3.2.4 Ejemplo de Uso

```
Matrix4 viewMatrix = Matrix4.LookAt(cameraPosition, target, up);
Matrix4 projectionMatrix = Matrix4.CreatePerspectiveFieldOfView(
                                MathHelper.DegreesToRadians(45.0f),
                                aspectRatio,
                                0.1f,
                                100.0f);

Renderización renderizacion = new Renderización();
Escenario escenario = new Escenario();
escenario.AgregarObjeto(objeto);

escenario.DibujarEscenario(renderizacion, viewMatrix, projectionMatrix);

// Esto dibuja todos los objetos en el escenario
// utilizando las matrices de vista y proyección especificadas.
```

4 Conclusión

La clase `Escenario` es fundamental para la organización y renderización de escenas en aplicaciones de gráficos 3D utilizando OpenTK. Al permitir agregar y gestionar múltiples objetos en un entorno estructurado, facilita la creación de escenas complejas y detalladas. El método `DibujarEscenario` proporciona un mecanismo eficiente para renderizar todos los objetos en una escena, aplicando las transformaciones necesarias para proyectar la escena correctamente en la pantalla.