Documentación de la Clase Objeto

Gerstep

August 28, 2024

1 Introducción

La clase `Objeto` es una estructura diseñada para representar un objeto 3D en un espacio tridimensional utilizando la biblioteca OpenTK. Esta clase maneja la posición, rotación y escala de un objeto, y está compuesta por múltiples partes (`Parte`), cada una de las cuales puede contener varios polígonos (`Poligono`).

Un `Objeto` puede verse como una entidad compleja que agrupa diferentes formas geométricas para construir un modelo 3D completo.

2 Propiedades de la Clase

2.1 Posicion (Posición)

- **Tipo de Dato:** `Vector3`
- **Descripción:** Esta propiedad almacena la posición del objeto en el espacio tridimensional. La posición se define mediante un vector 3D (`Vector3`) que contiene las coordenadas X, Y, y Z.
- **Acceso:** Lectura y escritura. Puedes obtener o modificar la posición del objeto usando esta propiedad.

2.2 Rotacion (Rotación)

- **Tipo de Dato:** `Vector3`
- **Descripción:** Esta propiedad almacena la rotación del objeto en el espacio tridimensional. La rotación se define mediante un vector 3D (`Vector3`) que contiene los ángulos de rotación en grados alrededor de los ejes X, Y, y Z.
- **Acceso:** Lectura y escritura. Puedes obtener o modificar la rotación del objeto usando esta propiedad.

2.3 Escala

- **Tipo de Dato:** `Vector3`
- **Descripción:** Esta propiedad almacena la escala del objeto en el espacio tridimensional. La escala se define mediante un vector 3D (`Vector3`) que contiene los factores de escala en las direcciones X, Y, y Z.
- **Acceso:** Lectura y escritura. Puedes obtener o modificar la escala del objeto usando esta propiedad.

2.4 Partes

- **Tipo de Dato:** `List<Parte>`
- **Descripción:** Esta propiedad almacena una lista de objetos de tipo `Parte`. Cada `Parte` representa una sección del objeto 3D y puede contener varios `Poligonos`. Esta estructura permite crear objetos complejos compuestos por múltiples formas geométricas.
- **Acceso:** Es de solo lectura, lo que significa que puedes acceder a la lista para agregar o manipular las partes del objeto, pero no puedes asignar una nueva lista a esta propiedad.

3 Constructor de la Clase

```
public Objeto(Vector3 posicion, Vector3 rotacion, Vector3 escala)
```

3.1 Descripción

Este constructor se utiliza para crear una nueva instancia de la clase `Objeto`. Al inicializar un objeto, se establecen su posición, rotación y escala en el espacio 3D. Además, se invoca el método `ConfigurarVertices_T()` para agregar una parte inicial al objeto.

3.2 Parámetros

- `posicion`: Un vector (`Vector3`) que define la posición del objeto en el espacio 3D.
- `rotacion`: Un vector (`Vector3`) que define la rotación del objeto en grados, alrededor de los ejes X, Y, y Z.
- `escala`: Un vector (`Vector3`) que define la escala del objeto en las direcciones X, Y, y Z.

3.3 Funcionamiento

Al crear un nuevo objeto utilizando este constructor, se inicializan las propiedades `Posicion`, `Rotacion`, y `Escala` con los valores proporcionados. Después, se llama automáticamente al método `ConfigurarVertices_T()`, que agrega una parte inicial al objeto, definiendo su geometría básica.

4 Métodos de la Clase

4.1 AgregarParte (Agregar una Parte)

```
public void AgregarParte(Parte parte)
```

4.1.1 Descripción

Este método se utiliza para agregar una nueva `Parte` al objeto. Las partes representan secciones del objeto 3D que pueden contener varios polígonos.

4.1.2 Parámetro

- `parte`**:** Un objeto de tipo `Parte` que se desea agregar a la lista de partes del objeto.

4.1.3 Funcionamiento

El método `AgregarParte` realiza las siguientes acciones:

1. Verifica si el parámetro `parte` es `null`.
2. Si `parte` es `null`, lanza una excepción de tipo `ArgumentNullException`, indicando que no se puede agregar una parte nula.
3. Si `parte` no es `null`, se agrega a la lista `Partes`, permitiendo que la parte se incluya en la representación del objeto.

4.1.4 Excepciones

- `ArgumentNullException`**:** Se lanza si el parámetro `parte` es `null`. Esto es para evitar errores al intentar agregar una parte no válida al objeto.

4.2 ConfigurarVertices_T (Configurar los Vértices de un T)

```
public void ConfigurarVertices_T()
```

4.2.1 Descripción

Este método configura la geometría de una parte inicial del objeto en forma de la letra `T`. Crea los vértices, colores e índices necesarios para formar un polígono en forma de `T` y luego lo agrega como una parte del objeto.

4.2.2 Funcionamiento

El método `ConfigurarVertices_T` realiza las siguientes acciones:

1. Define un arreglo de vértices que representan las posiciones en 3D para formar la figura en forma de `T`.
2. Define un arreglo de índices que establece cómo se conectan esos vértices para formar triángulos, que son las unidades básicas para construir superficies en gráficos 3D.
3. Define un arreglo de colores, que asigna un color específico a cada vértice del polígono.
4. Crea un objeto `Poligono` utilizando los vértices, colores e índices definidos.
5. Crea un objeto `Parte` y agrega el polígono recién creado a esta parte.
6. Agrega la parte a la lista de partes del objeto utilizando el método `AgregarParte`.

4.2.3 Ejemplo de Uso

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

// Esto crea un objeto en la posición (0, 0, 0)
// con rotación (0, 0, 0) y escala (1, 1, 1).
// También configura la geometría inicial en forma de "T".
```

4.3 ObtenerMatrizModelo (Obtener la Matriz de Modelo)

```
public Matrix4 ObtenerMatrizModelo()
```

4.3.1 Descripción

Este método devuelve la matriz de modelo del objeto, que es una combinación de las transformaciones de escala, rotación y traslación. Esta matriz es fundamental en gráficos 3D, ya que transforma los vértices del objeto desde su espacio local al espacio global.

4.3.2 Funcionamiento

El método `ObtenerMatrizModelo` realiza las siguientes transformaciones:

1. Aplica la escala del objeto mediante `Matrix4.CreateScale(Escala)`.
2. Aplica la rotación del objeto alrededor de los ejes X, Y, y Z mediante `Matrix4.CreateRotationX`, `Matrix4.CreateRotationY`, y `Matrix4.CreateRotationZ`, respectivamente.
3. Aplica la traslación del objeto utilizando `Matrix4.CreateTranslation(Posicion)`.

La matriz resultante es el producto de estas transformaciones, en el orden adecuado, y se utiliza para transformar el objeto en el espacio 3D durante el proceso de renderización.

4.3.3 Ejemplo de Uso

```
Matrix4 matrizModelo = objeto.ObtenerMatrizModelo();

// Esta matriz se puede utilizar para transformar
// y renderizar el objeto en la escena 3D.
```

5 Conclusión

La clase `Objeto` es una estructura fundamental para la construcción y manipulación de objetos 3D en aplicaciones gráficas utilizando OpenTK. A través de sus propiedades y métodos, permite definir la geometría, posición, rotación y escala de un objeto, así como gestionar sus diferentes partes. Esta clase es esencial para crear representaciones complejas y dinámicas en un entorno gráfico tridimensional.