Documentación del Código Program.cs

Gerstep

August 28, 2024

1 Introducción

El archivo `Program.cs` es el punto de entrada principal de una aplicación en C#. Este archivo contiene el código que se ejecuta primero cuando se inicia el programa. En el contexto de un juego o aplicación gráfica 3D desarrollada con OpenTK, `Program.cs` es responsable de crear e iniciar la instancia principal del juego, que es manejada por la clase `Game`.

El código que se describe a continuación muestra cómo crear una instancia de la clase `Game` y ejecutar el juego en una ventana de 1920x1080 píxeles con el título `"Letra T 3D"`.

2 Código Fuente

```
// Crear una instancia de la clase Game y ejecutar el juego
using OpenTK_Tarea_3;

using (Game game = new Game(1920, 1080, "Letra T 3D"))
{
    game.Run();
}
```

3 Explicación Detallada

3.1 Importación del Espacio de Nombres

```
using OpenTK_Tarea_3;
```

3.1.1 Descripción

Esta línea importa el espacio de nombres `OpenTK_Tarea_3`, que contiene la definición de la clase `Game` y otras clases necesarias para la aplicación. La importación permite el uso directo de las clases definidas en `OpenTK_Tarea_3` sin necesidad de escribir el nombre completo del espacio de nombres cada vez.

3.1.2 Funcionalidad

El espacio de nombres `OpenTK_Tarea_3` incluye la clase `Game`, que es la pieza central del juego. Esta clase hereda de `GameWindow` de OpenTK y maneja la inicialización, renderización y gestión de recursos del juego.

3.2 Creación de una Instancia de la Clase Game

```
using (Game game = new Game(1920, 1080, "Letra T 3D"))
```

3.2.1 Descripción

Esta línea de código crea una instancia de la clase `Game`, utilizando el constructor que toma tres parámetros: el ancho y la altura de la ventana en píxeles, y el título de la ventana.

3.2.2 Parámetros

- `1920`: El ancho de la ventana del juego en píxeles. En este caso, se establece en 1920 píxeles, lo que equivale a una resolución de pantalla Full HD en el eje horizontal.
- `1080`: La altura de la ventana del juego en píxeles. Se establece en 1080 píxeles, lo que equivale a una resolución de pantalla Full HD en el eje vertical.
- `"Letra T 3D"`: Una cadena de texto que establece el título de la ventana del juego. Este título se mostrará en la barra de título de la ventana cuando el juego esté en ejecución.

3.2.3 Funcionalidad

Al crear una instancia de `Game`, se inicializan todos los componentes necesarios del juego, incluyendo la ventana, la cámara, la entrada del usuario, los shaders, y el escenario 3D. El uso de `using` asegura que los recursos utilizados por el juego se liberen correctamente cuando el juego termine, llamando automáticamente al método `Dispose` de la clase `Game`.

3.3 Ejecución del Juego

```
{
    game.Run();
}
```

3.3.1 Descripción

Esta línea ejecuta el método `Run()` de la instancia de `Game`. Este método inicia el bucle principal del juego, que controla la actualización y renderización de cada frame.

3.3.2 Funcionalidad

El método `Run()` inicia el ciclo de vida del juego, que incluye:

1. **Inicialización:** Configuración inicial del entorno de OpenGL y carga de recursos.
2. **Bucle de Juego:** Un bucle que se ejecuta continuamente hasta que el juego se cierra. Este bucle maneja la entrada del usuario, la lógica del juego, y la renderización de la escena en cada frame.
3. **Limpieza:** Cuando el juego se cierra, se liberan todos los recursos utilizados, como shaders, buffers de OpenGL, y texturas.

El método `Run()` se asegura de que el juego funcione correctamente desde el momento en que se inicia hasta que se cierra la ventana.

3.4 Finalización Automática del Juego

```
using (Game game = new Game(1920, 1080, "Letra T 3D"))
```

3.4.1 Descripción

El uso de la palabra clave `using` asegura que, cuando el método `Run()` finaliza, el juego se cierre de manera ordenada y todos los recursos sean liberados correctamente. Este enfoque es esencial para evitar pérdidas de memoria y otros problemas relacionados con la gestión inadecuada de recursos.

3.4.2 Funcionalidad

Cuando el bloque `using` termina, se llama automáticamente al método `Dispose()` de la clase `Game`, que se encarga de liberar los recursos asociados con la instancia del juego, tales como shaders, buffers y otros recursos de OpenGL.

4 Conclusión

El archivo `Program.cs` es el punto de entrada para la ejecución del juego. Este código simple y conciso se encarga de crear la ventana del juego, inicializar todos los componentes necesarios, y ejecutar el ciclo de vida del juego hasta que el usuario decida cerrarlo. El uso del bloque `using` asegura una gestión adecuada de los recursos, garantizando que el juego se ejecute de manera eficiente y que los recursos se liberen correctamente al finalizar.