```
using System;
using UnityEngine;

/// <summary>
/// GridMaintenance instantiates and maintains the visible grid in the editor and preview scenes.
/// </summary>
public class GridMaintenance : MonoBehaviour
{
    // Singleton state reference
    private static GridMaintenance instance;

    /// <summary>
    /// The global height that all in-scene placements are placed at.<br/><br/>
    /// The game calculates the x and z positions by raycasting to an x-z plane located at this height.
    /// </summary>
    [SerializeField]
    float gridHeight;

    /// <summary>
    /// The prefab that displays the in-scene grid.
    /// </summary>
    [SerializeField]
    GameObject gridReference;

    /// <summary>
    /// The instantiated grid; a copy of <seealso cref="gridReference"/>.
    /// </summary>
    private GameObject grid;

    /// <summary>
    /// The material of the grid.
    /// </summary>
    private Material gridMaterial;

    /// <summary>
    /// The material texture offset of the grid.<br/><br/>
    /// Because the grid follows the camera, its material must move opposite to the direction of camera movement to create an illusion of it standing still.
    /// </summary>
    private Vector2 materialOffset;

    /// <summary>
    /// Utilized to calculate the delta position between frames.
    /// </summary>
    private Vector3 currentPos;

    // Enforces a singleton state pattern.
    private void Awake()
    {
        if (instance != null)
        {
            Destroy(this);
            throw new Exception("GridMaintenance instance already 
              established; terminating.");
        }

        instance = this;
    }

    // Instantiates the grid and its respective values.
    private void Start()
    {
        grid = Instantiate(gridReference);
        grid.name = "Grid";
        grid.transform.position = new Vector3(transform.position.x, 
          gridHeight, transform.position.z);
        grid.transform.eulerAngles = Vector3.zero;
        gridMaterial = grid.GetComponent<MeshRenderer>().material;
        currentPos = transform.position;
        materialOffset = gridMaterial.GetTextureOffset("_MainTex");
    }

    // Obtains the change in position from the last frame and alters 
      materialOffset by an opposite value.
    private void Update()
    {
        grid.transform.position = new Vector3(transform.position.x, 
          gridHeight, transform.position.z);

        Vector3 oldPos = currentPos;

        currentPos = transform.position;

        Vector3 deltaPos = currentPos - oldPos;
        Vector2 realDeltaPos = new Vector2(deltaPos.x, deltaPos.z);

        materialOffset += realDeltaPos;
        materialOffset = new Vector2(materialOffset.x % 1, materialOffset.y 
           % 1); // The grid is 1x1, therefore it can be clamped.
        gridMaterial.SetTextureOffset("_MainTex", materialOffset);
    }

    // Getter methods
    public static GridMaintenance Instance { get { return instance; } }

    public float GridHeight { get { return gridHeight; } }
}
```