









































The screenshot shows the Visual Studio code editor with the file `RoundCameraPos.cs` open. The code implements a `CinemachineExtension` that rounds the camera's final position after the post-processing pipeline. It uses `Mathf.Round` to round the `finalPos` to `PixelsPerUnit`.

```
using UnityEngine;
using Cinemachine;

public class RoundCameraPos : CinemachineExtension
{
    public float PixelsPerUnit = 32;

    protected override void PostPipelineStageCallback(
        CinemachineVirtualCameraBase vcam,
        CinemachineCore.Stage stage, ref CameraState state, float deltaTime)
    {
        // Check to see what stage of post-processing we're in
        if (stage == CinemachineCore.Stage.Body)
        {
            // Get the VC's final position
            Vector3 finalPos = state.FinalPosition;

            // Call the method we wrote to round the position
            Vector3 newPos = new Vector3(Round(finalPos.x), Round(finalPos.y), finalPos.z);
            // Set the VC's final position to the difference between the old
            // position and the new rounded position that we just calculated
            state.PositionCorrection += newPos - finalPos;
        }
    }

    float Round(float x)
    {
        return Mathf.Round(x * PixelsPerUnit) / PixelsPerUnit;
    }
}
```









