```
using System.Collections.Generic;
using UnityEngine;

/// <summary>
/// Logical representation of a DISPLAY.
/// </summary>
public class Display : Circuit
{
    /// <summary>
    /// Similar to <seealso cref="pins"/>, each value refers to the material that is rendered for a corresponding pin when the user's cursor hovers over its corresponding node.
    /// </summary>
    private GameObject[] previewPins = new GameObject[8];

    /// <summary>
    /// Similar to <seealso cref="previewPins"/>, each value refers to the material that is rendered for a corresponding pin when its corresponding node is powered.
    /// </summary>
    private MeshRenderer[] pins = new MeshRenderer[8];

    public Display() : this(Vector2.zero) { }

    public Display(Vector2 startingPos) : base("DISPLAY", 8, 0, startingPos) { }

    /// <summary>
    /// Updates each pin based on the power status of its corresponding node.
    /// </summary>
    /// <returns>An empty list of outputs.</returns>
    protected override List<Output> UpdateOutputs()
    {
        for (int i = 0; i < 8; i++) pins[i].material = Inputs[i].Powered ? CircuitVisualizer.Instance.PowerOnMaterial : CircuitVisualizer.Instance.PowerOffMaterial;

        // Always returns an empty list as a DISPLAY has no output nodes.
        return new List<Output>();
    }

    // Getter and setter method
    public GameObject[] PreviewPins { get { return previewPins; } set { previewPins = value; } }

    // Setter method
    public MeshRenderer[] Pins { set { pins = value; } }
}
```