```
using System;
using System.IO;
using UnityEngine;

/// <summary>
/// ConnectionSerializer stores mesh and index information pertaining to the assigned connection for serialization.
/// </summary>
[Serializable]
public class ConnectionSerializer
{
    /// <summary>
    /// Whether <seealso cref="startingMesh"/> is equal to <seealso cref="endingMesh"/>.
    /// </summary>
    [SerializeField]
    bool singleWired;

    /// <summary>
    /// Contains relevant index information used to identify the connection's input and output circuit(s).
    /// </summary>
    [SerializeField]
    CircuitConnectorIdentifier circuitConnectorIdentifier;

    /// <summary>
    /// Serialized mesh data for the starting wire.
    /// </summary>
    [SerializeField]
    MeshSerializer startingMesh;

    /// <summary>
    /// Serialized mesh data for the ending wire.
    /// </summary>
    [SerializeField]
    MeshSerializer endingMesh;

    /// <summary>
    /// Serialized mesh data for the non-starting/ending wires.<br/><br/>
    /// If there is only a starting and ending wire, its value will be null.
    /// </summary>
    [SerializeField]
    MeshSerializer parentMesh;

    // Private constructor; a ConnectionSerializer can only be instantiated through its primary constructor.
    private ConnectionSerializer() { }

    /// <summary>
    /// Instantiates and populates a <seealso cref="ConnectionSerializer"/> with the assigned values; saves to the provided path.
    /// </summary>
    /// <param name="connection">The connection to serialize.</param>
    /// <param name="circuitConnectorIdentifier">The obtained <seealso cref="CircuitConnectorIdentifier"/> representing this connection.</param>
    /// <param name="path">The directory to save the serialized information.</param>
    public static void SerializeConnection(CircuitConnector.Connection connection, CircuitConnectorIdentifier circuitConnectorIdentifier, string path)
    {
        ConnectionSerializer connectionSerializer = new ConnectionSerializer();

        // Assigns relevant values
        connectionSerializer.circuitConnectorIdentifier = circuitConnectorIdentifier;
        connectionSerializer.startingMesh = new MeshSerializer(connection.StartingWire.transform.GetChild(0).GetChild(0).GetComponent<MeshFilter>().mesh, connection.StartingWire.transform);
        connectionSerializer.endingMesh = new MeshSerializer(connection.EndingWire.transform.GetChild(0).GetChild(0).GetComponent<MeshFilter>().mesh, connection.EndingWire.transform);
        connectionSerializer.singleWired = connection.StartingWire == connection.EndingWire;

        // If the connection has a parent mesh, serialize its information and save to parentMesh.
        if (connection.GetComponent<MeshFilter>() != null) connectionSerializer.parentMesh = new MeshSerializer(connection.GetComponent<MeshFilter>().mesh, connection.transform);

        // Writes object to directory
        File.WriteAllText(path, JsonUtility.ToJson(connectionSerializer));
    }

    // Getter methods
    public bool SingleWired { get { return singleWired; } }

    public CircuitConnectorIdentifier CircuitConnectorIdentifier { get { return circuitConnectorIdentifier; } }

    public MeshSerializer StartingMesh { get { return startingMesh; } }

    public MeshSerializer EndingMesh { get { return endingMesh; } }

    public MeshSerializer ParentMesh { get { return parentMesh; } }
}
```