Typical code structure for C# custom simulations

The using statements import external libraries so that their classes, fields and methods could be used in the ***Script_Instance*** *class*

RunScript *is a method executed at each canvas update or data input update. The variables created inside are burned at the end of the execution of the method (i.e. connecting a timer to the component). This is the place for volatile data.*

This area is a more general context outside ***RunScript*** *and inside the* ***Script_Instance*** *class, therefore classes, methods and variables created here do not expire once RunScript is executed. They expire and are reset at every new instance of this component on the GH canvas*
This is the place for permanent data.

```
// <Custom using>

// </Custom using>

public class Script_Instance : GH_ScriptInstance
{

        private void RunScript( <RunScript parameters> )
        {
            // <Custom code>

            // </Custom code>
        }

    // <Custom additional code>

    // </Custom additional code>
        …
}
```

Typical code structure for C# custom simulations

Import any necessary additional library here

A reset button restores the state of classes and variables to the first execution (clears old values, initializes new instances, etc.)

Secondly, any field that might be updated during the execution is passe here from the outside to the simulation class.

The simulation runs one step, typically calling an Update() method and then using Component.ExpireSolution(true) to trigger a new execution.

The last thing to do here is to extract the data to be sent to the output parameters

Global variables (whose value does not expire at each step, like a counter)

Typically, there is a general class that runs the entire simulation and then there are sub-classes that share properties with the general class triggered during the execution of the main simulation.

The script might also contain helper classes and methods (such as format converters etc.)

```
// <Custom using>

// </Custom using>

public class Script_Instance : GH_ScriptInstance
{

        private void RunScript( <RunScript parameters> )
        {
            // <Custom code>

            // 00 . reset and initialization code

            // 01 . update live fields

            // 02 . run the simulation and expire component solution

            // 03 . extract to output parameters

            // </Custom code>
        }

      // <Custom additional code>

      // global variables

      // simulation classes

      // helper classes

      // miscellaneous helper methods

      // </Custom additional code>
        …
}
```