

# Custom script

---

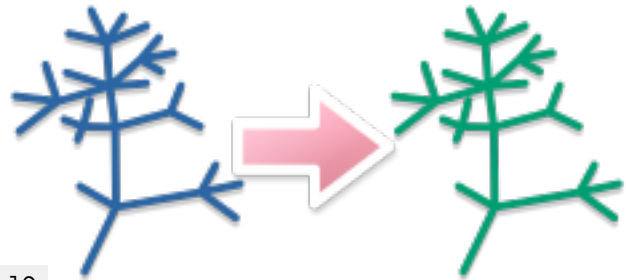
*Version 1.0.0, by Giorgio Bianchini*

**Description:** Executes custom code.

**Module type:** FurtherTransformation

**Module ID:**

a76d00d2-95e0-4274-a77d-1439a013e3d9



This module makes it possible to execute custom C# code to transform the tree. This can be useful either to perform one-off complicated modifications of the tree, or as a first step in developing a new module for TreeViewer.

## Parameters

---

### Description

**Control type:** Text box

**Default value:** Describe the script

This parameter can be used to provide a short description to quickly identify what the module does without having to look at the source code. It is ignored by the module.

### Source code

**Control type:** Source code

**Default value:**

```
using PhyloTree;
using System.Collections.Generic;
using TreeViewer;

namespace a67db8653574f44b3b3d6143e7157f24f
{
    //Do not change class name
    public static class CustomCode
    {
        //Do not change method signature
        public static void PerformAction(ref TreeNode tree,
            TreeCollection trees, InstanceStateData stateData)
        {
            //TODO: do something with the tree
        }
    }
}
```

```
}  
}  
}
```

This parameter contains the source code of the script. The arguments to the `PerformAction` method are as follows:

- `tree`: the transformed tree that has been computed by the Transformed module and any preceding Further transformation modules.
- `trees`: the collection of trees that were originally read from the file.
- `stateData`: an `InstanceStateData` object that can be used to access features in way that does not depend on the program running in command-line or GUI mode.

## Further information

---

The difference between this module and the other module with the same name is that this module acts as a Further transformation, while the other *Custom script* module (id `cdb74bfb-8a90-48b3-815a-8f908d2a1ff5`) is instead a Plot action.

The code in the module can do anything, including loading additional data from a file on disk. However, this is discouraged, because it ties the tree file on the computer it was created on. A better approach to load additional data would be to import the data file as an attachment and read the data from the attachment. Attachments can be accessed using the `Attachments` property of the `stateData` object that is passed as a method parameter.

Furthermore, since the code in the module can do anything, it may also be a security risk to open files originating from unknown sources; thus, you should either make sure that any file you open comes from a reputable source, or avoid loading source code from tree files at all.