ScriptClass JavaScript Code

Write ScriptClasses in JavaScript.

Technical Reference **Programming and APIs Reference** **Scripting Code Library** JavaScript

---

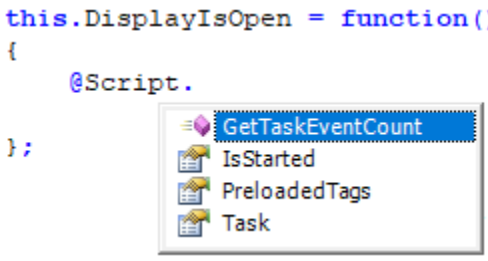
This technical note has information related to how to call a class on HTML5 (@Script.Class.<ClassName>).

How to Use
- Framework does not allow to call a class on HTML5
- The Implementation
- Example 1
- Example 2

How to Use

Framework does not allow to call a class on HTML5

It is not possible to call `@Script.Class` using default method that is used on WPF Pages.



You can see that there is no option to call a class. The following topic shows how to make it possible.

The Implementation

To make the call of classes on a HTML5 page possible, it is necessary to execute a code behind where `TK.ExecuteClassMethodOnServerAsync` is called with `await`.

Example 1

You can see an example of the implementation:

CodeBehind Example

```
this.MouseLeftButtonDown1 = async function(sender, e)
{
    var parameters = new Array("Tag1");
    var result = await TK.ExecuteClassMethodOnServerAsync("TagDescription", "Func", parameters);
}
```

Where:

- **TagDescription:** Scripts Classes "Class Name"
- **Func:** Scripts Classes "Class Name" "Function Name"
- **Parameters:** Tag passed as parameter

You can see in the Code Block below an example of the Script implementation calling the function "func":

Script Example

```
public string Func(String TagName)
{
    string tag = TagName;
    return tag + TK.GetObjectValue("Tag." + TagName + ".Description");
}
```

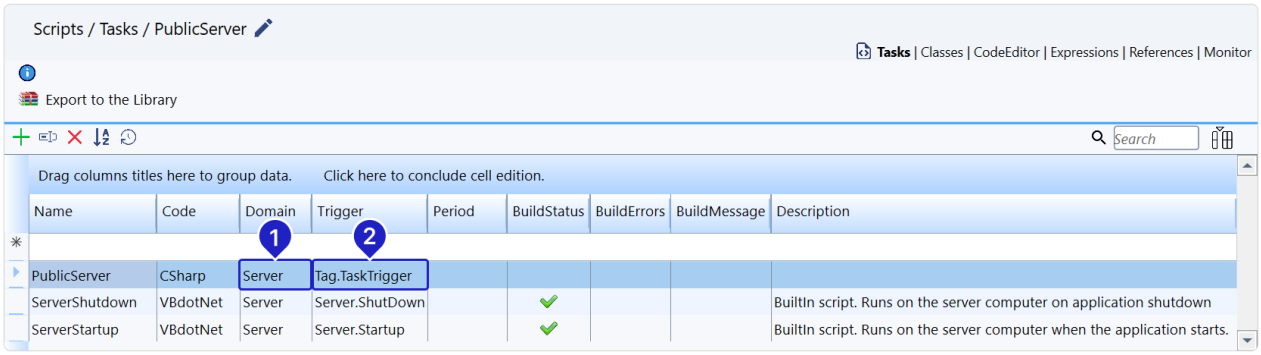
In our case, it will return the Tag Name and its description. Following those steps you can call a Class from HTML5.

Example 2

You can run a Task or a Class through an HTML5 page. However, since the majority of the scripts are .NET and will run on the Server and not directly on the Client, follow the steps below.

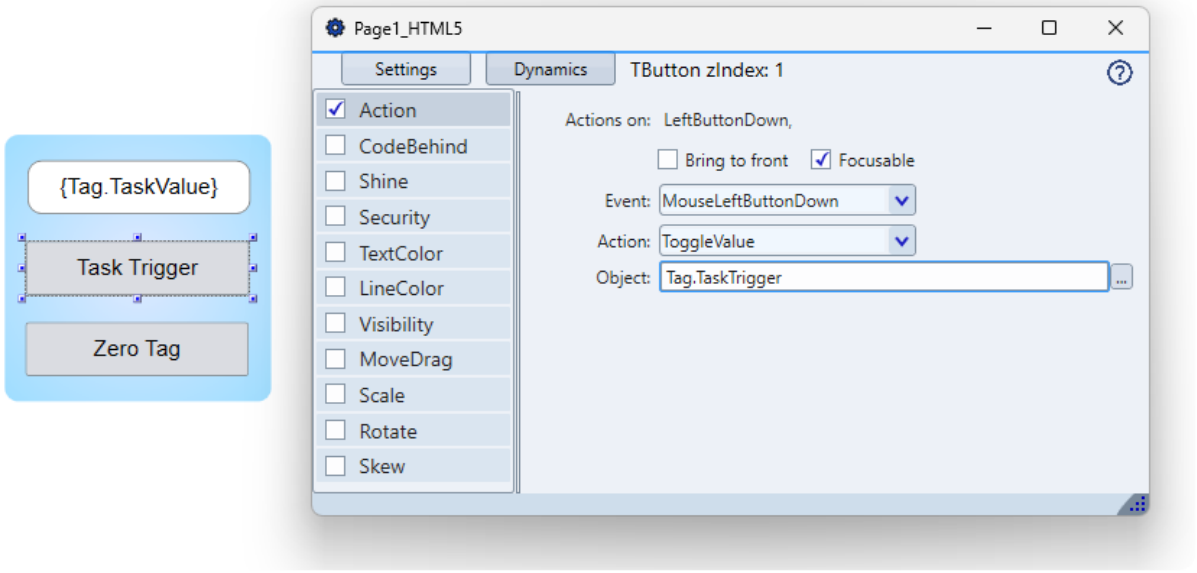
When creating a Task or Class, you need to set the "Domain" column to the Server option. When creating the Tags that will go inside the Task or Class, you need to create them with the Server type Domain as well.

To activate a Task, you need to create a preferably Digital type tag that will serve as a Trigger as in the image below:



On the HTML5_MainPage display, we have a button called "Trigger Task" which is set to perform a toggle directly on the Tag previously configured in the Task Trigger.

Each time the button is pressed, the task will run on the Server.



To use a Class, it is necessary to use the TK class and the `ExecuteClassMethodOnServerAsync()` method, called with `await` from an `async` function.

There are three parameters that must be passed to this method. The first parameter is the name of the class created in **Scripts/Classes**, this parameter is a String.

The second parameter is the name of the method created within the Class, this parameter is also a String. And the Third parameter are the parameters that can be passed to this class, or can be left as null, the passed parameters need to be Array type.

Here is the syntax of the server-side .NET method. In JavaScript, use the Async version: it takes the same three parameters and returns the result through `await`.

C# Syntax

```
public static object ExecuteClassMethodOnServer(
   string className,
   string methodName,
   object[] parameters
)
```

In the code block below we have a class called PublishClass and two methods: the Main method, which does not receive any parameters and does not return anything, and the ReturnValue method, which receives a string as a parameter and returns a string.

JS Examples

```
this.CallClass = async function(sender, e)
{
        await TK.ExecuteClassMethodOnServerAsync("PublishClass", "Main", null);
}


this.CallClassWithReturn = async function(sender, e)
{
        var parameter = new Array("Value 01");
        @Tag.ResultClass = await TK.ExecuteClassMethodOnServerAsync("PublishClass", "ReturnValue",
parameter);
}
```

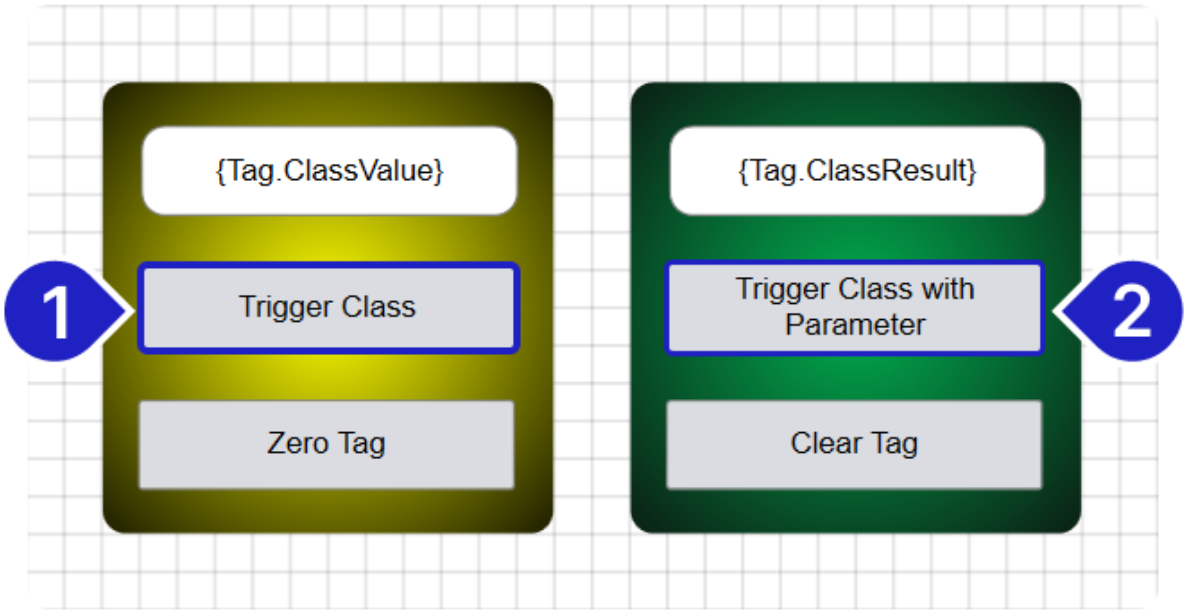
To call each of the methods mentioned above, we use the following code in the Code Behind of an HTML5 page.

Within the CallClass function we call the Main method, and within the CallClassWithReturn function we call the ReturnValue method.

```
public void Main ()
{
        Tag.TagClassValue += 1;
        @Info.Trace ("TEST | | Main Class");
}

public string ReturnValue(string msg)
{
        string result = "msg: " + msg;
        @Info.Trace("TEST | | Return Class");
        return result;
}
```

These two functions are called through the Actions of the buttons created in the HTML5_MainPage display as shown in the image below:



In this section...
