

## Visual Studio Code (VSC)

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, MacOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtime (such as .NET and Unity). Begin your journey with VS Code with these introductory [videos](#).

VSC is the base for the [ExtendScript debugger for VSCode \(VSC-ESD\)](#) on page 264.

- <https://code.visualstudio.com/>
- <https://github.com/microsoft/vscode/wiki/Coding-Guidelines>
- <https://code.visualstudio.com/docs/getstarted/userinterface>

### Command line for VSC

Open code with current directory

```
code .
```

Open the current directory in the most recently used code window

```
code -r .
```

Create a new window

```
code -n
```

Change the language

```
code --locale=es
```

Open diff editor

```
code --diff <file1> <file2>
```

Open file at specific line and column <file:line[:character]>

```
code --goto package.json : 10:5
```

See help options

```
code --help
```

Disable all extensions

```
code --disable-extensions .
```

Open from other application

The commands assumes the environment path set for code. This can be replaced by the full path, for example (Editpad Pro > Tools):

```
H:\Programming\Microsoft_VS_Code\Code.exe --goto %FILE%:1
```

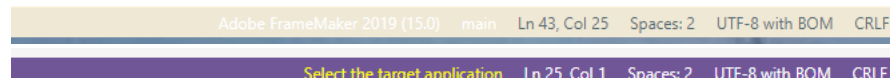
This can work only, if the saved workspace includes an open folder.

To start debugging you need to select the target engine at the bottom.

Syntax highlighting is optimised for dark themes - not to my liking. The light theme I settled with is NetBeans Light Theme.

Attention

Some light colour schemes do not show relevant things in a recognisable manner. For example the very important last line of the window:



```
Adobe FrameMaker 2019 (15.0) main Ln 43, Col 25 Spaces: 2 UTF-8 with BOM CRLF
Select the target application Ln 25, Col 1 Spaces: 2 UTF-8 with BOM CRLF
```

Another annoyance (at least for me) is the location where information is presented: Clicking on Select the target

application at the *very bottom* opens a list at the *top of the window*.

## Personal settings

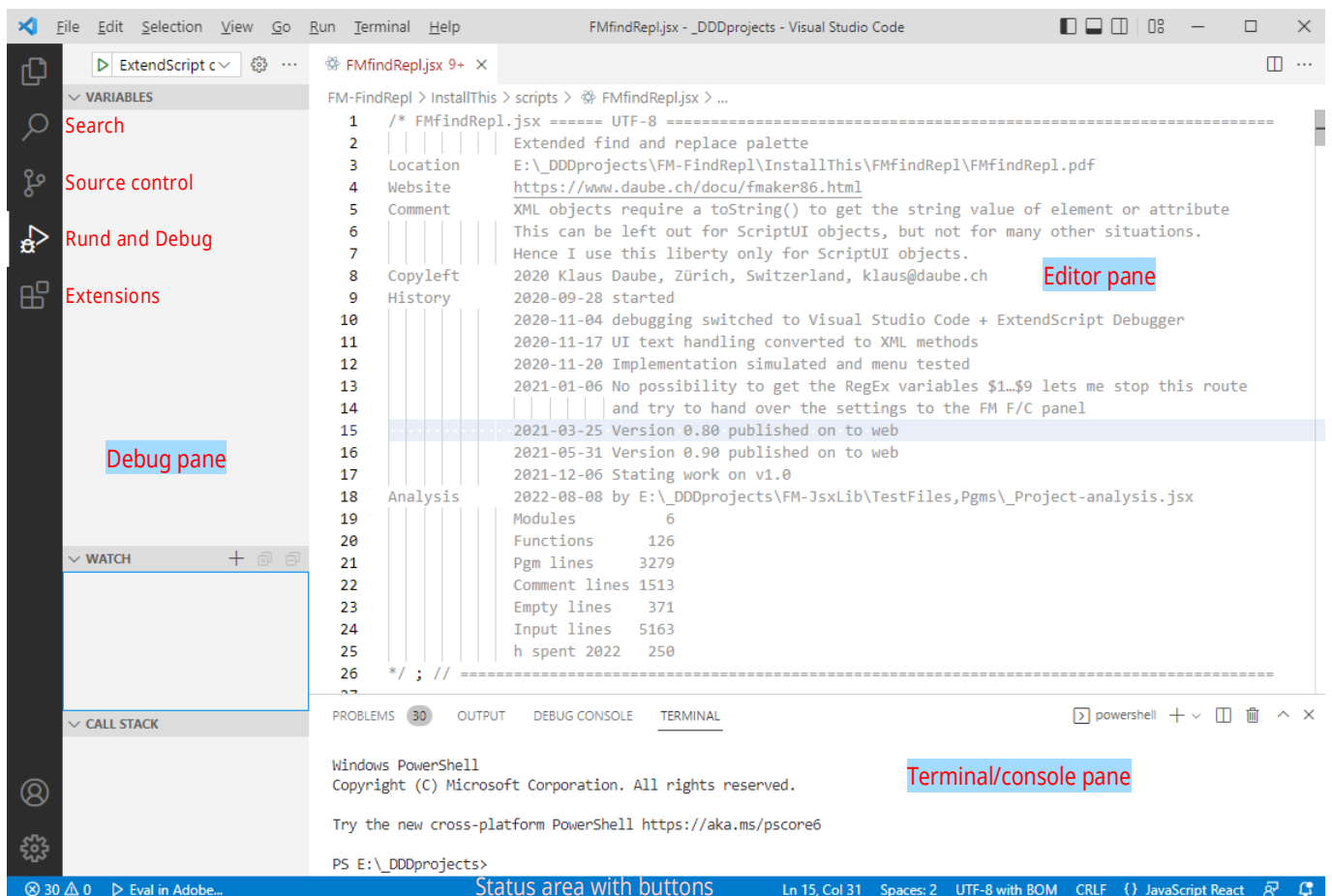
**File > Preferences > Settings >...**

(Only changes to the default are listed)

Editor	Render ctrl chars	off
	Render white space	all
	Rounded selection	off
	Tab Size	2
Minimap	Enabled	off
Appearance	Color Customization	→ settings.json
	Color Theme	Netbeans Light Theme
	Pref. Light theme	Netbeans Light Theme
	Side Bar: location	left
Editor Management	Title Scrollbar Sizing	large (this does not widen editor scroll bar).
	Zen mode	Full screen
	Extension	Ignore Recommend.

## Main user interface

The following view is the result of opening VSCode and then opening a folder with **File > Open Folder**. This folder contains a sub-folder `.vscode` which in turn contains the file `launch.json`.



## ExtendScript debugger for VSCode (VSC-ESD)

Download and Documentation	<a href="https://marketplace.visualstudio.com/items?itemName=Adobe.extendscript-debug">https://marketplace.visualstudio.com/items?itemName=Adobe.extendscript-debug</a>
Current version	2.0.3 (2022-07-14)

**Note:** *The documentation calls this an extension, because it is a ‘«A Visual Studio Code Extension that enables debugging ExtendScript-based extensions in Adobe’s applications that support ExtendScript.»’.*

### forum

[forums.adobeprerelease.com/exmancmd/categories/estkvsc](https://forums.adobeprerelease.com/exmancmd/categories/estkvsc)  
[community.adobe.com/t5/coding-corner/ct-p/ct-coding-corner](https://community.adobe.com/t5/coding-corner/ct-p/ct-coding-corner)

### Main features

For details see the [documentation](#).

- Breakpoints
  - Conditional Breakpoints
  - Expression Condition
  - Hit Count
  - Exception Breakpoints:
  - Caught Exceptions <sup>42)</sup>
- Logpoints are a variant of a breakpoints that does not break into the debugger but instead logs a message to the console.
- Variables View
  - Local and Global Scope
  - Modify variables
- Watch View
- Call Stack View
- Debug Actions
  - Continue / Pause                    F5
  - Step Over                              F10
  - Step Into                                F11
  - Step Out                                 Shift+F11
  - Restart                                 Ctrl+Shift+F5
  - Disconnect / Stop                    Shift+F5
- Debug Console: Expression Evaluation
- Expression Evaluation of Code on Hover <sup>43)</sup>
- Script Evaluation and Halting
- Export ExtendScript to JSXBin

### Unsupported features

- Unsupported features
- Profiling support
  - OMV (Object Model View)
  - Auto-Completion
  - Scripts Panel

<sup>42</sup> Changes to the Caught Exceptions setting while a script is running or stopped at a breakpoint will only apply to scopes created after the setting is changed.

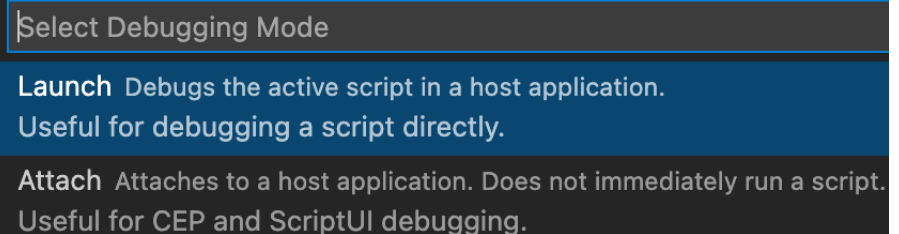
<sup>43</sup> Requires active debug session.

## Some details from the documentation

Attach and Launch Mode Support	VSC-ESD ignores the <code>#target</code> and <code>#targetengine</code> preprocessor directives. The extension will always use either the configured <code>hostAppSpecifier</code> and <code>engineName</code> settings or, if not otherwise specified, those dynamically chosen in the relevant UI.
Conflict with ESTK	If ESTK connects to a host application <sup>44</sup> ), then the ExtendScript Debugger extension will no longer be able to function correctly as a debugger. Restarting the host application is enough to fix this issue.
VS Code Status Bar Buttons	VSC-ESD adds two new buttons to the Status Bar that appear/disappear based on context.
<b>Eval in Adobe...</b> Button	<p>This button appears in the status area when a document either is recognized by VS Code as <code>javascript</code>, <code>javascriptreact</code>, or <code>extendscript</code>, or has a file extension of <code>.jsxbin</code></p> <p>Clicking this button triggers the <code>Evaluate Script in Host...</code> command. Once a target host application/engine combination is chosen, the contents of the focused document will be evaluated within it.</p> <p>When an attach mode debug session is active, the <b>Eval in Adobe...</b> button changes to read <b>Eval in Adobe [name of application] (engine)...</b> Clicking this button will evaluate the focused script in the application being debugged.</p>

## Starting behaviour

- When you click **Run and Debug**, you are not using your configuration file `launch.json`.  
That said, the `no-configuration` approach still works. By default, a `no-configuration` run will attach to a target application. This process does not run the script in the host. If you want to run the script, you need to click that **Eval in Adobe Framemaker...** button. At that point, breakpoints will be active and Debug output should appear in the Debug Console.
- When there is no `launch.json` configuration available, the **Run and Debug** button has been updated to ask if you would like to start an "Attach" or "Launch" mode debug session. This should hopefully make things a lot clearer going forward.



## launch.json

This file must reside in a directory named `.vscode` hierarchically above the script libraries:

```
E:\_DDDprojects\.vscode\launch.json
```

<sup>44</sup> This may happen, if you run the script from the FM script library and an error occurs - which will start the ESTK. Later use of VSC-ESD does not work, until you restart FM

D:\System\_ddd\scripts\.vscode\launch.json

I have defined the following contents in all of these files (imported from the second noted above):

```
{
// E:\_DDDprojects\.vscode\launch.json and D:\System_ddd\scripts\.vscode\launch.json
// Use IntelliSense to learn about possible attributes.
// Hover to view descriptions of existing attributes.
// For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
      "name": "ExtendScript debug",    // V 2.0.3
      "type": "extendscript-debug",
//      "request": "laucnch",           // immediately run script
      "request": "attach",           // does not immediately run script
      "hostAppSpecifier": "framemaker",
      "engineName": "main",
//      "debugLevel": 2                // requires "launch"
    },
    {
      "name": "AHK debug",            // AHK++ debugger
      "type": "Autohotkey Plus Plus",
      "request": "launch",
      "program": "${file}",
      "stopOnEntry": true
    }
  ]
}
```

## launch.statement

If the `launch.json` file is not active (not found), then a no configuration approach is taken. By default, this will attach to a target application.

<code>request</code>	<code>attach</code>	does not run the script in the host. To run the script, you need to click that Eval in Adobe ...
	<code>launch</code>	Required for <code>debuglevel</code> to work
<code>hostAppSpecifier</code>		Either a generic or a specificity such as <code>framemaker-15.0</code>
<code>engineName</code>		The name of the engine to target. For FM this is always <code>main</code> .
<code>hiddenTypes</code>		An array of data types and class names that should be hidden in the Variables view. Valid names are: <code>undefined</code> , <code>null</code> , <code>boolean</code> , <code>number</code> , <code>string</code> , <code>object</code> , <code>this</code> , <code>prototype</code> , <code>builtin</code> , <code>function</code> and any valid ExtendScript class name.  The string <code>this</code> hides the <i>this</i> object. The string <code>prototype</code> hides all elements from the prototype chain, and the string <code>builtin</code> hides all elements that are part of the core ExtendScript language.
<code>aliasPath</code>		The absolute path to a file system alias (symbolic link) for the root directory loaded by a host application.
<code>debuglevel</code>		Works only for <code>request: launch</code> , not for <code>attach</code> . 0 No debugging 1 Break on breakpoints, errors or exceptions 2 Stop at the first executable line (~ Stop on Entry)
<code>script</code>		<code>\${workspaceFolder}/\${command:AskForScriptName}</code> "

## My personal method to start ESD

I have not yet figured out why the `launch.json` file is not considered and I'm working in *no-configuration* mode. The following note seems not to be the full truth:

**Note:** *At the start of the VSC session a folder must be opened which contains the `launch.json` file within its root. Only this way the `json` file becomes active.*

1 From a currently saved file I launch VSC via this statement in my general editor<sup>45</sup>):

```
H:\Programming\Microsoft_VS_Code\Code.exe --goto %FILE%:1
```

2 This opens VSC editor:

```

File Edit Selection View Go Run Terminal Help
VerySimple.jsx - Visual Studio Code
VerySimple.jsx x
E: > _DDDprojects > FM-JsxLib > TestFiles,Pgms > VerySimple.jsx > ...
1 // E:\_DDDprojects\FM-JsxLib\TestFiles,Pgms\VerySimple.jsx
2 //@target framemaker
3
4 function main () {
5   var aaa = 123456789;
6   $.writeln("Does this appear? ", aaa);
7   alert ("Hello, here I am");
8   $.bp(true);
9 } // end of main
10
11 main ();
Ln 1, Col 1 Spaces: 2 UTF-8 CRLF {} JavaScript React

```

3 Now I use **F5** to get the pick-list of the extensions.

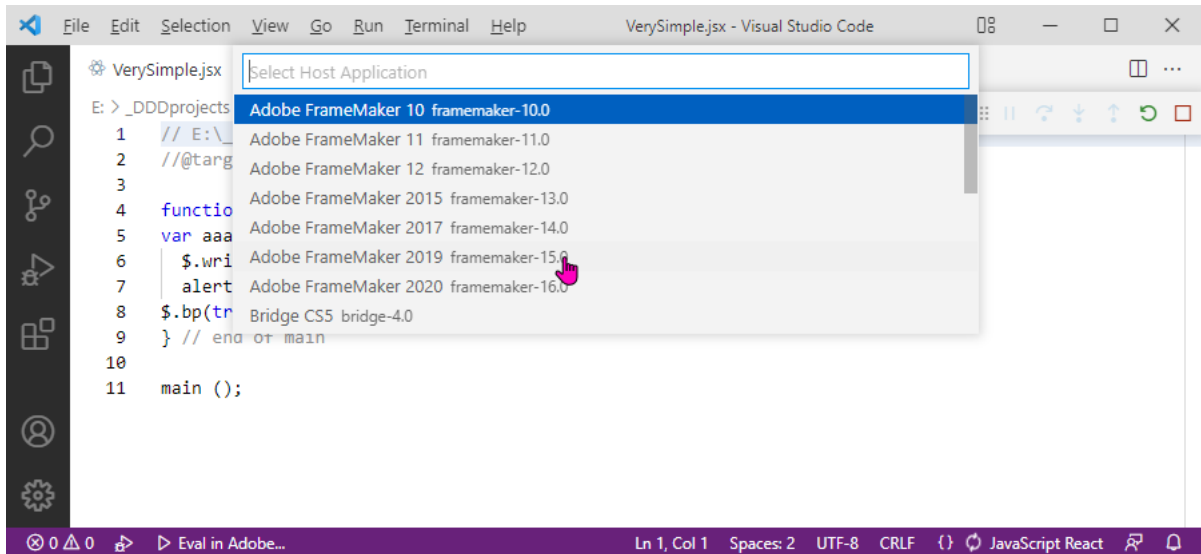
```

File Edit Selection View Go Run Terminal Help
VerySimple.jsx - Visual Studio Code
VerySimple.jsx Select environment
E: > _DDDprojects Chrome
1 // E:\_ Edge: Launch
2 //@targ ExtendScript
3 Node.js
4 functio VS Code Extension Development
5 var aaa Install an extension for JavaScript React...
6 $.wri alert ("Hello, here I am");
7 $.bp(true);
8 } // end of main
9
10
11 main ();
Ln 1, Col 1 Spaces: 2 UTF-8 CRLF {} JavaScript React

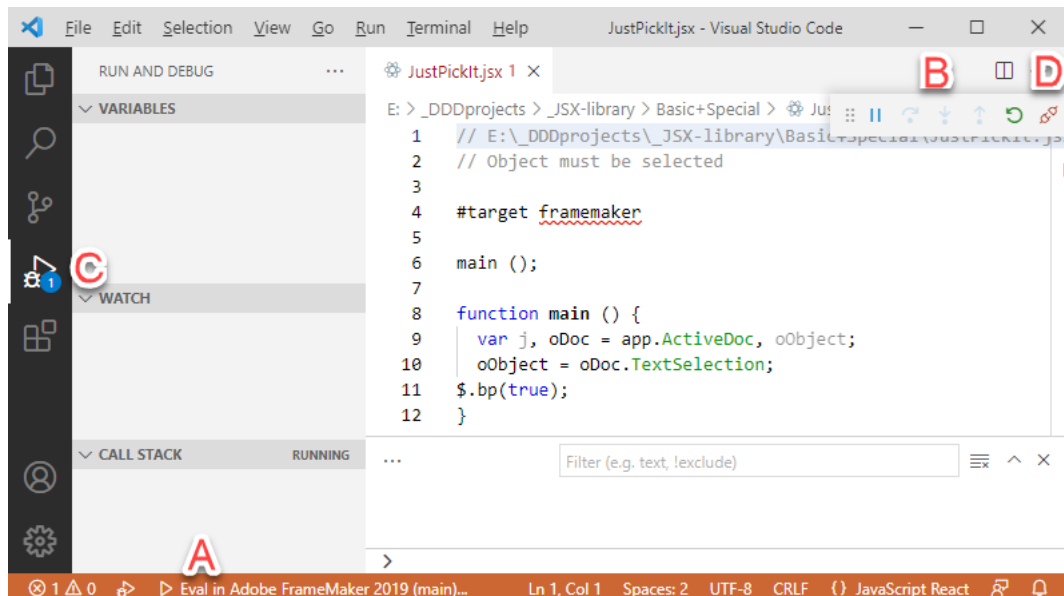
```

<sup>45</sup> My favourite editor is [EditPad Plus](#)

- 4 Having selected ExtendScript I get the pick-list of the valid targets.



- 5 After this selection it takes some time (> 3 sec) until the target is 'connected. If not, use **F5** again...'

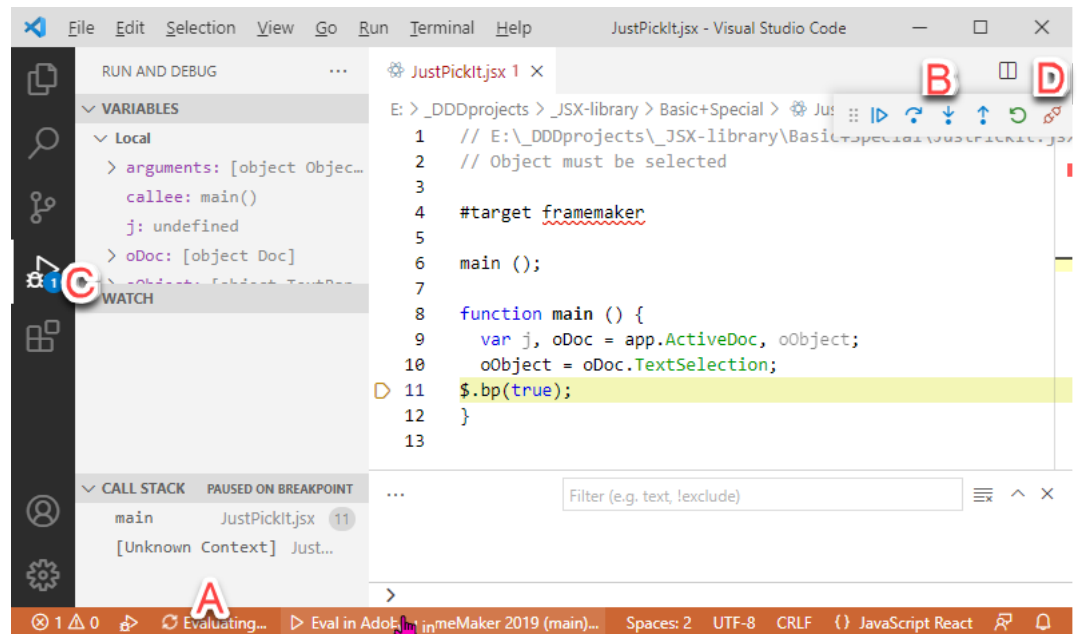


This is indicated by the elements **A** to **C** of the UI. The most right icon in the debugger tool bar **D** identifies as “Disconnect”.

- 6 If you do not have a `$.bp(true)` statement (probably somewhere at the beginning) in your code already, it is time now to set a break point. Otherwise the script will just run away and you have not chance for stepping

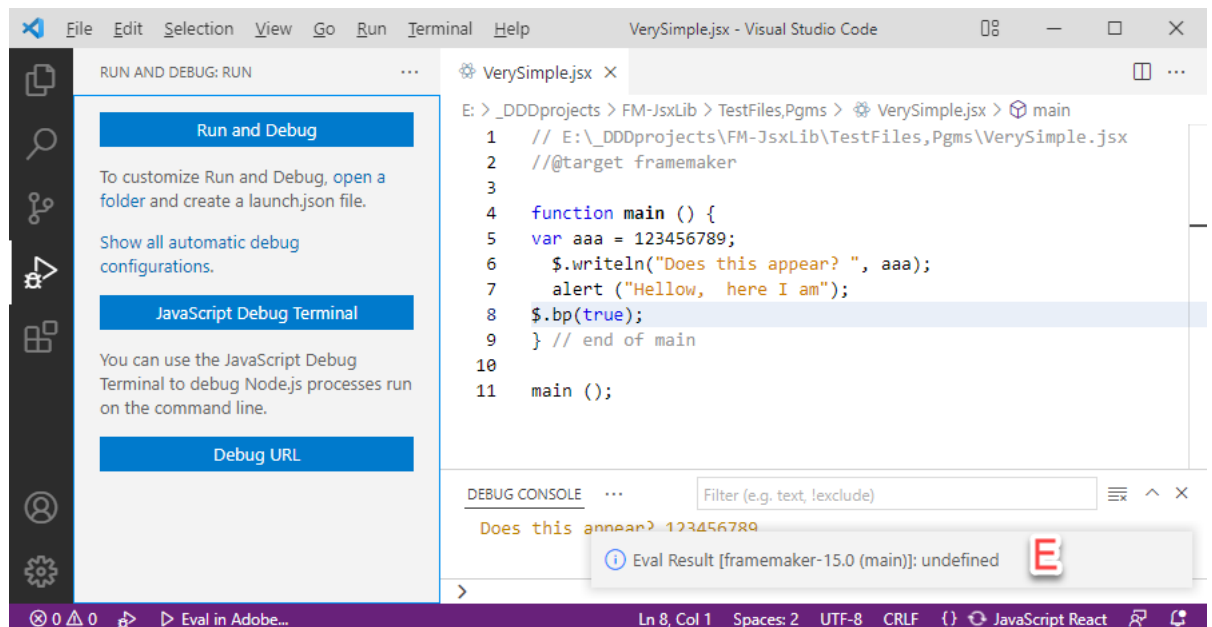
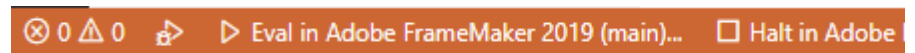
To set a break point in the debugger use **F9** at an appropriate code line. A red dot indicates the break point.

## 7 Clicking **Eval i Adobe FrameMaker ... (A)** immediately runs the script..



## 8 I want to leave before it would end. I use the button far right (**D**) and normally the evaluation terminates (blue i at **C** disappears), which is also indicated by a message (**E**):

You can also stop the debug session by clicking on the button introduced by a square symbol:



Then I can continue work in the target application.

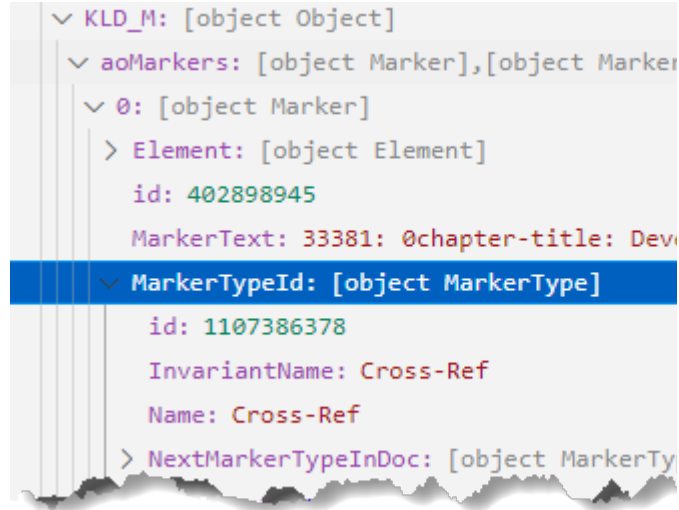
If the script doesn't stop then there is another issue (possibly an infinite loop?). Unfortunately, the extension does not yet have a *Terminate Script Evaluation Process* feature, so if you wind up in this scenario then you will need to kill the process.



## Some experience with VSCode and ES debugger

Starting a script	Multi step procedure, less intuitive than in ESTK.
Close the environment	instantly
Short cuts	Standard assignment
Syntax highlighting	<ul style="list-style-type: none"> <li>Reasonable; properties are displayed in green.</li> <li>Undefined variables displayed in light grey.</li> </ul>
Variable display	<ul style="list-style-type: none"> <li>Local and Global variables have separate list.</li> <li>In the global variables it is possible to dive into the objects<sup>46)</sup></li> <li>Hovering over a variable in the code display shows its value.</li> <li>Undefined variables are listed, but have no type or contents.</li> <li>In code undefined variables are dimmed</li> <li>Under certain conditions the list of local and global variables is exchanged<sup>47)</sup></li> </ul>
Display of object properties	Both in local and global variables properties can be expanded. For array items at least one item must be explicitly used in the code
Display of XML objects (Code)	I have not found a setting to avoid this kind of error-report:
	<code>oMenus.MenuDocu = KLD_M.UItxt.SetupMenus.@menu01.toString();</code>
	@ → Identifier expected
	→ Experimental support for decorators ...

## Visual appearance



46 In ESTK this is not possible (At least for me)

47 Experience during beta phase of debugger.