VBScript Projects in Visual Studio

The debugger in Visual Studio is a very nice tool for debugging VBScripts. Visual Studio will also allow you to have your scripts organized and if you have a version control system like, CVS, Subversion or Source Safe it will allow your to keep track of changes made to your scripts. Visual Studio also provides some Intellisense for COM objects and the basic VBScript functions.

However I have not been able to find a Visual Studio Project Template to create VBScript projects. Some people suggest setting up a custom build action, but as far as I have searched, nobody has actually tryed it. Therefore I have written this blog to share my findings.

As there is no template for VBScript Projects, I have tryed to find the most basic template provided by Visual Studio. This is the Makefile Project. Now choose new Project from the menu, and create a project as shown in figure 1:

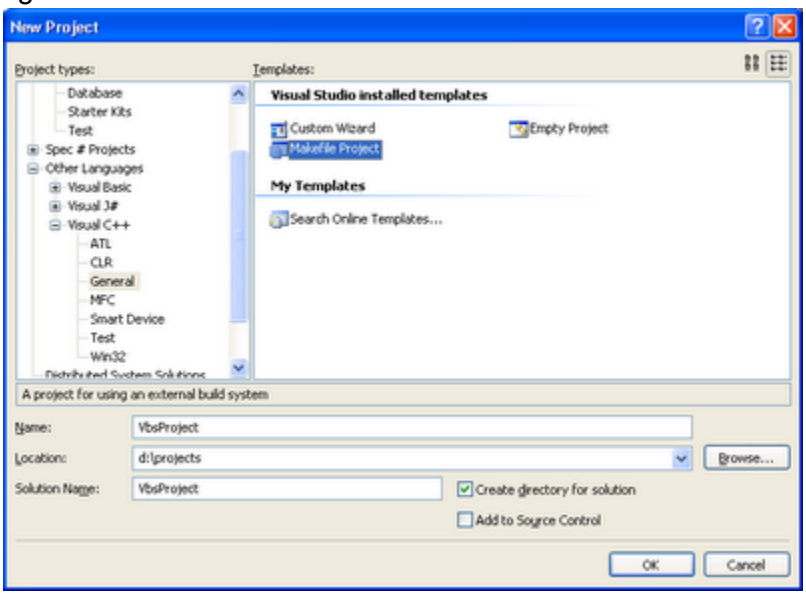

Figure 1: Create a Makefile Project.

The wizard will pop up and allow detailed configuration of the project (Figure 2), however we do not need it, so just click Finish.

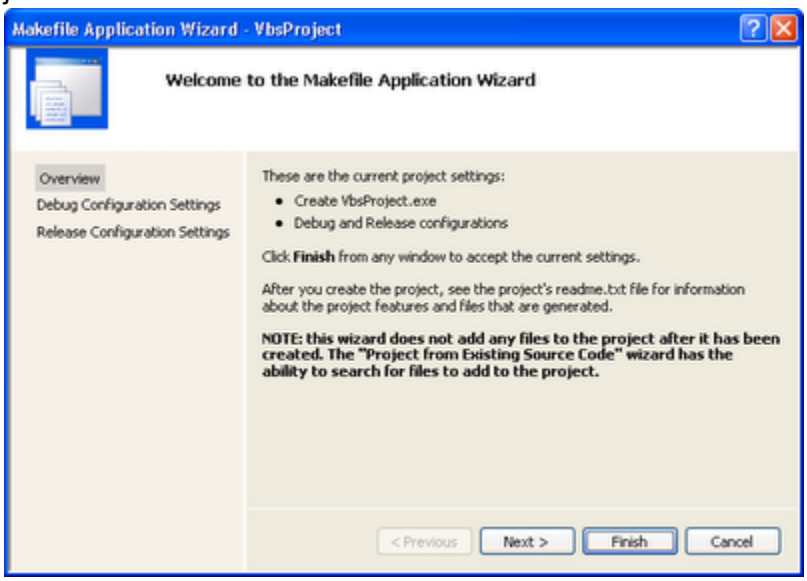

Figure 2: Wizard for the Makefile Project.

The template has created 3 folders for typic C++ projects and a readme.txt file as shown in figure 3. We don't

want any of those so delete them right away.

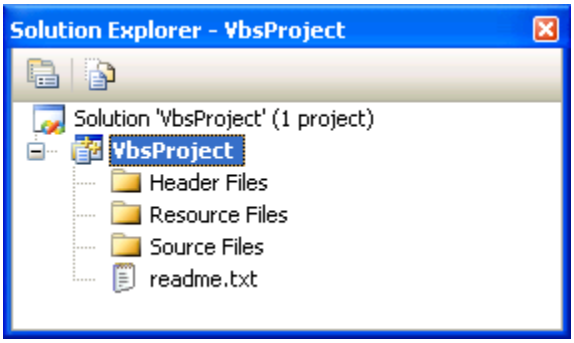

Figure 3: Default folders and files in the Makefile project to delete.

Now add a new VBScript file to the project as shown in figure 4.

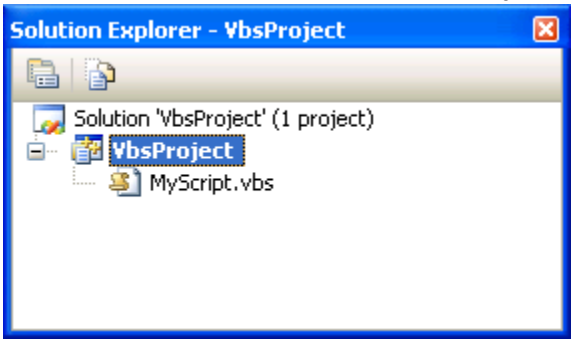

Figure 4: View of solution with the new vbs file added.

Now right click the Project and choose Properties. This will open the Property Pages dialog shown in figure 5. Choose the General Page, and change the "Configuration Type" from "Makefile" to "Utility".

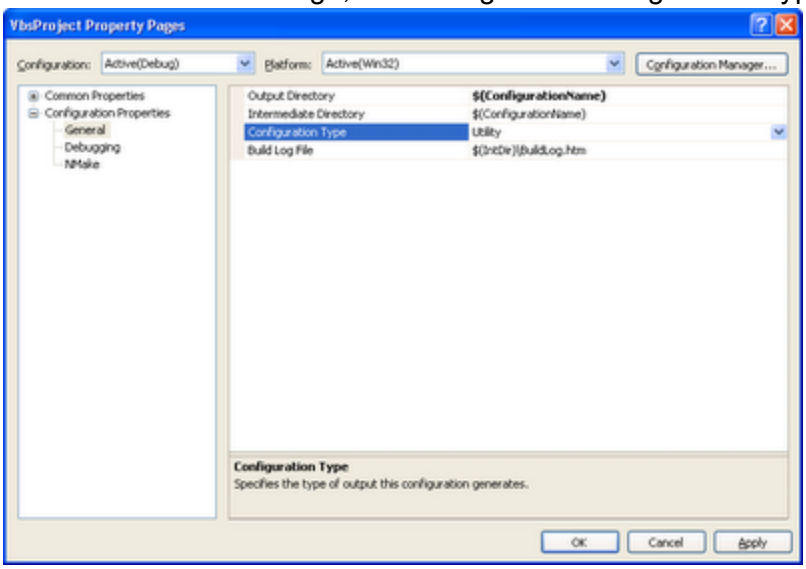

Figure 5: Change the Configuration Type to Utility

Next choose the "Debugging" Page and set the "Command" field to "cscript.exe" and set the "Command Arguments" field to "//X MyScript.vbs". This will fire the command "cscript.exe //X MyScript.vbs" which will execute script in debug-mode.

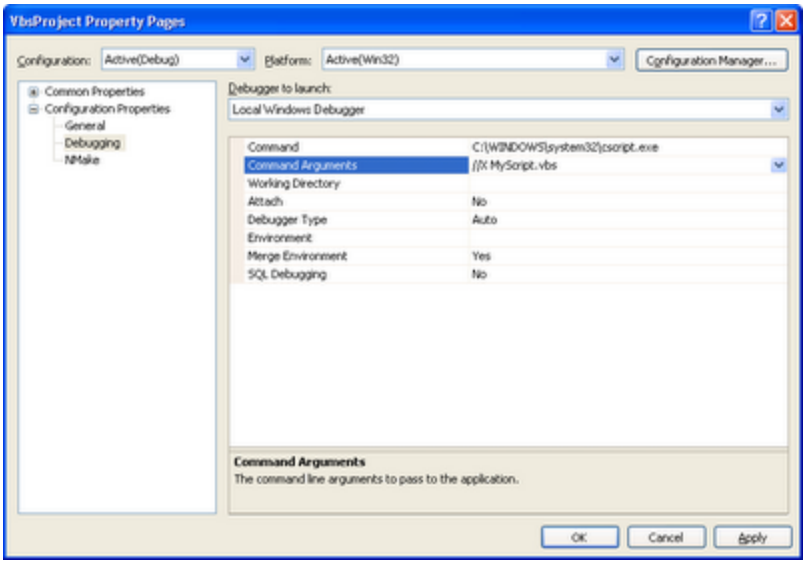

Figure 6: Set the actual command to run when debugging the script.

Now add the code shown in Figure 7. Note that you have some Intellisense if you press CTRL+Space.

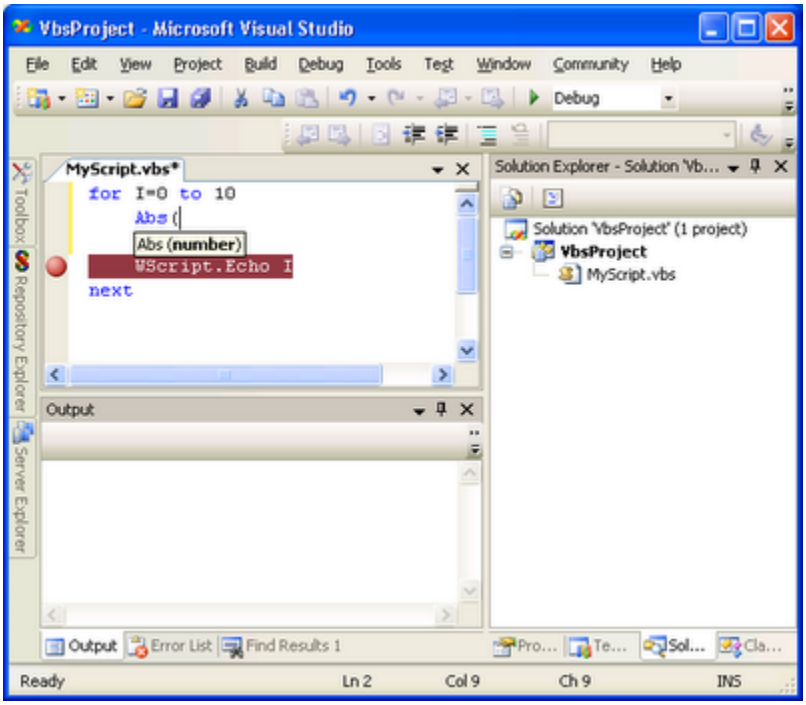

Figure 7: Here you can see that the Intellisense feature works.

Now press Ctrl+F5 for "Start without Debugging" to launch the project. The reason why it is to be started with out the debugger is because Visual Studio will try to attach to the cscript.exe process, however it is the underlyling script that we want to debug.

As we have set the parameter //X the scripting engine will throw a "Script Breakpoint" exception, which the Visual Studio Just-In-Time Debugger will catch and ask you what to do about it, by showing you the dialog in figure 8. Select the current Visual Studio instance and press "Yes".

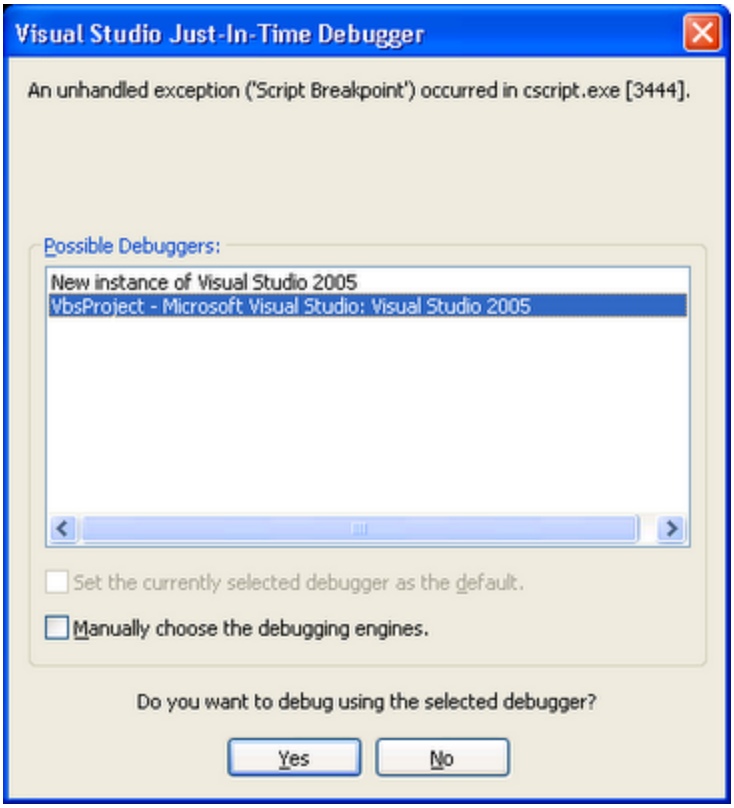

Figure 8: Visual Studio JIT debugger.

Now Visual Studio will attach to the script and you will be able to debug it as you are used to. On Figure 9 you can see that I have placed a breakpoint which I have hit. You can also see the value of the variable "I".

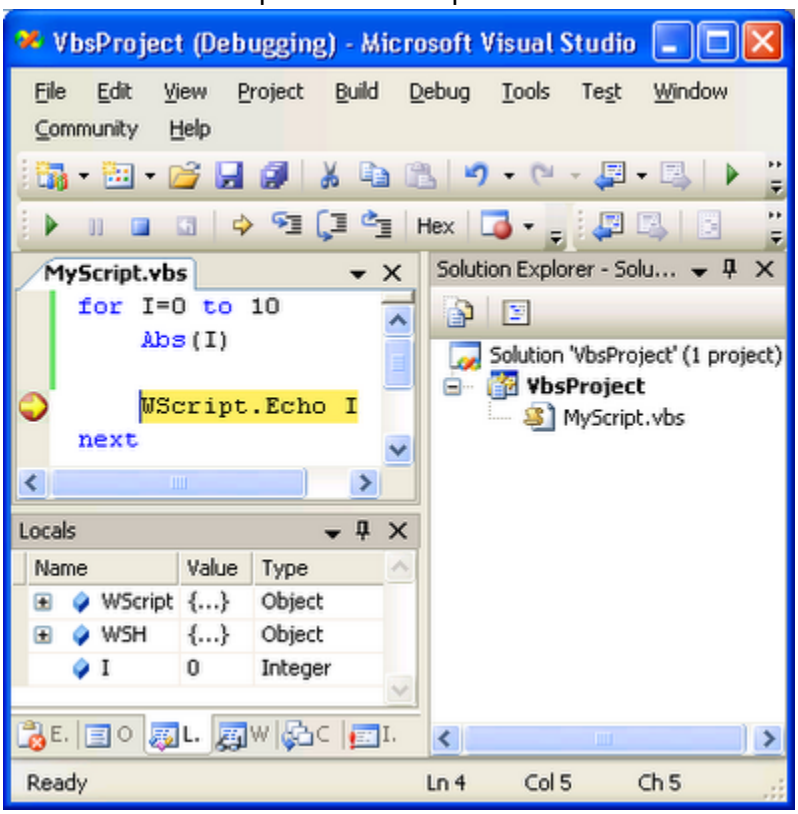

Figure 9: View of the script being debugged.

This is my first shot a making a VBScript project in Visual Studio. After having done this you can experiment with different setup.

As you will soon discover you will probably want to have more scripts inside the solution. This will make it neccesary to change the project configuration everytime you switch script. I suggest either having more

configuration (like debug and release) where you specify the script to run, or to setup a custom build action for each script. Having a custom build action will require you to press CTRL+F7 to compile the active file that you are editing. If you select "Start Without Debugger", the build actions for all files will be activated, and you will be asked to select a debugger for each script.