Debugging Workflow Agents With Visual Studio .NET

Application Note

Date	June 21, 2012
Applies To	Kofax Capture 8.0, 9.0, 10.0
Summary	This application note provides the information needed to step through a .NET-based Workflow Agent during execution.
Revision	1.0

Overview

When developing Workflow Agents (WFA), it is often desirable to step through the code while it is running. This provides the ability to examine the actions taking place, along with values being used, in the code during execution. This information can be invaluable during development, or when troubleshooting any issues that may arrive at a later time.

This application note contains information on the requirements, preparation and basic steps a developer can use to step through the source code for a Workflow Agent using Visual Studio .NET. This application note specifically deals with the debugging of Workflow Agents created with Visual Studio .NET 2008 or later.

Requirements

To implement the following procedures, you must have a Visual Studio .NET development environment installed on a machine with a Kofax Capture Stand-Alone installation, a Capture Client installation or a Capture Server installation with client applications installed. Also necessary is a batch containing documents that are ready to be processed for the specific module (queue) to which the Workflow Agent will be attached.

Compiling with Binary Compatibility

Previous versions of Kofax Capture made use of Visual Basic 6.0 for development. In these cases, it was recommended to set the Source Code project for Binary Compatibility.

In Visual Basic .NET, Binary Compatibility is accomplished through attributes giving a developer direct control over the information placed in the compiled component, such as class and interface identifiers. As a consequence, Binary Compatibility is no longer an issue.

Debugging with .NET Framework 4.0

Capture was created using Visual C++ and Visual Basic 6 as well as .NET Framework 2.0 (Capture 8.0) or .NET Framework 3.5 SP1 (Capture 9.0 and 10.0).

Because of this blend of Managed (.NET) and Unmanaged (Non-.NET) components, some modules, such as Administration, Validation and Export, are "Unmanaged", and will run against the latest version of the .NET Framework installed, which can be .NET Framework 4.0.

Kofax Knowledge Base Article QAID 16107 addresses this issue and provides a solution. Use the link below to see this Article.

http://knowledgebase.kofax.com/faqsearch/results.aspx?QAID=16107

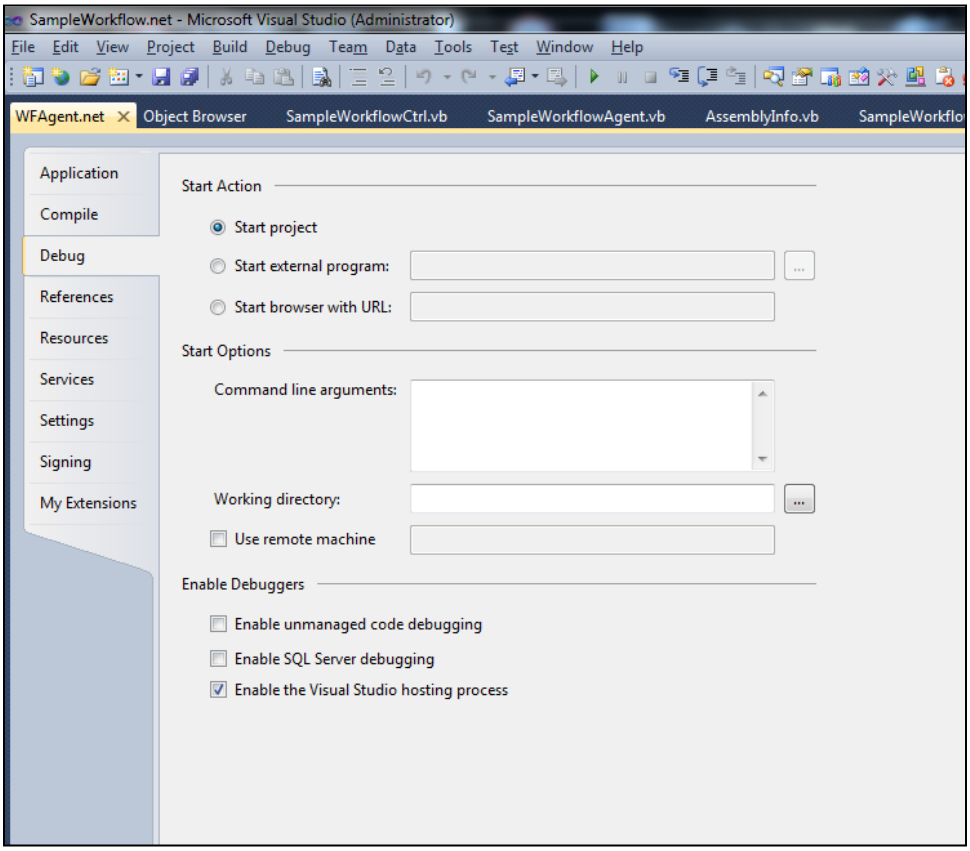
Stepping Through the Code

The most flexible method of examining what is happening in a Workflow Agent is to step through the code during execution. The first step is to load the source code for the Workflow Agent in Visual Studio .NET.

The screenshots in this application note are from the sample Workflow Agent provided with Capture.

Setting the Workflow Agent to Launch Into Debug Mode

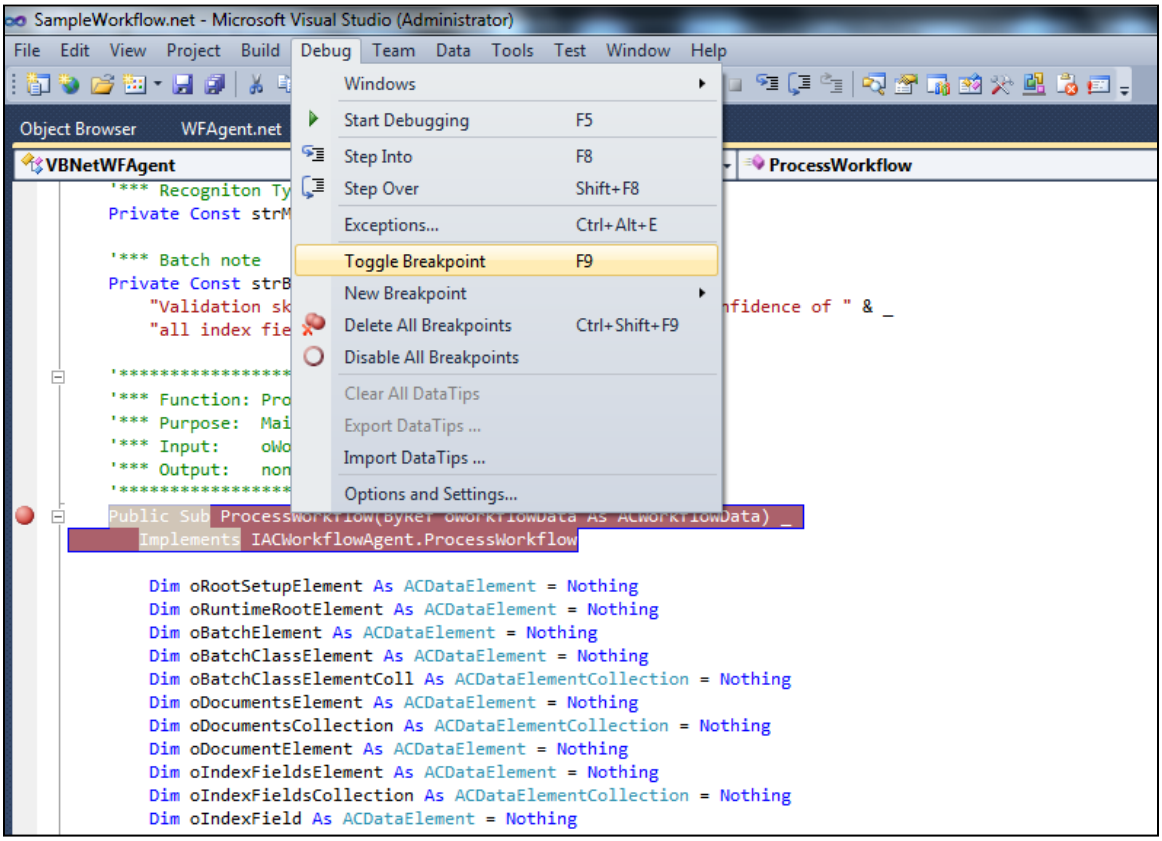
It is not possible to set the Workflow Agent to automatically launch into debug mode in the same manner as a Custom Panel or Custom Module. The Project's properties, "Properties…", selection available under the Project pull down menu (Project ¦ Properties...) must be set to the "Start Project" option.



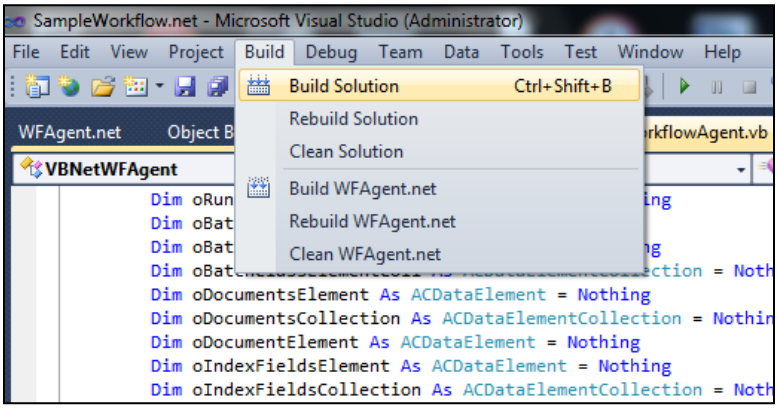
Setting the Breakpoint

In this example, a breakpoint is set in the ProcessWorkflow() function with the first conditional “If… Then…” statement looking at which queue (module) is being executed.

The breakpoint can be activated by clicking the left hand margin of the code window on the line you wish to set it, or by selecting "Toggle Breakpoint" from the Debug menu or pressing F9.



After the breakpoint has been created, build the project from the Build menu.



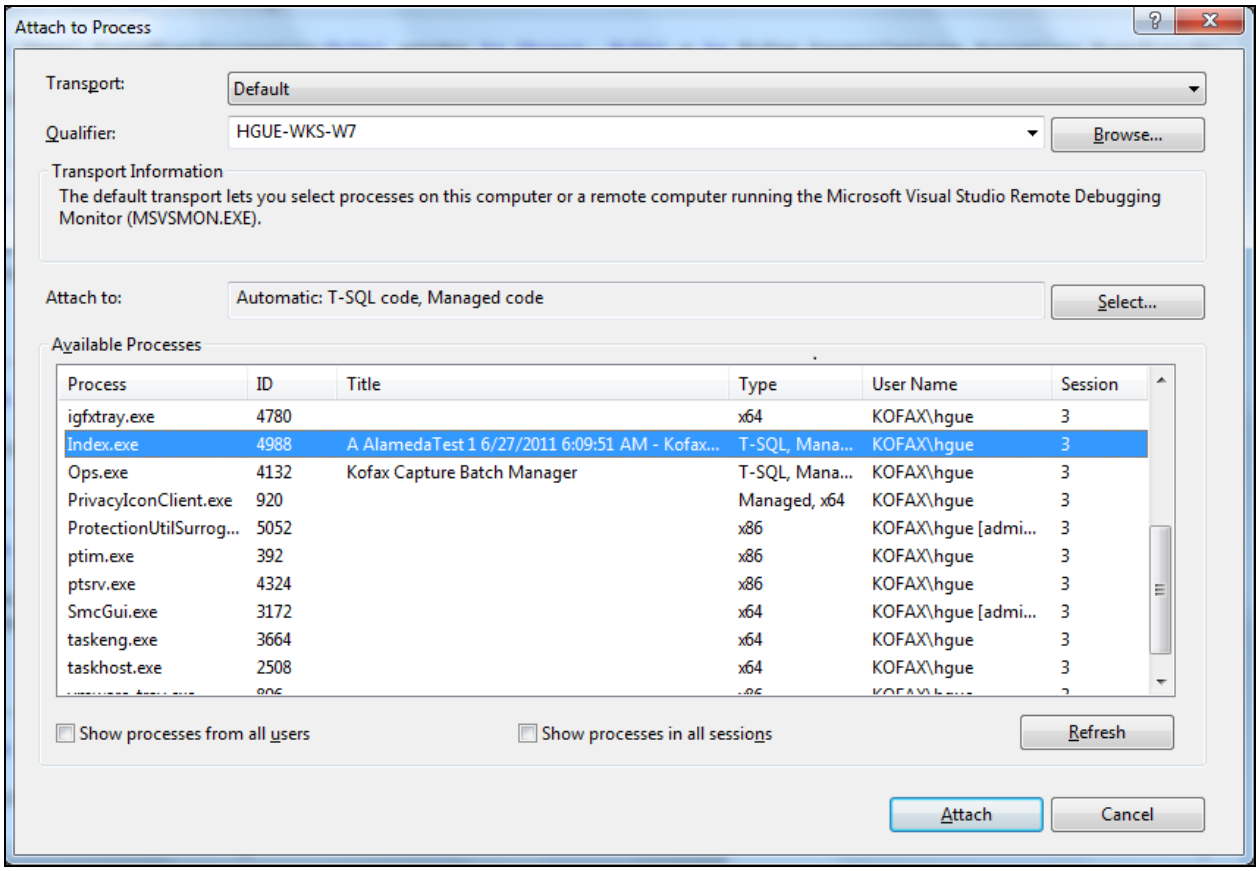
Stepping Through the Code

Process the batch up to the selected module. This can be done by either launching the module directly or by selecting a batch in the Batch Manager and clicking the process button. Visual Studio .NET is now ready to begin the debugging process and step through the code.

Attaching to an Existing Process

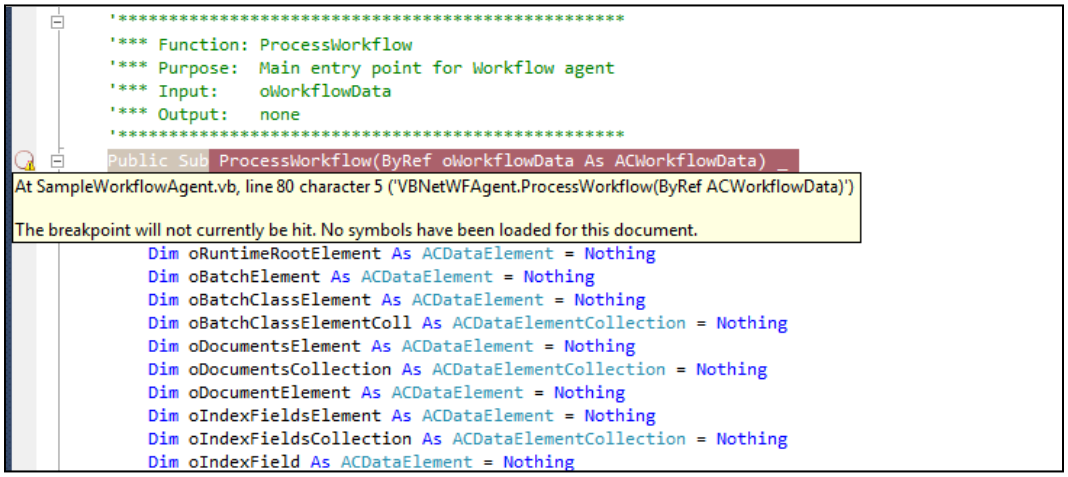
The project will need to be attached to this process to start debugging. This can be done from the Processes dialog box. This is available under the Debug menu (Debug ¦ Attach to Process...) or under the Tools menu (Tools ¦ Attach to Process...).

Select the desired process (module) from the Processes dialog box. For example, if the Workflow Agent is to be checked with the Validation module, select the “Index.exe” process.

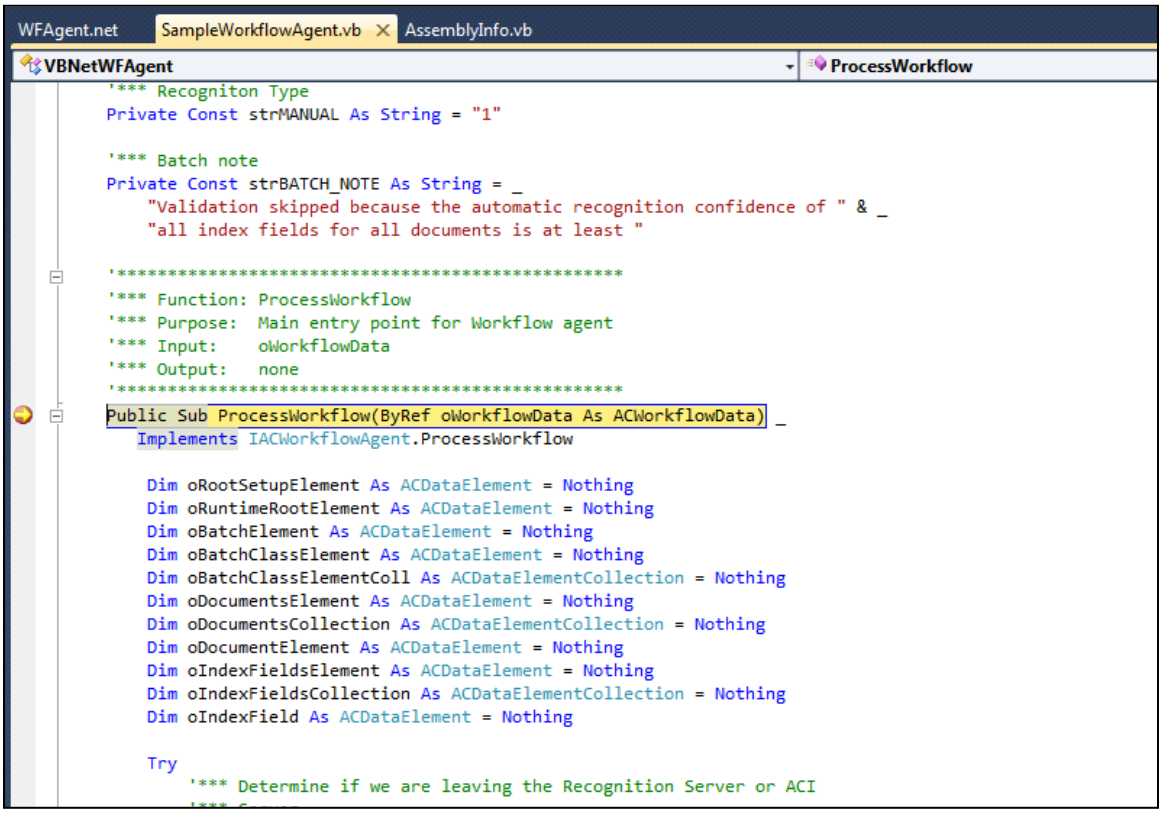


It is not necessary to manually start debugging from the Debug menu, as attaching to the process will put the code into debug mode. In this example, a breakpoint is set on the ProcessWorkflow() function and the code stops at this point.

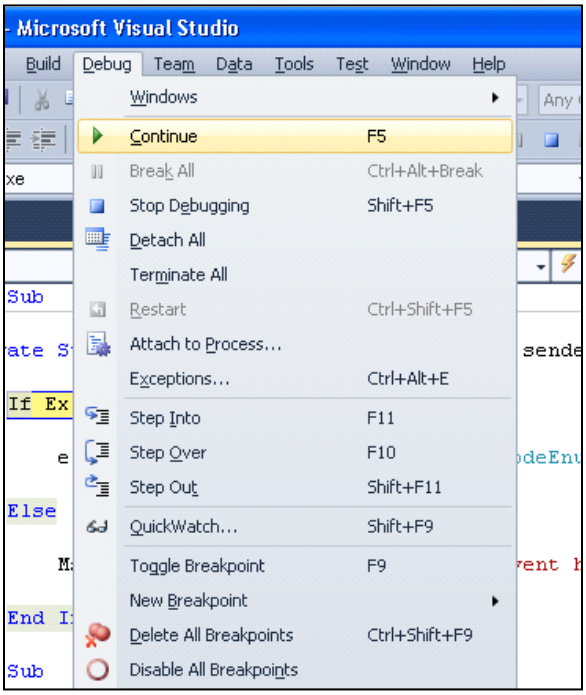
As the Workflow Agent has not yet been started, you will see the message “The breakpoint will not currently be hit.”



When the batch is closed, the module will launch the Workflow Agent and the breakpoint will be reached if it is placed appropriately.



You can now step through the code using the "Step Into" F11, "Step Over" F10 and "Step Out" Shift+F11. Each time the F11 key is pressed, one operation will be performed. This will advance to the next line of code and reveal exactly what code is being run. To continue processing normally, simply press the F5 key and the code will continue to process until the next time a Breakpoint is encountered or the process has been completed.



To stop debugging, simply click the "Stop Debugging" Shift+F5 button.

Summary

Using the techniques noted in this Application Note, a developer can step through a Workflow Agent as it is running to see the actions taking place and the values being used. This will provide the information needed to address any issues such as fixing an error condition that occurs, or modifying the Workflow Agent to handle any given situation as needed.