Debugging Export Connectors With Visual Studio .NET

Application Note

Date	June 22, 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 Export Connector (Release Script) during execution.
Revision	1.2

Overview

When developing and debugging Export Connectors (Release Scripts), it is often desirable to step through the code while a document is being exported.

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 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 the Export Connector using Visual Studio .NET. This Application Note specifically deals with the debugging of Export Connectors created with Visual Studio .NET 2008 or later. If debugging an older Export Connector or Release Script created using Visual Basic 6.0, the "Debugging Export Connectors (Release Scripts) Created Using Visual Basic 6" Application Note should be consulted.

Requirements

To implement the following procedures, you must have a Visual Studio .NET development environment installed on a machine with a Kofax Capture Standalone installation, a Capture Client installation or a Capture Server installation with client applications installed. Also, a batch containing documents from the Document Class making use of the Export Connector and ready to be exported will be required.

Compiling with Binary Compatibility

Previous versions of Kofax Capture made use of Release Scripts built using Visual Basic 6.0. 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

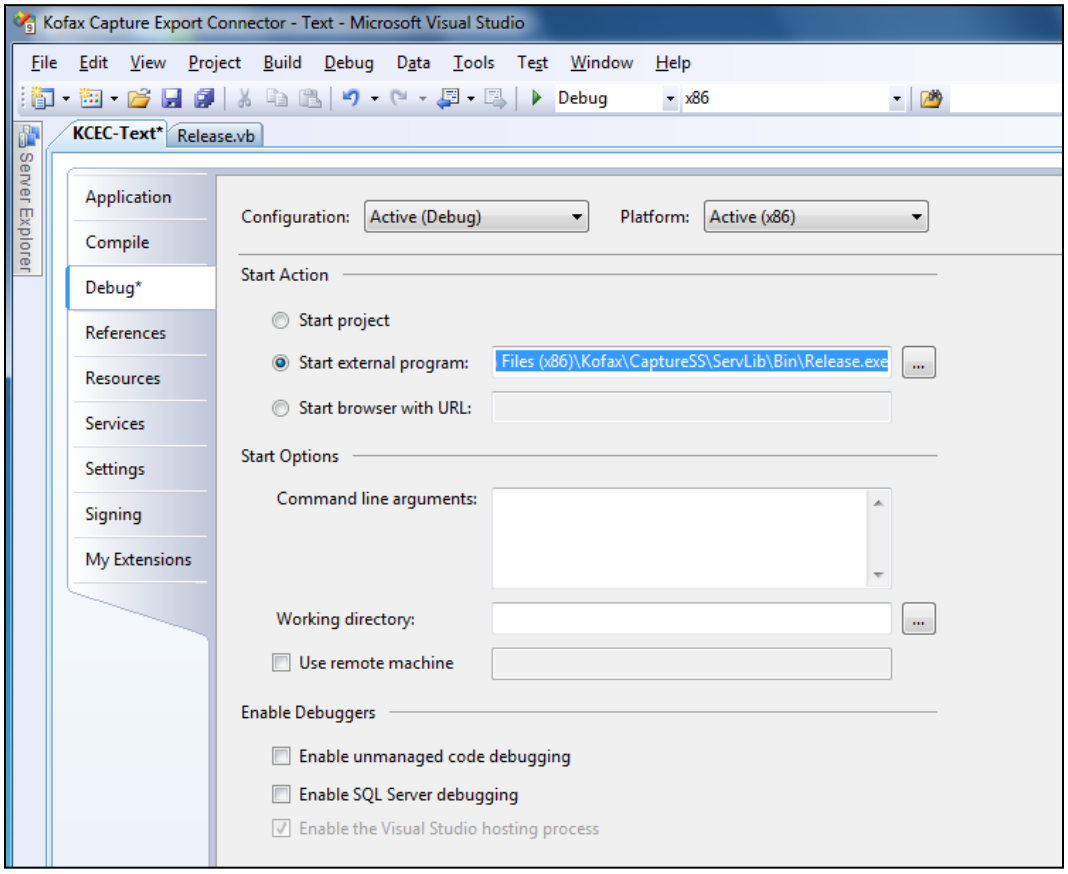
The most flexible method of examining what is happening in an Export Connector is to step through the code during execution.

Setting the Export Connector to Automatically Launch Into Debug Mode

After loading the source code for the Export Connector in Visual Studio .NET, the Export Connector can be set to automatically launch Export. This is set in the project's properties selection available under the Project menu (Project ¦ <Project Name> Properties...).

Under the Debug tab are the debug options available for the project, including the ability to start an external program such as the Export (Release.exe) module.

Great care should be taken to verify that only the desired Export Connector will be run if this option is selected. Other Batches making use of other Export Connectors may raise issues under this scenario.



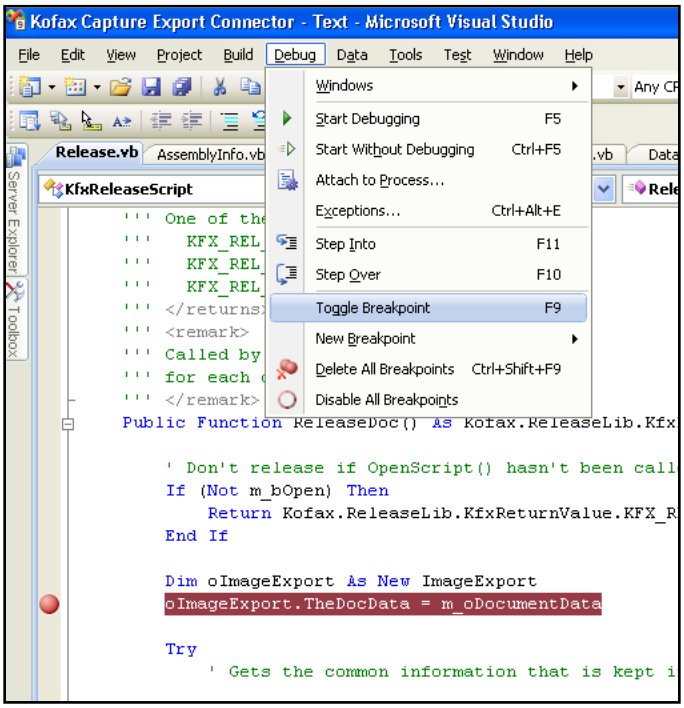
If the Export (Release.exe) process is already running, it is possible to attach to it from the Processes dialog box. This is available under the Debug menu (Debug ¦ Attach to Process...) and under the Tools menu (Tools ¦ Attach to Process...).

Setting the Breakpoint

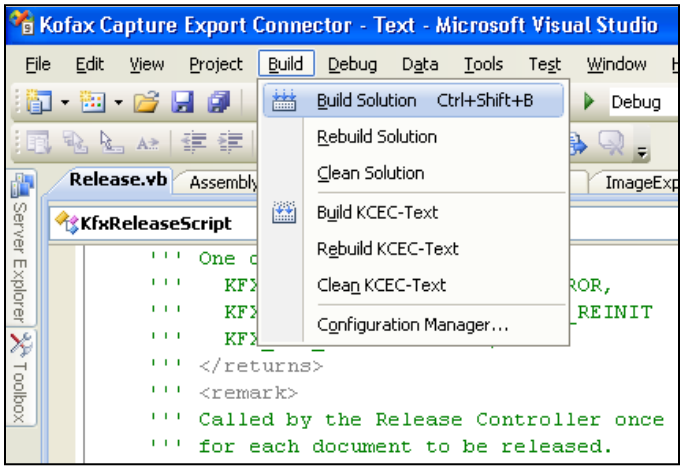
In this screenshot, a breakpoint is set early in the code of the "ReleaseDoc()" function which is found in the Release script class, "KfxReleaseScript", in the "Release.vb" file.

This function is the document's export point and uses the ReleaseData object to export all of the current document's data to the specified external data repository. It is called once for each document in the batch that is to be exported.

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



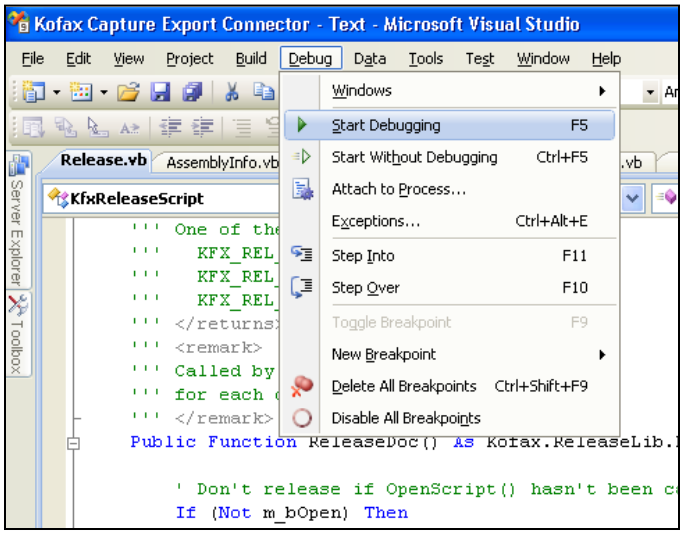
After the breakpoint has been created, build the project from the Build menu.



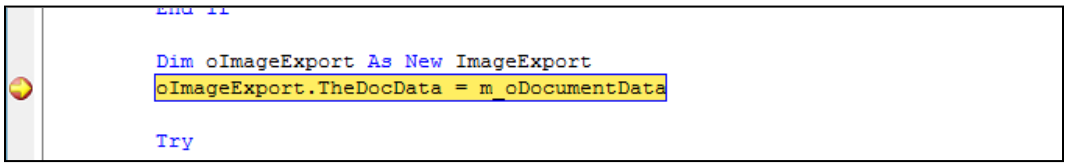
Stepping Through the Code

Process a batch that contains a Document Class that makes use of the Export Connector up to just before the Export queue. This way, the batch will be ready when the developer starts debugging.

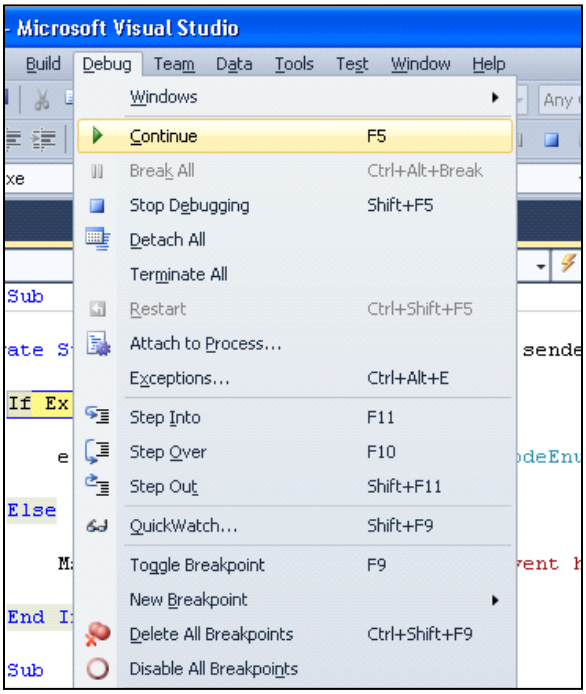
Visual Studio .NET is now ready to begin the debugging process and step through the code. Start the Visual Studio debugger from the Debug menu or pressing **F5**.



The Export module will launch and call the Export Connector to export the document. When the Export Connector process reaches the selected breakpoint, Export will stop and highlight the running line of code.



You can now step through the code using "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 "Continue" **F5** button and Export will continue to process until the next time a Breakpoint is encountered or the export process has been completed.



To stop debugging, simply click the "Stop Debugging" **Shift+F5** keys or select the menu entry.

Summary

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