nicKmiddleweeK
web applications developer
nick@middleweek.co.uk

4D and IIS Integration.

"This document outlines an alternative approach to using the 4disapi.dll and how you can use it to integrate 4th Dimension and IIS. A zip file accompanies this document containing the sample 4D database, and the PowerPoint presentation used at the 4D UK Developers Conference, 2001."

For those of you that don't know, IIS is the Web Server built into the Microsoft Windows operating system. The key behind this integration of 4D and IIS is the 4disapi.dll. So if you've ever wondered what the file is and how it can help you, you should know by the time you've read this article. I will outline how to set up the 4disapi.dll and any configuration changes needed to IIS. This article demonstrates a different approach to using the 4disapi.dll than what has been previously documented.

The term 'ISAPI' stands for Internet Server Application Programming Interface. It's really a way to develop extensions for the Microsoft Web Server or any other application that supports the ISAPI interface. To help you understand how it fits in you could compare it to writing a plug-in for WebSTAR or 4D. Essentially the 4disapi.dll lets IIS communicate with 4D, passing over the full HTTP request from the browser.

Using the 4disapi.dll will improve the overall performance of your web applications. You will minimise the number of web hits and overall processing time on 4D. This is because you are only asking 4D to serve the html or skeleton of the dynamic web pages. All other static objects, such as gifs, jpegs, applets, external .css and .js files, etc are served by IIS. This let's 4D get on with its primary role, to be a database and serve data to its clients, and IIS now being a type of client to 4D requires its data in html!

Not only do we get a performance gain, we also inherit the features and functionality that IIS has to offer. We now have built in ftp, support for WebDAV, greater mime type support and larger file handling. Apart from that, Microsoft IIS is often the chosen web server by many companies and we now have a way for *our* chosen and preferred database environment to be used and work in harmony with any new existing web services.

This separation of the web server from the database allows us to improve the security of our web applications. Picture a typical company network with an Internet connection and a firewall separating the LAN from the WAN. Lets assume there's a 4D Client/Server application being used by the internal staff. A requirement has developed to allow home workers or clients to gain access to certain data or screens; the chosen solution for this remote connectivity is to use a web browser.

If we were to give the web browsers direct access to the 4D Server machine, we would have to open a 'public-hole' in the firewall on port 80, exposing that computer on the LAN to the Internet. Using the 4disapi.dll you can have an IIS web server on Internet side, the WAN. You can then open a 'hole' in the firewall on a port of your choice and restrict the access to only the IP address of the IIS web server. This prevents public computers from accessing the LAN.

Using the sample 4D application

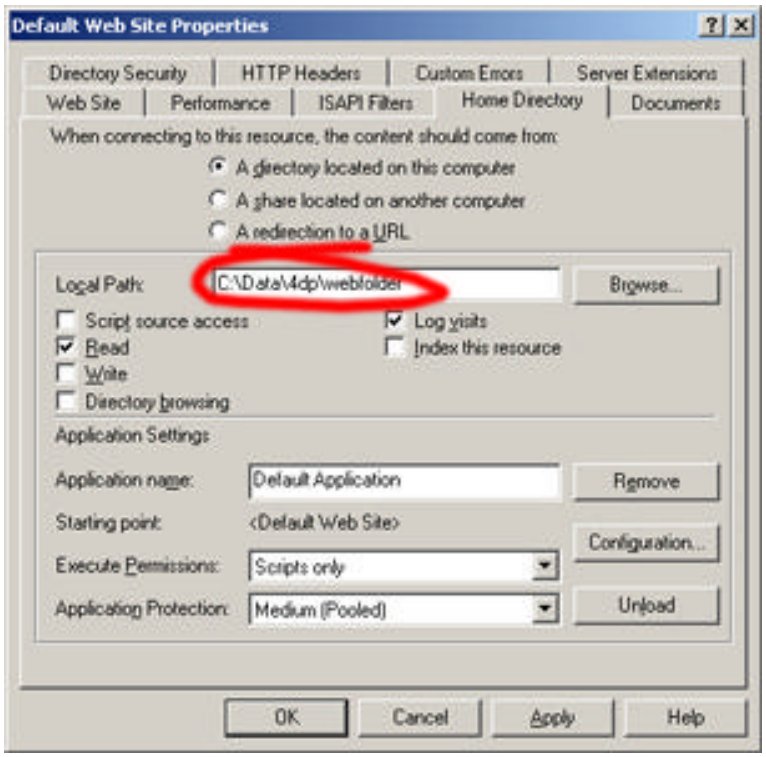
As I demonstrated at the 4D UK Dev Con, there two main ways in which you can integrate 4D and IIS. The latter being my preferred and I believe is the most sensible choice as it gives you a more flexible approach to creating your web systems. You can develop your application using 4D as the web server and then deploy the working solution using IIS as the web server and 4D as the data source.

"This demonstration can only be used on the Windows platform and uses 4D version 6.7.x"

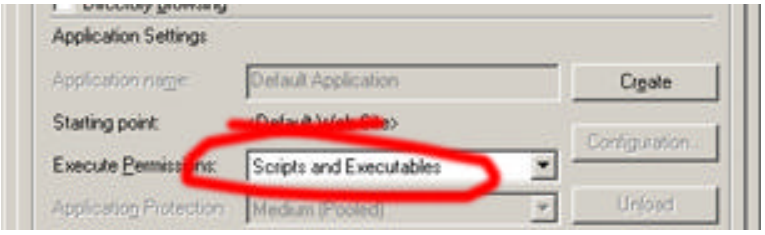
1. Download the accompanying 4dp.exe file and unzip it to a folder of your choice.
2. Open the structure with 4D 6.7.x. To check the web server is running simply connect to it with a browser, using port 8080. e.g. http://127.0.0.1:8080/
3. You should see the home page. Clicking the link will generate an error from 4D because it doesn't recognise the /scripts/4disapi.dll/ part – this is for IIS to use.

4. Assuming you have IIS already installed on your Windows machine, open the IIS console from *Control Panel* > *Administrative Tools* > *Internet Services Manager*. Right click the web site and choose properties. Select the Home Directory tab and set the Local Path (Web Folder) of IIS to the Web Folder that 4D uses.

   N.B. You can compare the IIS Console to the WebSTAR Admin application.



5. Select the Web Site tab and check IIS will listen on port 80 when started. Click OK and start the IIS Web Server (if not already) by right clicking the web site and choosing Start. *"At this point, you will have two web servers running on your machine. 4D is listening on port 8080 and IIS is listening on port 80 - the standard. They are both using the same web folder."*
6. By default there is a scripts folder for IIS. This is located at C:\Inetpub\Scripts\. The scripts folder serves the same purpose as a cgi or cgi-bin folder. To use the first examples you will need to make sure it has execute permissions. In the IIS Console, right click the Scripts folder, choose properties and check the Execute Permissions is set to "Scripts and Executables".



7. Copy the 4disapi.dll and 4disapi.ini files into this Scripts folder. You shouldn't have to change the default .ini file settings.
8. Type http://127.0.01/ or http://localhost/ into your browser. This connects to IIS and serves the same home page as you previously saw. This time, there has been no interaction with 4D.
9. The URL on this home page, when clicked, will send a request to IIS for the 4disapi.dll. IIS will load the DLL (if not already) and the 4disapi.dll sends an HTTP request to 4D. 4D then processes the request and sends the HTML back to the browser via IIS. This isn't the most exciting demo but it demonstrates a simple way to integrate IIS and 4D. Click the Demo 2 button.
10. Demo 2 shows how you can use the same approach but within html forms. You can see that when you submit the form, the 4disapi.dll is invoked but it's passed different parameters, which are used by 4D, when it receives the full HTTP request.

11. 4D sends back an html page containing an updated record count of the [CONTACTS] table and a button to take us to the next demo. Click the demo 3 button.

"N.B. Before we go any further you'll need to make a small change in the IIS console to use this new approach to integrating Microsoft IIS and 4D."

nicKmiddleweeK
web applications developer
nick@middleweek.co.uk

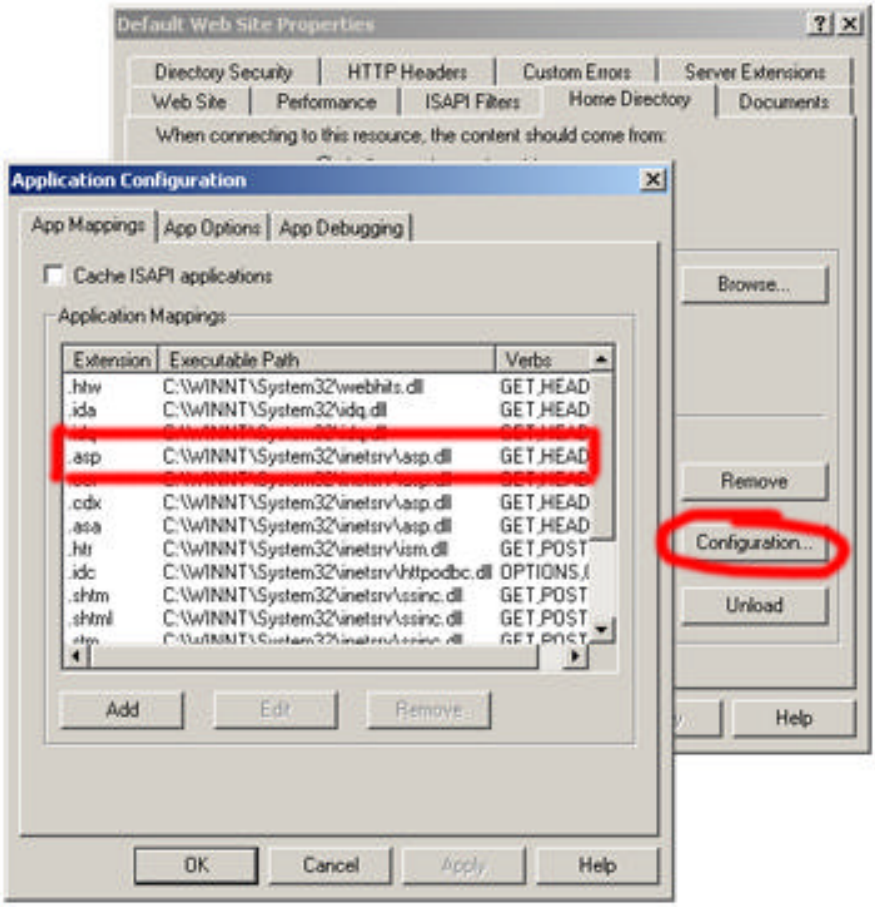
We will configure the 4disapi.dll as an ISAPI Extension in IIS. By doing this, we improve the security of the web server, we make the web application more flexible, we no longer need the ugly /scripts/4disapi.dll/ prefix in our URLs and it makes 4D and IIS more tightly integrated.

Microsoft's Active Server Pages (ASP) uses the same technique. You may have noticed the .asp file extension in URLs when browsing various web sites. The way ASP is integrated to IIS is by using the ASP.dll.

So what actually happens when IIS gets a request for an asp page?

IIS receives the request. It detects that the URL ends in .asp and hands over the request to the ASP Engine (ASP.dll) using what's called an ISAPI Interface. The ASP.dll processes the request, generates HTML in its buffer and sends the HTML back to the browser via IIS.

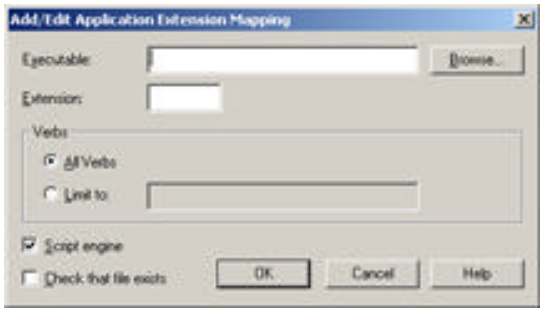
The next couple of steps show how you can use the 4DISAPI.dll to get IIS to hand over **.4dp** requests to 4th Dimension in the same way IIS hands over .asp requests to the ASP Engine.



12. Right click the web site in the IIS console, choose properties and click the Home Directory tab.
    In the Application Settings section, click the Configuration button.
    This will display all the application mappings.

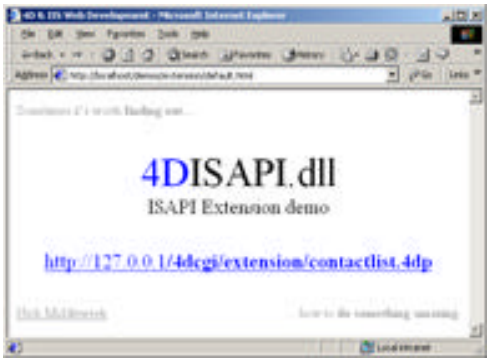
13. You can see in the list, the Application Mapping for .asp. There are others listed such as .shtm and .shtml which are server side include (ssi) files. You can see that the bcation of the asp.dll and some others are nested inside the \winnt\system32\inetsrv\ folder. You can at this stage move the 4disapi.dll, and it's .ini file to this same location if you want to. Any instructions though that follow will assume you have left it inside the Scripts folder.

14. Click the Add button to create a new Application Mapping. Click the Browse to locate the 4disapi.dll and type ".4dp" in the Extension field. Limit the verbs to "GET,POST,HEAD". Select Script engine and deselect Check that the file exists. Finally click OK.



15. In the previous screen there's a "Cache ISAPI applications" checkbox. For performance reasons, make sure this is selected. This prevents IIS from having to keep loading the DLL each time it needs it. Click OK, and OK again to close all other dialogs.

16. Back to our browser. You should have a hyperlink with the extension .4dp ready to be clicked. When you click the link it sends a request to IIS. IIS knows by the extension it needs to hand over this request to 4th Dimension via the 4DISAPI.dll. One thing you will be pleased to see is the /4disapi.dll/ prefix has disappeared. This means you can test out your web systems using 4D as the web server and then use the IIS and 4D combination in the live environment.



Let me explain what is actually happening when you click the .4dp link. IIS receives the request for a URL. Because the URL ends in .4dp, IIS knows that it needs to hand over this request for further processing because it's not just a static html page. From the Application Mapping entry you set up

earlier IIS passes this full request overt to the 4DISAPI.dll, which forwards on the web request to the 4D Web Server on a different port (8080). It is possible to have 4D on a different machine, port, behind a firewall, etc to secure your environment further.

So 4D gets the original web request that IIS received. Your existing 4D code processes this request as normal, there are no changes needed. You then build any arrays, variables, selections, etc and SEND HTML FILE or any other method you choose. This HTML is then sent back to the browser via IIS.

17. Clicking on a record in the list displays it's input form. Using the same .4dp approach for the form action will send a request to IIS, then onto 4D for processing and finally 4D sends back the html to browser via IIS.

<u>Conclusion</u>

Integrating 4th Dimension with new and existing Microsoft IIS web systems is now even easier with the 4DISAPI.dll. It gives 4D a new lease life in the world of Internet development. You can now have a totally integrated solution that gives you an even greater advantage over your competitors. With other connectivity options available to 4D, we can use the 4DP technique as a middleware solution connecting to other data stores such as Microsoft SQL Server and Oracle, replacing other non-compiled middleware technologies such as Active Server Pages and Cold Fusion.