

The VS7 Debugger doesn't work. What can I do?

By mkpark@microsoft.com (Visual CSharp Debugger QA team)

Introduction.....	1
ASP.NET Debugging.....	2
Message: Unable to start debugging on the web server.....	2
Message: You do not have permission to debug the server.....	3
Message: Server side-error occurred on sending debug HTTP request.....	4
Message: The project is not configured to be debugged.....	5
Can start debugging without error message, but breakpoints are not hit.....	6
Message: The debugger is not properly installed.....	6
Message : The server does not support debugging of ASP.NET or ATL server applications.....	7
Message: Access is denied. Verify that you are an administrator or a member of	7
Message: Could not start ASP.NET or ATL server debugging.....	8
Message: Access is denied.....	8
Can't debug with an included File	9
After changing your password, you need to log off/log in for ASP.NET debugging.....	9
After installing Windows2000 SP4, ASP.NET debugging doesn't work and it says "Access denied".....	9
Can hit breakpoint only when the page is loaded first time.....	9
Need to share web server for debugging but I don't want to let other users to be admin on my machine.....	9
General Debugging	10
Message: Unable to start debugging. Unable to start program	10
Message: Unable to start debugging. Access is denied	10
Can start Managed debugging, but PDB is not loaded. So, can't hit any Breakpoints at all.....	10
Managed debugging is not working.....	11
Managed debugger doesn't respond	11
Stepping with C# code is not correct... ..	11
Causality debugging: Stepping between web service client and web service.	12
Can't set from web service client into web service.....	12
After enabling impersonation of web service, can't do causality stepping.....	12
Debugger hangs	13
Remote debugging	13
Can't see any processes on a remote machine	13
Can't connect to a remote machine because of RPC issue	13
Remote debugging fails with machines on a work group.....	14

Introduction

During the last several months, I have dealt with many users from both inside and outside of Microsoft who have debugging issues. I noticed that there are many common mistakes and problems that can be solved, if users are provided with proper diagnosis. Hence, I've

written this document to provide you with this information which will help you if you run into any issues while using the debugger.

This document contains:

- Error message dialog or description of error situation
- The causes for error
- How to fix the problem.

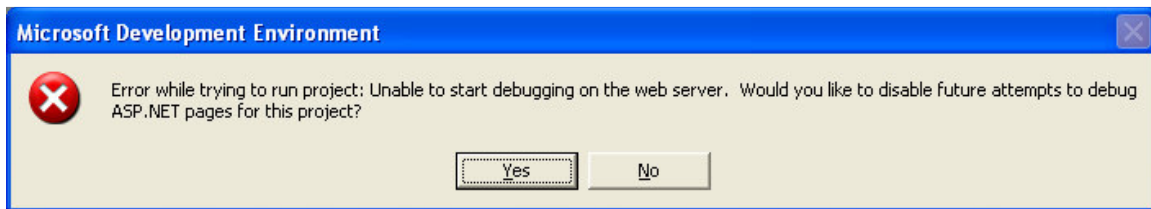
And I especially thanks to VCS debugger team and other people who helped me to complete this document with various feed back.

Debugging issues

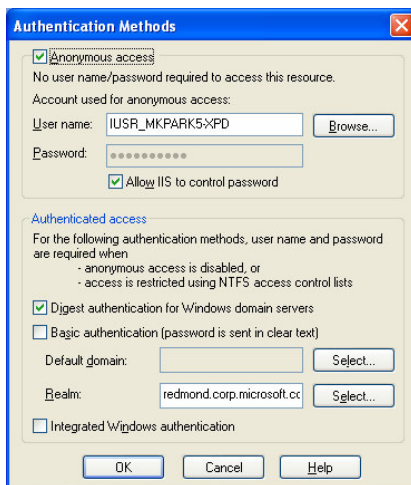
ASP.NET Debugging

***If you can't find the error message that you're looking for in this section, please check the section which deals with general debugging issues or remote debugging issues.**

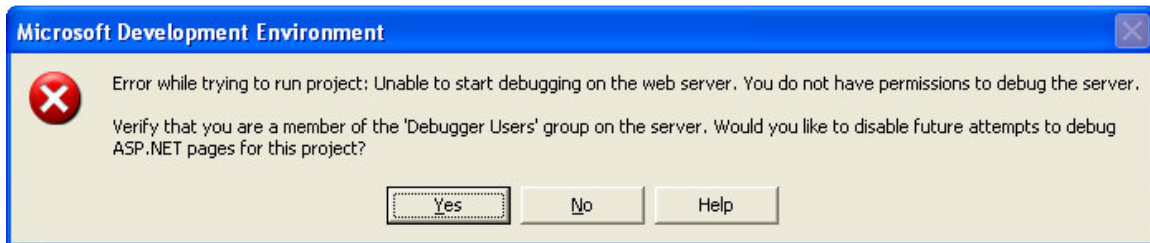
Message: Unable to start debugging on the web server



Your IIS application of IIS is not configured to use “integrated Windows authentication”. Please make sure that the “integrated windows authentication” checkbox on the “authentication method” dialog box is checked.

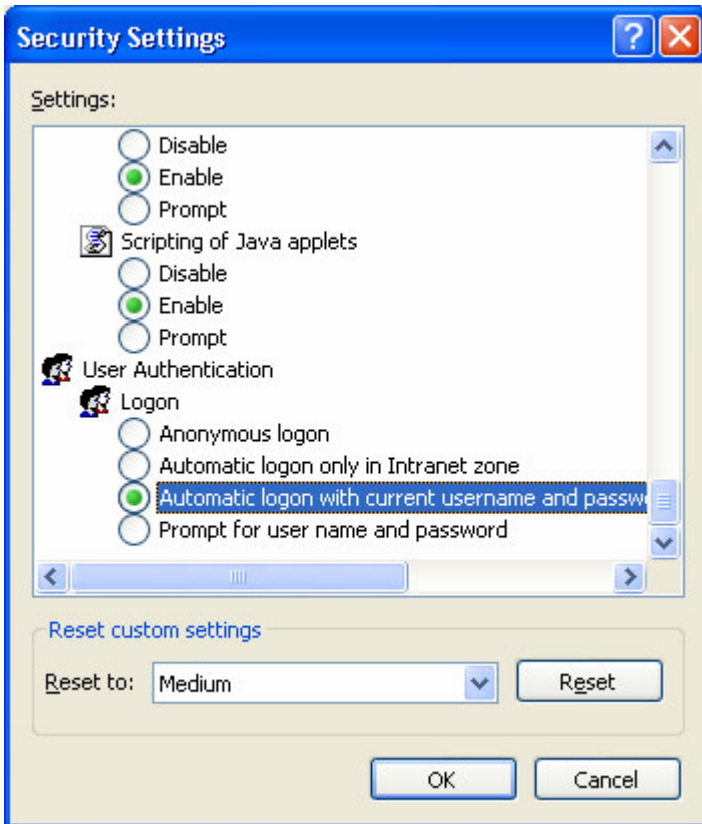


Message: You do not have permission to debug the server

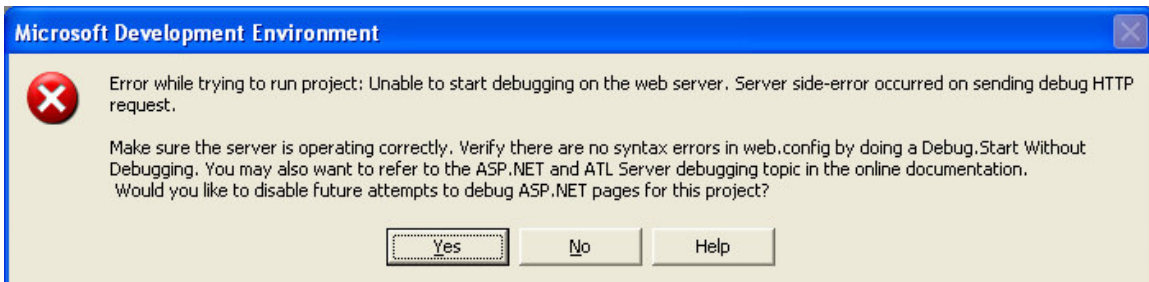


- Cause 1: Make sure that “Integrated Windows Authentication” is enabled. Probably, you enabled only “Basic authentication” for Directory security of IIS.
- Cause 2: If you are using “Integrated Windows authentication”, you need to make sure that your user account has full control on the directory of the IIS.
- Cause 3: If you created the web project with a full machine name (like “machinename.domainname.something”), the web site is recognized as “Internet” site. So the default setting of IE will impact on the behavior of log on. In this case, you need to enable logging on with your current user account in “Internet” area with IE setting.

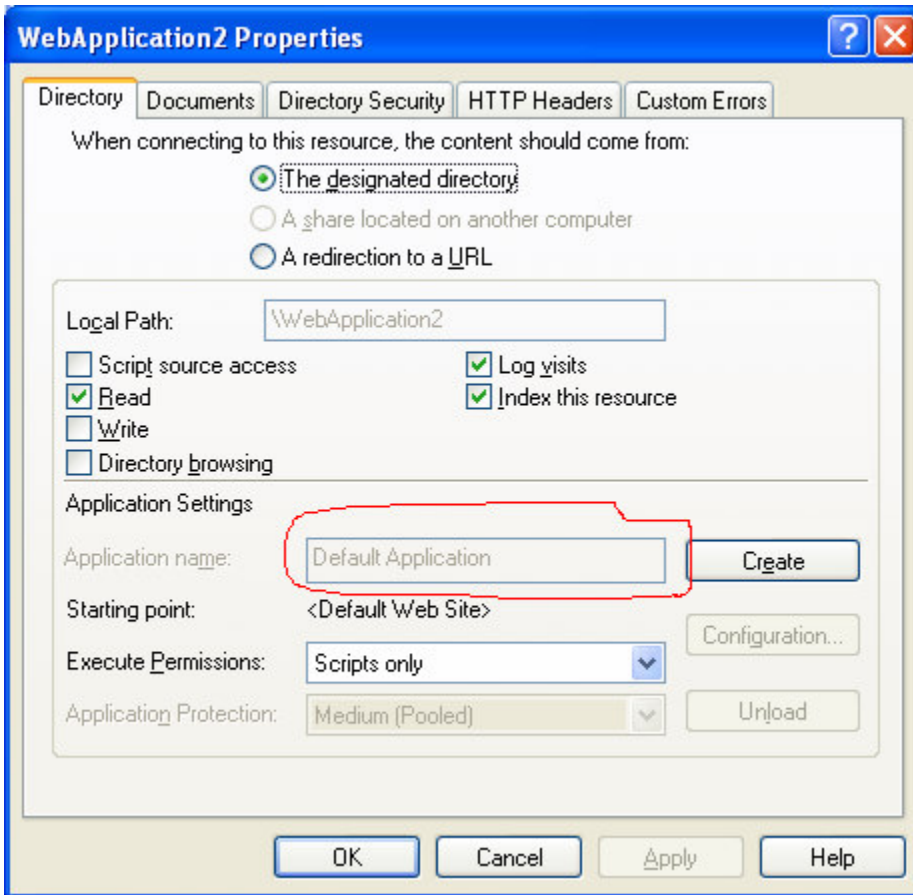
But it is not the default setting of IE, so you’d be better off if you create project with only the machine name.



Message: Server side-error occurred on sending debug HTTP request.



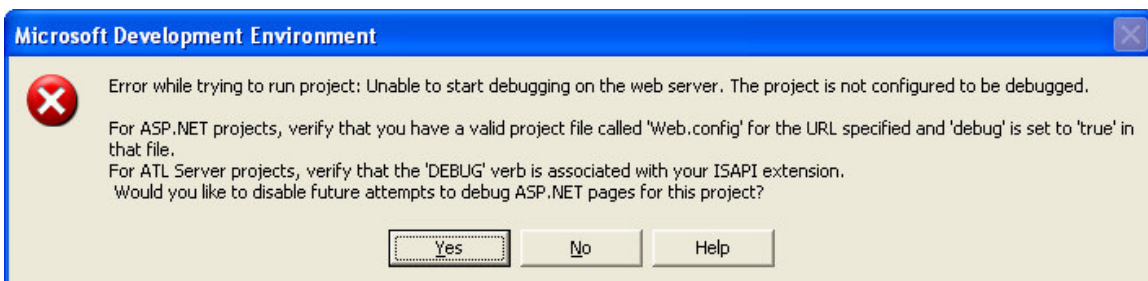
Cause 1: Your web application doesn't have an Application name. Please check the properties of the web project using the IIS MMC to ensure that your web project has an application name



You need to create an application name for debugging.

Cause 2: If you are using the NTFS file format, please make sure that “aspnet” has proper privilege on “wwwroot” or your folder for virtual directory to access and write on the folders.

Message: The project is not configured to be debugged.

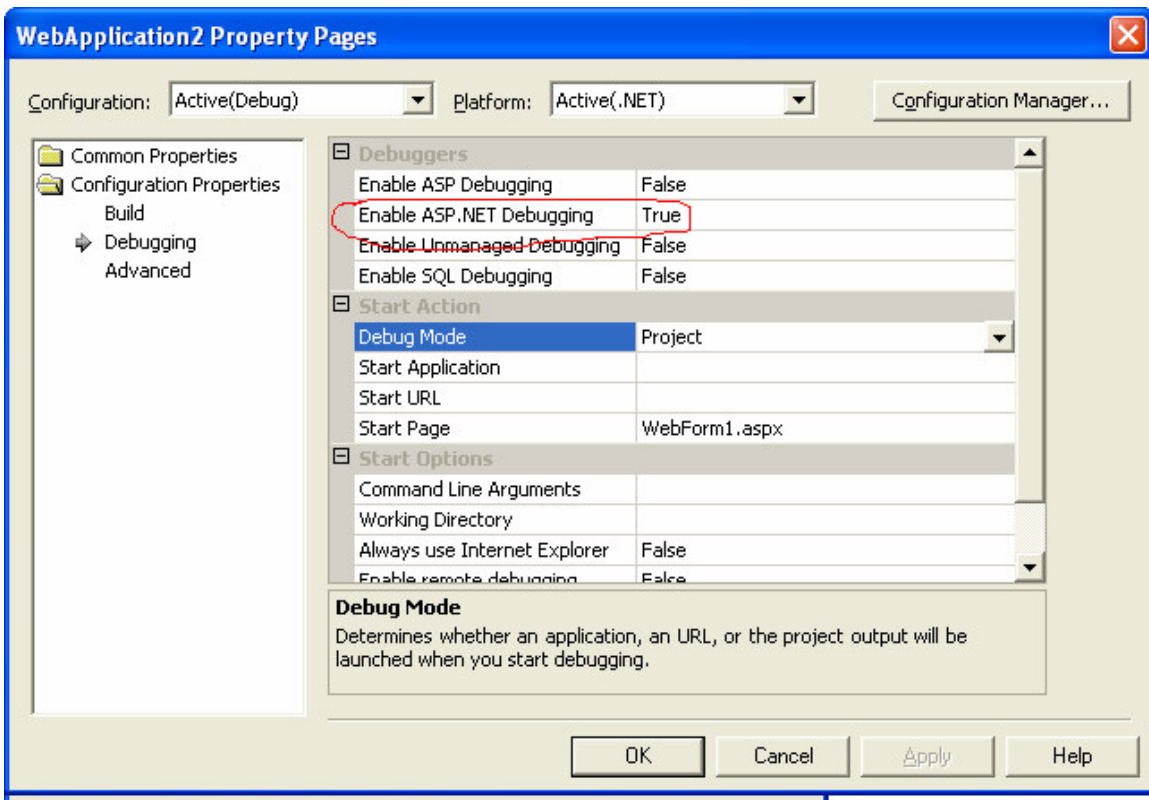


You need to make sure that your web is configured for debugging. To do this, you need set “debug = true” in the “web.config” file. You may find this file in your web project folder.

Can start debugging without error message, but breakpoints are not hit.

You started debugging with “F5” and it looks like debugging is started properly, and IE is launched properly. But you can’t hit a breakpoint on my code behind code.

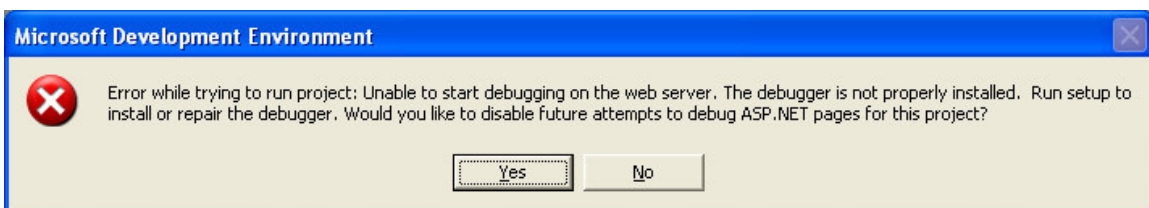
Cause 1: Please make sure that “asp.net debugging” is enabled in the properties of project.



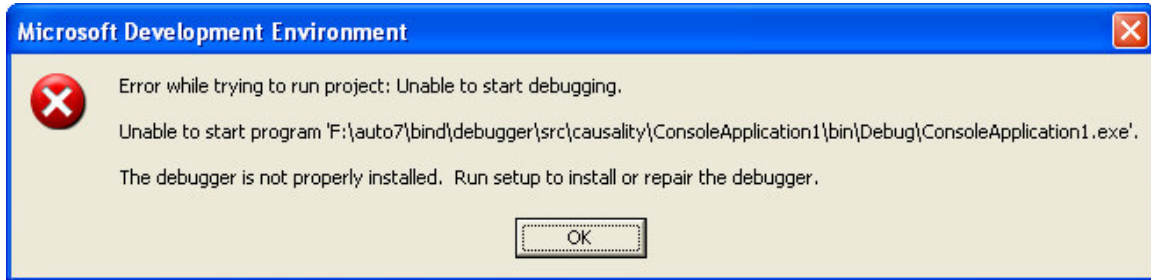
In the case of VB project, the UI is different. But you can recognize the equivalent one easily.

Cause 2: Please make sure that the expected DLL is loaded with matched debug symbol file. You can check it with “Modules” window.

Message: The debugger is not properly installed.

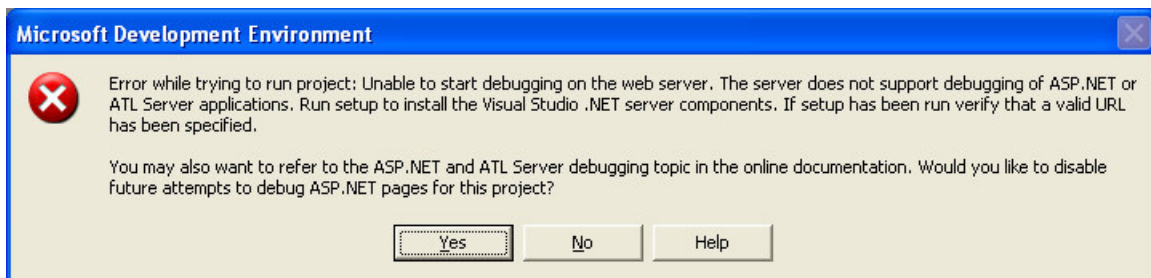


If you see this problem, please check debugging with console application project. And if the console application project shows the error message like



It means that your .Net framework is not installed properly. So you need to register “mscordbi.dll” manually by executing “regsvr32 mscordbi.dll”.

Message : The server does not support debugging of ASP.NET or ATL server applications.



If you have an XP Pro or W2K Pro machine, you may need to think about the order of installation between VS7 and IIS. If you install IIS after VS7, you will get this error. In this case, please register “aspnet_isapi.dll” with “aspnet_regiis.exe -i”.

Message: Access is denied. Verify that you are an administrator or a member of ...

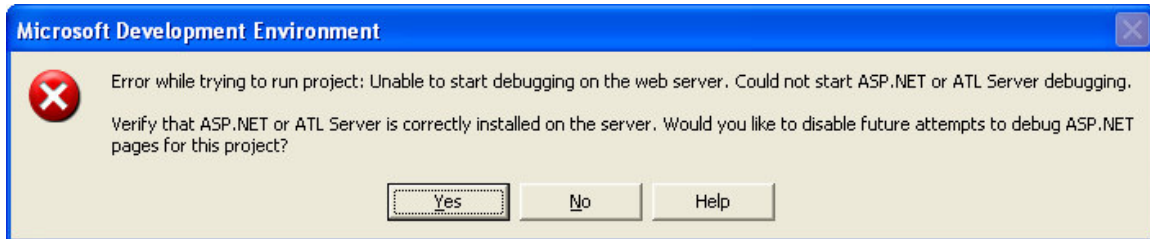


You may not be the member of “Debugger users” group on the machine. Please add your user account into “Debugger users” group on the machine.

To add your user account into “Debugger users” group, you need to do the following:

1. Log in as “Administrator”
2. Run “Computer management” in “Administrator tools”
3. Choose “Local users and groups\groups” node
4. Double-click “Debugger users” group on right pane.
5. Click “Add” button on “Debugger users properties” dialog box.
6. Type your user account and click “Ok” button.

Message: Could not start ASP.NET or ATL server debugging.

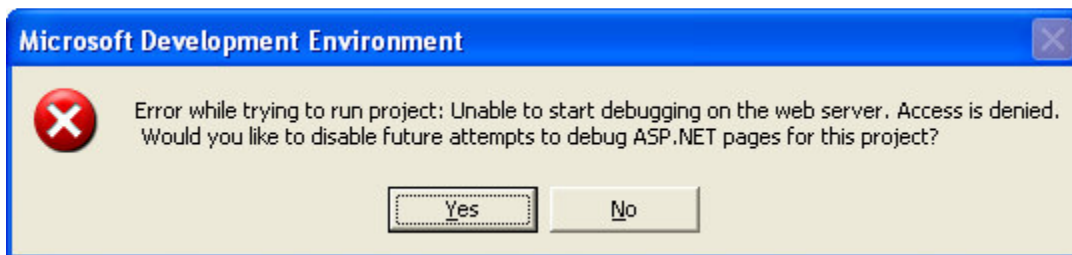


Case 1: You may have installed “IIS Lockdown” tool. If so, Please find “urlscan.ini” file, and add “DEBUG”(case sensitive) into “[allowverbs]” section in “urlscan.ini” file.

Case 2: if you are using Domain controller as server, and your project is created with machine name than full domain name, you may need to change the URL of project to use full domain name.

Case3: if your IIS is set to use dedicated IP as “Web site identification”(you can find this option in IIS setting with IIS MMC), you may see this error message. In this case, you need to change your project name to use the IP address directly. For existing project, you need to change project to use IP address than just machine name by editing “.sln” file and “.webinfo” file.

Message: Access is denied.



You may be the member of “Debugger users” group, but you don’t have the right to debug the aspnet worker process, because you are not the “aspnet” user account or the member of “Administrators” group. Please add your user account to the “Administrators” group on the machine.

Can't debug with an included File

Inside of ASPX, you can't debug with an included file. It may easily happen, when you convert old ASP project to ASPX.

If you include file with "`<!--#include file = "file name"-->`", you may not be able to debug the include file properly. In this case, you need to use "`<!--#include virtual="file name"-->`".

After changing your password, you need to log off/log in for ASP.NET debugging.

After you change your password, you need to log off and log in to do ASP.NET debugging properly.

After installing Windows2000 SP4, ASP.NET debugging doesn't work and it says "Access denied".

After Windows 2000 SP4, if you ASP.NET debugging doesn't work by pressing "F5" and it says "Access denied", please re-register "aspnet_isap.dll" with "regsvr32 -i aspnet_isap.dll". it will fix the problem.

Can hit breakpoint only when the page is loaded first time.

There may be several different cases of this problem. but one of well known cases is that you may turn on page cache option in "web.config" file.

If you see something like "`<add key="<your web project name".Web.EnablePageCache" value="True" />`" in "web.config", you need to set the value as "False" to turn off web page cache. It will let you hit breakpoint when you refresh the page.

Need to share web server for debugging but I don't want to let other users to be admin on my machine.

In VS.Net, we have two things to decide whether user can start debugging. One is "Debugger users" group and the other is user privilege like administrator or power user or SEDebug.

Debugger users group decide whether the user can access VS debug component (mainly MDM-Machine Debug Manager- which is part of VS). So being the member of debugger users group means that you are guaranteed for accessing MDM. So at this point, you can debug your open process and see the list of process on your machine.

But after this, whether you can debug other user's process is decided by your privilege. For example, if you want to debug other people's native process, you should have SEDebug privilege. For the other users' Managed process, you should be administrator on the machine.

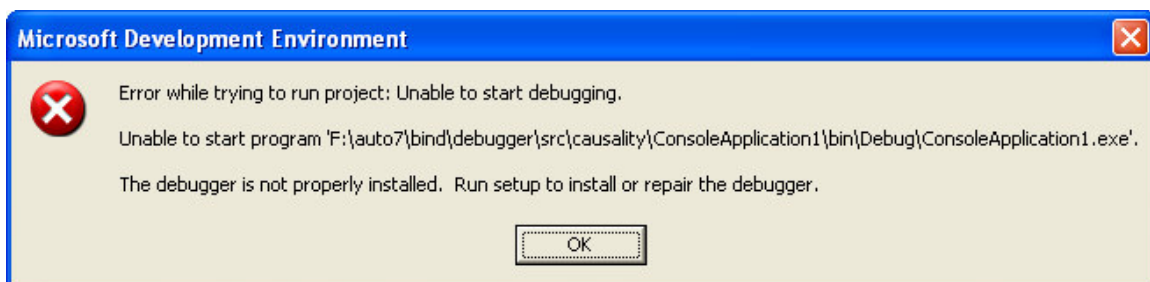
Because of this restriction, in your scenario, the student should be granted as admin. If not ASP.Net work process can't be debugged by default.

We have a sort of work-around. Cassini is stand-alone small ASP.NET server. So for students, they can use Cassini for developing things, and later they just need to deploy thing over to the actual server for submit the result. The Cassini is available on <http://www.asp.net/Projects/Cassini/Download/>.

General Debugging

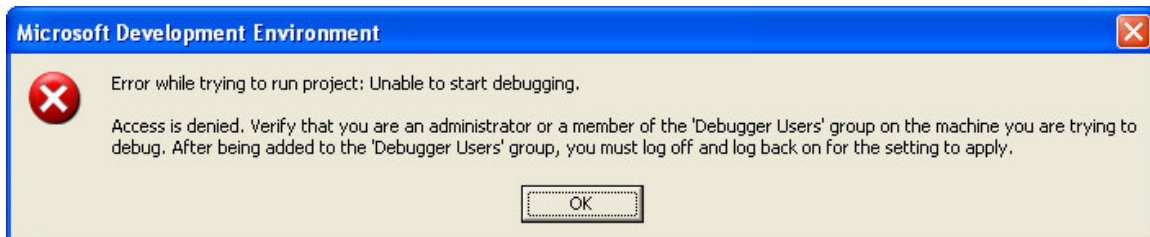
These cases are based on “Console application” project type.

Message: Unable to start debugging. Unable to start program ...



You can't start debugging because “mscordbi.dll” is not registered properly. Please register it manually.

Message: Unable to start debugging. Access is denied



Please make sure that “Machine Debugger Manager” service is started properly. Please check that you are the member of “Debugger users” group or “administrators” group.

Can start Managed debugging, but PDB is not loaded. So, can't hit any Breakpoints at all.

If you can start debugging and debuggee is launched properly, but you couldn't hit any BP, you may need to check the installation of “diasymreader.dll”.

The file may not be registered. So you need to register it manually.

To register it, you need to do

1. find out “diasymreader.dll”

2. regsvr32 diasymreader.dll

Managed debugging is not working

You attach to native process in CLR mode before the process creates the CLR object. Managed debugging is not working.

Solution 1: Attach to the process after CLR code is used in the process.

Solution 2: Attach to the process in InterOp mode. In this case, you don't have to attach to the process after CLR code is invoked.

Managed debugger doesn't respond

When you start debugging with managed code, the debugger doesn't respond at all.

Please make sure that the ".Net framework support" service is stopped and disabled forever. (just stopping the service is not enough.)

If you don't have ".Net framework support" service, Please disable "IIS admin" service. It may help you.

Stepping with C# code is not correct...

Consider the following code

```
string someStr;  
someStr = "SomeValue";  
if(someStr == null)  
    Console.WriteLine("what's up?");  
  
try  
{  
}  
catch(Exception e)  
{  
}
```

If you step through this code, you will see that when you step into the "if" statement, the instruction pointer is moved to the body("Console.WriteLine("what's up?");") of the "if" statement.

This is not debugger bug, it is known issue with try catch block debug information. See the following disassembled code

```
if(someStr == null)  
0000002a  cmp             dword ptr [ebp-18h],0  
0000002e  jne            0000003c  
  
Console.WriteLine("what's up?");
```

```

00000030  mov          ecx,dword ptr ds:[01C50070h]
00000036  call        dword ptr ds:[02F0257Ch]
0000003c  jmp         00000048

catch(Exception e)
0000003e  mov         dword ptr [ebp-1Ch],eax
00000041  call        762C0846
00000046  jmp         00000048

}
00000048  nop

```

When the value is not matched, the instruction pointer is moved to "3c" line, but the line is mistakenly matched with the body of "if" statement. While the code functions correctly, the appearance is incorrect.

Causality debugging: Stepping between web service client and web service.

Can't set from web service client into web service

With the default settings, you can't step from web service client into a web service. It works like step-over.

The ASPNET worker process (aspnet_wp.exe or w3wp.exe) is running under the "aspnet" or "network service" user accounts, and these accounts don't have privilege to access MDM via DCOM. So you need to add these accounts into the "Debugger users" group.

After enabling impersonation of web service, can't do causality stepping.

When impersonation is enabled for the web service with "web.config", stepping into from web service client code to web service code is not working. So it works like step over.

You need to do these things for correct stepping between client and service.

- turn off "Anonymous access" in IIS.
- change your clinet code to set credential to webservice like


```
Service1 obj = new Service1();
obj.Credentials = System.Net.CredentialCache.DefaultCredentials;
```

After this, you will be able to step into the web service from client.

These are needed, because when you step into web service, some framework components need to use the system level debugger component (MDM) to cocreate it. If you don't give your proper credentials, it just fails to do the "cocreation". Without these the proper credentials, "step into" works like "step over"

Debugger hangs

If your web service client code is running in STA(Single thread Apartment) model, and it is waiting for an async call completion like:

```
service1 obj = new service1();
System.IAsyncResult ar = obj.BeginHelloWorld(new
System.AsyncCallback(Class1.Handle),obj);
while(ar.IsCompleted != true)
{
    System.Threading.Thread.Sleep(1000);
}
```

The debugger will hang. This hang occurs because one debugger component is locked by your code inside of debugger

Solution 1:

Change your code to use thread sync with event or mutex.

Solution 2:

Unregister “csm.dll”. In this solution, it will disable causality stepping (stepping from web service client code to web service method). You may need to attach to “aspnet_wp.exe” manually for debugging web service.

Remote debugging

Can't see any processes on a remote machine

Can't see any processes on a remote machine.

Please make sure that you installed “Remote full debug” setup on the remote machine, and that you are a member of “debugger users” group.

Can't connect to a remote machine because of RPC issue

* this information is from one of KB articles. Thanks for “Mike Clay” who wrote this KB. If you see the below error messages while you are connecting to remote machine with “Processes” dialog.

Error while trying to run project: Unable to start debugging on the web server. Not enough storage is available to complete this operation.

Or, you see the below, during ASP.NET debugging.

Error while trying to run project: Unable to start debugging on the web server. Unable to map the debug start page URL to a machine name.

Please make sure that RPC is working properly between your machine and the remote machine. the reason of RPC issue can be from

1. Debugging through a firewall. Microsoft does not recommend or support remote ASP.Net debugging through a firewall. The best way to do overcome this is use Terminal Services to log into the remote server and debug locally.
2. One common failure for RPC is the inability to resolve the remote machine name. RPC relies on name resolution for communication between machines. If you are unable to resolve the remote servers machine name to the correct IP address errors may occur.
3. RPC Traffic can flow in one direction but not the other. RPC traffic must be able to go from the machine used to do the debugging to the remote server and from the remote server back to the machine used to do the debugging in order to successfully debug remotely. Make sure that RPC communication is enabled in both directions.

To analyze your RPC, you can use "RPCping" tool.

Remote debugging fails with machines on a work group

You have two XP Pro machines that are not on a domain, but on a work group. When you do remote debugging between them, you can't access remote machine at all. What's wrong with it?

In the work group environment, you need to make sure that you have the same named user account on both machines with the same password. If not, DCOM will fail to authenticate you.

There is one more consideration:

On XP Pro, because of the default security setting for "sharing and security model for local accounts", this remote debugging is not allowed by default. Below are the steps to change this setting:

1. Run "Local Security Settings" in Administrator tools.
2. Select "Security settings\Local policies\Security options.
3. Change "Network access : Sharing and Security model for local accounts" from "Guest only - local users authenticate as Guest" to "Classic - local users authenticate as themselves".
4. After this, you need to reboot the machine.

This change should be applied onto both machines for remote debugging.

After you make the setting change, you will be able to do remote debugging with the same name user account on both machines.

Please make sure that your user accounts on each machine has password. In some cases, without actual password it doesn't work.

However, *****there is a concern with security with this change*****. Because you changed default settings of the security model, it can expose:

- Unexpected file sharing
- Unexpected DCOM components sharing.

Before you make this change, any kind of connection from remote machine to your machine was guaranteed as "Guest", but after this change, he/she could be authenticated with your local user account. So, like the case of debugging, if you are sharing out a folder or DCOM object, there is a possibility that any matched user (same user name and same password) on other machine could access your shared objects.

I strongly recommend that, if you want to use this work-around, you make sure that all user accounts have strong passwords, or should setup network-Island for the debugging machines to prevent any malicious attack.