



If I Remove or Don't Use Internet Explorer 8, 9, or 10, Can I Avoid Upgrading to Internet Explorer 11?

1

26 days ago by [Pat Altimore MSFT](#)

On January 12, 2016, only the most current version of Internet Explorer available for Windows will be supported. For older OS such as Windows 7 and Windows Server 2008 R2, you will need to upgrade to IE11. Even if you don't use IE on a PC such as a server, you should upgrade to the latest version of the IE browser.

Why Upgrade?

There are many components that constitute the browser. Most of the components are part of the operating system including the JavaScript / HTML rendering engine (MSHTML.dll), the Web Browser control (ieframe.dll), and the Windows Internet Protocol Handler (Wininet.dll). The browser application (IEExplore.exe) uses these OS components for script execution, rendering, HTTP requests, etc. When you upgrade the browser, you potentially upgrade all of these components.

For example, if we use [Process Explorer](#) to view the IEExplore.exe process, the loaded DLLs will include IEFram.dll, MSHTML.dll, URLMon.dll, and Wininet.dll. The Web Browser control (IEFrame.dll) is the entry point and it loads the other DLLs.

Process Explorer - Sysinternals: www.sysinternals.com [REDMOND\patricka]

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name	Session
ieexplore.exe	0.04	14,880 K	40,948 K	5628	Internet Explorer	Microsoft Corporation	
ieexplore.exe	2.33	108,460 K	120,944 K	7816	Internet Explorer	Microsoft Corporation	
EmbeddedIE.exe	0.03	66,820 K	54,404 K	6720	EmbeddedIE	Microsoft Corporation	
SnippingTool.exe	0.28	4,132 K	33,460 K	768	Snipping Tool	Microsoft Corporation	
CSISYNCLIENT.EXE	0.11	25,914 K	20,014 K	8124	Microsoft Office Document...	Microsoft Corporation	
CSISMTTTaskManager.exe							

Name	Description	Company Name	Path
ieframe.dll	Internet Browser	Microsoft Corporation	C:\Windows\System32\ieframe.dll
ieframe.dll.mui	Internet Browser	Microsoft Corporation	C:\Windows\System32\en-US\ieframe.dll.mui
ieproxy.dll	IE ActiveX Interface Marshaling Lib...	Microsoft Corporation	C:\Windows\System32\ieproxy.dll
urlmon.dll	Run-time utility for Internet Explorer...	Microsoft Corporation	C:\Windows\System32\urlmon.dll

CPU Usage: 21.42% Commit Charge: 48.39% Processes: 125 Physical Usage: 50.55%

Now, let's view an application that uses the Web Browser control. EmbeddedIE.exe is a simple test application I created that uses the Web Browser control.

Process Explorer - Sysinternals: www.sysinternals.com [REDMOND\patricka]

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name	Session
WindowsLiveWriter.exe	0.14	119,480 K	103,760 K	1056	Windows Live Writer	Microsoft Corp.	
ieexplore.exe	0.02	14,664 K	39,860 K	5628	Internet Explorer	Microsoft Corporation	
ieexplore.exe	0.22	104,724 K	94,280 K	7816	Internet Explorer	Microsoft Corporation	
EmbeddedIE.exe	0.01	66,496 K	52,524 K	6720	EmbeddedIE	Microsoft Corporation	
SnippingTool.exe	1.20	4,106 K	20,510 K	768	Snipping Tool	Microsoft Corporation	
CSISYNCLIENT.EXE							

Name	Description	Company Name	Path
ieframe.dll	Internet Browser	Microsoft Corporation	C:\Windows\SysWOW64\ieframe.dll
ieframe.dll	Internet Browser	Microsoft Corporation	C:\Windows\SysWOW64\ieframe.dll
ieframe.dll.mui	Internet Browser	Microsoft Corporation	C:\Windows\SysWOW64\en-US\ieframe.dll.mui
urlmon.dll	Run-time utility for Internet Explorer...	Microsoft Corporation	C:\Windows\SysWOW64\urlmon.dll

CPU Usage: 24.79% Commit Charge: 48.73% Processes: 135 Physical Usage: 48.60%

Notice that IEFram.dll is in the list of loaded DLLs. If we looked through the list of loaded DLLs, we would find MSHTML.dll, URLMon.dll, and Wininet.dll.

One last example... I'm using Windows Live Writer to create this blog post. It uses the Web Browser control too. Notice that IEFram.dll is in the list of loaded DLLs. Just like the previous example, all of the other DLLs are also loaded.

Process Explorer - Sysinternals: www.sysinternals.com [REDMOND\patricka]

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name	Session
proceexp.exe	3.092 K	8,700 K	8,700 K	8508	Sysinternals Process Explorer	Sysinternals - www.sysinter...	
proceexp64.exe	4.48	51,380 K	46,384 K	13768	Sysinternals Process Explorer	Sysinternals - www.sysinter...	
WindowsLiveWriter.exe	0.86	121,608 K	108,512 K	1056	Windows Live Writer	Microsoft Corp.	
ieexplore.exe	0.01	19,652 K	51,360 K	5628	Internet Explorer	Microsoft Corporation	
ieexplore.exe	0.01	14,664 K	39,860 K	7816	Internet Explorer	Microsoft Corporation	
EmbeddedIE.exe							

Name	Description	Company Name	Path
ieframe.dll	Internet Browser	Microsoft Corporation	C:\Windows\SysWOW64\ieframe.dll
ieurl.dll	Run-time utility for Internet Explorer	Microsoft Corporation	C:\Windows\SysWOW64\ieurl.dll
igddim32.dll	User Mode Driver for Intel(R) Graph...	Intel Corporation	C:\Windows\SysWOW64\igddim32.dll
igddim32.dll	User Mode Driver for Intel(R) Graph...	Intel Corporation	C:\Windows\SysWOW64\igddim32.dll

CPU Usage: 16.47% Commit Charge: 49.80% Processes: 135 Physical Usage: 50.82%

After January 12, 2016, only the most current version of IE will receive security updates. Therefore, if you aren't upgraded to the current version of IE, you won't be able to apply the current security updates. This could result in some Windows components not being serviced. To ensure applications using

components (e.g. Web Browser control) are fully patched, update to the latest version of IE and apply future cumulative IE updates.

If I Remove Internet Explorer, Will My IE System Components Update?

In order for IE related Windows components to update, you will need to be on the latest version of IE. If you're planning on removing IE, you should upgrade to the latest version first and then remove IE.

When you remove IE from "Windows Features" by unchecking the selection box, IE isn't actually removed from the PC. All of the system components remain for use by the operating system and other applications. The web browser application (IExplore.exe) is "hidden" and not removed. When you apply security updates to the PC, the Windows components as well as the hidden IExplore.exe are serviced. Therefore, if you re-enable IE, it should be up to date.

For tips and tricks on migrating to IE11 in an enterprise environment, see my [Internet Explorer 11 Migration Quick Start](#) post.



Internet Explorer 11 Migration Quick Start

2 months ago by [Pat Altimore MSFT](#)

0

On Jan 12, 2016, [IE11 will be the only supported version of IE on the most Windows versions](#). In this post, I will cover the essentials to jumpstart your migration. There are three main areas to plan for when you migrate: deployment, configuration, and compatibility.

Deployment

There are several ways to deploy IE. Choose the method that works best for your environment. Here are a few tips to consider when choosing a method

- When choosing a method, keep in mind prerequisites, number of reboots, language packs, and cumulative updates.
- If you are planning to leverage Enterprise Mode or Site Discovery functionality, you will need cumulative updates applied (June '15 or greater). Otherwise, this functionality will not work until the updates are applied.

The main deployment options are Microsoft System Center Configuration Manager (SCCM) or similar management tool, Windows Server Update Services (WSUS), Deployment Image Servicing and Management (DISM), and Internet Explorer Administration Kit (IEAK).

There is a great blog post that [compares the deployment options here \(French\)](#). Alternatively, use Microsoft Translator to [translate the post to your preferred language](#).

More resources:

- [Internet Explorer 11 Deployment Guide](#)
- [Choose how to install Internet Explorer 11](#)
- [How to create an all-inclusive deployment package for Internet Explorer 11](#)

Configuration

The recommended way to configure and manage settings in IE11 is to use group policy and group policy preferences (GPP).

- Leverage the ADM template for Internet Explorer (inetres.admx). This contains the recommended default policies that apply to IE.
- Group policy preferences for IE are available as an [Internet Settings Extension](#) in the User Configuration section.
- If you used Internet Explorer Maintenance (IEM) in the past, GPP is its replacement. Use [this reference](#) to map IEM settings to GPP.
- When you test sites for compatibility, configuration can influence compatibility. Test in an environment that matches your planned production configuration.

A full list of GPO settings to configure IE is available in a downloadable [Excel sheet](#). Filter the sheet on inetres.admx to view all the IE settings from version 6 through 11. A great GPO search engine that contains all ADM template settings including IE is available [here](#).

Compatibility

In a perfect world, you would update all sites to modern standards as part of your Internet Explorer 11 migration. This would include all your existing line of business sites, 3rd party vendor sites, and external sites. This usually isn't possible in a short amount of time. Therefore, you can leverage the compatibility features included in IE11 to mitigate your older web sites.

To quickly learn IE's compatibility features, I highly recommend the following resources:

1. View the [Enterprise Mode Deep Dive](#) from Ignite.
2. Download the [Web Application Compatibility Lab Kit](#). This contains an excellent overview presentation and documentation to understand, configure, and use Internet Explorer's compatibility features.

Inventory

Having a good understanding of your inventory of web sites in your enterprise is valuable preparation for a migration. Enterprise Site Discovery can help you collect information about what sites users are visiting.

For more information, review the blog post about [Enterprise Site Discovery](#) and the TechNet article on collecting data with [Collecting data with Enterprise Site Discovery](#).

What Compatibility Mitigation should I Use?

IE can "emulate" prior version of the browser using Document Modes, Compatibility View, and Enterprise Modes.

The following is a quick definition of each compatibility feature and how it can be controlled.

Compatibility feature	Controlled using
Document Modes interpret webpages and will display them similar to older version of IE. The available modes are 5, 7, 8, 9, 10, and 11. Document mode 5 (Quirks mode) represents IE versions 1 through 6.	X-UA-Compatible meta tag Enterprise Mode site list F12 Developer tools
Compatibility view was introduced in IE8 to support sites designed for IE7 and earlier. It emulates document mode 7.	User defined list in IE's Compatibility View settings Microsoft defined list for the Internet zone IT defined domain list using group policy IT defined for the entire Intranet zone using group policy
Enterprise Mode is a "high fidelity" emulation of IE8, IE7, or IE5.	Enterprise Mode site list F12 Developer tools

Since there are several ways to control the compatibility features, IE prioritizes the different methods to allow you to "layer" the compatibility options. When planning your compatibility strategy, it's important to understand the following is the priority list (highest to lowest):

Priority	Compatibility setting	Owner
1	F12 Developer Tools	Development, Test
2	Enterprise Mode site list	IT
3	X-UA-Compatible meta tag	Development
4	Compatibility View	IT
5	<!DOCTYPE> directive	Development

See [How Internet Explorer Chooses Between Document Modes](#) for more information.

For guidance on how to determine a document mode using F12 developer tools and adding it to your Enterprise Mode site list, see: [Fix web compatibility issues using document modes and the Enterprise Mode site list](#)

Modernizing Web Sites

It's recommended to run in the highest document mode available. A good analogy would be purchasing a car. You will get the best performance, reliability, and security with a newer model car. Similarly, higher document modes give you the best performance, reliability and security. If you have web site that you want to update to modern standards, here are some resources to get you started.

- [IE compatibility changes by version](#)
- [Modernization tools](#)

Summary

- Pick a deployment technique that will deploy IE with the latest cumulative updates.
- Leverage the ADM templates and decide on your configuration settings early. Test using your planned production settings.
- Leverage Enterprise Mode site list to mitigate older sites.

- Keep in mind that compatibility features can be layered. Add only broken sites to the Enterprise mode site list.



How to Shutdown Windows with Your Eyes Closed

11 months ago by [Pat Altmore MSFT](#)

0

It seems inevitable. At some point, you get into a situation where you need to shutdown Windows without a display or a mouse. Maybe your laptop screen has failed or your display driver is problematic. Or perhaps, you need to drive some automation to shutdown Windows. This little trick will allow you to shutdown Windows XP, Vista, 7, 8, 8.1, and Windows 10 from the keyboard.

⌘+d, Alt+F4, Enter Method

This method works on Windows XP, Vista, 7, 8, 8.1, and 10. I prefer this method since it works on the most versions of Windows. Here's how this method works.

1. The **Window Key+d** shortcut displays the desktop.
2. The **Alt+F4** shortcut closes the active window. In this case, since the active window is the desktop, you will get the Shut Down Windows dialog.



3. The default option is to shut down. Therefore, you can press the **Enter** key to shut down Windows.

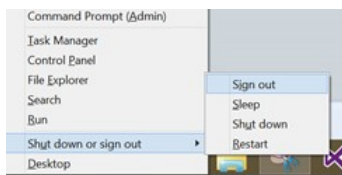
⌘+x, u, u Method

This method only works on Windows 8 or greater. It requires the Start shortcut menu that was introduced in Windows 8. Here's how this method works.

1. The **Window key+x** shortcut displays the Start menu shortcut menu.



2. Pressing **u** will select the "Shut down or sign out" option.



3. Pressing **u** a second time will select "Shut down" option and Windows will shut down.



Controlling WebBrowser Control Compatibility

over 1 year ago by [Pat Altmore MSFT](#)

16

If you have an embedded browser in your Windows application, you may be encountering a rendering or some other kind of compatibility issue with the content you are trying to display. In this post, I'll talk about some options to try to fix those issues. The [web browser control](#) is known by many names: WebBrowser control, WebOC, shdocvw, iframe, etc. I prefer calling it the Web Browser control but whatever you like to call it, the compatibility issues and solutions are the same.

What are the Defaults?

I was at a restaurant once with some friends. When the waitress asked my friend what he would like on his hamburger, he said "What are the defaults?" The developers at the table knew what he meant but the waitress was very confused. 😊

Like burgers, the WebBrowser control has a default configuration. By default, the WebBrowser control runs in compatibility view mode. Compatibility view mode is basically document mode 7.

Below is a screenshot of a simple .NET application using the web browser control.



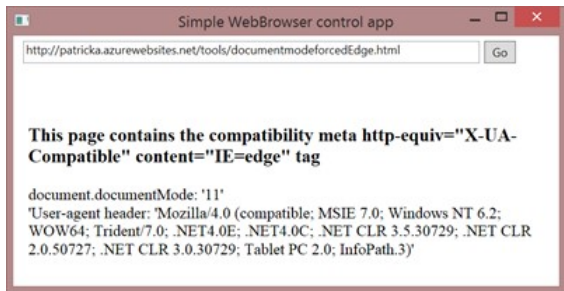
The web page reports back the document mode and the user agent string received by the server. By default, the WebBrowser control sends a "IE 7" user agent string and renders the content using document mode "7".

How do I Change the Default WebBrowser Document Mode?

If compatibility view does not render your content correctly, you have a couple of options. Depending on your scenario, you may need one or both of these options.

Compatibility Meta Tag

If you have control over the web sites or web pages that are being accessed by the web browser control, you can add compatibility meta tags to control the document mode. Below is a screenshot of the same WebBrowser control application displaying a page which includes `<meta http-equiv="X-UA-Compatible" content="IE=edge">`.



The `"IE=edge"` causes the browser control to render the content in the highest available document mode. In this case, IE 11 is installed resulting in the content being displayed in document mode 11.

Note: The user agent string sent to the server in the request is IE 7. Be aware of this mismatch if you will also be browsing to the same content in the IE browser. The full IE browser would send an IE 11 user agent string unless the site was included in the compatibility view list. If your web page is doing browser sniffing via the user agent string, you could encounter rendering differences or errors.

The WebBrowser control can only support the document modes available for the installed browser version on the client. If the client has IE 8 installed, a compatibility meta tag with the value of `"IE=10"` would default to the highest mode available of 8. You could consider defining multiple document modes and allow the control to pick the highest available document mode. However, your content would need to render correctly in the different modes.

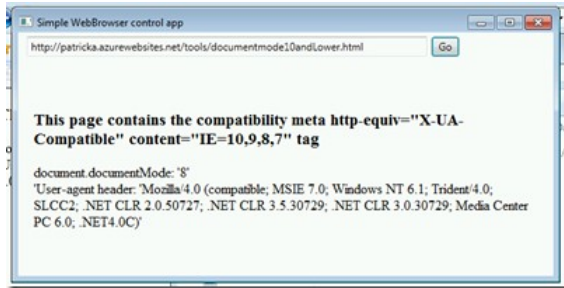
Here's an example page with the following compatibility meta tag: `<meta http-equiv="X-UA-Compatible" content="IE=10,9,8,7">`

Here is the example app running on a client with IE 11 installed:



Because of the compatibility meta tag, the highest document mode specified "10" is chosen.

Here is the example app running on a client with IE 8 installed:



Since the highest available document mode is 8, the content is displayed in document mode 8.

FEATURE_BROWSER_EMULATION Registry Key

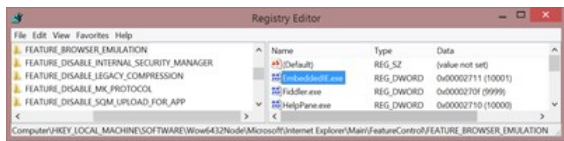
If you need to target a specific version of the browser, you may consider using the [FEATURE_BROWSER_EMULATION registry key](#). This option may work for scenarios where you have a line of business application that has specific content such as help files or an internal web site. The registry key allows you to set a different default document mode for the WebBrowser control for a given application.



32-bit and 64-bit Complexities

You have the option to add the browser emulation key to the current user (HKCU) or the local machine ((HKLM) hive. If you add to the current user HKCU hive, you don't need to worry about "bitness" of your OS or application. However, if you add the key to the local machine hive, you need to consider the following complexities. Depending on the "bitness" of your application, you need to add the key to the correct registry location on a Windows 64-bit machine. On a 64-bit OS, 32-bit applications run in the [WOW64 subsystem](#). For 32-bit apps, [WOW64 redirects registry calls](#) to a separate location for certain keys. The FEATURE_BROWSER_EMULATION key is redirected. Therefore, on a 64-bit OS, a 32-bit application's value needs to be placed in

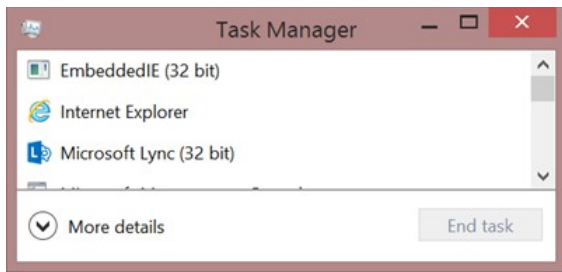
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE_BROWSER_EMULATION.



In this case, we are always requesting the content to be displayed using document mode 10 and ignores the !DOCTYPE for the page.



If you're wondering if your app is a 32-bit app running on a 64-bit OS, you can quickly verify by checking Task Manager.



32-bit applications running on a 64-bit OS will have "(32 bit)" after the name.

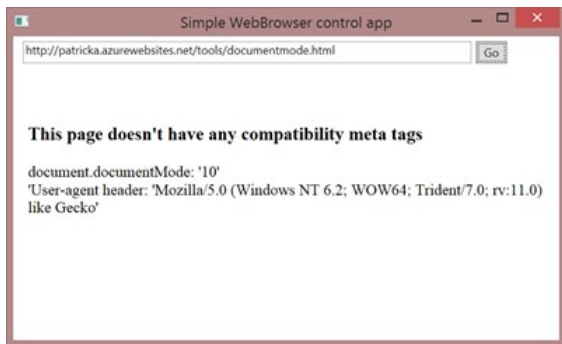
What's the Difference Between a Setting that Ignores the !DOCTYPE and the One that Doesn't?

When you pick one of the settings that ignore the !DOCTYPE, the user agent string matches the requested document mode. In the following example, the FEATURE_BROWSER_EMULATION value is 10000. This ignores the !DOCTYPE.



Note: The browser control sends a IE 10 user agent string.

If we choose a setting that looks for a standards doctype, the browser control sends the default standards based user agent string. In the following example, the FEATURE_BROWSER_EMULATION value is 10001.



Note: The browser control sends the default standards based user agent string. In this case, IE 11 is installed on the client. Therefore, the user agent string is an IE 11 user agent string.

What Happens if I Set the FEATURE_BROWSER_EMULATION Document Mode Value Higher than the IE Version on the Client?

Obviously, the browser control can only support a document mode that is less than or equal to the IE version installed on the client. Using the FEATURE_BROWSER_EMULATION key works best for

enterprise line of business apps where there is a deployed and support version of the browser. In the case you set the value to a browser mode that is a higher version than the browser version installed on the client, the browser control will choose the highest document mode available.

Here is an example of setting the value to 9000 on a client that has IE 8 installed.



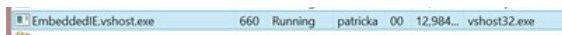
We can see that the browser control displays the content in the highest available document mode of 8.

Does the Compatibility View List or Enterprise Mode Affect the Behavior of the WebBrowser Control?

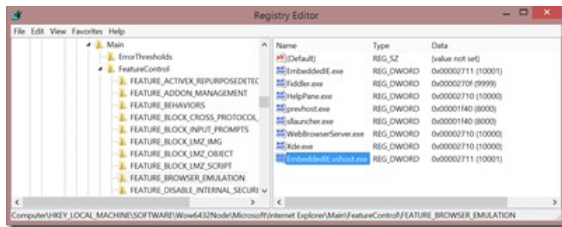
Compatibility View list and Enterprise mode are features of the full Internet Explorer (iexplore.exe). Therefore, the web browser control does not support the Compatibility View list or Enterprise mode.

Debugging Your WebBrowser Control App in Visual Studio

The FEATURE_BROWSER_EMULATION key values require the executable name. If you are debugging your app in Visual Studio, your app is being hosted in Visual Studio. Therefore, you need an entry for the host. Task manager details can help determine what the value should be named.



Based on this information, the app is 32-bit and the executable name is "EmbeddedIE.vshost.exe". Therefore, the value needs to be added to Wow6432Node and name is "EmbeddedIE.vshost.exe".



Otherwise, you will see different behavior when debugging verses launching the executable.

Using IE to Determine what Document Mode You Need

The Emulation F12 development tool is very handy for determining what document mode you might need for your content.

In IE, just press F12 and select the "Emulation" tool.



Using the Document mode setting, you can change the document mode to quickly see how the page would be rendered in a different mode. You can also change the User agent string to investigate how the server will respond to different user agent strings.



Check it Out! I Wrote a Windows Store App! You can too with APIMASH Starter Kits.

over 3 years ago by [Pat Altimore MSFT](#)

If you are trying to get started with creating a Windows 8 Store App, there are some great starter kits that show you how to use several public web service APIs. The kits have sample code to access APIs such as Edmunds, Tom-Tom, Twitter, Tumblr, Yelp, Meetup and many others. Check them out on [GitHub](#).

[Mark Rosewater](#) is a friend of mine. He posts a daily comic about [Magic the Gathering](#) on card game his [Tumblr blog](#). I modified the Tumblr starter kit to allow users to flip through the last 30 comics on his blog. Check out my ["Tales from the Pit" App from the Windows Store \(it's free!\)](#).

Enjoy!

Pat



THE "App Compat Guy" - Still Crazy After All These Years

0

over 3 years ago by [Pat Altmore MSFT](#)

Believe it or not, way back when Windows Vista was in beta, there were a bunch of "app compat guys". Over the years, many of us have moved on to other things but one guy has stood the test of time. Chris Jackson has earned the title of "The App Compat Guy" hands down. Chris is always a joy to watch and I [recommend watching his recent appearance on the Edge Show at TechEd](#). Chris talks about how compatibility has changed over the years and gives his insights.

As a bonus, I'll share my favorite Chris stories... Back when we were doing a lot of App Compat stuff with ISV's in building 20, Chris was visiting Redmond and stopped by. There used to be an Xbox room with a nearby kitchen stocked with really good ice cream bars. Chris had never played Guitar Hero before and he tried it out one afternoon. I saw him the next day looking very tired. He played the game for 12 hours straight until 4 am only stopping to eat ice cream bars. 😊



How Do I Deploy a Windows 8 App to Another Device for Testing?

14

over 4 years ago by [Pat Altmore MSFT](#)

If your developing a new Windows 8 app and you want to test it on another device (e.g. Surface), you'll need to use a technique called sideloading. This can easily be done through a few steps that I'll describe in this post.

What is Sideloading?

Windows 8 Store Apps are deployed through the Windows Store. This may seem like an obvious statement. However, by default, *only* Windows Store apps are permitted to install and run on Windows 8. When you install an app from the Windows Store, a Windows Store signed app package is downloaded and installed on your machine. Since only Windows Store signed apps are permitted to be executed on Windows 8, it greatly reduces the introduction of malware. For development and enterprise scenarios, you can use a technique called sideloading to install and run apps that are not Windows Store signed. I'll cover development sideloading in this post. If you are interested in how to use sideloading for line of business apps in an enterprise, [please see this post](#).

Development Sideloading

Development sideloading is used by Visual Studio to deploy, debug and test your app.

The general requirements for sideloading are:

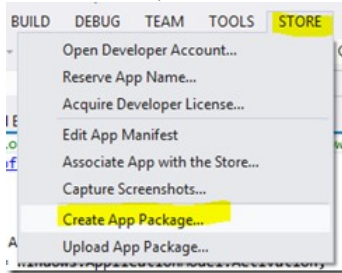
1. Your appx package needs to be signed.
2. The certificate used for signing needs to be trusted.
3. A policy (or registry key) needs to be set to allow trusted apps to run.

On a development machine, these requirements are automatically met for you by Visual Studio. Visual Studio provides a self-signing test certificate that is used to sign the package. Also, when Visual Studio is installed, a registry key is set to allow trusted apps to run. Therefore, when you debug a Windows 8 Store app using Visual Studio, the app is test certificate signed, deployed, and allowed to run.

How do I Sideload My App on Another PC or ARM Device?

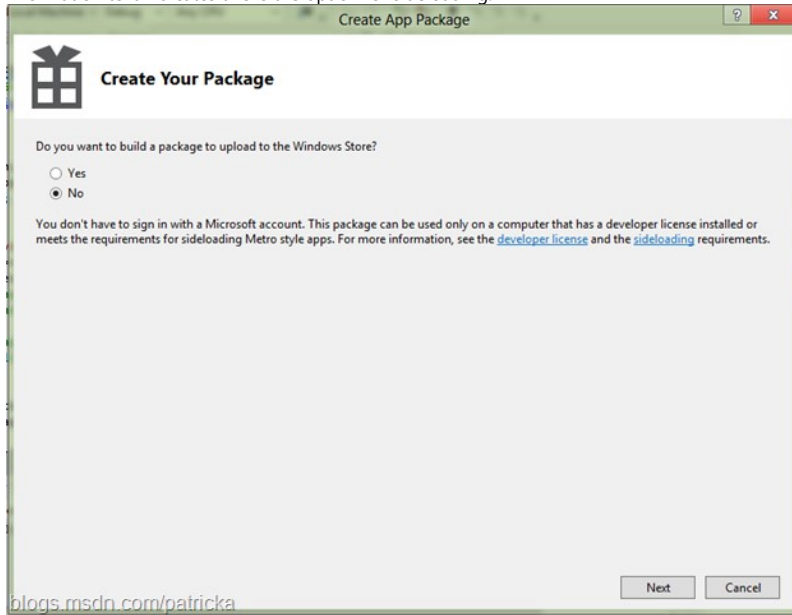
In order to deploy an app, you need an [App Package](#). Visual Studio has functionality to create an App Package you can use to deploy your app to another machine. The following steps are how to create a local App Package.

1. In Visual Studio Express 2012, choose "Create App Package" in the Store menu.

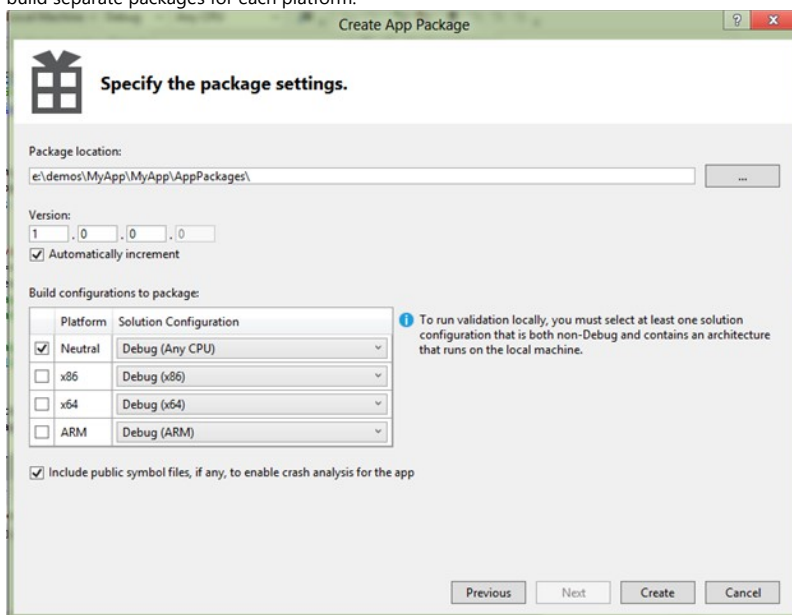


Note: In Visual Studio 2012 Ultimate, the Store menu is in a different location. In the "Project" menu, use the "Store" option.

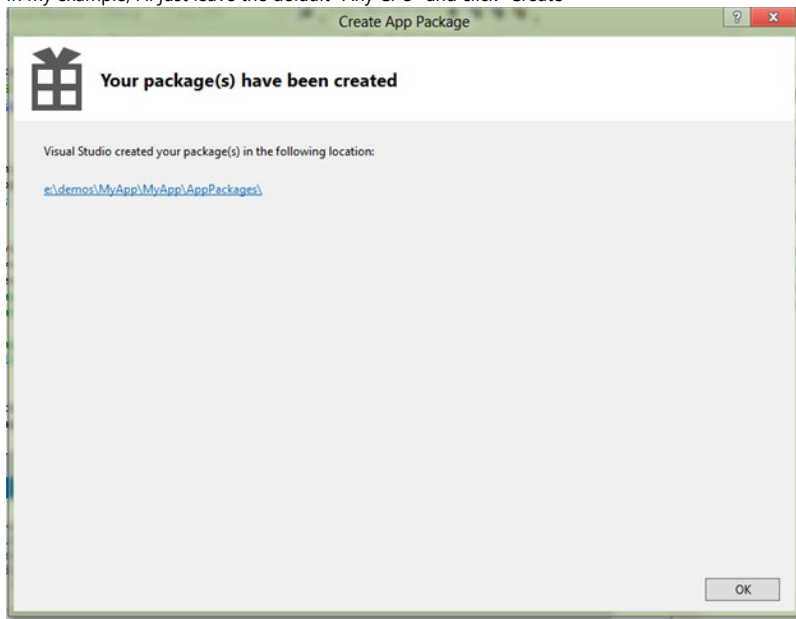
2. In the Create App Package wizard, you will want to create a local package. Therefore, answer "No" to the "Do you want to build a package to upload to the Windows Store". Notice the information text indicates this is the option for sideloading.



3. The next dialog provides a choice of processor platforms to target. In most cases, you would want to choose "Any CPU". This will allow you to target all of the processor platforms and architectures. However, if you are using platform specific binaries in your app, you'll need to build separate packages for each platform.



4. In my example, I'll just leave the default "Any CPU" and click "Create"



Note the wizard gives us a link to where the package was created.

If we click the link, here is an example of what was created:

Name	Date modified	Type	Size
Dependencies	8/7/2012 4:50 PM	File folder	
Add-AppDevPackage.ps1	5/21/2012 9:17 PM	Windows PowerShell Script	29 KB
Add-AppDevPackage.psd1	4/1/2012 10:10 PM	Windows PowerShell Data File	5 KB
MyApp_1.0.0.0_AnyCPU_Debug.appx	8/7/2012 4:50 PM	APPX File	22 KB
MyApp_1.0.0.0_AnyCPU_Debug.cer	8/7/2012 4:50 PM	Security Certificate	1 KB

Here is a quick summary of the relevant files in the package directory:

Appx package (MyApp_1.0.0.0_AnyCPU_Debug.appx)

The appx package is a zip file of all of the program files for your app. This file gets deployed and installed into Windows. Try renaming the extension to ".zip" and open the zip file to see what is inside. Don't forget to change it back before you try and deploy it.

Developer test certificate (MyApp_1.0.0.0_AnyCPU_Debug.cer)

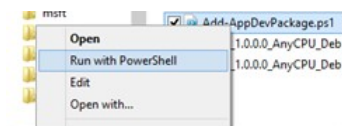
This is your development certificate. This is the certificate that was used to sign the appx package.

PowerShell script (Add-AppDevPackage.ps1)

This PowerShell script can be used to install the development certificate and install the App.

Installing the App on the Target Machine

Everything you need to deploy the app is included in the package directory created by Visual Studio. I usually copy the directory to a USB memory stick to transfer the app to the target machine. On the target machine, you just need to run the PowerShell script (Add-AppDevPackage.ps1) to deploy the app. You can do this easily by right clicking on the script and selecting "Run with PowerShell".



The first action performed by the script is to install the certificate. It needs to run elevated in order to perform this step. If PowerShell isn't running elevated, the script will prompt to start an elevated process to install the certificate.

```
Windows PowerShell
Found package: E:\demos\MyApp\MyApp\AppPackages\MyApp_1.0.0.0_AnyCPU_Debug_Test\MyApp_1.0.0.0_AnyCPU_Debug.appx
Found certificate: E:\demos\MyApp\MyApp\AppPackages\MyApp_1.0.0.0_AnyCPU_Debug_Test\MyApp_1.0.0.0_AnyCPU_Debug.cer
Before installing this package, you need to do the following:
- Install the signing certificate
Administrator credentials are required to continue. Please accept the UAC prompt and provide your administrator password if asked.
Press Enter to continue...:
```

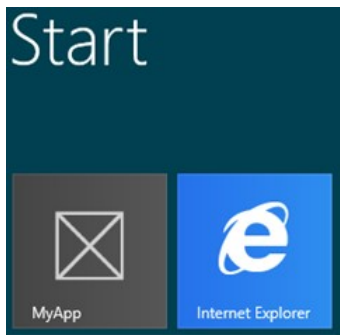
You will be warned and prompted to allow the certificate to be installed.

```
Administrator: Windows PowerShell
Installing certificate...
WARNING: You are about to install a digital certificate to your computer's Trusted People certificate store. Doing so carries serious security risk and should only be done if you trust the originator of this digital certificate.
When you are done using this app, you should manually remove the associated digital certificate. Instructions for doing so can be found here:
http://go.microsoft.com/fwlink/?LinkId=243053
Are you sure you wish to continue?
[Y] Yes [N] No (default is "N"):
```

Once the certificate is installed, the app package is installed.

```
The certificate was successfully installed.
Installing package...
Success: Your package was successfully installed.
Press Enter to continue...:
```

Your app should now be installed on the target machine. You should be able to find the tile on the Start screen (usually at the end of the list):



The final requirement is to allow trusted apps to run. If your target machine has Visual Studio installed, this requirement will already be met. However, if you don't have Visual Studio installed, you can meet the requirement through group policy or setting a registry key. For development purposes, it's usually easiest to set the registry key.

- Use RegEdit to navigate to the key:
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Appx
- Set AllowAllTrustedApps (REG_DWORD) to the value of 1

Your app should now run on the target machine.

One final note... *This is the process for development sideloading.* There is a more formal process for enterprise sideloading. If you are interested in enterprise sideloading, [please start with this post](#).



How do I Test Internet Explorer 10 Touch Behavior Without a Touch Device?

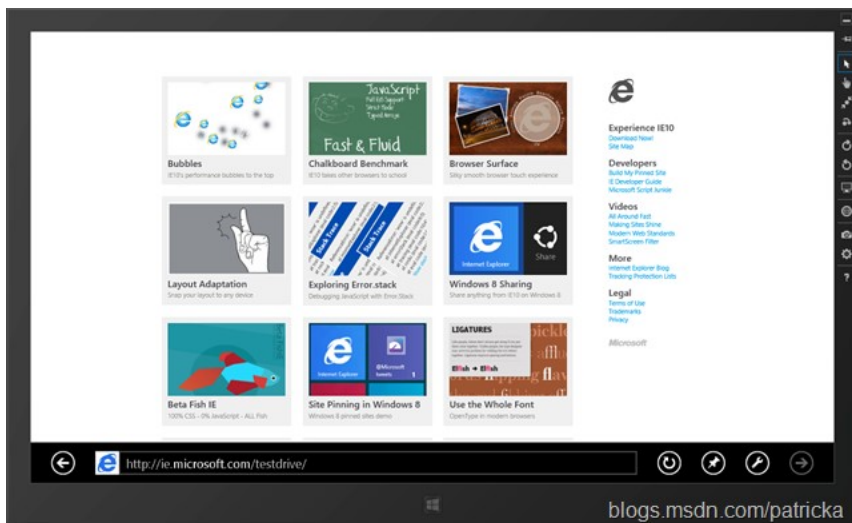
over 4 years ago by [Pat Altimore MSFT](#)

0

For browsing, IE 10 provides a great touch experience with the Metro style browser. If you don't have touch enabled hardware available, here's a quick way to simulate touch behavior to test your site.

Visual Studio Simulator

[Visual Studio Express 2012](#) is designed for writing Metro Style apps. It includes a very nice "slate" simulator. The simulator is designed for testing various screen sizes, resolutions, orientations, layouts, and touch. Therefore, if you don't have a touch monitor, you can simulate it. Here's what IE Metro Style browser looks like in the simulator. The controls on the left allow you to simulate resolution, rotation, and touch gestures.

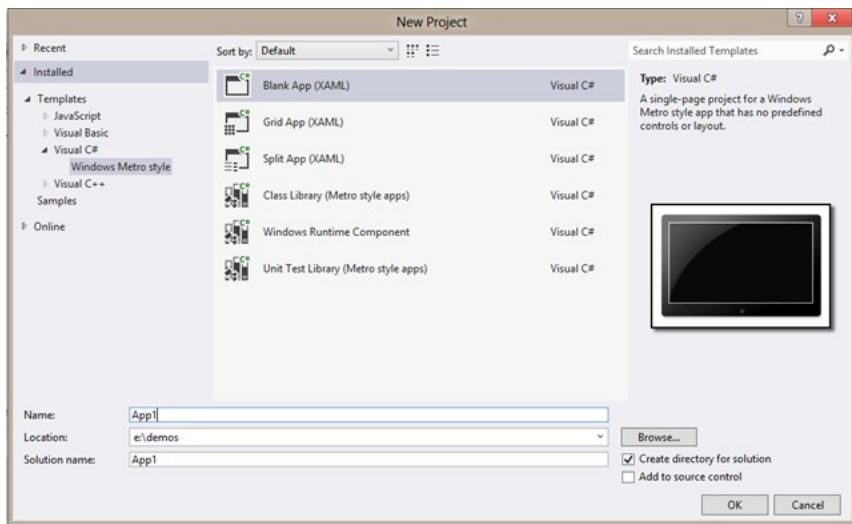


blogs.msdn.com/patricka

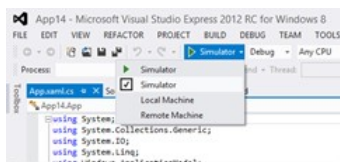
How Do I Get the Simulator?

First, you need Visual Studio 2012 Express. You can get the latest version as well as Windows 8 from MSDN. You can download from the [Downloads for Metro style app development](#) page.

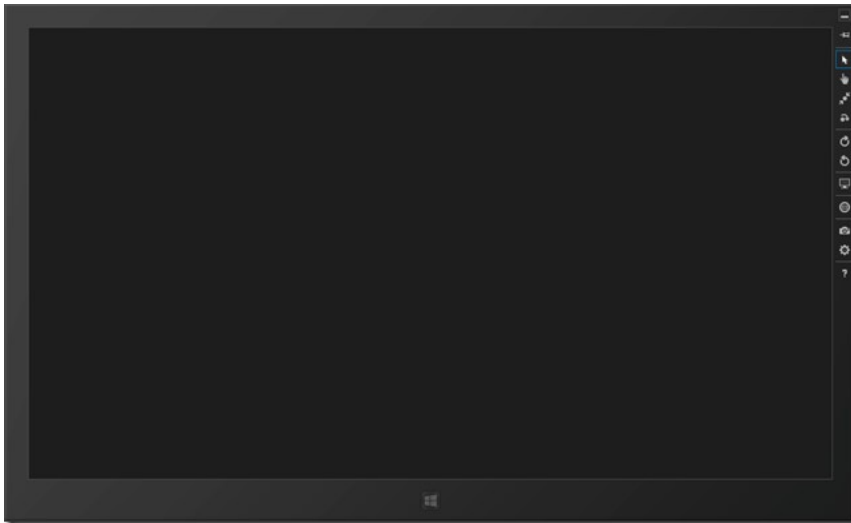
Next, we need to create a Metro style app. The Metro style app doesn't need to do anything. We need a project to launch the simulator. This can be done by creating a new project and selecting one of the Metro style templates.



Now we just need to debug our new app in the simulator. I like using the debug toolbar to target and launch the simulator.



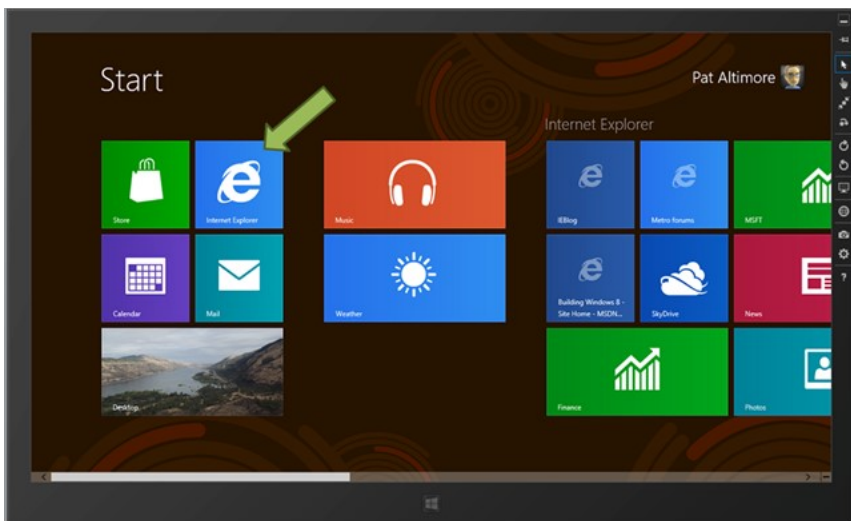
You should now see your simple Metro style app running in the simulator. I used the "blank" template in my example resulting in a black screen.



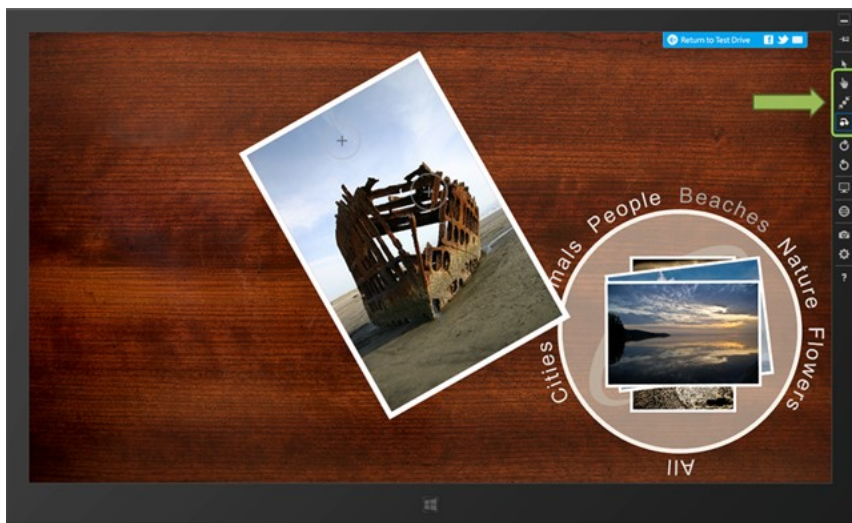
Starting the IE 10 Metro Browser in the Visual Studio Simulator

The simulator is a full virtual desktop environment. Everything that is available on the local machine is also available in the simulator including Internet Explorer 10.

Since the simulator is a representation of a slate, switch to the desktop by clicking the "Windows" start button at the bottom center of the simulator. This will take you to the start menu. Click the IE tile to launch the IE 10 Metro style browser.



The touch controls are grouped together at the top of the toolbar. Here's an example from the [IE Test Drive site](#) that allows you to zoom and rotate pictures with multi-touch gestures.

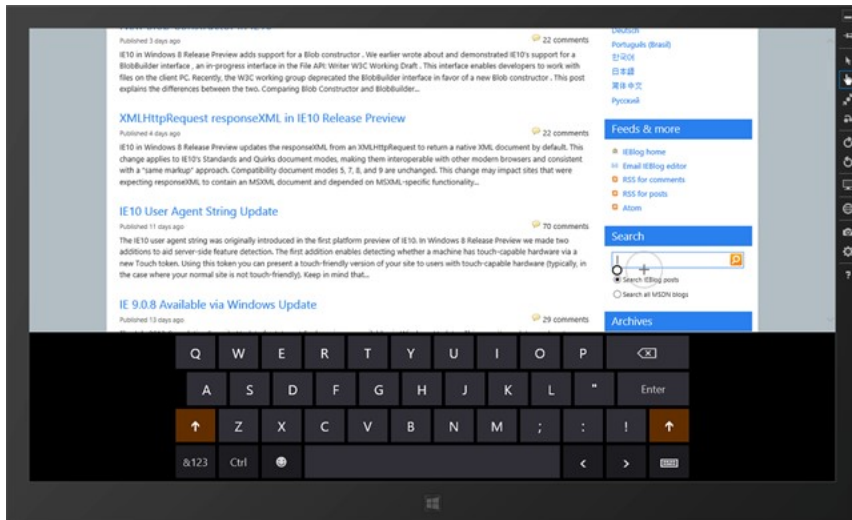


The “hand” control allows you to test single touch gestures such as tap and pan. The pinch zoom and rotate controls allow you to test multi-touch gestures. Hold down the left mouse button and roll the mouse wheel to change the distance between the multi-touch touch points. See this [article in MSDN](#) for more information.

What Should I Test?

Some behavior does change in the Metro style browser based on touch. This isn’t an exhaustive list of things to test but will give you an idea of what to test.

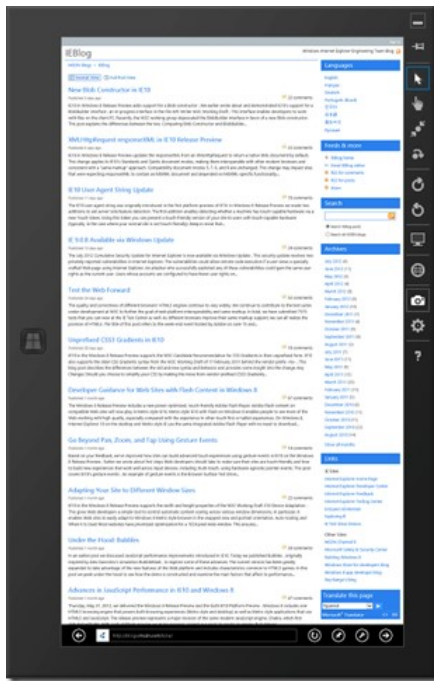
On-Screen Touch Keyboard – The on-screen touch keyboard is displayed if a user taps into a text field. This is different behavior from the mouse and keyboard.



Gestures – You should test the multi-touch gesture pinch zoom to see the behavior of your site. Don’t forget to test single touch panning and “flick” to scroll through a list.

Bonus Testing

Beyond the touch simulation, you can test orientation (Portrait) and resolution with the other controls on the simulator. These are good tools for inspecting layout for slate devices.



Happy testing!



Windows 7 Touch Development Sessions from TechEd

over 5 years ago by [Pat Altimore MSFT](#)

0

I had the opportunity to go to [TechEd North America 2011 in Atlanta](#) this year. Channel 9 has made [all the sessions from TechEd available](#) on their website. Now you can listen to [Ben Farmer](#) and me discuss Windows 7 multitouch development.

In the [Windows 7 Touch Application Development: Selecting the Right Platform](#) presentation, I cover details regarding which development platform is right for your app. I include discussion of Windows Native, Silverlight, WPF, and HTML platforms along with some respective examples of shipping applications. For your enjoyment, you can watch my demo machine crash twice. 😊 We found out it was caused by the KVM screen switcher not being powered correctly. Oh well....

In [Building a Windows 7 Touch Optimized App: A Practical Guide from Design to Release](#), I kick off the presentation discussing a lot of the same concepts in the "Selecting the Right Platform" talk and then turn it over to Ben who does an awesome job discussing lessons learned in developing Touch optimized applications from a partner's perspective.



Developing a Native Win32 Application for Windows 7 Touch

over 5 years ago by [Pat Altimore MSFT](#)

0

In a previous [post](#), I discussed pros and cons of different development platforms for Windows 7 touch applications. In this post, I'll continue elaborating on each development environment and discuss common development considerations and issues for a touch application when using native C++. If you haven't read the [Windows Touch Guidance](#) whitepaper, please review this document for good general guidance on developing a touch application. Note: Some of this base material is repeated from my [WPF post](#).

How Much "Touch" is Enough?

One benefit of Windows touch is single touch and multitouch gestures are converted to mouse messages if they aren't handled explicitly by your application. Therefore, your application may be somewhat touch friendly already. Depending on the touch scenario you want to support, you may only need to tweak your application in a few areas and be able to provide a good touch user experience. A good reference to explain what makes an application touch "friendly", "enabled" or "optimized" is the [MSDN Touch Guidelines](#).

For the remainder of this post, I'll focus on key touch considerations and suggest ways to implement using native C++.

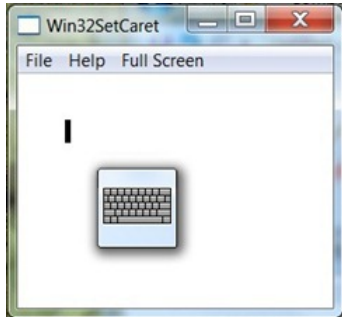
Touch Input Panel (TIP)

The TIP can be very handy if you want to allow users to enter text without a keyboard. Several "slate" PC's now exist and either have an external keyboard or allow the user to configure the notebook to hide the keyboard. The TIP has existed since Tablet PC edition and is available in all versions of Windows 7. For native code, the TIP is displayed when the system caret is displayed.

Here is sample code that displays the system caret in a window if you click the left mouse button.

```
CreateCaret(hWnd, NULL, 5, 20);  
xPos = LOWORD(lParam);  
yPos = HIWORD(lParam);  
  
SetCaretPos(xPos, yPos);  
ShowCaret(hWnd);
```

This is the behavior in an application:



Since the TIP is displayed when the caret is displayed, most native controls that allow text entry will display the TIP. If you are planning on using DirectX, the TIP will not display in full screen mode. You will need to implement your own text entry for full-screen mode DirectX applications.

Screen Orientation and Rotation

The [Windows 7 Engineering Guidance for Slate PCs](#) recommends PC's to enable screen rotation automatically through sensors or a manual hardware button. If you plan on targeting mobile touch PC's, you should consider supporting screen rotation for your touch application. To support rotation, you generally detect the screen dimensions have changed and then change the layout. If your application supports window resizing, you can leverage that same functionality for screen orientation and rotation.

For native C++ applications, you can determine landscape or portrait orientation by using the [GetSystemMetrics](#) function to retrieve the height and width of the full screen. For screen rotation, you can listen for the [WM_DISPLAYCHANGE](#) message in your [WindowProc](#) function's switch statement. For code samples on detecting screen orientation and rotation, see [Detecting Screen Orientation and Screen Rotation in Tablet PC Applications](#).

Gestures, Manipulations, and Inertia

Developing in native C++ allows you full access to all of the Windows 7 touch API's. Therefore, you can implement simple gestures or very complex custom manipulations. Here's a quick summary of each item.

- For raw touch information, you can register your application to receive [WM_TOUCH](#) messages. [WM_TOUCH](#) messages combine down, up, move and other states into one message.
- If you plan on just implementing well known gestures such as pinch zoom, rotate, pan, etc., you can use the [WM_GESTURE](#) message which is generated for common gestures.
- For manipulations, the application needs to implement the [IManipulationProcessor](#) interface.
- For inertia, the [IInertiaProcessor](#) interface can be used in conjunction with the [IManipulationProcessor](#) interface.

One of the best resources for native touch development the [Touch topic](#) in the Windows Platform SDK. Also be sure to review the [SDK samples on touch](#) as well.



Developing a WPF Application for Windows 7 Touch

over 5 years ago by [Pat Altimore MSFT](#)

0

In my previous [post](#), I discussed pros and cons of different development platforms for Windows 7 touch applications. In this post, my goal is to discuss common development considerations and issues for a touch application when using WPF. If you haven't read the [Windows Touch Guidance](#) whitepaper, please review this document for good general guidance on developing a touch application.

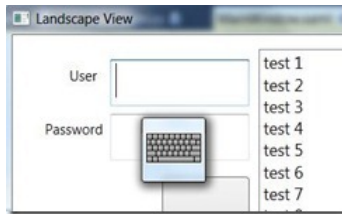
How Much “Touch” is Enough?

One benefit of Windows touch is single touch and multitouch gestures are converted to mouse messages if they aren't handled explicitly by your application. Therefore, your application may be somewhat touch friendly already. Depending on the touch scenario you want to support, you may only need to tweak your application in a few areas and be able to provide a good touch user experience. A good reference to explain what makes an application touch “friendly”, “enabled” or “optimized” is the [MSDN Touch Guidelines](#).

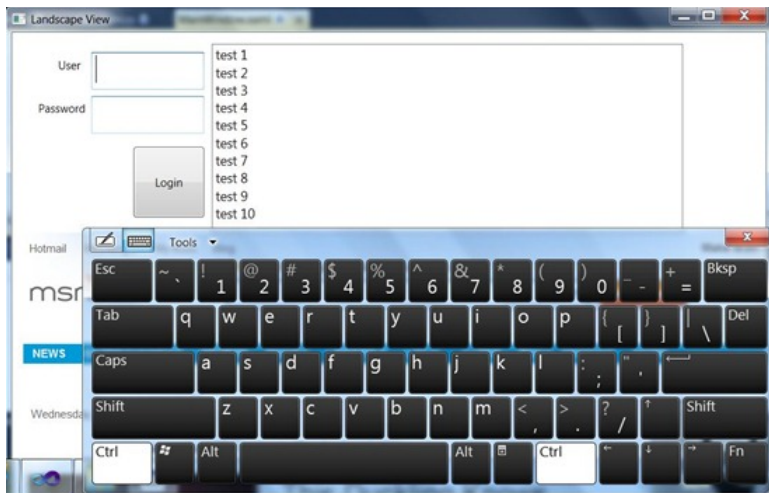
For the remainder of this post, I'll focus on key touch considerations and suggest ways to implement using WPF.

Text Input Panel (TIP)

The [TIP](#) can be very handy if you want to allow users to enter text without a keyboard. Several “slate” PC's now exist and either have an external keyboard or allow the user to configure the notebook to hide the keyboard. The TIP has existed since Tablet PC edition and is available in all versions of Windows 7. WPF will display the TIP automatically for most controls such as the TextBox control. If you touch the TextBox, you will see a TIP icon.



If you touch the TIP icon, the virtual keyboard is displayed



Almost all of WPF's controls that accept text entry support TIP. However, PasswordBox does not support TIP. You will want to avoid using PasswordBox or implement your own way to input passwords without a keyboard.

Screen Rotation

The [Windows 7 Engineering Guidance for Slate PCs](#) recommends PC's to enable screen rotation automatically through sensors or a manual hardware button. If you plan on targeting mobile touch PC's, you should consider supporting screen rotation for your touch application. To support rotation, you generally detect the screen dimensions have changed and then change the layout.

One way to detect a screen rotation is to register for the [DisplaySettingsChanged](#) event. The following is some sample code that registers for the DisplaySettingsChanged event. The DisplaySettingsChanged handler changes the Window title and size of a WrapPanel to change the control layout.

```
public MainWindow()
{
    Microsoft.Win32.SystemEvents.DisplaySettingsChanged += new System.EventHandler(displaySettingsChanged);
}

private void displaySettingsChanged(object sender, EventArgs e)
{
    if (System.Windows.SystemParameters.PrimaryScreenHeight > System.Windows.SystemParameters.PrimaryScreenWidth)
    {

```

```

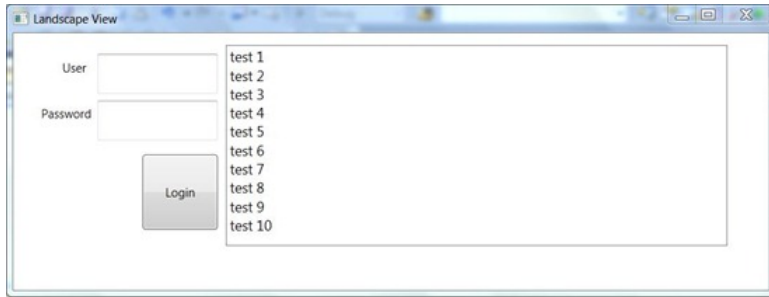
        //Run the application in portrait

        this.Title = "Portrait View";
        wrapPanel1.Width = 600;
        wrapPanel1.Height = 800;
    }
    else
    {
        //Run the application in landscape

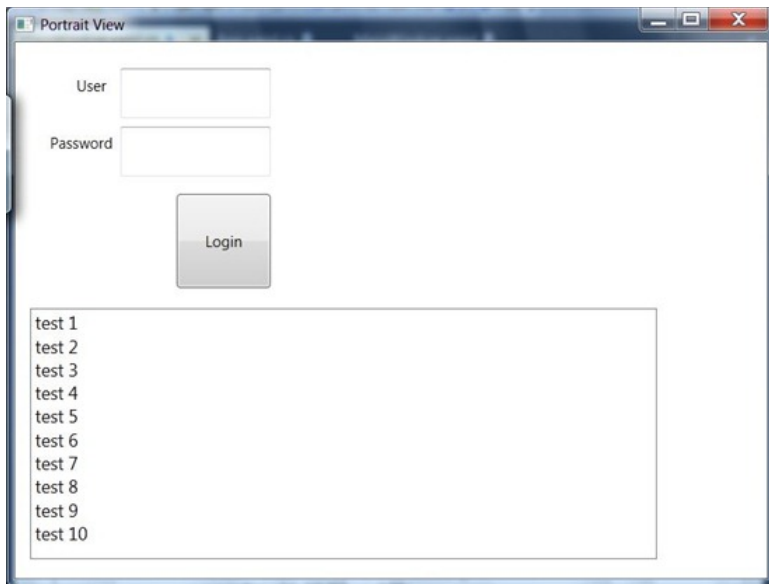
        this.Title = "Landscape View";
        wrapPanel1.Width = 800;
        wrapPanel1.Height = 600;
    }
}

```

Landscape view:



Portrait view:



For more information on detecting screen rotation, see [Detecting Screen Orientation and Screen Rotation in Tablet PC Applications](#).

Gestures, Manipulations, and Inertia

WPF is one of the easiest Windows development platforms that enable touch manipulation of objects. The [UIElement](#), [UIElement3D](#), and [ContentElement](#) classes expose events that occur when a user touches an element. Several other controls support touch-enabled scrolling. The [UIElement](#) supports manipulation which is interpreted to scale, rotate, or translate. I won't go into the details of implementing manipulations within this post because there is great information on this already. To get started, please review [Walkthrough: Creating Your First Touch Application](#) on TechNet.



Development Platforms for Windows 7 Touch

over 5 years ago by [Pat Altimore MSFT](#)

If you are planning on developing a Windows 7 Touch application, you may be wondering what development platform is the right choice for your application. Reviewing your requirements for your application and picking the right platform before you start can help you avoid roadblocks and design around limitations. Each development platform that is enabled for designing for touch has pros and cons. Here are some questions you should ask yourself before picking a development platform for touch.

Existing code and Developer Skillsets

- Are you a native code guy or .NET?
- Maybe you're targeting web development?
- Are you planning to reuse some code?
- Do you have an existing application?
- Maybe you plan on also targeting Windows 7 Phone and want to reuse some of the code for your phone app.

User Experience & Design Goals

- Are you targeting cross platform?
- What types of gestures make sense for your application?
- Single touch good enough or do you need multi-touch?
- Is there a requirement for user install experience?
- Performance & Target Form Factors.
- Do you need text entry? If you're targeting slates, you need to plan for no keyboard. You'll need to consider how text will be input.

Let's walk through some of the capabilities of each development platform. The following capabilities will be covered for each platform.

Capability	Description
Raw touch messages	Access to each touch point state and data associated (down, up, move, etc.)
Gesture support	Access to predefined touch actions (gestures) such as pan, pinch, press and hold, two finger tap, etc.
Manipulations and inertia	Allows combinations of translation, rotation, and scaling with multiple touch points. Manipulations are essentially gestures with values associated with them that describe the entire gesture. Ability to use inertia on objects after touch up.
Promotion to mouse messages	If manipulations or gestures aren't handled by your application, the touch input is converted into mouse messages. For example, an unhandled tap generates a left mouse click message. Another example would be an unhandled pinch gesture is generates a control mouse wheel message.
Deployment	How is the application installed? Is installation required?
TIP support	Ability to activate the Text Input Panel (TIP) . Some of the development platforms allow you to leverage the Text Input Panel (TIP) while others do not. If the development platform you choose doesn't support using the TIP, you will need to implement your own virtual keyboard for text input. See Todd Landstad's presentation for a good overview of leveraging the TIP.

Native (Win32)

If you want full control over your development, this platform has the least limitations to Windows Touch API's. However, as with any native code project, you may have to write more code because there is less framework to do the work for you.

Capability	Description
Raw touch messages	Yes. WM_TOUCH combines down, up, move and other states into one message.
Gesture support	Yes. WM_GESTURE message is generated for common gestures.
Manipulations and inertia	Yes. For manipulations, the application needs to implement the IManipulationProcessor interface. For inertia, the IIInertiaProcessor interface can be

	used in conjunction with the <code>IManipulationProcessor</code> interface.
Promotion to mouse messages	Yes. If you don't handle a touch or gesture message, <code>DefWindowProc</code> will generate a corresponding mouse message.
Deployment	Installer required. Administrative rights required for install.
TIP support	Yes. TIP can be invoked by using creating and showing a <code>Caret</code> . However, the TIP can't be displayed in a full screen DirectX application.

More info on native touch development: [Windows Platform SDK – Windows Touch](#)

.NET

.NET has two choices: Silverlight and WPF. Depending on the functionality of your application and your requirements, you may want to pick one of the development platforms over the other.

Silverlight 4

Silverlight can be hosted in the browser or out of the browser. For “in browser”, Internet Explorer forwards platform touch messages to plug-ins such as Silverlight that are running within Internet Explorer. For “out of browser”, Silverlight registers for touch input but there are some limitations when running out of browser (see below).

Capability	Description
Raw touch messages	Yes (with some exceptions). Silverlight registers for raw touch input. Silverlight processes touch messages at the level of the raw message, analogous to the platform <code>WM_TOUCH</code> message. However, touch input is not supported for Silverlight out of browser applications running in full-screen mode. For full-screen mode, you must rely on promotion to mouse messages.
Gesture support	No. If your requirements include gestures, you have to process touch input into gestures using your own application code, within the context of Silverlight.
Manipulations and inertia	No.
Promotion to mouse messages	Yes. Silverlight generally promotes raw touch events to mouse messages for legacy support.
Deployment	In browser: No installation required but Silverlight add on is a prerequisite; Out of browser: Elevated trust install.
TIP support	In browser: Yes. Out of browser: No.

More info on Silverlight touch development: [http://msdn.microsoft.com/en-us/library/dd894494\(VS.95\).aspx](http://msdn.microsoft.com/en-us/library/dd894494(VS.95).aspx)

WPF 4

Windows Presentation Foundation has support raw touch data as well as manipulation and inertia. WPF applications can detect and respond to touch in a manner similar to responding to other input, such as the mouse or keyboard, by raising events when touch occurs.

Capability	Description
Raw touch messages	Yes. The <code>UIElement</code> , <code>UIElement3D</code> , and <code>ContentElement</code> classes expose events that occur when a user touches an element. Several other controls support touch-enabled scrolling.
Gesture support	Yes through manipulations.
Manipulations and inertia	Yes. The <code>UIElement</code> supports manipulation. A manipulation is interpreted to scale, rotate, or translate.
Promotion to mouse messages	Yes. By calling the <code>Cancel</code> method on the event arguments in a manipulation event.
Deployment	Click once installer.

TIP support	Yes.
-------------	------

More info on WPF touch development: [Touch and Manipulation Input for .NET 4](#)

HTML 5

[HTML 5](#) is still under development but is quickly being supported by all modern browsers. While it is not targeted for touch, it is an option for cross platform, cross browser, web based applications.

Capability	Description
Raw touch messages	No.
Gesture support	Enabled through browser. More info on IE9 support.
Manipulations and inertia	No. Inertia enabled through browser for panning and scrolling.
Promotion to mouse messages	N/A. Hosted in browser.
Deployment	Hosted on website. Not required.
TIP support	Yes.

Final Advice

The [Windows Touch Guidance](#) whitepaper is an excellent resource for details about developing for Windows Touch. A must read prior to starting your touch development.



Troubleshooting Interactive Services Detection

over 5 years ago by [Pat Altimore MSFT](#)

34

My earlier post on Interactive Services Detection is very popular. It was targeted toward developers. Based on the comments I receive, I think most readers are users looking for some help. If you're a developer, make sure to check out my previous [post](#). If you are just a user trying to figure out why Interactive Services Detection is blinking at you and how to possibly troubleshoot the issue, this post is for you.

Why is this happening (when I boot; every 5 minutes; intermittently)?

First, some background... There are two main types of "processes" that run on Windows – desktop applications and services.

Desktop applications "interact" with the user through a user interface (windows, dialogs, etc.).

Services run in the "background" and do system stuff and communicate with other applications.

Services should not interact with the user. Windows 7 and Vista have a boundary that isolates services from trying to interact with the user. If the service tries to present some kind of user interface, you will get the Interactive Service Detection flashing toolbar button.



There could be a couple of reasons why this is occurring. However in most cases, the service experienced an "[unhandled exception](#)". Unhandled exceptions are errors that the programmer didn't expect and didn't handle with code. Most people call this a "crash".

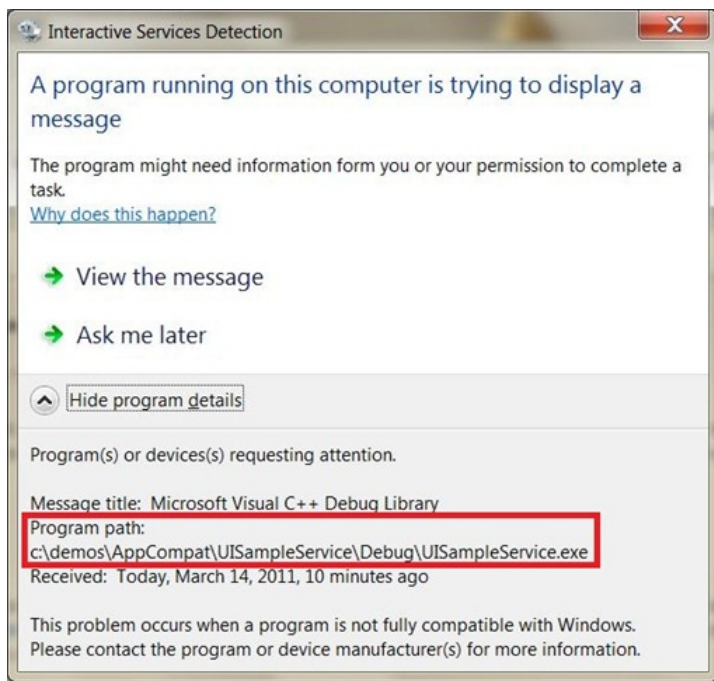
Services generally start at boot time. Therefore, if the service is crashing when it starts, you will see the interactive services detection button at boot time. If you ignore the notification or select "Ask me later", Interactive Services Detection will notify you again in 5 minutes.

If you only see the Interactive Services Detection intermittently, this is very likely the service crashing due to some unexpected condition. For example, I have 3rd party printer software on my personal laptop that crashes occasionally and I get a Interactive Services Detection notification. I'll keep the brand name of the printer to myself 😊.

Identifying the Service

As a user, your best option is to identify what service is causing the notification. If you know what software is causing the notification, you can check the manufacturer's website for an update or contact support.

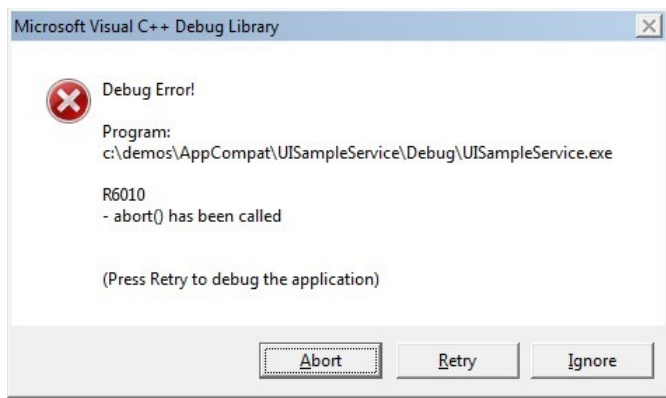
The first thing to inspect is the Interactive Services Detection dialog. Be sure to click the "Show Program Details" button.



This is a message from a service I created that has an intentional unhandled exception. If you look at the “Program path” this can give you some clues about the service that is crashing. The path can give you the company name, application, and most importantly the executable name. In most cases, you can search for the executable name (e.g. UISampleService.exe) using your favorite search engine.

More Clues

Next, you should select “View the Message”. This will take you to the “secured desktop” where you can read the message.



Here we can see some additional information about the crash. Collect this information and try searching using some of this information. If this was a real application, I would type the following into a search engine: **UISampleService.exe “Microsoft Visual C++ Debug Library” R6010 “abort() has been called”** and see if you can find any support articles, fixes, discussions, or updates for the issue.

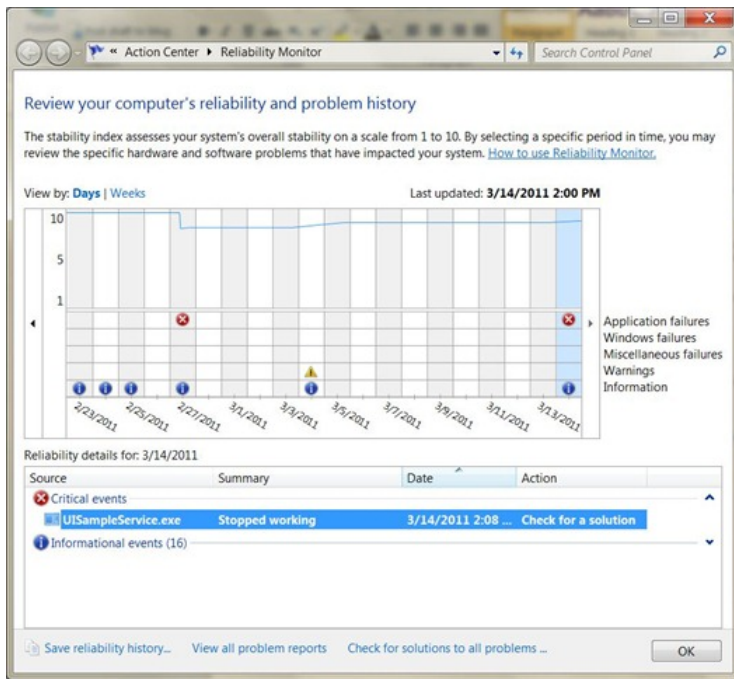
If you can’t find anything with a search, you may want to consider contacting support for the product that is having the issue. Keep all the information you collected so far to pass along to the support engineer. Also, feel free to point the support engineer to this blog. 😊

Even More Clues...

If it is a crash that is causing the issue, you may also want to investigate what information is being collected. The best way to see the history of the problem is with Reliability Monitor. Type “reliability” in the Start menu search box and click the “View reliability history” link.



This utility allows you to see the history of all the issues on your computer. If a service is crashing, it should show up here. We can see that there is an entry for my sample crashing service, UISampleService.exe.



In this utility, you can identify when and how often the crash is occurring. You may want to try the "Check for a solution" action for the error. This will check to see if the application vendor posted a solution for this issue on the Windows Error Reporting system.

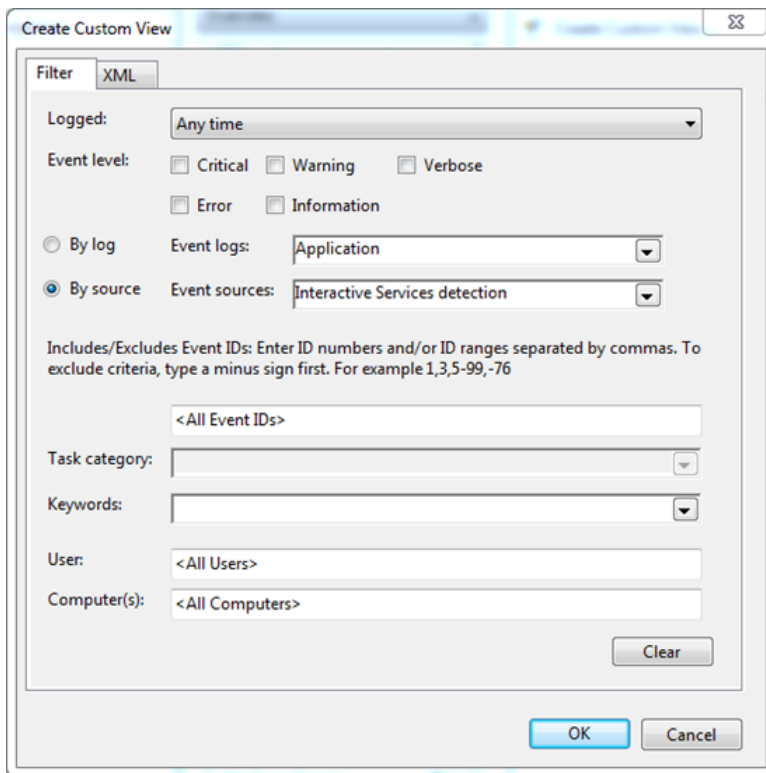
The Tile Button is Displayed Too Quickly and I Can't Click it Fast Enough

I received this question a few times in the comments. Great question! Here's a technique you can use to troubleshoot the issue. You will need to create a custom view in the event viewer.

From the Start menu search box, type **eventvwr** to start the Event Viewer.

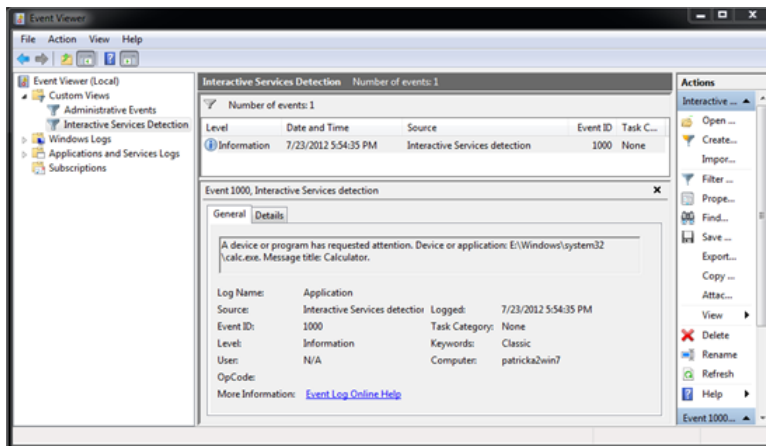
In the tree view in the left pane, right click **Custom Views** and select **Create Custom View...**

In the Custom View dialog, select the "By Source" radio button and then select **Interactive Services detection** in the "Event sources" drop down box.



Click **OK**. In the "Save Filter" dialog, name your custom view. e.g. "Interactive Services Detection". Click **OK**. You are now configured to capture these events when they occur.

If you ever see the Interactive Services Detection tile appear on the taskbar, you can return to the Event Viewer (eventvwr) and investigate the information that has been logged. Here is an example event where I "tricked" calculator to run in session 0 to cause the issue.



Can I Disable the Notification?

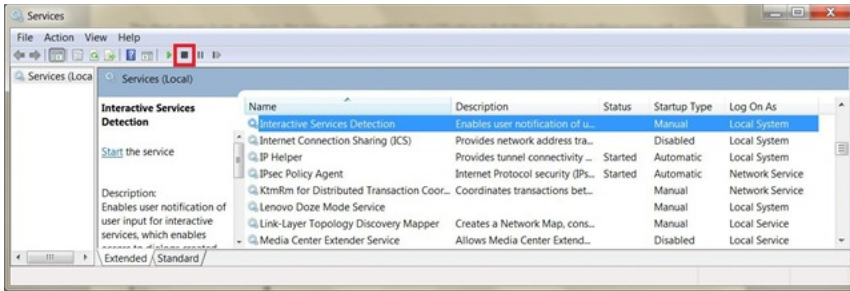
The short answer is yes but it is not recommended. The likely reason for the notification is there is something wrong with a service on your computer. You can disable the notification but you are not fixing the problem. Some good analogies would be putting black tape over the "Check Engine" warning light on your car dashboard or removing the battery from your smoke detector.

The Interactive Services Detection notification is initiated by the Interactive Services Detection Service. If you disable this service, you will no longer receive the notification for the problem service. Also, you will not receive notification for any misbehaving service on your system.

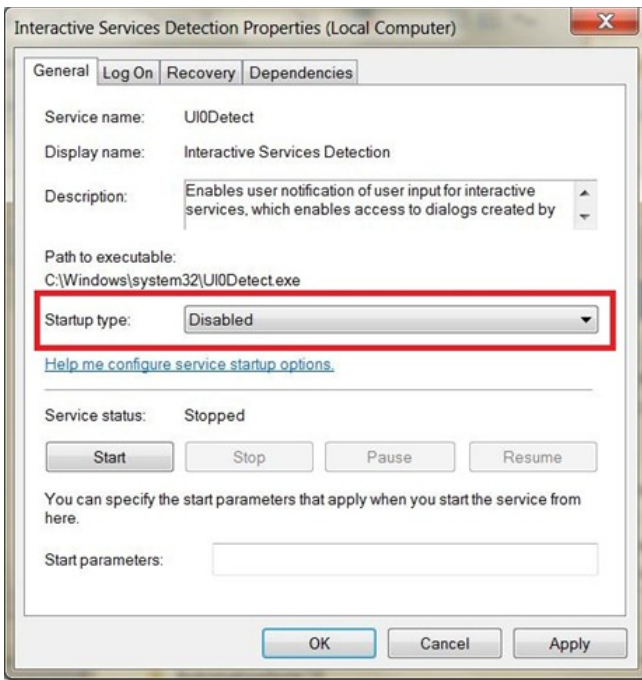
To disable the service, open the Services control panel by typing **services.msc** in the Start menu search box.



In the Services control panel, find "Interactive Services Detection" in the list. Click the "stop" button to stop it if it is started.



Double click the Interactive Services Detection entry to open the Properties dialog. Set the "Startup type" to **Disabled**.



You are now blissfully unaware of any misbehaving services.



Developing for Windows Touch? Windows Touch Guidance Document Now Available

over 5 years ago by [Pat Altimore](#) MSFT

Did you know Windows 7 has built in multi-touch API's? Have you been wondering how to convert your current application to be touch enabled? Perhaps you want to develop the next cool slate PC app? Maybe you would like to start developing for touch but are wondering how to get started in C++, Silverlight, or WPF.

There are several resource documents available and the [Windows Touch Guidance](#) document does a great job pulling all of the information into a single document including the following:

- Design Principles - Recommendations on application design
- Windows 7 Touch APIs - Windows 7 Touch, Gesture, and Manipulation APIs
- Developer Platform Choices - Windows 7 developer platform support for Windows Touch
- Windows Touch developer resources



Troubleshooting Application Compatibility Issues – Tools, Tips, and Tricks

1

over 6 years ago by [Pat Altimore MSFT](#)

When I need to troubleshoot a compatibility problem, there are a handful of tools that I use that are very helpful. In this post, I'll talk about some of my favorite tools and how to use them to identify compatibility issues.

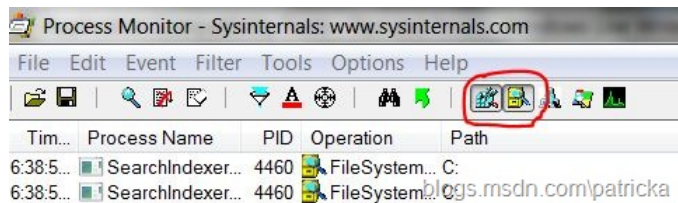
Most applications do a poor job of reporting unexpected errors. A lot of times, the developer didn't plan for the error and it gets bubbled up as a misleading error or a crash to the user. Compatibility issues tend to fall in the unexpected category. I've categorized the usual causes of these expected issues and the troubleshooting tools I use.

UAC: Permission Issues

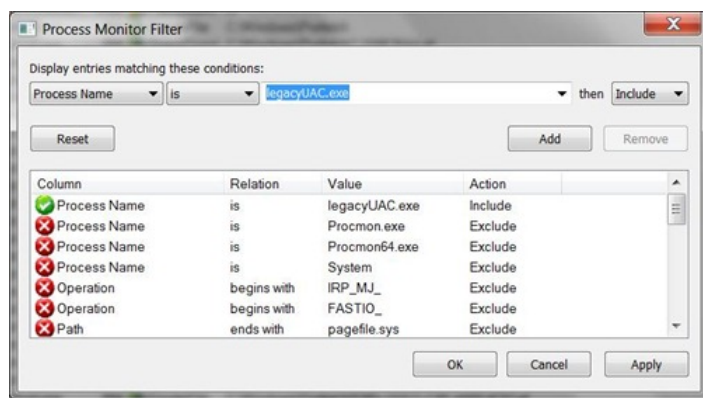
Application compatibility permission issues are usually a result of applications not being designed for standard user. The application assumes that the user has full administrator privileges and has issues running with User Account Control (UAC) enabled.

[Process Monitor](#) is the first tool you should use in identifying a possible permission issue. Process Monitor logs all file and registry activity (among other things). This can be very handy when looking for a permission issue.


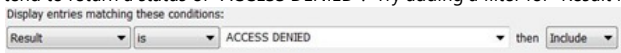
Start Process Monitor prior to the issue occurring. Make sure you are capturing at least the file and registry activity.



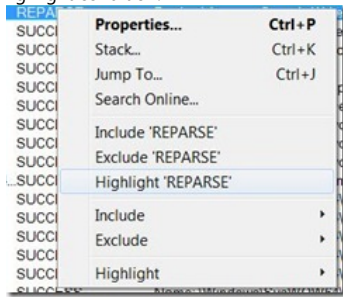
Since this tool logs everything that is happening on the system, make sure to take advantage of the filtering feature. You may want to filter on PID or Process Name. For example, I'm only including the process name "legacyUAC.exe".



Tips:

- You probably have a good idea about what file or registry is giving you a problem. You can use the "find" function  to find the file or key.
- Permission issues tend to return a status of "ACCESS DENIED". Try adding a filter for "Result is ACCESS DENIED":

- UAC virtualization will be indicated by a result of "REPARSE" followed by a log entry to the virtual store location. See this post for [details on UAC virtualization](#).

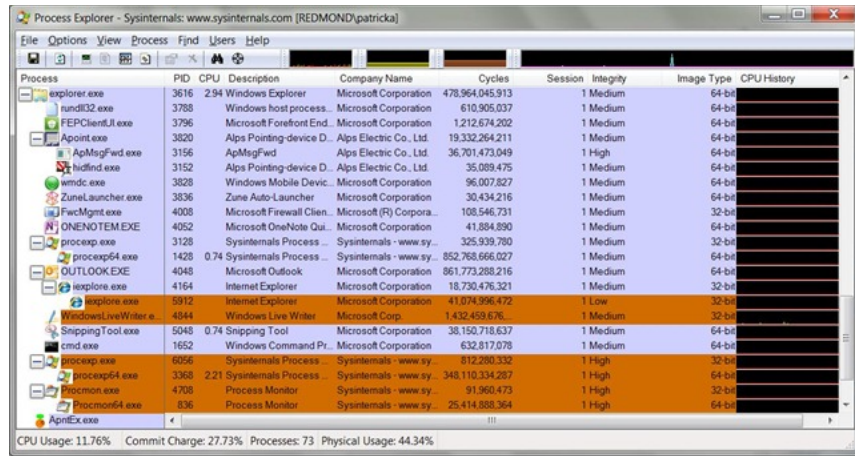
- Sometimes highlighting can be more useful than filtering. For example, you may want to see the log entries that occur around the same time as another log entry. REPARSE is a good example. Highlighting works like filtering. You can get to the highlighting dialog from the menu. A shortcut to add highlighting (and filtering) is to right click on the column/row and add the highlight condition.



UAC: Integrity Level Issues

If you have applications that communicate to other processes especially via Windows messages, you may run into [Mandatory Integrity Level \(MIC\) issues](#).

[Process Explorer](#) is a great tool for a lot of troubleshooting tasks. If you need to see at what integrity level a process is executing, Process Explorer has a column that reports the integrity level of running processes. Simply check "Integrity Level" under View | Select Columns... in the menu. Here's an example screenshot:

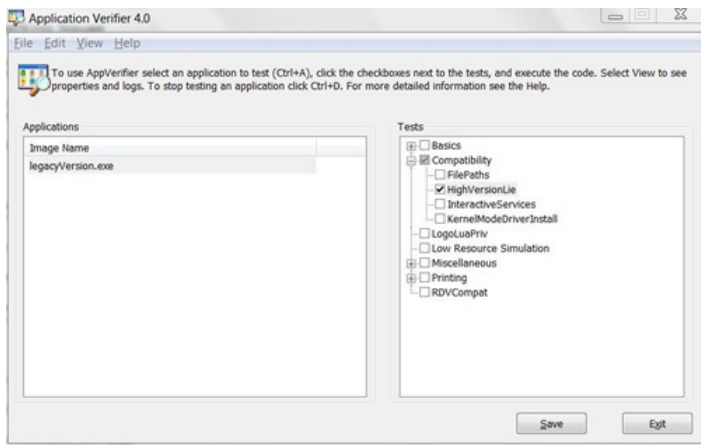


Version Checking Issues

[Version Checking issues are a very common compatibility issue](#). Applications will check the OS version and make decisions on whether to run or exit, use certain APIs, perform functionality, etc. Depending on how the version check logic is written, the code may make an incorrect choice on a new OS Version.

[Application Verifier](#) is one of the best tools to "lie" about the OS version to an application. Application Verifier intercepts Windows version API's and returns any version number you want to specify. So, if I'm having an issue, I can set the version number to a previous OS to verify I have a version checking issue.

Just add your application to the application list Application Verifier will monitor and test. Check the HighVersionLie test **only** for version checking. Application Verifier has a lot of test that focus on reliability. You may get unexpected crashes if you do some of the default memory tests.



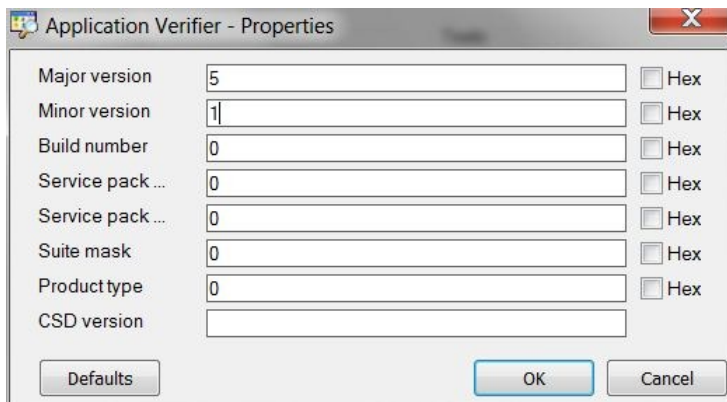
By default, the HighVersionLie test increments the major and minor OS version numbers by 2. Windows 7 is version 6.1. Therefore, your application will think it's running on OS version 8.3. Here's sample output from my legacyVersion sample app:

```
>legacyVersion.exe
OS Version using GetVersion (Major.Minor.Build): 8.3.8600
OS Version using GetVersionEx (Major.Minor.Build): 8.3.8600

MajorVersion==5 is FALSE. I'm NOT Win2000 or XP
MajorVersion==5 && MinorVersion==1 is FALSE. I'm NOT XP
MajorVersion==6 is FALSE. I'm NOT Vista
MajorVersion==6 && MinorVersion==0 is FALSE. I'm NOT Vista
MajorVersion>=5 && MinorVersion>=1. I'm XP or greater

*** Most correct version check ***
MajorVersion>=6 && MinorVersion>=0. I'm Vista or greater
```

If you want to test some other version number, right click the HighVersionLie test and select properties. Enter the version you want to test:



Note: Application Verifier always attaches to the applications in its list regardless if the UI is active. Remember to delete the application from the list when you are done troubleshooting.

The nice thing about Application Verifier is that you can set the version number to anything. This can be useful in determining if your app has any version checking issues on a future OS. By default, the HighVersionLie test increments both the major and minor version numbers by 2.

Checking Manifests

[Sigcheck](#) is a signature checking tool. However, the `-m` switch allows you to dump the manifest. This is a quick way to see an application manifest for an executable.

The application manifest is a resource in the executable. You can also use a resource editor to view the manifest. With a resource editor, you could also modify the manifest which could help in debugging the issue. For more information on manifests, [see my manifest post](#).

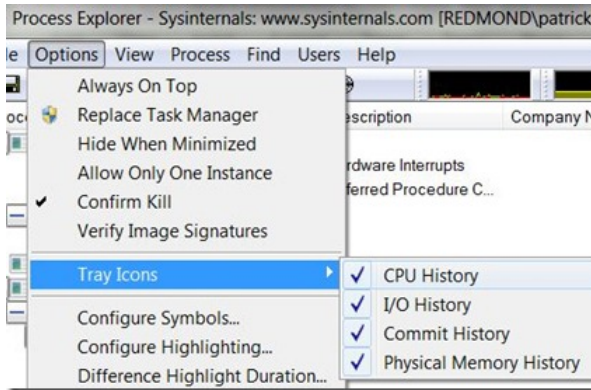
Performance

If you are trying to uncover a performance issue, [Process Explorer](#) can help you narrow down the issue quickly. I run Process Explorer with the system tray icons. This allows you to monitor thumbnail

performance graphs for the system. If you notice a high value or spike, you can hover over the icon and get a possible clue of what might be causing the issue. In this screenshot, you can see I hovered over the CPU icon and we see the suspicious application CPUHog.exe consuming 49% of the total CPU.



To turn on the tray icons, select the icons you would like to see from the Options menu.



Another handy performance option in Process Explorer is to use the CPU History columns. This will show the CPU history graph for all processes. This can be useful if you have intermittent CPU spikes. Just add the CPU History column in the View | Select Columns menu.

A screenshot of the Process Explorer application window. The 'View' menu is open, and the 'Select Columns' sub-menu is also open. The 'CPU History' column is selected. The main window shows a list of processes with the 'CPU History' column added. The table below shows the data for the selected processes.

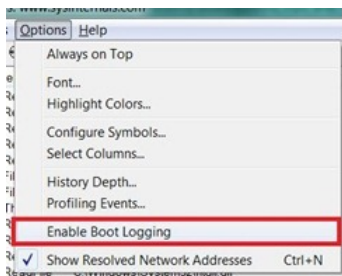
Process	PID	CPU	Description	Company Name	CPU Time	CPU History
CPUHog.exe	5776	49.25			0:00:20.295	
System Idle Process	0	43.19			13:14:59.177	
procexp64.exe	5688	3.79	Sysinternals Process ...	Sysinternals - www.sy...	0:00:18.595	
dw.exe	3568	1.52	Desktop Window Man...	Microsoft Corporation	0:03:47.730	
Interrupts	n/a	0.76	Hardware Interrupts		0:15:14.727	
explorer.exe	3608	0.76	Windows Explorer	Microsoft Corporation	0:03:41.786	

If you need advanced performance analysis, you can use XPerf in the [Windows Performance Toolkit \(WPT\)](#). XPerf allows you to capture and view Windows event data for everything that is going on in the operating system. This is a very powerful tool that you can obtain very detailed information. It may take some practice before you get proficient with this tool.

Boot Performance

If you are encountering slow boot times, you may need to analyze what is in the boot path to pinpoint the problem. Process Monitor provides a quick and easy way to log boot time activity.

Select Options | Enable Boot Logging



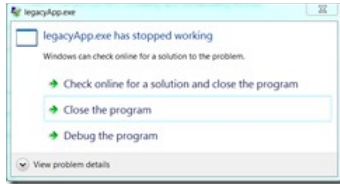
This will log the file, registry, profiling information during your next boot. You can then review the log to identify possible applications that are consuming a lot of resources during the boot cycle.

If you are interested in in depth boot performance analysis, you can use the Windows Performance Toolkit (WPT). See the [Windows On/Off Transition Performance Analysis whitepaper](#) for details on how to use WPT to analyze boot performance.

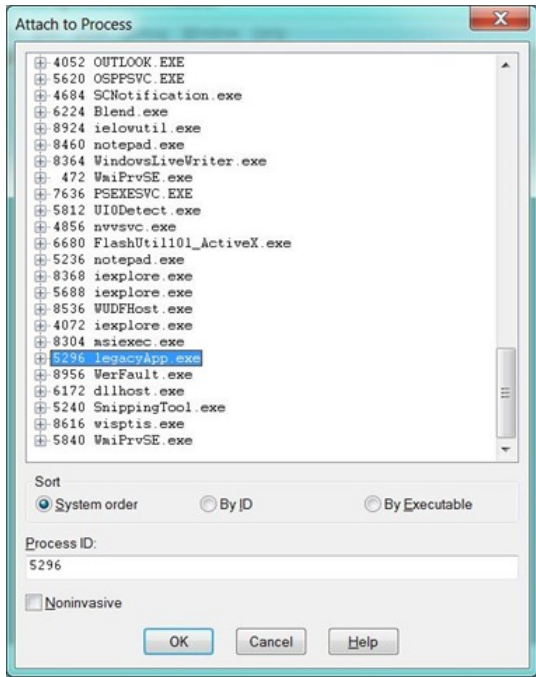
Crashes

For crashes, you will need to use a debugger. I generally use [WinDbg](#).

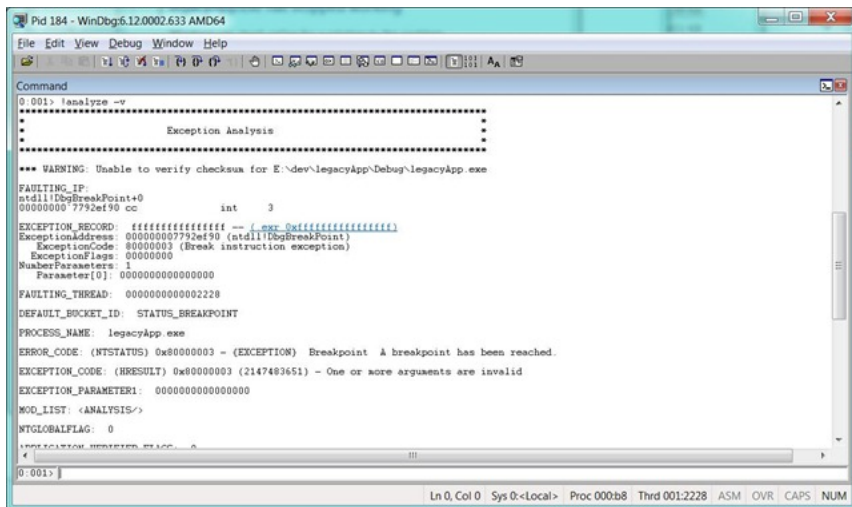
If you can reproduce the crash, don't dismiss the crash dialog. By leaving the dialog, the process still exists and you can attach to the process with the debugger and diagnose the exception.



Press F6 and attach to the process.



Ideally, you can load your symbols at this point. If you have debug symbols for the application, the debugger can give you more information about the exception. Once you have your symbols loaded, try a `!analyze -v` command. The debugger will try to give you some analysis of the exception.



In this case, it looks like we have an invalid argument.

Saving dumps

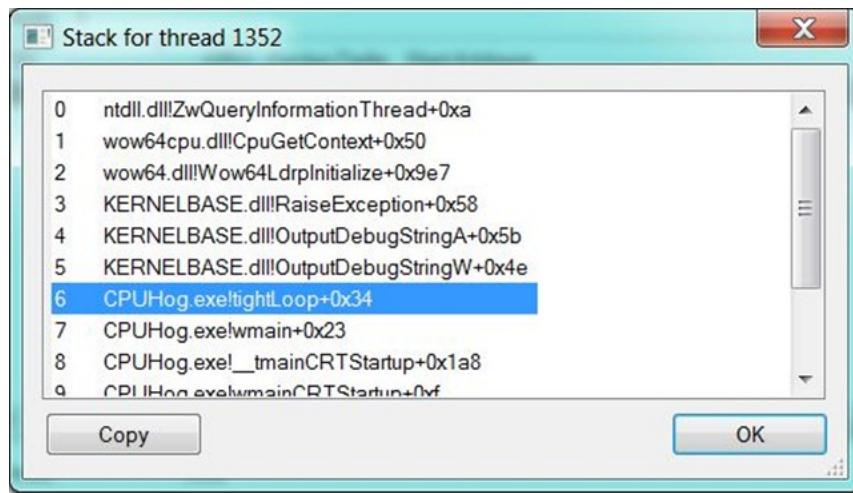
Sometimes, you may want to save dumps for analysis later. Maybe it's on a non-development machine or you want to collect several dumps. You can configure Windows 7 and Windows Server 2008 R2 to always generate and save a dump file.

- Create a key named: HKLM\Software\Microsoft\Windows\Windows Error Reporting\LocalDumps
- Dumps will default to the following directory: %LOCALAPPDATA%\CrashDumps
- You can override the default with a DumpFolder value (REG_EXPAND_SZ)
- You can also limit the number of dumps saved with a DumpCount value (DWORD)

For more info see: <http://blogs.technet.com/b/askperf/archive/2008/02/05/ws2008-windows-error-reporting.aspx>

What about Hangs?

For hangs, the first tool I use is Process Explorer. If an application is slow and unresponsive, I start Process Explorer and use crosshair on the toolbar and drag and drop on to the unresponsive window. This will highlight the process in the list. I then double click the process in the list and then look at the Threads tab. You can then double click the threads to look at the stack. This usually points you in the right direction. Here we can see that in CPUHog.exe, the tightLoop routine is probably a good place to start looking for an issue:



Error codes

Sometimes it's easy to forget that there's a tool to look up error codes. Err.exe in the SDK. This utility can be used to look up what the description for a particular code. Err.exe uses error codes defined in header files. It exists in the Bin directory of the SDK. e.g. C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin.

For example, let's look up error code 5.

```
C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin>err 5
# for hex 0x5 / decimal 5
BTH_ERROR_AUTHENTICATION_FAILURE          bthdef.h
INVALID_PROCESS_ATTACH_ATTEMPT             bugcodes.h
CDERR_LOADSTRFAILURE                       cderr.h
CR_INVALID_DEVNODE                         cfgmgr32.h
DHCP_DROP_UNAUTH                          dhcpssdk.h
LLC_STATUS_PARAMETER_MISSING               dlcati.h
HIDP_GETCOLDESC_PREPARSE_RESOURCES         hidpddi.h
IAAPI_TOOBIG                              iaapi.h
# /* Value exceeds size constraint */
MD_ERROR_SUB400_INVALID_TRANSLATE          iiscnfg.h
MD_ERROR_SUB401_APPLICATION                iiscnfg.h
MD_ERROR_SUB403_SSL128_REQUIRED            iiscnfg.h
MD_ERROR_SUB404_URL_SEQUENCE_DENIED        iiscnfg.h
MD_ERROR_SUB503_CONNECTION_LIMIT          iiscnfg.h
IME_RS_TOOLONG                             ime.h
# given string is too long
KDC_ERR_S_OLD_MAST_KVNO                    kerberr.h
# 5 Server's key encrypted in old master key
RSVP_Err_BAD_STYLE                         lpmapi.h
# /* Conflicting style */
POLICY_ERRV_IDENTITY_CHANGED               lpmapi.h
POLICY_ERRV_SUBNET_DEF_FLOW_COUNT          lpmapi.h
MAPI_DIAG_MAXIMUM_TIME_EXPIRED            mapidefs.h
SE_CATEGID_DETAILED_TRACKING               msaudite.h
# Detailed Tracking
MSIDBERROR_UNDERFLOW                      msiquery.h
# data less than minimum value allowed
```

```

NRC_CMDTMO                                nb30.h
# /* command timed out                      */
NDDE_INVALID_SHARE                        nddeapi.h
NMERR_NO_MORE_FRAMES                      netmon.h
SMART_INVALID_DRIVE                       ntdddisk.h
# Drive number not valid
DS_NAME_ERROR_DOMAIN_ONLY                 ntdsapi.h
SAM_PWD_CHANGE_NOT_COMPLEX                 ntsam.h
ODBC_ERROR_INVALID_REQUEST_TYPE            odbciinst.h
OLE_ERROR_STREAM,                         ole.h
# (OLESTREAM) stream error                 */
MFE_OIF_PRUNED                            routprot.h
# no downstream receivers exist on oif
SCESTATUS_BUFFER_TOO_SMALL                scesvc.h
SE_ERR_ACCESSDENIED                       shellapi.h
# access denied
SNMP_ERRORSTATUS_GENERR                   snmp.h
SNMP_GENERICTRAP_EGPNEIGHLOSS             snmp.h
TWCC_OPERATIONERROR                       twain.h
# /* DS or DSM reported error, app shouldn't */
CMC_STATUS_CONFIRM_REQUIRED               wincrypt.h
CMC_FAIL_UNSUPPORTED_EXT                  wincrypt.h
ERROR_ACCESS_DENIED                       winerror.h
# Access is denied.
LDAP_COMPARE_FALSE                        winldap.h
SNMP_ERROR_GENERR                         winsnmp.h
# as an HRESULT: Severity: SUCCESS (0), FACILITY_NULL (0x0), Code 0x5
# for hex 0x5 / decimal 5
ERROR_ACCESS_DENIED                       winerror.h
# Access is denied.

```

We can see there are lots of definitions for the error code of 5. Depending on what header files you include in your code, you can figure out what the actual error is. If you don't have access to the code, you can usually make an educated guess on which error you are encountering.

I hit on most of the tools I use to solve most of the issues I've seen. I really hope this helps you find issues and be able to solve them.

Pat



If I'm Using Internet Explorer 9 Beta/RC, What Should I Do If a Site is Broken or Hangs or Crashes, etc.?

0

over 6 years ago by [Pat Altimore MSFT](#)

If you haven't noticed, IE9 public beta has been released ([download it here](#)). Please, don't be scared by the title of my post and avoid using the beta. I'm going to step through a few compatibility tricks to make sure you can enjoy the beta until the rest of the web gets ready for IE9.

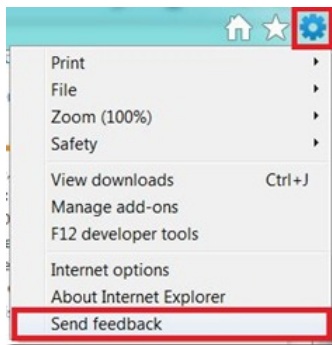
Why Don't Some Sites Work?

By default, IE9 will display sites using a new rendering engine and script engine. These engines support the latest web standards and are different than what IE8 supported. Most sites do support the latest browsers that implement latest standards. However, the most common issue is sites use browser detection logic to decide what content to deliver to a browser. If the site delivers IE8 specific content instead of standards based content, the site may appear broken. Over time, site developers will update their sites to fix this mismatch.

Submit Feedback

If you find a site that doesn't work, the IE encourages you to submit feedback. This helps identify bugs and identify top sites that may have issues. You can do this two ways.

Option 1: Use the Send Feedback menu option in IE9 Beta by clicking the Tools "Gear" | Send Feedback option:



Option 2: Submit feedback through the Microsoft beta program site for IE:

<https://connect.microsoft.com/ie>

Quick Fix #1 – Compatibility View

IE9 includes the brand new rendering engine but it also includes two previous rendering engines as well. So, you can display individual sites in "Compatibility View" and it should work as it did in IE before.

To add a site to Compatibility View, click the Compatibility View button in the address bar:

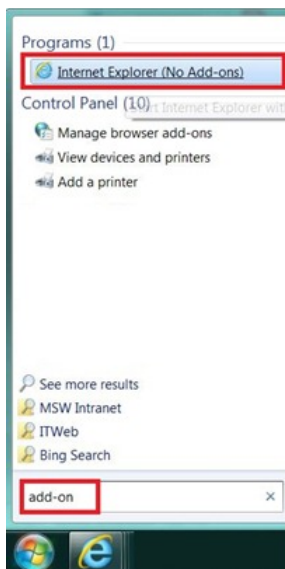


This will cause the IE to behave a lot like IE7. Note that we get the "Download Now" button because the site thinks we are running IE7. If you select the Compatibility View button, IE9 will add this site to the local compatibility view list and will remember next time you visit the site to behave like IE7 (until you uncheck the compatibility view button). This is a bit of a "big hammer" approach. However, in most cases, this will fix most issues until the site developers update the site to be IE9 friendly.

Quick Fix #2 – Disable Add-ons

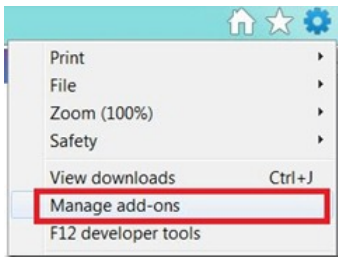
Add-ons are things like toolbars and browser helper objects (BHO). They are extension applications that provide extra functionality to the browser. You probably have a bunch of add-ons that were added by software applications that you didn't even know were there. After installing the beta, misbehaving add-ons can cause hangs, crashes, performance issues, etc.

To quickly find out if you have an add-on issue, launch IE in No Add-on mode. Search for "add-on" in the Start menu:



Browse to the site with the issue. If the issue doesn't occur, you probably have an issue with an add-on.

The next step is to figure out what add-on is causing the issue. To manage add-ons, click the Tools "Gear" | Manage Add-ons option:



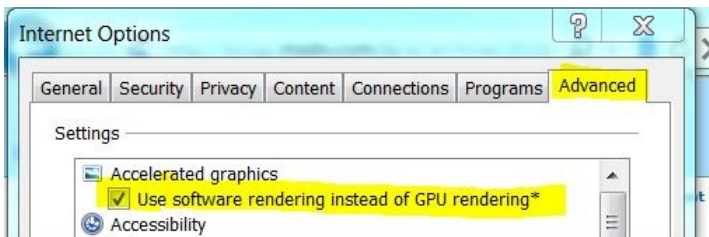
In the Manage add-on dialog, try disabling individual add-ons. You should be able to figure out the issue by a simple trial and error process.

Quick Fix #3 – Update Graphics Driver / Disable Hardware Acceleration

Internet Explorer 9 takes [advantage of graphics card to improve performance through hardware acceleration](#). If you don't have the latest driver installed for your graphics card, you may run into problems with graphics acceleration.

The first option you should try is check your hardware manufacturer's website for the latest graphics card driver. Be sure you have the latest driver installed for your graphics card.

The second option to try is to disable hardware acceleration. This can be found in Tools "Gear" | Internet Options | Advanced tab.

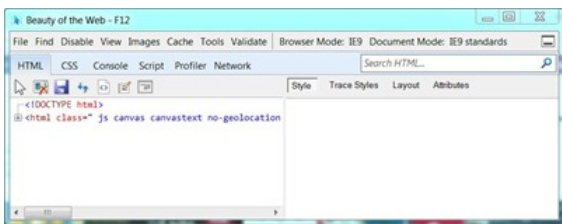


Check the "Use software rendering instead of GPU rendering" checkbox. This will force IE to disable graphics hardware acceleration. If this solves the problem, you may need to wait until your graphics hardware vendor releases a new driver before you can use hardware acceleration in IE9.

More Details for Developers (and the Curious)

If you're a site developer or want to try to figure out more about the issue, the built in [developer tools](#) can be very helpful.

Browse to the site and press F12.



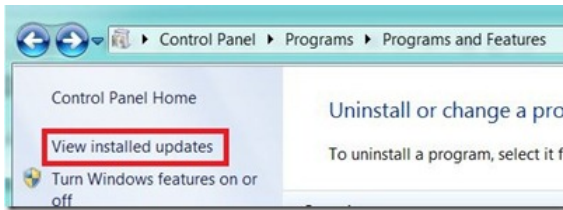
Notice the top menu contains the Browser and Document Mode. These modes control how IE9 renders the content. You can also use Tools | Change user agent string to try tricking browser detection logic. For example, if you change the user agent string to Chrome, the web site will likely send standards based content because it thinks the browser is Chrome.

See [Compatibility Features for Developers](#) for more information.

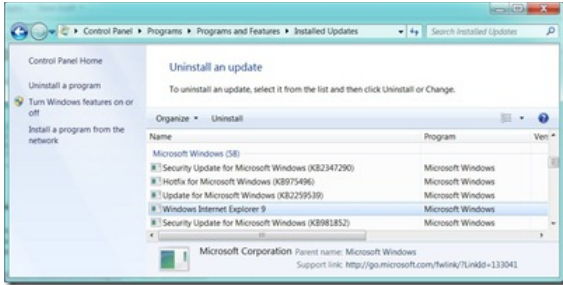
Uninstalling

I hope I've given you enough information to work through any issue. But I understand that you might want to uninstall IE9 and revert back to IE8. If you choose to uninstall IE, here are the steps:

1. Open Control Panel | Uninstall a Program.
2. Click "View installed updates" in the left pane:



3. Find "Windows Internet Explorer 9" in the list. You can use the search box to quickly find it. Click Uninstall in the menu bar.



I hope this information helps while using IE9 Beta!



Leveraging Windows 7 Client Software Logo Toolkit to Test Your Application

0

over 6 years ago by [Pat Altimore MSFT](#)

You may have heard of [Windows 7 Software Client Logo](#). Did you know that it has an automated test? Did you know the toolkit was designed to be scripted so you can add it to your testing process? In this post, I'll walk through how you might want to leverage the logo toolkit in your development or testing process to find potential compatibility issues.

What's Logo?

Windows Logo has been around since Windows 2000. There is a hardware and a software logo and it changes with each release of the OS. Basically, it is a [set of requirements](#) that all "well behaved" applications should meet. There is a [set of tests](#) that your application needs to pass to prove that it meets the requirements. If you pass the tests, you get the logo to put on your product. The good news is that the Windows 7 client logo is a self test. If you pass, it's free to submit and get the logo.



That's Great but I Just Want to Leverage the Tests...

There may be several reasons why you are interested in the logo -- marketing, partner points, your boss told you to do it, etc. Or maybe, you are just looking for ways to test your application for a base level of Windows 7 compatibility. What ever the reason, the goal of logo is to increase the quality of applications. I think one of the best improvements to logo is to have a tool to do the testing for you.

Using the Toolkit

Everything you need for logo including the toolkit is on [Connect](#). After you install the toolkit, you can run it interactively from the Start menu (Windows 7 Client Software Logo Toolkit shortcut). This will launch a wizard that walks you through adding and removing your application while the toolkit monitors the system for changes. At the end of the wizard, you can generate a report that looks like this:

Windows 7 Software Logo Toolkit Test Results

Application Name: OEM Ready Demo
Application Version: 1.0.0
Application Publisher: Microsoft

Summary: Incomplete - This is an interim report; please continue with the Toolkit wizard to finalize the report.

1. Clean, reversible, installation

Test case: Write appropriate Add/Remove Program values: **PASS WITH WARNINGS**

- **WARNING:** Applications are expected to create these registry entries Display.Name, InstallLocation, Publisher, UninstallString, VersionMajor*, and VersionMinor*. This application did not create the following registry entries:
 - Value InstallLocation missing or invalid for program OEM Ready Demo.
- **IMPACT IF NOT FIXED:** A user might remove an application not only to free up disk space, but also to return the computer to its state prior to the application being installed. Failure to restore the machine to its original state is a poor user experience. Also applications that do not create the above registry entries will not be found by enterprise inventory tools, and may experience issues in OS migrations and or upgrades scenarios. Windows telemetry tools may not accurately report information about your application.
- **HOW TO FIX:** You can supply all of the information needed to configure Add/Remove Programs in Control Panel by setting the values of certain installer properties in your application's Windows Installer package. Setting these properties automatically writes the corresponding values into the registry. The latest information and best practices that illustrate how to do this can be found at these links [1](#) and [2](#).

Test case: Do not force an immediate reboot during installation: **PASS**

Test case: Do not force an immediate reboot during uninstallation: **PASS**

Test case: Remove all non-shared files and folders: **PASS**

2. Install to the correct folders by default

Test case: Do not write to the %WINDIR% or %SystemDrive% folders: **PASS**

Test case: Install to Program Files: **PASS WITH WARNINGS**

- **WARNING:** This application wrote the following files to a location other than %ProgramFiles%:
 - Program OEM Ready Demo fails due to missing install location.

blogs.msdn.com/patricka/

The report outlines what tests you passed or failed with details on any issues found with respect to the requirements.

Automating the Toolkit Tests

The toolkit process can be accomplished via the command line. Therefore, you can script the process for your application. By scripting the logo test, you could put it in your development or testing process to find issues. Since the generated report is XML, you can parse it to generate reports or populate tracking databases, etc. We use the toolkit in our readiness lab process to look for issues before a customer visits the lab (see [this post about our Win 7 compatibility checklist](#)).

You can get the command line usage information by running "**C:\Program Files\Microsoft Windows Software Logo Kit\wslk.exe /?**". The Users Guide document in the toolkit download also outlines using the toolkit via the command line.

For example, here's a sample batch file I created to test a 32-bit based MSI installer and application:

```
rem --- Runs Windows 7 Software Logo for a 32-bit msi and application.
rem --- Run in same directory as MSI. Creates Logo.xml and outputs
failures and warnings.
rem --- Usage: AutoLogo.bat <MSI setup file> e.g. >AutoLogo.bat
MySetup.msi

rem --- Reset the toolkit
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe /reset

rem --- Pre-install for 32-bit
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe
/preinstall /setupexepath . /32bit

rem --- Install application quietly and run post install
msiexec /i %1 /q
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe
/postinstall

rem --- Run pre-uninstall and uninstall quietly
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe
/preuninstall
msiexec /uninstall %1 /q

rem --- Run post uninstall
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe
/postuninstall

rem --- clean up old logo report file and create a new Logo.xml report
file
del Logo.xml
"c:\Program Files\Microsoft Windows Software Logo Kit"wslk.exe
/createreport Logo.xml

rem --- Execute the Powershell script to output failures and warnings
PowerShell -ExecutionPolicy Unrestricted -File LogoResult.ps1
```

The batch file accepts the MSI file name as the first argument on the command line. It will reset the toolkit, execute the toolkit process, install/uninstall the application, generate the report, and execute a PowerShell script to report only "non-passing" tests.

Here's the PowerShell script (LogoResult.ps1) to parse the XML report to output failures and warnings:

```
[xml]$LogoRpt = get-content .\Logo.xml
Write-Host "-----"
-ForegroundColor green
Write-Host "OVERALL RESULT: " $LogoRpt.REPORT.OVERALL_RESULT
-ForegroundColor black -BackgroundColor white

foreach( $result in $LogoRpt.REPORT.REQUIREMENTS.REQUIREMENT)
{
    foreach( $test in $result.TEST)
    {
        if ($test.RESULT -ne "PASS")
        {
            Write-Host
"-----" -ForegroundColor
green
            write-Host "TEST:" $test.NAME
            Write-Host "RESULT:" $test.RESULT
            foreach( $msg in $test.MESSAGES.MESSAGE)
            {
                Write-Host $msg.TEXT
            }
        }
    }
}
```

Here's the sample output:

```
-----
OVERALL RESULT:  WARNING
-----

TEST: Write appropriate Add/Remove Program values
RESULT: WARNING
Value InstallLocation missing or invalid for program OEM Ready Demo.
-----

TEST: Install to Program Files
RESULT: WARNING
Program OEM Ready Demo fails due to missing install location.
-----

TEST: Install signed driver and executable files
RESULT: WARNING
Non-driver file c:\program files (x86)\microsoft\oem ready
demo\oemreadydemo.exe does not have a valid signature, either embedded or
via a catalog file.
-----

TEST: Perform version checking properly at runtime
RESULT: IN_PROGRESS
-----

TEST: Don't block reboot
RESULT: WARNING
-----

TEST: Multi User registry check
RESULT: WARNING
Registry key [HKEY_CURRENT_USER\Control Panel\Personalization\Desktop
Slideshow][HKEY_CURRENT_USER\Control Panel\Personalization\Desktop
Slideshow] Interval=DWord:1800000[HKEY_CURRENT_USER\Control Pan
el\Personalization\Desktop Slideshow] LastTickHigh=DWord:30079071
[HKEY_CURRENT_USER\Control Panel\Personalization\Desktop Slideshow]
LastTickLow=DWord:1766643285[HKEY_CURRENT_USER\Control Panel\Person
alization\Desktop Slideshow] Shuffle=DWord:0 was modified during
installion.
-----

TEST: Multi User session test
RESULT: WARNING
Application fails due to no shortcuts.
-----

TEST: Multi User Check Logs
RESULT: WARNING
```

Since the report is XML based, we could do just about anything with the information. You could generate reports, assign bugs to developers, send e-mails, etc. If you do plan on getting the logo, the earlier you find issues in your development or testing process, the more time you will have to fix them.

I hope this gives you some ideas on how you might leverage these test to help find issues in your application. Who knows?... You might be eligible for logo and can submit to get the logo "sticker" for your application.



If I'm an Administrator, Why Do I Get Access Denied?

over 6 years ago by [Pat Altimore MSFT](#)

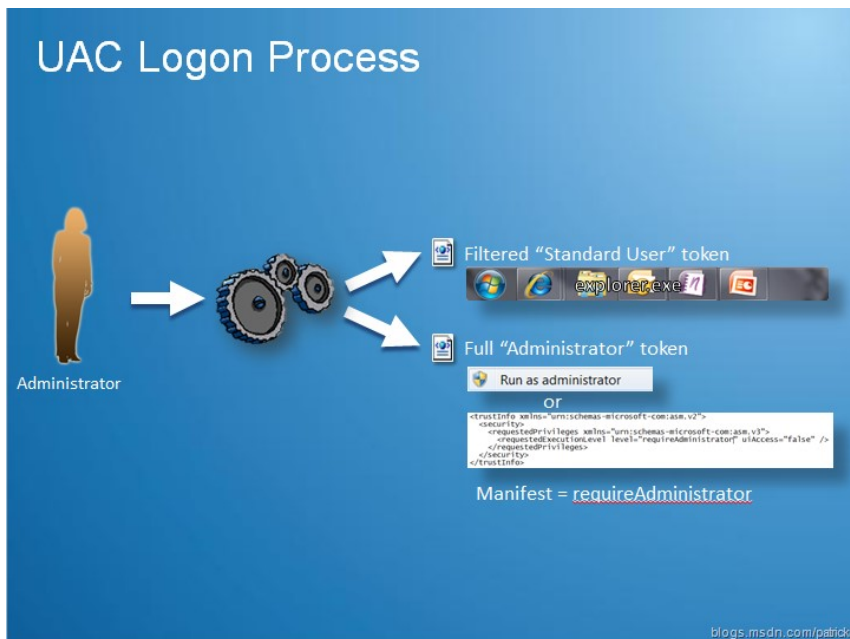
3

User Account Control (UAC) can seem mysterious. After all, if I'm a member of the Administrators group, shouldn't I have access to everything? If you're a developer, you'll run into a UAC issue sooner or later. In this post, I'll cover the basics that every developer should know.

Standard User by Default

Running applications as "Standard User" by default is what UAC is all about. Running processes with as few rights and privileges as possible is a common security defense in depth technique. UAC accomplishes this during the logon process. The traditional NT security model created a single access token at logon that all processes on the desktop use. The token contained privileges based on the groups the logged on user belonged. With UAC enabled, the logon process creates the traditional "full" token and also creates a limited "filtered" token. The filtered token contains only "standard user" privileges.

Here's a slide to help illustrate the "by default" part.



The magic happens when explorer.exe is started using the filtered token. Explorer.exe is your taskbar and is responsible for starting everything on the desktop. By default, child processes are started with the same token as their parent. Therefore, since explorer.exe is started with the filtered token, everything that is launched by the taskbar is **standard user by default** (presto!).

Bonus question: What token is used to start explorer.exe with UAC disabled? *Answer:* The full administrator token. This is why Windows 7 (and Vista) behave like traditional NT with UAC disabled.

What if you want to use the full token? You can run your application elevated (right click and select "Run as Administrator"). This will start the application with the full token instead of the filtered token. As an application developer, if you have a good reason to always start an application with the full token, you would set the appropriate run level in the manifest.

Let's take a look at two different command prompts to illustrate this concept.

The following is a command prompt started from the taskbar:

```

C:\Windows\system32\cmd.exe

C:\Users\DP196>whoami /priv /user
USER INFORMATION
-----
User Name      SID
-----
patricka02\dp196  S-1-5-21-1009

PRIVILEGES INFORMATION
-----
Privilege Name      Description      State
-----
SeShutdownPrivilege Shut down the system Disabled
SeChangeNotifyPrivilege Bypass traverse checking Enabled
SeUndockPrivilege Remove computer from docking station Disabled
SeIncreaseWorkingSetPrivilege Increase a process working set Disabled
SeTimeZonePrivilege Change the time zone Disabled

C:\Users\DP196>

```

The following is a elevated command prompt started from the taskbar via right clicking and choosing "Run as Administrator":

```

Administrator: C:\Windows\System32\cmd.exe

C:\Windows\system32>whoami /user /priv
USER INFORMATION
-----
User Name      SID
-----
patricka02\dp196  S-1-5-21-1009

PRIVILEGES INFORMATION
-----
Privilege Name      Description      State
-----
SeIncreaseQuotaPrivilege Adjust memory quotas for a process Disabled
SeSecurityPrivilege Manage auditing and security log Disabled
SeTakeOwnershipPrivilege Take ownership of files or other objects Disabled
SeLoadDriverPrivilege Load and unload device drivers Disabled
SeSystemProfilePrivilege Profile system performance Disabled
SeSystemtimePrivilege Change the system time Disabled
SeProfileSingleProcessPrivilege Profile single process Disabled
SeIncreaseBasePriorityPrivilege Increase scheduling priority Disabled
SeCreatePagefilePrivilege Create a pagefile Disabled
SeBackupPrivilege Back up files and directories Disabled
SeRestorePrivilege Restore files and directories Disabled
SeShutdownPrivilege Shut down the system Disabled
SeDebugPrivilege Debug programs Disabled
SeSystemEnvironmentPrivilege Modify firmware environment values Disabled
SeChangeNotifyPrivilege Bypass traverse checking Enabled
SeRemoteShutdownPrivilege Force shutdown from a remote system Disabled
SeUndockPrivilege Remove computer from docking station Disabled
SeManageVolumePrivilege Perform volume maintenance tasks Disabled
SeImpersonatePrivilege Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege Create global objects Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Disabled
SeTimeZonePrivilege Change the time zone Disabled
SeCreateSymbolicLinkPrivilege Create symbolic links Disabled

C:\Windows\system32>

```

We can see that the user name and the SID are identical. However, the non-elevated command prompt only has the privileges of a standard user.

What's a Standard User?

A standard user is only allowed to change their own settings and environment (per user). An administrator can change system wide settings (per machine). Here's a table of examples of what you can and can't do as a standard user:

Allowed	Not Allowed
Run most applications	Install applications
Change per user settings	Change system components
Read from per user and per machine locations	Change per machine settings
Create local objects	Perform operations requiring Admin privileges
	Write to per machine locations
	Create global objects

Just by simply limiting the scope of changes a user/application can perform can have a big impact on security, reliability, and stability of a machine. Enterprise guys who manage lots of desktops like the concept of standard user because it helps lower their [TCO](#).

Finding UAC Issues

The first step to determine if you have a UAC issue is to try running the application elevated (right click – Run as Administrator). You can also try turning UAC off. If your application works in one of these scenarios, you probably have a UAC issue.

UAC issues are caused by an application trying to make a change system wide. This means it's trying to write to a per machine location or create something globally across all users.

Is your application trying to:

- Write to Program Files, Windows, System32, HKLM/Software, or Root?
- Create anything "globally" like a memory mapped file?
- Send Windows messages between an app running as standard user to one running elevated?
(See: [this post](#))

If so, this is where to start looking for the problem.

After you have an idea of where the problem might be, [Process Monitor](#) is a great tool to start isolating the issue. Process Monitor will log all file and registry activity and will usually point you in the right direction.

Here's a snip from a Process Monitor log where an application is trying to write to the HKLM hive in the registry. HKLM is protected by UAC and [Mandatory Integrity Control](#) (MIC). Therefore, you will get an Access Denied:

5213...	legacyUAC.exe	4552	RegOpenK...	HKLM\SOFTWARE\legacyUAC	NAME NOT FOUND	Desired Access: Read, Maximum Allowed
5213...	legacyUAC.exe	4552	RegCreate...	HKLM\SOFTWARE\legacyUAC	ACCESS DENIED	Desired Access: Write

We can see in the log what key is trying to be opened for write and the result of ACCESS DENIED.

Looking for UAC issues can be tricky. You need to watch out for UAC Virtualization as well. UAC Virtualization may allow your program to avoid access denied errors. However, when you manifest your app, it will disable this mitigation. See my post on [Demystifying UAC Virtualization](#).

Here's a sample app trying to write junk.txt to Program Files. Program Files is protected by UAC and MIC. However, the program doesn't fail because of UAC Virtualization in this case:

5054...	legacyUAC.exe	2612	CreateFile	C:\Program Files\junk.txt	REPARSE	Desired Access: Generic Write, Read Attribu...
5054...	legacyUAC.exe	2612	CreateFile	C:\Users\patrick\AppData\Local\VirtualStore\Program Files\junk.txt	SUCCESS	Desired Access: Generic Write, Read Attribu...
5054...	legacyUAC.exe	2612	WriteFile	C:\Users\patrick\AppData\Local\VirtualStore\Program Files\junk.txt	SUCCESS	Offset 0, Length: 22, Priority: Normal
5054...	legacyUAC.exe	2612	CloseFile	C:\Users\patrick\AppData\Local\VirtualStore\Program Files\junk.txt	SUCCESS	

We see that we try to write to Program Files but get a "REPARSE" result. This is UAC Virtualization intercepting the access denied. We then see the file being written to the virtual store instead.

Next Steps

The goal of this post was to give you a base understanding of UAC from a developer perspective. UAC is a really big topic. There's a lot of good resources. Here are some of my favorites:

[Application Compatibility Cookbook: User Account Control](#)

[Inside Windows Vista User Account Control](#) – Read this one first :-)

[Inside Windows 7 User Account Control](#)

[MSDN: User Account Control](#)



What is Interactive Services Detection and Why is it Blinking at Me?

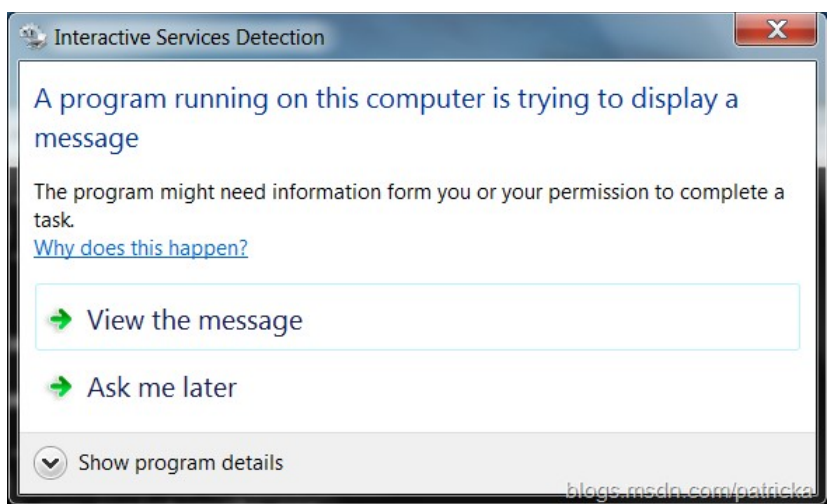
over 6 years ago by [Pat Altimore MSFT](#)

UPDATE: If you're a Windows user trying to figure out what's wrong with your PC, please check out my other post: [Troubleshooting Interactive Services Detection](#). If you are a developer (or want nerdy details), please read on.

Have seen this button flashing on the taskbar?



When you click on the button, you get this dialog.



If you click "View the message", your screen blinks and you are taken to a blank desktop with a couple of dialog boxes.

Why is this Happening?


Services and system processes run in session 0. Prior to Vista, the console (first logged on user's desktop) ran in session 0 as well. Vista introduced session 0 isolation to protect services from elevation of privilege exploits from the console desktop. Now, the first user's desktop runs in session 1.

Interactive Services Detection (the blinking button on the taskbar) is a mitigation for legacy applications that detects if a service is trying to interact with the desktop. This is handled by the Interactive Services Detection (UI0Detect) service.

When you choose "View the message", you are taken to session 0's desktop and you can only interact with the dialog or message that services have tried to display on the desktop.

Behavior Depends on Your Bits

The Interactive Services Detection service is set to start "manually". This means that it won't start automatically when the system boots.

Name	Description	Status	Startup Type
 Interactive Services Detection	Enables user notification of ...		Manual

On a 32-bit version of Windows, the OS will detect desktop interaction and start the UI0Detect service and you will see the flashing taskbar button.

On a 64-bit version of Windows, if the service is a native 64-bit application, the OS will not start the UI0Detect service. Therefore, the service that is trying to interact with the desktop will appear to hang. If you need the mitigation for a 64-bit service, you will need to be sure the service is running in order to get the mitigation.



Here's the really weird part... If you have a 32-bit service on a 64-bit Windows, WOW64 will start the UI0Detect service and you will see the mitigation. The reason for this is that the WOW64 environment will behave as close to native 32-bit as possible – including mitigations. (Special thanks to Maarten for his assistance on figuring this out. :-)

A way to play around with Interactive Services Detection is to use PsExec tool.

For example, if we try this:

```
> psexec \\localhost -i 0 calc.exe
```

This will launch the calculator application in session 0. This is a quick way to simulate a service interacting with the desktop. I find this example interesting. PsExec is a 32 bit application. Therefore, UI0Detect will always get started even though calc.exe is 64-bit. Here's a snip from Process Explorer.

Process	PID	CPU	Description	Company Name	Cycles	Session	Image Type
 UI0Detect.exe	7452		Interactive services det..	Microsoft Corporation	1,435,321,175	0	64-bit
 UI0Detect.exe	7048		Interactive services det..	Microsoft Corporation	174,381,258	1	64-bit
 PsExecSvc.EXE	3564		PsExec Service	Sysinternals	45,805,801	0	32-bit
 calc.exe	4068		Windows Calculator	Microsoft Corporation	178,575,988	0	64-bit

We can see that UI0Detect service is started in session 0 which creates a new process in session 1. The UI0Detect process in session 1 is the Interactive Service Detection dialog. PSEXESVC.exe is the PsExec command service and note that it is a 32-bit app in session 0. Calc.exe is started in session 0 because we specified session 0 as an argument to PsExec.

Fixing Multiple Issues with One Solution

If a service has this issue, it probably also has an issue with [remote desktop services](#) and/or [fast user switching](#). Session 0 isolation, remote desktop services and fast user switching all use the same session isolation plumbing. So, if you fix this issue, you are probably fixing several issues you may have not know you had.

The [Services in Windows Vista](#) whitepaper that talks about all the changes to services including Session 0 isolation. Also, check out [this classic post on Session 0 Isolation with developer guidance](#).



Why does a High DPI Setting Make My Application Look Fuzzy and Have Clipped Text?

2

over 6 years ago by [Pat Altmore MSFT](#)

If you increase your DPI setting in Windows 7 (or Vista), you may notice your application doesn't look the way you intended.

What's the Big Deal?

You may say, "Nobody changes the default DPI, right?" Well, Windows 7 might. [Windows 7 may increase the default DPI setting based on the resolution the video card supports](#). The default DPI decision occurs when you do a clean install. Over time as people buy newer machines supporting higher resolution, you will see a trend of users running at a higher DPI.

Why is Making Your Application DPI Aware Important?

As display resolution increases, stuff on the screen gets smaller. When the text is too small to read, people tend to reduce the resolution. By reducing the resolution, you are relying on the monitor to do the scaling and the desktop can look stretched or blurry. Also, [ClearType](#) will only work at native (recommended) resolution.

So, how do you get all the readability benefits of font smoothing and still have bigger text? You increase the DPI.

Can this Affect my Application?

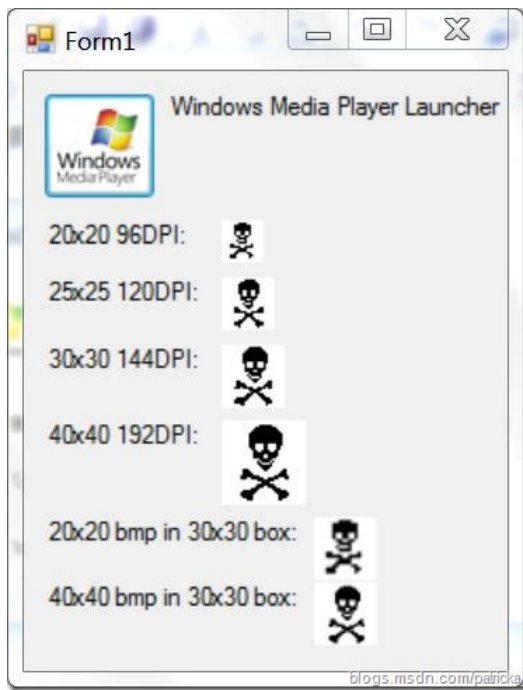
In most cases, applications are designed for the default of 96 DPI. As you increase the DPI, your application appears to get smaller. If your application doesn't declare that it is high DPI aware, [Windows 7 Desktop Windows Manager \(DWM\)](#) will scale it to make it look bigger at a DPI setting of 144 DPI (150%) or greater.

Blurry UI (Pixilation)

Blurry UI occurs when [DWM](#) scales an application window. Here's an example of an application designed for 96 DPI and doesn't declare DPI awareness. This is what the application looks like with a setting of 144 DPI (150%). Notice the text in the title bar is clear but the text in the application is blurry.



Here's another sample app that shows a bunch of bitmaps which are designed for different DPI settings. However, the application does not set DPI awareness and everything looks kind of blurry including the larger bitmaps because the whole window is being scaled by DWM at 144 DPI.



Clipped UI

Another very common problem when you increase the DPI is clipped text. This happens because the application has code to size windows and assumes 96 DPI. Here's the demo app at a setting of 120 DPI (125%). Notice that the text in the window and buttons are clipped.



How do I Fix It?

Before you fix it, you need to find it. The DPI setting can be found in Control Panel\Appearance and Personalization\Display. You should test your application at a DPI setting of 125% looking for clipped UI. You should also test at a DPI setting of 150% to see if DWM scaling is acceptable. The good news is DPI testing on Windows 7 is much easier. You can now change the DPI and it only requires a log off/on instead of a restart. Also, DPI settings are now per user rather than per machine. I usually create additional users with different DPI settings on my machine for testing. When I want to test at a different DPI setting, I just log in / switch to a different user. As a bonus, you can test multi-user as well. :-)

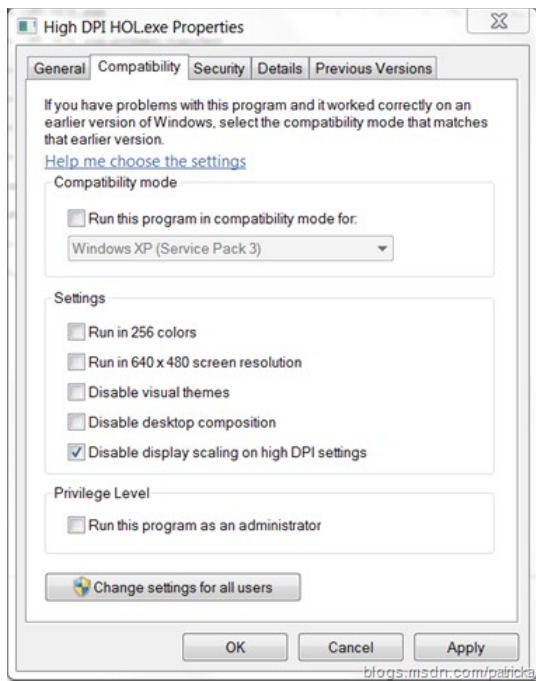
Guidance

The best guidance for designing high DPI aware applications can be found in the [High DPI section of MSDN](#). This documentation has lots of info on how to fix high DPI issues in applications. I'll cover the common topics I mentioned in this post but please refer to the documentation for a complete list.

Fuzzy UI

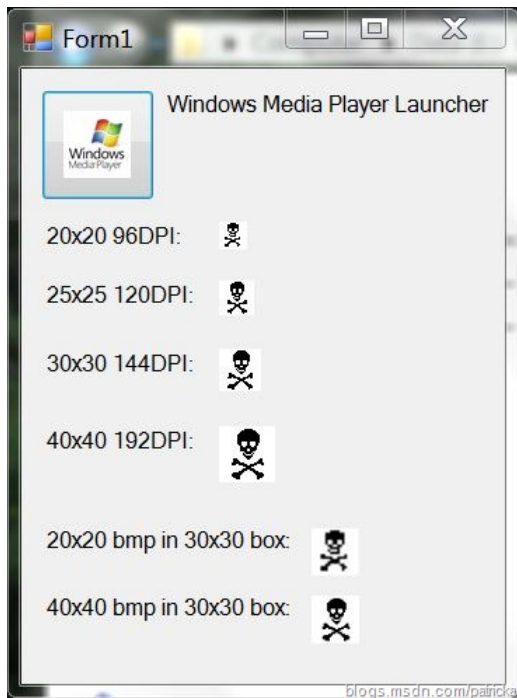
Fuzzy UI is caused by DWM scaling. So, you need handle the scaling yourself (see the MSDN info mentioned above) and then tell the OS your application is high DPI aware. This can be done with an [API](#) but the preferred method on Windows 7 or greater is to declare DPI awareness in the application manifest.

You can see what affect setting DPI awareness will have on your application by using a compatibility setting for the application. In the executable properties, under the compatibility tab, check the "Disable display scaling on high DPI settings" option.



Now when you run the application, DPI scaling will be disabled for the application. Note that the compatibility property is just a mitigation for legacy applications. Ideally, you need to be declaring your application as DPI aware in the manifest.

Here's our sample application again with the `<dpiAware>true</dpiAware>` element in the manifest:



Note that the text and most of the icons are no longer fuzzy. However, there are still some problems. The Windows Media Player bmp doesn't fill the button. To fix this, you would need to detect the DPI and place the appropriate sized bmp in the button. The 20x20 bmp in the 30x30 box is also fuzzy. This is because the application is doing the scaling. This would happen on any DPI setting. I threw this in to the app to show you may have other scaling issues that aren't related to DPI. Also note that it's usually better to scale down a bigger bmp into a smaller box as you can see in the final icon.

Clipped Text

Clipped text is caused by incorrectly calculating the size of a window. Basically, you need to be sure you are scaling your UI based on relative pixel size. There are helper functions to do this in the CDPI class. Please refer to the step by step tutorial in MSDN which is based on the High DPI Demo App that I'm using for this blog post.

Final Thoughts

In the labs, when we test a native application at a high DPI setting, we usually find at least one issue. Sometimes they are minor, sometimes they are not. We see a lot fewer issues in managed code. However, some controls aren't very high DPI friendly. Regardless, DPI testing is worthwhile. It's better to discover the problem before your customer.



Where Should I Store my Data and Configuration Files if I Target Multiple OS Versions?

6

over 6 years ago by [Pat Altimore MSFT](#)

Over the past few releases of Windows, you may have noticed common folder locations have moved around a bit. What should you do if you want your code to target multiple OS's? Perhaps you are updating an application from XP to Windows 7 and wondering where those old directories went. Hopefully, this post will answer those questions and help you design your application to continue to work with future OS releases.

What's the Recommended Location for Application Files?

Where to store application files depends on how that data is used by the application and the user. Here is a table to outline the recommended locations for user documents and configuration data. ~~I've mapped the locations across OS's in this giant table for comparison.~~ UPDATE: The table was a little too giant for the new MSDN template. I've reformatted the information into sections:

Per user configuration files synchronized across domain joined machines via Active Directory Roaming

Configuration data files that the application uses and are unique per user. This per user data is synchronized across the domain via Active Directory.

Example:	MyAppSettings.xml
Windows 7:	%USERPROFILE%\AppData\Roaming\<MyCompany>\<MyApp>
Vista:	%USERPROFILE%\AppData\Roaming\<MyCompany>\<MyApp>
XP:	%USERPROFILE%\Application Data\<MyCompany>\<MyApp>
Environment Variable:	%APPDATA%
<u>Known Folder ID:</u>	FOLDERID_RoamingAppData
<u>System.Environment.SpecialFolder:</u>	System.Environment.SpecialFolder.ApplicationData
<u>CSIDL:</u>	CSIDL_APPDATA

Local per user configuration files

Configuration data files that the application uses and is unique per user. It stays local to the individual machine and is not synchronized via Active Directory.

Example:	MyMachineSpecificData.xml
Windows 7:	%USERPROFILE%\AppData\Local\<MyCompany>\<MyApp>
Vista:	%USERPROFILE%\AppData\Local\<MyCompany>\<MyApp>
XP:	%USERPROFILE%\Local Settings\Application Data\<MyCompany>\<MyApp>
Environment Variable:	%LOCALAPPDATA% <i>Note: Does not exist on XP</i>
<u>Known Folder ID:</u>	FOLDERID_LocalAppData
<u>System.Environment.SpecialFolder:</u>	System.Environment.SpecialFolder.LocalApplicationData
<u>CSIDL:</u>	CSIDL_LOCAL_APPDATA

Per machine configuration data

Configuration data files the application uses and is per machine. It is used across all users of the application.

Example:	AppConfigDatabase.xml
Windows 7:	%SystemDrive%\ProgramData\MyCompany\MyApp
Vista:	%SystemDrive%\ProgramData\MyCompany\MyApp

XP: %SystemDrive%\Documents and Settings\All Users\Application Data
Environment Variable: Vista/Win7: %PROGRAMDATA% XP: %ALLUSERSPROFILE%
Known Folder ID: FOLDERID_ProgramData
System.Environment.SpecialFolder: System.Environment.SpecialFolder.CommonApplicationData
CSIDL: CSIDL_COMMON_APPDATA

Per user “Documents”

“Document” type files that users create/open/close/save in the application that are per user.

Example: MyDoc.doc
Windows 7: Libraries
Vista: %USERPROFILE%\Document
XP: %USERPROFILE%\My Documents
Environment Variable: *Not applicable*
Known Folder ID: FOLDERID_Documents
System.Environment.SpecialFolder: System.Environment.SpecialFolder.MyDocuments
CSIDL: CSIDL_MYDOCUMENTS, CSIDL_PERSONAL

Per Machine “Documents”

“Document” type files that users create/open/close/save in the application that are used across users. These are usually template or public documents.

Example: MyTemplate.dot
Windows 7: C:\Users\Public
Vista: %SystemDrive%\Users\Public
XP: %ALLUSERSPROFILE%\Documents
Environment Variable: Vista/Win7: %PUBLIC% *Note: Does not exist on XP*
Known Folder ID: FOLDERID_PublicDocuments
System.Environment.SpecialFolder: System.Environment.SpecialFolder.CommonDocuments
CSIDL: CSIDL_COMMON_DOCUMENTS

It's obvious after looking at all these locations that where you store your files can be challenging if you are targeting multiple OS versions. The best guidance is to use API's to find the special folder path. API's will return the appropriate location for the target OS.

Targeting Vista and Higher

Native Code

The best API to use if you are targeting Vista and beyond is the new SHGetKnownFolderPath. This function replaces SHGetFolderPath and has the following advantages.

Example:

```
// Roaming AppData - Vista and greater
if(SUCCEEDED(SHGetKnownFolderPath ( FOLDERID_RoamingAppData,
KF_FLAG_CREATE,
                                NULL, &wszPath )))
{
    printf("\nSHGetKnownFolderPath FOLDERID_RoamingAppData    =%S\n",
wszPath);
}

// Per user Documents - Vista and greater

if(SUCCEEDED(
SHGetKnownFolderPath ( FOLDERID_Documents, KF_FLAG_CREATE,
                                NULL, &wszPath )))
```

```
{
    printf("SHGetKnownFolderPath FOLDERID_Documents      =%S\n",
wszPath);
}
```

The output on Windows 7 is:

```
SHGetKnownFolderPath FOLDERID_RoamingAppData
=C:\Users\patricka\AppData\Roaming

SHGetKnownFolderPath FOLDERID_Documents      =\\zaw\Mydocs\patricka\My
Documents
```

Managed Code

You can use [System.Environment.SpecialFolder](#) along with the [System.Environment.GetFolderPath](#) function to get the path.

Example:

```
// Roaming Application Data - C#
SpecialFolderPath = System.Environment.GetFolderPath(

System.Environment.SpecialFolder.ApplicationData);

Console.WriteLine("SpecialFolder.ApplicationData path      ={}",
SpecialFolderPath);

// Per user Documents
SpecialFolderPath = System.Environment.GetFolderPath(

System.Environment.SpecialFolder.MyDocuments);

Console.WriteLine("SpecialFolder.MyDocuments path      ={}",
SpecialFolderPath);
```

The output on Windows 7 is:

```
SpecialFolder.ApplicationData path      =C:\Users\patricka\AppData\Roaming

SpecialFolder.MyDocuments path          =\\zaw\Mydocs\patricka\My Documents
```

Note that the path for Documents returns my redirected [IntelliMirror](#) server path because I ran it on a domain joined machine and my domain account is IntelliMirror enabled.

Targeting XP and Higher

Native Code

If you are still targeting XP, you'll need to use the legacy API [SHGetFolderPath](#) that uses [CSIDL's](#). This API is still supported in Vista and Windows 7 and will map to the correct locations on all three versions of the OS.

Example:

```
// Roaming AppData - Legacy

if (SUCCEEDED(SHGetFolderPath(NULL,
                             CSIDL_APPDATA|CSIDL_FLAG_CREATE,
                             NULL,
                             0,
                             szPath)))
{
    printf("\nSHGetFolderPath CSIDL_APPDATA      =%S\n", szPath);
}
```

```
// Per user documents (My Documents) - Legacy
if (SUCCEEDED(SHGetFolderPath(NULL,
                             CSIDL_MYDOCUMENTS|CSIDL_FLAG_CREATE,
                             NULL,
                             0,
                             szPath)))
{
    printf("\nSHGetFolderPath CSIDL_MYDOCUMENTS    =%S\n", szPath);
}
```

The output on Windows 7 is:

```
SHGetFolderPath CSIDL_APPDATA      =C:\Users\patricka\AppData\Roaming

SHGetFolderPath CSIDL_MYDOCUMENTS =\\zaw\Mydocs\patricka\My Documents
```

The output on XP is:

```
SHGetFolderPath CSIDL_APPDATA      =C:\Documents and
Settings\XPMUser\Application Data

SHGetFolderPath CSIDL_MYDOCUMENTS =C:\Documents and Settings\XPMUser\My
Documents
```

The API returns the appropriate path for the given OS version. Please note: I ran this in [XP Mode](#) on Windows 7. My XP Mode machine was not domain joined; therefore, I did not get IntelliMirror redirection.

Managed Code

Since we are using the .NET framework, we can still use the same technique as described in the previous section. We saw the output for Windows 7. Here's the output we get on XP:

```
SpecialFolder.ApplicationData path  =C:\Documents and
Settings\XPMUser\Application Data

SpecialFolder.MyDocuments path      =C:\Documents and Settings\XPMUser\My
Documents
```

Similar to the native API's, we get the appropriate location for XP. Again, because my XP Mode virtual machine was not domain joined, there is no IntelliMirror redirection.

Final Thoughts

Think about the type of files your application is writing. Are they configuration files, data files, or documents? Are they per machine or per user? Choose the appropriate location for the file type and then use the right API to get the path to that location. [Chris has a great post](#) on this topic. There is also a [support article](#) and a [whitepaper](#) that has good info on storing application data.



Solving Compatibility Issues with Internet Explorer 8's Built-in Developer Tools

0

over 6 years ago by [Pat Altimore MSFT](#)

IE8 has some outstanding developer tools built-in and can be accessed by pressing F12. If you happen to be viewing this in IE8, go ahead and press F12 now. After pressing F12, you may be really excited and have stopped reading this or you may be wondering how this could help you. I'll cover some of my favorite functionality. For more complete documentation, please see [Discovering Internet Explorer Developer Tools](#).

Compatibility Modes

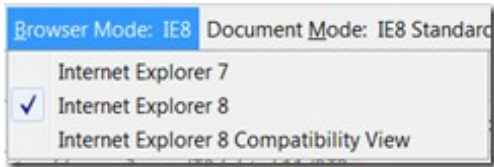
IE8 renders content in [IE8 standards mode by default](#). However, IE8 has the ability to render pages in [IE8 Compatibility](#) and [IE7 Standards modes](#) as well. One of the first benefits you may notice is that the tool tells you what the browser mode and document mode is for your page.

If I go to my blog, I can see that Browser Mode=IE8 and Document Mode=IE7 Standards

Browser Mode: IE8 Document Mode: IE7 Standards

In this example, IE defaults to IE8 and Document mode is determined by the page's `<!DOCTYPE>`.

Let's say your web page is broken and you want to try different compatibility modes. This can be easily tested by clicking on the mode you want to change.



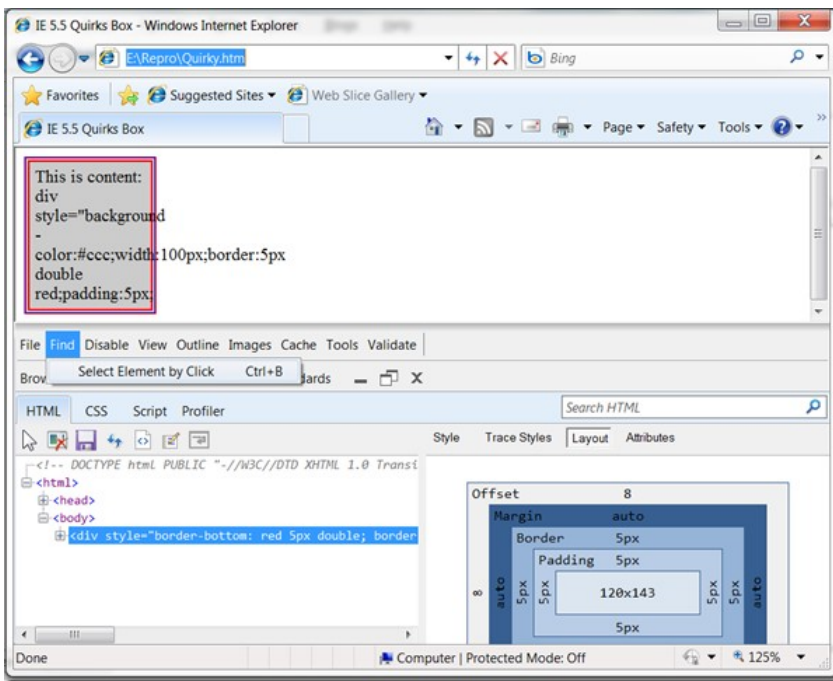
If you find that one of the compatibility modes fix your issue, follow the guidance in the [IE8 – Defining Document Compatibility](#) article.

Experiment by Changing Content and Code

Compatibility modes can be a workaround or a solution. However, you may just have a small issue that you want to fix or you may want to move your site to support IE8 Standards mode. The developer tools can help a great deal in finding problems and allowing you to try a fix.

Jumping to Content with Issues

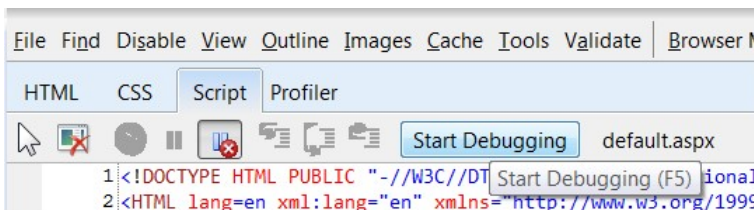
Use the **Find | Select Element by Click** functionality to highlight the area in the page that has a problem. A blue box will highlight the section you are hovering over. Click the section and you will jump to that location in the content.



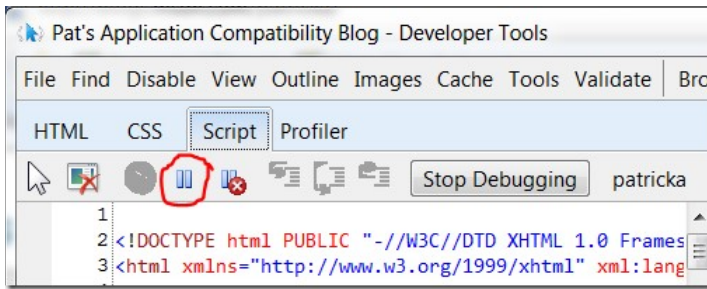
You can click on the html code to edit it. When you make a change to the content, it will be reflected in the current browser session. You can quickly experiment with potential fixes using this method.

Debugging Script

I don't think this could be easier. You can quickly start debugging script by clicking the **Script** tab and clicking the **Start Debugging** button.



You can now set breakpoints, look at call stacks, variable values, console errors, etc. Finding your issue may still be challenging but the tools at your disposal are very useful. One trick to find the piece of script you are interested in, click the **Break All** button.



Now, you can invoke the script you are interested in by clicking, typing or causing the event that you want to debug. The debugger will break at the first line of the script.

But Wait, There's More...

I obviously haven't covered everything. I wanted to show you some of the features I use to figure out compatibility problems. Please see [Discovering Internet Explorer Developer Tools](#) for more info.



Getting Started with IE8 Compatibility

over 6 years ago by [Pat Altimore MSFT](#)

0

I frequently get asked about IE8 compatibility. I usually point people to the [IE8 Readiness Toolkit's developer section](#). This is a good resource but there are various topics and resources that I'd like to summarize in this post.

Three Browsers for the Price of One

If you're in the browser business, you need to balance implementing new standards and not breaking legacy websites. Over the years, this was handled using [quirks mode](#). This works in most cases but is tricky because different browsers handle quirks mode differently.

IE8 enhances this by having three ways of rendering content – IE 8 Standards, IE 7 Standards, and Quirks mode. The choice of how the page is rendered can be controlled at the client or at the server.

For more info, see: [IEBlog: Defining Document Compatibility](#); [IE8 Readiness Toolkit: Versioning](#); [IE8 Readiness Toolkit: Compatibility](#)

New Browser = New User Agent String

When a new IE version released, the [user agent string](#) is updated to reflect the browser's version. Some websites use the user agent string to detect the browser and decide what content to display. If you're wondering what user agent string you are sending to websites, you can type the following into IE's address bar:

```
javascript:alert(navigator.userAgent)
```

On my Windows 7 machine, my user agent string is:

```
Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; MS-RTC LM 8)
```

If I turn on [compatibility view](#), my user agent string is:

```
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; MS-RTC LM 8)
```

If your site uses the user agent string to make decisions on what to display, you may have issues with your web site.

For more info, see: [Understanding User-Agent Strings](#); [IE8 Readiness Toolkit: Versioning](#); [Cookbook: User Agent String](#)

Loosely Coupled Internet Explorer (LCIE)

IE7 on Vista defaulted to [low integrity level protected mode](#) when you browsed to a site. If you browsed to a trusted site, IE would display a message and open another instance of IE that ran at a [medium integrity level](#). This wasn't the best user experience.

To avoid the poor user experience, IE8 introduced [LCIE](#). This separates the IE frame from the tabs. Tabs can be serviced in different iexplore processes. This solves the user experience problem with trusted and non-trusted sites while still providing isolation. This also improves stability and performance of IE.

What's the Compatibility Impact?

To improve compatibility, IE8 runs all sites in the Intranet zone with protected mode off. In most cases, this helps with internal website compatibility. However, it is a behavior change from Vista and is worth noting.

If you develop IE add-ons, LCIE may affect you. See the [IE8 Readiness Toolkit – LCIE section](#) for more info.

Data Execution Protection (DEP/NX) is on by default

[DEP/NX](#) mitigates memory-related vulnerabilities. An application or ActiveX control that runs in Internet Explorer and is not compatible with DEP/NX will crash on startup. IE may crash on startup if add-ons not compatible with DEP/NX are installed. This is common with older versions of the ATL library which are not DEP/NX compatible.

For more info, see: [IEBlog](#); [IE8 Readiness Toolkit: DEP/NX](#); [Cookbook: DEP/NX](#)

Anything Else?

Yes, there's more. I didn't cover everything. The goal of this post was to talk about some of the changes in IE8 that impact compatibility and point you to some really good resources to find information.

Here are my favorite resources for IE8 compatibility:

- [Windows 7 and Windows Server 2008 R2 Application Quality Cookbook](#)
- [IE8 Readiness Toolkit](#)
- Blogs : [IEBlog](#), [AskIE](#)



Q: Why Doesn't Drag-and-Drop work when my Application is Running Elevated? – A: Mandatory Integrity Control and UIPI

over 6 years ago by [Pat Altimore MSFT](#)

If you run notepad elevated (Right click | Run as Administrator), and you try and drag-and-drop a file from Windows Explorer, nothing happens. It looks like it is going to work because the pointer icon changes but the file doesn't open. Weird, huh?

What's Going On?

In the traditional NT Security model (prior to Vista), all processes on the same desktop ran with the same security token and had all the same privileges. UAC changed this by allowing processes with different privilege levels on the same desktop.

Lower Privilege Processes Can't Interfere with Higher Privilege Processes

In order to prevent potential [elevation of privilege](#) attacks, certain functionality needs to be blocked. This is implemented through [Mandatory Integrity Control](#) (MIC). All processes and all resources (files, registry, etc.) have an integrity level assigned. MIC prevents a standard user process from writing to a protected per machine location like Program Files or the HKLM registry hive. I won't go too deep into MIC in this post but the following is a great resource if you want more info: [Inside Windows Vista User Account Control](#).

User Interface Privilege Isolation (UIPI)

Okay, back to our drag and drop issue... A "sister" technology that works in conjunction with MIC is [UIPI](#). UIPI blocks Windows messages being sent from process with a lower MIC level to one running at a higher MIC level. Drag-and-drop is implemented via Windows messages. Therefore, if you try and drag-and-drop a file from Windows Explorer (medium MIC) to Notepad running elevated (high MIC), the Windows messages are blocked and drag-and-drop doesn't work.

You can use [ChangeWindowsMessageFilterEx](#) in your application to allow specified Windows messages to not be blocked. Unfortunately, this isn't recommended as a safe solution for drag and drop due to the messages that drag and drop uses.

Okay. Now What?

The best solution is to only use drag and drop between the same MIC levels. With UAC enabled, Windows Explorer will run at a medium MIC level. Therefore, your application (Notepad in our example) needs to run at medium (or lower) MIC level. The bottom line is that drag and drop from Windows Explorer will not work if your application is elevated. If you find yourself in this situation, you may need to rethink your application design or not support drag-and-drop with UAC enabled.



Common Application Installer Issues on Windows 7 (and Vista)

0

over 6 years ago by [Pat Altimore MSFT](#)

There are some very common application installer issues developers run into due to [UAC](#). Most of the issues are easy to fix. I'll cover the most common issues in this post.

Per Machine Installers Need Admin Privileges

Unless you have a [ClickOnce](#) installer, your installer probably installs to a per machine location. The recommended location is Program Files which requires administrative privileges. Therefore, your installer will need to run elevated. MSI based installers handle this automatically. Exe based installers require an [application manifest](#) with a run level=requireAdministrator. If your exe installer is elevating without a manifest, this is due to the [Installer Detection](#) mitigation. It is recommended you explicitly manifest the installer rather than relying on the mitigation.

My "Updater" Fails

Applications may "self update" or have a separate application that checks for update and applies the changes. The updater periodically checks for an update. If an update is available, it downloads and applies the update. If the updater is running as a standard user under UAC, it will fail when trying to write to Program Files.

Possible solutions with consent dialog:

- [Manifest](#) the updater with a run level=requireAdministrator. This will cause the updater elevate when it executes.
- Launch the updater by [elevating using ShellExecute](#).

Possible solutions without consent:

- Use [User Account Control Patching](#) (not supported on Server)
- Have an updater Windows service. Run the service with an account with administrator privileges.
- Use a scheduled task to perform the update check and installation. Configure the task to have administrator privileges.

My Installer Fails and Does a Rollback

I was talking with a developer who had been testing his app on Windows 7 and had very few issues. I asked him if he had any UAC issues. He said, "No, it works just fine under UAC. But we need to tell the user to disable UAC, install the app, then turn UAC back on." It turns out, he was having the following issue.

[MSI Installers Have a user context and an admin context](#). [Custom actions](#) can run in either context and default to the user context. If your custom action is doing something that requires administrative privilege (e.g. register a COM component), it will fail and cause the installer to rollback.

Possible Solution:

You need to make the custom action run in the admin context. This is done by setting the noImpersonate bit for the custom action. Chris has a great [post](#) on this topic.

A quick test to verify if this is likely the issue is to right click the installer and select "Run as Administrator". If it succeeds, it's probably this issue.

My Settings are Missing or in the Wrong Profile

I like running my machines as a Standard User (not a member of Administrators group). If I need to install a program, I get prompted for an administrator account and the installer runs with the administrator account I choose. This works great for my home machines that my kids use. :-)

Frequently, this scenario is not tested for installers (see our [testing document](#) for a good list of tests). An issue can occur if settings are configured at the end of the install. Since you are running the install as a different user, the settings are configured for the administrator account rather than the standard user account.

Possible Solution:

Per user configuration should happen during first run of the application by the user. This avoids this issue and does a better job supporting multiple users.