Fun with the ICA Client Object (ICO) and .Net Console Applications

By John Cattaneo · Published March 2, 2010 · 32 Comments
Products: XenApp

SHARE

Perusing the ICO Documentation , there are a bunch of diagrams and techie-talk about "embedding the ICA Client Object into a container" for proper usage. Some of the documentation goes on to define such containers as Internet Explorer, Netscape Navigator, Delphi application, MFC applications, etc. (holy dated material!?!). But who wants to write some web page with embedded scripting to launch a simple session? Furthermore, who wants to try and debug that? There has to be an easier/friendlier way … right?

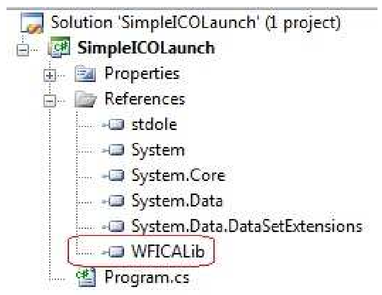
In an attempt to go from "0 to … Hey, I think I know what I'm doing now," I want to provide a step-by-step "getting started" tutorial for consuming ICO and the ICA Client Simulation APIs. I'll do so, by leverage one of my favorite, simple, and commonly used "containers" … a .Net Console Application. I'll also provide insight on some of the pitfalls and oddities that occur when trying consuming ICO in such a manner.

Downloadable Examples of what's covered below

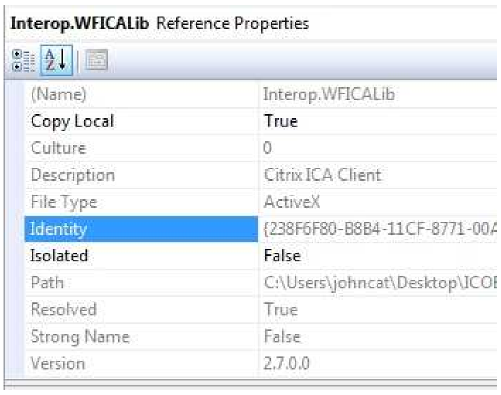
Getting Started

One of the best parts of the Ica Client Object (ICO) SDK is that is ships right along side your favorite version of the Citrix Online plug-in (formerly called ICA client). That is, there is no need to download any external binaries to reference as they all reside on your system, post plug-in install. So, with plug-in installed on your local development machine, start off by creating a Console Application in your favorite flavor of Visual Studio (these example uses VS2008).

Add a reference, within your project to the following file %ProgramFiles%\Citrix\ICA Client\Wfica.ocx (or on x64 development machines %ProgramFiles(x86)%). Doing so, will generate an interop reference for the ICO ActiveX control within your project. Pretty simple and friendly so far … right?



```
Solution 'SimpleICOLaunch' (1 project)
  SimpleICOLaunch
    Properties
    References
      stdole
      System
      System.Core
      System.Data
      System.Data.DataSetExtensions
      WFICALib
    Program.cs
```

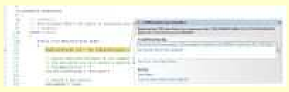


Interop.WFICALib Reference Properties

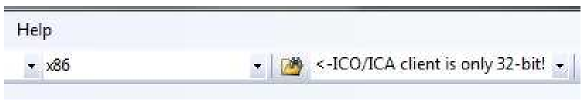
(Name)	Interop.WFICALib
Copy Local	True
Culture	0
Description	Citrix ICA Client
File Type	ActiveX
Identity	{238F6F80-B8B4-11CF-8771-00A
Isolated	False
Path	C:\Users\johncat\Desktop\ICOI
Resolved	True
Strong Name	False
Version	2.7.0.0

Be Careful

At this time, I think it's important to point out the Citrix Online plug-in is a 32-bit compiled application. As such, when the Citrix Online plug-in is installed the ICO ActiveX (COM) control is registered on the system. The registration occurs within the 32-bit hive of the registry (which is taken for granted until you're testing/developing on a 64-bit machine). So, in order to have your console application run properly on both x86 and x64 architectures, you must specifically compile it to target an "x86″ processor architecture within Visual Studio. Compiling "AnyCPU" will cause your application to look for the ICO COM class registration in the native registry hive on a 64-bit machine (not the Wow6432Node where it was registered). Subsequently, at runtime, you will see COM exceptions/errors thrown as seen (in the thubmnail) below on my 64-bit development machine.



Make a quick change to your target platform as seen here within Visual Studio:



Now, that we've got that small tidbit squared away. Let's get on to the launch …

Launching Your First Session

The main class that's used to launch and interact with an ICA session is the ICAClientClass. Ensure that you add the appropriate using statement to your program.cs file in order to access the namespace of the ICAClientClass class (yeah … that's just weird sounding).

Featured Products
- XenDesktop (2395)
- XenApp (2286)
- XenServer (1434)
- XenClient (612)
- NetScaler (1233)
- CloudGateway (108)
- Access Gateway (622)
- Citrix Online (232)
- Cloud Solutions (740)
- Non-Specific (2143)
- Mobility (342)
- AppDNA (71)
- VDI in a Box (166)
- Essentials for Hyper-V (404)

Featured Bloggers
- Laura Whalen
- Chris Fleck
- Kim DeCarlis
- Martin Duursma
- Sumit Dhawan
- Daniel Feller

Featured Tags
- virtualization (331)
- citrix_synergy (303)
- architecture (266)
- geek_speak (143)
- hdx (191)
- hypervisor (28)
- security (131)
- partners (700)
- events (136)
- byoc (32)
- virtual_desktop (59)
- customersuccess (17)

View All Tags ▸

Popular Posts
- Do you want Citrix XenApp to run Windows apps on the iPhone ? (853,278)
- Installing Ubuntu on XenServer (350,362)
- The Nirvana device a Smartphone as a PC alternative (248,552)
- iPhone, iCitrix ! (238,682)
- Free, as in Virtual Infrastructure (231,598)
- Do You Want To Use Citrix Receiver To Run Windows 7 From The iPad ? (228,032)
- What's the coolest app that doesn't work on the iPhone …. yet ? (187,383)

The various methods and properties of the class can be found outlined in the ICA Client Object Guide (here). For simplicity sake (and to build on this example further) the following example leverages only a very few properties and shows how we can launch our first application session, to Notepad, hosted on a XenApp server (read the code comments for extended detail).

Handy Hint

In the example, I'm targeting a server that has a Notepad published for my user, however you can easily launch an ICA desktop session by removing InitialApplication="#Notepad"; … and setting the Application="";

Simple ICO Launch – Program.cs

```
/// <summary>
/// This program demo's the basic of launching and ICO session from within an application.
/// </summary>
class Program
{
    static void Main(string[] args)
    {
        ICAClientClass ica = new ICAClientClass();

        // Launch published Notepad if you comment this line, and uncommented
        // the one above you will launch a desktop session
        // ica.Application = "";
        ica.InitialProgram = "#Notepad";

        // Launch a new session
        ica.Launch = true;

        // Set Server address
        ica.Address = "10.8.X.X";

        // No Password property is exposed (for security)
        // but can explicitly specify it by using the ICA File "password" property
        ica.Username = "johncat";
        ica.Domain = "xxxx";
        ica.SetProp("Password", "xxxx");

        // Let's have a "pretty" session
        ica.DesiredColor = ICAColorDepth.Color24Bit;

        // Reseach the output mode you want, depending on what your trying
        // to attempt to automate. The "Client Simulation APIs" are only available under c
        // (i.e. things like sending keys to the session, enumerating windows, etc.)
        ica.OutputMode = OutputMode.OutputModeNormal;

        // Height and Width
        ica.DesiredHRes = 1024;
        ica.DesiredVRes = 786;

        // Launch/Connect to the session
        ica.Connect();

        Console.WriteLine("\nPress any key to log off");
        Console.Read();

        // Logoff our session
        Console.WriteLine("Logging off Session");
        ica.Logoff();
    }
}
```

If you've made it this far, and your session is launching, your probably feeling pretty good. However, let's delay the pre-emptive celebration, and try our hand at more advanced functionality …

Handling Events

With our first session launch behind us, let's start asking the fun questions. "Well, how does my automation/script know when I've successfully logged on? And Can I figure out when Notepad is available in the session?" Questions like these start to unveil the power and granularity that the object can provide.

Let's add some event handlers to our application so we can try and figure out when the client has actually logged on successfully. In the following example, you can see that I've added an explicit handler to the application to handle the OnLogon event from the Ica Client Object. Within the main thread of the program I use an AutoResetEvent as a control mechanism to block program execution until our session is logged on. You can see in my event handler, I trigger the AutoResetEvent explicitly, so that the program execution will continue within our "Main()" method as soon as ICO tells us the session has logged on. If we don't receive the OnLogon event within a few minutes we'll time out waiting and attempt to log off the session anyways.

Knowledge Drop

If you're using any version of the client, earlier than 11.2, and experimenting with this example, you'll never receive the OnLogon event. Previous clients had to explicitly start a Windows application pump to handle various ICO events from the ICO object. This was not intuitive by any means, and many users may have thought ICO events and/or the client simulation APIs were even broken. However, this is not the case, and I've submitted a code example (along with the ones outlined in this blog) that illustrates a work around for previous clients. Talking to some of the developers, there were design changes introduced to the 11.2 Ica Client Object that switched ICO use the Common Connection Manager (CCM) and COM based events. This architectural change freed the ICO object from having to have an explicit windows message pump, and allows events to be raised/handled within the main thread of our single-threaded console application. A lot of info, but let's just say "thank you fine developers for making life easier" and be on our way …

```
Simple ICO Launch With Events – Program.cs
/// <summary>
/// This program demo's the basic of launching and ICO session from within an application.
/// </summary>
class Program
{
    public static AutoResetEvent onLogonResetEvent = null;

    static void Main(string[] args)
    {
        ICAClientClass ica = new ICAClientClass();
        onLogonResetEvent = new AutoResetEvent(false);

        // Launch published Notepad if you comment this line, and uncommented
        // the one above you will launch a desktop session
        // ica.Application = "";
        ica.InitialProgram = "#Notepad";

        // Launch a new session
        ica.Launch = true;

        // Set Server address
        ica.Address = "10.8.X.X";

        // No Password property is exposed (for security)
        // but can explicitly specify it by using the ICA File "password" property
        ica.Username = "johncat";
        ica.Domain = "xxxx";
        ica.SetProp("Password", "xxxx");

        // Let's have a "pretty" session
        ica.DesiredColor = ICAColorDepth.Color24Bit;

        // Reseach the output mode you want, depending on what your trying
        // to attempt to automate. The "Client Simulation APIs" are only available under c
        // (i.e. things like sending keys to the session, enumerating windows, etc.)
        ica.OutputMode = OutputMode.OutputModeNormal;

        // Height and Width
        ica.DesiredHRes = 1024;
        ica.DesiredVRes = 786;

        // Register for the OnLogon event
        ica.OnLogon += new _IICAClientEvents_OnLogonEventHandler(ica_OnLogon);

        // Launch/Connect to the session
        ica.Connect();

        if(onLogonResetEvent.WaitOne(new TimeSpan(0,2,0)))
            Console.WriteLine("Session Logged on sucessfully! And OnLogon Event was Fired!
        else
            Console.WriteLine("OnLogon event was NOT Fired! Logging off ...");

        // Do we have access to the client simulation APIs?
        if (ica.Session == null)
            Console.WriteLine("ICA.Session object is NULL! ☹ ");

        Console.WriteLine("\nPress any key to log off");
        Console.Read();

        // Logoff our session
        Console.WriteLine("Logging off Session");
        ica.Logoff();
    }

    /// <summary>
    /// OnLogon event handler
    /// </summary>
    static void ica_OnLogon()
    {
        Console.WriteLine("OnLogon event was Handled!");
        onLogonResetEvent.Set();
    }
}
```

Great! Now we know when our session is logged on. What else can we do? Well, how about we get our feet wet with something fun! Let's shift gears and try a stab at interacting with our session by consuming, what are termed, the "Client Simulation APIs" within the Ica Client Object.

Be Careful

Before attempting to consume the Client Simulation APIs (or more directly, the "Session interface object") you must be informed about another change that was introduced within the 11.2 client. There was added security to restrict access to the session object. Access is controlled via the client's registry. The following HKLM\Software\Citrix\ICA Client\CCM key must be created (it's not there by default). Under that key a DWORD value of 0 or 1 disables and enables access to the Session object. So make sure this is set properly, or you might risk destroying your monitor out of frustration, after launching multiple sessions that return a NULL Session object. Wait what was the definition of insanity again?

KB Article 123616 ICO Changes in 11.2 XenApp plug-in

The following code example illustrates the use of Client Simulation Keyboard APIs to type within our Notepad application session, take a screen shot and save it to the %temp% directory, and finally enumerate and output the top level windows within the session. Finally, you can take this opportunity to print out a crisp copy of your "screenshoted" session, hand it to your boss (or hang in your office/room), and declare that … indeed … "Greatness has been achieved!"

```
Simple ICO Launch With Events & Client Simulation – Program.cs
```

```
/// <summary>
/// This program demo's the basic of launching and ICO session from within an application.
/// </summary>
class Program
{
    /// <summary>
    /// Auto Reset Event used to block execution until we receive the OnLogon event
    /// </summary>
    public static AutoResetEvent onLogonResetEvent = null;

    static void Main(string[] args)
    {
        ICAClientClass ica = new ICAClientClass();
        onLogonResetEvent = new AutoResetEvent(false);

        // Launch published Notepad if you comment this line, and uncommented
        // the one above you will launch a desktop session
        // ica.Application = "";
        ica.InitialProgram = "#Notepad";

        // Launch a new session
        ica.Launch = true;

        // Set Server address
        ica.Address = "10.8.X.X";

        // No Password property is exposed (for security)
        // but can explicitly specify it by using the ICA File "password" property
        ica.Username = "johncat";
        ica.Domain = "xxxx";
        ica.SetProp("Password", "xxxx");

        // Let's have a "pretty" session
        ica.DesiredColor = ICAColorDepth.Color24Bit;

        // Reseach the output mode you want, depending on what your trying
        // to attempt to automate. The "Client Simulation APIs" are only available under c
        // (i.e. things like sending keys to the session, enumerating windows, etc.)
        ica.OutputMode = OutputMode.OutputModeNormal;

        // Height and Width
        ica.DesiredHRes = 1024;
        ica.DesiredVRes = 786;

        // Register for logon event
        ica.OnLogon += new _IICAClientEvents_OnLogonEventHandler(ica_OnLogon);

        // Launch/Connect to the session
        ica.Connect();

        if (onLogonResetEvent.WaitOne(new TimeSpan(0, 2, 0)))
            Console.WriteLine("Session Logged on sucessfully! And OnLogon Event was Fired!"
        else
            Console.WriteLine("OnLogon event was NOT Fired! Logging off ...");

        // Do we have access to the client simulation APIs?
        if (ica.Session == null)
            Console.WriteLine("ICA.Session object is NULL! ☹ ");

        // Give notepad a few seconds to setup
        // (or detect the window as seen below)
        // Obvious Tips: Sleep is usually bad in automation, race conditions ... now you k
        Thread.Sleep(5000);

        //////////////////////////////////////////////////////////////////////
        // Keyboard Simulation
        // Let's script some key input with the Keyboard interface
        Keyboard kbrd = ica.Session.Keyboard;

        // "Greatness has been achieved!" - Indeed!
        kbrd.SendKeyDown((int)Keys.G);
        kbrd.SendKeyDown((int)Keys.R);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.A);
        kbrd.SendKeyDown((int)Keys.T);
        kbrd.SendKeyDown((int)Keys.N);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.S);
        kbrd.SendKeyDown((int)Keys.S);

        kbrd.SendKeyDown((int)Keys.Space);

        kbrd.SendKeyDown((int)Keys.H);
        kbrd.SendKeyDown((int)Keys.A);
        kbrd.SendKeyDown((int)Keys.S);

        kbrd.SendKeyDown((int)Keys.Space);

        kbrd.SendKeyDown((int)Keys.B);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.N);

        kbrd.SendKeyDown((int)Keys.Space);

        kbrd.SendKeyDown((int)Keys.A);
        kbrd.SendKeyDown((int)Keys.C);
        kbrd.SendKeyDown((int)Keys.H);
        kbrd.SendKeyDown((int)Keys.I);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.V);
        kbrd.SendKeyDown((int)Keys.E);
        kbrd.SendKeyDown((int)Keys.D);

        // Exclamation !
        kbrd.SendKeyDown((int)Keys.ShiftKey);
        kbrd.SendKeyDown((int)Keys.D1);

        ////////////////////////////////////////////////////////////////////
        // ScreenShots
        // Let's take a screen shot of our session
        ScreenShot ss = ica.Session.CreateFullScreenShot();
        ss.Filename = Path.GetTempPath() + Path.GetRandomFileName() + ".bmp";
        Console.WriteLine("Saving Screen Shot to: " + ss.Filename);
        ss.Save();

        ///////////////////////////////////////////////////////////////////
        // Session Windows Enumeration
        // Enumerate top level windows and output them
        Windows allTopLevelWindows = ica.Session.TopLevelWindows;
        if (allTopLevelWindows.Count > 0)
        {
            Console.WriteLine("\nDisplaying all Top Level Windows:");

            foreach (window w in allTopLevelWindows)
            {
                Console.Write("\n");
                Console.WriteLine("Window ID: " + w.WindowID);
                Console.WriteLine("Window Caption: " + w.Caption);
                Console.WriteLine("Window Position X: " + w.PositionX);
                Console.WriteLine("Window Position Y: " + w.PositionY);
            }
        }

        Console.WriteLine("\nPress any key to log off");
        Console.Read();

        // Logoff our session
        Console.WriteLine("Logging off Session");
        ica.Logoff();
    }

    /// <summary>
    /// OnLogon Event Handler
    /// </summary>
    static void ica_OnLogon()
    {
        Console.WriteLine("OnLogon event was Handled!");
        onLogonResetEvent.Set();
    }
}
```

I've uploaded working examples to the code share which can be found here

Happy Coding!

- JohnCat

32 Comments

Anonymous

on 2 years ago · Reply

Thanks for the tutorial. This is exactly what I've been searching for.

Anonymous

on 2 years ago · Reply

Can tje Sumilation API be used with javascript or vbscript ?

> Hyma
>
> on 8 months ago · Reply
>
> Yes.It can be well used with Javascript or VBscript. Just that the embedding ICO part would be different. You would have to use the tag.

brian_cahill

on 2 years ago · Reply

I can't get this to work in VB.NET. Any ideas? Here is the code:
Imports SystemImports System.ThreadingImports WFICALib"' <summary> "' This program demo's the basic of launching and ICO session from within an application.

"' </summary> Class Program Public Shared onLogonResetEvent As AutoResetEvent = NothingShared Sub Main(ByVal args As String()) Dim ica As New ICAClientClass()onLogonResetEvent = New AutoResetEvent(False) ' Launch published Notepad if you comment this line, and uncommented ' the one above you will launch a desktop session ' ica.Application = ; ica.InitialProgram = #LaunchTest' Launch a new session ica.Launch = True' Set Server address ica.Address = 10.x.x.x' No Password property is exposed (for security) ' but can explicitly specify it by using the ICA File password property ica.Username = testica.Domain = testica.SetProp(Password, xxxx) ' Let's have a pretty session ica.DesiredColor = ICAColorDepth.Color24Bit' Reseach the output mode you want, depending on what your trying ' to attempt to automate. The Client Simulation APIs are only available under certain modes ' (i.e. things like sending keys to the session, enumerating windows, etc.) ica.OutputMode = OutputMode.OutputModeNormal' Register for the OnLogon event AddHandler ica.OnLogon, AddressOf ica_OnLogon ' Launch/Connect to the session ica.Connect()If onLogonResetEvent.WaitOne(New TimeSpan(0, 2, 0)) ThenConsole.WriteLine(Session Logged on sucessfully! And OnLogon Event was Fired!)

```
ElseConsole.WriteLine(OnLogon event was NOT Fired! Logging off …) End If' Do we have access to the
client simulation APIs? If ica.Session Is Nothing ThenConsole.WriteLine(ICA.Session object is NULL! 🙁)
End IfConsole.WriteLine(vbLf & Press any key to log off)
Console.Read()' Logoff our session Console.WriteLine(Logging off Session)
ica.Logoff()End Sub"' <summary> "' OnLogon event handler "' </summary> Private Shared Sub
ica_OnLogon()Console.WriteLine(OnLogon event was Handled!)
onLogonResetEvent.[Set]()End SubEnd Class
```

---

brian_cahill

on 2 years ago · Reply

To elaborate. The app launches but I don't get notification that it launched via the console.

---

brian_cahill

on 2 years ago · Reply

alright, I got this to work but I had to put a msgBox after the ica.connect. If I wait to hit OK on the message box, I get the messages in the console and everything seems to work fine.   Obviously, I don't want to interact with this to get it to test connections. What is different with VB.NET or am I missing something?

> hymanc
>
> on 1 year ago · Reply
>
> Hi Brian,
>
> Is it possible for you to attach the .vb source code to my e-mail hyma.nallanchakravarthula@citrix.com , so that I can try out some ways?

---

johncat

on 2 years ago · Reply

Hey Brian, sorry to hear about your troubles. What version of the ICA client are you trying to automate?

Feel free to send me an email john.cattaneo@citrix.com with your .vb source file (the formatting is pretty horrible trying to copy and paste what's above) and I can play around with it and/or try to figureout any specific issue you're having

---

Anonymous

on 2 years ago · Reply

Anyone figure how to get rid of extremey annoying Citrix on-line plug in cannot connect to the server which comes up every time I turn on my computer. Any help appreciated—please reply to:

allen.solomon@comcast.net, thanks.

---

millerj

on 2 years ago · Reply

Thanks for the tutorial, this gets me really close to where I need to be. I would like to allow the farm to do its load balancing instead of giving the server ip/name. Any way to do this?

---

johncat

on 2 years ago · Reply

Hey Jeff, glad that I was able to steer you in the right direction. As far as the load balancing goes that is one of the limitations of ICO at this time. Talking to one of the ICA principle developers, the ICO bypasses application enumeration, and causes the initial launch not to go through the web server.

The connection goes something like this:

web server -> xml relay -> IMA resolves IP Address to load balanced server, and returns it in the ICA file.

Since the web server portion of the connection is being omitted, load balancing doesn't occur. This would be a future enhancement needed in ICO.

However, I have something for you to possibly try.

Define the following settings within an ICA file (see below) then within your code that's consuming ICO, use the .LoadIcaFile() function of the ICO object (note: this call is asynchronous so ensure you're handling the proper events). Once the ICA files has been loaded into your ICO object, try a launch and see if the proper load balancing takes place.

```
HTTPBrowserAddress=Name of your server or application
BrowserProtocol=HTTPonTCP
```

Let me know what happens. (I'd try it myself except I'm 30,000 feet in the air on the way to Citrix Synergy)

> millerj
>
> on 2 years ago · Reply

John,

Thanks for the help again. Using the ICA file was the trick I needed. I now have a custom C# login app to use for our SSO initiative with Sentillion!

You rock! Thanks a million.

chunguang

on 1 year ago · Reply

Hi, Jeff, John,

I have a similar problem. My ICO object can connect and launch from server residing on the same machine with the WI server, but can not launch any application from other server in the same farm. Have you tried the workaround using LoadIcaFile()? Any other idea?

Thanks,
Chunguang

Anonymous

on 2 years ago · Reply

This works great for me to launch a new application. But my situation is that a user has already launched my .NET application from the Program Neighborhood, and now I would like the application to get information about itself using GetSessionCounter calls. Is there anyway to use this SDK (or MFCOM for that matter) to allow a .NET process to connect/attach to its own information?

Anonymous

on 2 years ago · Reply

Hi, I try to get a c# example up and running. It’s really frustrating. The Citrix Session starts, notepad application starts, I get Session Logged on sucessfully! And OnLogon Event was Fired! and then … I get ICA.Session object is NULL! 🙁. 🙁 indeed. I use the Pre11_2 client. Any ideas?

Many thanks in advance!

Best regards,

Jörg

johncat

on 2 years ago · Reply

Jorg, I also noticed that one of your posts says that you are using a client that is pre 11.2? If that is the case please reference the ICOLaunchPre11_2Client example that I posted in the code share. There are some subtle differences in the client architecture (of clients pre 11.2) that requires you to explicitly activate the Window Message Pump in order to handle proper events from the ICO client. That examples shows you how to do it within a separate thread.

Anonymous

on 2 years ago · Reply

Hello John,

many thanks for the examples and the tutorial!

I have got a really strange situation. The c# example SimpleICOLaunchWithEventsAndClientSimulation works fine except for the session object which is null and stays null. The session starts, notepad starts I receive events and the session object is null. It’s exactly as described above and I feel like or you might risk destroying your monitor out of frustration. I use the current CitrixOnlinePluginWeb and the CCM key in the registry is set to AllowSimulationAPI=1. Now I’ m stuck and cannot get the session object. Does anyone have an idea as to what the problem might be?

Many thanks in advance,

best regards,

Jörg

johncat

on 2 years ago · Reply

I asked around and found out that there are some subtle differenced between the full online plugin install vs. the web plugin. One of them being the web client is installed per user, so the CCM key and the AllowSimulationAPI=1 value should be set under HKCU (HKEY_CURRENT_USER). Try that out and let us know if that works out for you, and I can update my blog accordingly 🙂

Anonymous

on 2 years ago · Reply

Hi John,

many thanks for your inquiry! Unfortunately the situation still stays the same. Now I’ve got the CCM flag under HKCU and HKLM but still to no avail. I’ m quite helpless and there’s a clear and present danger that

I'll start throwing things at my monitor. The strange thing is that everything seems to work fine except for the session object. Do you have any other ideas? Do you need any other information to better understand this particular behaviour?

Many thanks in advance!

Best regards,

Jörg

Anonymous

on 2 years ago · Reply

Hi,

I was wondering whether the sending of keystrokes would work if we set our output mode as NonHeadLess ??

Thanks

---

tbroder1003

on 1 year ago · Reply

A question about the InitialProgram / Application properties:

I have a Windows Forms application with an ICA Client Object that I reuse for multiple sessions, e.g.:

Launch app, set ICA properties, launch session, log off session, set new ICA properties, launch session etc.

After setting the InitialProgram property to a value for a session, setting it to blank again afterwards doesn't seem to have any effect and all future sessions for that ICA object will use the value that was specified earlier. Using the Application property instead doesn't seem to be any different, neither does using SetProp.

Do you know anything about this issue?

- Thomas

Anonymous

on 1 year ago · Reply

They allow customers resellers to quickly attach solutions directly to the Citrix Infrastructure.http://www.simplyrest.com

---

Anonymous

on 1 year ago · Reply

an idea that started grew and has ultimately changed the way we view and use technology.http://www.simplyrest.comhttp://www.simplyrest.com

---

Anonymous

on 1 year ago · Reply

Hello;

We are trying to develop an application for a client who is using XepApp6 for App. Virtualization and XenServer for hardware virtualization.

Our application seeks for data on the PORT of the host computer @ Client which is running ICA, The data to this PORT is sent by another device in the local network..

Can you experts please let me know if, usind the Simulation SDK work for me?? Or is there any other approach to get this fixed??

I would be very thankfull if any of you could write me at sujay0000 [at] gmail [dot] com; if you would like to help me on this further ahead too..

Thanks.

Sujay

Anonymous

on 1 year ago · Reply

Currently we are developing application in c# 64bit system (Deployment configuration is 64bit). Can i get any 64bit support binaries for this.

Thanks,
Satya.

> Hyma
>
> on 8 months ago · Reply
>
> No Satya. There is no 64 bit version of ICO yet.

Joey Green

on 10 months ago · Reply

Is there a way to pass a parameter to the InitialApplication? For example if the initial application is Internet Explorer, how can I pass it a dynamic URL?

Suyesh Khandelwal

on 9 months ago · Reply

We are trying to run any of the 2 applications(SimpleICOLaunch, ICOLaunchPre11_2Client) as a service which uses ICA Client 11.2 ICO object but it fails to connect when call to client.Connect() method is made with error code 13 – Unsupported function.

And also while trying to connect ICA server using ica file (using same application mentioned above running as service) we are getting Error code 63 – No Window found.

Please note above two errors(error code 13 & 63) are only observed on clients 11.2 onwards, our application works flawlessly on ICA client 11.0 & lower versions.

Please comment on this issue.

---

annonymous

on 7 months ago · Reply

Hi I am trying to launch 2 applications simultaneosly but I am not able to achieve intercation among 2. what currently I am trying is I have simulated a calculator on ICA client. What I want now is to show the resuklt of calculator (1 of ICA hosted application) on notepad (other hosted ICA application).

but the keyboard interface is not pasting any result on noteapd.

Andrey Fedyashov

on 5 months ago · Reply

Hi John, I noticed that for some Citrix environments (probably using CAG) – the ICA Address field returned from the Web Interface is not actually an address but something like:

Address=;40;STA243C13F1E724;3762D0B8E82EB498580951563C5E598D

In this case the sample code is not able to connect (tried with ICA clients 11.2, 12.1).
Do you know any workaround for this?

Franco

on 1 month ago · Reply

Somebody has tried launch a xendesktop sesión with Ica client object ?

Shannon Wagner

on 1 week ago · Reply

I got the code working (thank you!) in order to connect and to detect a successful logon (program waits on the AutoResetEvent until OnLogon is fired). But I need to also handle logon / connect failures or aborts. I would prefer to do this without waiting for the timeout to expire since Id like the application to be more responsive to what is happening as it happens.

For example two scenarios I would like to handle: if the server is not accepting connections or the user hits Cancel instead of typing credentials.

Ive tried handling the following events, but none of them seem to fire in the case I am trying to handle: OnDisconnect, OnConnectFailed, OnLogonFailed

Is there an event which fires for this scenario?

Any help much appreciated!

Post a Comment

Name *(required)*

Email *(required, but never shared)*

Web

You have 2000 characters left.

Post Comment