<http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms>

# Programming Microsoft Agent in Windows Forms

By [**Mohammad Elsheimy**](http://www.codeproject.com/script/Membership/View.aspx?mid=4386833) | 28 Dec 2010 | [Unedited contribution](http://www.codeproject.com/script/Articles/Unedited.aspx)

[C#](http://www.codeproject.com/search.aspx?aidlst=81)[.NET](http://www.codeproject.com/search.aspx?aidlst=98)[COM](http://www.codeproject.com/search.aspx?aidlst=130)[Intermediate](http://www.codeproject.com/search.aspx?aidlst=153)[WinForm](http://www.codeproject.com/search.aspx?aidlst=1372)

Learn how to program with Microsoft Agent in Windows Forms applications

**See Also**

* [More like this](http://www.codeproject.com/search.aspx?q=Programming+Microsoft+Agent+in+Windows+Forms)
* [More by this author](http://www.codeproject.com/script/Articles/MemberArticles.aspx?amid=4386833)

[[http://s.codeproject.com/script/Content/Images/print.gif](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms?display=Print)](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms?display=Print)

[Print Article](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms?display=Print)

add

[[http://www.codeproject.com/images/AddTo_Twitter.gif](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Twitter](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Digg.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Digg](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Facebook.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Facebook](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Delicious.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Del.icio.us](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Reddit.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Reddit](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_stumbleupon.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Stumbleupon](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Newsvine.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Newsvine](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Technorati.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Technorati](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_MrWong.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Mr. Wong](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Yahoo.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Yahoo!](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Google.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Google](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/AddTo_Live.png](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Windows Live](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

[[http://www.codeproject.com/images/envelope.gif](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)Send as Email](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

Add to your CodeProject bookmarks

[[Discuss](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms#_comments)](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms" \l "_comments)

[Discuss this article](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms" \l "_comments)

16

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Article** | | [**Browse Code**](http://www.codeproject.com/script/Articles/ViewDownloads.aspx?aid=35921) | | [**Stats**](http://www.codeproject.com/script/Articles/Statistics.aspx?aid=35921) | [**Revisions (13)**](http://www.codeproject.com/script/Articles/ListVersions.aspx?aid=35921) | |
| http://s.codeproject.com/script/Ratings/Images/stars-fill.png  http://s.codeproject.com/script/Ratings/Images/stars-empty.png | | 4.85 (20 votes) | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| http://s.codeproject.com/Images/t.gif 1 | http://s.codeproject.com/Images/t.gif 2 | http://s.codeproject.com/Images/t.gif 3 | 3 votes, 15.0% 4 | 17 votes, 85.0% 5 |

4.85/5 - 20 votes

μ 4.85, σa 0.95 [[?](http://www.codeproject.com/KB/FAQs/RatingReputationFAQ.aspx#noisefilter)]

Principio del formulario





**This article is available in my blog too, check it out** [***here***](http://justlikeamagic.com/2009/04/06/programming-microsoft-agent-in-windows-forms/)**.**

**اقرأ النسخة العربية من هذا الموضوع** [**هنا**](http://withdotnet.net/2009/08/10/programming-microsoft-agent/)**.**

* [**Download source code, PartIt**](http://www.box.net/shared/ug3r3y3i14)
* [**Download standalone application installer for PartIt**](http://justlikeamagic.wordpress.com/2009/04/06/part-your-files-using-geming-partit/)

### Contents

* Contents
* Overview
* Microsoft Agent Characters
* Microsoft Agent SDK
* Microsoft Agent Windows Forms Support
* Loading Characters
* Ordering the Character to Speak
* Moving the Character
* Character Animation
* IAgentCtlCharacterEx Properties
* AxAgent Control Events
* Creating a Custom Popup Menu (**NEW**)
* Creating a Microsoft Agent Controller
* Freeing up Resources
* The Last Word
* Microsoft Agent Compatibility (**NEW**)
* Where to Go Next?
* A Real-World Example

### Overview

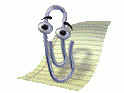
Microsoft Agent is an unprecedented technology to create innovative, new conversational interfaces for applications and Web pages. It provides powerful animation capability, interactivity, and versatility, with incredible ease of development. Microsoft Agent is a technology that provides a foundation for more natural ways for people to communicate with their computers. It is a set of software services that enable developers to incorporate interactive animated characters into their applications and Web pages. These characters can speak, via a **Text-to-Speech (TTS)** engine or recorded audio, and even accept spoken voice commands. Microsoft Agent empowers developers to extend the user interface beyond the conventional mouse and keyboard interactions prevalent today. Enhancing applications and Web pages with a visible interactive personality will both broaden and humanize the interaction between users and their computers. There are a limitless number of roles and functions that developers can create for these personalities to perform.

* A welcome host could greet new users and provide a guided tour the first time a computer is turned on, an application is run, or a Web site is browsed.
* A friendly tutor could lead someone through a task or a decision tree with instructions step-by-step along the way.
* A messenger could deliver a notification or alert that a new e-mail has arrived and then offer to read it to you.
* An assistant could perform tasks for you like looking up information on the Internet and then reading it out loud.

Although, this lesson focuses on Windows Forms, you can take advantage of Microsoft Agent in Web applications, and also XAML applications. For Web application, you can refer to the Microsoft Agent Platform SDK documentation. For XAML applications, it is the same as Windows Forms, except that you will need to add Microsoft Agent Control inside a *Sysetm.Windows.Forms.Integration.WindowsFormsHost* WPF control.

### Microsoft Agent characters

Microsoft Agent characters can be found in various applications and Websites, including Office 2003 and its ascendants, it comes as the Office Assistant. Figure 1 shows the Clippit office assistant. And figure 2 shows Merlin the most widely-used Microsoft Agent character in Websites and applications.



There're plenty of characters available for download. Though, Merlin is included by default in Windows. In addition, Office 2003 comes with many characters including Clippit.

Visit [*Microsoft Agent Ring*](http://www.msagentring.org/) and download plenty of characters.

### Microsoft Agent SDK

Programming Microsoft Agent is very easy. But, in order to start programming you need to download the Microsoft Agent SDK. It's available for downloadalong with its documentation in the [Microsoft Agent SDK](http://www.microsoft.com/msagent/downloads/developer.aspx) download page on Microsoft. In addition, you need to download the Speech API and Text-to-Speech (TTS) Engine in order to enable the speech feature of Microsoft Agent. These packages are available in the [Speech SDK](http://www.microsoft.com/downloads/details.aspx?FamilyID=5e86ec97-40a7-453f-b0ee-6583171b4530&DisplayLang=en) in this page. The most recent version is 5.1.

The Text-to-Speech (TTS) Engine is used to translate text into voice.

It is worth mentioning that, if you need to extend speech feature of Microsoft Agent to support other languages other than English, you need to download your preferred language package from the Speech SDK page. In addition, you are not end up using characters created for you. If you need to create your own characters you can download the Agent Character Editor and start using character creation. Take into consideration that, for the application to run correctly on user's machine, the user need to install two packages, Microsoft Agent and Microsoft TTS. Remember to inculde them as prerequisites in your application installer package. After downloading and installing the Microsoft Agent SDK, you may notice that Microsoft Agent SDK offers you many components for many purposes. The most component that we are interested in is the ActiveX COM library **AgentCtl.dll** that contains the Microsoft Agent support features for Windows Forms. It contains the Microsoft Agent Control that's the door for Microsoft Agent Windows Forms programming. This component -and many others- resides on **%windir\%MSAgent**.

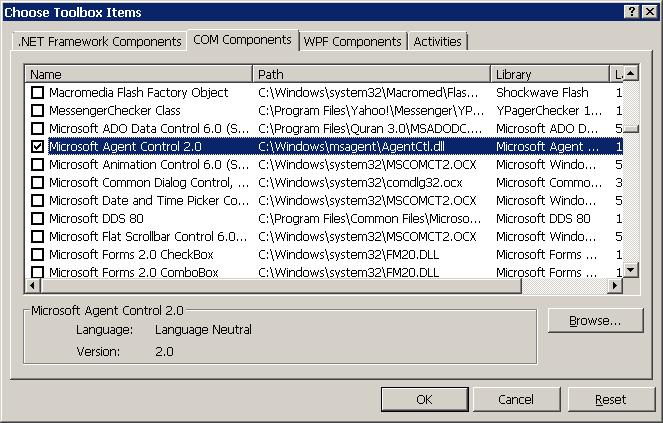
### Microsoft Agent Windows Forms Support

Because **AgentCtl.dll** is an ActiveX component you cannot use it directly in your code. You need to create a RCW (Runtime-Callable Wrapper) component (assembly) that will act as a bridge between .NET and the COM component. This is done automatically while adding the ActiveX component control to your toolbox in Visual Studio. Another way is to use the **AxImp.exe** tool that comes with the .NET Framework SDK.

You might be wandering why we haven't used the ***TlbImp.exe*** tool? Because this is an ActiveX COM component contains ActiveX controls. Therefore, we used the ***AxImp.exe***.

Because .NET and COM cannot call each other directly, two types of wrapper components exist in the space, RCWs (Runtime-Callable Wrappers) and CCWs (COM-Callable Wrappers). RCWs are .NET assemblies act as the bridge between .NET and COM. Conversely, CCWs are COM components act as the bridge between COM and .NET.

To use Microsoft Agent in our Windows Forms application, simply, we'll start by adding the Microsoft Agent Control to our toolbox. Figure 3 shows the Choose Toolbox Items dialog in the COM Components tab.

[](http://www.codeproject.com/KB/cs/agent-winforms/Microsoft_Agent_Control_in_Choose_Toolbox_Items.jpg)

This component contains a single control Microsoft Agent Control that will be added automatically to the toolbox. After dragging it to the form, two components will be added automatically to the project references, AgentObjects and AxAgentObjects. These are the interoperability RCW wrapper components for the COM component. AxAgentObjects is the wrapper for the control and its related types. On the other hand, AgentObjects component is the wrapper for other types in the COM component. You might have noticed that a square-shaped control of type AxAgentObjects.AxAgent added to the form. This control will not be visible at runtime. Figure 4 shows the form.



AgentCtl uses STA (Single-Threaded Apartment) threading model. Means, that the COM component supports -and bounds- only to a single thread. To enable STA for your application you need to adorn the *Main()* function with the STAThreadAttribute attribute. Conversely, MTA (Multi-Threaded Apartment) threading model means that the COM component supports and can be accessed by multiple threads. If neither STAThreadAttribute nor MTAThreadAttribute attributes has not been set, the threading model by default is MTA. Another way to set the threading model is the *System.Threading.Thread.SetApartmentState* ***instance (opposite of static)*** method. To know whether a COM component supports STA, MTA, or both threading models, you can check the registry value ThreadingModel on ***HKCR\CLSID\<com guid>\InprocServer32***.

### Loading Characters

After adding the AxAgent control to the form, you can now order the control to load the character(s) and show it. AxAgent control contains a property Characters that's of type AgentObjects.IAgentCtlCharacters that acts as a dictionary collection contains the loaded characters and its names. For loading a character we use the Load() method of the Characters collection. For getting the loaded character we use the Character() method that returns an object of type AgentObjects.IAgentCtlCharacterEx that represents the loaded character. Here's the code for loading the character Merlin and showing it:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

// Here I'll specify the character name

// and the character location.

this.axAgent1.Characters.Load("Merlin",

@"C:WindowsMSAgentcharsmerlin.acs");

AgentObjects.IAgentCtlCharacterEx character =

this.axAgent1.Characters.Character("Merlin");

// A single argument contains a value to whether

// to animate the character while showing it

// or skipping the animation.

// You should set this to null or false.

// Set it to True to skip the animation.

character.Show(null);

In addition to Load() and Character() methods of the IAgentCtlCharacters object, you can use the Unload() method to unload a given loaded character. Also IAgentCtlCharacterEx supports the Hide() method for hiding the character, that's opposite to the Show() method.

### Ordering the Character to Speak

After creating the character and showing it you can order it to speak whether a specific text using the Text-to-Speech (TTS) Engine or a recorded audio file. The following code segment makes the character introduces itself using coded string first and a recorded audio file next:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

character.Speak(

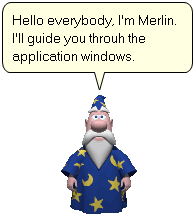
"Hello everybody, I'm Merlin. " +

"I'll guide you throuh the application windows."

, null);

character.Speak(null, @"D:Recorded Introduction.wav");

As you have noticed, the Speak() method can take either a text for the first argument or a file path for the second argument. Figure 5 shows Merlin while speaking the text.



Take a look at the *Think()* method, it is very similar to *Speak()*.

### Moving the Character

You can move the character around the screen by using the MoveTo() method.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

character.MoveTo(100, 100, null);

This method takes three arguments, the first and second argument is the location on the screen that you wish to move the character to. The third argument takes a value indicating the moving speed. If this argument is null, then the default value 1000 is used. If you specified the value 0 the character will move without playing an animation. Obviously, a value under 1000 will slow down the character move, and a value greater than 1000 will speed up the move.

I had reasons to merge this section with the animation section. But, I had reasons too to put it in a single section, I think it's a special type of animations!

### Character Animation

Every character has its own animations. Although, many characters share many animation names but that's not required. Every animation has a name. And to get the animation names that a character supports, you need to dig into the AnimationNames property that's of type AgentObjects.IAgentCtlAnimationNames which implements the System.Collections.IEnumerable interface. The following code enumerates the animation names of the character:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

IEnumerator enumerator =

character.AnimationNames.GetEnumerator();

while (enumerator.MoveNext())

this.listBox1.Items.Add(enumerator.Current.ToString());

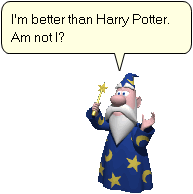
Now, it's time to have some fun with the character. Try the following code:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

character.Play("DoMagic1");

character.Speak("I'm better than Harry Potter. Am not I?", null);

Figure 6 shows Merlin speaking while doing some magic. (Proud Merlin)



Be aware that animation names that contain directions like left and right, it's the left and right of the character not you.

In addition to the Play() method, the character object supports two methods for stopping playing animations, Stop() and StopAll(). Stop() stops the animation that's specified by its single argument. StopAll() stops all animation of a specific type. This method has a single argument that can takes a string value of three "Move", "Play", and "Speak" to stop moving, animation or speaking commands. In addition to the three values, StopAll() can take a null value, and that means stopping all types of animations.

You might have noticed that many methods of the character object returns an object of type *AgentObjects.IAgentCtlRequest*. This type encapsulates a request to the character. If *IAgentCtlRequest.Status* is non-zero then the operation failed, otherwise succeeded. Pay attention to the methods that return *IAgentCtlRequest* object, and the methods that requires it as an argument like *Stop()* method.

Another type of animation is gesturing to a specific point. Try this animation using the GestureAt() method.

### IAgentCtlCharacterEx Properties

This list contains the most common properties of the character object (IAgentCtlCharacterEx):

* **AutoPopupMenu:**  
  A Boolean property indicates whether to allow the character pop-up menu or not. Pop-up menu is the menu that is shown when the user right-clicks on the character or its icon -if exists- on the taskbar. Default value is False.
* **Balloon:**  
  This is a read-only property that's in turn contains read-only properties that indicate the characteristics like the back color of the balloon displayed while the character talking. This property contains many properties that most are self-explained from its names.
* **Left and Top:**  
  Controls the character location on the screen. (It is better using the MoveTo() method instead of these two properties if you want to play an animation while moving the character).
* **MoveCause:**  
  A read-only property returns a value indicates what caused the character's last move. This property can return one of 5 values:
  + **0:**  
    The character has not been moved.
  + **1:**  
    The user moved the character.
  + **2:**  
    Your application moved the character.
  + **3:**  
    Another client application moved the character.
  + **4:**  
    The Agent server moved the character to keep it onscreen after a screen resolution change.
* **Pitch:**  
  A read-only property returns a value indicates the Pitch setting of the TTS.
* **SoundEffectsOn:**  
  Set this property to True to enable the sound effects, of False to disable it. Default is True.
* **Speed:**A read-only property indicates the speed of the character. (Also there're some operations that supports specifying the speed like the MoveTo() method).
* **VisibilityCause:**  
  A read-only property returns a value indicates the character's visibility state. This property can return one of 8 values:
  + **0:**  
    The character has not been shown.
  + **1:**  
    User hid the character using the command on the character's taskbar icon pop-up menu or using speech input.
  + **2:**  
    The user showed the character.
  + **3:**  
    Your application hid the character.
  + **4:**  
    Your application showed the character.
  + **5:**  
    Another client application hid the character.
  + **6:**  
    Another client application showed the character.
  + **7:**  
    The user hid the character using the command on the character's pop-up menu.

Agent Server is the hard-core component that controls the Microsoft Agent behind the scenes. It controls all the characters loaded by your application and other applications and Websites. It's the one which servers all orders for every Microsoft Agent character. Worth mentioning, it's contained in the executable file ***AgentSvr.exe*** located on ***%windir%MSAgent***. This file will be launched automatically when the first character loaded, and will shuts down automatically when the last character unloaded. After all, you can think about the Agent Server as the hard-core internal component of the Microsoft Agent.

### AxAgent Control Events

An odd behavior of that control is that it defines events that related to the character and most likely to be located in the character object itself. Maybe it's a feature to be located here because AxAgent controls the birth and death of the characters, but most of the time you will find that annoying. The most widely-used events are:

* **BalloonShow and BalloonHide:**  
  Occur when a balloon of a loaded character shown or hid.
* **ClickEvent and DblClick:**  
  Occur when a loaded character clicked or double-clicked.
* **DragStart and DragComplete:**  
  Occur when a user tries to drag a loaded character.
* **ShowEvent and HideEvent:**  
  Occur when a loaded character shown or hide, whether the user who made this or your code. If you want to check what caused the character to be shown check the IAgentCtlCharacterEx.VisibilityCause. As a refresher, read the last section to know about this property.
* **MoveEvent:**  
  Occurs when a loaded character moved whether the user moved it or it's moved by your code. See the IAgentCtlCharacterEx.MoveCause in the previous section if you want to know what caused the character to be moved.
* **RequestStart and RequestComplete:**  
  Occur when a request being restarted or after it completed. Note that many methods accept and many others return IAgentCtlRequest objects that defines requests sent to the server.

### Creating a Custom Popup Menu

After you have included your desired agent in your application, are you feeling bad with the default popup menu? If so, then you are in the right place.

First, create your System.Windows.Forms.ContextMenuStrip and add your required items (well, including “Hide” maybe) and complete the item event handlers.

Now, let’s get it. Get to the code that loads the agent character (e.g. calls the Characters.Load() method of the agent control object, AxAgentObjects.AxAgent) and just disable the AutoPopupMenu flag/property of the character object, AgentObjects.IAgentCtlCharacterEx. This flag/property determines whether to allow the default popup menu or not.

For example, the following code disables this property:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

AxAgentObjects.AxAgent agentCtl;

AgentObjects.IAgentCtlCharacterEx agentChar;

// initializing 'agentCtl'

// . . .

agentCtl.Characters.Load("Merlin", "merlin.acs");

agentChar = agentCtl.Characters.Character("Merlin");

agentChar.AutoPopupMenu = false;

Next comes the interesting point. When the character is clicked, the ClickEvent event of the agent control (AxAgent) fires. So the next step is to handle this event and to provide your code that brings up your custom context menu. Consider the following code:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

// agentCtl.ClickEvent += agent\_ClickEvent;

public void agentCtl\_ClickEvent(object sender, AxAgentObjects.\_AgentEvents\_ClickEvent e)

{

if (e.characterID == "Merlin") // check for this if you have many characters

{

if (e.button == 2) // 1 = left, 2 = right

{

myContextMenu.Show(e.x, e.y);

}

}

}

### Creating a Microsoft Agent Controller

When dealing with Microsoft Agent, there's recommended practices. One of is abstracting the Microsoft Agent implementation code from the UI code for achieving the maintainability and usability goals, and these goals can be achieved through a controller class that encapsulates the functionality that you need and provides easy access to lots of operations using a single call. A good controller class example is like this:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

internal sealed class AgentController

{

private AgentObjects.IAgentCtlCharacterEx \_char;

private AxAgentObjects.AxAgent \_agent;

public AgentController(AxAgentObjects.AxAgent agent,

string characterLocation)

{

\_agent = agent;

\_agent.Characters.Load("CHAR", characterLocation);

\_char = \_agent.Characters.Character("CHAR");

\_char.Show(null);

}

public bool IsVisible() { return \_char.Visible; }

public void Animate(string animation, bool stopAll)

{

if (stopAll)

\_char.StopAll(null);

\_char.Play(animation);

}

public void Speak(string text, bool stopAll)

{

if (stopAll)

\_char.StopAll(null);

\_char.Speak(text, null);

}

public void MoveTo(int x, int y, bool stopAll)

{

if (stopAll)

\_char.StopAll(null);

\_char.MoveTo((short)x, (short)y, null);

}

// Encapsulating many operations into one

public void Surprise()

{

\_char.StopAll(null);

\_char.Play("Alert");

\_char.Play("Sad");

}

}

### Freeing up Resources

After finishing your work with Microsoft Agent you should first unload the character by calling AxAgent.Characters.Unload() method. Second, because all objects of the RCW (Runtime-Callable Wrapper) assemblies created earlier are just a wrapper around the unmanaged COM objects, they exhaust a great amount of unmanaged resources that can retain in memory for a long time, you strictly should always dispose these object when you have done with it. Fortunately, AxAgent control inherits indirectly from the IDisposable interface, so you can easily dispose it by calling its Dispose() method. Conversely, most objects (like the character object IAgentCtlCharacterEx) in the RCWs do not implement this interface, so you need to do it the hard way and use the Marshal class to dispose it. Here's the code that can do that:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

while (System.Runtime.InteropServices.

Marshal.FinalReleaseComObject(\_char) > 0);

The FinalReleaseComObject() simple tries to decrement the references of the COM object to 0. You need to call FinalReleaseComObject() again and again until it returns 0, means that no references remain.

FinalReleaseComObject() tries to decrement the reference count to 0 in one time, *ReleaseComObject()* tries to decrement it gradually. So *ReleaseComObject()* needs more calls than *FinalReleaseComObject().* and *FinalReleaseComObject()* is faster than *ReleaseComObject().* Decrementing object's references means disposing it.

You learned how to free-up the resources. But, there's a question now. At which point should you dispose objects? It depends! The point at which you need to dispose objects depends on your use of objects. First, if you finished working with the AxAgent control and you do not want it any more till the user closes your application, it's recommended that you dispose it as soon as possible. Otherwise, add the Dispose() call to the protected Dispose(bool) method of the form, so that it disposed when the form closes (honestly, disposes). The following code snippet demonstrates this:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

protected override void Dispose(bool disposing)

{

try

{

if (disposing)

{

this.axAgent1.Dispose();

if (components != null)

components.Dispose();

}

}

finally { base.Dispose(disposing); }

}

This code strictly follows the disposing design guidelines.

You can step to the *Dispose()* method from the Members combo box in the top-right of the Code Editor window of the form.

A *programming* extremist like me (one of the Qaeda squad *programming* team :P) always disposes his objects even if the application is shutting down. To be honest, when the application shuts down all resources and memory managed by the application will be released immediately, so you actually do not need to release objects while the application shuts down.

On the other hand, for some objects like IAgentCtlCharacterEx you should take the Marshal class approach. But where to use it? A great location for the FinalReleaseComObject() method (or ReleaseComObject() as well) is in the controller class. You can make your controller class implements the IDisposable interface and add your code inside the Dispose() method. See the following example:

Code abbreviated for clarity.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/35921/Programming-Microsoft-Agent-in-Windows-Forms)

internal sealed class AgentController : IDisposable

{

............

public void Dispose()

{

\_char.StopAll(null);

\_char.Hide(null);

\_agent.Characters.Unload("CHAR");

while (System.Runtime.InteropServices.

Marshal.FinalReleaseComObject(\_char) > 0) ;

}

}

It's not a joke, but you should dispose the controller class as soon as you finish using your character, or even with the form disposal along with the disposing of the *AxAgent* control.

### The Last Word

Needless to say that you can take advantage of the Microsoft Agent and TTS technology to produce applications that provide friendly tutor for the user. You are not end up using existing Microsoft Agent characters, you can create your own characters and animations using the Microsoft Agent Character Editor that's available for download in the Microsoft Agent SDK download page.

### Microsoft Agent Compatibility

Microsoft announced that Windows 7 would not be compatible with Microsoft Agent. However, you still can install the SDK and run your applications correctly.

### Where to Go Next?

Good references for Microsoft Agent are:

* Microsoft Agent SDK Documentation. Available for download with the SDK, see "Micrososft Agent SDK" section on the top.
* Book: Microsoft Agent Software Development Kit.  
  ISBN: 0-7356-0567-X
* Book: Developing for Microsoft Agent.  
  ISBN: 1-5723-1720-5

### A Real-World Example

This lesson does not contain a sample project, it contains a real-world application, Geming PartIt! that's used for parting large files into small pieces for merging them later, so you can send them via e-mail or storing them on CDs and Floppy Disks. This application uses Microsoft Agent character Merlin to provide a friendly guide for the user through the application.

Final del formulario

<http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent>

# Announcer in C#.NET using Microsoft Agent

By [**Ashraf Mohamed**](http://www.codeproject.com/script/Membership/View.aspx?mid=329708) | 11 Jun 2003

[.NET1.0](http://www.codeproject.com/search.aspx?aidlst=10)[Win2K](http://www.codeproject.com/search.aspx?aidlst=23)[C++](http://www.codeproject.com/search.aspx?aidlst=78)[C#](http://www.codeproject.com/search.aspx?aidlst=81)[MFC](http://www.codeproject.com/search.aspx?aidlst=107)[Dev](http://www.codeproject.com/search.aspx?aidlst=118)[Intermediate](http://www.codeproject.com/search.aspx?aidlst=153)

A tool to announce message, sing a song and tell a joke.

**See Also**

* [More like this](http://www.codeproject.com/search.aspx?q=Announcer+in+C%23.NET+using+Microsoft+Agent)
* [More by this author](http://www.codeproject.com/script/Articles/MemberArticles.aspx?amid=329708)

[[http://s.codeproject.com/script/Content/Images/print.gif](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent?display=Print)](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent?display=Print)

[Print Article](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent?display=Print)

add

[[http://www.codeproject.com/images/AddTo_Twitter.gif](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Twitter](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Digg.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Digg](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Facebook.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Facebook](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Delicious.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Del.icio.us](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Reddit.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Reddit](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_stumbleupon.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Stumbleupon](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Newsvine.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Newsvine](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Technorati.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Technorati](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_MrWong.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Mr. Wong](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Yahoo.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Yahoo!](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Google.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Google](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/AddTo_Live.png](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Windows Live](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

[[http://www.codeproject.com/images/envelope.gif](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)Send as Email](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

Add to your CodeProject bookmarks

[[Discuss](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent#_comments)](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent" \l "_comments)

[Discuss this article](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent" \l "_comments)

19

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Article** | | [**Browse Code**](http://www.codeproject.com/script/Articles/ViewDownloads.aspx?aid=4305) | | [**Stats**](http://www.codeproject.com/script/Articles/Statistics.aspx?aid=4305) | [**Revisions**](http://www.codeproject.com/script/Articles/ListVersions.aspx?aid=4305) | |
| http://s.codeproject.com/script/Ratings/Images/stars-fill.png  http://s.codeproject.com/script/Ratings/Images/stars-empty.png | | 4.50 (18 votes) | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 vote, 5.6% 1 | 1 vote, 5.6% 2 | 1 vote, 5.6% 3 | 6 votes, 33.3% 4 | 9 votes, 50.0% 5 |

4.50/5 - 18 votes

2 removed

μ 4.10, σa 2.01 [[?](http://www.codeproject.com/KB/FAQs/RatingReputationFAQ.aspx#noisefilter)]

Principio del formulario





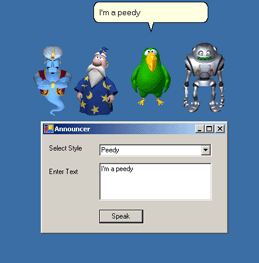
* [Download demo project - 28.9 Kb](http://www.codeproject.com/KB/cs/announcer/Announcer_demo.zip)



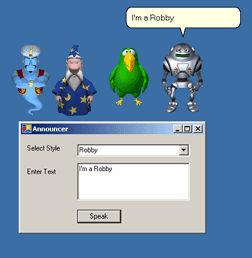
**Picture 1.** I'm genie



**Picture 2.** I'm Merlin



**Picture 3.** I'm Peedy



**Picture 4.** I'm Robby

## Overview

Microsoft® Agent is a software technology that enables an enriched form of user interaction that can make using and learning to use a computer, easier and more natural. Microsoft® Agent is a set of programmable software services that supports the presentation of interactive animated characters within the Microsoft Windows® interface I'm using Microsoft® Agent to develop this announcer. I have used four type of announcers here. They are Genie, Merlin, Peedy and Robby . More.. <http://www.microsoft.com/msagent/default.asp>.

This class demonstrates the use of following namespaces.

* using System.Runtime.InteropServices
* using System.Reflection
* AgentServerObjects
* AgentObjects

## Things required, where to get and how to install

1. You have to add reference *AgentObjects.dll* and *AgentServerObjects.dll*. You can get this two DLLs from *F:\Program Files\Microsoft Visual Studio .NET\ FrameworkSDK\Samples\Technologies\Interop\Applications\MSAgent\AgentExplorer*. Bear in mind, you have installed .NET in F:.

Or Just go to *F:\Program Files\Microsoft Visual Studio .NET\FrameworkSDK\Samples\Technologies\Interop\Applications\MSAgent\Hello1* and take a look at read *me.htm* file how to get those DLLs. You just give, nmake all. This command will create *AgentObjects.dll* and *AgentServerObjects.dll* in *hello1* folder.

Or just use those DLLs from this setup. I have bind these DLLs with this setup.

Assure those 2 DLLs in bin debug/release folder and add reference to solution explorer in you project.

1. We have to put *.acs* in debug or release folder. Announcer will show error if *.acs* file is not there. You can get four *.acs* files from

<http://www.microsoft.com/msagent/downloads/user.asp>. There you get genie, Merlin, Peedy and Robby, Microsoft agent character files (*.acs*)

and put into *release* or *debug* folder.

1. Just download and open the *MicrosoftAgentApplication.sln* file to use it.

Before run this, you have to install Text to speech engine. Download Text-to-Speech Engineer *tv\_enua.exe* American English - 1 MB from here <http://www.microsoft.com/msagent/downloads/user.asp>. I could not add this with my demo setup due to size restriction.

Just click *tv\_enua.exe* to install. Then announcer is ready to announce anything, sing a song and tell a joke.

## Source code

I have given sample code to use it.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

using System.Data;

using System.Drawing;

using System.Reflection;

using System.Collections;

using System.Windows.Forms;

using System.ComponentModel;

using System.Runtime.InteropServices;

using AgentObjects;

using AgentServerObjects;

Before we make a function to create an error files, we have to declare some variables that will be used in our function. This is an example of variables declaration and the main function is ErrorRoutine.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/4305/Announcer-in-C-NET-using-Microsoft-Agent)

AgentServer Srv = new AgentServer(); //Create a agent server

//If unable to create show the error

if (Srv == null)

{

MessageBox.Show("ERROR: Agent Server couldn't be started!");

}

IAgentEx SrvEx;

// The following cast does the QueryInterface to

//fetch IAgentEx interface from the IAgent interface,

//directly supported by the object

SrvEx = (IAgentEx) Srv;

// First try to load the default character

int dwCharID=0, dwReqID=0;

try

{

// null is used where VT\_EMPTY variant is expected by the COM object

String strAgentCharacterFile = null;

//Check microsoft agent character filename(.acs) is empty

if (!strFileName.Equals(string.Empty))

{

//Get the acs path

strAgentCharacterFile = strPath + strFileName;

}

else

{

MessageBox.Show("Select Style");

return;

}

if (!TxtSpeakInput.Text.Equals(string.Empty))

{

//load the acs file

SrvEx.Load(strAgentCharacterFile, out dwCharID, out dwReqID);

}

else

{

MessageBox.Show("Enter Text");

return;

}

}

catch (Exception)

{

MessageBox.Show("Failed to load Agent character! Exception details:");

}

IAgentCharacterEx CharacterEx=null;

SrvEx.GetCharacterEx(dwCharID, out CharacterEx);

//CharacterEx.SetLanguageID(MAKELANGID(LANG\_ENGLISH, SUBLANG\_ENGLISH\_US));

// Show the character. The first parameter tells Microsoft

// Agent to show the character by playing an animation.

CharacterEx.Show(0, out dwReqID);

// Make the character speak

// Second parameter will be transferred to the COM object as NULL

CharacterEx.Speak(TxtSpeakInput.Text, null, out dwReqID);

}

Here, We can select a different announcer and enter different text to speak. Just try it. It is simple to use.

We can create our own character files. Just visit <http://www.microsoft.com/msagent/downloads/user.asp> to explore more..

## License

This article has no explicit license attached to it but may contain usage terms in the article text or the download files themselves. If in doubt please contact the author via the discussion board below.

A list of licenses authors might use can be found [here](http://www.codeproject.com/info/Licenses.aspx)

## About the Author

|  |  |
| --- | --- |
| [**Ashraf Mohamed**](http://www.codeproject.com/Members/Ashraf-Mohamed)  http://www.codeproject.com/script/Membership/ProfileImages/%7b82939985-4ED8-4A63-A1A2-35D76A40F9EC%7d.gif  Web Developer  United StatesUnited States  Member | I am a system analyst and have been with Microsoft & Sun Technologies for more than 7 years. I have always been fascinated by java and .NET. I take lot of technical articles and writing them.   I am a Sun Certified Java Programmer for Java 2 Platform 1.4 , Web component developer Java 2 Platform - Enterprise Edition 1.4 and Microsoft certified developer using C#.NET in Web Development ASP.NET.   Visit my web site www.mohamedashraf.tk |

Final del formulario