

vuAgent Version 1.10



Installation and Instruction Manual

vuAgent Version 1.10 consists of a 32-bit Dynamic Link Library (DLL) and an executable file that work in concert to provide an extremely easy to use interface and implementation of Microsoft's agents, i.e., Merlin, Genie, Peedy, and Robby. Since all agents follow a common protocol, vuAgent will work with all Microsoft-compatible agents, such as those available from third-party vendors. Please note that vuAgent ships with Merlin only.

First, a friendly word about Microsoft agents. Microsoft offers four (4) agents that you can download and use freely (Merlin, Genie, Peedy, and Robby). In addition, you can obtain a license from Microsoft that allows you to distribute these agents with your application (available online and also free of charge). In addition, a number of companies offer agents that can be distributed freely and those that can be purchased for your own use. A good source for agent files is <http://www.msagentring.org>, offering information, support, and downloads for agents and their support files (including text to speech engines for languages other than English). vuAgent is compatible with all agents and support files that follow the Microsoft Guidelines for Agents.

Please Read This Carefully

Valutilities.com does not sell Microsoft agents or their support files (text to speech, etc), but rather provides an easy to use and intuitive interface for inclusion into your programs. Although the agents and associated support files are included in the distribution file (as a convenience to our customers), Valutilities.com does not warrant these agents nor guarantees their performance or fitness for any particular use or function.

Please read the End Users License Agreement (EULA) carefully prior to installing this utility. Installation or use of this utility constitutes acceptance of the terms and conditions of the EULA. If you do not agree to be bound by the EULA, please return the software to the place of purchase prior to installation.

In the functional write-ups, when a calling parameter is given, a descriptive name is used. This does not mean that you must use this name. You may use any variable name as long as it is defined as the same type (CString, Long, etc). Also, the length of the CStrings is arbitrary as well, and can be any length as long as the entire data will fit in the CString (plus 1 character for the terminating NULL).

The distribution package of vuAgent includes the following files:

| | |
|-------------------|--|
| License.txt | The End Users License Agreement (EULA) |
| VULogos.gif | The Valutilities Logo |
| vuAgentManual.pdf | This Document |
| vuAgent.dll | The Royalty Free DLL file to include in your distribution list |
| vuATray.exe | The Royalty Free EXE file to include in your distribution list |
| vuAgt.lib | The library source file necessary for Clarion to understand the DLL. |
| vuAgent.tpl | The Template that gets registered in Clarion |
| vuAgentDemo.exe | The compiled sample Agent Demo program. |
| vuAgentDemo.app | The Clarion Agent Demo source code written in Clarion 5.5 |
| vuAgent.dct | The Clarion Agent Demo dictionary necessary to compile the app |
| Actions.tps | The database file used by the Agent Demo program |
| MSAgent.exe | Microsoft's agent interface (Run during installation) |
| spchapi.exe | Microsoft's agent speech api (Run during installation) |
| spchcpl.exe | Microsoft's speech control panel (Run during installation) |
| tv_enua.exe | Lernout & Hauspie text to speech engine (Run during installation) |
| Merlin.exe | Merlin Agent Installation File (Run during installation) |
| Merlin.pdf | Merlin Agent Command Detail List |

NOTE: If including any other agents in your program, be sure to include the agent installation file (i.e., Merlin.exe) and execute it during your application's installation.

vuAgent Version 1.10



Support

If you have difficulty with any function, please be specific with your feedback (operating system, etc.) and provide source code if possible. You may contact support@valutilities.com at anytime. Your email will usually be answered by the close of the next business day. Their may be times when technical support will be unavailable for a period of time. Valutilities will make every effort to inform the Clarion community prior to this occurring. Messages will be posted in the Third Party news group at SoftVelocity.

Version History

| | |
|--------------|---|
| Version 1.0 | Initial Release of vuAgent |
| Version 1.01 | LibMain was inadvertently being exported which caused a conflict with vuFileTools. |
| Version 1.10 | Increased command size from 386 bytes to 32,768 bytes (32K) Increased Speech and Think text length from 128 bytes to 32,768 bytes (32K) Added 1 new function: <code>vuAgentInitEx(AgentName, fShow)</code> – Initializes the agent and allows you to show (or hide) the Agent after initialization (vuAgentInit() always hid the agent after initialization, making it difficult to know when it is safe to issue commands). |

Template Registration

If you have successfully completed the installation program, all of the files you need have been copied to the proper directories required for use. However, the Valutilities installation program **DOES NOT AUTOMATICALLY REGISTER THE TEMPLATE**. Registering the template is an easy, straightforward process and needs to be performed only once. Follow the steps outlined below. If you have any questions, please do not hesitate to contact us at support@valutilities.com.

1. Start Clarion. DO NOT load or open an application. If Clarion opens with a “Pick List”, hit the CANCEL button.
2. From the menu across the top, click on “Setup” and then “Template Registry”.
3. When the “Template Registry” window opens, press the “Register” button in the upper right-hand corner.
4. A window will open with a list of files in the Clarion Template directory. Scroll over until you see “vuAgent.tpl” and double-click on it (or highlight it and press the “Open” button).
5. THAT’S IT! You have now successfully registered vuAgent! Close the “Template Registry” window and open or create your application.

Using vuAgent in your Applications

In order to use the vuAgent functions in your application, you must globally initialize them. You will only need to perform this step once for each application that is using vuAgent.

1. After loading or defining your application, press the “Global” button on the toolbar (this is also represented by a light blue globe in Clarion 6). This will open up the “Extension and Control Templates Window”.
2. Pressing “Insert” will open the “Select Extension” Window.
3. Scroll down until you see the folder “**Class vuAgent – Valutilities – vuAgent Version 1.10**”
4. In this folder you will see “**vuAgent_Global – vuAgent Initialization**”
5. Either double-click this file or highlight it and press the “Select” button.
6. Press the “OK” button on the Extension and Control Templates window.

CONGRATULATIONS! You can now use any of the vuAgent functions, anywhere in your program.

vuAgent Version 1.10



Incorporating and starting vuAgent in your application takes exactly one (1) function call...vuAgentInit(). To initialize and use Merlin for example, embed the following 2 lines of code in anywhere in Window Init:

```
cAgent = 'Merlin'      !Assuming cAgent is a CString of at least 7 characters
vuAgentInit(cAgent)   !DONE – Merlin is now yours to command!
```

Or use the new init function to display the agent after initialization.

```
vuAgentInitEx(cAgent, 1)
```

Oh, and don't forget to terminate Merlin when your done. That takes exactly one (1) call as well:

```
vuAgentKill()        !DONE – Put this function anywhere in the end of your application
```

A complete demonstration application, including source code, is shipped with vuAgent to help you get started.

Deploying vuAgent Enabled Applications

“You won't believe how easy it is to deploy vuAgent in your applications!”

Everything you need is included with your vuAgent Installation file, including the Microsoft Agent support files necessary to load on your client's machine(s)

Step 1

Be sure to distribute both vuAgent.dll and vuATray.exe files in your application's directory (royalty free of course).

NOTE: *These files can be found under C:\C55\Valutilities\vuAgent\Distribute*

Step 2

In your setup file, at the conclusion of your installation, run the following five (5) files in this order (this is a piece of cake using Lindersoft's Setup Builder):

| | |
|-------------|--|
| MSAGENT.EXE | - Microsoft's Agent Management Files |
| SPCHAPI.EXE | - Microsoft's Speech API |
| SPCHCPL.EXE | - Microsoft's Speech Control Panel (adds Text to Speech Management) |
| TV_ENUA.EXE | - Lernout & Hauspie text to speech engine (That's right, your Agents will be speech enabled – at no add'l cost) |
| MERLIN.EXE | - Microsoft's Merlin agent (used in the demo) (Or any of the over 50 other Agent files that are freely available) |

NOTE: *All of these files can be found under C:\C55\Valutilities\vuAgent\MS_Files*

Step 3

There is no step 3, you are done!

Calling vuAgent Functions within your Application

PLEASE NOTE: Most functions require calling parameters. You must create and use the proper data elements for each function. Although you can use a number directly in a function (where appropriate), you **CAN NOT** use a literal string (you must assign the string value to a CSTRING variable and call the function with that variable).

We have made every effort to ensure that these functions will work well for you. If you have any problems, please contact us at support@valutilites.com. You will normally get a return email within one working day.

All of the exported calls in vuAgent are declared as Functions. This means that the compiler will be expecting you to use them as if they are a variable. If you use a function without testing the return value, the Clarion compiler may give you a warning. This is not a problem. Please see each functional description below for a complete list of calling parameters and return values.

vuAgent Version 1.10



vuAgentInit(AgentName) – Automatically initializes the selected agent, and starts the vuATray.exe file.

Calling Parameters

AgentName CString(32)

Calling Parameters Values

AgentName = The name of a previously installed Agent (Merlin, etc.)

Returns

Always returns true. However, if the vuATray.exe application has difficulty locating the agent or agent initialization files, it will post an error message and automatically terminate.

Example

```
AgentName = 'Merlin'  
vuAgentInit(AgentName)
```

NOTE: The Agent MUST be initialized before it can be used. Only one agent can be initialized per application. If the agent is properly initialized, vuATray.exe will place an icon in the system tray. A right-click on this icon allows the operator to show/hide the agent.

vuAgentInitEx(AgentName, fShow) – Automatically initializes the selected agent and starts the vuATray.exe file, optionally displaying or hiding the Agent after initialization.

Calling Parameters

AgentName CString(32)
fShow Long

Calling Parameters Values

AgentName = The name of a previously installed Agent (Merlin, etc.)
fShow = 0 to hide or 1 to show the Agent after initialization is complete.

Returns

Always returns true. However, if the vuATray.exe application has difficulty locating the agent or agent initialization files, it will post an error message and automatically terminate.

Example

```
AgentName = 'Merlin'  
vuAgentInit(AgentName, 1)
```

NOTE: The Agent MUST be initialized before it can be used. Only one agent can be initialized per application. If the agent is properly initialized, vuATray.exe will place an icon in the system tray. A right-click on this icon allows the operator to show/hide the agent.

vuAgentKill() – Terminates the current agent.

Calling Parameters

NONE

Calling Parameters Values

NONE

Returns

True on success

Examples

```
vuAgentKill()
```

NOTE: Although an agent that is visible on the screen will hide immediately, it may take up to three (3) seconds for the tray icon to disappear.

vuAgent Version 1.10



vuAgentHide() – Hides the agent from view.

Calling Parameters

NONE

Calling Parameter Values

NONE

Return

True on success

Example

vuAgentHide()

NOTE: Only the Move and Kill functions have any effect on an agent when it is hidden.

vuAgentShow() – Shows an agent that was previously hidden.

Calling Parameters

NONE

Calling Parameter Values

NONE

Return

True on success

Example

vuAgentShow()

vuAgentWait(wSeconds) – Pauses agent action for the specified number of seconds

Calling Parameters

wSeconds Long

Calling Parameter Values

wSeconds = The number of seconds for the agent to wait prior to continuing its actions

Returns

True on success

Example

vuAgentWait(3)

Issuing this function puts the agent interface into a wait state for the specified number of seconds.

NOTE: This function will NOT cause the agent to pause between commands that have already been received. When this function is issued, the wait state is for the receipt of additional commands.

vuAgentStop() – Stops the agent when performing a “looping” action. When a looping action is being performed (see the actions for each specific agent’s list of looping actions), the agent will not perform any other actions (pending or not) until a “Stop” is received.

Calling Parameters

NONE

Calling Parameters Values

NONE

Returns

True on success

Example

vuAgentStop()

vuAgent Version 1.10



vuAgentAbsMove(x, y) – Moves the agent (hidden or visible) to the specified x,y position (relative to the upper left corner of the screen)

Calling Parameters

X Long

Y Long

Calling Parameter Values

X = The position (in pixels) to move across (relative to the upper left corner of the screen)

Y = The position (in pixels) to move down (relative to the upper left corner of the screen)

Returns

True on success

Example

X = 512

Y = 384

vuAgentAbsMove(X,Y)

This will move the agent to the center of a screen that is set to a density of 1024 X 768

NOTE: Since this function performs no boundary checks it is possible to move the agent completely off the screen.

vuAgentRelMove(x, y) – Moves the agent (hidden or visible) to the specified x,y position (relative to the upper left corner of the Window)

Calling Parameters

X Long

Y Long

Calling Parameter Values

X = The position (in pixels) to move across (relative to the upper left corner of the Window)

Y = The position (in pixels) to move down (relative to the upper left corner of the Window)

Returns

True on success

Example

X = 100

Y = 100

vuAgentRelMove(X,Y)

This will move the agent to 100 by 100 pixels within the Window from which the function was issued

NOTE: Since this function performs no boundary checks, it is possible to move the agent completely off the Window.

vuAgentSpeak(cSentence) – Speaks the sentence specified in cSentence. The text displays in a balloon and the voice is heard through the sound card.

Calling Parameters

cSentence CString(256) !Maximum size limit is 32,768

Calling Parameter Values

cSentence = Any sentence or words you would like the agent to speak

Returns

True on success

Example

cSentence = 'I can talk, and use speech tags to modify my voice.'

vuAgentSpeak(cSentence)

NOTE: You can use “Speech Tags” (defined later in this document) to modify the agent’s voice and/or speaking cadence.

vuAgent Version 1.10



vuAgentSpeakMono(cSentence) – Speaks the sentence specified in cSentence using a monotone voice. The text displays in a balloon and the voice is heard through the sound card.

Calling Parameters

cSentence CString(256) !Maximum size limit is 32,768

Calling Parameter Values

cSentence = Any sentence or words you would like the agent to speak

Returns

True on success

Example

cSentence = 'I can talk, and use speech tags to modify my voice.'
vuAgentSpeakMono(cSentence)

NOTE: This function uses “Speech Tags” (defined later in this document) to modify the agent’s voice.

vuAgentWhisper(cSentence) – Speaks the sentence specified in cSentence using a whispered voice. The text displays in a balloon and the voice is heard through the sound card.

Calling Parameters

cSentence CString(256) !Maximum size limit is 32,768

Calling Parameter Values

cSentence = Any sentence or words you would like the agent to speak

Returns

True on success

Example

cSentence = 'I can talk, and use speech tags to modify my voice.'
vuAgentWhisper(cSentence)

NOTE: This function uses “Speech Tags” (defined later in this document) to modify the agent’s voice.

vuAgentThink(cSentence) – Shows the text in a bubbled balloon as if the agent is thinking the text.

Calling Parameters

cSentence CString(256) !Maximum size limit is 32,768

Calling Parameter Values

cSentence = Any sentence or words you would like the agent to think

Returns

True on success

Example

cSentence = 'I can think, it just hurts my head'
vuAgentThink(cSentence)

vuAgent Version 1.10



vuAgentPlay(cCommand) – “Plays” any valid command for the agent (gesture, congratulate, etc.)

Calling Parameters

cCommand CString(64)

Calling Parameter Values

cCommand = Any valid command that the agent is capable of performing.

Returns

True on success

Example

cCommand = 'DoMagic2!'

vuAgentPlay(cCommand)

The agent will do a magic trick (Merlin waves his wand)

NOTE: Commands that expect an exit branch (see individual commands for a list of these actions) will pause when no other command is received. To avoid this problem and prevent the programmer from having to issue two (2) commands, you can end your command action with an exclamation point. Doing so actually issues two (2) commands to the agent. The one you sent plus a “RestPose” (return to normal position). Try commands both with and without an exclamation point to see the difference.

vuAgentSequence(cCommands) - “Plays” multiple actions and commands with a single function. Your commands and actions must be separated with an asterisk in order for vuAgentSequence() to distinguish between them.

Calling Parameters

cCommand CString(32768)

Calling Parameter Values

cCommand = Any series of valid commands or actions

Returns

True on success

Example

cCommand = '*Alert *DontRecognize *Speak Did you say something? *Restpose *Speak
Sometimes I am hard of hearing.'

vuAgentSequence(cCommand)

The agent will play the entire sequence of commands

NOTE: vuAgentSequence will accept any valid command or action. In addition, the action word “Play” is optional. For example, you can enter either *Play Alert or just enter *Alert (the PLAY action command is assumed). As in vuAgentPlay(), you can terminate any valid command with an exclamation point to return the agent to a rest pose.

Speech Tags

The Microsoft Agent services support modifying speech output through special tags inserted in the speech text string. These tags help you change the characteristics of the output expression of the character. Speech output tags use the following rules of syntax:

- All tags begin and end with a backslash character (\).
- The single backslash character is not enabled within a tag. To include a backslash character in a text parameter of a tag, use a double backslash (\\).
- Tags are NOT case-sensitive. For example, \pit\ is the same as \PIT\.
- Tags are whitespace-dependent. For example, \Rst\ is NOT the same as \ Rst \.
- Unless otherwise specified or modified by another tag, the speech output retains the characteristic set by the tag within the text specified in a single Speak method.
- Speech output is automatically reset through the user-defined parameters after a Speak method is completed.
- Some tags include quoted strings. For Example:
 Speak This is \map="Spoken text" = "Balloon text"
 (The agent will say "Spoken text" but will display "Balloon text")

The following tags are supported:

Chr - Sets the character of the voice.

\Chr=string\

| Part | Description |
|------------|---|
| string | A string specifying the character of the voice. |
| "Normal" | (Default) A normal tone of voice. |
| "Monotone" | A monotone voice. |
| "Whisper" | A whispered voice. |

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Ctx - Sets the context of the output text.

\Ctx=string\

| Part | Description |
|-----------|---|
| string | A string specifying the context of the text that follows, which determines how symbols or abbreviations are spoken. |
| "Address" | Addresses and/or phone numbers. |
| "E-mail" | Electronic mail. |
| "Unknown" | (Default) Context is unknown. |

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Emp - Emphasizes the next word spoken (This tag must immediately precede the word).

\Emp\

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Lst - Repeats last spoken statement for the character.

\Lst\

This tag enables an agent to repeat its last spoken statement. This tag must appear by itself in the Speak method; no other text or parameters can be included. When the spoken text is repeated, any other tags included in the original text are repeated, except for bookmarks. Any .WAV and .LWV files included in the text are also repeated.

vuAgent Version 1.10



Map - Maps spoken text to text displayed in the work balloon.

`\Map="spokentext"="balloontext"`

| Part | Description |
|-------------|---|
| spokentext | A string specifying the text for spoken output. |
| balloontext | A string specifying the text for word balloon output. |

This tag enables you to use different spoken text than that displayed in the word balloon.

Mrk - Defines a bookmark in the spoken text.

`\Mrk=number`

| Part | Description |
|--------|--|
| number | A Long integer value that identifies the bookmark. |

When the server processes a bookmark, it generates a bookmark event. You must specify a number greater than zero (0) and not equal to 2147483647 or 2147483646.

Pau - Pauses speech for the specified number of milliseconds.

`\Pau=number`

| Part | Description |
|--------|--------------------------------------|
| number | The number of milliseconds to pause. |

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Pit - Sets the baseline pitch of the output to the specified value in hertz.

`\Pit=number`

| Part | Description |
|--------|---------------------|
| number | The pitch in hertz. |

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Rst - Resets all tags to the default settings.

`\Rst`

Spd - Sets the baseline average talking speed of the speech output.

`\Spd=number`

| Part | Description |
|--------|--|
| number | Baseline average talking speed, in words per minute. |

This tag is supported only for TTS-generated output. The range of values for the parameter may vary depending on the installed TTS engine.

Vol - Sets the baseline speaking volume of the speech output.

`\Vol=number`

| Part | Description |
|--------|---|
| number | Baseline speaking volume: 0 is silence and 65535 is maximum volume. |

The volume setting affects both left and right channels. You cannot set the volume of each channel separately. This tag is supported only for TTS-generated output.

Microsoft Agent enables you to build in some variation for a character. For example, Microsoft Agent will randomly select one of the following statements when processing this text as part of the Speak method:

"I can say this.|I can say that.|I can say something else."

See the vuAgent demonstration program for a complete overview of all available commands and actions!

NOTE: *The Demo can be found under C:\C55\Examples\Valutilities\vuAgent*