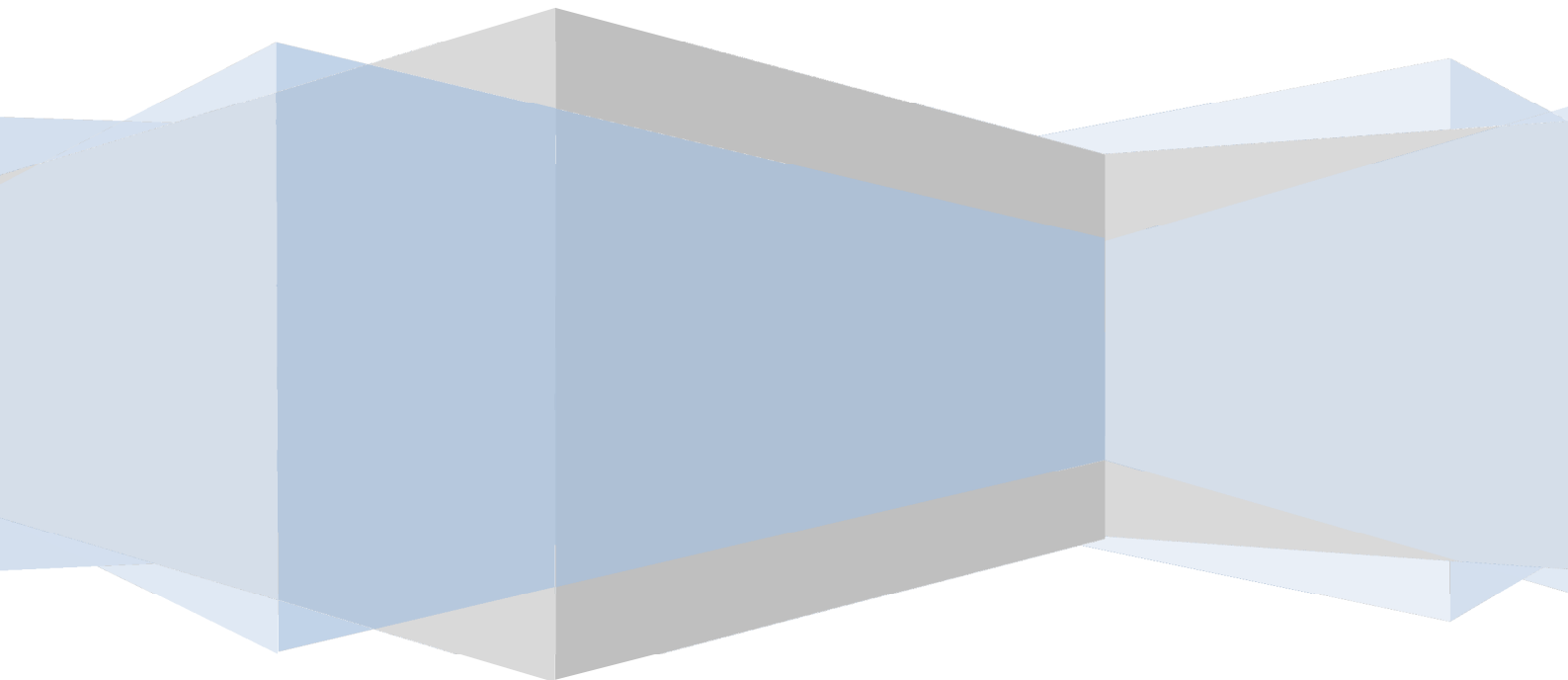




**Guile 3D Studio**  
**Artificial Intelligence SDK**  
**AIML and Script Tags**  
Version 1.0.20101026



## Sumário

1. Introduction.....	4
What is AIML? .....	4
Using AIML files with Denise .....	4
Using custom English AIML files with Denise.....	5
Using Guile's 3D Studio custom AIML tags inside AIML categories .....	6
2. Custom Tags and Scripts .....	8
1) AIML .....	8
2) Webbrowser.....	9
3) Search.....	13
4) AgentSpeakinChat.....	17
5) SysVolume .....	18
6) Run .....	19
7) Module .....	20
8) EmptyRecycle .....	22
9) MyIP .....	23
10) Hostname.....	24
11) DisplayAdapter.....	25
12) Download .....	26
13) Upload.....	27
14) ASK_WIKI.....	28
15) ASK_DIC.....	29
16) AgentShow .....	30
17) AgentResolution.....	31
18) AgentStayOnTop .....	32
19) AgentQuality .....	33
20) Weather .....	34
21) Humor .....	36
22) Date .....	37
23) Time.....	38
24) Play.....	39
25) Expression .....	40
26) Translate.....	41
27) Shutdow .....	43

---

28) Restart.....	44
29) TurnOn e TurnOff.....	45
30) Close.....	46
31) Pascal.....	47
32) Tracker.....	48
33) Bible.....	49
34) Calendar (Agenda and Scheduler).....	50
35) Contact.....	61
36) Volume (SAPI5 TTS Tag).....	62
37) Rate (SAPI5 TTS Tag).....	63
38) Spell (SAPI5 TTS Tag).....	65
39) Pitch (SAPI5 TTS Tag).....	66
40) Silence (SAPI5 TTS Tag).....	67
41) Emph (SAPI5 TTS Tag).....	68
42) Pron (SAPI5 TTS Tag).....	69
43) PartOfSp (SAPI5 TTS Tag).....	70
44) Context (SAPI5 TTS Tag).....	71
45) Memory.....	72
46) MediaPlayer.....	74
47) ChDir.....	78
48) ChatDialog.....	84
49)ChatInput.....	85
50) Buy.....	86
51) CDAudio.....	87
52)SoundCard.....	88
53)Conjugate.....	89
54) Battery.....	90
55)Keyboard.....	91
56) Skype.....	94
57) Feeds.....	95
58) EMail.....	96
59) OpenResult.....	98
60) Chat.....	99
61) ChatClear.....	100

## 1. Introduction

Guile 3D Artificial Engine uses several algorithms for its Artificial Intelligence. One of these layers uses AIML programming language.

This SDK gives users a fast and easy way to create, edit and modify Virtual Assistant Denise knowledge, as well as translate the given editable AIML categories to other languages.

**Important:** You should be familiar with AIML language and basic Windows files and folder operations for fully understand and use this SDK.

### What is AIML?

AIML stands for Artificial Intelligence Markup Language. It's an XML dialect for creating natural language applications, developed by Dr. Richard Wallace and a worldwide free software community. Applications using this technology already won the annual Loebner Prize Contest for Most Human Computer three times, and was also the Chatterbox Challenge Champion in 2004.

Guile 3D Studio has expanded the already great power of AIML with many custom tags and scripts. These tags and scripts allow us to make several custom computer and internet operations from within AIML categories.

Note: To create or modify AIML categories you can use Guile 3D Studio Artificial Intelligence Brain Editor (Denise Platinum and Business Only). Click here for instructions about using this editor.

You can find [here](#) more details about AIML language and its syntax.

A great AIML manual is the book "Be Your Own Botmaster" from Dr. Richard Wallace can be purchased [here](#).

### Using AIML files with Denise

When you start Virtual Assistant Denise, she loads Guile's 3D AIML files from `\Guile3D\bin\Database\AIML\EN\` folder. These AIML files cannot be modified since they are cryptographed (.aiml-x) for security and compatibility reasons.

Also, Denise will load any User custom AIML files found in `\Guile3D\bin\Database\AIML\Users\{user name}\` folder.

The editable AIML files that you can edit or use as reference to create new ones are located in \Guile3D\bin\SDK\AIML Source\EN\ folder.

You don't need to use these files to create new ones, but they are a good English reference to understand how to integrate Guile's 3D tags inside AIML categories and translation for other languages.

Note: Not every AIML files that go with Denise are available in the editable files, just the ones that uses Guile's 3D custom AIML tags and are responsible for executing actions, like searching web, e-mail etc..

**Important:** Before using any of these editable AIML files found in the \Guile3D\bin\SDK\AIML Source\EN\, select all files and copy them to a different folder in your computer. Please do not work directly on the files in this original folder, since Guile 3D will often overwrite these files with new updated ones, so this updates could overwrite your changes.

So, below are the initial steps to start editing or creating new AIML files for Denise:

1. Go to \Guile3D\bin\SDK\AIML Source\EN\ folder
2. Select all files and make a copy of them to a different folder in your computer.
3. Open the files with any AIML or text editor, or with Guile's 3D AIML Brain Editor. For instructions about using Guile's 3D AIML Brain Editor, please read its Manual [here](#).
4. After editing or creating new English AIML categories, save and cryptograph them with the AIML Brain Editor in Guile3D\bin\Database\AIML\Users\{(user name)\ folder
5. Close and reopen Denise's system. The system will now load your AIML's and also load the original English AIML files that comes with Denise

## Using custom English AIML files with Denise

After editing or creating new English AIML categories, save and cryptograph them with the AIML Brain Editor in Guile3D\bin\Database\AIML\Users\{(user name)\ folder

For English language, Denise will also always load the original cryptographed files located in \Guile3D\bin\Database\AIML\EN\ folder.

**Important:** Note that due to AIML characteristics, sometimes your custom AIML categories can be "shadowed" by Guile's 3D original English AIML categories and if this happen, your category will not work. This will happen for example, if there is an identical category both in Guile's 3D original AIML files and in your custom AIML set. We have plans to do a workaround for this issue in a future upgrade.

One thing you can do if you don't want to load Guile's 3D original English AIML sets, is deleting all .aiml files in \Guile3D\bin\Database\AIML\EN\ folder, and save your own custom AIML English sets in this folder. Note that you will have to always use cryptographed AIML files in this folder (.aiml-x), done with Guile's 3D AIML Brain Editor Application.

**Important:** If you do so, please always make a backup of your original files in a different folder, because after every system update, your files will be overwritten by updated Guile's 3D AIML files, and you will have to again delete Guile's AIML files and save your custom ones in the same folder. (\Guile3D\bin\Database\AIML\EN\)

## Using Guile's 3D Studio custom AIML tags inside AIML categories

Guile 3D custom tags should be inserted and used inside the AIML "Template" section of an AIML category, and within the < (less than) > (greater than) signs.

**Example 1** – Making Denise do a exclamation face expressions:

```
<category>
<pattern>NICE TO MEET YOU</pattern>
<template>Thanks. Nice to meet you too. <Expression="Exclamation"> </template>
</category>
```

In this example, if user type "Nice to meet you" in Denise's chat window, she will answer "Thanks. Nice to meet you too", and then makes an exclamation face expression.

**Example 2** – Searching web using MIT START on-line database:

```
<category>
<pattern>WHO IS THE PRESIDENT OF UNITED STATES</pattern>
<template><Search="WHO IS THE PRESIDENT OF UNITED STATES" Types="Start">
Searching...</template>
</category>
```

Or using the "\*" wild card AIML tag:

```
<category>
<pattern>WHO IS THE *</pattern>
<template><Search="WHO IS THE <star />" Types="Start"> Searching...</template>
</category>
```

Custom tags can be placed in any position inside the AIML “Template” section, in the beginning, middle or at the end. For searching operations, is always better to place tags before any text that Denise should speak while doing the tag action, this way you will avoid any delay to start the action that could occur from the Text to Speech engine.

## 2. Custom Tags and Scripts

### 1) AIML

The "AIML" tag creates a new AIML category and saves it in file "User.aiml" located in folder "\Guile3D\bin\Database\AIML\Users\"(user name)\"

This tag is very useful to create new AIML categories in real-time during a conversation with the Virtual Assistant Denise.

Important: New AIML categories added using this tag will only work after closing and re-starting Virtual Assistant Denise system.

#### 1.1) Syntax

```
<AIML="Add" pattern="AIML Pattern Value" template="AIML Template Value" that="AIML That Value" topic=" AIML Topic Value ">
```

Parameters values:

Add: Add

AIML Pattern Value: The Question text

AIML Template Value: The Answer Text

AIML That Value: AIML That Text. Can be left empty.

AIML Topic Value: AIML Topic Text. Can be left empty.

#### 1.2) Examples

```
<AIML="Add" pattern="Who is Guile Lindroth" template="Guile is the founder of Guile 3D Studio" that="" topic="Guile">
```

This example creates the following AIM category at the end of the file "User.aiml" located in \Guile3D\bin\Database\AIML\Users\"(user name)\ folder:

```
<category>
```

```
<pattern>WHO IS GUILLE LINDROTH</pattern>
```

```
<template>Guile is the founder of Guile 3D Studio
```

```
<think><set name="topic">Guile</set></think> </template>
```

```
</category>
```

## 2) Webbrowser

Guile 3D Studio has built its own Web Browser application to surf the web. It has all the most important functions from well known web browsers like Windows Internet Explorer, Firefox or Chrome.

On having its own web browser application, Guile 3D Studio has more power and flexibility to adapt and integrate advanced web page navigation features with Voice Recognition to Virtual Assistant Denise.

### 2.1) Syntax

```
<webbrowser="Parameter Value">
```

Parameter Values are:

- *Web address: Any valid internet address.*
- DOWN: Down Key
- PGDOWN: Page Down Key
- UP: Up Key
- PGUP: Page Up Key
- NEXTTAB: Go to Next Opened Tab
- PREVIEWSTAB: Go back to Previous Opened tab
- BACK: Go back to last web page viewed
- FORWARD: Go to next web page

Important: The above navigation values works only with Guile's 3D Web Browser. To make a similar behaviour on browsers like Internet Explorer, or in other applications like word processor, use the Guile 3D custom tag "Keyboard"

### 2.2) Examples

```
<webbrowser="http://www.guile3d.com/">
```

Open the web browser showing Guile 3D Studio web site.

```
<webbrowser="http://en.wikipedia.org/wiki/Main_Page">
```

Open Wikipedia home page on Guile's 3D Web Browser.

<WebBrowser="DOWN">

Simulates the Down Key, making the showing web page rolls down one line

<WebBrowser="PGDOWN">

Simulates the Page Down Key, making the showing web page rolls down one page

<WebBrowser="UP">

Simulates the Up Key, making the showing web page rolls up one line

<WebBrowser="PGUP">

Simulates the Page Up Key, making the showing web page rolls Up one page

<WebBrowser="NEXTTAB">

Go to and show the next opened Tab on Guile's 3D Web Browser

<WebBrowser="PREVIEWSTAB">

Go to and show the previous opened Tab on Guile's 3D Web Browser

<WebBrowser="BACK">

Simulates the Back Key, opening the previous web page

<WebBrowser="FORWARD">

Simulates the Forward Key, opening the next web page

### 2.3) Examples using AIML tags within <webbrowser>

```
<webbrowser="http://www.google.com.br/search?q=<star />">
```

Searches the web for the value of the AIML wildcard tag <star /> using Google.

A complete AIML category examples:

```
<category>
```

```
<pattern>GOOGLE *</pattern>
```

```
<template><webbrowser="http://www.google.com.br/search?q=<star />"></template>
```

```
</category>
```

Using the example above, when user types “Google Virtual Assistants”, Denise will search Google for “Virtual Assistants” and show results in Guile’s Web Browser.

Note: You can get the same above results using Guile 3D tag <search>:

```
<category>
```

```
<pattern>GOOGLE *</pattern>
```

```
<template><Search="<star />" Types="Web"></template>
```

```
</category>
```

There is also an option to show searching results using your computer’s default Web Browser:

```
<category>
```

```
<pattern>GOOGLE *</pattern>
```

```
<template><Run=" http://www.google.com.br/search?q=<star />"></template>
```

```
</category>
```

The only difference from these three examples are that the first one will open results in the Web Browser Window, the second opens results in Denise’s Chat lateral Window, and last one opens results using computer’s default Web Browser.

### 2.4) Examples using two or more custom tags in the same AIML category

```

<category>
<pattern>GOOGLE *</pattern>
<template>
<webbrowser="http://www.google.com.br/search?q=<star />">
<WebBrowser="PGDOWN">
</template>
</category>

```

Using the above example, if users type "Google Apple Pie Recipes", Denise will search the web using Google, than open Guile's 3D Web Browser with the search results, and makes the page go down.

```

<category>
<pattern>GO TO GUILLE 3D WEB PAGE</pattern>
<template>
<webbrowser="http://www.guile3d">
<WebBrowser="PGDOWN">
<WebBrowser="DOWN">
<WebBrowser="DOWN">
</template>
</category>

```

Using the above example, if users type "Go to Guile 3D Web Page", Denise will open Guile's 3D Web site, and then makes the page go down one page, and also down two lines.

## 3) Search

Use the “Search” tag to get information from several web locations, databases and search engines, as well as from local computer files. It’s possible to use combined tags in the same AIML category.

### 3.1-Syntax

<Search=“Search Query” Types=“Search Type” >

where:

Search Query: The subject of the search

Search Type: The type of search, and can be of types:

- Web: Search the internet using the default search engine set at the SETUP Module.
- ImageLocal: Search your computer for local image files
- ImageWeb: Search the Internet for images.
- VideoLocal: Search your computer for local video files.
- VideoWeb: Search the internet for video files.
- Maps: Search for locations and routes in maps.

The map Search Query should be a country, state, city, street, avenue or any point of interest. Optionally, we can use the “From” parameter to get an origin point. This will return directions of how to get from one point to another. The “From” parameter can receive the value %HOME% to get your home address automatically from Denise’s database.

- Cinema: This tag returns theatres hours from the web. You can also use the “From” parameter to get theatres hours from a specific location. The “From” parameter can receive the value %HOME% to get and use your home address.
- News: Search news related to the Search Query word or phrase.
- Books: Search Books related to the Search Query word or phrase. You can use Books name, Books author or Books editor for the Search Query.
- Wiki: Search the Wikipedia.
- Start: Search M.I.T START Project database
- Concept: Search M.I.T ConceptNet (Open Mind Common Sense) Project database

- Dic: Get word definitions from a Dictionary.
- Files: Search your computer for local files. (Documents, songs, images etc.)
- Bible: Search The Holy Bible.

### 3.2- Examples

<Search="Pascal Programming" Types="Web">

<Search="Dolphins" Types="ImageWeb">

<Search="Androids" Types="Books">

<Search=" Earth Heating " Types="News">

<Search="Artificial Intelligence" Types="Wiki">

<Search="Artificial Intelligence" Types="Start">

<Search="Love" Types="Concept">

<Search="1500 Lombard Street, San Francisco-CA, United States" Types="Maps" >

<Search="1500 Lombard Street, San Francisco-CA, United States" Types="Maps" From=" 940 Union Street, San Francisco-CA, United States">

<Search="1500 Lombard Street, San Francisco-CA" Types="Maps" From="%HOME%">

<Search="Ferrari" Types="Videos">

<Search="Salt" Types="Cinema" From="%HOME%">

<Search="Tron" Types="Cinema" From=" San Francisco-CA">

<Search=" Tron" Types="Cinema">

<Search="Birthday" Types="ImageLocal">

<Search="Lord is my Shepherd" Types="Bible">

### 3.2.1- Examples within AIML

```
<category>
<pattern>SEARCH Denise from Guile 3D</pattern>
<template><Search=" Denise from Guile 3D " Types="Web"> Searching...
</template>
```

Or a more intelligent way of doing the same query:

```
<category>
<pattern>SEARCH *</pattern>
<template><Search="<star />" Types="Web"> Searching...
</template>
</category>
```

### 3.2.2- Combined Tags Examples

```
<Search="Planets" Types="Web+ImageWeb+ImageLocal+Wiki">
```

Will search for web info about Planets, as well as local images and in Wikipedia.

```
<Search="Artificial Intelligence" Types="Books+Start+Wiki">
```

Will search for books about Artificial Intelligence in the web, in the START database and in Wikipedia.

Any combination of Search Types is valid.

### 3.2.3 - Searching web using Web Sites URL

You can also retrieve web results directly using a URL address inside both the <Webbrowser> and <Run> Guile 3D tags. Note that every website has its own URL format for querying data.

<webbrowser="http://www.bing.com/search?q=Recipes">

Searches Microsoft Bing for "recipes" and shows the result in Guile 3D Web Browser

<Run=" http://www.bing.com/search?q=Recipes ">

Searches Microsoft Bing for "recipes" and shows the result in the computer's default Web Browser

## 4) AgentSpeakinChat

This tag can be used to send any text to the Assistant Chat Window, as the user had typed it manually.

This tag is very useful for integrating third part applications that needs to send text, commands and messages to the Assistant Interface, making the communication link between Guile 3D Assistant application and external applications.

For more details on using this tag and third part integration, please contact Guile 3D at [support@guile3d.com](mailto:support@guile3d.com).

### 4.1) Syntax

```
<AgentSpeakinChat="Parameter">
```

### 4.2) Examples

```
<AgentSpeakinChat="Hello Denise">
```

```
<AgentSpeakinChat="Open Calculator">
```

## 5) SysVolume

The tag "SysVolume" is used to change the master volume of the operating system. The operating system has many audio channels, as the CD volume, Wave Sound, Line In and Master Volume. The Master Volume is the volume that influences all other computer channels.

This tag should not be confused with the "Volume" tag which is used to control the Assistant Speech Volume, nor with the tag `<MediaPlayer="Play" volume="80" >` that control de volume of music or Web Radio being played.

Using "SysVolume", all computer sounds will be affected, including the Assistant Speech volume.

The tag "SysVolume" uses values from 000 to 100.

### 5.1) Syntax

You can use SysVolume in two ways: Volume Value or incremental.

`<SysVolume=0 to 100 value>`

or

`<SysVolume=Value to increment or decrease volume>`

### 5.2) Examples

`<SysVolume=0>`

No sound at all

`<SysVolume=100>`

Turn Volume Up

`<SysVolume=50>`

Middle Volume

`<SysVolume=90>`

High Volume

Incremental Use:

`<SysVolume=+10>` or `<SysVolume=-10>`.

Turn the Volume Up or Down in 10 percent.

## 6) Run

This is a very powerful tag that calls Windows API "ShellExecute" to open or run Windows applications, web pages or a windows command.

### 6.1) Syntax

```
<Run="Command">
```

```
<Run="Command" Param="Parameters">
```

```
<Run="STARTUP" Param="Program Shortcut to Run">
```

"Command" may be the name of an executable file, or the name of any file. In the case of an executable, the application will be started. If a file, the operating system will find the application responsible for this type of file and will try to open it.

"Param" is used to inform the application to run more details about how to be executed.

\* "Command" can receive the "StartMenu" word. In this case, "Parameter" becomes a keyword. The system will try to find within Windows directories CSIDL\_STARTMENU CSIDL\_COMMON\_STARTMENU the shortcut (\*.lnk) that contains the keyword.

### 6.2) Examples

```
<Run="http://www.guile3d.com /">  
Open web page using default web browser
```

```
<Run="calc.exe">  
Open "calc.exe", Windows Calculator.
```

```
<Run="calc">  
Open "calc.exe", Windows Calculator. In some cases, extensions can be ignored.
```

```
<Run="c:\temp.txt">  
Open file temp.txt with default Text Editor.
```

```
<Run="Notepad" Param="c:\temp.txt">  
Opens file temp.txt using Notepad
```

```
<Run="StartMenu" Param="Photoshop">  
Tries to find a system shortcut for opening Adobe Photoshop.
```

## 7) Module

Similar to the “Run” tag that opens Windows applications, “Module” tag opens a proprietary Guile 3D Studio module or application.

### 7.1) Syntax

```
<Module="DLL file" Function="Function Name" Param="Parameters">
```

Module: Name of Guile 3D Studio Module to be executed. It is the name of .DLL file found in \Guile3D\bin\Modules\ folder.

Function: Function to be executed.

IMPORTANT: Function name is case sensitive

Param: Parameters to be passed to the function. It varies from function to function.

### 7.2) Examples

```
<Module="TextStudio.dll" Function="Open" Param="Some Initial Text">
```

Opens Guile 3D Studio Text Studio Module

```
<Module="Chat" Function="ChatOpenMenu">
```

Opens Assistant Chat Windows

```
<Module="Chat" Function="ChatCloseMenu">
```

Close Assistant Chat Windows

```
<Module="Agent" Function="OpenDebug">
```

Opens Guile 3D Studio Debug Module

```
<Module="EMAIL" Function="Open" Parameters="NOW">
```

Check e-mail accounts for new e-mails. If so, Virtual Assistant speaks number of new e-mails

```
<Module="EMAIL" Function="Open" Parameters="WebMail">
```

Opens Guile’s 3D Web Browser with default’s e-mail account Webmail page.

```
<Module="Kernel" Function="OpenSetup">
```

Opens Guile 3D Studio Setup Module

<Module="MediaPlayer" Function="ListRadios">  
List Web Radio Stations

<Module="MediaPlayer" Function="ListTVs">  
List Web TV Stations

<Module="AIMLEditor", Function="Open">  
Opens Guile 3D Studio Artificial Intelligence Brain Editor

## 8) EmptyRecycle

Permanently deletes all files from the Windows Recycle Bin.

Important: This action will delete all files from Windows Trash, and this operation cannot be reversed.

### 8.1) Syntax

```
<EmptyRecycle>
```

### 8.2) Example within AIML

```
<category>
```

```
<pattern>CLEAR RECYCLE BIN</pattern>
```

```
<template><EmptyRecycle> All files were deleted from the Recycle Bin.</template>
```

```
</category>
```

## 9) MyIP

Returns Computer IP address.

### 9.1) Syntax

<MyIP>

## 10) Hostname

Returns Computer Host name.

### *10.1) Syntax*

<hostname>

## 11) DisplayAdapter

Returns information about Computer's Display adapter.

### 11.1) Syntax

<DisplayAdapter>

## 12) Download

The “Download” tag download files from the Internet. Files can be of any type. (Songs, videos, documents, images etc..)

### 12.1) Syntax

<Download="File Web address">

Parameter value:

File Web address: any web address (URL) that has a valid link for a downloadable file.

<Download="RESUME">

Opens the Download Chat lateral window. If there is some download stopped, this also resumes all downloads.

### 12.2) Examples

<Download="http://www.guile3d.com.br/denise/images/area\_grande.jpg ">

Starts downloading file “area\_grande.jpg”

<Download="RESUME">

Opens the Download Chat lateral window and resumes stopped downloads.

## 13) Upload

The “UPLOAD” tag sends files to Picasa and YouTube. Files must be images, pictures or videos in any format.

### 13.1) Syntax

Sending videos to YouTube and images to Picasa

```
<Upload="Upload Type" Files="Files list" Album="Album name" youtube="yes">
```

Sending videos and images to Picasa

```
<Upload="Upload Type" Files="Files list" Album="Album name" youtube="no">
```

Parameter values:

- Upload Type: IMAGES (valid both for images and videos)
- Files list: Computer files to be sent with their hard disk folder information.
- Album name: Picasa album name to place images or videos, or YouTube video title.

### 13.2) Examples

```
<Upload="Images" Files="C:\Planets.png" Album="Universe" youtube="no">
```

Sends image “Planets.png” to Picasa album named “Universe”

```
<Upload="Images" Files="C:\Guile\Pictures\Earth.png", "d:\Jupiter.wmv" Album="Planets"
youtube="yes">
```

Sends image “Earth.png” to Picasa album named “Planets”, and also send video “Jupiter.wmv” to YouTube and place the title “Planets” for this video.

## 14) ASK\_WIKI

This tag execute a Wikipedia search. Works the same way as using the “Search” with the “Wiki” parameter.

### 14.1) Syntax

```
<Ask_Wiki="Query text">
```

### 14.2) Example

```
<Ask_Wiki="Planets">
```

Is the same as using:

```
<Search=" Planets" Types="Wiki">
```

## 15) ASK\_DIC

This tag searches a dictionary for the meaning of a word.

### *15.1) Syntax*

```
<Ask_Dic="Word">
```

### *15.2) Example*

```
<Ask_Dic="Mythology">
```

Searches a dictionary for the meaning of the word "Mythology".

## 16) AgentShow

Controls the Virtual Assistant display mode in the computer's desktop screen.

### 16.1) Syntax

<AgentShow=wsMaximized>

Maximizes Assistant

<AgentShow=wsNormal>

Show Assistant in normal size

<AgentShow=wsMinimized>

Minimizes Assistant

## 17) AgentResolution

Controls the Virtual Assistant resolution mode in the computer's desktop screen.

### 17.1) Syntax

<AGENTRESOLUTION=Assistant Screen Resolution>

Assistant Screen Resolution values are:

0 for Auto

1 for 800x600

2 for 1024x768

3 for 1280x1024

### 17.2) Example

<AgentResolution=2>

Virtual Assistant will have a 1024 x 768 screen size.

## 18) AgentStayOnTop

Controls the Virtual Assistant position relative to other open Windows applications.

### 18.1) Syntax

<AGENTSTAYONTOP=TRUE>

Keeps Assistant always on top of other open desktop applications

<AGENTSTAYONTOP=FALSE>

Turns off Assistant always on top of other desktop applications

### 18.2) Example

<AGENTSTAYONTOP=TRUE>

Virtual Assistant will always show on top of other windows applications.

## 19) AgentQuality

Temporarily changes the Virtual Assistant graphic quality. Useful to improve Assistant performance in older and less powerful computers.

Note: To change Assistant's quality and keep this setting for the next system startup, use Denise's SETUP module.

### 19.1) Syntax

<AgentQuality=SuperHigh>

Temporarily changes Assistant's graphic quality to Super High. Note that this will work only in fast computers with dedicated Graphic Video Cards.

<AgentQuality=High>

Temporarily changes Assistant's graphic quality to High. (Recommended setting)

<AgentQuality=Low>

Temporarily changes Assistant's graphic quality to Low.

### 19.2) Example

<AgentQuality=High>

Temporarily changes Assistant's graphic quality to High.

## 20) Weather

Checks temperature and weather conditions for actual day and upcoming 4 days.

### 20.1) Syntax

```
<Weather="City Name" State="State" Day="Days" NextDays="Number of upcoming days">
```

- Weather: City name

- State: State name. This tag value can be left empty.

-Day: Upcoming day for the forecast. Valid values range from 1 to 4, where 1 is for the actual day, 2 is for tomorrow etc.


-NextDays: Weather Forecast for upcoming days. Valid values range from 1 to 4, where 1 is for the actual day forecast, 2 is for today and tomorrow forecast etc.


### 20.2) Examples


```
<Weather="San Francisco" State="CA" NextDays="4">
```


Will return:

This is the Weather Forecast for the upcoming 4 days:

 Mon 10/06 - Lower will be 11 and Highest will be 18 degrees.

 Tue 11/06 - Lower will be 18 and Highest will be 12 degrees.

 Wed 12/06 - Lower will be 17 and Highest will be 11 degrees.

 Thu 13/06 - Lower will be 09 and Highest will be 03 degrees.

```
<Weather=" San Francisco" State="CA" Day="4">
```

Will return:



Thu 13/06 - Lower will be 09 and Highest will be 03 degrees.

Weather Tag can also get user home City and State name from user's database info by using the "<get name=" " />" AIML tag:

```
<Weather="<get name="P_WEATHER_CITY" />" State="<get name="P_WEATHER_UF" />"  
NextDays="4">
```

Note that the parameters "P\_WEATHER\_CITY" and "P\_WEATHER\_UF" are actually database field names from the file "Brain.mdb" used to store user preferences.

## 21) Humor

The “Humor” tag makes the Virtual Assistant to speak random messages. These messages are stored in file “Humor\_EN.xml” located in “\Guile3D\bin\Database\Languages” folder of your computer.

### 21.1) Syntax

<Humor=Hello>

Returns salutations phrases like:

- Hello (your name). Good to see you!
- Hello (your name). How can I help you?

<Humor=WakeUp>

Returns wakeup phrases like:

- Good morning! It's time to wake up!
- Good morning! Let's wake up to enjoy this beautiful day!

## 22) Date

Makes Virtual Assistant speaks the current date.

### 22.1) *Syntax*

<Date>

Assistant speaks current Date.

## 23) Time

Makes Virtual Assistant speaks the current time.

### 23.1) *Syntax*

<Time>

Virtual Assistant speaks current time.

## 24) Play

Makes Virtual Assistant to play one of her pre built animations.

### 24.1) Syntax

```
<Play="Animation">
```

Valid values for the Animation parameter are:

```
<Play="Doubt">
```

```
<Play="Fear">
```

```
<Play="Indecisive">
```

```
<Play="Kiss">
```

```
<Play="Loving">
```

```
<Play="Sleeping">
```

```
<Play="Smile">
```

```
<Play="SmileHappy">
```

```
<Play="Upset">
```

```
<Play="Upset2">
```

```
<Play="VerySad">
```

### 24.2) Examples

```
<Play="Smile">
```

Makes Virtual Assistant to smile.

## 25) Expression

Changes Virtual Assistant facial expression.

Note. This expressions also works while the Assistant is talking. Always use <Expression=Blink> to make Assistant returns to its normal face expression.

### 25.1) Syntax

<Expression="Eyes Expression">

Valid Values for Eyes Expression parameter are:

<Expression="OneBrowUp">

<Expression="Exclamation">

<Expression="Sad">

<Expression=Blink>

<Expression="Angry">

### 25.2) Example

<Expression="Exclamation">

Virtual Assistant will make an exclamation face.

## 26) Translate

Tag for translating words or phrases from one language to another.

### 26.1) Syntax

```
<Translate="Text" From="Origin Language" To="Target Language">
```

If we don't know in what language the text to be translated is, we can use "AUTO" for the "From" parameter value.

```
<Translate="Text" From="AUTO" To="Target Language">
```

### 26.2) Available Languages for getting translations

"From" and "To" attributes can receive these Language values:

- Afrikaans
- Albanian
- Arabic
- Belarusian
- Bulgarian
- Catalan
- Chinese
- Croatian
- Czech
- Danish
- Dutch
- English
- Tagalog
- Finnish
- French
- Galician
- German
- Greek
- Hebrew
- Hindi
- Hungarian
- Icelandic

- Indonesian
- Irish
- Italian
- Japanese
- Korean
- Latvian
- Lithuanian
- Macedonian
- Malay
- Maltese
- Norwegian
- Polish
- Portuguese
- Romanian
- Russian
- Serbian
- Slovak
- Slovenian
- Spanish
- Swahili
- Swedish
- Thai
- Turkish
- Ukrainian
- Vietnamese
- Welsh
- Yiddish

### *26.3) Examples*

<Translate="Hello World" From="English" To="French">

<Translate="Hello World" From="AUTO" To="German">

## 27) Shutdown

Shuts the computer down.

### 27.1) Syntax

<Shutdown>

Shuts the computer down.

## 28) Restart

Re-starts the computer.

### *28.1) Syntax*

<Restart>

## 29) TurnOn e TurnOff

Turns electrical devices on and off. (Not available in this version)

This tag will be used for Home Automation.

### 29.1) Syntax

<TurnOn=Device Code>

<TurnOff=Device Code>

## 30) Close

The "CLOSE" tag can be used to close windows applications as well as to close Denise's system.

### 30.1) Sintaxe

<CLOSE>

Closes Denise's system.

<CLOSE="Application">

Closes some application

If there is more than one application open, let's say two Notepad documents with the Windows Title text as:

No Title – Notepad

Text.txt – Notepad

If we use:

<CLOSE="Notepad"> , both Notepad sessions are closed.

But if we use:

< CLOSE ="Text.txt"> just the Notepad session opened with text.txt will be closed.

### 30.2) Examples

<Close>

Close Denise's System

<Close="WordPad">

Close Windows WordPad.

## 31) Pascal

The PASCAL tag is target to advanced users only that have some knowledge of Pascal Programming Language.

With this tag, Guile 3D Studio has expanded even further AIML capabilities, giving it all the power of Delphi Object Pascal Programming Language.

We will soon make our Development Application SDK available with further documentation and sample codes and applications.

Note: You should be familiar with Object Pascal Programming Language to use this tag.

### 31.1) Syntax

A sample code using <Pascal> tag inside an AIML template category:

The sample below chooses a random number between 0 and 100 and returns the result to the Assistant to speak aloud:

```
<Pascal> uses Math, Classes, Graphics, Controls, Forms, DateUtils, SysUtils, GuileSDK; var
i:integer; var c:integer; var a:integer; var b:integer; var Text: string; Begin Text:=' '; for i := 1 to
1 do begin Text:=Text+IntToStr(RandomRange(0,100)); end; Result:=Text; end;</Pascal>
```

This example changes Virtual Assistant desktop position.

```
<pascal>uses GuileSDK; begin AgentPosition(0,530); end; </pascal>
```

The function AgentPosition(0,530) uses two parameters (x,y) where X is horizontal position from left to right, and Y is the vertical position from top to bottom of the computer screen.

This example makes the Assistant to count from 5 to 10.

```
<pascal>uses Classes, Graphics, Controls, Forms, GuileSDK; var I:integer; var Text: string; Begin
Text:=""; For I:=5 to 10 do begin Text:=Text+IntToStr(I)+' '; end; Result:=Text; end; </pascal>
```

This example converts a temperature from degrees Celsius to Fahrenheit .

```
<pascal>uses Classes, Graphics, Controls, Forms, Math, GuileSDK; var c:integer; var a:integer;
var b:integer; Begin a:=<star />; c:=(a * 1.8) + 32; Result := IntToStr(c); end; </pascal>
```

## 32) Tracker

Not available yet. This tag will be used to GPS tracking devices integration.

### 32.1) Syntax

```
< Tracker="Code" Show="True/False" Zoom="Zoom Level">
```

Tracker: Tracker device code

Show: display map with object position. Values are True or False

Zoom: Map zoom level of the object. Ranges from 0 to 100

### 32.2) Example

```
< Tracker="1" Show="True" Zoom="15">
```

## 33) Bible

Makes the Virtual Assistant to search and speak a Bible verse or passage.

### 33.1) Syntax

```
<Bible="Book" Chapter="Chapter" Verse="Verse">
```

```
<Bible="Book" Chapter="Chapter" FromVerse="Initial Verse" ToVerse="Final Verse" >
```

- Bible: Bible Book

- Chapter: Bible Chapter

- FromVerse and ToVerse: Verses to read. You may use numbers from 1 to 999 to read the whole book, even if it doesn't contain some verse number.

- Verse: Verse number to read. Use Verse=1 to read first Verse. Also, you may use parameters values like FromVerse=1 and ToVerse=5.

### 33.2) Examples

```
<Bible="Exodus" Chapter="2" Verse="1">
```

Returns: "Now a man of the house of Levi married a Levite woman."

```
<Bible="Exodus" Chapter="2" FromVerse="1" ToVerse="20">
```

Returns: And there went a man of the house of Levi, and took to wife a daughter of Levi. And the woman conceived, and bare a son: and when she saw him that he was a goodly child, she hid him three months. And when she could not longer hide him, she took for him an ark of bulrushes, and daubed it with slime and with pitch, and put the child therein; and she laid it in the flags by the river's brink....

## 34) Calendar (Agenda and Scheduler)

This is a very powerful tag to control user's alarms, appointments, tasks, birthdays, medicines hours, meetings, maintenances dates and courses dates.

### Checking personal appointments, alarms, meetings etc:

#### 34.1) Syntax

```
<Calendar="All" Action="Check" StartDate="From Date" EndDate="To Date">
```

Parameters Values:

Calendar:

- All: Will check for all personal schedule events due
- Birthday: Will check for Birthdays
- Alarm: Will check for Alarms
- Events: Will check for Agenda Events
- Courses: Will check for Courses
- Medicines: Will check for Medicines
- Maintenance: Will check for Maintenances
- Meeting: Will check for Meetings
- Tasks: Will check for Tasks
- Doctor: Will check for Appointments

Action: Check

StartDate and EndDate:

- Today
- Tomorrow
- ThisSunday or Sunday
- ThisMonday or Monday
- ThisTuesday or Tuesday
- ThisWednesday or Wednesday
- ThisThursday or Thursday
- ThisFriday or Friday
- ThisSaturday or Saturday
- NextSunday
- NextMonday
- NextTuesday

- NextWednesday
- NextThursday
- NextFriday
- NextSaturday

StartDate and EndDate can also accept direct date values in the Month/Day/Year format, like 07/22/10

### 34.2) Examples

```
<Calendar="All" Action="Check" StartDate="ThisFriday" EndDate="ThisFriday">
```

Check all my Agenda including Meetings, Tasks, Appointments etc. for Friday.

```
<Calendar="Meeting" Action="Check" StartDate="Today" EndDate="10/2/10">
```

Check my schedule meetings from today until October second.

## **Adding or deleting appointments, alarms, meetings etc.:**

### 34.3) Syntax

For Birthdays:

```
<Calendar="Birthday" Action="Add" Name="Subject" Date="Date" Speak="True"
Reminder="Reminder Time" Reminder_Type="Hours/Minutes">
```

Parameters Values:

Calendar: Birthday

Action: Add

Name: Birthday person's name

Date: Birthday's date

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Alarms

```
<Calendar="Alarm" Action="Add" Description=" Subject" Date="Date" Time="Time"  
Speak="True" AgentText="Assistant Speech Text" />
```

Parameters Values:

Calendar: Alarm

Action: Add

Description: Alarm Title

Date: Alarm Data

Time: Alarm Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

AgentText: Text to be spoken by the Virtual Assistant on alarm

For Appointments:

```
<Calendar="Doctor" Action="Add" DoctorsName="Doctor or Person Name" Date="Date"
Time="Time" Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Doctor

Action: Add

DoctorsName: Doctor or Person Name

Date: Appointment Date

Time: Appointments Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Courses:

```
<Calendar="Courses" Action="Add" CoursesName="Course Title" DateStart="Date"
TimeStart="Time" Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder
Type">
```

Parameters Values:

Calendar: Courses

Action: Add

CoursesName: Course Name

DateStart: Course Date

TimeStart: Course Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Events:

```
<Calendar="Events" Action="Add" What="Event Name" DateStart="Date" TimeStart="Time"
Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Events

Action: Add

What: Event Name

DateStart: Event Date

TimeStart: Event Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours

- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Maintenance:

```
<Calendar="Maintenance" Action="Add" Vehicle="Vehicle/House/Other name" Date="Date"  
Time="Time" Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Maintenance

Action: Add

Vehicle: Vehicle/House/Other item for maintenance name

Date: Event Date

Time: Event Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Medicines:

```
<Calendar=" Medicines" Action="Add" MedicinesName ="Medicine Name" Time= "Time to  
take Medicine" HowMany="How many to take" DateStart="Date to start" DateEnd="Date to  
stop" Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Medicines

Action: Add

MedicinesName: Medicine Name

DateStart: Date to start taking Medicine

DateEnd: Date to stop taking Medicine

Time: Time to take Medicine in HH:MM format (Example: 20:35)

HowMany: How many units to take

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Meetings:

```
<Calendar="Meeting" Action="Add" Subject="Meeting Title" Date="Date" TimeStart="Start Time" TimeEnd="End Time" Info="Meeting Detail Info" Speak="True" Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Meeting

Action: Add

Subject: Meeting Title or Subject

Date: Meeting Date

TimeStart: Meeting Start Time in HH:MM format (Example: 20:35)

TimeEnd: Meeting End Time in HH:MM format (Example: 20:35)

Info: Meeting Detail Info if necessary

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

For Tasks:

```
<Calendar="Tasks" Action="Add" What="Task Title" Date="Date" Time="Time" Speak="True"  
Reminder="Reminder Time" Reminder_Type="Reminder Type">
```

Parameters Values:

Calendar: Tasks

Action: Add

What: Task Title

Date: Task Date

Time: Task Time in HH:MM format (Example: 20:35)

Speak:

True: Virtual Assistant speaks reminder

False: Virtual Assistant doesn't speak reminder

Reminder: Time before event occurs to get a reminder message

Reminder\_Type:

- Minutes: Reminder time in minutes
- Hours: Reminder time in hours
- Days: Reminder time in days
- Weeks: Reminder time in weeks
- Months: Reminder time in months

### 34.4) Examples

```
<Calendar="Birthday" Action="Add" Name="Guile" Date="08/14" Speak="True" Reminder="2"
Reminder_Type="Days">
```

Sets a Birthday's Reminder for Guile's Birthday on Aug 14. Virtual Assistant will speak a reminder 2 days before.

```
<Calendar="Alarm" Action="Add" Description="Time to take a break!" Date="Today"
Time="18:00" Speak="True" AgentText="Guile, it's time to take a break!" />
```

Sets an Alarm to 18:00 today with the Assistant speaking the message "Guile, it's time to take a break!" when the times comes.

```
<Calendar="Doctor" Action="Add" DoctorsName="Dr. Ryan" Date="10/30/2010" Time="14:30"
Speak="True" Reminder="3" Reminder_Type="Days">
```

Sets an Appointment with Dr. Ryan on October 30, 2010 at 14:30. Virtual Assistant will speak a reminder three days before.

```
<Calendar="Courses" Action="Add" CoursesName="Marketing 3.0" DateStart="10/05/2010"
TimeStart="09:00" Speak="True" Reminder="1" Reminder_Type="Week">
```

Sets a Course event for Marketing 3.0 on October 5, 2010 starting at 09:00. Virtual Assistant will speak a reminder one week before.

```
<Calendar="Events" Action="Add" What="Office Party" DateStart="11/02/2010"
TimeStart="21:00" Speak="True" Reminder="5" Reminder_Type="hours">
```

Sets a new event named "Office Party" for November 2, 2010 at 21:00. Virtual Assistant will speak a reminder five hours before.

```
<Calendar="Maintenance" Action="Add" Vehicle="Mustang Revision" Date="11/05/2010"
Time="15:30" Speak="True" Reminder="1" Reminder_Type="days">
```

Sets a new Maintenance event for car "Mustang" on November 05, 2010 at 15:30. Virtual Assistant will speak a reminder one day before.

```
<Calendar=" Medicines" Action="Add" MedicinesName ="trandolapril" Time= "09:00"  
HowMany="2" DateStart="11/01/2010" DateEnd="12/01/2010" Speak="True" Reminder="10"  
Reminder_Type="Minutes">
```

Sets a new Medicine event to take two units of medicine "trandolapril" at 09:00 starting on November 1, 2010 and ending on December 1, 2010. Virtual Assistant will speak a reminder ten minutes before every event.

```
<Calendar="Meeting" Action="Add" Subject="Meeting with Development Team"  
Date="11/15/2010" TimeStart="08:30" TimeEnd="09:30" Info="New Products Review"  
Speak="True" Reminder="1" Reminder_Type="Day">
```

Sets a new Meeting event named "Meeting with Development Team" with detail info "New Products Review" for November 15, 2010 starting at 08:30 to 09:30. Virtual Assistant will speak a reminder one day before the event.

```
<Calendar="Tasks" Action="Add" What="Buy Milk" Date="Tomorrow" Time="19:00"  
Speak="True" Reminder="1" Reminder_Type="Day">
```

Sets a new Task called "Buy Milk" for tomorrow at 19:00. Virtual Assistant will speak a reminder one day before the task.

## 35) Contact

The “Contact” tag adds and deletes new personal contacts to user’s personal Agenda. This tag is not yet available.

## 36) Volume (SAPI5 TTS Tag)

The “Volume” tag controls the volume of the Assistant speech voice. This tag complies with Microsoft SAPI 5.3 (Speech Application Programming Interface) syntax.

### 36.1) Syntax

```
<volume level="Level">  
  Text to be spoken  
</volume>
```

“Level” valid values range from 0 to 100

### 36.2) Examples

```
<volume level="100">  
  This text should be spoken at volume level one hundred.  
</volume>
```

```
<volume level="80"/>  
All text which follows should be spoken at volume level eighty.
```

## 37) Rate (SAPI5 TTS Tag)

The “Rate” tag controls the rate of the Assistant speech voice. This tag complains with Microsoft SAPI 5.3 (Speech Application Programming Interface) syntax.

The Rate tag has two attributes, Speed and AbsSpeed, one of which must be present. The value of both of these attributes should be an integer between negative ten and ten. The AbsSpeed attribute controls the absolute rate of the voice, so a value of ten always corresponds to a value of ten, a value of five always corresponds to a value of five.

The Speed attribute controls the relative rate of the voice. The absolute value is found by adding each Speed to the current absolute value.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 37.1) Syntax

```
<rate absspeed="Level">
  This text should be spoken at rate five.
</rate>
```

Level values range from -10 to +10

```
<rate speed="Level">
  This text should be spoken at rate five.
</rate>
```

```
<rate speed="-5">
  This text should be spoken at rate zero.
</rate>
```

### 37.2) Examples

```
<rate absspeed="5">
  This text should be spoken at rate five.
</rate>
```

```
<rate absspeed="-5">
  This text should be spoken at rate negative five.
</rate>
```

```
<rate absspeed="10"/>
All text which follows should be spoken at rate ten.
<rate speed="5">
```

This text should be spoken at rate five.  
</rate>

<rate speed="-5">  
This text should be spoken at rate zero.  
</rate>

## 38) Spell (SAPI5 TTS Tag)

The “Spell” tag forces the voice to spell out all text, rather than using its default word and sentence breaking rules, normalization rules, and so forth. All characters should be expanded to corresponding words (including punctuation, numbers, and so forth).

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 38.1) Syntax

```
<spell>  
These words should be spelled out.  
</spell>  
These words should not be spelled out.
```

### 38.2) Examples

I will spell the word <spell>Love</spell> for you.

## 39) Pitch (SAPI5 TTS Tag)

The Pitch tag controls the pitch of a voice. The tag can be empty, in which case it applies to all subsequent text, or it can have content, in which case it only applies to that content.

The Pitch tag has two attributes, `Middle` and `AbsMiddle`, one of which must be present. The value of both of these attributes should be an integer between negative ten and ten.

The `AbsMiddle` attribute controls the absolute pitch of the voice, so a value of ten always corresponds to a value of ten, a value of five always corresponds to a value of five.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 39.1) Syntax

```
<pitch absmiddle="5">  
This text should be spoken at pitch five.  
</pitch>
```

```
<pitch absmiddle="-5">  
This text should be spoken at pitch negative five.  
</pitch>
```

```
<pitch absmiddle="10"/>  
All text which follows should be spoken at pitch ten.
```

The `Middle` attribute controls the relative pitch of the voice. The absolute value is found by adding each `Middle` to the current absolute value.

```
<pitch middle="5">  
This text should be spoken at pitch five.  
</pitch>
```

```
<pitch middle="-5">  
This text should be spoken at pitch zero.  
</pitch>
```

Zero represents the default middle pitch for a voice, with positive values being higher and negative values being lower.

## 40) Silence (SAPI5 TTS Tag)

The “Silence” tag inserts a specified number of milliseconds of silence into the output audio stream. This tag has one attribute, Msec.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 40.1) Syntax

```
<silence msec="Number of Milliseconds"/>
```

1000 Milliseconds = 1 second

5000 Milliseconds = 5 seconds

20000 Milliseconds = 20 seconds

### 40.2) Examples

Two seconds of silence `<silence msec="2000"/>` has just occurred.

## 41) Emph (SAPI5 TTS Tag)

The “Emph” tag instructs the voice to emphasize a word or section of text.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 41.1) Syntax

```
<emph>Text</emph>
```

### 41.2) Examples

The following words should be emphasized.

I will scare you. `<emph>Boo</emph>!`

This is `<emph>very</emph>` important!

The method of emphasis may vary from voice to voice.

## 42) Pron (SAPI5 TTS Tag)

The “Pron” tag inserts a specified pronunciation. The voice will process the sequence of phonemes exactly as they are specified.

The “Pron” tag has one attribute, Sym, whose value is a string of white space separated phonemes.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 42.1) Syntax

```
<pron sym="Phonemes"> Text </pron>
```

### 42.2) Examples

```
<pron sym="h eh l l ow & w er l l d"> hello world </pron>
```

## 43) PartOfSp (SAPI5 TTS Tag)

The “PartOfSp” tag provides the voice with the part of speech of the enclosed word(s). Use this tag to enable the voice to pronounce a word with multiple pronunciations correctly depending on its part of speech.

The “PartOfSp” tag has one attribute, Part, which takes a string corresponding to a SAPI part of speech as its attribute. Only SAPI defined parts of speech are supported - "Unknown", "Noun", "Verb", "Modifier", "Function", "Interjection".

This tag is very useful to make TTS voices to correct pronounce words that are written identically but have different pronunciations or meanings (**heteronyms**), like the word “live” that as a verb means “to be alive”, and as an adjective means “having life”, or the word “present”, that as a verb means “to reveal” , and as a noun means “a gift” .

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 43.1) Syntax

```
I will <partofsp part="verb"> present </partofsp> my documentation as soon as possible.
```

```
This is a <partofsp part="noun">present</partofsp> from my friend to you.
```

```
Did you <partofsp part="verb"> record </partofsp> that important database <partofsp part="noun"> record </partofsp>?
```

## 44) Context (SAPI5 TTS Tag)

The “Context” tag provides the voice with information which the voice may then use to determine how to normalize special items, like dates, numbers, and currency. Use this tag to enable the voice to distinguish between confusable date formats.

The Context tag has one attribute, `id`, which takes a string corresponding to the context of the enclosed text. Several contexts are defined by SAPI and are more likely to be recognized by SAPI compliant voices, but any string may be used. See documentation for a particular voice for more details.

Important: TTS voices are built by different companies, and this tag may not work with some TTS voices.

### 44.1) Syntax

```
<context id="date_mdy"> 03/04/10 </context> should be spoken March  
fourth, two thousand ten.
```

```
<context id="date_dmy"> 03/04/10 </context> should be April third, two  
thousand ten.
```

```
<context id="date_ymd"> 03/04/10 </context> should be April ten, two  
thousand three.
```

## 45) Memory

The “Memory” tag saves new knowledge in Denise’s databases. It is similar to the “AIML” tag, but instead of saving new questions / answers in an AIML file, the “Memory” tag saves them in a database.

This tag tries to implement some of what the PROLOG computer language does, expressing logic in terms of objects or terms relations.

This tag is still being improved and we recommend that you use it just for experimental purposes.

### 45.1) Syntax

```
<Memory="Object" Property="Object Property" Value=" Property Value" Query="Search Query">
```

- Memory: Object to memorize. Ex: Car, dog, my family, The Universe
- Property: Property type to be associated with the Object. Ex: colour (car), class (dog), adjective (family), size (The Universe)
- Value: Value to be associated with the Object Property. Ex: black (car colour), mammal (dog class), wonderful (family adjective), infinite (The Universe size)
- ActionQuery: executes an action or a query on memorized data. There are two available ActionQueries:
  - Exist: Returns True or False
  - Delete: Removes an object or a property’s value

### 45.2) Examples

```
<Memory="My Car" Property="Colour" Value="Red">
```

Denise will memorize that the colour of my car is red

```
<Memory="My Car" Property="Colour">
```

Denise will check if there is an object with the value “My Car” with a “Colour” property associated with it. If so, she will return its value, in this example would be “Red”

```
<Memory="My Car" Property="Colour" ActionQuery="Exist">
```

Denise will check if there is an Object named “My Car” with a “Colour” property. If so, she returns True, otherwise returns False.

<Memory="My Car" ActionQuery="Exist">

Denise will check if there is an Object named "My Car". If so, she returns True, otherwise returns False.

This example can be used to ask questions like: Do I have a car?

<Memory="My Car" Value="Red" ActionQuery="Exist">

Denise will check if there is an Object named "My Car" that is associated with a property that has a "Red" value. If so, she returns True, otherwise returns False.

This example can be used to ask questions like: Does My Car has anything red on it?

<Memory="My Car" Property="Colour" Value="Red" ActionQuery="Exist">

Denise will check if there is an Object named "My Car" that is associated with a property "Colour" that has a value "Red"

This example can be used to ask questions like: Is My Car red?

<Memory="" Property="Colour" Value="Red" ActionQuery="Exist">

Denise will check if there is any Object associated with a property "Colour" that has a "Red" value. If so, she returns True, otherwise returns False.

This example can be used to ask questions like: Do you know anything with the red colour?

<Memory="" ActionQuery="Exist">

If memory has no saved data, returns False. If memory has some data, returns True.

<Memory="My Car" Property="Colour" ActionQuery="Delete">

Deletes property "Colour" that was associated with object "My Car"

<Memory="My Car" ActionQuery="Delete">

Deletes all values associated with object "My Car"

<Memory="" Property="Colour" ActionQuery="Delete">

Deletes all properties with "Colour" value from all objects in memory

<Memory="My Car" Value="Red" ActionQuery="Delete">

Deletes all properties with "Red" value from object "My Car"

<Memory="" Value="Mammals" ActionQuery="Delete">

Deletes all properties with "Mammals" value from all objects

This example can be used to ask questions like: Forget all you know about mammals.

<Memory="" ActionQuery="Delete">

Deletes everything from Denise's memory.

## 46) MediaPlayer

The “MediaPlayer” tag is used to open and play multimedia files, allowing the Assistant to play songs, videos, listen to Web Radio and watch Web TV Stations.

### 46.1) Syntax

```
<MediaPlayer="Action" FileName="File Name" Artist="Artist" Album="Album" Title="Song Title" Genre="Song Genre" Volume="100" Match="Key Word" Function="Function">
```

- MediaPlayer: There are several actions available:

1- Play: Starts playing a song or resume a paused song.

2- Pause: Pauses a song, video, Web Radio or Web TV

3- Stop: Stops a song, video, Web Radio or Web TV

4- Add: Add a song to active playlist.

5- Next: Skip to next song.

6- Previous: Go back one song.

7- Shuffle: Chooses a random song from your music library.

8- Shuffle-ON: Sets Random mode on. Songs will be played randomly.

9- Shuffle-OFF: Sets Random mode off. Songs will be played according listing order.

10- Say: Assistant Speaks artist name, album or song title.

11-Say-ON: Assistant Speaks artist name, album or song title every new song.

12-Say-OFF: Assistant stops speaking artist name, album or song title every new song.

- Artist: Song’s artist name.

- Album: Song’s album name.

-Genre: Song’s Genre. Not every song has this information, so some queries can result no songs.

- Title: Song’s title or name.

- Match: Is a wildcard to find songs by telling just part of its name.

- Volume: Changes song playing Volume. Value can range from 0 to 100. To change System Volume use the "SysVolume" tag. To change Assistant volume use the tag <Volume>

- FileName: Song and Video file name or Web TV or Web Radio Station streaming files. You can also use the word "CDAUDIO" to play songs from a Music CD that is placed in your CD Rom Player. To play a specific CD Track, use "E:\TRACK01.CDA" as the filename. In case you don't know the CD Drive path, you can use "TRACK01.CDA".

You can also play latest songs recorded to your computer from a Music CD using  
<mediaplayer="play" filename="LAST">

IMPORTANT: The .CDA extension must be informed to play songs from a Music CD.

- Function: Used to do some especial actions, like listing Web Radio and Web TV Stations

<Module="MediaPlayer" Function="ListTVs">

List all Web TV Stations

<Module="MediaPlayer" Function="ListRadios">

List all Web Radio Stations

## 46.2) Examples

<MediaPlayer="Add" Artist=" Coldplay">

Add all songs from the Group Coldplay at the end of the current playlist.

<MediaPlayer="Pause">

Pauses current media (can be a song, video, Web Radio or Web TV)

< MediaPlayer="Play" FileName="http://www.fusionchicago.com/streams/128.asx">

Plays the Chicago Fusion Web Radio Station

<MediaPlayer="Play" FileName=http://www.bloomberg.com/streams/video/LiveBTV200.asxx>

Plays the Bloomberg Web TV Station

<Mediaplayer="Play" filename="LAST">

Plays latest songs recorded to your computer from a Music CD

<MediaPlayer="Stop">

Stops playing.

<MediaPlayer="Play">

Play or resumes song or video.

<MediaPlayer="Next">

Go to next song.

<MediaPlayer="Previous">

Go back to previous song.

<MediaPlayer="Play" volume="50">

Set Volume to middle level.

<MediaPlayer="Next" volume="100">

Go to next song and Sets the Volume to highest level.

<MediaPlayer="Play" Album="Oasis" Volume="100">

Cleans current playlist, starts playing Oasis songs and Sets the Volume to highest level.

. This example shows a combination of many parameters inside the tag.

<MediaPlayer="Play" Genre="Classic">

Play only Classic songs. Note that not all songs have Genre information. (Although this can be set by user)

<MediaPlayer="Shuffle-ON" >

Sets Random mode on. This plays all songs in random mode.

<MediaPlayer="Shuffle-OFF" >

Sets Random mode off.

<MediaPlayer="Say" Title="True" Artist="True">

Assistant speaks current song name and Artist name.

<MediaPlayer="Say" FileName="Happy.mp3" Artist="True" Title="True" Album="True">

Assistant speaks current song name and Album name.

Tags can be used combined inside the same AIML categorie template, giving interesting results.

<MediaPlayer="Play" Genre="SoundTrack"> <MediaPlayer="Say" Title="True">

Plays just Soundtracks songs with Assistant speaking every new song name as soon as they start playing.

<MediaPlayer="Next"> <MediaPlayer="Say" Title="True">

Skips one song and speaks next song title.

<MediaPlayer="Say-ON" Title="True">

Sets Speaking Title songs on

<MediaPlayer="SHUFFLE">

If media player is closed or not playing, starts playing a random song.

<MediaPlayer="PLAY" Match="love">

Builds a playlist with songs that has the word "love". This can be songs titles like "The Love of My Life", "My Love", "Endless Love" etc.

```
<MediaPlayer="Play" FileName="CDAUDIO">
```

Plays all songs from a Music CD, if there is one in the CD ROM Player.

```
<MediaPlayer="Play" FileName="TRACK03.CDA">
```

Plays Track 3 from a Music CD, if there is one in the CD ROM Player.

## 47) ChDir

The “ChDir” tag opens a Windows folder or directory that will be opened using Windows Explorer. This tag is a direct call to Windows API interface.

### 47.1) Syntax

<CHDIR=”Folder to Open”> or

We can also use the “CSIDL” word from the Windows API (Constant special item ID list).

<CHDIR=” CSIDL Code”>

- CHDIR: Type what Windows folder to open

For further information on using CSIDL and update options, please go to:

[http://msdn.microsoft.com/en-us/library/bb762494\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/bb762494(VS.85).aspx)

Available CSIDL code are:

#### **CSIDL\_ADMINTOOLS** (FOLDERID\_AdminTools)

The file system directory that is used to store administrative tools for an individual user. The Microsoft Management Console (MMC) will save customized consoles to this directory, and it will roam with the user.

#### **CSIDL\_ALTSTARTUP** (FOLDERID\_Startup)

The file system directory that corresponds to the user's nonlocalized Startup program group. This value is recognized in Windows Vista for backward compatibility, but the folder itself no longer exists.

#### **CSIDL\_APPDATA** (FOLDERID\_RoamingAppData)

The file system directory that serves as a common repository for application-specific data. A typical path is C:\Documents and Settings\*username*\Application Data. This CSIDL is supported by the redistributable Shfolder.dll for systems that do not have the Microsoft Internet Explorer 4.0 integrated Shell installed.

#### **CSIDL\_BITBUCKET** (FOLDERID\_RecycleBinFolder)

The virtual folder that contains the objects in the user's **Recycle Bin**.

#### **CSIDL\_CDBURN\_AREA** (FOLDERID\_CDBurning)

The file system directory that acts as a staging area for files waiting to be written to a CD. A typical path is C:\Documents and Settings\*username*\Local Settings\Application Data\Microsoft\CD Burning.

**CSIDL\_COMMON\_ADMINTOOLS** (FOLDERID\_CommonAdminTools)

**Version 5.0.** The file system directory that contains administrative tools for all users of the computer.

**CSIDL\_COMMON\_ALTSTARTUP** (FOLDERID\_CommonStartup)

The file system directory that corresponds to the nonlocalized Startup program group for all users. Valid only for Microsoft Windows NT systems. This value is recognized in Windows Vista for backward compatibility, but the folder itself no longer exists.

**CSIDL\_COMMON\_APPDATA** (FOLDERID\_ProgramData)

**Version 5.0.** The file system directory that contains application data for all users. A typical path is C:\Documents and Settings\All Users\Application Data. This folder is used for application data that is not user specific. For example, an application can store a spell-check dictionary, a database of clip art, or a log file in the CSIDL\_COMMON\_APPDATA folder. This information will not roam and is available to anyone using the computer.

**CSIDL\_COMMON\_DESKTOPDIRECTORY** (FOLDERID\_PublicDesktop)

The file system directory that contains files and folders that appear on the desktop for all users. A typical path is C:\Documents and Settings\All Users\Desktop. Valid only for Windows NT systems.

**CSIDL\_COMMON\_DOCUMENTS** (FOLDERID\_PublicDocuments)

The file system directory that contains documents that are common to all users. A typical path is C:\Documents and Settings\All Users\Documents. Valid for Windows NT systems and Microsoft Windows 95 and Windows 98 systems with Shfolder.dll installed.

**CSIDL\_COMMON\_FAVORITES** (FOLDERID\_Favorites)

The file system directory that serves as a common repository for favorite items common to all users. Valid only for Windows NT systems.

**CSIDL\_COMMON\_MUSIC** (FOLDERID\_PublicMusic)

**Version 6.0.** The file system directory that serves as a repository for music files common to all users. A typical path is C:\Documents and Settings\All Users\Documents\My Music.

**CSIDL\_COMMON\_OEM\_LINKS** (FOLDERID\_CommonOEMLinks)

This value is recognized in Windows Vista for backward compatibility, but the folder itself is no longer used.

**CSIDL\_COMMON\_PICTURES** (FOLDERID\_PublicPictures)

**Version 6.0.** The file system directory that serves as a repository for image files common to all users. A typical path is C:\Documents and Settings\All Users\Documents\My Pictures.

**CSIDL\_COMMON\_PROGRAMS** (FOLDERID\_CommonPrograms)

The file system directory that contains the directories for the common program groups that appear on the **Start** menu for all users. A typical path is C:\Documents and Settings\All Users\Start Menu\Programs. Valid only for Windows NT systems.

**CSIDL\_COMMON\_STARTMENU** (FOLDERID\_CommonStartMenu)

The file system directory that contains the programs and folders that appear on the **Start** menu for all users. A typical path is C:\Documents and Settings\All Users\Start Menu. Valid only for Windows NT systems.

**CSIDL\_COMMON\_STARTUP** (FOLDERID\_CommonStartup)

The file system directory that contains the programs that appear in the Startup folder for all users. A typical path is C:\Documents and Settings\All Users\Start Menu\Programs\Startup. Valid only for Windows NT systems.

**CSIDL\_COMMON\_TEMPLATES** (FOLDERID\_CommonTemplates)

The file system directory that contains the templates that are available to all users. A typical path is C:\Documents and Settings\All Users\Templates. Valid only for Windows NT systems.

**CSIDL\_COMMON\_VIDEO** (FOLDERID\_PublicVideos)

**Version 6.0.** The file system directory that serves as a repository for video files common to all users. A typical path is C:\Documents and Settings\All Users\Documents\My Videos.

**CSIDL\_COMPUTERSNEARME** (FOLDERID\_NetworkFolder)

The folder that represents other computers in your workgroup.

**CSIDL\_CONNECTIONS** (FOLDERID\_ConnectionsFolder)

The virtual folder that represents Network Connections, that contains network and dial-up connections.

**CSIDL\_CONTROLS** (FOLDERID\_ControlPanelFolder)

The virtual folder that contains icons for the Control Panel applications.

**CSIDL\_COOKIES** (FOLDERID\_Cookies)

The file system directory that serves as a common repository for Internet cookies. A typical path is C:\Documents and Settings\username\Cookies.

**CSIDL\_DESKTOP** (FOLDERID\_Desktop)

The virtual folder that represents the Windows desktop, the root of the namespace.

**CSIDL\_DESKTOPDIRECTORY** (FOLDERID\_Desktop)

The file system directory used to physically store file objects on the desktop (not to be confused with the desktop folder itself). A typical path is C:\Documents and Settings\username\Desktop.

**CSIDL\_DRIVES** (FOLDERID\_ComputerFolder)

The virtual folder that represents My Computer, containing everything on the local computer: storage devices, printers, and Control Panel. The folder can also contain mapped network drives.

**CSIDL\_FAVORITES** (FOLDERID\_Favorites)

The file system directory that serves as a common repository for the user's favorite items. A typical path is C:\Documents and Settings\username\Favorites.

**CSIDL\_FONTS** (FOLDERID\_Fonts)

A virtual folder that contains fonts. A typical path is C:\Windows\Fonts.

**CSIDL\_HISTORY** (FOLDERID\_History)

The file system directory that serves as a common repository for Internet history items.

**CSIDL\_INTERNET** (FOLDERID\_InternetFolder)

A virtual folder for Internet Explorer.

**CSIDL\_INTERNET\_CACHE** (FOLDERID\_InternetCache)

The file system directory that serves as a common repository for temporary Internet files. A typical path is C:\Documents and Settings\*username*\Local Settings\Temporary Internet Files.

**CSIDL\_LOCAL\_APPDATA** (FOLDERID\_LocalAppData)

The file system directory that serves as a data repository for local (nonroaming) applications. A typical path is C:\Documents and Settings\*username*\Local Settings\Application Data.

**CSIDL\_MYDOCUMENTS** (FOLDERID\_Documents)

The virtual folder that represents the My Documents desktop item. This value is equivalent to [CSIDL\\_PERSONAL](#).

**CSIDL\_MYMUSIC** (FOLDERID\_Music)

The file system directory that serves as a common repository for music files. A typical path is C:\Documents and Settings\User\My Documents\My Music.

**CSIDL\_MYPICTURES** (FOLDERID\_Pictures)

The file system directory that serves as a common repository for image files. A typical path is C:\Documents and Settings\*username*\My Documents\My Pictures.

**CSIDL\_MYVIDEO** (FOLDERID\_Videos)

The file system directory that serves as a common repository for video files. A typical path is C:\Documents and Settings\*username*\My Documents\My Videos.

**CSIDL\_NETHOOD** (FOLDERID\_NetHood)

A file system directory that contains the link objects that may exist in the **My Network Places** virtual folder. It is not the same as [CSIDL\\_NETWORK](#), which represents the network namespace root. A typical path is C:\Documents and Settings\*username*\NetHood.

**CSIDL\_NETWORK** (FOLDERID\_NetworkFolder)

A virtual folder that represents Network Neighborhood, the root of the network namespace hierarchy.

**CSIDL\_PERSONAL** (FOLDERID\_Documents)

The virtual folder that represents the My Documents desktop item. This is equivalent to [CSIDL\\_MYDOCUMENTS](#).

The file system directory used to physically store a user's common repository of documents. A typical path is C:\Documents and Settings\*username*\My Documents. This should be distinguished from the virtual **My Documents** folder in the namespace. To access that virtual folder, use [SHGetFolderLocation](#), which returns the [ITEMIDLIST](#) for the virtual location, or refer to the technique described in [Managing the File System](#).

**CSIDL\_PRINTERS** (FOLDERID\_PrintersFolder)

The virtual folder that contains installed printers.

**CSIDL\_PRINTHOOD** (FOLDERID\_PrintHood)

The file system directory that contains the link objects that can exist in the **Printers** virtual folder. A typical path is C:\Documents and Settings\*username*\PrintHood.

**CSIDL\_PROFILE** (FOLDERID\_Profile)

The user's profile folder. A typical path is C:\Users\*username*. Applications should not create files or folders at this level; they should put their data under the locations referred to by [CSIDL\\_APPDATA](#) or [CSIDL\\_LOCAL\\_APPDATA](#). However, if you are creating a new Known Folder the profile root referred to by CSIDL\_PROFILE is appropriate.

**CSIDL\_PROGRAM\_FILES** (FOLDERID\_ProgramFiles)

The Program Files folder. A typical path is C:\Program Files.

**CSIDL\_PROGRAM\_FILESX86** (FOLDERID\_ProgramFilesX86)**CSIDL\_PROGRAM\_FILES\_COMMON** (FOLDERID\_ProgramFilesCommon)

A folder for components that are shared across applications. A typical path is C:\Program Files\Common. Valid only for Windows NT, Windows 2000, and Windows XP systems. Not valid for Windows Millennium Edition (Windows Me).

**CSIDL\_PROGRAM\_FILES\_COMMONX86** (FOLDERID\_ProgramFilesCommonX86)**CSIDL\_PROGRAMS** (FOLDERID\_Programs)

The file system directory that contains the user's program groups (which are themselves file system directories). A typical path is C:\Documents and Settings\*username*\Start Menu\Programs.

**CSIDL\_RECENT** (FOLDERID\_Recent)

The file system directory that contains shortcuts to the user's most recently used documents. A typical path is C:\Documents and Settings\*username*\My Recent Documents. To create a shortcut in this folder, use [SHAddToRecentDocs](#). In addition to creating the shortcut, this function updates the Shell's list of recent documents and adds the shortcut to the **My Recent Documents** submenu of the **Start** menu.

**CSIDL\_RESOURCES** (FOLDERID\_ResourceDir)

Windows Vista. The file system directory that contains resource data. A typical path is C:\Windows\Resources.

**CSIDL\_RESOURCES\_LOCALIZED** (FOLDERID\_LocalizedResourcesDir)**CSIDL\_SENDTO** (FOLDERID\_SendTo)

The file system directory that contains **Send To** menu items. A typical path is C:\Documents and Settings\*username*\SendTo.

**CSIDL\_STARTMENU** (FOLDERID\_StartMenu)

The file system directory that contains **Start** menu items. A typical path is C:\Documents and Settings\*username*\Start Menu.

**CSIDL\_STARTUP** (FOLDERID\_Startup)

The file system directory that corresponds to the user's Startup program group. The system starts these programs whenever any user logs onto Windows NT or starts Windows 95. A typical path is C:\Documents and Settings\*username*\Start Menu\Programs\Startup.

**CSIDL\_SYSTEM** (FOLDERID\_System)

The Windows System folder. A typical path is C:\Windows\System32.

**CSIDL\_SYSTEMX86** (FOLDERID\_SystemX86)

**CSIDL\_TEMPLATES** (FOLDERID\_Templates)

The file system directory that serves as a common repository for document templates. A typical path is C:\Documents and Settings\*username*\Templates.

**CSIDL\_WINDOWS** (FOLDERID\_Windows)

The Windows directory or SYSROOT. This corresponds to the %windir% or %SYSTEMROOT% environment variables. A typical path is C:\Windows.

## 47.2) Examples

<CHDIR=" C:\">

Opens root folder of C Hard Disk.

< CHDIR ="CSIDL\_MYPICTURES">

Opens "My Pictures" folder. (This folder is on "C:\Documents and Settings\Username\My Documents\My Pictures")

< CHDIR="CSIDL\_MYDOCUMENTS">

Opens "My Documents" folder.

## 48) ChatDialog

This tag creates a “Yes” “No” style menu button in the Input area of the Assistant Chat Menu.

This is a nice way to offer user a quick way to answer a question, giving him two options that trigger different actions.

### 48.1) Syntax

```
<ChatDialog="Question" Buttons="Button 1 Caption, Button 1 Caption" ButtonsValues="AIML Category to Trigger if Button 1 is chosen, AIML Category to Trigger if Button 2 is chosen">
```

### 48.2) Examples

```
<ChatDialog="Would you like to Know more?" Buttons="Yes,No Thanks" ButtonsValues="Tell me More,No">
```

In this sample, the Assistant will ask user “Would you like to Know more?” and two buttons will appear in the Input Chat box with a “Yes” and “No Thanks” captions.

Clicking on button “Yes”, the AIML “Tell me More” category will be trigger.

## 49)ChatInput

The “ChatInput” simulates some typing from user. It will perform a text input in the chat input window, as user had typed the text using the keyboard or voice. It also simulates the <ENTER> keyboard key pressing.

This tag can be very useful for 3<sup>rd</sup> part software integration, when a external application simulates a text input for Denise. This will be discussed further at the moment guile 3D Studio releases its SDK (Software Development Kit).

This tag will also be useful for the upcoming Home Automation Kit from Guile 3D Studio.

Other useful way to use this tag from inside an AIML category, is to show the text used inside the tag in the Chat input window before executing the correspondent AIML action.

### 49.1) Syntax

```
<ChatInput="Text or AIML category to be executed ">
```

### 49.2) Examples

```
<ChatInput="Good morning Denise ">
```

```
<ChatInput="Date ">
```

This example will show the word “Date” as the user has typed with the keyboard, and also will simulate the press of <Enter> keyboard key. The engine will then try to find inside all AIML categories the best match for the word “Date” and then return the answer. Actually, would be the same as the user had typed “Date” using the keyboard.

## 50) Buy

Use the tag “Buy” to search the Internet for information about anything to buy.

### 50.1) Syntax

```
<Buy="Product" Type="Product Type">
```

Buy: Product of interest. Can be a book, a music CD, a video DVD, a computer, or anything else.

Type: Directs the search for specific product type web vendors. Available values are:

Book: Book search

DVD: DVD search

TICKET: Tickets search

RENTCAR: car rental search

APPLIANCES: Appliance search

OTHER: any other product

### 50.2) Examples

```
<BUY="Artificial Intelligence" Type="Book">
```

Will search for books about Artificial Intelligence

```
<BUY="The Corrs" Type="DVD">
```

Will search for DVD from the Band The Corrs

```
<BUY="I7 Laptop" Type="other">
```

Will search for Laptops with I7 Processor

## 51) CDAudio

The "CDAUDIO" tag is used to open and close the CD Rom drive, as well as RIP (copy from a Music CD and save in Hard Disk in the .MP3 format) all song tracks from a Music CD.

### 51.1) Syntax

<CDAUDIO="Action">

Available actions are:

- EJECT: Ejects CD
- CLOSE: Closes CD Rom drive
- CopyToFile: Copy all music tracks from a music CD.

### 51.2) Examples

<CDAUDIO="CLOSE">  
Closes CD Rom drive

<CDAUDIO="EJECT">  
Ejects CD

<CDAUDIO="CopyToFile">  
Copy all music tracks from a music CD.

## 52)SoundCard

Return system's Sound Cards information.

### 52.1) *Syntax*

<SoundCard>

## 53)Conjugate

This tag conjugates verbs to a certain tense. This tag is not available yet.

### 53.1) Syntax

```
<Conjugate="Verb" Person="Person" Time="Tense">
```

- Conjugate: Verb to be conjugated.
- Person: Person to conjugate to. (First, Second, or Third).
- Time: Verb Tense to conjugate. Example: Present Perfect Tense.

### 53.2) Example

```
<Conjugate="I know it" Person="First" Time=" Present Perfect">
```

Result: "I have known it"

## 54) Battery

The “Battery” tag returns portable computer battery charge status. Denise will automatically use this tag and speaks a message if your portable computer is running out of battery charge.

### 54.1) Syntax

```
<Battery="Parameter">
```

Parameter Value:

- Life: returns portable computer battery charge status.

### 54.2) Example

```
<Battery="Life">
```

Virtual Assistant will speak battery status:

“Your battery is 86% charged.”

or:

“Could not find any battery in your computer” (for desktop computers)

## 55)Keyboard

The “Keyboard” tag simulates a keyboard input or a mouse action. This tag is very useful to control 3<sup>rd</sup> part applications, simulating a text input in other desktop windows, just like if user had typed a text using the keyboard or clicked a mouse button manually.

This tag also can simulate a keyboard key press, like pressing the “ENTER” key to execute some action.

### 55.1) Syntax

<Keyboard=“Text to be typed or key to be pressed” Window=“Application Window Title” Delay=“Delay between simulation typing”>

- Keyboard: Text to be typed or keyboard key or mouse button to be pressed. For keyboard key and mouse button values, use below table as reference:

Keyboard or Mouse Key Code	Keyboard or Mouse Key
VK_LBUTTON	Left mouse button
VK_RBUTTON	Right mouse button
VK_CANCEL	Control-break processing
VK_MBUTTON	Middle mouse button (three-button mouse)
VK_BACK	BACKSPACE key
VK_TAB	TAB key
VK_CLEAR	CLEAR key
VK_RETURN	ENTER key
VK_SHIFT	SHIFT key
VK_CONTROL	CTRL key
VK_MENU	ALT key
VK_PAUSE	PAUSE key
VK_CAPITAL	CAPS LOCK key
VK_ESCAPE	ESC key
VK_SPACE	SPACEBAR
VK_PRIOR	PAGE UP key
VK_NEXT	PAGE DOWN key
VK_END	END key
VK_HOME	HOME key
VK_LEFT	LEFT ARROW key
VK_UP	UP ARROW key
VK_RIGHT	RIGHT ARROW key
VK_DOWN	DOWN ARROW key
VK_SELECT	SELECT key
VK_PRINT	PRINT key
VK_EXECUTE	EXECUTE key
VK_SNAPSHOT	PRINT SCREEN key
VK_INSERT	INS key
VK_DELETE	DEL key
VK_HELP	HELP key
VK_NUMPAD0	Numeric keypad 0 key

VK_NUMPAD1	Numeric keypad 1 key
VK_NUMPAD2	Numeric keypad 2 key
VK_NUMPAD3	Numeric keypad 3 key
VK_NUMPAD4	Numeric keypad 4 key
VK_NUMPAD5	Numeric keypad 5 key
VK_NUMPAD6	Numeric keypad 6 key
VK_NUMPAD7	Numeric keypad 7 key
VK_NUMPAD8	Numeric keypad 8 key
VK_NUMPAD9	Numeric keypad 9 key
VK_SEPARATOR	Separator key
VK_SUBTRACT	Subtract key
VK_DECIMAL	Decimal key
VK_DIVIDE	Divide key
VK_F1	F1 key
VK_F2	F2 key
VK_F3	F3 key
VK_F4	F4 key
VK_F5	F5 key
VK_F6	F6 key
VK_F7	F7 key
VK_F8	F8 key
VK_F9	F9 key
VK_F10	F10 key
VK_F11	F11 key
VK_F12	F12 key
VK_F13	F13 key
VK_F14	F14 key
VK_F15	F15 key
VK_F16	F16 key
VK_F17	F17 key
VK_F18	F18 key
VK_F19	F19 key
VK_F20	F20 key
VK_F21	F21 key
VK_F22	F22 key
VK_F23	F23 key
VK_F24	F24 key
VK_NUMLOCK	NUM LOCK key
VK_SCROLL	SCROLL LOCK key
VK_LSHIFT	Left SHIFT key
VK_RSHIFT	Right SHIFT key
VK_LCONTROL	Left CONTROL key
VK_RCONTROL	Right CONTROL key
VK_LMENU	Left MENU key
VK_RMENU	Right MENU key
VK_PLAY	Play key
VK_ZOOM	Zoom key

- Window: Application Window Title Name (the name that appears on any application top window area).

- Delay: time in milliseconds between every typing simulation. If zero or not informed, Denise send all text at once. If informed, Denise sends every letter of the text with a delay, giving the illusion that a real person is typing a text in another application.

## 55.2-Examples

```
<Keyboard="Hello everybody!" Window="Notepad" Delay="10">
```

Types the text "Hello everybody!" inside an opened Windows Notepad.

```
<Keyboard="VK_RETURN" Window="Winamp">
```

Press "ENTER" key on an opened Winamp application.

```
<Keyboard="VK_NEXT" Window="Explorer">
```

Press "Page DOWN" key on an opened Internet Explorer web page.

## 56) Skype

Use the tag “Skype” to interface the Virtual Assistant with Skype chat and calling features.

Using this tag, you can ask the Virtual Assistant to make a Skype phone call or send a Skype text message to someone in your Skype buddy list.

Important: Free Skype software must be installed in your computer to use this functionality.

### 56.1) Syntax

```
<Skype="SendMessage" To="Skype contact name" Msg="Message Text ">
```

```
<Skype="Call" To=" Skype contact name">
```

```
<Skype="FinishCall">
```

Important: The Skype buddy name used in the “To” parameter must already be in your Skype buddy list.

### 56.2) Examples

```
<Skype="SendMessage" To="Guile" Msg="Hi Guile, what's up? ">
```

Sends the message “Hi Guile, what’s up?” to your Skype buddy “Guile”

```
<Skype="Call" To=" Guile">
```

Starts a Phone call to your Skype buddy “Guile”

```
<Skype="FinishCall">
```

Finish a Skype Phone call session.

## 57) Feeds

Use tag “Feeds” to show RSS Feeds in the Chat Lateral Window. This tag can also make the Assistant speaks latest new from a RSS channel, and open the news’s full text inside Guile’s 3D Web Browser.

To add, delete or activate RSS Feeds Channels, use Denise Setup.

### 57.1) Syntax

<Feeds="Show">

Show all feeds from all RSS Channels

<Feeds="Show" Title="RSS Feed Channel">

Show all feeds from a RSS Channel using the name in the Title property

<Feeds="Say" Title=" RSS Feed Channel ">

Assistant reads latest new from a RSS Channel using the name in the Title property

<Feeds="Open" Title=" RSS Feed Channel ">

Opens latest news with its content in Denise’s Web Browser

### 57.2) Examples

<Feeds="Show">

Show all feeds from all RSS Channels

<Feeds="Show" Title="Sports">

Show all feeds from a RSS Channel that contains the “Sports” text on it’s title.

<Feeds="Say" Title="BBC ">

Assistant reads latest new from BBC RSS Channel (If this Channel exists)

<Feeds="Open" Title="CNN ">

Opens latest news from CNN RSS Channel with its full text content in Guile’s 3D Web Browser

## 58) EMail

Use the tag "EMail" to show, receive, check and send new e-mails.

To add, delete or edit E-mail accounts, use Denise's Setup.

### 58.1) Syntax

```
<EMail="Send" To="Contact" subject=" E-mail Subject" Message="E-mail Text Message ">
```

Parameters Values:

Contact: Name or e-mail of contact to send (Can be left empty)

E-mail Subject: E-mail Subject (Can be left empty)

E-mail Text Message: E-mail Body Text (Can be left empty)

```
<EMail="Receive">
```

Show all e-mails received from all accounts in Denise's Chat Lateral Window.

```
<Email="receive" type="INBOX">
```

Show all e-mails received from all accounts in Denise's Chat Lateral Window.

```
<Email="receive" type="SENT">
```

Show all e-mails sent from all accounts in Denise's Chat Lateral Window.

```
<Email="receive" type="OUTBOX">
```

Show all e-mails that are still being sent from all accounts in Denise's Chat Lateral Window.

```
<Email="closemail">
```

Closes the e-mail message window

```
<Email="receive" type="TRASH">
```

Show all e-mails deleted from all accounts in Denise's Chat Lateral Window.

Note: You can check for new e-mails as well as open your e-mail Webmail webpage using "Module" tag:

```
<Module="EMAIL" Function="Open" Parameters="NOW">
```

Check e-mail accounts for new e-mails. If so, Virtual Assistant speaks number of new e-mails

```
<Module="EMAIL" Function="Open" Parameters="WebMail">
```

Opens Guile's 3D Web Browser with default's e-mail account Webmail page.

## 58.2) Examples

```
<EMail="Send" To="Guile" subject="Denise is just Great!" Message="Dear Guile, Just to let you know that I'm in love with Denise. Regards.">
```

Write and send an e-mail to Guile with the subject "Denise is just Great!" and the body text "Dear Guile, Just to let you know that I'm in love with Denise. Regards."

Note: The "Guile" name used in "To" parameter has to exist already in database. If not, Denise will ask for his e-mail and save it on her database.

You can also type an e-mail address instead of a name in "to" parameter:

```
<EMail="Send" To="guile3d@guile3d.com" subject="Denise is just Great!" Message="Dear Guile, Just to let you know that I'm in love with Denise. Regards.">
```

And also send the e-mail for more than one contact:

```
<EMail="Send" To="guile3d@guile3d.com;support@guile3d.com" subject="Denise is just Great!" Message="Dear Guile, Just to let you know that I'm in love with Denise. Regards.">
```

You don't need to fill all parameters:

```
<EMail="Send" To="guile3d@guile3d.com" subject="" Message="">
```

In this case, Denise will open her lateral chat window so you can type empty fields.

```
<EMail="Receive">
```

Show all e-mails received from all accounts in Denise's Chat Lateral Window.

## 59) OpenResult

Use the tag "OpenResult" to execute actions over Denise's Lateral Chat Window. These actions can be:

- Show a image
- Plays a Video
- Opens a RSS feed
- Open an e-mail
- Shows a web search result

This tag is very useful for Voice Commands.

Denise shows e-mails, RSS Feeds, search queries, images and videos in her Lateral Chat Window in a listing format. Every e-mail, feed or image being displayed has a number showed at the beginning. Use this number as reference to call the desired action.

Denise knows what the Lateral Chat Window is showing at the moment (if it's showing your e-mails, feeds, images or search queries.) and according its type, opens, plays or show it.

### 59.1) Syntax

```
<OpenResult="Number">
```

### 59.2) Examples

```
<OpenResult="3">
```

If Denise is showing your e-mails, it will open a window with the third e-mail body content.

If Denise is showing RSS Feeds, it will open the number three news full text in Guile's 3D Web Browser.

If Denise is showing images, it will show number three image on its full size in Guile's 3D Web Browser.

If Denise is showing videos, it will play number three video in Guile's 3D Web Browser.

If Denise is showing web search results, it will open number three result link in Guile's 3D Web Browser.

## 60) Chat

Use tag "Chat" in conjunction with the "Module" tag to open and close Denise's lateral Chat Window.

### 60.1) Syntax

```
<Module="Chat" Function="ChatOpenMenu">
```

Open Denise's lateral Chat Window.

```
<Module="Chat" Function="ChatCloseMenu">
```

Close Denise's lateral Chat Window.

```
<Module="Chat", Function="AIMLTester">
```

### 60.2) Examples

```
<Module="Chat" Function="ChatOpenMenu">
```

Open Denise's lateral Chat Window.

```
<Module="Chat" Function="ChatCloseMenu">
```

Close Denise's lateral Chat Window.

## 61) ChatClear

Use the tag “ChatClear” to clear all text in Denise’s Chat Window.

### 61.1) Syntax

<ChatClear>

Clear all text in Denise’s Chat Window.

### 61.2) Examples

<ChatClear>