# Table of Contents

## Introduction

## Programming for Mobile

## Function Reference

### Animation Control Functions
- followCursor(mode) 6
- freezeToggle () 6
- recenter() 6
- setGaze(degrees, duration, [amplitude]) 7
- setFacialExpression(expression, amplitude, duration ) 7
- clearExpressionList( ) 8
- setIdleMovement(frequency, [amplitude]) 8
- setSpeechMovement(amplitude) 9
- animate(animationName) 9

### Speech Functions
- loadAudio(name) 9
- loadText(txt,voice,lang,engine,[effect], [effLevel]) 10
- sayAudio(name, [startTime]) 10
- sayText (txt,voice,lang,engine,[effect], [effLevel]) 11
- sayAIResponse (txt,voice,lang,engine,[botID],[effect], [effLevel]) 12
- saySilent (seconds) 13
- setPlayerVolume (level) 13
- stopSpeech () 13
- replay (ignorelimit) 14

### Scene Attributes
- setBackground (bgName) 15
- setColor (part,color) 15
- setLink (href,target) 15
- setStatus (interruptMode,progressInterval,gazeSpeed) 16
- dynamicResize (width, height) 16
- is3D () 17

### Embed Overlay Functions
- overlayOpen (mode, play) 18
- overlayClose () 18

### Navigation Flow Functions
- gotoNextScene () 19
- gotoPrevScene () 19
- gotoScene (sceneRange) 19
- loadScene (sceneIndex) 19
- loadShow (showIndex) 20
- preloadNextScene () 20
- preloadScene (sceneIndex) 20
- setNextSceneIndex (sceneRange) 21
Status Callback Functions
vh_aiResponse ()
vh_audioLoaded (audioName)
vh_ttsLoaded (text)
vh_audioProgress (percentPlayed)
vh_sceneLoaded (sceneIndex)
vh_scenePreloaded (sceneIndex)
vh_talkStarted ()
vh_talkEnded ()
vh_audioStarted ()
vh_audioEnded ()
vh_playPause (status)

Appendix A: API Examples
Appendix B: Text to Speech Languages and Voices
Appendix C: SSML Tags for Text to Speech
Structure Elements
Break
Paragraph
Sentence
Prosody Elements
Volume
Rate
Pitch
The Voice Element
Introduction

The SitePal/Avatar Studio playback environment supports an API that allows you to control character speech and other runtime attributes by making JavaScript function calls from your web page. The API enables communication between your web page and the embedded Scene or Show.

Note: SitePal users do not have access to Shows, only to Scenes. Show-specific functions which apply only to Avatar Studio are clearly marked as such.

Embedding in your web page

To use this API in your web page you must add your embed code to the page. In your account, click on the “Publish” button for the Scene or Show you wish to embed. Proceed to copy your embed code and add it to the BODY section of your page, where you wish your character to appear.

Important caveats / pitfalls to avoid -

- API function calls may not work as expected until your embedded Scene or Show is fully loaded. It is therefore advisable to implement the 'vh_sceneLoaded' callback – and not call any API function until this callback is received. For more information please review Callback Functions documentation
- For your protection certain API functions may only work when your page is loaded from a domain you authorize. Such domains are called Licensed Domains. Specifically:
  o 'sayText' and 'sayAIResponse' will only work under a licensed domain.
  o If you turn on 'secure playback' for your account, all API functions will only work under a licensed domain.
  o 'localhost' and '127.0.0.1' are always authorized and do not need to be specifically declared.
  o Add/edit your Licensed Domains in your Account Info page.

Error Handling

The API is designed to fail silently in most cases of error. This means that if your API call cannot be executed for some reason, nothing will happen. Except for a Licensed Domain infraction - no overt alert or message will appear in your page. Error messages will be written to the console.

Examples & Additional Reference Material

Before you get started – you may want to check out our comprehensive examples – please see reference links in Appendix A.

Programming for Mobile

This API operates on ‘desktop’ as well as mobile browsers (the term ‘desktop’ is used here to refer to non-mobile client side environments, such as desktop and laptop computers of all types). This means that you need not do anything special in order to support mobile browsers in your web pages when using this API. That said, there are a couple of differences between mobile and desktop that you should be aware of.

The API is fully compatible with all major browsers, both desktop and mobile – and with two exceptions will function in the same way within mobile browsers as it will on desktop browsers. One difference is with the function ‘setPlayerVolume’ – which does not have any effect in some mobile browsers – but there is no harm in making the call.
"Play-on-load": Another point you should be aware of is that on mobile browsers, any audio playback must be preceded by user interaction with the page (e.g. user clicks on a button). This restriction prevents a web page from speaking to the viewer unprompted. Attempting to do so using this API will not cause a problem – but will simply not work.

This same restriction has been introduced into desktop browsers as well, over the past 12 to 18 months. On desktop browsers though the restriction is not absolute. Some browsers will allow play-on-load for a user who has visited your page before and interacted with media on the page. The policies enacted by browsers in this regard are both evolving & undocumented.

The main takeaways here should be:

- It is ok to try to initiate speech as soon as the page loads (verify that API has loaded first!)
- You should be aware that such playback attempt may or may not be blocked, and will always be blocked on mobile.
- There should be no need for implementing special cases when calling this API on mobile browsers - the API operates the same across all browsers and devices.
Function Reference

Animation Control Functions

**followCursor(mode)**

Available for: Studio SitePal

Turn “follow cursor” to the OFF, ON IN BOX, or ON IN PAGE state. If OFF, the character’s gaze ignores cursor movement. If ON IN BOX, the character’s head and eyes follow the cursor within the embed rectangle. If ON IN PAGE, the character’s gaze follows the cursor in the entire page, including areas outside the embed rectangle.

**Arguments:**

mode Required, Numeric (0/1/2):
- 0: follow cursor is set to OFF.
- 1: follow cursor is set to ON IN BOX
- 2: follow cursor is set to ON IN PAGE.

**Example:**

```
followCursor(1)
```

**freezeToggle ()**

Available for: Studio SitePal

Toggle between the frozen and normal states. When frozen – all character movement stops. If the character is speaking, speech is paused. Unfrozen - character wakes up. If the character was previously paused in mid-speech, speech resumes from that point.

**Arguments:**

None.

**Example:**

```
freezeToggle()
```

**recenter()**

Available for: Studio SitePal

Cause the character to set its gaze to the default, centered position.

**Arguments:**

None.

**Example:**

```
recenter()
```
setGaze(degrees, duration, [amplitude])

Available for: Studio SitePal

Set the direction & amplitude of the character's head and eye movement. This call will cause the character to divert the orientation of its gaze to the specified direction, and maintain the new orientation for the specified period of time. The orientation will naturally shift towards the center (default) position when the specified time is up, or when/if the character is requested to speak. The optional amplitude parameter governs the “intensity” of the head & eye movement.

Arguments:
- degrees Required. Numeric. 0-360 (0 deg.=top, 90 deg.=right, etc.)
- amplitude Optional. Numeric. In percent. 0-100. Default = 100.

Example:
```
setGaze(90,6);
```

setFacialExpression(expression, amplitude, duration )

Available for: Studio SitePal

Note: this call is only supported for 3D characters. If called for a 2D character, it has no effect. (See related function is3D).
Set the facial expression animation for a character. setFacialExpression calls do not queue, but interrupt. If a call is made while a previous call’s duration is still in effect, the first expression transforms into the second expression immediately.

Arguments:
- expression Required, Text string
  - "None" neutral - the default expression. Other parameters are ignored.
  - "ClosedSmile" happy (closed mouth smile)
  - "OpenSmile" very happy (open mouth smile)
  - "Sad" sad
  - "Angry" angry
  - "Fear" afraid
  - "Disgust" disgusted
  - "Surprise" surprised
  - "Thinking" thinking
  - "Blush" embarrassed (blush)
  - "LeftWink" wink with left eye
  - "RightWink" wink with right eye
  - "Blink" blink with both eyes
"Scream" mouth wide open for screen.

amplitude Numeric. Range: 0, 1.0

The extent to which the expression should be applied. Using higher values then 1.0 is not blocked, and might be useful in some cases - but can lead to unexpected results. Feel free to experiment.

duration Required if expression!="None", Integer.

Time in seconds. Use -1 for indefinite duration.

Example:

setFacialExpression("OpenSmile", 0.8, 5);

// sets expression to OpenSmile for 5 seconds

// at 80% amplitude

clearExpressionList()

Available for: Studio SitePal

Note: This function is only supported for 3D characters. If called for 2D character it has no effect.

Clear all expressions.

Example:

clearExpressionList();

setIdleMovement(frequency, [amplitude])

Available for: Studio SitePal

Note: This function is fully supported for 3D characters, and only partly supported for 2D characters. If called for a 2D character, frequency is interpreted as follows: 0 – turn OFF idle movement; non-zero – turn ON idle movement. Amplitude is ignored.

Characters that are not engaged in speaking, following the cursor, or gazing (via the setGaze api) randomly look around by default. This function enables users to set the frequency and intensity of the character’s movement when not otherwise engaged.

Arguments:

frequency Optional. Numeric. The frequency with which the character performs idle time head movement. Values are 0 to 100. Default is 50.

Use 0 to turn off Idle movement.

amplitude Optional. Numeric. The distance from center the character sets its random gaze. Values are 1 to 100. Default is 50.

Example:

setIdleMovement(20,100);
setSpeechMovement(amplitude)

Available for: Studio SitePal

Note: This function is only supported for 3D characters.
Characters perform random head movements during speech. This function enables users to set
the intensity of the character’s movement when speaking or disable the movement altogether.

Arguments:

amplitude Numeric. The intensity with which the character performs head movements
while speaking. Values are 0 to 100. Default is 50.

Example:

setSpeechMovement(100);

animate(animationName)

Available for: Studio SitePal

Note: This function is only supported for FullBody characters.
The character can be directed to perform a specific animation. The list of animation names
available for use will be made available at a later time. For the time being this function is available
for internal use only.
When function is called, behavior is subject to the status of the "interrupt mode" status. Use
'setStatus' to update the interrupt mode.
- If the character is speaking, and interrupt mode is on, speech is interrupted and the character
  animates.
- If the character is speaking, and interrupt mode is off, no action is taken.
- If the character is not speaking, and interrupt mode is on, idle animation is interrupted and the
  character animates.
- If the character is not speaking, and interrupt mode is off, the character animates after current
  idle animation concludes and returns to idle state.

Arguments:

animationName The name of the animation to be invoked.

Example:

animate('wave');

Speech Functions

loadAudio(name)

Available for: Studio SitePal

Preload a specific audio track by name. Calling loadAudio in advance can reduce the loading time
when the audio is played. Calling loadAudio a second time, while audio is loading or after audio
has been loaded has no effect.
Implement the \texttt{vh\_audio\_Loaded()} event callback to be notified when the audio track is done loading. Use the \texttt{sayAudio()} function to play the audio.

**Arguments:**

- \texttt{name} Required. String. The name of the audio track from the account.

**Example:**

\begin{verbatim}
loadAudio('audioname')
\end{verbatim}

---

\begin{verbatim}
loadText(txt,voice,lang,engine,[effect], [effLevel])
\end{verbatim}

**Available for:** \checkmark Studio \checkmark SitePal

Preload a specific Text To Speech audio. Calling \texttt{loadText} in advance can reduce the loading time when the audio is played. Calling \texttt{loadText} a second time, while audio is loading or after audio has been loaded has no effect. Implement the \texttt{vh\_tts\_Loaded()} event callback to be notified when the audio track is done loading. Use the \texttt{sayText()} function to play the audio.

**Arguments:**

- \texttt{txt} Required. String - The text to speak. Text is limited to 900 characters. (225 characters in Chinese & Japanese). Longer text will be truncated.
- \texttt{voice} Required. Integer – Voice ID, as listed in Appendix B.
- \texttt{lang} Required. Integer – Language ID, as listed in Appendix B.
- \texttt{engine} Required. Integer – Voice Family ID. See languages and voices listed in Appendix B.
- \texttt{effect} Optional. Character. Audio effect – one of:
  - “D” – Duration levels: -3, -2, -1, 1, 2, 3
  - “P” – Pitch levels: -3, -2, -1, 1, 2, 3
  - “S” – Speed levels: -3, -2, -1, 1, 2, 3
  - “R” – Robotic:
    - “Bullhorn” level: 3 (note: levels 1 and 2 are deprecated)
  - “T” – Time:
    - “Echo” level: 1
    - “Reverb” level: 2
    - “Flanger” level: 3
    - “Phase” level: 4
  - “W” – Whisper levels: 1, 2, 3
- \texttt{effLevel} Optional. Integer. Effect level must be provided if effect is provided.

**Example:**

\begin{verbatim}
loadText('Hello World',1,1,1)
loadText('Hello World',1,1,1,'D',3)
\end{verbatim}

---

\begin{verbatim}
sayAudio(name, [startTime])
\end{verbatim}
Available for: Studio SitePal

Play a specific audio track by name.

**Arguments:**

- **AudioTrackName** Required. String. The logical name of the audio as specified within the account.
- **startTime** Optional. Floating. The offset, in seconds, from the beginning of the audio from which to start audio playback.

**Example:**

```
sayAudio('audio name', 1.9)
```

---

**sayText (txt, voice, lang, engine, [effect], [effLevel])**

Available for: Studio SitePal

Real-time (dynamic) Text-To-Speech (TTS).

For detailed step by step instructions, please review the [Guidelines for Using the TTS API](#).

Note: This function is available only to a TTS enabled account and will work only within a specified licensed domain for the account. Domain specific licensing is a security measure. If the account is not TTS enabled, or the Scene is used within a domain that you have not included within your licensed domains, then this call will generate an alert. To edit your licensed domains please login to your account and select Account Info.

**Arguments:**

- **txt** Required. String - The text to speak. Text is limited to 900 characters. (225 characters in Chinese & Japanese). Longer text will be truncated.
- **voice** Required. Integer – Voice ID, as listed in [Appendix B](#).
- **lang** Required. Integer – Language ID, as listed in [Appendix B](#).
- **engine** Required. Integer – Voice Family ID. See languages and voices listed in [Appendix B](#).
- **effect** Optional. Character. Audio effect – one of:
  - "D" – Duration levels: -3, -2, -1, 1, 2, 3
  - "P" – Pitch levels: -3, -2, -1, 1, 2, 3
  - "S" – Speed levels: -3, -2, -1, 1, 2, 3
  - "R" – Robotic:
    - Bullhorn level: 3 (note: levels 1 and 2 are deprecated)
  - "T" – Time:
    - Echo level: 1
    - Reverb level: 2
    - Flanger level: 3
    - Phase level: 4
  - "W" – Whisper levels: 1, 2, 3
- **effLevel** Optional. Integer. Effect level must be provided if effect is provided.
Examples:
sayText('Hello World',1,1,1)
sayText('Hello World',1,1,1,'S',-2)

sayAIResponse (txt,voice,lang,engine,[botID],[effect], [effLevel])

Available for: ☑Studio  ☑SitePal

An Artificial Intelligence knowledge base provides a real time text and audio response to a text “question”. The Audio is generated & spoken automatically via a Text To Speech engine according to the selected voice, language and engine. The response text is provided via the Event function ‘vh_aiResponse()’.

The Artificial Intelligence knowledge base is based on the extensive A.L.I.C.E. knowledge base, which includes over 23,000 data entries. Your knowledge base can be edited and customized in the AI Management Center (AIMC) – click on AIMC from the main menu within your account.

Note:
This function is available only to a TTS enabled account and will work only within a licensed domain for the account. Domain specific licensing is a security measure. If the account is not TTS enabled, or the Scene is used within a domain that you have not included within your licensed domains, then this call will generate an alert. To edit your licensed domains please login to your account and select Account Info.

Arguments:

- **txt**  Required. String - The text to speak. Text is limited to 900 characters. (225 characters in Chinese & Japanese). Longer text will be truncated.
- **voice** Required. Integer – Voice ID, as listed in Appendix B.
- **lang** Required. Integer – Language ID, as listed in Appendix B.
- **engine** Required. Integer – Voice Family ID. See languages and voices listed in Appendix B.
- **effect** Optional. Character. Audio effect – one of:
  - “D” – Duration levels: -3, -2, -1, 1, 2, 3
  - “P” – Pitch levels: -3, -2, -1, 1, 2, 3
  - “S” – Speed levels: -3, -2, -1, 1, 2, 3
  - “R” – Robotic:
    - Bullhorn level: 3 (note: levels 1 and 2 are deprecated)
  - “T” – Time:
    - Echo level: 1
    - Reverb level: 2
    - Flanger level: 3
    - Phase level: 4
  - “W” – Whisper levels: 1, 2, 3
- **effLevel** Optional. Integer. Effect level must be provided if effect is provided.

Example:
sayAIResponse('Sing me a song',2,1,2)
sayAIResponse('Sing me a song',2,1,2,'P',-1)
saySilent (seconds)

Available for: Studio  SitePal

Speech is visually simulated: No audio is downloaded, no streams are consumed and no interaction with the server is performed.
This function call may be useful when you want to call attention to the character but do not want use audio to do so. Most pertinent example is use in ad banners, where (in some cases) the use of audio may only be permitted after mouse rollover.
saySilent is always in ‘InterruptMode’ ON, meaning that any function call which invokes actual speech will interrupt simulated speech. saySilent calls cannot be queued.

Arguments:

- Seconds length of time desired for simulated speech, in seconds

Example:

saySilent(10)

setPlayerVolume (level)

Available for: Studio  SitePal

Note: This function has no effect on some mobile browsers.
Set playback volume, or mute the audio.

Arguments:

- level Required. Integer (0-10) – Default = 7.
  a value from 0 to 10; 0 is equivalent to mute, 1 is softest, 10 is loudest.

Example:

setPlayerVolume(10)

Note: Setting the volume to 0, does not stop the speech (lip movement continues) or stop the audio stream. It affects only the volume. To stop the speech, use the function stopSpeech().

stopSpeech ()

Available for: Studio  SitePal

Stop the speech of a currently speaking character. If the character is not currently speaking, stopSpeech has no effect (i.e. it does not prevent speech that has not yet begun).

Arguments:

None.

Example:

stopSpeech()
replay (ignorelimit)

Available for: Studio  SitePal

Plays or replays current Scene, from the start.
If `interruptMode` is ON, ongoing playback (if any) is interrupted, and immediately begins again.
If `interruptMode` is OFF, playback is queued. See `setStatus` to learn about `interruptMode`.

Playback limit is a Scene attribute which can be set in the Scene Options dialog. Set the `ignoreLimit` parameter to override this Scene option.

Arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ignoreLimit</td>
<td>Optional. Boolean default value is 0. If 1 then Scene ignores playback limit status.</td>
</tr>
</tbody>
</table>

Example:

```
replay()
replay(1)
```
Scene Attributes

**setBackground (bgName)**

Available for: Studio  SitePal

Background is modified for playback session in progress. Change is not persistent.

Arguments:
- bgName: Required. String. The logical name of the background as specified within the account.

Example:
```
setBackground('background name')
```

**setColor (part,color)**

Available for: Studio  SitePal

Dynamically change the color of the part specified. Colors are based on a gray offset in order to maintain the shading and detail. Therefore, the effect of a color on the specified area may not exactly match the input hex color value. The results may differ slightly from one character to another.

Arguments:
- part: Required. String. The character part to color. One of: `eyes`, `hair`, `make-up`, `mouth`, `skin`

Example:
```
setColor('eyes','0000AA')
```

**setLink (href,target)**

Available for: Studio  SitePal

Set the URL address for the link of the currently executing scene. For this call to have an effect, the scene should have a specified Trigger to activate the link.

Arguments:
- href: Required. String. The URL address for the link.
- target: Optional. String. A frame name or a window name. An optional parameter specifying the window or HTML frame that the document should load into. For `Window` you can enter the name of a specific window or choose from the following reserved target names:
  - `_self`: specifies the current frame in the current window
  - `_blank`: specifies a new window
  - `_parent`: specifies the parent of the current frame
  - `_top`: specifies the top-level frame in the current window
  If no value is specified, default value is `_blank`.

Example:
```
setLink('example.com', '_blank')
```
Example:
```
setLink('http://www.yoursite.com','_blank')
```

---

**setStatus (interruptMode,progressInterval,gazeSpeed)**

**Available for:** ✅ Studio ✅ SitePal

This function is used to set several status values which govern various aspects of playback.

**Arguments:**

- `interruptMode`  
  Required. Integer (0/1) – Default = 0.  
  If set to 0 consecutive audio playback function calls (sayText and sayAudio) are queued for consecutive playback.  
  If set to 1 current audio is interrupted when sayAudio or sayText are called.

- `progressInterval`  
  Required. Non-negative Integer – Default = 0.  
  The audio progress interval value controls progress callbacks which take place during playback. The callback function `vh_audioProgress(percent_played)` is called during playback if the value of 'progressInterval' is non-zero. The non-zero value determines the frequency of the call.  
  The value must be an integer greater than or equal to 0. When greater than 0, the callback "vh_audioProgress(percent_played)" is triggered at the frequency specified by the number (in seconds). The callback returns the percent of the current audio that has played. Callbacks will continue for all subsequent audios played once this field is set. Set back to 0 for the callbacks to cease.

- `gazeSpeed`  
  Required. Integer (0/1/2) – Default = 0.  
  controls the reaction speed of the character when responding to setGaze function calls.  
  - 0 - slow  
  - 1 - medium  
  - 2 - fast

**Example:**
```
setStatus(0,0,0)
```

---

**dynamicResize (width, height)**

**Available for:** ✅ Studio ✅ SitePal

The dimensions of the embedded Show or Scene are dynamically modified without reloading the character. This can be used to support responsive design. Change is not persistent - if the page is reloaded, the embedded Show or Scene will load as originally embedded.

Maintaining the original aspect ratio is not required. If you would like to retain the relative position of the character within the Scene frame, you should retain the aspect ratio. Otherwise, character will be re-positioned as best possible.
Tip: To control the dimensions of the embedded Show or Scene when page is initially loaded, you could set the width and height *before* the page loads, using a back end programming language such as Java or php.

<script type="text/javascript">AC_VHost_Embed(accountid, height, width, '', 1, 1, showsceneid, 0, 1, 0, '94fb29b9a4767343f36dd16fc8c0f81a', 0);</script>

Arguments:

Example:

dynamicResize(300, 200)

---

is3D ()

Available for: Studio SitePal

Is the character in the current Scene a 3D character?
Boolean function – returns true if 3D character is used, false otherwise.

Arguments:
None.

Example:

is3D()
Embed Overlay Functions

The following functions apply only to Scenes and Shows that are embedded as an overlay on top of the page. If any of these functions is called for a Scene or Show which is embedded IN the page, the function call will have no effect.

**overlayOpen (mode, play)**

Available for: Studio SitePal

Scene or Show is opened, or toggled between minimize and maximize display mode. If mode is ‘max’ then play parameter governs the playback behavior of the Scene/Show when opened.

**Arguments:**

- **mode**
  - ‘max’ – open Scene/Show in maximized mode or toggle to maximize mode. If the Scene is already maximized, this call has no effect.
  - ‘min’ – open Scene/Show in minimized mode or toggle to minimize mode. The effect is equivalent to a click on the minimize button. If the Scene is already minimized, this call has no effect.

- **play**
  - Optional. Integer (0/1/2) – Default = 2.
  - Relevant only for ‘max’ mode. If mode is ‘min’ then parameter is ignored.
  - If set to 0, playback does not start. Settings ignored.
  - If set to 1, playback immediately starts. Settings ignored.
  - If set to 2, playback may start depending on the values of the “playback limit” and “play on load” settings.

**Examples:**

- overlayOpen('max',1)
- overlayOpen('max')
- overlayOpen('min')

---

**overlayClose ()**

Available for: Studio SitePal

Scene or Show is closed. The effect is equivalent to a click on the close button. If Scene is already closed, this call has no effect.

**Arguments:**

- None.

**Example:**

- overlayClose()
Navigation Flow Functions

gotoNextScene ()

Available for: ⌘ Studio  □ SitePal

The current Scene is interrupted, and the next Scene (according to the preset show flow) immediately begins. This has the same effect as pressing the player’s ‘Next’ button.

Arguments:
None
Example:
gotoNextScene()  

gotoPrevScene ()

Available for: ⌘ Studio  □ SitePal

The current Scene is interrupted, and the previous Scene immediately begins. This has the same effect as pressing the player’s ‘Previous’ button.

Arguments:
None
Example:
gotoPrevScene()  

gotoScene (sceneRange)

Available for: ⌘ Studio  □ SitePal

The current Scene is interrupted, and the specified scene immediately begins.

Arguments:
sceneRange Required. String
Indicates the next Scene to follow the currently playing Scene. Can be either:
- Index of specific Scene
- Range of consecutive scenes to randomly choose from. The format is hyphen delimited.

Example

gotoScene('3') – goto a specific Scene  
gotoScene('4-7') – goto a randomly selected scene from the set: 4,5,6,7

loadScene (sceneIndex)
SitePal ONLY - The current Scene is interrupted, and the specified Scene is loaded instead.

**Arguments:**

- `sceneIndex` Required. Integer
  Indicates the Scene to load.

**Example**

```
loadScene(3)  // load Scene #3
```

---

**loadShow (showIndex)**

**Available for:** Studio SitePal

Studio ONLY - The current Show is interrupted, and the specified Show is loaded instead.

**Arguments:**

- `showIndex` Required. Integer
  Indicates the Show to load.

**Example**

```
loadShow(3)  // load Show #3
```

---

**preloadNextScene ()**

**Available for:** Studio SitePal

Preloads the assets of the next scene in a show. Upon successful preloading of a scene the callback `vh_scenePreloaded()` will be invoked. If there is no next scene the call is ignored. Subsequent calls to this function or `preloadScene` while a scene is loading will be ignored.

**Arguments:**

- None

**Example:**

```
preloadNextScene()
```

---

**preloadScene (sceneIndex)**

**Available for:** Studio SitePal

Preloads the assets of the specified scene. Upon successful preloading of a scene the callback `vh_scenePreloaded()` will be invoked. If there is no scene with the specified index number the call is ignored. Subsequent calls to this function or `preloadNextScene` while a scene is loading will be ignored.

**Arguments:**

- `sceneNumber` Required. Integer
  The index of the scene to preload
Example

`preloadScene(3)` – preload a specific Scene

---

**setNextSceneIndex (sceneRange)**

**Available for:** ☑️Studio  ☐SitePal

Set the next Scene in the playback order. Actual transition to the specified Scene will not take place until a user clicks on the player’s Next button or the function `gotoNextScene()` is called.

**Arguments:**

- `sceneRange` Required. String. Indicates the next scene to follow the currently playing scene. Can be either:
  - Index of specific scene
  - Range of scenes that specify a set of scene indexes. In this case one of the specified set is selected at random. The format is comma delimited, where consecutive ranges can be specified by a hyphen.

**Examples:**

- `setNextSceneIndex('3')` – the next scene is a specific scene
- `setNextSceneIndex('1,4-6,12')` – next scene is a random selection from the set: 1,4,5,6,12
Status Callback Functions

Status Callback Functions (SCFs) can help improve coordination between the embedded Scene or Show and your page / application.

Events during playback trigger calls to specific JavaScript functions in your page, if such functions exist. To take advantage of these calls you must add the appropriate JavaScript functions to your page. Note that you do not need to add callback functions which you do not intend to use.

API functions work only after Scene or Show has completed loading
Keep in mind that certain aspects of the API may not function in a predictable manner until the "vh_sceneLoaded" status callback has been called/dispatched. It is therefore advisable to always implement the "vh_sceneLoaded" callback & check that it has been called before calling any API function.

vh_aiResponse ()
Available for: ⚫Studio  ⚫SitePal

Triggered when an AI Response is returned, this call returns the text that is generated by the AI knowledge base in response to the function call 'sayAIResponse'.

Arguments:
- response_text  Response text

Example -
function vh_aiResponse(response_text){
}

vh_audioLoaded (audioName)
Available for: ⚫Studio  ⚫SitePal

Triggered when an audio preload is done, and returns the name of the audio that was provided as input to 'loadAudio()'.

Arguments:
- audio_name  Loaded audio name

Example -
function vh_audioLoaded(audio_name){
    sayAudio(audio_name);
}

vh_ttsLoaded (text)
Available for: ⚫Studio  ⚫SitePal

Triggered when a Text-To-Speech audio preload is done and returns the text that was provided as input to 'loadText()'.

Arguments:

audio_text  Loaded audio text

Example -

```javascript
function vh_ttsLoaded(audio_text) {
    sayText(audio_text,2,1,1);
}
```

---

**vh_audioProgress (percentPlayed)**

Available for: ✅Studio  ✅SitePal

Called during playback, if and only if the 'progressInterval' status is set.

vh_audioProgress is repeatedly called at regular intervals during playback. The intervals are
determined according to the value of the 'progressInterval' status. See 'setStatus' API call for
information about how to set this status.

This callback can be used to enable synchronization between playback and other events taking
place at the same time. For example: highliting text segments, or visual elements on the page in
coordination with speech playback.

Arguments

percentPlayed  A value between 0 and 100 which indicated the proportion of audio
already played.

Example -

```javascript
function vh_audioProgress(percentPlayed) {
}
```

---

**vh_sceneLoaded (sceneIndex)**

Available for: ✅Studio  ✅SitePal

Triggered when the Scene is fully loaded & displayed, just before the audio starts playing. Use this
callback to verify Scene is ready to accept API calls.

Note: sceneIndex parameter is only relevant to Studio accounts.

Arguments:

sceneIndex  For Studio accounts: The index of the loaded Scene in the Show.
Undefined for SitePal accounts.

Example -

```javascript
function vh_sceneLoaded(sceneIndex) {
    alert("the scene has started");
}
```

---

**vh_scenePreloaded (sceneIndex)**

Available for: ✅Studio  ✅SitePal
Triggered after a successful call to preloadScene or preloadNextScene. The assets of the scene are loaded to memory. Subsequent display of the specified scene should be immediate.

Arguments:

sceneIndex For Studio accounts: The index of the loaded Scene in the Show.

Example -

```javascript
function vh_scenePreloaded(sceneIndex){
    alert("the scene is preloaded. index: "+ sceneIndex);
}
```

---

**vh_talkStarted ()**

Available for: Studio SitePal

Triggered when the character starts talking. When several audios are played in sequence, this callback will be dispatched at the start of the sequence.

Example -

```javascript
function vh_talkStarted(){
}
```

---

**vh_talkEnded ()**

Available for: Studio SitePal

Triggered when the character is done talking. When several audios are played in sequence, this callback will be dispatched at the end of the sequence.

Example -

```javascript
function vh_talkEnded(){
}
```

---

**vh_audioStarted ()**

Available for: Studio SitePal

Triggered when audio playback begins. Unlike vh_talkStarted() this event is fired for each audio playback in a sequence.

Example -

```javascript
function vh_audioStarted(){
}
```

---

**vh_audioEnded ()**

Available for: Studio SitePal
Triggered when the an audio ends. Unlike talkEnded() this event is fired for each audio in a sequence.

**Example** -
```javascript
function vh_audioEnded(){
    
}
```

---

**vh_playPause ( status )**

**Available for:** Studio  SitePal

Triggered when the play/pause button is pressed. This enables synchronization

**Arguments:**
- **status**
  - 0=paused; 1=playing.

**Example - JavaScript**
```javascript
function vh_playPause(status){
    alert("play/pause button pressed. status: " + status);
}
```
Appendix A: API Examples

We’ve put together a collection of technical examples that demonstrate how to use every one of our API functions and callbacks.

As you review these examples, feel free to view their source code, and to copy that source code for use in your own test pages as you get started. You will of course need to replace the embed code in the example with your own embed code from your own account - if you want to have control over character & speech.

Note: a common problem when getting started is not setting up you own "licensed domain" - which is required for certain functions to work. This feature is provided for your security. To enter your domain(s) select "Account Info" from the main menu bar. You can add your domain(s) at the bottom of the page.

Our API support examples can be found here:

Using the Client API - Technical Examples
Using the Client API - Advanced Examples

Additional reference material can be found in our support section.
Appendix B: Text to Speech Languages and Voices

The following tables list all Voice Family IDs, Language IDs and Voice IDs supported by the sayText & sayAIResponse API calls.

<table>
<thead>
<tr>
<th>Language</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>27</td>
</tr>
<tr>
<td>Basque</td>
<td>22</td>
</tr>
<tr>
<td>Catalan</td>
<td>5</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
</tr>
<tr>
<td>Czech</td>
<td>18</td>
</tr>
<tr>
<td>Danish</td>
<td>19</td>
</tr>
<tr>
<td>Dutch</td>
<td>11</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>Esperanto</td>
<td>31</td>
</tr>
<tr>
<td>Finnish</td>
<td>23</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>Galician</td>
<td>15</td>
</tr>
<tr>
<td>German</td>
<td>3</td>
</tr>
<tr>
<td>Greek</td>
<td>8</td>
</tr>
<tr>
<td>Hindi</td>
<td>24</td>
</tr>
<tr>
<td>Hungarian</td>
<td>29</td>
</tr>
<tr>
<td>Indonesian</td>
<td>28</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
</tr>
<tr>
<td>Norwegian</td>
<td>20</td>
</tr>
<tr>
<td>Polish</td>
<td>14</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
</tr>
<tr>
<td>Romanian</td>
<td>30</td>
</tr>
<tr>
<td>Russian</td>
<td>21</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Swedish</td>
<td>9</td>
</tr>
<tr>
<td>Thai</td>
<td>26</td>
</tr>
<tr>
<td>Turkish</td>
<td>16</td>
</tr>
<tr>
<td>Language</td>
<td>Lang. ID</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
</tr>
<tr>
<td>German</td>
<td>3</td>
</tr>
<tr>
<td>German</td>
<td>3</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
</tr>
<tr>
<td>Catalan</td>
<td>5</td>
</tr>
<tr>
<td>Catalan</td>
<td>5</td>
</tr>
<tr>
<td>Catalan</td>
<td>5</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
</tr>
</tbody>
</table>
Expressive Cues are a set of special tags which you may use in your text to specify distinct non-verbal expressions, such as laughing, crying, sighing, coughing, etc. Expressive Cues can be used only with a subset of Loquendo voices, as indicated above. For a complete list of Expressive Cue tags see Appendix D.

TTS Engine ID = 3

<table>
<thead>
<tr>
<th>Language</th>
<th>Lang. ID</th>
<th>Voice Name</th>
<th>Voice ID</th>
<th>Gender</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>1</td>
<td>Kate</td>
<td>1</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Paul</td>
<td>2</td>
<td>M</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Julie</td>
<td>3</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Bridget</td>
<td>4</td>
<td>F</td>
<td>UK</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Hugh</td>
<td>5</td>
<td>M</td>
<td>UK</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Ashley</td>
<td>6</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>Language</td>
<td>Group</td>
<td>Name</td>
<td>Age</td>
<td>Gender</td>
<td>Nationality</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>----------</td>
<td>-----</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>James</td>
<td>7</td>
<td>M</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Beth</td>
<td>8</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Violeta</td>
<td>1</td>
<td>F</td>
<td>Mexican</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Francisco</td>
<td>2</td>
<td>M</td>
<td>Mexican</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Gloria</td>
<td>3</td>
<td>F</td>
<td>Mexican</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Lola</td>
<td>4</td>
<td>F</td>
<td>Castilian</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Manuel</td>
<td>5</td>
<td>M</td>
<td>Castilian</td>
</tr>
<tr>
<td>German</td>
<td>3</td>
<td>Lena</td>
<td>1</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>German</td>
<td>3</td>
<td>Tim</td>
<td>2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Chloe</td>
<td>1</td>
<td>F</td>
<td>Canadian</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Leo</td>
<td>2</td>
<td>M</td>
<td>Canadian</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Roxane</td>
<td>3</td>
<td>F</td>
<td>European</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Louis</td>
<td>4</td>
<td>M</td>
<td>European</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
<td>Helena</td>
<td>1</td>
<td>F</td>
<td>Brazilian</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
<td>Rafael</td>
<td>2</td>
<td>M</td>
<td>Brazilian</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
<td>Elisa</td>
<td>1</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
<td>Roberto</td>
<td>2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Lily</td>
<td>1</td>
<td>F</td>
<td>Mandarin</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Hui</td>
<td>3</td>
<td>F</td>
<td>Mandarin</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Liang</td>
<td>4</td>
<td>M</td>
<td>Mandarin</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Qiang</td>
<td>5</td>
<td>M</td>
<td>Mandarin</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Kaho</td>
<td>6</td>
<td>M</td>
<td>Cantonese</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Kayan</td>
<td>7</td>
<td>F</td>
<td>Cantonese</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Yafrang</td>
<td>8</td>
<td>F</td>
<td>Taiwanese</td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Show</td>
<td>2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Misaki</td>
<td>3</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Sayaka</td>
<td>4</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Hikari</td>
<td>5</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Haruka</td>
<td>6</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Ryo</td>
<td>7</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Takeru</td>
<td>8</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Yumi</td>
<td>1</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Junwoo</td>
<td>2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Hyeryun</td>
<td>4</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Jimin</td>
<td>5</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Sena</td>
<td>6</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Dayoung</td>
<td>7</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Hayuna</td>
<td>8</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Yura</td>
<td>9</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Jihun</td>
<td>10</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Thai</td>
<td>26</td>
<td>Sarawut</td>
<td>1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Thai</td>
<td>26</td>
<td>Somsri</td>
<td>2</td>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>
TTS Engine ID = 4

<table>
<thead>
<tr>
<th>Language</th>
<th>Lang. ID</th>
<th>Voice Name</th>
<th>Voice ID</th>
<th>Gender</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>1</td>
<td>Jill</td>
<td>2</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Tom</td>
<td>3</td>
<td>M</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Karen</td>
<td>4</td>
<td>F</td>
<td>Australian</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Daniel</td>
<td>5</td>
<td>M</td>
<td>UK</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Serena</td>
<td>7</td>
<td>F</td>
<td>UK</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Moira</td>
<td>8</td>
<td>F</td>
<td>Irish</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Sangeeta</td>
<td>9</td>
<td>F</td>
<td>Indian</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Lee</td>
<td>10</td>
<td>M</td>
<td>Australian</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Samantha</td>
<td>11</td>
<td>F</td>
<td>US</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Fiona</td>
<td>12</td>
<td>F</td>
<td>Scottish</td>
</tr>
<tr>
<td>English</td>
<td>1</td>
<td>Tessa</td>
<td>13</td>
<td>F</td>
<td>South African</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Duardo</td>
<td>1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Monica</td>
<td>3</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Paulina</td>
<td>4</td>
<td>F</td>
<td>Mexican</td>
</tr>
<tr>
<td>Spanish</td>
<td>2</td>
<td>Javier</td>
<td>5</td>
<td>M</td>
<td>Mexican</td>
</tr>
<tr>
<td>German</td>
<td>3</td>
<td>Steffi</td>
<td>1</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>German</td>
<td>3</td>
<td>Yannick</td>
<td>2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>German</td>
<td>3</td>
<td>Anna</td>
<td>3</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Felix</td>
<td>1</td>
<td>M</td>
<td>Canadian</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Julie</td>
<td>2</td>
<td>F</td>
<td>Canadian</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Sebastien</td>
<td>3</td>
<td>M</td>
<td>European</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Virginie</td>
<td>4</td>
<td>F</td>
<td>European</td>
</tr>
<tr>
<td>French</td>
<td>4</td>
<td>Thomas</td>
<td>5</td>
<td>M</td>
<td>European</td>
</tr>
<tr>
<td>Catalan</td>
<td>5</td>
<td>Nuria</td>
<td>1</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
<td>Raquel</td>
<td>2</td>
<td>F</td>
<td>Brazilian</td>
</tr>
<tr>
<td>Portuguese</td>
<td>6</td>
<td>Joana</td>
<td>3</td>
<td>F</td>
<td>European</td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
<td>Paolo</td>
<td>1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td>7</td>
<td>Silvia</td>
<td>2</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Greek</td>
<td>8</td>
<td>Alexandros</td>
<td>1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Swedish</td>
<td>9</td>
<td>Alva</td>
<td>1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Swedish</td>
<td>9</td>
<td>Oskar</td>
<td>3</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Sin-Ji</td>
<td>1</td>
<td>F</td>
<td>Cantonese</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Ya-Ling</td>
<td>2</td>
<td>F</td>
<td>Taiwanese Mandarin</td>
</tr>
<tr>
<td>Chinese</td>
<td>10</td>
<td>Ting-Ting</td>
<td>4</td>
<td>F</td>
<td>Mandarin</td>
</tr>
<tr>
<td>Dutch</td>
<td>11</td>
<td>Ellen</td>
<td>1</td>
<td>F</td>
<td>Belgian</td>
</tr>
<tr>
<td>Dutch</td>
<td>11</td>
<td>Clair</td>
<td>2</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>11</td>
<td>Laura</td>
<td>3</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>Number</td>
<td>Name</td>
<td>Gender</td>
<td>Age</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>11</td>
<td>Xander</td>
<td>M</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td>12</td>
<td>Kyoko</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Korean</td>
<td>13</td>
<td>Narae</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Polish</td>
<td>14</td>
<td>Agata</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Turkish</td>
<td>16</td>
<td>Aylin</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Czech</td>
<td>18</td>
<td>Zuzana</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Danish</td>
<td>19</td>
<td>Ida</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Norwegian</td>
<td>20</td>
<td>Stine</td>
<td>F</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Russian</td>
<td>21</td>
<td>Milena</td>
<td>F</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Basque</td>
<td>22</td>
<td>Arantxa</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Finnish</td>
<td>23</td>
<td>Mikko</td>
<td>M</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Hindi</td>
<td>24</td>
<td>Lekha</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Thai</td>
<td>26</td>
<td>Narisa</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Arabic</td>
<td>27</td>
<td>Maged</td>
<td>M</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indonesian</td>
<td>28</td>
<td>Damayanti</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Hungarian</td>
<td>29</td>
<td>Eszter</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Romanian</td>
<td>30</td>
<td>Simona</td>
<td>F</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Appendix C: SSML Tags for Text to Speech

The Speech Synthesis Markup Language (SSML) is an XML based language used to represent instructions to Text-To-Speech engines when processing input text. SSML Tags are inserted within the actual text to be processed, and are subsequently interpreted by the TTS engine to affect the manner in which voice audio is generated.

Using SSML Tags in your input text is not necessary, but allows you to achieve more precise control over the manner in which the text is spoken.

The syntax for SSML is an emerging standard, governed by the W3C. The specification for SSML 1.0 has only recently been finalized (see SSML Specification for more information). It should therefore come as no surprise that support for SSML is not yet fully or uniformly implemented.

We have reviewed what we consider the most relevant tags, and verified their implementation and functionality within the available TTS Engines. The following list summarizes our findings. For each of the listed tags, we note the support status per each of the TTS Engines (a.k.a Voice Family) #2 and #3 (Loquendo and Neospeech). Note that where specific languages are mentioned, this means that other languages for that TTS Engine have been reviewed and are not supported. This list will be updated from time to time.

**Note:** TTS Engine #4 does not support SSML tags. Please select only voices from Engines #2 and #3 for use with SSML.

Additional SSML tags, which are part of the SSML Specification but not listed here, might be useful for your purposes. Please feel free to experiment and come to your own conclusion regarding the suitability of unlisted tags.

*Note that SSML tag interpretation is case sensitive, and the case of opening and closing tags must match!*

**Examples:**

```
<Prosody volume="loud"> very loud </prosody> Wrong
<prosody volume="loud"> very loud </prosody> Correct
<Prosody volume="loud"> very loud </Prosody> Correct
```

Structure Elements

**Break**

The Break tag instructs the TTS engine to insert a pause in the synthesized text in one of three ways.

**Loquendo:** Partial support.

Loquendo does not support the "size" attribute of the <break /> element, only the "time" attribute.

**Neospeech:** Supported

**Syntax:** <BREAK/>

**Example:** Time for a pause <Break/> Okay, keep going.

Inserts a brief break after the word "pause".
Syntax: `<BREAK Size="none | small | medium | large"/>`
Example: No time for a pause `<BREAK Size="none"/>` Keep going.
   Inserts no break after the word "pause".
Example: Time for a pause `<BREAK Size="medium"/>` Okay, keep going.
   Inserts a brief silence, the equivalent of the silence following a sentence, after the word "pause".
Example: Time for a pause `<BREAK Size="large"/>` Keep going.
   Inserts only the default break after the word "pause".
Example: Time for a pause `<BREAK Size="medium"/>` Okay, keep going.
   Inserts the equivalent of a paragraph break of silence after the word "pause".

Syntax: `<BREAK time=" duration "/>`
Example: Break for 100 milliseconds `<BREAK time="100ms"/>` Okay, keep going.
   Inserts 100 milliseconds of silence after the word "milliseconds".
Example: Break for 3 seconds `<BREAK time="3s"/>` Okay, keep going.
   Inserts 3 seconds of silence after the word "seconds".

**Paragraph**

The **PARAGRAPH** tag tells the TTS engine to change the prosody to reflect the end of a paragraph, regardless of the surrounding punctuation.

Syntax: `<PARAGRAPH> text </PARAGRAPH>`
   `<P> text </P>`
Loquendo: Supported
Neospeech: Supported

Example: `<Paragraph> This example has only one sentence in the paragraph </Paragraph>`
Example: `<P> The paragraph tag can be abbreviated as just the letter P. </P>`
   The TTS engine changes the prosody to reflect the paragraph boundaries.

**Sentence**

The **SENTENCE** tag tells the TTS engine to change the prosody to reflect the end of a sentence, regardless of the surrounding punctuation.

Syntax: `<SENTENCE> text </SENTENCE>`
   `<S> text </S>`
Loquendo: Supported
Neospeech: Supported

Example: `<Sentence> This text is a sentence. </Sentence>`
Example: `<S> The sentence tag can be abbreviated as just the letter S. </S>`
   The TTS engine changes the prosody to reflect the sentence boundaries.
Prosody Elements

Volume

The Volume attribute of the Prosody tag allows the application to change the volume of the TTS voice. Note that this does not change the volume of the output device, but it does raise or lower the volume of the text spoken within the context of the tag.

Syntax: `<PROSODY VOLUME=" level "> text </PROSODY>`

where level is a value from 0.0 to 200.0. A value of 100 is the voice’s default volume, a value of 0 changes the volume to 0 and a value of 200 doubles the volume. The volume changes linearly.

Syntax: `<PROSODY VOLUME=" silent | soft | medium | loud "> text </PROSODY>`

Sets the absolute volume to the specified level.

Loquendo: Supported
Neospeech: Supported

Example: This is the default volume
   `<prosody volume="silent"> silence </prosody>`
   `<prosody volume="soft"> Now I'm whispering </prosody>`
   `<prosody volume="120"> a little louder </prosody>`
   `<prosody volume="medium"> medium volume</prosody>`
   `<prosody volume="loud"> very loud </prosody>`

Rate

The RATE attribute of the Prosody tag changes the rate at which the text is spoken. You can specify either the absolute rate or a relative change in the current speaking rate.

Syntax: `<PROSODY RATE="x-fast | fast | medium | slow | x-slow | default"> text </PROSODY>`

Syntax: `<PROSODY RATE="relativeChange"> text </PROSODY>`

changes the speaking rate which is expressed in Words Per Minute (WPM) or in percentage terms. relativeChange is a floating point number that is added to or subtracted from the current rate. A “+” or “-” sign must precede the number. If a percent sign follows then the change is interpreted as a percentage change..

Loquendo: Supported
Neospeech: Supported

Example: This is the default speed
   `<prosody rate="slow"> this is speaking slowly </prosody>`
   `<prosody rate="fast"> this is speaking fast </prosody>`
   back to slow
   `<prosody>`
   back to the default rate

Example: This is the default speed
   `<prosody rate="-50%"> this is 50% slower </prosody>`
   back to 50% slower
   `<prosody>`
   back to the default rate

Pitch
The **PITCH** attribute of the **Prosody** tag changes the pitch at which the text is spoken. You can specify either the absolute pitch or a relative change in the current speaking pitch.

**Syntax:**

```
<PROSODY PITCH="x-high | high | medium | low | x-low | default"> text </PROSODY>
```

**Syntax:**

```
<PROSODY PITCH="relativeChange"> text </PROSODY>
```

*relativeChange* is an floating point number, expressed as a percentage that is added to or subtracted from to the current pitch. A "+" or "-" sign must precede the number, and the percent sign must follow.

**Loquendo:** Supported  
**Neospeech:** Supported  
**Example:** `<prosody pitch="+12.5%"> Higher pitch sentence </prosody>`  
**Example:** `<prosody pitch="high"> High pitch sentence </prosody>`

---

### The Voice Element

The **Voice** tag enables control the voice of the TTS speaker from the input text. You can use this feature to change voices, e.g. you might use different voices to speak different sections of an email message or carry on a conversation between two different voices. You can even use different languages within the same sentence.

**Note:** this can only work when switching voices within the same voice family.

Select a voice by specifying one of the following attributes:

- **Gender**, **Name**.

It is best to specify the speaker by **Name**, in which case the **Gender** attribute is unnecessary.

**Syntax:**

```
<VOICE Gender="male | female | neutral" Name= voicename>
</VOICE>
```

**Loquendo:** Supported by some voices  
**Neospeech:** Supported  
**Example:** `<voice name="Bridget"> This is Bridget, <Voice Name="Violeta"> Hola, me llamo Violeta, </Voice> and this is Bridget again. </voice>`

This string is pronounced in Bridget's voice "This is Bridget", then in Violeta's voice in Spanish, "Hola, me llamo Violeta", then in Bridget's voice, "This is Bridget again".