Using control sequences

A control sequence is a piece of text that is not to be read out, but instead affects how other text is spoken, or performs a task. In this way the user can alter the way in which a text will be read, and acquire full control over the pronunciation of the input text. Control sequences can also be used to insert bookmarks in the text.

Vocalizer supports both W3C SSML and native control sequences. While the remainder of this section focuses on native control sequences, there are SSML equivalents for all the controls described here, so this provides a good introduction even for SSML users.

Vocalizer supports a number of control sequences which are covered in the following sections:

- Inserting a pause
- Guiding text normalization
- Inserting a bookmark
- Changing the speaking rate
- Changing the volume
- Setting the end-of-sentence pause duration
- Setting the spelling pause duration
- Controlling end-of-sentence detection
- Controlling the read mode
- Changing the voice
- Labeling text for language identification
- Indicating a paragraph break
- Identifying a word to accent within a sentence
- Resetting control sequences to the default
All the control sequences follow this general syntax notation:

\(<ESC>\ \<parameter> = <value> \)

- \(<ESC>\) normally represents the escape character “\(\text{x1B}\)” (decimal 27) that generates the ASCII character 27 (Hex 1B).

The use of the escape character “\(\text{x1B}\)” can be supplemented with a user-defined control sequence using the escape_sequence parameter in the Vocalizer configuration file. This is particularly helpful for using native control sequences within SSML or other XML documents because the escape character “\(\text{x1B}\)” is not permitted within XML documents.

- \(<parameter>\) is the name of the control parameter that the control sequence affects

- \(<value>\) is the value you want to assign to the control parameter

A value that is set with a control sequence remains active until another control sequence sets a new value, or until the end of the input text. Note that control sequences should be located outside of words; when entered inside a word they will break the word into two words.

**Inserting a pause**

This control sequence inserts a pause of a specified duration at a specific location in the text.

For example:

His name is \(<ESC>\text{pause=300}\) Michael.

The control sequence \(<ESC>\text{pause=dur_ms}\) inserts a pause of \text{dur_ms} milliseconds; the supported range is 1–65535 msec.

**Guiding text normalization**

The control sequence \(<ESC>\text{tn=<type>}\) is used to guide the text normalization processing step. For details on all the supported text normalization types and supported input formats, see the Language Supplement for each language. If the text within the control sequence doesn’t match a supported input format, Vocalizer spells the content. While Vocalizer supports a broad range of input formats, application developers should still be careful about the text input format, and should always specify \(<ESC>\text{tn=normal}\) to specify the end of the text block with that specialized text normalization format.
Some common text normalization (TN) types are listed below. These types can also be used in SSML via `<say-as>`, where the TN type is specified using the interpret-as attribute, as indicated in these examples.

<table>
<thead>
<tr>
<th>TN Type</th>
<th>USE</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Address reading</td>
<td><code>&lt;ESC&gt;\tn=address\nApt. 7-12, 28 N. Whitney St., Saint Augustine Beach, FL 32084-6715&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;address&quot;&gt;Apt. 7-12, 28 N. Whitney St., Saint Augustine Beach, FL 32084-6715&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>alphanumeric</td>
<td>Alias of spell:alphanumeric</td>
<td></td>
</tr>
<tr>
<td>boolean</td>
<td>Boolean reading</td>
<td><code>&lt;ESC&gt;\tn=boolean\ntrue&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;boolean&quot;&gt;true&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>cardinal</td>
<td>Alias of number</td>
<td></td>
</tr>
<tr>
<td>characters</td>
<td>Alias of spell:alphanumeric</td>
<td></td>
</tr>
<tr>
<td>currency</td>
<td>Currency reading</td>
<td><code>&lt;ESC&gt;\tn=currency\n12USD&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;currency&quot;&gt;12USD&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>date</td>
<td>Date reading</td>
<td><code>&lt;ESC&gt;\tn=date\n12/3/1995&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;date&quot;&gt;12/3/1995&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>digits</td>
<td>Alias of spell:alphanumeric</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>Proper name reading</td>
<td><code>&lt;ESC&gt;\tn=name\nCare Telecom Ltd&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;name&quot;&gt;Care Telecom Ltd&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>number</td>
<td>Number reading</td>
<td><code>&lt;ESC&gt;\tn=number\n1343455&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;number&quot;&gt;1343455&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>ordinal</td>
<td>Ordinal number</td>
<td><code>&lt;ESC&gt;\tn=ordinal\n12th&lt;ESC&gt;\tn=normal\n</code></td>
</tr>
<tr>
<td>Category</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>phone</td>
<td>Telephone number reading</td>
<td><code>&lt;ESC&gt;\tn=xml:phone\1-800-688-0068&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;phone&quot;&gt;1-800-688-0068&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>raw</td>
<td>Block expansions of abbreviations and acronyms</td>
<td><code>&lt;ESC&gt;\tn=raw\app.&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;raw&quot;&gt;app.&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>sms</td>
<td>Short message service (SMS) reading</td>
<td><code>&lt;ESC&gt;\tn=sms\CU (:&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;sms&quot;&gt;CU (:&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>spell</td>
<td>Alias of spell:strict</td>
<td></td>
</tr>
<tr>
<td>spell:alphanumeric</td>
<td>Spell alphanumeric characters except for white space and punctuation</td>
<td><code>&lt;ESC&gt;\tn=spell:alphanumeric\a34y&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;spell&quot; format=&quot;alphanumeric&quot;&gt;a34y&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>spell:strict</td>
<td>Spell all characters including white space and punctuation</td>
<td><code>&lt;ESC&gt;\tn=spell:strict\ a34y-347&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;spell&quot; format=&quot;strict&quot;&gt;a34y-347&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>state</td>
<td>(Not all languages) State, city, and province names and abbreviations reading</td>
<td><code>&lt;ESC&gt;\tn=state\ FL&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;state&quot;&gt;FL&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>streetname</td>
<td>(Not all languages) Street name and abbreviation</td>
<td><code>&lt;ESC&gt;\tn=streetname\ Emerson Rd.&lt;ESC&gt;\tn=normal\&lt;say-as interpret-as=&quot;streetname&quot;&gt;Emerson</code></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>streetnumber reading</td>
<td>(Not all languages) Street number reading</td>
<td><code>&lt;ESC&gt;\tn=streetnumber\n11001-11010&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;streetnumber&quot;&gt;11001-11010&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>telephone</td>
<td>Alias of phone</td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>Time of day reading</td>
<td><code>&lt;ESC&gt;\tn=time\n10:00&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;time&quot;&gt;10:00&lt;/say-as&gt;</code></td>
</tr>
<tr>
<td>zip</td>
<td>(US English only) ZIP codes</td>
<td><code>&lt;ESC&gt;\tn=zip\n01803&lt;ESC&gt;\tn=normal\n&lt;say-as interpret-as=&quot;vxml:time&quot;&gt;01803&lt;/say-as&gt;</code></td>
</tr>
</tbody>
</table>

**address:** This type provides optimal reading for complete postal addresses. The address portion (name portion) should not be included in the address to avoid undesired expansions of name specific abbreviations. Instead, the name portion should be included in a separate `<ESC>\tn=name\n` section prior to the `<ESC>\tn=address\n`.

**normal:** The end of a text fragment that should be normalized in a special way is tagged with `<ESC>\tn=normal\n`.

Some examples:

`<ESC>\tn=address\n244 Perryn Rd Ithaca, NY 14850<ESC>\tn=normal\n`  
That’s spelled `<ESC>\tn=spell\nlthaca<ESC>\tn=normal\n`  
`<ESC>\tn=sms\nCarlo, can u give me a lift 2 Helena's house 2nite? David <ESC>\tn=normal\n`  

**raw:** This type provides a more literal reading of the text, such as blocking an undesired abbreviation expansion. `<ESC>\tn=raw\n` operates on the abbreviations and acronyms as listed in each Language Supplement, but may impact the surrounding text as well. For example, the `<ESC>\tn=raw\n` in the following text would also block recognition of “12/ 6” as a date:

`<ESC>\tn=raw\nWed. <ESC>\tn=normal\n12/6`  

**spell:** Vocalizer supports two TN types for spelling text: `<ESC>\tn=spell:alphanumeric\n` and `<ESC>\tn=spell:strict\n`.

- `<ESC>\tn=spell:strict\n` has the following behavior:
• All characters are spelled, including white space, special characters, and punctuation marks.

• Characters with diacritics are pronounced as such. (For example, ü is spoken as “u with acute accent.”)

• “Upper case” is pronounced for upper case letters. (For example, “Abc” is spoken as “Upper case a, b, c.”)

• `<ESC>\tn=spell:alphanumeric\` has the following behavior:

  • All alphabetic and numeric characters are spelled. This excludes white space, special characters, and punctuation marks.

  • Characters with diacritics are pronounced as such. (For example, ü is spoken as “u with acute accent.”)

  • “Upper case” is pronounced for upper case letters. (For example, “Abc” is spoken as “Upper case a, b, c.”)

### Inserting a bookmark

The control sequence `<ESC>\mrk=name\` marks the position where it appears in the input text, and has Vocalizer track this position throughout the TTS conversion. A Vocalizer bookmark (name) can be any text sequence. After synthesis it delivers a bookmark marker that refers to this position in the input text and the corresponding position in the audio output.

The use of this control sequence does not affect the speech output process. Some examples:

This bookmark `<ESC>\mrk=bookmark 1\` marks a reference point. Another `<ESC>\mrk=bookmark 2\` does the same.

### Changing the speaking rate

The control sequence `<ESC>\rate=level\` sets the speaking rate to the specified value, where level is between 50 (half the default rate) and 400 (four times the default rate), where 100 is the default speaking rate.

Example:

I can `<ESC>\rate=150\` speed up the rate `<ESC>\rate=75\` or slow it down.
Changing the volume

The control sequence `<ESC>\vol=level\` sets the volume to the specified level, where `level` is a value between 0 (no volume) and 100 (the maximum volume), where 80 is typically the default volume. For example:

`<ESC>\vol=10\` I can speak rather quietly, `<ESC>\vol=90\` but also very loudly.

Setting the end-of-sentence pause duration

The control sequence `<ESC>\wait=value\` sets the end of sentence pause duration (wait period) to a value between 0 and 9, where the pause will be 200 msec multiplied by that number. Some examples:

`<ESC>\wait=2\` There will be a short wait period after this sentence. `<ESC>\wait=9\` This sentence will be followed by a long wait period. Did you notice the difference?

Setting the spelling pause duration

The control sequence `<ESC>\spell=duration\` sets the inter-character pause to the specified value in msec. For example:

The part code is `<ESC>\tn=spell\<ESC>\spell=200\a134b<ESC>\tn=normal\`

Note: The spelling pause duration does not affect the spelling done by `<ESC>\readmode=char\` because that mode treats each character as a separate sentence. To adjust the spelling pause duration for `<ESC>\readmode=char\`, set the end of sentence pause duration using `<ESC>\wait\` instead.

Controlling end-of-sentence detection

The control sequences `<ESC>\eos=1\` and `<ESC>\eos=0\` control end of sentence detection, with `<ESC>\eos=1\` forcing a sentence break and `<ESC>\eos=0\` suppressing a sentence break. To suppress a sentence break, the `<ESC>\eos=0\` must appear immediately after the symbol that triggers the break (such as after a period). To disable automatic end-of-sentence detection for a block of text, use `<ESC>\readmode=explicit_eos\` as described below.

Some examples:

Tom lives in the U.S. `<ESC>\eos=1\` So does John. 180 Park Ave. `<ESC>\eos=0\` Room 24
Controlling the read mode

The control sequence \texttt{\textless ESC\textbackslash readmode=mode\textgreater} can change the reading mode from sentence mode (the default) to various specialized modes:

<table>
<thead>
<tr>
<th>Read mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{\textless ESC\textbackslash readmode=sent\textgreater}</td>
<td>Sentence mode (the default)</td>
</tr>
<tr>
<td>\texttt{\textless ESC\textbackslash readmode=char\textgreater}</td>
<td>Character mode (similar to spelling)</td>
</tr>
<tr>
<td>\texttt{\textless ESC\textbackslash readmode=word\textgreater}</td>
<td>Word-by-word mode</td>
</tr>
<tr>
<td>\texttt{\textless ESC\textbackslash readmode=line\textgreater}</td>
<td>Line-by-line mode</td>
</tr>
<tr>
<td>\texttt{\textless ESC\textbackslash readmode=explicit_eos\textgreater}</td>
<td>Explicit end-of-sentence mode (sentence breaks only where indicated by \texttt{\textless ESC\textbackslash eos=1\textgreater})</td>
</tr>
</tbody>
</table>

Example:

\texttt{\textless ESC\textbackslash readmode=sent\textgreater} Please buy green apples. You can also get pears.
(This input will be read sentence by sentence.)

\texttt{\textless ESC\textbackslash readmode=char\textgreater} Apples
(The word "Apples" will be spelled.)

\texttt{\textless ESC\textbackslash readmode=word\textgreater} Please buy green apples.
(This sentence will be read word by word.)

\texttt{\textless ESC\textbackslash readmode=line\textgreater}
Bananas
Low-fat milk
Whole wheat flour
(This input will be read as a list, with a pause at the end of each line.)
<ESC>\readmode=explicit_eos\n
Bananas.
Low-fat milk.
Whole wheat flour.
(This input will be read as one sentence.)

Changing the voice

The control sequence <ESC>\voice=voice_name\ changes the speaking voice, which also forces a sentence break. For example:

<ESC>\voice=samantha\ Hello, this is Samantha. <ESC>\voice=tom\ Hello, this is Tom.

Labeling text for language identification

The control sequence <ESC>\lang=unknown\ labels all the text from that position up to a <ESC>\lang=normal\ or the end of the input as being from an unknown language. When the language identifier scope is configured to user-defined using the Vocalizer configuration file or API (user-defined is the default), this triggers Vocalizer to use its built-in language identifier to determine the language. For other language identifier scopes, this control sequence is simply ignored.

Vocalizer does language identification on a sentence-by-sentence basis within that region, where for each sentence it will use statistical models to determine the language, then switch the synthesis voice to a voice for that language if necessary. The synthesis voice will be restored to the original voice at the next <ESC>\lang=normal\ or the end of the synthesis request.

Note: Vocalizer does not support specifying an explicit language name instead of “unknown”.

Language identification is only supported for a limited set of languages.

Example:

Le titre de la chanson est : <ESC>\lang=unknown\ In Between <ESC>\lang=normal\n
Indicating a paragraph break
The control sequence `<ESC>\para\` indicates a paragraph break, and also implies a sentence break. The only difference between this and `<ESC>\eos=1\` (end of sentence) is that this triggers the delivery of a paragraph mark event.

Example:

Introduction to Vocalizer. `<ESC>\para\` Vocalizer is a state-of-the-art text to speech system.

**Identifying a word to accent within a sentence**

The control sequence `<ESC>\sent_accent\` is a hint to place a sentence accent on the word that follows. Note that manually inserted sentence accents may have no effect in Vocalizer, as the Vocalizer synthesis module may have reasons to override the requested sentence accent.

```
<ESC>\sent_accent\John is coming tomorrow. (that is, not Jim)
John is coming <ESC>\sent_accent\tomorrow. (that is, not today)
```

**Resetting control sequences to the default**

The control sequence `<ESC>\rst\` resets all parameters to the original settings used at the start of synthesis.

For example:

```
<ESC>\vol=10\ The volume is set to a low value. <ESC>\rst\ Now it is reset to its default value.
<ESC>\rate=10\ The rate is set to a low value. <ESC>\rst\ Now it is reset to its default value.
```