Hub One

VoiXtreme Grammar file

Table of contents

1 General configuration

The grammar descriptor source file is called **cd_xx.bnf**. This file is compiled with the VoixtremeGrammar comparator installed in the C:\SofToGo\voiXtremeGrammar\V.V.V folder.

V.V.V: is the version of the VoiXtreme package.
xx: 'es ', 'en', ' fr ', etc. depending on the used language (es = spanish; en = english; fr = french).

Once the compilation is done, the files **cd_xx.lcf** and **cd_xx_orig.lcf** are generated. These files are copied to the folder corresponding to the "profile" selected at compilation time when compilation is run through ScreenTracker developer environment.

Usually project files are in the following folder:

C:\SofToGo\ScreenTrackerProjects\<project name>\<project version>\Export\<profile name>.

2 Voice recognition

Voice recognition is done over words (called "orthography")defined into any grammar rule present in the grammar source file (cd_xx.bnf file).

To be recognized, each word must have the corresponding sound representation (pronunciation). The sounds corresponding to each word are calculated by the grammar compiler (cd_xx.lcf file).

So to be recognized a word must be defined in a grammar, and this grammar must be activated. Grammar activation / deactivation is managed by the voice engine following application requirements at each voice input.

No other words can be recognized by the voice engine.

3 Grammar rules:

Several grammar rules are defined in the cd_xx.bnf source file.

Grammar rules are nested rules in a grammar tree which the starting root is <speech>.

Grammar rules are already defined in the file, so the voice system manages them to activate / deactivate following application needs. Grammars rules can be modified, but they should not be added / removed.

Grammar rules contains a list of words (orthography) separated by vertical bars “pipes”.
i/e: `<digits>: 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ;`

Three types of grammar rules are managed:

- Digits user grammar rules
- Internal managed grammar rules
- Customer defined grammar rules

3.1.1 The digits user grammar rules

Grammar rules are already defined in the file, so the voice system manages them corresponding to the selected “grammar letter” in the voice project.

Grammar letter	Grammar rule(s)	Meaning
D	<digit>	Connected digits from 0 to 9. i/e “*one two five*” for “125”
E	<digex>	Extended (double) digits from 00 to 99. i/e: “*twenty one*” for “21”.
M	<millenium1>	One pair of extended (double) digits from 00, 00 to 99, 99.in one single utterance. i/e: “*eleven forty eight*” for “1148”
N	<millenium2>	List of pairs of extended digits from 00, 00, 00… to 99, 99, 99… in several utterances (echoed). i/e: “*zero nine*” 09, “*twenty five*” 25, “*fifteen*” 15 for “092515”
H	<hundreds>	Single, double, hundred + single or hundred + double digits from 0 to 999. i/e: “two hundred and seventy eight” for “278”.

3.1.2 Internal managed grammar rules

These grammars are automatically managed by voice system following input types

- Confirmed input
- Calculator
- Pause and activity mode

Grammar letter	Grammar rule	Meaning
C	<controls>	ResAccet and ResCancel in confirmation mode
F	<functions>	Customer functions (deprecated)
N/A	<calculator>	Keywords managed in calculator mode.
N/A	<suspend>	Valid Keywords in active mode (allowing to suspend the recognition).
N/A	<resume>	Valid Keywords in pause mode (allowing to resume the recognition).

3.1.3 Customer defined grammar rules

These grammars contain the user defined keywords. These keywords should be associated to several actions defined in the triggers file.

Grammar letter	Grammar rule	Meaning
0 … 9	<custom0> to <custom9>	Customer defined keywords
Q … Z	<customQ> to <customZ>	

4 Pronunciation

Each word or “orthography” (digit or keyword) has a corresponding phonetic pronunciation calculated by the grammar compiler, and used to recognize each word.

This grammar compiler will generate one or more pronunciations for a word. These several pronunciations will cover some dialectic variants or local usages. These pronunciations may be replaced complemented by different ones.

4.1 Alternate pronunciation

It will be needed to use a different pronunciation for any word, to replace the current one. Especially if application uses some words that have not current pronunciation as coded keywords used as well defined input data. i/e: “IEVWPLR103” or “PRN01”, and so on that designates specific actions.

Also some input words will be used in a different language than used. i/e: typing English words in a French environment.

For this case, an alternative pronunciation will be globally defined in the grammar file.

```
!pronounce IEVWPLR103 PRONAS "Reception printer"
```

In this case the keyword (orthography) “IEVWPLR103” is defined in one or more grammar rules:

```
<custom1>: IEVWPLR103 | <custom1_uw>;
```

Note: The alternative pronunciation replaces any other calculated pronunciation, and is global for the entire grammar file. So the word (orthography) should be defined only one time in the “!pronounce” statement.

If more than one pronunciation is needed for a single word, they can added to the “!pronounce” statement:

```
!pronounce IEVWPLR103 PRONAS "Reception printer" | PRONAS "Zebra
printer"
```

To have different ways to pronounce the same word.

4.2 Complementary pronunciation

It is possible to keep the original pronunciation generated by grammar compiler (instead to replace these one) by adding add any local variant to the existing ones.

In this case it is needed to add an “alias” word in the grammar to carry the added pronunciations to complement the original one. This alias word will be the same one preceded by “xlt_” prefix. This prefix indicates it as an alias, and will be processed by the voice engine. Example: we wish add the “nothing” pronunciation to designate the number 0, and allow to say “zero” or “nothing” by the user to input a “0”.

In this case we will add an alias to “0” in the <digit> grammar rules:

```
<digits>: 0 | xlt_0 | 1 | 2 | 3 |  4 | 5 | 6 | 7 | 8 | 9 ;
```

Then give to this alias “xlt_0” the alternate pronunciation of *“nothing”* in the !pronounce statement.

```
!pronounce xlt_0 PRONAS "nothing";
```

4.3 Phonetic transcription

Some specific words or variants will not exist in the built-in dictionary, or the system is used by non-native speakers and we need a variant that not exists. In this case it is possible to write the pronunciation as a phonetic transcription. The voice engine uses the L&H phonetic transcription that allows writing down any phoneme by Latin letters and other current symbols.

Each language package has specific L&H phonetic transcriptions that will be found in the “Phoneme Table” corresponding to the used language.

Consider we wish to add the pronunciation “leven” to the number 11 as a variant. We can write a phonetic transcription in the !pronounce statement. The phonetic transcription is not preceded by the “PRONAS” modifier.

```
!pronounce xlt_11 "#'lE.v$n#";

<digex>: 10 | 11 | xlt_11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 (…)
```