# ESPEAK

eSpeak is a compact open source software speech synthesizer for Linux, Windows, and other platforms. It uses a formant synthesis method, providing English, Vietnamese and many other languages in a small size. ESpeak was written in C. eSpeak is derived from the "Speak" speech synthesizer for British English for Acorn RISC OS computers which was originally written in 1995 by Jonathan Duddington. A rewritten version for Linux appeared in February 2006 and a Windows SAPI 5 version in January 2007. Subsequent development has added and improved support for additional languages. Because of its small size and many languages, it is included as the default speech synthesizer in the NVDA open source screen reader for Windows, and on the Ubuntu and other Linux installation discs.

ESpeak provides two methods of synthesis: the original eSpeak synthesizer and a Klatt synthesizer. In addition, eSpeak can be used as a front-end, providing text-to-phoneme translation and prosody, to MBROLA diphone voices. The eSpeak and Klatt synthesizers use different types of formant synthesis. The eSpeak synthesizer creates voiced speech sounds such as vowels and sonorant consonants by adding together sine waves to make the formant peaks. Unvoiced consonants such as /s/ are made by playing recorded sounds. Voiced consonants such as /z/ are made by mixing a synthesized voiced sound with a recorded unvoiced sound. The Klatt synthesizer mostly uses the same formant data as the eSpeak synthesizer. It produces voiced sounds by starting with a waveform which is rich in harmonics (simulating the vibration of the vocal cords) and then applying digital filters in order to produce speech sounds.

ESpeak is available a command line program (Linux and Windows) to speak text from a file or from stdin, shared library version for use by other programs and SAPI5 version for Windows, so it can be used with screen-readers and other programs that support the Windows SAPI5 interface. ESpeak has been ported to other platforms, including Android, Mac OSX and Solaris. It is Includes different Voices, whose characteristics can be altered, can produce speech output as a WAV file. SSML (Speech Synthesis Markup Language), HTML is supported. Espeak has compact size. The program and its data, including many languages, totals about 2 Mbytes. Language voices are identified by the language's ISO 639-1 code. They can be modified by "voice variants". These are text files which can change characteristics such as pitch range, add effects such as echo, whisper and croaky voice, or make systematic adjustments to formant frequencies to change the sound of the voice. eSpeak uses an ASCII representation of phoneme names which is loosely based on the Kirshenbaum system. Phonetic representations can be included within text input by including them within double square-brackets. For example: espeak -v en "Hello [[w3:ld]]" will say "Hello world" in English. The speech is clear, and can be used at high speeds, but is not as natural or smooth as larger synthesizers which are based on human speech recordings. The quality of the language voices varies greatly. Some have had more work or feedback from native speakers than others. Most of the people who have helped to improve the various languages are blind users of text-to-speech. Therefore, espeak support development tools for producing and tuning phoneme data. Developer can modify phonemes, voice of any languages for improve quality of speech.

There is a separate set of pronunciation files for each language, their names starting with the language name. There are two separate methods for translating words into phonemes: Pronunciation Rules source file is <language>\_rules and lookup dictionary source files are <language>\_list and <language>\_extra

Each of the language's phonemes is represented by a mnemonic of 1, 2, 3, or 4 characters. Together with a number of utility codes (eg. stress marks and pauses), these are defined in the phoneme data file. The utility 'phonemes' are:

' primary stress

, secondary stress

% unstressed syllable

= put the primary stress on the preceding syllable

\_: short pause

\_ a shorter pause

|| indicates a word boundary within a phoneme string

| can be used to separate two adjacent characters, to prevent them from being considered as a multi-character phoneme mnemonic It is not necessary to specify the stress of every syllable. Stress markers are only needed in order to change the effect of the language's default stress rule.

If we want to improve articulation of each language, we will modify phonemes in file < language >\_rules, <language>\_list and <language>\_extra. Example if we must modify phonemes of Vietnamese, we must modify file vi\_rules, vi\_list, vi\_extra.

In this:

* The rules in the <language>\_rules file specify the phonemes which are used to pronounce each letter, or sequence of letters. Some rules only apply when the letter or letters are preceded by, or followed by, other specified letters.To find the pronunciation of a word, the rules are searched and any which match the letters at the in the word are given a score depending on how many letters are matched. The pronunciation from the best matching rule is chosen. The pointer into the source word is then advanced past those letters which have been matched and the process is repeated until all the letters of the word have been processed. The rules are organized in groups, each starting with a ".group" line.:
* .group <character>

A group for each letter or character.

* .group <2 characters>

Optional groups for some common 2 letter combinations. This is only needed, for efficiency, in cases where there are many rules for a particular letter. They would not be needed for a language which has regular spelling rules. The first character can only be an ascii character (less than 0x80).

* .group

A group for other characters which don't have their own group.

* .L<nn>

Defines a group of letter sequences, any of which can match with Lnn in a pre or post rule. nn is a 2 digit decimal number in the range 01 to 25. eg: .L01 b bl br pl pr

* .replace

Character substitutions can be specified by using a .replace section at the start of the \_rules file. Each line specified either one or two alphabetic characters to be replaced by another one or two alphabetic character. This substitution is done to a word before it is translated using the spelling-to-phoneme rules. Only the lower-case version of the characters needs to be specified

* The <language>\_list file contains a list of words whose pronunciations are given explicitly, rather than determined by the Pronunciation Rules. The <language>\_extra file, if present, is also used and it's contents are taken as coming after those in <language>\_list. Also the list can be used to specify the stress pattern, or other properties, of a word. If the Pronunciation rules are applied to a word and indicate a standard prefix or suffix, then the word is again looked up in Pronunciation Dictionary List after the prefix or suffix has been removed.

When we modify file file < language >\_rules, file < language >\_list completed. We must compile it to file < language >\_dict in folder espeak\_data. In linux, we can be compile by command line: espeak –compile=<language>. Eg complie Vietnamese rule, list: espeak –compile=vi

Note before we compile, we must do the command from inside directory dictsource which contains the file vi\_rule and vi\_list.