`to-kana`, a program for converting romaji to kana.

Ethan Long

ABSTRACT

`to-kana` converts a incoming stream of romaji to either hiragana or katakana either from a file, or from standard input, and outputs to a file or standard output. `to-kana` aims to be as modular as possible, it should be usable for any project that requires a stream of romaji to be converted.

1. Purpose in `nIME`

`nIME` uses `to-kana` to convert the stream of keyboard inputs into basic kana for `to-kanji` to then convert to kanji. `to-kana` plays a very pivotal role in `nIME`, without it no kana conversion could take place.

2. Usage

`to-kana` takes in a very rigid but simple form of romaji, every character except the basic vowels takes two letters to type. The style is most similar to the ‘‘modern’’ IME, rather than `ktrans`’ SKK format. Some examples of common usage:

`watashi` → わたし
`korega!epikku!desu` → これがエピックです
`nannde` → なんで
`donnna` → どんな

In the raw stream, ‘`!`’ is used as a placeholder for the katakana trigger character, `nIME`’s GUI will trigger said placeholder when conversion is requested.

As one can see from this usage, the common letters all take two keystrokes to type.

日本語	ロマ字
あ	a
か	ka
きゃ	kya
っか	kka
っきゃ	kkya
ん	nn

3. Function documentation

All the functions in `to-kana` serve the purposes of generating kana from a stream.

3.1. *The `eval` function*

The `eval` function takes in a file pointer and converts the contents assuming that they are in the previously described romaji form. The function should run with buffered IO (as of writing buffered IO has not been implemented yet), waiting on any new input from a file, and outputting to a separate file. The final filesystem layout is TBD.

3.2. *The `strappend` function*

The `strappend` function takes in a pointer to a string, along with a character to append, it then appends the character in the first null address of the string.

Additionally, since its purpose is in the context of Japanese input, it returns an integer value representing whether or not the string now represents a full Japanese character in Romaji, if it does it returns 1.

3.3. *The `kanalook` function*

The `kanalook` function takes in a pointer to a complete Romaji buffer representing a single syllable, and converts the output to a string of `Rune`s. It does this with the help of the `kanafill` function.

3.4. *The `kanafill` function*

The `kanafill` function is designed to output a string of runes representing a single syllable in kana. It takes in a `Rune` representing the base ’a’ family of the kana (like あ, か, ら, etc.), and then uses the `in` string of characters representing the ending of the kana, along with an integer representing the presence of dakuten to determine which kana / kana combination to output.

Here is an example table of output from the `kanafill` function:

`Rune base`	`char* in`	`int dakuten`	`Rune* out`
か	a	0	か
か	e	0	け
か	i	1	ぎ
は	i	2	ぴ
は	ya	0	ひゃ
は	yu	2	ぴゅ
ハ	u	0	フ