か　ん　な
マニュアル

Version 3.2

1994 年 4 月 25 日

はじめに

本文書は『かんな』を使った日本語入力のしかた、『かんな』の各種ユーティリティコマンドの使い方、『かんな』ライブラリの使い方などを述べたマニュアルです。

本文中で、$(CANNALIBDIR) などの表現を用いますが、これらは、『かんな』のコンパイル時に設定されるディレクトリを表しており、それぞれ以下の意味を持ちます。

表 0.1: 『かんな』で用いるディレクトリ

表現	デフォルト	内容
$(CANNALIBDIR)	/usr/local/canna/lib	データなどを置くディレクトリ
$(CANNABINDIR)	/usr/local/canna/bin	実行形式コマンドを置くディレクトリ
$(CANNALOCKDIR)	@(LockDir)	cannaserver のロックファイルを置くディレクトリ
$(CANNASPOOLDIR)	/usr/spool/canna	cannaserver のログファイルを置くディレクトリ
$(CANNAINCLUDEDIR)	/usr/local/canna/include/canna	『かんな』関連のヘッダファイルを置くディレクトリ
$(IROHASPOOLDIR)	/usr/spool/iroha	『かんな』Version 1.2 の頃のスプールディレクトリ
$(IROHAINCLUDEDIR)	/usr/include/iroha	『かんな』Version 1.2 の頃のヘッダファイル用ディレクトリ

本マニュアル中、XFER、NFER など、マシンによっては存在しないキーを使用している場合がありますが、他のキーにカスタマイズして使って頂けるようお願いします。

一太郎はジャストシステム (株) の商標です。

目次

第 1 章

日本語入力

1.1 概説

1.1.1 特徴

- 日本語入力方式

  カナ入力、ローマ字入力を用いて、次のような日本語入力が行えます。

  - かな漢字変換 (学習機能を持つ連文節変換および逐次自動変換)
  - ひらがな入力
  - カタカナ変換
  - ローマ字への変換
  - 大文字小文字変換
  - 16 進コード変換
  - 記号選択入力
  - 部首一覧選択変換
  - 単語登録 / 単語削除
- キーバインドのカスタマイズが可能です。
- クライアントサーバ方式です。
- 複数の辞書を同時に使用することが可能です。
- EUC の G0 〜 G3 の 4 つの文字セットを使えます。

  JIS 第 1・2 水準の文字をサポートします。

1.1.2 かな漢字変換辞書

システム辞書を含め、複数のかな漢字変換辞書を同時に使用することができます。
かな漢字変換辞書には次の 2 つの形式があります。

- テキスト形式辞書

通常のテキスト形式の辞書です。単語登録などはこの辞書に対して行われます。テキスト形式ですので通常のエディタでメンテナンスすることもできます。

- バイナリ形式辞書

  辞書の読み込みが高速になるような形式に変換された辞書です。単語の追加登録は行えません。

辞書ユーティリティツールを使ってテキスト形式とバイナリ形式を相互変換することが可能です。

辞書はデフォルトでは $(CANNALIBDIR)/dic/canna 配下のものが使われます。 $(CANNALIBDIR) は『かんな』のデータファイルなどが置かれるディレクトリです。デフォルトでは /usr/local/canna/lib になっておりますが、ご使用のマシンでどのディレクトリになっているかは、『かんな』をインストールした人に確認してください。

1.2 かな漢字変換の仕様

本システムのかな漢字変換の仕様は以下のようになっています。

基本辞書の語数	約 8 万語
同時使用可能辞書数	システム、部首、ユーザ、グループ辞書それぞれについて 15 個
品詞数	440 品詞
読みとして用いることのできる字種	EUC コード G0、 G1、 G2、 G3 領域の文字
候補として用いることのできる字種	EUC コード G0、 G1、 G2、 G3 領域の文字
ひとつの読みに対する候補の最大数	255 個
読みの最大長	(251 − 候補バイト長) バイト
候補の最大長	31 文字
テキスト辞書の一行の最大長	1023 バイト

1.3 日本語入力におけるモードの種類

入力モードには、アルファベット入力モードと日本語入力モードがあります。

(1) **アルファベット入力モード**

ASCII 文字の入力と、かなロックキーを使って半角カナ文字の入力を行う通常のモードです。

(2) **日本語入力モード**

ひらがな・カタカナ・漢字・記号などを入力するためのモードです。このモードでは、英字キーを使ってローマ字で入力するとひらがなに変換されます。

日本語入力モードには、変換入力モード・記号入力モード・確定入力モードがあります。

(a) 変換入力モード

かな漢字変換を伴って、日本語文字列を入力するためのモードです。

このモードでは、入力された読みに対して、かな漢字変換を行ったり文字種変換を行って読み入力をカタカナや英数文字に変換したりすることができます。

また、部首名による変換や、 16 進コードによる変換を行うことができます。

(b) 部首入力モード

漢字はわかっているけれども読みがわからない場合に、部首による変換を行うためのモードです。

(c) 16 進コード入力モード

16 進コードによる入力を行うモードです。

(d) 記号入力モード

特殊な記号などを入力するためのモードです。

(e) 確定入力モード

ひらがな、全角カタカナ、半角カタカナ、全角英数、半角英数などを入力するためのモードです。ただし、デフォルトでは、この機能に対するキーは割り当てられていません。

1.4 日本語入力のモード表示

日本語入力モードでは、画面の最下行をガイドラインとして使用し、このガイドラインに入力モードを表示します。

ガイドラインには、［ あ ］、［記号］など、入力モードに応じたメッセージが表示されます (図 1.1 参照)。デフォルトでは、変換入力モード時は［ あ ］と表示されます。候補の一覧などは、ガイドラインの右側の余白部分に表示されます。

図 1.1: ガイドラインのデータ入力モード表示

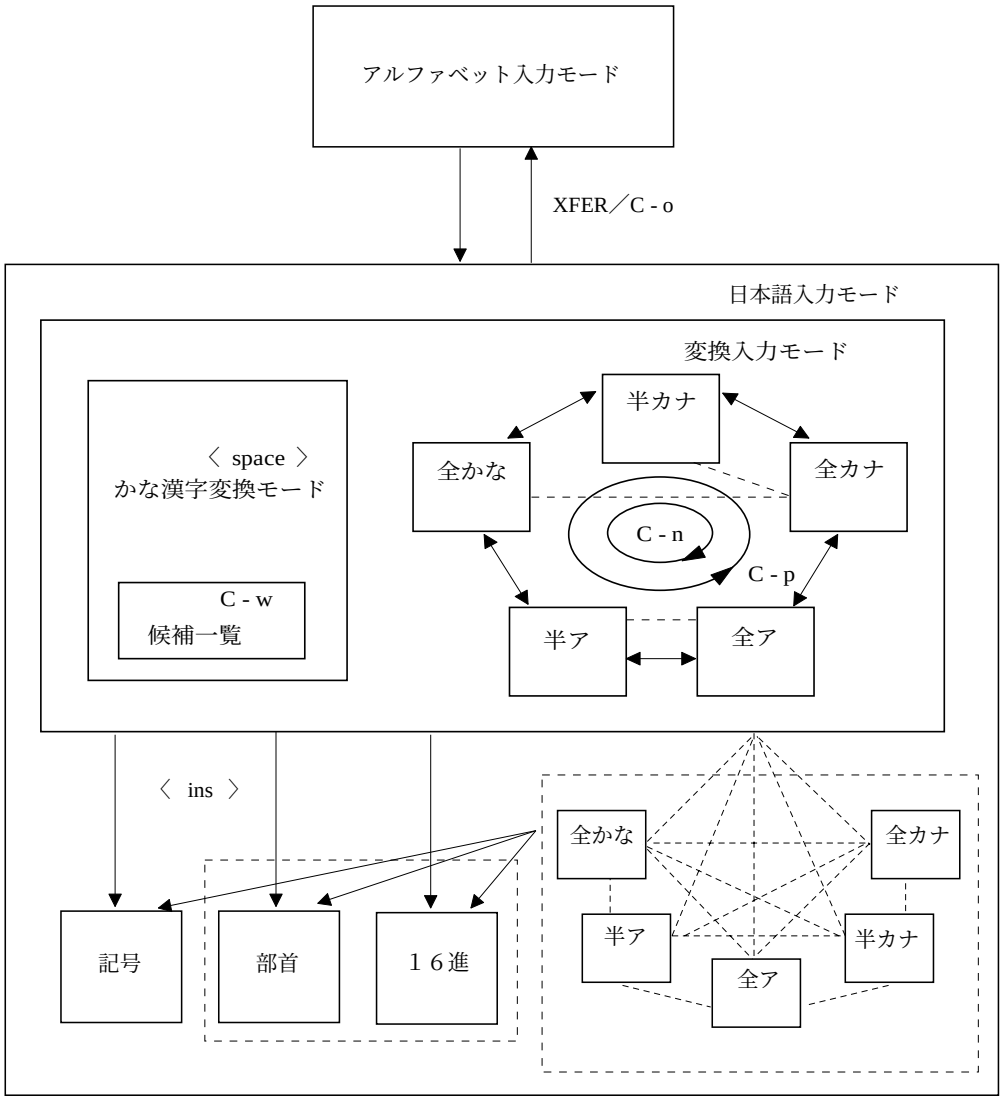
［ あ ］	全角ひらがな入力モード
［全ア］	全角カタカナ入力モード
［半ア］	半角カタカナ入力モード
［全英］	全角アルファベット入力モード
［半英］	半角アルファベット入力モード
［逐次］	逐次変換状態
［１６進］	１６進コード入力状態
［部首］	部首入力状態
［記号］	記号一覧表示状態
［字種］	文字種変換状態
［漢字］	単一候補表示状態
［一覧］	候補一覧表示状態
［ロ］	ロシア文字選択状態
［ギ］	ギリシア文字選択状態
［罫線］	罫線選択状態
［変更］	サーバ変更状態
［削除］	単語削除状態
［登録］	単語登録状態
［品詞］	単語登録モードの品詞選択状態
［辞書］	単語登録モードの辞書選択状態
［q］	引用入力状態
［拡張］	拡張機能選択状態

1.5 入力モードの切り替えの操作とモードの移行

入力モードの切り替えはキー操作によって行います。図 1.2は、切り替え操作とモードの移行の様子を示しています。

ただし、”CTRL + □”はコントロールキーを押しながら次のキーを押すという意味を表します。同じ意味で C- ××のような表記をすることもあります。

図 1.2: 入力モードの切り替えの操作とモードの移行



全かな：全角ひらがな

全カナ：全角カタカナ　　半カナ：半角カタカナ

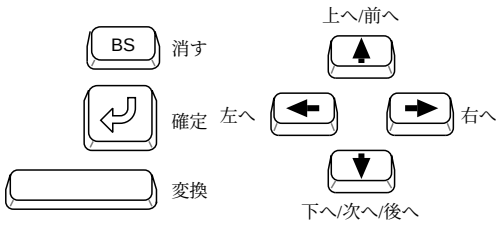
全ア：全角アルファベット　　半ア：半角アルファベット

かな漢字変換モード：単候補表示状態と候補一覧表示状態をまとめたもの

1.6 基本操作

『かんな』では以下のキーを基本操作のキーとして使います。

図 1.3: 基本操作のキー



また、矢印キーについては、 Emacs のキー割り当てに合わせてコントロールキーで代替することもできます。

表 1.1: キー操作

キー	かな漢字変換での機能
XFER	アルファベット入力モードと日本語入力モードとの相互切り替え、変換、次候補
空白	変換、次候補
RETURN 、 CTRL−m	確定
BS 、 CTRL−h	カーソル左文字削除、漢字を読みに戻す
← 、 CTRL−b	カーソル左、左候補、左文節選択
→ 、 CTRL−f	カーソル右、右候補、右文節選択
CTRL−g	取りやめ
↓ 、 CTRL−n	次字種、次候補表示、次の候補一覧表示
↑ 、 CTRL−p	前字種、前候補表示、前の候補一覧表示

1.7 アルファベット入力モードと変換入力モードの切り替え操作

アルファベット入力モードと、日本語が入力できる変換入力モードとを切り替えるには XFER または CTRL + o を用います。

(1) 何もしない状態では、アルファベット入力モードです。

```
% 
```

(2) XFER または CTRL + o を押すと変換入力モードになります。

```
% 
[ あ ]
```

(3) もう一度 XFER または CTRL + o を押すとアルファベット入力モードに戻ります。

```
% 
```

1.8 ローマ字かな変換

変換入力モードでは、ローマ字かな変換を行うことができます。ローマ字による入力をかなに変換する規則は **付録 D ローマ字かな変換表** に従っています。入力されたローマ字のデータは、かな文字に変換可能になった時点で、かな文字に変換されます。ここでは、特に注意すべき点だけを示します。

(1) ローマ字かな変換不可能文字の処理

入力されたローマ字のうち、かな文字に変換できない文字はそのままになります。

例) aiu()BCD → あいう ()BCD

デフォルトの設定では大文字に関してはいずれの文字もローマ字かな変換には用いられません。

(2) バックスペースの処理

子音を入力している状態で、かなに変換される前に BS が押されても、入力済の子音はローマ字かな変換の対象として有効です。次に入力される文字と組合わされてローマ字かな変換されます。たとえば、以下の例ではキー操作の結果は「しぇ」になり「ｓへ」にはなりません。

sh BS he と入力した場合

キー操作	表示
s	s
h	sh
BS	s
h	sh
e	しぇ

ローマ字かな変換入力中に、 BS を押すと、ローマ字の 1 つ前の文字ではなく、表示されている 1 つ前の文字が消去されます。以下の例では BS の入力によって「く」だけが残り「くｔ」にはなりません。

kutu BS と入力した場合

キー操作	表示
k	k
u	く
t	くｔ
u	くつ
BS	く

同様に、以下の例では BS の入力によって「ゃ」が消去され「き」は残ります。

kya BS と入力した場合

キー操作	表示
k	k
y	ky
a	きゃ
BS	き

ただし、カスタマイズにより、BSで最後のかなをローマ字に戻すようにすることも可能です。詳細は、**2.6.4 カスタマイズのキーワードの（12） そのほかのキーワード** を参照してください。

(3) ハツ音(ン) の入力

(a) 「nn」、「mn」または「n'」と入力します。

tanni	たんい
tamni	たんい
tan'i	たんい

(b) 次に続く文字が子音の場合は、「n」一文字でも「ん」になります。

tango	たんご

(4) 促音の入力

次に続く子音を重ねて入力します。

hatten	はってん

(5) ヨウ音の入力

「kya」で「きゃ」が入力できるように子音の組み合わせにより自動的に生成されますが、単独のときは「x」を先行させて入力します。たとえば以下のように入力します。

yakixyuu	やきゅう

(6) 長音の入力

長音は、「−」 (ハイフン) キーにより入力します。

(7) 全角文字の入力時の変換

キーボード上に存在する記号はおよそ、そのキーの刻印の全角文字が入力されます。ただし、以下に示すような例外があります。

入力キー	表示	備考
[	「	
]	」	
{	『	
}	』	
,	、	
.	。	
~	￣	
\	¥	
−	ー	
SPACE	半角スペース	0x20
(	(	半角
)	)	半角

(8) 変換キー (RETURN) の処理

ローマ字かな変換途中で RETURN が押されたとき、まだ変換されていない部分のアルファベットは、アルファベットのまま確定されます。ただし、 n などの変換可能な文字は変換されます。

(例 1)

キー操作	表示
k	k
y	ky
o	きょ
u	きょう
h	きょう h
RETURN	きょう h

(例 2)

キー操作	表示
m	m
i	み
k	み k
a	みか
n	みか n
RETURN	みかん

(9) 未確定部分の表示

最長一致パターンと照合するまではアルファベットが表示され、一致したと同時に、かなが表示されます。

(例 1)

キー操作	表示
k	k
y	ky
o	きょ

(例 2)

キー操作	表示
a	あ
n	あ n
n	あん
n	あん n
a	あんな
i	あんない

(10) かなロックキーでの入力

かなロックキーを押すことにより、ローマ字かな変換を使わずにキーボードからかなを入力することも可能です。その際、「゛」や「゜」については、直前に入力された文字の濁点・半濁点として結合できる場合は結合されます。

(例) カ + ゛ → が

1.9 読みの入力

1.9.1 読みの確定

読みの確定には RETURN または CTRL + m または NFER を用います。

(1) 読みを入力します。

w a t a s i h a

```
% わたしは
[ あ ]
```

(2) RETURN または NFER または CTRL + m を押して確定します。

```
% わたしは
[ あ ]
```

1.9.2 読みの取り消し

CTRL + g を押すと未確定の読みがすべて取り消されます。

(1) 読みを入力します。

```
% わたしは
[ あ ]
```

(2) CTRL + g を押して読みを取り消します。

```
%
[ あ ]
```

1.9.3 左方向削除

BS または CTRL+ h を押すとカーソルの左の 1 文字を削除することができます。

(1) 読みを入力します。

```
% わたしが
[ あ ]
```

(2) BS または CTRL+ h を押してカーソルの左の 1 文字を削除します。

```
% わたし
[ あ ]
```

(3) 新たに読みの続きを入力します。

```
% わたしの
[ あ ]
```

1.9.4 カーソルの移動

読みを入力しているときに入力を間違えたり入力し忘れたりした場合は、カーソルを移動してその部分だけ入力し直すことができます。

左への移動は ← または CTRL+ b 、右への移動は → または CTRL+ f 、左端への移動は CTRL+ a 、右端への移動は CTRL+ e を押します。

(1) 読みを入力します。

```
% ながさきはきょうはあめです
[ あ ]
```

(2) ← または CTRL+ b を 4 回押してカーソルを左に移動します。

```
% ながさきはきょうはあめです
[ あ ]
```

(3) BS を押して『は』を削除します。

```
% ながさきはきょうあめです
[ あ ]
```

(4) 『も』を入力します。

```
% ながさきはきょうもあめです
[ あ ]
```

1.9.5 右方向削除

CTRL + d を押すとカーソル上の 1 文字が削除されます。

(1) 読みを入力してカーソルを移動します。

```
% わたくしはきょうは
[ あ ]
```

(2) CTRL + d を押してカーソル上の 1 文字を削除します。

```
% わたしはきょうは
[ あ ]
```

1.9.6 カーソル以降削除

CTRL + k を押すとカーソルから右の読みがすべて削除されます。

(1) 読みを入力してカーソルを移動します。

```
% わたしは[き]ょうは
[ あ ]
```

(2) [CTRL]+ [k] を押してカーソルから右の読みをすべて削除します。

```
% わたしは
[ あ ]
```

1.10 漢字の入力

かなを漢字に変換する方法には、連文節変換と逐次自動変換の 2 つの方法があります。逐次自動変換での漢字入力のキー操作は、連文節変換での漢字入力のキー操作とほぼ同じです。

かな漢字変換の際、候補を 1 つだけ表示している状態を単候補表示状態といい、候補一覧を表示している状態を候補一覧表示状態といいます。

1.10.1 連文節変換

連文節変換では SPACE または XFER を押して読みを漢字に変換します。

(1) 読みを入力します。

```
% きょうははれです
[ あ ]
```

(2) SPACE または XFER を押して読みを漢字に変換します。

```
% 今日は晴れです
[漢字]
```

1.10.2 逐次自動変換

連文節変換では SPACE または XFER を押して漢字に変換していました。これに対して逐次自動変換では SPACE または XFER を押さなくても、読みが適当な長さになると自動的に漢字に変換されます。逐次自動変換を行っているときの操作は、読みを入力すると自動的に変換されること以外は連文節変換とほぼ同じになります。

逐次自動変換の流れを、「明日は雪が降るでしょう」を入力する例で説明します。

(1) 「あしたはゆきが」までを入力します。

```
% あしたはゆきが
[逐次]
```

(2) 「ふ」を入力します。

```
% 明日は|ゆきがふ
[逐次]
```

このように、読みが適当な長さになると先頭から途中までの読みが漢字の候補に変換されます。変換されてできる漢字の候補を候補文節といいます。一度の変換で複数の候補文節ができることもあります。新しい候補文節ができたときには、新しくできた候補文節の最後のものが反転表示されます。

(3) 続けて「るで」を入力します。

```
% 明日は|ゆきがふるで
[逐次]
```

(4) 「しょ」を入力します

```
% 明日は|雪が|ふるでしょ
[逐次]
```

さらに新しい候補文節ができると、反転している箇所も移動します。

(5) 最後に、「う」を入力します。

```
% 明日は|雪が|ふるでしょう
[逐次]
```

読みの入力が終わりましたが、「ふるでしょう」の部分が漢字の候補に変換されずに残っています。一般に逐次自動変換では、読みの最後の部分が漢字の候補に変換されずに読みのまま残ってしまいます。このようなときには、 SPACE または XFER を押して、残っている読みを強制的に漢字の候補に変換します。

(6) SPACE または XFER を押します。

```
% 明日は|雪が|降るでしょう
[漢字]
```

最後までが漢字の候補になりました。この状態はすでに述べた連文節変換の状態と同じであり、連文節変換と同じ操作が可能です。

(7) 正しい漢字になっているので、 RETURN または NFER または CTRL + m を押して確定します。

```
% 明日は雪が降るでしょう
[逐次]
```

1.10.3 候補選択

1.10.3.1 候補の選択

読みが漢字に変換された状態で ↓ または CTRL + n を押すと次候補を選択することができます。また、↑ または CTRL + p を押すと前候補を選択することができます。

(1) 読みを漢字に変換します。

```
% 今日は晴れです
[漢字]
```

(2) ↓ または CTRL + n を押して次候補を表示します。

```
% 京は晴れです
[漢字]
```

1.10.3.2 漢字変換の取り消し

BS または CTRL + g または CTRL + h を押すと漢字変換が取り消されて読みの状態に戻ります。

(1) 読みを漢字に変換します。

```
% 今日は晴れです
[漢字]
```

(2) CTRL + g または BS または CTRL + h を押して読みに戻します。

```
% きょうははれです
[ あ ]
```

1.10.3.3 候補の確定

RETURN または NFER または CTRL + m を押すと候補を確定することができます。

(1) 読みを入力します。

```
% きょうははれです
[ あ ]
```

(2) SPACE または XFER を押して読みを漢字に変換します。

```
% 今日は晴れです
[漢字]
```

(3) RETURN または NFER または CTRL + m を押して候補を確定します。

```
% 今日は晴れです
[ あ ]
```

1.10.4 候補一覧

候補がたくさんある場合は、候補一覧を使います。

1.10.4.1 候補一覧の表示

読みを漢字に変換した後に、SPACE または XFER を 2 回押すか、または CTRL + w を押すと候補一覧を表示することができます。候補一覧モードになるために変換キーを押す回数をカスタマイズす

ることができます。
詳細は、 **2.6.4 カスタマイズのキーワードの (12) そのほかのキーワード** を参照してください。

(1) 読みを漢字に変換します。

```
% 今日は晴れです
[漢字]
```

(2) SPACE または XFER または CTRL+ w を押して候補一覧を表示します。

```
% 京は晴れです
[一覧]  1 今日は  2 京は  3 卿は  4 経は  5 強は  6 鏡は  2/17
```

1.10.4.2 候補の移動

次候補に移動するには → または CTRL+ f または SPACE または XFER を押します。前候補に戻るには ← または CTRL+ b を押します。

1.10.4.3 候補列の移動

表示されている候補一覧に目的の候補が見当たらなかった場合は、 ↓ または CTRL+ n を押して次の候補列に移動します。前の候補列に移動するには ↑ または CTRL+ p を押します。

1.10.4.4 候補の選択

候補を選択したい場合は、その候補まで移動してから RETURN または CTRL+ m または NFER を押します。または、候補の番号を押すことによって、選択することもできます。

1.10.4.5 候補一覧の取り消し

候補一覧は BS または CTRL+ g または CTRL+ h を押して取り消すことができます。

1.10.5 文節

長い読みを漢字変換すると、いくつかの文節に分かれます。各文節に移動することができます。また、各文節は伸ばしたり縮めたりすることもできます。

1.10.5.1 文節の移動

次の文節に移動したい場合は、 → または CTRL + f を押します。前の文節に移動したい場合は、 ← または CTRL + b を押します。

(1) 読みを漢字に変換します。

```
% 長崎は今日は晴れです
[漢字]
```

(2) → または CTRL + f を押して次の文節に移動します。

```
% 長崎は今日は晴れです
[漢字]
```

(3) → または CTRL + f を押して次の文節に移動します。

```
% 長崎は今日は晴れです
[漢字]
```

(4) ← または CTRL + b を押して前の文節に移動します。

```
% 長崎は今日は晴れです
[漢字]
```

1.10.5.2 文節の伸縮

文節を縮めたい場合は、 CTRL + i を押します。文節を伸ばしたい場合は、 CTRL + o を押します。

(1) 読みを入力します。

```
% ぼくのいえにはにわがない
[ あ ]
```

(2) SPACE を押して漢字に変換します。

```
% 僕の家に埴輪がない
[漢字]
```

(3) → を押して文節を伸ばしたい所まで文節を移動します。

```
% 僕の家に埴輪がない
[漢字]
```

(4) CTRL + o を押して文節を伸ばします。

```
% 僕の家には庭がない
[漢字]
```

1.10.6 部分確定

CTRL + k を押すことにより、カーソルのあった文節より左側の部分を確定させることができます。カーソルより右側の部分は読みに戻されます。

(1) 読みを漢字に変換します。

```
% 頼りになるのは直感だけですよ
[漢字]
```

(2) → を押して文節を移動します。

```
% 頼りになるのは直感だけですよ
[漢字]
```

(3) CTRL + k を押してカーソルより左側を確定し、カーソルより右側を読みに戻します。

```
% 頼りになるのは ちょっかんだけですよ
[ あ ]
```

(4) 読みに戻った部分は削除したりして直すことができます。

CTRL + f または ← を押してカーソルを移動し、CTRL + d を 3 回押して 3 文字削除します。

```
% 頼りになるのはかんだけですよ
[ あ ]
```

1.11 カタカナの入力

カタカナを入力するには、かな漢字変換による入力・字種変換による入力・カタカナ確定入力モードによる入力という 3 種類の方法があります。

1.11.1 かな漢字変換による入力

1.11.1.1 かな漢字変換による方法 1

多くのカタカナ語に関しては、かな漢字変換辞書に登録されていますので、通常のかな漢字変換の操作でカタカナが得られます。

(1) 読みを入力します。

```
% わーぷろを
[ あ ]
```

(2) SPACE を押して読みをかな漢字変換すると、カタカナになります。

```
% ワープロを
[漢字]
```

1.11.1.2 かな漢字変換による方法 2

漢字候補の最後から 2 番目にカタカナの候補が入っていますので、SPACE を押してかな漢字変換した直後に ↑ を 2 回押せばカタカナになります。

(1) 読みを入力します。

```
% これは かん
[ あ ]
```

(2) SPACE を押して漢字に変換します。

```
% これは 完
[漢字]
```

(3) ↑ を押すと前候補つまり最後の候補 (ひらがな) を表示します。

```
% これは かん
[漢字]
```

(4) もう一度 ↑ を押すとカタカナを表示します。

```
% これは カン
[漢字]
```

(5) 次の読みを入力すると確定します。

```
% これはカン ですよ
[ あ ]
```

1.11.1.3 文節を伸ばしていく方法

2 つ以上の文節がある場合は、文節をどんどん伸ばしていくと、候補がなくなった時点でカタカナに変換されます。

(1) 読みを入力します。

```
% あれきさんどら
[ あ ]
```

(2) SPACE を押して漢字に変換します。

```
% あれ起算どら
[漢字]
```

(3) CTRL + o で文節を伸ばしていくと、候補がなくなった時点でカタカナになります。

```
% アレキサンドラ
[漢字]
```

1.11.2 字種変換による入力

読みを入力している状態では、読みの文字列の字種を変換することができます。この機能を利用してカタカナを得ることができます。

(1) 読みを入力します。

```
% あれきさんどら
[ あ ]
```

(2) ↓ を押すとカタカナに変換されます。

```
% アレキサンドラ
[字種]
```

1.11.3 カタカナ確定入力モードによる入力

カスタマイズ機能を使ってカタカナ確定入力モードにして、カタカナを入力することができます。詳細は、**2 カスタマイズ** を参照してください。

1.12 全角アルファベットの入力

全角アルファベットを入力するには、字種変換による方法と全角英数確定入力モードによる方法とがあります。

1.12.1 字種変換による入力

(1) 読みを入力します。

c o n g r a t u l a t i o n

```
% こんｇらつらちおｎ
[ あ ]
```

(2) ↓ を押して、全角アルファベットに変換します。

```
% ｃｏｎｇｒａｔｕｌａｔｉｏｎ
[字種]
```

(3) CTRL + u を押すと、大文字に変換できます。

```
% ＣＯＮＧＲＡＴＵＬＡＴＩＯＮ
[字種]
```

(4) 逆に小文字にするには CTRL - l を押します。

```
% ｃｏｎｇｒａｔｕｌａｔｉｏｎ
[字種]
```

1.12.2 全角英数確定入力モードによる入力

カスタマイズ機能でファンクションキーなどに全角英数確定入力モードを割り当てることにより、全角アルファベットを入力することができます。詳細は、 **2 カスタマイズ** を参照してください。

1.13 字種変換

1.13.1 字種の変更の仕方

読みを入力している状態や単候補表示状態では、読みの文字列を全角カタカナ・半角カタカナ・全角アルファベット・半角アルファベットなどの文字列に変換することが可能です。

字種変換には CTRL+ n または ↓ と、CTRL+ p または ↑ を用います。CTRL+ n または ↓ を入力することで、ひらがな～全角カタカナ～半角カタカナ～全角アルファベット～半角アルファベットと字種変換が行われ、もう一度 CTRL+ n または ↓ を押すと元の読みの状態に戻ります。CTRL+ p または ↑ を押すと逆回りの字種変換が行われます。

単候補表示状態から字種変換する場合は、CTRL+ c を押してから CTRL+ n (または ↓) や CTRL+ p (または ↑) を押します。

1.13.1.1 読みを入力している状態での字種変換

(1) 読みを入力します。

```
% ふぉとぐらふ
[ あ ]
```

(2) ↓ を押すと、全角カタカナに変換されます。

```
% フォトグラフ
[字種]
```

(3) ↓ を押すと、半角カタカナに変換されます。

```
% フォトグラフ
[字種]
```

(4) ↓ を押すと、全角アルファベットに変換されます。

% f o t o g u r a h u
[字種]

(5) ↓ を押すと、半角アルファベットに変換されます。

% fotogurahu
[字種]

(6) ↓ を押すと、読みに戻ります。

% ふぉとぐらふ
[あ]

1.13.1.2 単候補表示状態での字種変換

『この日本語入力システムは utility が充実しています』と入力するとします。

(1) 読みを漢字に変換し、文節を移動します。

% この日本語入力システムは鵜地理 ty が充実してます。
[漢字]

(2) CTRL + o を 4 回入力して、カレント文節長を伸ばします。

% この日本語入力システムはウチリ tyケ充実してます。
[漢字]

(3) CTRL + c を押すと、カレント文節が読みモードになります。

```
% この日本語入力システムはうちり ty ケ充実してます。
[ あ ]
```

(4) ↑ を押すと、半角アルファベットに変換されます。

```
% この日本語入力システムはutilityケ充実してます。
[字種]
```

(5) → を押して、次文節に移動します。

```
% この日本語入力システムは utilityケ充実してます。
[漢字]
```

(6) ↑ を押すと、カレント文節が読みに戻ります。

```
% この日本語入力システムは utilityが充実してます。
[漢字]
```

(7) RETURN を押すと、確定します。

```
% この日本語入力システムは utility が充実してます。
[ あ ]
```

入力している読みの中にかなに変換されないアルファベットが入っている場合は、カタカナには変換されず、ひらがな〜全角アルファベット〜半角アルファベット〜ひらがなの順で字種変換が行われます。

(1) 読みを入力します。

c o n g r a t u l a t i o n

```
% こんｇらつらちおｎ
[ あ ]
```

(2) ↓ を押すと、全角アルファベットに変換されます。

```
% ｃｏｎｇｒａｔｕｌａｔｉｏｎ
[字種]
```

カナキーで入力を行っている場合は、アルファベットには変換されず、ひらがな〜全角カタカナ〜半角カタカナ〜ひらがなの順で字種変換が行われます。

(1) 読みを入力します。

```
% ふぉとぐらふ
[ あ ]
```

(2) ↓ を押すと、全角カタカナに変換されます。

```
% フォトグラフ
[字種]
```

(3) ↓ を押すと、半角カタカナに変換されます。

```
% フォトグラフ
[字種]
```

(4) ↓ を押すと、読みに戻ります。

```
% ふぉとぐらふ
[ あ ]
```

1.13.2 字種変換領域の伸縮

かな漢字変換と同様に字種変換も変換領域を伸縮することができます。

CTRL + o で変換領域を伸ばして、 CTRL + i で変換領域を縮めます。

たとえば、『〜の』のような入力を行った場合のように、助詞である『の』は字種変換の対象にはしたくないことがあります。このような場合は、 CTRL + i で字種変換となる対象を縮めます。

(1) 読みを入力します。

```
% えーびーしーの
[ あ ]
```

(2) ↓ を押すと、全角カタカナに変換されます。

```
% エービーシーノ
[字種]
```

(3) CTRL + i を押すと、変換領域が縮まります。

```
% エービーシーの
[字種]
```

1.13.3 字種変換領域の開始点の設定 (マーク機能)

字種変換する際、マークを設定することで字種変換領域の開始点を変更することができます。マークを設定しない場合は読みの先頭か、以前字種変換した場合は字種変換した直後の位置から字種変換されます。

この機能は、入力した読みの任意の位置から字種変換を行いたい場合に使用します。セットされた字種変換領域の開始点を「マーク」、開始点を設定することを「マークをセットする」といいます。

デフォルトでは、 CTRL + SPACE でマークがセットされます。

(1) 読みを入力します。

```
% これは
[ あ ]
```

(2) CTRL + SPACE を押し、マークをセットします。

```
% これは
[ あ ]
```

(3) 読みを最後まで入力します。

```
% これはぺんです
[ あ ]
```

(4) ↓ を押すと、全角カタカナに変換されます。

```
% これはペンデス
[字種]
```

(5) CTRL + i を押すと、変換領域が縮まります。

```
% これはペンです
[字種]
```

(6) ↓ を押すと、半角カタカナに変換されます。

% これはペンです
[字種]

(7) ↓ を押すと、全角英数に変換されます。

% これはｐｅｎです
[字種]

(8) ↓ を押すと、半角英数に変換されます。

% これはpenです
[字種]

1.14 記号の入力

変換入力モードでは、いろいろな記号を入力することができます。記号入力は、記号入力モードで入力する方法と「きごう」をかな漢字変換する方法があります。

1.14.1 記号入力モードでの入力

(1) 読みがないときに INS を押すと記号入力モードになります。

```
% 
[記号]  [2121]  □ 、 。 , . ・ : ; ? ! ゛ ゜
```

(2) → を押して候補を選択します。

```
% 
[記号]  [2126]  、 。 , . ・ : ; ? ! ゛ ゜
```

```
% ・
[記号]  [2126]  、 。 , . ・ : ; ? ! ゛ ゜
```

(3) 記号入力モードを抜け、 変換入力モードに戻るためには、CTRL + g を押します。

```
% ・
[ あ ]
```

そのほかに拡張機能でも、記号入力を行うことができます。詳細は、 **1.17 拡張機能の一覧** を参照してください。

1.14.2 かな漢字変換での入力

読みで「きごう」と入力して漢字変換すると一般的な記号が候補として出てきます。記号の選択の方法は、かな漢字変換の候補選択時と同じです。

1.15 16 進コードによる入力

16 進コードで文字を入力するには、 2 つの方法があります。

1 つは、 16 進コードを通常の読みの入力と同様に入力した後に、 CTRL + y を押すことにより、対応する文字に変換する方法で、もう 1 つは、 16 進コード入力モードで入力する方法です。

1.15.1 通常のモードでの 16 進コード入力

(1) 通常の読みを入力するのと同様に 4 桁の 16 進コードを入力します。 16 進コードについては、**付録 I 16 進漢字コード一覧表** を参照してください。

```
% 2276
[ あ ]
```

(2) CTRL + y を押します。

```
% ♪
[ あ ]
```

1.15.2 16 進コード入力モードでの入力

16 進コード入力モードへは HELP キーで拡張機能一覧を表示し、**コード入力** を選択することで入ることができます。詳細は、 **1.17 拡張機能の一覧** を参照してください。

(1) 日本語入力モードで読みを入力していない状態で HELP を押すと、拡張機能の一覧が表示されます。

```
%
[拡張] 1 記号入力  2 コード入力  3 部首入力  4 単語登録  5 環境設定          1/5
```

(2) **コード入力** を選択 (→ および ← でカーソル移動後 RETURN または番号の入力) すると、 16 進コード入力モードへ移行します。

```
%
[16 進]   コード：
```

(3) コード (2 2 7 6) を入力するとそのコードに対応する文字を表示します。

```
% ♪
[16 進]    コード： 2276
```

(4) RETURN を押すと確定します。

```
% ♪
[ あ ]
```

1.16 部首変換による入力

読みがわからない漢字は部首変換を用いて入力します。

部首変換入力には 2 つの方法があります。

1 つは、部首名を通常の読みの入力と同様に入力した後に、CTRL + w を押すことにより、その部首名の漢字の一覧を表示させる方法で、もう 1 つは、部首入力モードに切り替えて入力する方法です。

1.16.1 通常のモードでの部首変換

(1) 通常の読みを入力するのと同様に部首名を入力します。
部首名については **付録 F 部首入力の部首名一覧表** を参照してください。

```
% きがまえ
[ あ ]
```

(2) CTRL + w を押します。

```
% 氛
[部首]      1 氛  2 气  3 氤  4 氣                    1/4
```

(3) 選択したい候補の番号を押すか、または CTRL + f および CTRL + b でカーソル移動後、RETURN を押して、候補を選択します。

```
% 氣
[ あ ]
```

1.16.2 部首入力モードでの入力

部首入力モードへは HELP キーで拡張機能一覧を表示し、**部首入力** を選択することで入ることができます。詳細は、**1.17 拡張機能の一覧** を参照してください。

(1) 『勀』(読み：ちょく、部首名：りきづくり) を部首入力で入力するとします。日本語入力モードで読みを入力していない状態で HELP を押すと、拡張機能の一覧が表示されます。

```
% ■
[拡張] 1 記号入力  2 コード入力  3 部首入力  4 単語登録  5 環境設定          1/5
```

(2) **部首入力** を選択 (→ および ← でカーソル移動後 RETURN または番号の入力) すると、部首一覧が表示されます。

```
% ■
[部首] 1 一  2 丿  3 凵  4 十  5 冂  6 刀  7 刈 (りっとう)  8 力          1/150
```

(3) 『力』を選択する (番号を入力するか、または CTRL + f および CTRL + b でカーソル移動後 RETURN) と、部首が『力』の文字の候補一覧が表示されます。

```
% ■
[部首] 1 飭  2 劫  3 劭  4 劵  5 勍  6 勣  7 勠  8 劬  9 勵          1/18
```

(4) 『勍』を選択する (番号を入力するか、または CTRL + f および CTRL + b でカーソル移動後 RETURN) と、『勍』が確定されます。

```
% 勍■
[ あ ]
```

1.17 拡張機能の一覧

かな漢字変換をしながら辞書やサーバの操作などを行う拡張機能を提供します。以下に機能の項目の詳細を示します。

表 1.2: 文字入力の簡略化のための機能

メ ニ ュ ー		参照章番号
記号入力	記号全般	1.18.1
	ロシア文字	1.18.2
	ギリシャ文字	1.18.3
	罫 線	1.18.4
コード入力		1.15
部首入力		1.16

表 1.3: 日本語入力をスムーズにするための機能

メ ニ ュ ー			参照章番号
単語登録	単語登録		1.19.1
	単語削除		1.19.2
	辞書マウント / アンマウント		1.20
環境設定	変換方式	連文節変換	1.21
		逐次自動変換	1.21
	サーバ操作	サーバの切り離し	1.22.1
		サーバの切り替え	1.22.2
		サーバの表示	1.22.3
	辞書マウント / アンマウント		1.20
	学習状態表示		1.23
	バージョン表示		1.24
	ファイル表示	ローマ字かな変換テーブル	1.25.1
		カスタマイズファイル	1.25.2

日本語入力中に上記の各拡張機能を実行するには以下のようなキー操作を行います。

［例］単語登録を行う場合。

(1) 日本語入力モードで読みを入力していない状態で HELP を押すと、拡張機能の一覧が表示されます。

```
% ■
[拡張] 1 記号入力  2 コード入力  3 部首入力  4 単語登録  5 環境設定          1/5
```

(2) **単語登録** を選択 (→ および ← でカーソル移動後 RETURN または番号の入力) すると、以下のような一覧が表示されます。

```
% ■
[拡張] [1] 単語登録　2 単語削除　3 辞書マウント / アンマウント　1/3
```

(3) **単語登録** を選択 ([→] および [←] でカーソル移動後 [RETURN] または番号の入力) します。

このように機能を選択することにより各機能が開始します。選択後については各機能の詳細で述べます。

拡張機能の項目はカスタマイズすることもできます。詳細は、**2　カスタマイズ** を参照してください。以下の各節では、デフォルトの拡張機能について説明します。

1.18 記号入力

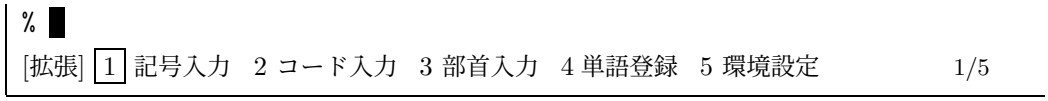
記号入力には、記号全般、ロシア文字、ギリシャ文字、および罫線入力があり、それぞれの一覧を表示します。記号全般については、**1.14.1 記号入力モードでの入力** も参照してください。

1.18.1 記号全般

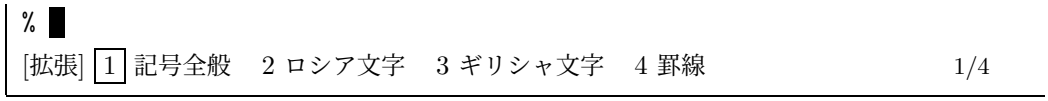
スクリーンの最下行などに記号一覧を表示します。カーソルの上下左右への移動を行うことにより、求める記号を容易に選択することができます。また、いくつでも連続して入力することができます。

ここでは、記号の入力の仕方、および記号入力モードからの取り止め処理について説明します。

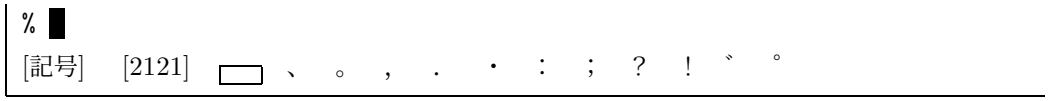
(1) 日本語入力モードで読みを入力していない状態で HELP を押すと、拡張機能の一覧が表示されます。



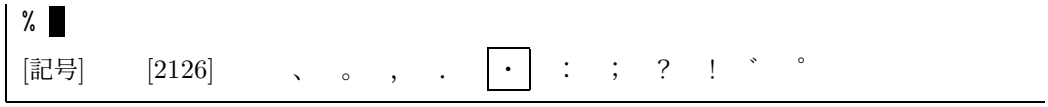
(2) **記号入力** を選択すると、以下のような一覧が表示されます。



(3) **記号全般** を選択します。



(4) → を押して候補を選択します。



(5) RETURN を押すと確定されます。

```
% ・
[記号]      [2126]      、  。  ,  .  ・  :  ;  ?  !  ゛  ゜
```

(6) 続けて、『〜』を選択し (→ および ← でカーソル移動)、RETURN を押すと『〜』が確定されます。

```
% ・〜
[記号]      [2141]  々  〆  〇  ー  ―  -  ／  ＼  〜  ‖  |  …
```

(7) 記号入力モードを抜け、 変換入力モードに戻るためには、CTRL + g を押します。

```
% ・〜
[ あ ]
```

(8) (3) の状態から、記号を 1 つも選択しないで、記号入力モードを抜けるために CTRL + g を押すと、 1 つ前の画面に戻ります。

```
%
[拡張] 1 記号全般  2 ロシア文字  3 ギリシャ文字  4 罫線                1/4
```

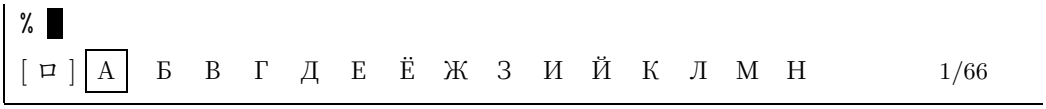
1.18.2 ロシア文字

ロシア文字の候補一覧を表示します。候補選択、確定および取り止めの方法は記号入力と同じです。

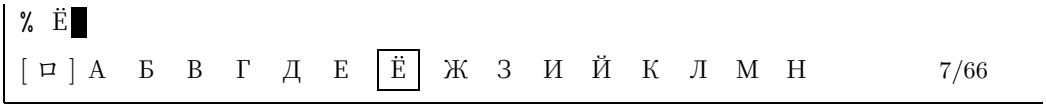
(1) 『Ë』をロシア文字入力で入力します。

```
%
[ あ ]
```

(2) HELP を押し、**記号入力→ロシア文字** を選択します。



(3) ロシア文字の一覧が表示されますので、『Ё』を選択し (→ および ← でカーソル移動) RETURN を押すと『Ё』が確定されます。



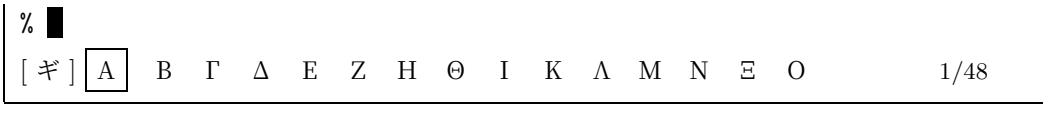
(4) ロシア文字入力モードを抜け、 変換入力モードに戻るためには、CTRL + g を押します。

1.18.3 ギリシャ文字

ギリシャ文字の候補一覧を表示します。候補選択、確定および取り止めの方法は記号入力と同じです。

(1) 『Θ』をギリシャ文字入力で入力します。

(2) HELP を押し、**記号入力→ギリシャ文字** を選択します。



(3) ギリシャ文字の一覧が表示されますので、『Θ』を選択し (→ および ← でカーソル移動) RETURN を押すと『Θ』が確定されます。

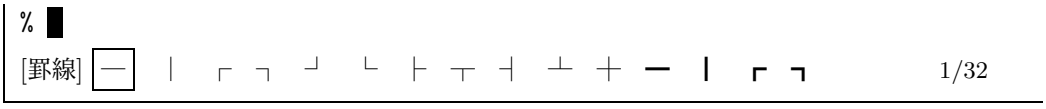
(4) ギリシャ文字入力モードを抜け、 変換入力モードに戻るためには、CTRL+ g を押します。

1.18.4 罫線

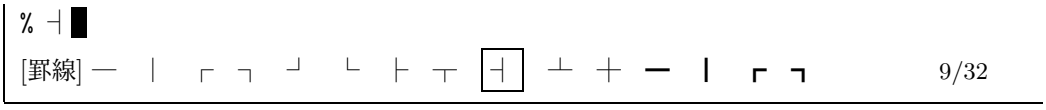
罫線の候補一覧を表示します。候補選択、確定および取り止めの方法は記号入力と同じです。

(1) 『┤ 』を罫線入力で入力します。

(2) HELP を押し、**記号入力→罫線** を選択します。



(3) 罫線の一覧が表示されますので、『┤』を選択し (→ および ← でカーソル移動) RETURN を押すと『┤』が確定されます。



(4) 罫線入力モードを抜け、 変換入力モードに戻るためには、CTRL+ g を押します。


% ㄐ
[あ]

1.19 単語登録削除

1.19.1 単語登録

日本語入力機能には、変換をより早く的確に行えるようにするために、単語登録機能があります。本節では、この単語登録の方法を記述します。

ただし、単語が登録できるのは読みがない状態のときのみで、読みが入力されている状態では、単語の登録はできません。

1.19.1.1 品詞の種類

単語は、品詞ごとに分けて登録します。

登録する際必要な品詞の種類と例を以下に示します。

品　　詞	例
名詞	山、青空、日本語
固有名詞	日本電気
人名	鈴木
地名	東京
動詞	遊ぶ、急ぐ、扱う
形容詞	早い、美しい
形容動詞	感心だ、大慌てだ
副詞	ふっくら、突然
単漢字	島、村
数詞	いくら、第二、三人
連体詞	あの、いわゆる
接続詞	そして、また
感動詞	ああ、おや、いいえ

1.19.1.2 単語登録

以下のようにして単語を登録します。ただし、カスタマイズファイルで :user で指定されている辞書が 1 つもマウントされていない場合は単語登録は行えません。:user で指定されている辞書が 1 つもマウントされていないが、マウントに失敗した辞書が存在する場合、単語登録を行う際にその辞書を作成するか否かを問い合わせてきます。「y」と答えると辞書を作成した後に以下の単語登録を行います。「n」と答えると辞書を作成せず、単語登録は行えません。

(1) 「第 6 開発部」を「ろっかい」で登録します。ただし、カスタマイズファイルに :user user と指定された辞書があり、マウントに失敗しているものとします。

```
% 
[ あ ]
```

(2) HELP を押し、**単語登録→単語登録** を選択します。

```
% ■
[登録] 単語登録用辞書がありません。辞書 (user) を作成しますか?(y/n)
```

(3) **y** を入力すると user という辞書が作成され、単語登録が行えるようになります。

```
% ■
[登録] 単語?[]
```

(4) 登録する単語を入力します。動詞、形容詞、形容動詞は終止形で入力し、ほかはすべて語幹で入力します。

```
% ■
[登録] 単語?[第 6 開発部]
```

(5) RETURN を押すと、読みの入力要求がされます。

```
% ■
[登録] 単語 [第 6 開発部] 読み?[]
```

(6) 次に、読みを入力します。

```
% ■
[登録] 単語 [第 6 開発部] 読み?[ろっかい]
```

(7) RETURN を押すと、品詞の候補が表示されます。

```
% ■
[品詞] 1 名詞  2 固有名詞  3 動詞  4 形容詞  5 形容動詞  6 副詞          1/7
```

(8) **名詞** を選択 (→ および ← でカーソル移動後 RETURN または番号の入力) します。このとき、選択した品詞によってはさらに細かい品詞分けをするかどうか質問されます。この質問はカスタマイズによって抑制することもできます。カスタマイズについては **2　カスタマイズ** を参照してください。

```
% ■
[品詞] さらに細かい品詞分けのための質問をしても良いですか？ (y/n)
```

(9) **y** と答えると、質問が表示されます。

```
% ■
[品詞] 「第 6 開発部する」は正しいですか?(y/n)
```

(10) 「第 6 開発部する」は不自然なので **n** と答えます。このとき、選択した品詞によっては次の質問が表示されることがあります。

```
% ■
[品詞] 「第 6 開発部な」は正しいですか?(y/n)
```

(11) 「第 6 開発部な」は不自然なので **n** と答えると、登録する辞書の候補が表示されます。

```
% ■
[辞書] 1 user
```

(12) 登録する辞書を選択する (→ および ← でカーソル移動後 RETURN または番号の入力) と、メッセージが出力されて登録が完了します。

```
% ■
[ あ ] 『第 6 開発部』 (ろっかい) を登録しました
```

(13) (3) の状態で、単語を入力しないで RETURN を押すと、以下のメッセージが表示されます。メッセージが表示されたら、適当なキーを押して (3) の状態に戻し、単語を入力します。

```
% ■
[登録] 単語を入力してください
```

(14) (5) の状態で、読みを入力しないで RETURN を押すと、以下のメッセージが表示されます。メッセージが表示されたら、適当なキーを押して (5) の状態に戻し、読みを入力します。

```
% ■
[登録] 読みを入力してください
```

1.19.2 単語削除

(1) 単語が「アプリケーションプログラム」で読みが「えーぴー」のエントリを辞書 user と iroha から削除します。

```
% ■
[ あ ]
```

(2) HELP を押し、**単語登録→単語削除** を選択します。

```
% ■
[削除] 読み?[]
```

(3) 削除する単語の読みを入力します。

```
% ■
[削除] 読み?[えーぴー]
```

(4) RETURN を押すと、登録されている単語の一覧が表示されます。

```
% ■
[削除] 1 アプリケーションプログラム　2 AP                    1/2
```

(5) **アプリケーションプログラム** を選択する (→ および ← でカーソル移動後 RETURN または番号の入力) と、選択した候補が登録されている辞書で、単語削除可能な辞書の一覧が表示されます。このとき、選択した候補が 1 つの辞書のみに登録されている場合は、辞書の一覧は表示されません。

```
% ■
[削除] ○ user　○ kimura　○ tanshuku　○ iroha                1/4
```

(6) 辞書一覧では、選択されている辞書は ◎、選択されていない辞書は ○ で示されます。

辞書 user と iroha から単語を削除するので、まず、○ **user** にカーソルを移動 (→ および ← でカーソル移動) し、SPACE を押します。すると、表示が ◎ になり、辞書 user が選択されます (もう一度 SPACE を押すと、表示が ○ になり、選択されなくなります)。

```
% ■
[削除] ◎ user　○ kimura　○ tanshuku　○ iroha                1/4
```

(7) 辞書を複数選択することができます。

辞書 iroha からも単語を削除するので、○ **iroha** にカーソルを移動 (→ および ← でカーソル移動) し、SPACE を押します。

```
% ■
[削除] ◎ user　○ kimura　○ tanshuku　◎ iroha                4/4
```

(8) 辞書の選択が終了したら、RETURN を押すと、メッセージが表示されます。メッセージがガイドラインに入り切らない場合には、プロンプトの表示されている場所にメッセージが表示されます。

```
% 『アプリケーションプログラム』 (えーぴー) を辞書 user と iroha から削除しますか?(y/n)
[削除]
```

(9) **y** を入力すると、メッセージが表示されて単語削除が完了します。

n を入力すると、単語は削除されません。

```
% 『アプリケーションプログラム』 (えーぴー) を辞書 user と iroha から削除しました
[ あ ]
```

(10) (2) の状態で、読みを入力しないで RETURN を押すと、以下のメッセージが表示されます。メッセージが表示されたら、適当なキーを押して (2) の状態に戻し、読みを入力します。

```
%
[削除] 読みを入力してください
```

(11) (5) の状態で、辞書を 1 つも選択しないで RETURN を押すと、以下のメッセージが表示されます。メッセージが表示されたら、適当なキーを押して (5) の状態に戻し、辞書を選択します。

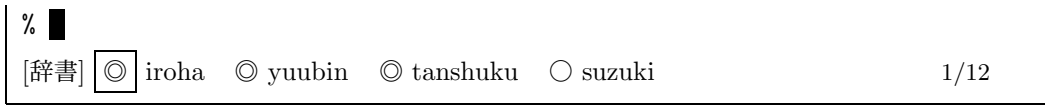
```
%
[削除] 辞書を選択してください
```

1.20 辞書マウント / アンマウント

辞書をマウントおよびアンマウントします。ただし、ここでマウントした辞書は単語登録の対象にはなりません。

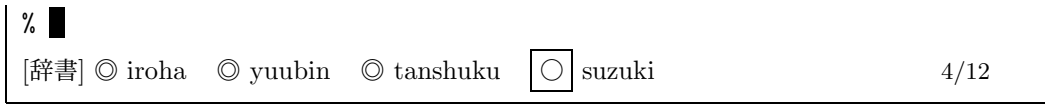
(1) 辞書 suzuki をマウントし、辞書 tanshuku をアンマウントするとします。

(2) HELP を押し、**環境設定→辞書マウント / アンマウント** または **単語登録→辞書マウント / アンマウント** を選択します。



(3) 現在マウント可能な辞書の一覧が表示されます。現在マウントされていない辞書は○、すでにマウントされているものは◎で示されます。

辞書 suzuki は○なのでマウントされていません。辞書 suzuki をマウントしますので、○ suzuki にカーソルを移動 (→ および ← でカーソル移動) します。



(4) SPACE を押すと、表示が◎になり、マウントが指定されます (もう一度 SPACE を押すとアンマウントが指定されます)。ここでは設定を行っているだけで、実際のマウント / アンマウントは行われません。

(5) 辞書 tanshuku は◎なのでマウントされています。辞書 tanshuku をアンマウントするので、◎ tanshuku にカーソルを移動 (→および←でカーソル移動) します。

```
% ■
[辞書] ◎ iroha  ◎ yuubin  ◎ tanshuku  ◎ suzuki  3/12
```

(6) SPACE を押して、アンマウントを指定します。

```
% ■
[辞書] ◎ iroha  ◎ yuubin  ○ tanshuku  ◎ suzuki  3/12
```

(7) 各辞書のマウント / アンマウントを指定した後、RETURN を押すと、その指定にしたがって辞書のマウント / アンマウントが行われます。

```
% ■
[ あ ]  辞書のマウント / アンマウントを行いました
```

1.21 変換方式の変更

変換方式を変更したい場合は、拡張機能を利用します。

変換方式には、**連文節変換** と **逐次自動変換** があります。

(1) HELP を押し、**環境設定→変換方式** を選択します。

```
%
[拡張] 1 連文節変換　2 逐次自動変換
```

(2) 連文節変換に変更したい場合は、**連文節変換** を選択します。

```
%
[ あ ] 連文節変換に切替えました
```

(3) 逐次自動変換に変更したい場合は、**逐次自動変換** を選択します。

```
%
[逐次] 逐次自動変換に切替えました
```

(4) 逐次自動変換をサポートしていない古いサーバに接続している場合は、逐次自動変換への切り替えはできず、以下のメッセージが表示されます。

```
%
[逐次] サーバが逐次自動変換をサポートしていません
```

1.22 サーバ操作

1.22.1 サーバの切り離し

アプリケーションプログラムを終了させることなくかな漢字変換サーバとの接続を切ります。辞書のメンテナンスをするときなどに使います。

次に日本語入力モードで何らかのキーを入力しようとすると自動的に接続が行われます。

(1) HELP を押し、**環境設定→サーバ操作→サーバの切り離し** を選択します。

```
% ■
[ あ ] かな漢字変換サーバとの接続を切りました
```

1.22.2 サーバの切り替え

アプリケーションプログラムを終了させることなく別のかな漢字変換サーバと接続し直します。

(1) ws1 のかな漢字変換サーバとの接続を切り、 ws2 のかな漢字変換サーバと接続するとします。

```
% ■
[ あ ]
```

(2) HELP を押し、**環境設定→サーバ操作→サーバの変更** を選択します。

```
% ■
[変更] マシン名?[]
```

(3) マシン名を入力します。

```
% ■
[変更] マシン名?[ws2]
```

(4) RETURN を押すと、 ws2 のかな漢字変換サーバに接続します。

```
% ■
[ あ ] ws2 のかな漢字変換サーバに接続しました
```

1.22.3 サーバの表示

現在接続しているかな漢字サーバが動いているホストの名前を表示します。

(1) HELP を押し、**環境設定→サーバ操作→サーバの表示** を選択すると、接続しているホストの名前が表示されます。

```
%
[ あ ] ws1 のかな漢字変換サーバに接続しています
```

特に設定を変更していなければ、自分自身のかな漢字変換サーバに接続しています。その場合はホスト名は unix になります。

```
%
[ あ ] unix のかな漢字変換サーバに接続しています
```

1.23 学習状態表示

現在の学習の設定を調べます。

(1) HELP を押し、**環境設定→学習状態表示** を選択します。
学習している場合は、以下のメッセージが出力されます。

```
%
[ あ ] 学習が ON の状態です
```

(2) 学習していない場合は、以下のメッセージが出力されます。

```
%
[ あ ] 学習が OFF の状態です
```

学習状態の切り替えは、カスタマイズファイルの修正にて行います。**2.6.4 カスタマイズのキーワードの (12) そのほかのキーワード** を参照してください。

1.24 バージョン表示

現在使用しているかな漢字変換システムのバージョンを調べます。

(1) HELP を押し、**環境設定→バージョン表示** を選択します。

バージョンが以下のように出力されます (『かんな』 Version 3.3 の場合)。

```
%
[ あ ] 日本語入力システム『かんな』 Version 3.3 product version
```

1.25 ファイル表示

1.25.1 ローマ字かな変換テーブルの表示

現在使用しているローマ字かな変換テーブルを調べます。

(1) HELP を押し、**環境設定→ファイル表示** を選択します。

```
%
[拡張] 1 ローマ字かな変換テーブル　2 カスタマイズファイル　　1/2
```

(2) ローマ字かな変換テーブルを表示したい場合は、**ローマ字かな変換テーブル** を選択します。

```
%
[ あ ] ローマ字かな変換テーブルは default.kp を使用しています
```

1.25.2 カスタマイズファイルの表示

現在使用しているすべてのカスタマイズファイルを調べます。

(1) HELP を押し、**環境設定→ファイル表示** を選択します。

```
%
[拡張] 1 ローマ字かな変換テーブル　2 カスタマイズファイル　　2/2
```

(2) カスタマイズファイルを表示したい場合は、**カスタマイズファイル** を選択します。

```
%
[ あ ] カスタマイズファイルは default.canna を使用しています
```

(3) カスタマイズファイルを 2 つ以上使用している場合は、すべてのカスタマイズファイルがカンマ ”,”で区切られて表示されます。

メッセージが長くなる場合は、候補一覧を表示する行にメッセージが表示しきれなくなる場合があります。その場合は、プロンプトの表示されている場所にメッセージが表示されます。

```
% カスタマイズファイルは /usr/local/canna/lib/default.canna,/usr/local/canna/lib
/kterm.canna を使用しています
[ あ ]
```

第 2 章

カスタマイズ

かな漢字変換に関して変換に利用する辞書を始めとしてキー操作やローマ字かな変換テーブルにいたるまでユーザの好みに合うように カスタマイズ することができます。

カスタマイズに用いられるファイルとしては、以下のファイルがあります。

日本語入力の機能やキー操作を設定する	カスタマイズファイル
ローマ字かな変換の規則を設定する	ローマ字かな変換ソース (×××.kpdef)
	ローマ字かな変換テーブル (×××.kp)

ここではそれぞれのユーザに合わせたカスタマイズ方法について説明します。

2.1 カスタマイズファイル

カスタマイズはカスタマイズファイルを用いて行います。カスタマイズファイルは特別の指定がないかぎりホームディレクトリにある .canna というファイルが用いられます。

2.1.1 カスタマイズのための初期設定

カスタマイズを行うにはまず $(CANNALIBDIR)/sample/default.canna を自分のホームディレクトリに持ってきて `.canna` という名前でセーブします。

```
% cp /usr/lib/canna/sample/default.canna .canna
% chmod u+w .canna
```

すでに .canna を持っている人は上記の操作は不要です。

2.1.2 カスタマイズファイルの指定

環境変数の設定により参照するカスタマイズファイルを変更することができます。

カスタマイズファイルは以下の順にサーチされます。

(1) 環境変数 CANNAFILE が設定されているとき

環境変数 CANNAFILE によって示されるファイルが存在すれば、そのファイルがカスタマイズファイルとして使用されます。

(2) $HOME/.canna が存在するとき

そのファイルがカスタマイズファイルとして使用されます。

(3) 上記のいずれでもないとき

$(CANNALIBDIR)/default.canna がカスタマイズファイルとして用いられます。

また、(2), (3) の場合はそれぞれ次のファイルも読み込まれます。

- (2) の場合

  (2.1) 環境変数 DISPLAY に値が設定されており、その値が ×××:0 であるときに、ホームディレクトリに .canna-××× という名前のファイルが存在する場合、さらにこのファイルも読み込まれます。

  (2.2) 環境変数 TERM に値が設定されており、その値が ××× であるときに、ホームディレクトリに .canna-××× という名前のファイルが存在する場合、さらにこのファイルも読み込まれます。

- (3) の場合

  (3.1) 環境変数 DISPLAY に値が設定されており、その値が ××× :0 であるときに $(CANNALIBDIR) に ×××.canna という名前のファイルが存在する場合、さらにこのファイルも読み込まれます。

  (3.2) 環境変数 TERM に値が設定されており、その値が ××× であるときに $(CANNALIBDIR) に ×××.canna という名前のファイルが存在する場合、さらにこのファイルも読み込まれます。

以上をまとめるとカスタマイズファイルのサーチ順は 表 2.1 のとおりになります。

表 2.1: カスタマイズファイルのサーチ順

	ファイル
1	CANNAFILE で指定したファイル
2	$HOME/.canna $HOME/.canna-X のディスプレイ名 $HOME/.canna-term 名
3	$(CANNALIBDIR)/default.canna $(CANNALIBDIR)/X のディスプレイ名.canna $(CANNALIBDIR)/term 名.canna

2.1.3 サンプルとして提供するカスタマイズファイル

『かんな』以外のフロントエンドプロセッサになれていらっしゃる方もいるでしょう。 そのような方のために、システムでは、他のフロントエンドプロセッサの操作性を有したカスタマイズファイル[注 1] を提

供しています。

サンプルとして提供されているカスタマイズファイルとしては表 2.2 があります。これらのサンプルファイルは $(CANNALIBDIR)/sample の下にファイルの形で存在します。**注 2**

default.canna がデフォルトの設定をカスタマイズファイルで表したものになっています。

表 2.2: サンプルとして提供するカスタマイズファイル

ファイル名	備　　考
default.canna	デフォルトの設定をカスタマイズファイルで表したもの
sample.canna	カスタマイズの方法の例を記述したカスタマイズファイル
just.canna	一太郎[1]とほぼ同じキー操作を提供するカスタマイズファイル
matsu.canna	松茸[2]とほぼ同じキー操作を提供するカスタマイズファイル
vje.kp	vje−β[3]とほぼ同じキー操作を提供するカスタマイズファイル
wx2+.kp	WXII+[4]とほぼ同じキー操作を提供するカスタマイズファイル

サンプルのカスタマイズファイルを使用したい場合は、環境変数 CANNAFILE を設定する (1) か、自分のホームディレクトリ にサンプルファイルを.canna としてコピーして使用する (2) とよいでしょう。just.canna を使用したい場合は次のいずれかを実行します。

(1) `% setenv CANNAFILE /usr/lib/canna/sample/just.canna`

(2) `% cp /usr/lib/canna/sample/just.canna ~/.canna`

注 1 カスタマイズファイルは、できるかぎり各々のフロントエンドプロセッサのキー操作と同じになるように設定してありますが、 X ウィンドウにおいてのみ機能するキー (表 2.10 参照) なども存在するため、必ずしも、全く同じ操作方法が実現できるわけではありません。

注 2 サンプルファイルは、 cannasmp パッケージに含まれています。 $(CANNALIBDIR)/sample が存在しない場合は、 cannasmp パッケージをインストールしてください。

2.1.4 カスタマイズファイルの例

次頁にカスタマイズファイルの例 を示します。

カスタマイズファイルは Lisp 言語と同様のシンタックスにより記述します。また、「;」 (セミコロン) から行末まではコメントとみなされます。

[1] 一太郎はジャストシステム (株) の商標です

[2] 松茸は (株) 管理工学研究所の商標です

[3] VJE−β は (株) バックスの商標です

[4] WXII+ はエー・アイ・ソフト (株) の商標です

```
;;
;; カスタマイズファイルの例
;;
(setq cursor-wrap                t)    ; カーソルが循環するようにする。
(setq gakushu                    t)    ; 学習するようにする。
(setq kakutei-if-end-of-bunsetsu t)    ; 文節の最後で右に行くと確定する。
(setq break-into-roman           t)    ;BS でいったんローマ字に戻す。
(setq grammatical-question       nil)  ; 単語登録のとき品詞の質問をしない。
(setq kouho-count                t)    ; 候補一覧のとき 1/25 などの表示をする。
(setq n-henkan-for-ichiran       3)    ; 変換キー 3 回で一覧を出すようにする。

; 使用するローマ字かな変換テーブルの設定
(setq romkana-table   "default.kp")

; 使用するかな漢字変換辞書の設定
(use-dictionary
   "iroha"
   "fuzokugo"
   "hojoswd"
   :bushu  "bushu"      ; 部首変換用辞書
   )
; ローマ字かな変換の多候補設定
(defsymbol
  ?[" 「 " " 『 " " [ " " 〔 " " [ "
  ?]" 」 " " 』 " " ] " " 〕 " " ] ")

(defsymbol
  ?. " 。 " " . " " ."
  ?, " 、 " " , " " ,")

;; キーの定義
(global-set-key "\F1" 'kigou-mode)
(global-set-key "\F2" 'hex-mode)
(global-set-key "\F3" 'bushu-mode)
```

2.2 使用する辞書の指定

本日本語入力システムでは複数の辞書を同時に利用することができます。利用する辞書の指定方法は、その辞書をどのように利用するかにより、表 2.3に示す 4 種類があります。

注意 テキスト形式辞書、バイナリ形式辞書については **1.1.2 かな漢字変換辞書** および **3.2 辞書** を参照してください。

使用する辞書の指定はカスタマイズファイルの中で次のように行います。

```
(use-dictionary   "iroha"
                  "fuzokugo"
```

表 2.3: 辞書指定キーワード

キーワード	辞書の種類	備考
—	システム辞書	システム辞書として用いる辞書を指定します。この辞書には単語の登録は行えません。
:bushu	部首変換用辞書	部首変換に用いる辞書を指定します。
:grammar	文法情報用辞書	品詞の定義、品詞間の接続情報を持つ辞書を指定します。
:user	単語登録用辞書	単語登録用の辞書を指定します。この辞書には単語を登録することができます。この辞書の形式はテキスト形式でなければなりません。注

```
                "hojomwd"
                "hojoswd"
                "yuubin"
  :bushu        "bushu"
  :grammar      "grammar"
  :user         "mine" )
```

注意 use-dictionary は一度の記述でたくさんの辞書を指定しても (1)、複数回の記述に分けて、辞書を指定しても (2) 同じ効果が得られます。すなわち、以下の (1) と (2) は同じ意味を持ちます。

(1) (use-dictionary ”A” ”B” ”C”)
(2) (use-dictionary ”A”)
(use-dictionary ”B”)
(use-dictionary ”C”)

指定できるシステム辞書としては表 2.4に示すものがあります。

表 2.4: 指定できる辞書

辞 書 名	説　　明
iroha	基幹辞書
fuzokugo	付属語辞書
bushu	部首辞書
yuubin	郵便番号辞書
hojomwd	口語表現などの補助的な自立語
hojoswd	口語表現などの補助的な付属語

これらの辞書のうち、 iroha が基本的な辞書になります。また、 iroha を使用するときは fuzokugo も必ず指定しなければなりません。したがって最低限使用する辞書は iroha および fuzokugo であり、次のように指定することになります。

```
(use-dictionary  "iroha"  "fuzokugo" )
```

郵便番号辞書は郵便番号から地名への変換を行うための辞書です。必要に応じてご利用ください。

hojomwd,hojoswd は補助的に追加された付属語の辞書です。口語表現などはこの辞書を使用することにより変換し易くなります。必要に応じてご利用ください。

上記の 3 つの辞書は以下のように指定することにより利用できます。

```
(use-dictionary  "yuubin"  "hojomwd"  "hojoswd" )
```

bushu は部首変換を行うときに使われる辞書です。部首変換を利用する場合はこの辞書を使うことを以下のようにして指定してください。辞書の指定のためのキーワード :bushu が必要ですので、ご注意ください。

```
(use-dictionary  :bushu  "bushu" )
```

品詞の定義および品詞間の接続情報を持つ辞書を指定するためには、キーワード :grammar を用います。:grammar で指定された辞書がない場合、および :grammar で指定された辞書に付属語情報が記述されていない場合は、 fuzokugo.d に記述されている文法情報を使用します。また、:grammar が 2 つ以上指定された場合は、後に指定された辞書を使用します。:grammar の指定は以下のように行います。

```
(use-dictionary :grammar "myGrammarDictionary" )
```

ユーザ自身が追加登録した単語を納めておくための辞書を指定するためには、キーワード :user を用います。単語登録の際、登録された単語をどの辞書に納めるかを尋ねられますが、そのとき選択できる辞書は :user で指定した辞書に限られます。

:user で指定する辞書は単語登録を行う関係上、単語登録用辞書として作成されたテキスト形式の辞書でなければなりません[注]。:user の指定は以下のように行います。

```
(use-dictionary :user "myTextDictionary" )
```

注意 テキスト形式辞書、バイナリ形式辞書については **1.1.2 かな漢字変換辞書** および **3.2 辞書** を参照してください。

単語登録用辞書は mkdic コマンドで作成できます。最初は空のファイルを指定しておき、順次辞書登録により単語を追加していくのがよいでしょう。

参考用としてシステムファイル ($(CANNALIBDIR)/dic/user/user) に含まれている辞書を表 2.5に示します。単語登録用辞書を作成するときの参考にしてください。

表 2.5: $(CANNALIBDIR)/dic/user/user ディレクトリにあるファイル

辞 書 名	説　　明
katakana	カタカナ語
software	ソフトウェア語
chimei	地名

注意 デフォルトカスタマイズファイル ($(CANNALIBDIR)/default.canna) では、単語登録用辞書としてあらかじめ ”user” が指定されています。

2.3 ローマ字かな変換テーブルの設定

使用したいローマ字かな変換テーブルを記述します。

```
(setq romkana-table  "romaji.kp")
```

romkana-table に指定するものはファイル名です。指定されたファイル名をカレントディレクトリ、ユーザのホームディレクトリ、$(CANNALIBDIR)/dic の順にサーチし、最初に見つかったファイルがローマ字かな変換テーブルとして用いられます。

2.4 ローマ字かな変換テーブルのカスタマイズ

ローマ字かな変換に関して、たとえば「こんにちは」と入力するのに、「konnichiha」と入力するのに慣れている場合と、「konnnichiha」と入力する (「n」の回数が違います) のに慣れている場合があります。

このようにローマ字かな変換だけをとってみてもユーザの好みはいろいろあります。

ローマ字かな変換に関してはカスタマイズファイルにより好みのテーブルを指定することができます。また、標準的に用意しているテーブルに好みのものがない場合は、自分でローマ字かな変換テーブルを作成することもできます。

2.4.1 標準的に提供するローマ字かな変換テーブル

標準的に提供されるローマ字かな変換テーブルとしては表 2.6 のものがあります。 これらのテーブルは $(CANNALIBDIR)/dic の下にファイルの形で存在します。

表 2.6: 提供するローマ字辞書テーブル

辞書名	備　　考
default.kp	デフォルトのローマ字かな変換テーブル
just.kp	一太郎[5]と同一のローマ字かな変換規則を持つローマ字かな変換テーブル
matsu.kp	松茸[6]と同一のローマ字かな変換規則を持つローマ字かな変換テーブル
vje.kp	vje−β[7]と同一のローマ字かな変換規則を持つローマ字かな変換 テーブル
wx2+.kp	WXII+[8]と同一のローマ字かな変換規則を持つローマ字かな変換テーブル
kana.kp	アルファベットキーボードで疑似的にかな入力を行うためのテーブル。このテーブルを用いることにより、日本語モードとアルファベットモードの切り替えだけで、かな入力とアルファベット入力の切り替えも行えます。
newjis.kp	アルファベットキーボードで疑似的に新 JIS キーボードのカナ配列をシミュレートするためのテーブル
kaisoku.kp	快速ローマ字配列をシミュレートするためのテーブル

デフォルトでは default.kp が選ばれますが、そのほかのローマ字かな変換テーブルを利用する場合はカスタマイズファイルに次のような行を加えます。

```
;kana.kp を利用する場合
(setq romkana-table "kana.kp")
```

指定した名前はファイル名とみなされ、次の順にサーチされます。

(1) カレントディレクトリ
　　指定されたファイルをカレントディレクトリでサーチします。
(2) ホームディレクトリ
　　指定されたファイルをホームディレクトリでサーチします。
(3) 辞書ディレクトリ
　　指定されたファイルを $(CANNALIBDIR)/dic でサーチします。

いずれのディレクトリにもローマ字かな変換テーブルが存在しない場合はローマ字かな変換入力が行えませんのでご注意ください。

2.4.2 ローマ字かな変換テーブルの作成方法

システムで提供されているローマ字かな変換テーブルでは満足できない場合は、ローマ字かな変換テーブルをカスタマイズすることができます。

[5] 一太郎はジャストシステム (株) の商標です
[6] 松茸は (株) 管理工学研究所の商標です
[7] VJE−β は (株) バックスの商標です
[8] WXII+ はエー・アイ・ソフト (株) の商標です

カスタマイズする場合は、デフォルトのローマ字かな変換テーブルのソースファイルである $(CANNALIBDIR)/sample/src/default.kpdef をコピーして書き換えるのが簡単です。ローマ字かな変換テーブルは 1 行にローマ字とそれに対応する『かな』とが記述されているテーブルです。表 2.7 のような内容になっています。

表 2.7: ローマ字かな変換テーブルの内容

a	あ	
i	い	
u	う	
e	え	
o	お	
ka	か	
ki	き	
	・・・・	
n	ん	
n’	ん	
mn	ん	
nn	ん	
	・・・・	
tch	っ	ch
kk	っ	k
ss	っ	s
tt	っ	t
hh	っ	h

「同じ子音が重なった場合は促音を発生する」といった規則もローマ字かな変換テーブルに記述します。たとえば、以下のように記述します。

```
kk      っ      k
```

これは、 kk と入力が来たときは「っ」を発生させ、 k を次の入力とつなげるために残しておくという意味です。

この規則を用いて、「まっち」と入力するときに「matchi」のように入力するためのルールを以下のように記述することができます。

```
tch     っ      ch
```

特殊な文字を指定するときのためにバックスラッシュ (\) をエスケープ文字として使用することができます。特殊な文字としては、スペース文字、シャープ (#), バックスラッシュ (\) があります。

たとえば、バックスラッシュ (\) を入力したときに ¥ が入力されるようにする場合は、以下のように記述します。

\\ ¥

2.4.2.1 テーブルに記述されている規則の用いられ方

ローマ字かな変換テーブルを基にローマ字かな変換が行われますが、ローマ字かな変換テーブルで記述された変換規則は、最長一致法の原則で使用されます。たとえば、「べんりな」と入力するときは 図 2.1 のようにテーブルが参照されます。

図 2.1: 最長一致法の原則

入力	表 示	備 考
ben	べ n	「n」についてはもっと長い規則である「na」のようなものが存在するのでローマ字のままエコーされます。
r	べん r	「nr」で始まる規則がないので「n」→「ん」の規則が適用されます
i	べんり	
n	べんり n	
a	べんりな	「na」で始まる規則が「na」→「な」だけであるのでこの規則が適用されます。

2.4.2.2 ローマ字かな変換テーブルのコンパイル

ローマ字かな変換テーブルはシステムが読み易いバイナリ形式に変換しなければなりません。すなわち、上記のようなテキスト形式のファイルをバイナリ形式にコンパイルする必要があります。

次のようにして、ローマ字かな変換テーブルをテキスト形式からバイナリ形式にコンパイルしてください。

% mkromdic ファイル名 RETURN

mkromdic を行うと、∗.kp ファイルができます。

システムで提供されるローマ字かな変換テーブルのソースファイルが $(CANNALIBDIR)/sample/src の下に ∗.kpdef として存在しますので参照してください。

注意 バイナリ形式のローマ字かな変換テーブルは dpromdic コマンドによりテキスト形式に戻すことが可能です。詳細は、**4 かな漢字変換ユーティリティ** を参照してください。

2.5 キー操作のカスタマイズ

たとえば、日本語モードとアルファベットモードを切り替えるキーがデフォルトでは XFER または CTRL + o ですが、利用者はこのキーを好みに応じて変更することができます。カスタマイズファイルを用いて、かな漢字変換に用いられるキーの割り当てをデフォルトとは異なるキーに割り当て直すことができます。

2.5.1 キーの割り当ての有効範囲

かな漢字変換では多くのモードの遷移があります。キーの割り当ては各モードごとに個別に変更することができます。また、すべてのモードのキーの割り当てを一括して変更することもできます。

2.5.2 変更できる機能

キーの割り当ての変更はキー名に対して機能名を記述することにより行います。主な機能名としては 表 2.8 のものがあります。
そのほかの機能名に関しては**付録 C カスタマイズに用いる機能名一覧表** を参照してください。

表 2.8: 主な機能名

名　　前	機　　能	デフォルト	備　考
alpha-mode	アルファベットモードに移行する	Xfer,C-o	
quoted-insert	次の一文字を無条件に入力する	C-q	
kakutei	確定する	Return, Nfer	
extend	領域伸ばし	S- →, C-o	
shrink	領域縮め	S- ←, C-i	
touroku	単語登録	Help	
forward	右選択	→, C-f	
backward	左選択	←, C-b	
previous	前選択	↑, C-p	
next	次選択	↓, C-n	
beginning-of-line	先頭選択	C-a	
end-of-line	末尾選択	C-e	
delete-next	次一文字消去	C-d	
delete-previous	前一文字消去	Backspace	
kill-to-end-of-line	行末まで消去	C-k	
henkan	変換	Space, Xfer	
quit	取りやめ	C-g	
self-insert	文字挿入	a, b, c …	

2.5.3 機能へのキーの割り当て

ある機能に対してキーを割り当てるときは、 global-set-key または set-key を用いて、以下のように記述することにより割り当てます。ある機能に対して複数のキーを割り当てる場合は、 **2.5.4 複数処理でのキーの割り当て** を参照してください。

```
(global-set-key  "キー"    '機能名)
```

```
(set-key  'モード名  "キー"  '機能名)
```

global-set-key では、日本語入力時のすべてのモードに対してキーの割り当てが行われます。 set-key では、特定のモードに対してのみ、キーの割り当てが行われます。

set-key を指定することにより、特定のモードでのキーの割り当てをそのモードのときだけ単独で変更することができます。

キー は `"a"` や `"b"` のように指定します。コントロールキーを押しながら何らかのキーを押すというのはキーの指定の前に `"\C-"` を付けることで表現します。ファンクションキーなどの特殊キー については表 2.9に示す名前で指定することができます。

キーの名前は、大文字・小文字を区別します。ご注意ください。

表 2.9: 特殊キーの種類

キー	名前
f・1(以下 2,3 …)	"\F1", "\F2", ...
ESC	"\Escape"
TAB	"\Tab"
NFER	"\Nfer"
XFER	"\Xfer"
BS	"\Backspace"
INS	"\Insert"
DEL	"\Delete"
ROLLUP	"\Rollup"
ROLLDOWN	"\Rolldown"
↑	"\Up"
←	"\Left"
→	"\Right"
↓	"\Down"
HOME	"\Home"
CLR	"\Clear"
HELP	"\Help"
ENTER	"\Enter"
(SPACE)	"\Space"
(RETURN)	"\Return"

表 2.10: X ウィンドウのみのキー一覧

キー	名前
CTRL+NFER	"\C-Nfer"
CTRL+XFER	"\C-Xfer"
CTRL+ ↑	"\C-Up"
CTRL+ ←	"\C-Left"
CTRL+ →	"\C-Right"
CTRL+ ↓	"\C-Down"
SHIFT+NFER	"\S-Nfer"
SHIFT+XFER	"\S-Xfer"
SHIFT+ ↑	"\S-Up"
SHIFT+ ←	"\S-Left"
SHIFT+ →	"\S-Right"
SHIFT+ ↓	"\S-Down"

また、X ウィンドウにおけるアプリケーションには表 2.10の記述を受けつけるものもあります (TTY ベースの日本語入力では、これらのキー記述は無視されます)。

表 2.11のキーはほかのキーと等価な場合がありますので、どちらか一方のキーに機能を割り当てると、そのキーと等価なキー にも機能を割り当てたことになります。

表 2.11: ＜参考＞等価なキー一覧

キ ー	等 価 な キ ー
CTRL+h (C-h)	BS(Backspace)
CTRL+i (C-i)	TAB(Tab)
CTRL+k (C-k)	HOMECLR(Clear)
CTRL+m (C-m)	ENTER(Enter), RETURN(Return)
CTRL+[(C-[)	ESC(Escape)
CTRL+@ (C-@)	CTRL+SPACE(C+Space)

モード名 は表 2.12に示すものがあります。

表 2.12: キー割り当てのできるモード一覧

名 前	説 明
alpha-mode	アルファベットを入力している状態
empty-mode	日本語入力モードでまだ文字を入力していない状態
yomi-mode	読みを入力している状態
mojishu-mode	文字種変換を行っている状態
tankouho-mode	変換キーを押して候補を表示している状態
ichiran-mode	候補一覧表示をしている状態
henkan-nyuuryoku-mode	empty-mode と yomi-mode の総称
kigou-mode	記号一覧を入力している状態
yes-no-mode	YES/NO を問い合わせる質問をしている状態
on-off-mode	辞書マウントアンマウントを指定している状態
shinshuku-mode	文節を伸ばし縮めしている状態

モード名は、大文字・小文字を区別します。ご注意ください。

2.5.4 複数処理でのキーの割り当て (マルチシーケンス)

2.5.3 機能へのキーの割り当て では、1 つの機能に 1 つのキー操作を割り当てる方法を説明しました。しかし、キーの割り当てはキーと機能を 1 対 1 にしか対応させるだけではなく、多対多対応させることもできます。すなわち、複数の機能を複数のキー操作に割り当てることができます。次のような記述をすることにより割り当てられます。

```
(set-key  'モード名  "キー 1 キー 2 … キー n "
                          '(機能名 1 機能名 2 … 機能名 n))

(global-set-key  "キー 1 キー 2 … キー n "
                          '(機能名 1 機能名 2 … 機能名 n))
```

(例)

```
;C-x スペースと押されたとき、かな漢字変換し、
; 右端の文節が反転されるようにする。
(set-key 'yomi-mode "\C-x\Space"
                        '(henken end-of-line))
```

キーシーケンス入力中にその状態から抜ける場合は quit(取りやめ) を割り当てたキーを押してください。デフォルトでは CTRL + g です。

2.5.5 キーの割り当て解除

ある機能に対するキーの割り当てを未定義状態に戻すときは、 global-unbind-key-function または unbind-key-function を用いて、以下のように記述します。

```
(global-unbind-key-function ' 機能名)

(unbind-key-function ' モード名 ' 機能名)
```

(例)

```
; 全体的に「取りやめ」機能を未定義にする
(global-unbind-key-function 'quit)
; 読みモードで「変換」機能を未定義にする
(unbind-key-function 'yomi-mode 'henkan)
```

2.6 そのほかのカスタマイズ

ここで、カスタマイズファイル (.canna) の シンタックス規則や、カスタマイズファイルで用いられるデータについて説明します。

2.6.1 シンタックスの基本

.canna のシンタックスは Lisp と同じものです。すなわち、「 (」と「) 」で囲まれた部分で 1 つのことがらを表現しています。

通常は、「 (」と「) 」で囲まれたもののうち、左端に記述されているものが関数となっています。

(例)

```
(set-mode-display 'alpha-mode "--")
```

set-mode-display は関数です。

それ以外のもので「 (」と「) 」の間に入っているものは関数の引数です。

(例)
```
(set-mode-display 'alpha-mode "--")
```
'alpha-mode, "-" は引数です。

引数の部分にさらに「 (」と「) 」で囲まれた式を書くことも可能です。

(例)
```
(set-mode-display 'alpha-mode
                  (if use-default " " "--"))
```
use-default という変数の値の真理値によって alpha-mode のモードのモード表示文字列を切り替えています。

引数は一度「評価 」 (evaluate) されてから関数に渡されますが、いくつかの例外もあります。たとえば、 setq, if, defmode などの関数 (正確には関数とは呼ばない Lisp 処理系が多い) では、すべての引数が「評価」されるわけではありません。

ここで「評価」とは一度その式が実行されてその式の値を求めることを指します。

2.6.2 さまざまなデータ

.canna では表 2.13に示す型のデータが登場します。

表 2.13: .canna における評価型

データ型	例
真理値	t, nil
整　数	1, 2, 3, ……
文　字	?a, ?\C-a, ?\Xfer, ……
文字列	”romaji.kp”, ”fuzokugo”, ”[あ]”, ……
シンボル (記号)	forward, next, alpha-mode, ……
リスト	(henkan end-of-line), (katakana kakutei), …

特に、真理値を t(真) と nil(偽) で表現することにご注意ください。

2.6.3 変数

変数はシンボルで表現されます。

変数に値を代入する処理は setqで行われます。

(例)
(setq bunsetsu-kugiri t)
変数 bunsetsu-kugiri に真を代入します。
(setq n-henkan-for-ichiran 5)
変数 n-henkan-for-ichiran に 5 を代入します。

その変数を参照するには変数自身を関数の引数に記述します。

(例)
(if bunsetsu-kugiri (setq select-direct nil))
「文節区切り」が真になっていれば select-direct を偽にします。

シンボルは関数の引数に与えられたときは変数として「評価」され、変数としての内容が得られます。シンボル自身を値として与えたい場合は、「'」(シングルクォート) をシンボルの前につけます。

(例)
(global-set-key "\Space" 'self-insert)
スペースキーは変換ではなく入力として取り扱います。この場合
self-insert というシンボル自身を global-set-key という関数に与えたかったので、
self-insert に対して「'」を付けています。

シンボル以外のデータの「評価」後の値を表 2.14に示します。

表 2.14: シンボル以外のデータ型

データ型	評価後の値
真理値	その値自身
整数	整数データ自身
文字	文字データ自身
文字列	文字列データ自身
シンボル (記号)	変数の中身
リスト	関数を実行したリターン値

表 2.14からわかるように、シンボルとリストに関してはそれらのデータ自身を引数にしたい場合は「'」が必要になります。しかし、それ以外のデータの場合は「'」をつけなくとも構いません。

どのような名前のシンボルでも変数として用いることができます。代入処理を行わないで変数の参照を行うと「Unbound Variable」エラーになります。

2.6.4 (12) そのほかのキーワード に記述されている変数は、かな漢字変換のカスタマイズキーワードとしての意味を持ちます。そのような変数に値を代入する操作は、かな漢字変換処理に影響を与えます。

2.6.4 カスタマイズのキーワード

カスタマイズのためのキーワードはすべてシンボルとしての表記を持っていますが、その実態は関数であったり変数であったりします。以下に概要を示します。

(1) **辞書の利用 − use-dictionary**

```
(use-dictionary "iroha" "fuzokugo")
```

使用する辞書を指定します。

部首辞書、文法辞書、単語登録用辞書などを指定するには辞書の名前の直前に :bushu や:grammar, :user を先行させます。

```
(use-dictionary "iroha" "fuzokugo" :bushu "bushu"
                                   :grammar "mygram" )
                                   :user "mydic" )
```

注意 :bushu, :grammar, :user はシンボルなので本来は ':bushu, ':user のように「'」(シングルクォート) をつけなければならないのですが、「:」(コロン) から始まるシンボル (キーワードシンボルといいます) に限っては不要です。

「(」と「)」の間は適当に改行しても構いません。

```
(use-dictionary
"iroha" "fuzokugo" :bushu"bushu" :grammar
"mygram" :user "mydic")
```

use-dictionary に関する説明は **2.2 使用する辞書の指定** にもありますので参照してください。

(2) **モード表示の変更 − set-mode-display**

```
(set-mode-display 'henkan-nyuuryoku-mode "[変換入力]")
```

入力モードの表示文字列を変更します。結果は次のようになります。

```
% ▉
[変換入力]
```

モード表示文字列を変更しないで、前のモードのモード表示文字列と同じ表示にしたい場合は、文字列の代わりに nil を記述します。

指定できるモードを表 2.15に示します。

アルファベット入力モード (alpha-mode)・変換入力モード (henkan-nyuuryoku-mode) でモード表示文字列を nil に設定した場合、アプリケーションによっては、モード表示文字列が表示されなくなることがあります。アルファベット入力モード、変換入力モードでは nil の記述は避けた方が良いでしょう。

あまり長い文字列を指定すると、アプリケーションによっては文字列の後方が切れて表示されたり、不正動作したりすることがあります。

表 2.15: モード一覧

指　　定	デフォルト	説　　明
alpha-mode		アルファベット入力モード
henkan-nyuuryoku-mode	[あ]	変換入力モード
kigou-mode	[記号]	記号一覧表示状態
yomi-mode	nil	読みを入力している状態
mojishu-mode	[字種]	文字種変換状態
tankouho-mode	[漢字]	単候補表示状態
ichiran-mode	[一覧]	候補一覧表示状態
yes-no-mode	[質問]	利用者に質問をしている状態
on-off-mode	nil	辞書マウントアンマウント状態など
shinshuku-mode	[文節]	文節伸縮モード
chikuji-yomi-mode	[逐次]	逐次自動入力時の読み部分
chikuji-bunsetsu-mode	[逐次]	逐次自動入力時の文節部分
zen-hira-henkan-mode	[全あ]	全角ひらがなモード
han-hira-henkan-mode	[半あ]	半角ひらがなモード
zen-kata-henkan-mode	[全ア]	全角カタカナモード
han-kata-henkan-mode	[半ア]	半角カタカナモード
zen-alpha-henkan-mode	[全英]	全角英数モード
han-alpha-henkan-mode	[半英]	半角英数モード
zen-hira-kakutei-mode	< 全あ >	全角ひらがな確定モード
han-hira-kakutei-mode	< 半あ >	半角ひらがな確定モード
zen-kata-kakutei-mode	< 全ア >	全角カタカナ確定モード
han-kata-kakutei-mode	< 半ア >	半角カタカナ確定モード
zen-alpha-kakutei-mode	< 全英 >	全角英数確定モード
han-alpha-kakutei-mode	< 半英 >	半角英数確定モード
hex-mode	[16 進]	コード入力状態
bushu-mode	[部首]	部首入力モード
extend-mode	[拡張]	ユーティリティモード
russian-mode	[ロ]	ロシア文字一覧状態
greek-mode	[ギ]	ギリシャ文字一覧状態
line-mode	[罫線]	罫線一覧状態
changing-server-mode	[変更]	サーバ変更状態
henkan-method-mode	[変換]	変換方法変更状態
delete-dic-mode	[削除]	単語削除状態
touroku-mode	[登録]	単語登録状態
touroku-hinshi-mode	[品詞]	品詞選択状態
touroku-dic-mode	[辞書]	辞書選択状態
quoted-insert-mode	[q]	引用入力状態
mount-dic-mode	[辞書]	辞書マウント／アンマウント選択状態

(3) キーのバインド – set-key, global-set-key

```
(set-key ’モード名 "キー列 " ’機能列)
(global-set-key "キー列 " ’機能列)
```

set-key では「モード名」で指定されたモードに対してのみキーバインドします。 global-set-key はすべてのモードに対して同様にキーをバインドします。

キー列には 1 文字以上の文字を記述します。キーは `"a"`, `"B"` などと書けるほか、コントロールキーは `"\C-a"` のように `\C-` を先行させて記述することができます。 C-x を押してから C-a を

押した場合に対してキーを割り当てるというように C-x, C-a というシーケンスを表現するときは "\C-x\C-a" と記述します。

Xfer など特殊なキーに関しては以下のように記述します。

```
"\Space"    "\Escape"   "\Tab"      "\Nfer"     "\Xfer"     "\Backspace"
"\Delete"   "\Insert"   "\Rollup"   "\Rolldown" "\Up"       "\Left"
"\Right"    "\Down"     "\Home"     "\Clear"    "\Help"     "\Enter"
"\Return"   "\F1"       "\F2"       "\F3"       "\F4"       "\F5"
"\F6"       "\F7"       "\F8"       "\F9"       "\F10"      "\Pf1"
"\Pf2"      "\Pf3"      "\Pf4"      "\Pf5"      "\Pf6"      "\Pf7"
"\Pf8"      "\Pf9"      "\Pf10"     "\S-Nfer"   "\S-Xfer"   "\S-Up"
"\S-Down"   "\S-Left"   "\S-Right"  "\C-Nfer"   "\C-Xfer"   "\C-Up"
"\C-Down"   "\C-Left"   "\C-Right"
```

機能列には、単一の機能を記述するか複数の機能を「(」と「)」で囲んで指定します。複数の機能を指定した場合は、指定した機能が指定した順に実行されます。指定できる機能名の一覧は**付録 C カスタマイズに用いる機能名一覧表** を参照してください。

set-key, global-set-key については、**2.5 キー操作のカスタマイズ** にも記述がありますので参照してください。

(4) **キーと機能のアンバインド – unbind-key-function, global-unbind-key-function**

```
(unbind-key-function 'モード名 '機能名)
(global-unbind-key-function '機能名)
```

unbind-key-function では「モード名」で指定されたモードに対してのみキーをアンバインドします。 global-unbind-key-function ではすべてのモードに対してキーをアンバインドします。

機能名には機能リストを記述することはできません。

(5) **モード定義 – defmode**

```
(defmode モードシンボル "モード表示" "ローマ字かな変換テーブル"
                        '(機能リスト) 記号利用フラグ)
```

新しいモードを定義します。

新しいモードでは、キーバインド用のそのモード独自のキーマップテーブルと独自のローマ字かな変換テーブルと独自のモード表示文字列などを備えています。

モードシンボルには、そのモードを参照するときのためのシンボルを指定します。以下そのモードを参照するときにはこのモードシンボルで参照されます。モードシンボルは評価されず用いられます。したがって「'」をつける必要はありません。モードシンボルは必ず指定してください。

このシンボルは次のように利用できます。

(a) 「そのモードに移行する」という機能を表す機能名

```
(例)
    (set-key 'henkan-nyuuryoku-mode "\C-t"
                                '定義されたシンボル)
```

(b) そのモードでのみキーを割り当てたり、そのモードに対応するモード文字列を設定したりするときのモード名

```
(例)
    (set-mode-display '定義されたシンボル
                                "[新モード]")
    (set-key '定義されたシンボル
           "\C-t" 'henkan-nyuuryoku-mode)
```

モードシンボル以外の引数は省略可能です。省略した場合は nil が指定されたのと同じ処理が行われます。

機能リストには、 katakana, hiragana, romaji, kakutei, zenkaku, hankaku などが指定できます。これらはローマ字かな変換に伴って実行される機能を表しています。

最後の引数「記号利用フラグ」には、 defsymbol による設定を defmode で作成したモードで使用するかどうかを指定します (defsymbol については、 **(6) 記号定義** を参照してください)。

```
(例)
    ; カタカナ入力モードを定義します。
    (defmode katakana-mode "[カタ]" romkana-table
                            '(katakana) t)
    ;C-k でトグルするようにします。
    (set-key 'henkan-nyuuryoku-mode "\C-k"
                            'katakana-mode)
    (set-key 'katakana-mode "\C-k"
                            'henkan-nyuuryoku-mode)
    ; カナ入力モード (疑似的な) を定義します。
    (defmode kana-input-mode "[カナ]" "kana.kp")
    ;C-r でトグルするようにします。
    (set-key 'henkan-nyuuryoku-mode "\C-r"
                            'kana-input-mode)
    (set-key 'kana-input-mode "\C-r"
                            'henkan-nyuuryoku-mode)
```

(6) **記号定義 – defsymbol**

```
(defsymbol ?[ "「" "『" )
```

defsymbol はローマ字かな変換を補佐する規則を記述します。ローマ字かな変換規則においては入力のキーと変換される「かな」との規則が 1 対 1 でしか対応できませんでしたが、 defsymbol で定義される規則では 1 つのキーに対していくつもの候補を定義することができます。

上の例は "［ " に対して "「 " か "『 " に変換せよというルールを記述した例です。

このようにすることにより、通常は "［ " に対して "「 " が入力されますが、 "［ " だけを入力している状態で変換キーを打つことにより "［ " を "『 " に変換するように切り替えることができます。ひとたび切り替えられると、次からは "『 " が恒常的に入力されるようになります。

defsymbol は複数列記述をすることにより、切り替えをリンクすることができます。

```
(例)
    (defsymbol ?[ "「" "『 "
                ?] "」" "』 " )
```

このように定義することにより "［" が "『" に切り替わると、自動的に "］ " も " 』 " に切り替わります。

(7) **メニューの定義 – defmenu**

```
(defmenu メニュー名
    ("表示文字列 1" 機能シンボル 1)
    ("表示文字列 2" 機能シンボル 2)
         :
)
```

新しいメニューを定義します。

メニュー名に指定した名前は、そのメニューを呼び出すときの機能名として使えるようになるとともに、 set-mode-display でモード表示文字列を変更するときのモード名としても使えるようになります。 set-mode-display については、 **(2) モード表示の変更** を参照してください。

表示文字列には、そのメニューの一覧表示で表示される候補名を指定します。

機能シンボルには、表 2.16に示す機能名、または defselection で定義した機能シンボルを指定します。 defselection については、 **(8) 文字一覧の作成** を参照してください。また、機能シンボルに、別の defmenu で定義されたメニュー名を指定することによりメニューツリーを構成することもできます。

defmenu で定義したメニュー名を機能名として使用し、 set-key などでキーをバインドすることにより、自分で作成したメニューを使用することができるようになります。 set-key については、 **(3) キーのバインド** を参照してください。

表 2.16: **メニューに指定できる機能名一覧**

機 能 名	機　　能
kigou-mode	記号全般の一覧モードになる
russian-mode	ロシア文字一覧モードになる
greek-mode	ギリシャ文字一覧モードになる
line-mode	罫線一覧モードになる
hex-mode	16 進コード入力モードになる
bushu-mode	部首入力モードになる
touroku-mode	単語登録モードになる
delete-dic-mode	単語削除モードになる
jisho-ichiran	辞書マウント／アンマウントを行う
chikuji-mode	逐次自動変換に切り替える
renbun-mode	連文節変換に切り替える
disconnect-server	サーバとの接続を切る
switch-server	サーバの切り替えを行う
show-server-name	サーバ名を表示する
show-gakushu	学習状態を表示する
show-canna-version	バージョンを表示する
show-romkana-table	ローマ字かな変換テーブル名を表示する
show-canna-file	カスタマイズファイル名を表示する
sync-dictionary	辞書に書き込む

```
(例)
    ; メニューツリーを構成する
    ; topmenu の定義
    (defmenu  topmenu
      ("記号一覧 " kigou-mode)        ; 機能 kigou-mode を実行する
      ("登録 " menu1)                 ; menu1 へ
    )

    ; menu1 の定義
    (defmenu menu1
      ("登録 " touroku-mode)          ; 機能 touroku-mode を実行する
      ("削除 " delete-dic-mode)       ; 機能 delete-dic-mode を実行する
      ("辞書一覧 " jisho-ichiran)     ; 機能 jisho-ichiran を実行する
    )

    (set-key 'empty-mode "\Help" 'topmenu)
```

メニューツリーを構成する際、カスタマイズの順番は任意です。すなわち、上記の例のように (defmenu menu1 …) を設定するよりも前に (defmenu topmenu …) の中で menu1 を指定することが可能です。ただし、再帰的にメニューを設定することはできません。

カスタマイズファイルのメニューの設定にエラーが存在した場合、そのメニューを選択したとき、ガイドラインに ”この項目は正しく定義されていません ” と表示して、 empty モードに戻ります。

(8) **文字一覧の作成 – defselection**

```
(defselection 機能シンボル "表示文字列 " '(文字一覧))
```

文字一覧を作成します。

機能シンボルには、その機能を参照するときのためのシンボルを指定します。以下その機能を参照するときには、この機能シンボルで参照されます。

表示文字列には、その機能に移行したときにガイドラインに表示する文字列を指定します。表示文字列は省略することもできます。省略する場合は、表示文字列に nil を指定します。省略時は、文字列一覧を表示する直前の表示文字列が表示されます。

文字一覧には、表示したい一覧を記述します。文字一覧の指定方法は次の 2 つの方法があります。

(a) 各要素を文字列として指定する (ダブルクォーテーションで囲む)。
(b) 各要素を文字として指定する (要素の前にクエスチョンマークをつける)。

(8a) の場合は、各要素をすべて指定します。

```
(例 1) 1 文字ずつ表示する場合
    ; 文字一覧 kakko の定義
    (defselection kakko "[括弧]"
      '(" (" ") " " 〔" "〕 " " [" "] " " {" "} "))
(例 2) 2 文字ずつ表示する場合
    ; 文字一覧 kakko の定義
    (defselection kakko "[括弧]" '(" () " " 〔〕 " " [] " " {} "))
```

(8b) の場合は、文字一覧の要素をすべて記述するのではなく、範囲指定することもできます。ただし、範囲は EUC コードのコード番号の昇順に指定してください。

```
(例 3) 要素をすべて指定する場合
    ; 文字一覧 keisan の定義
    (defselection keisan "[計算]"
      '(? ＋ ? － ? ± ? × ? ÷ ? ＝ ? ≠ ? ＜ ? ＞ ? ≦ ? ≧ ? ∞ ? ∴))
(例 4) 要素を範囲指定する場合
    ; 文字一覧 keisan の定義
    (defselection keisan "[計算]" '(? ＋ - ? ∴))
```

さらに、(例 1) 〜 (例 4) までの指定を合わせて指定することもできます。

```
(例 5) いろいろな方法を合わせて指定する
    ; 部首 "ぎょうにんべん" の定義
    (defselection gyou "[ぎょうにんべん]"
      ("徽" ? イ - ? 徼 ? 衒 - ? 衢 ? 衙 ? 黴))
```

(9) **初期状態の設定 − initialize-function**

```
(initialize-function '(初期状態設定シーケンス))
```

initialize-function はアプリケーションが起動したときのかな漢字変換の状態を指定します。初期状態設定シーケンスの部分には機能名を記述します。複数の機能名を指定することも可能です。

たとえば、アプリケーションプログラムの起動時に日本語入力モードであってほしい場合は、

```
(initialize-function '(japanese-mode))
```

のように記述します。

起動時にはアルファベットモードであってほしいが、XFER で日本語入力モードに移行したときにカタカナベースのモードであってほしい場合は次のように記述します。

```
(initialize-function '(japanese-mode base-katakana
                                        alpha-mode))
```

(10) ほかのカスタマイズファイルの読み込み – load

```
(load "ファイル名")
```

.canna からほかのカスタマイズファイルを読み込むことができます。ファイル名は絶対パス名で書くことが望ましいです。ホームディレクトリを表すチルダマークは使用できません。

(11) バージョンを表す変数

以下の変数には『かんな』のバージョン、かな漢字変換サーバのバージョン、かな漢字変換サーバとの通信に用いられているプロトコルのバージョンがそれぞれ格納されています。

canna-version	数字	『かんな』のバージョン
protocol-version	数字	プロトコルのバージョン
server-version	数字	サーバのバージョン

たとえば『かんな』 Version3.3 では canna-version には 3003 が入っています。次のように利用することができます。

(例)

```
(if (> protocol-version 1999) (setq auto t))
```

かな漢字変換プロトコルが 2.0 以上のときに限って逐次自動変換を利用します。

(12) そのほかのキーワード

以下で、そのほかのキーワードについて説明します。

- abandon-illegal-phonogram

  ローマ字かな変換で不正なローマ字が入力として残るかどうかを指定します。デフォルトでは不正なローマ字は残りますが、 abandon-illegal-phonogram に t を 指定すると不正なローマ字は入力から捨てられます。デフォルトは nil です。

（例）

```
(setq abandon-illegal-phonogram nil)
```

- allow-next-input

  候補一覧表示状態で数字以外のキーを押したときに、現在カーソルが位置してい る候補が選択され次の入力となるかどうかを指定します。 allow-next-input が t の場合、候補一覧表示状態で数字以外のキーを打つと、次の 入力となります。 allow-next-input が nil の場合、「ピッ」という音が鳴り、次の入力には用いられません。デフォルトは t です。

  （例）

  ```
  (setq allow-next-input t)
  ```

- auto

  逐次自動変換を用いるかどうかを指定します。 t で逐次自動変換を用います。デフォルトは nil です。

  （例）

  ```
  (setq auto nil)
  ```

- auto-sync

  単語登録削除した直後に自動的に辞書の書き出し処理を行うかどうかを指定します。 auto-sync が t の場合は、自動的に辞書の書き出し処理を行います。 auto-sync が nil の場合は、辞書に対するアクセスがなくなった時点で辞書の書き出し処理が行われます。 デフォルトは t です。

  （例）

  ```
  ; 単語登録削除の直後に辞書の書き出し処理を行います。
  (setq auto-sync t)
  ```

- break-into-roman

  バックスペースキーを打ったときに直前のローマ字かな変換された文字列がローマ字に戻るかどうかを指定します。デフォルトは nil です。

  （例）

  ```
  (setq break-into-roman t)
  ```

- bunsetsu-kugiri

  候補を表示しているときに文節ごとに空白で区切るかどうかを指定します。 t で区切ります。デフォルトは nil です。

  （例）

  ```
  ; 候補表示状態で文節句切りを行います。
  (setq bunsetsu-kugiri t)
  ```

- character-based-move

  読みを入力しているときにカーソル移動を行う場合に、文字単位で移動を行うかどうかを指定します。デフォルトは t です。 character-based-move を nil にする と、ローマ字かな変換の確定の単位をもとにしてカーソルを移動します。すなわち、 ”ju” と入力した場合は、『じゅ』は一文字とみなされてカーソルが移動します。文字削除のときも同様に取り扱われます。

(例)

```
; 読みを入力しているとき、カーソルは文字単位で移動します。
(setq character-based-move t)
```

- cursor-wrap

  読みを入力している状態や候補を表示している状態でカーソルを移動するときに右端から右へ移動する操作をしたときや左端から左や移動する操作をしたときに反対側の端にカーソルが移動することを指定します。t で移動し、nil で移動しません。デフォルトは t です。

  (例)

```
; カーソルラップしないようにします。
(setq cursor-wrap nil)
```

- define-esc-sequence

  端末のタイプごとにエスケープシーケンスと機能キーの対応を設定します。vi(1) や cannad(1) などがファンクションキーなどを解釈する際に参照されます。

  なお、アプリケーションによってはこの機能に対応していないことがあります。

  (例)

```
(define-esc-sequence "kterm" "\ESC[28~" ?\Help)
(define-esc-sequence "kterm" "\ESC[210z" ?\Xfer)
```

  ターミナル名とエスケープシーケンスは文字列を許しますが、最後のキーは単キーとするため、文字を指定するように構文上の制約を加えています。

- gakushu

  かな漢字変換が学習を行うかどうかを指定します。デフォルトは t です。

  (例)

```
(setq gakushu t)
```

- grammatical-question

  単語登録で品詞を指定した後、詳細な品詞分類のための質問を行うか否かを指定します。t で質問を行い、nil では質問をしません。デフォルトは t です。

  (例)

```
(setq grammatical-question nil)
```

- henkan-or-do-nothing

  逐次自動変換を用いている場合、ベース切替え機能を利用してベースを切替えた際のキーの動作を指定します。

  たとえば、句点の入力時に変換を行うように指定してあるが、ベースがひらがな以外の場合は変換を行わないようにするためには以下のように記述します。

  (例)

```
; 句点入力時に変換を行うが、ベースが
; ひらがな以外の場合はなにも行わない。
(set-key 'chikuji-yomi-mode "."
  '(self-insert henkan-or-do-nothing))
(set-key 'chikuji-bunsetsu-mode "."
  '(self-insert henkan-or-do-nothing))
```

- henkan-or-self-insert

  連文節変換を用いている場合、ベース切替え機能を利用してベースを切替えた際のキーの動作を指定します。

  たとえば、ベース切替え機能を利用して、 bese-eisu 機能により英数入力モードに移行する際、入力ベースがひらがなの場合は変換を行い、ひらがな以外の場合は self-insert を行うためには以下のように記述します。

  （例）

```
;読み入力中に C-o を押すことにより
;日本語入力モードの ON/OFF を行う。
(set-key 'yomi-mode "\C-o" 'base-kana-eisu-toggle)
;ベースがひらがな以外の場合は self-insert を行う。
(set-key 'yomi-mode "\Space" 'henkan-or-self-insert)
```

- hex-direct

  16 進コード入力時に 4 ケタ目を入れた時点で 16 進コード入力モードが終了するかどうかを指定します。 nil を指定すると 4 ケタ目を入れた時点でも入力が確定しない状態となります。デフォルトは nil です。

  （例）

```
(setq hex-direct nil)
```

- ignore-case

  ローマ字かな変換テーブルには普通小文字の規則だけが定義されており、通常は大文字を入力するとアルファベットのまま入力され、かなには変換されません。 ignore-case を t にした場合、大文字で入力してもローマ字かな変換では小文字として取り扱われます。デフォルトは nil です。

  （例）

```
(setq ignore-case nil)
```

- index-hankaku

  候補一覧の番号を全角文字から半角文字に変更するかどうかを指定します。

  デフォルトは nil(全角文字) です。

- index-separator

  候補一覧の番号を半角文字にカスタマイズしている場合の、番号と候補との区切りのセパレータ文字を指定します。? の次の 1 文字をセパレータ文字として使用します。デフォルトはピリオドです。

  （例）

```
;セパレータをコロンにします。
(setq index-separator ?:)
```

- kakutei-if-end-of-bunsetsu

  最右文節で次文節へ移動しようとしたときに、確定するか否かを指定します。 t で確定します。 nil を指定すると最左文節がカレント文節になります。デフォルトは nil です。また、stay-after-validate が nil のときで kakutei-if-end-bunsetsu が t のときは最右文節で候補一覧状態から候補を選択すると全文が確定します。

（例）

```
(setq kakutei-if-end-of-bunsetsu nil)
```

- kouho-count

  候補一覧など一覧表示をしているときに、いま選択している項目が全体の項目の何番目であるのかを表示するかどうかを指定します。t で表示し、nil で表示しません。デフォルトは t です。

  （例）

  ```
  (setq kouho-count t)
  ```

- n-henkan-for-ichiran

  変換キー (デフォルトではスペースキーと Xfer) を何回か押すと候補一覧表示が行われるようにすることができます。n-henkan-for-ichiran に対して数を指定すると、 指定された回数変換キーを押すことにより候補一覧が表示されます。0 を指定すると何回変換キーを押しても一覧表示が行われなくなります。デフォルトは 2 で、2 回変換キーを押すと一覧表示になります。

  （例）

  ```
  ; 変換キーを 3 回押すと候補一覧が表示されます。
  (setq n-henkan-for-ichiran 3)
  ```

- n-keys-to-disconnect

  アルファベットを入力し続けた場合のサーバとの接続の切れるストローク数を指定します。デフォルトは 500 です。0 を指定した場合は、サーバとの接続は切れなくなります。

  （例）

  ```
  (setq n-keys-to-dicsonnect 500)
  ```

- n-kouho-bunsetsu

  逐次自動変換時に漢字に自動的に変換された文節をいくつまで確定せずに保持するかを 指定します。3 〜 32 の範囲で指定してください。デフォルトは 16 です。

  （例）

  ```
  (setq n-kouho-bunsetsu 16)
  ```

- numerical-key-select

  候補一覧を表示しているときに、数字キーを用いて候補を選択できるかどうかを指定します。デフォルトは t です。nil を指定すると、候補一覧の番号が表示されません。また、数字キーを押した場合、選択されている候補が確定し、押したキーは次の読みの入力として取り扱われます。

  （例）

  ```
  ; 一覧選択を数字キーで行いません。
  (setq numerical-key-select nil)
  ```

- reverse-widely

  t を指定すると読みを入力しているときの文字列の反転範囲が広くなります。デフォルトは nil です。

(例)

```
; 読み入力時、読み全体を反転させます。
(setq reverse-widely t)
```

- romaji-yuusen

  romaji-yuusen が t の場合、現在の入力モードが読み入力モードまたはそれに準ずるモードであり、ローマ字かな変換が途中の状態のとき、次の入力文字がローマ字かな変換で有効なキーであれば、 self-insert が行われます。 romaji-yuusen が nil の場合、どのような状態においても、キーに割り当てられた機能が実行されます。デフォルトは nil です。

(例)

```
; ローマ字かな変換を優先します。
;  たとえば "x" + " "(スペース) に対して、ローマ字かな変換テーブル
;  で "  "(全角のスペース) を割り当てている場合、 romaji-yuusen が
;  t の時は全角のスペースが表示されますが、 romaji-yuusen が nil
;  の時は、スペースが『変換』の意味として用いられます。
(setq romaji-yuusen t)
```

- quickly-escape-from-kigo-input

  記号入力モード時に、記号を連続して入力するかどうかを指定します。 quickly-escape-from-kigo-input が t の場合、記号入力モードで記号を 1 つ入力すると記号入力モードを終了します。 quickly-escape-from-kigo-input が nil の場合、記号を選択したのちも記号入力モードにとどまり、記号を連続して入力することができます。デフォルトは nil です。

(例)

```
; 記号を連続して入力します。
(setq quickly-escape-from-kigo-input nil)
```

- quit-if-end-of-ichiran

  候補一覧表示をしているときに、最終候補を表示している状態で次候補操作を行うと候補一覧表示を終了し、読みそのものを候補として表示するようになります。 2 打目のスペースキーを候補一覧表示にカスタマイズしているときなどは quit-if-end-of-ichiran を t にしておくと便利です。デフォルトは nil です。

(例)

```
(setq quit-if-end-of-ichiran t)
```

- select-direct

  numerical-key-select が t であるときに、数字キーで候補を選択したときに候補一覧表示のままかそうでないかを指定します。 t の場合は候補一覧が終了します。デフォルトは t です。

(例)

```
; 数字キー入力で一覧モードを終了します。
(setq select-direct t)
```

- stay-after-validate

  候補一覧表示状態で候補を選択して単一候補表示状態に戻ったときに、カレント文節を次の文節に移動させるかどうかを指定します。 nil でカレント文節は次の文節に移動します。 t ではカレント文節は変わりません。デフォルトは t です。

(例)

```
; 候補選択すると次の文節に移動します。
(setq stay-after-validate nil)
```

2.6.5 そのほかの Lisp の関数

今まで説明した以外の Lisp の関数を説明します。以下の関数はかな漢字変換には直接は影響を及ぼしません。

(1) **条件分岐 − cond, if**

```
(if A B C)
```

この式は、まず A を評価し、その結果が nil 以外であれば、 B を評価し、 nil であれば C を評価します。 (if ……) 式の結果は B または C の評価結果となります。

(例)

```
(if (> protocol-version 2000) (setq auto t)
                              (setq auto nil))
(setq auto (if (> protocol-version 2000) t nil))
```

```
(cond (C1 B1) (C2 B2)・・・・(Cn Bn))
```

まず、 C1 を評価し、その結果が nil 以外なら B1 を評価しその結果を cond 式の結果とします。その場合 C2 ～ Cn、 B2 ～ Bn は評価されません。 C1 の評価結果が nil なら次に C2 を評価します。 C2 の結果が nil 以外なら B2 を評価しその結果を cond 式の結果とします。

Ci を次々と nil 以外の値が出てくるまで評価し、 nil 以外の値が返って来たところの Bi を評価し、それを cond 式の値として返します。

(2) **そのほかの制御構造 − let**

```
(let ((VAR1 VAL1) (VAR2 VAL2) ....  (VARn VALn)) body)
```

変数 VAR1 ～ VARn に、それぞれ値 VAL1 ～ VALn を代入した後に body を評価します。 body の評価結果が let 式の結果となります。 body は複数の式を記述することが可能です。 let 式の評価が終了すると VAR1 ～ VARn の値は元の値に戻ります。

(3) **関数の逐次呼出し − progn**

```
(progn A B C ....)
```

式 A B C … を次々と評価して最後の式の評価結果を値とします。 if 文の実行部に複数の処理を記述するときに使います。

```
(例)
    (if (< canna-version 2000)
      (progn (setq n-henkan-for-ichiran 2)
             (setq quit-if-end-of-ichiran t)) )
```

(4) **値の比較 − equal, =, >, <, eq**

値の比較を行います。 eq はシンボル同士の比較と数値同士の比較のみにしか用いることができませんが、非常に高速に比較処理を行います。

(5) **真理値の調査 − not, and, or**

真理値を調査します。

and および or は引数を左側から必要な数だけ評価します。すなわち、 and の場合は最初に nil を結果として返す式があった場合、そこから右に書かれている引数は評価しません。 or の場合は、最初に nil 以外の値を結果として返す式があった場合、そこから右に書かれている引数は評価しません。

(6) **算術演算 − +, −, ∗, /, %**

算術演算を行います。意味は C 言語の場合と同じです。

(7) **ガーベジコレクション − gc**

Lisp 言語のメモリ管理システムを起動し、メモリの整理を行います。

(8) **リスト演算 − list, cons, car, cdr, atom, null**

リスト処理を行います。かな漢字変換で用いることはないでしょう。詳細は Lisp 関連の参考書をご覧ください。

(9) **引用 − quote**

```
(quote A)
```

'A と同じように A というデータをそのままデータとして使いたいときに用います。

(10) **代入 − set**

set は setq と同様の処理を行います。 setq との違いは第 1 引数を評価する点です。

(11) **関数定数 − defun, defmacro**

関数定義、マクロ定義を行います。かな漢字変換で用いることはないでしょう。詳細は Lisp 関連の参考書をご覧ください。

(12) **シンボルの複製 − copy-symbol**

シンボルの複製を作成します。 copy-symbol はシンボルの持つさまざまな属性をそのままコピーします。

```
(例)
    (copy-symbol '読みモード 'yomi-mode)
```

上記の処理により yomi-mode というキーワードの代わりに「読みモード」と記述することができるようになります。

2.7 こんな風にカスタマイズしてみたい

以下では、次のようにカスタマイズするときのカスタマイズファイルの記述例を示します。

1	変換は Xfer で行い、スペースは全角空白文字の入力として使用したい。
2	候補を選択したら次の文節に移動したい。
3	一太郎[9]のような操作法で入力したい。
4	「と［を両方使いたい。
5	拡張メニューを自分で定義したい。

以下のカスタマイズの説明では、.canna を書き換えることによってカスタマイズを実現します。書き換えは、.canna の最後の方に加えていけば良いでしょう。.canna がない場合は、次のコマンドを入力して、.canna を準備してください。

```
% cp $(CANNALIBDIR)/sample/default.canna .canna
% chmod u+w .canna
```

(1) **変換は Xfer で行い、スペースは全角空白文字の入力として使用したい**

まず、スペースキーに対して通常の入力機能 (self-insert) を割り当てます。 self-insert の self とはキー自身を指します。すなわち、スペースキーに対して self-insert を割り当てると空白を挿入するという機能がスペースキーに割り当てられます。

スペースキーに self-insert 機能を割り当てるには次のようにカスタマイズファイルします。

```
(global-set-key "\Space" 'self-insert)
```

"\Space" はスペースキーを示します。 "\Space" の代わりに " " のように記述しても同じ意味になります。

さて、これだけ設定するとスペースキーが変換操作に使用できなくなり、代わりに全角空白が挿入されるようになります。

全角空白ではなく、半角空白にしたいときはローマ字かな変換テーブルの書き換えが必要となります。

(2) **候補を選択したら次の文節に移動したい**

長い文章を入力して変換キーを押し、前の文章から候補選択をするという方法で入力している方も多いでしょう。そのような方の場合、候補一覧で選択したときにカレント文節が同じ文節にとどまってしまうといちいち文節移動をしなければならないので不便を感じてはいないでしょうか？

[9] 一太郎はジャストシステム (株) の商標です

```
% 校舎を改築しようと考えていますがどうしましょうね
[ あ ] 1 公社を 2 校舎を 3 降車を 4 巧者を                    2/8
```

↓確定 (RETURN) を入力

```
% 校舎を改築しようと考えていますがどうしましょうね
[ あ ]
```

「一覧を選択したのだからその文節での仕事は終わっているはずだ。一覧選択を終わったら次の文節に移動してほしい。」という風にお思いの方も多いでしょう。そのような場合は、次のようにカスタマイズします。

```
(setq stay-after-validate nil)
```

```
% 校舎を改築しようと考えていますがどうしましょうね
[ あ ] 1 公社を 2 校舎を 3 降車を 4 巧者を                    2/8
```

↓確定 (RETURN) を入力

```
% 校舎を改築しようと考えていますがどうしましょうね
[ あ ]
```

(3) 一太郎[10]のような操作法で入力したい

「一太郎に慣れているので一太郎とできるだけ同じ操作で日本語入力がしたい」という方もいらっしゃるでしょう。

一太郎とほぼ同じキー操作を提供するカスタマイズファイルが $(CANNALIBDIR)/sample/just.canna として提供されていますので、以下のコマンドを実行して、そのファイルをカスタマイズファイルとして使用するようにしてください。

[10] 一太郎はジャストシステム (株) の商標です

```
% cd
% cp $(CANNALIBDIR)/sample/just.canna .canna
```

(4) **「と［を両方使いたい**

ワープロによっては、「. 」(ピリオド) を打ったときに「。」が入力されるか「. 」が入力されるかを選択できたり、「[」(角かっこ) を入力したときに「「 」(かぎかっこ) が入力されるか「［ 」(全角角かっこ) が入力されるかを選択できたりするものがあります。

『かんな』でもローマ字かな変換テーブルを書き換えることにより、「. 」や「［ 」に対応する記号を変更することができますが、ローマ字かな変換テーブルの書き換えは若干面倒でもあります。

このような場合、以下のようにカスタマイズすることによって、それぞれの記号を両方定義しておき、必要に応じて簡単に切り替えることができます。

```
(defsymbol ?. "。" "." )
(defsymbol ?[ "「" "[" )
```

このようにしておくと以下の手順で記号を切り替えることができます。

まず、「. 」を入力します。

```
。
[ あ ]
```

↓変換キーを押し、候補一覧を表示させます。

```
。
[一覧]  1 。 2 .                                    1/2
```

↓ →, RETURN で 2 を選びます。

```
.
[ あ ]
```

つぎからはピリオドを打つと「. 」が入ります。

同様の操作により元に戻すことが可能です。

(5) **拡張メニューを自分で定義したい**

デフォルトでは、HELPキーを押したときに、ガイドラインにメニューが表示され、記号入力、単語登録、変換方式の変更などが行えます。しかし、この方法では、自分が使いたい機能をすぐに選択することができなかったり、記号入力時に自分が選択したい記号をすぐに見つけることができなかったりすることがあります。

このような場合、メニューを自分でカスタマイズすることができます。

ここでは、自分で文字一覧を作成し、さらに、自分の使いやすいようにメニューを定義してみましょう。

たとえば、

(a) 変換方式の変更などはほとんど行わないので、メニューには一覧表示と単語登録だけがあればいい。

(b) 一覧表示では、自分で作成した括弧と数学記号の一覧も表示したい。

という場合には、以下のようにカスタマイズします。

```
; 文字一覧 kakko の定義
(defselection kakko "[括弧]" '(? （ ?） ? 〔 ?〕 ? ［ ?］ ? ｛ ?｝ ? 〈 ?〉 ))
; 文字一覧 keisan の定義
(defselection keisan "[計算]"
  '(? ＋ ? － ? ± ? × ? ÷ ? ＝ ? ≠ ? ＜ ? ＞ ? ≦ ? ≧ ? ∞ ? ∴))

; メニューに一覧と登録を定義する
(defmenu  toplevel-menu       ; 初期 menu の定義
  ("一覧 " menu1)             ; menu1 へ
  ("登録 " menu2)             ; menu2 へ
)

; 一覧を選択すると、括弧、数学記号、ギリシャ語の一覧表示ができる
(defmenu menu1                ; menu1 の定義
  ("カッコ " kakko)           ; kakko を一覧表示する
  ("計算 " keisan)            ; keisan を一覧表示する
  ("ギリシャ " greek-mode)    ; 機能 greek-mode を行う
)

; 登録を選択すると、単語登録 / 削除、辞書一覧の表示ができる
(defmenu menu2                ; menu2 の定義
  ("登録 " touroku-mode)      ; 機能 touroku-mode を行う
  ("削除 " delete-dic-mode)   ; 機能 delete-dic-mode を行う
  ("辞書一覧 " jisho-ichiran) ; 機能 jisho-ichiran を行う
)

; C-t を押すと、定義したメニューを表示する
(set-key 'empty-mode "\C-t" 'toplevel-menu)
```

このようにカスタマイズすると、以下のようにメニューを利用できます。

(a) CTRL + t を押して、メニューに入る。

```
% ▉
[拡張] 1 一覧  2 登録                                    1/2
```

(b) **1 一覧** を選択する。

```
% ▉
[拡張] 1 カッコ  2 計算  3 ギリシャ                      1/3
```

(c) **1 カッコ** を選択する。

```
% ■
[括弧] (  )  〔 〕  [ ]  { }  〈 〉                    1/10
```

(d) (b) で、**2 計算** を選択する。

```
% ■
[計算] +  −  ±  ×  ÷  =  ≠  <  >  ≦  ≧  ∞  ∴      1/13
```

(e) (a) で、**2 登録** を選択する。

```
% ■
[拡張] 1 登録  2 削除  3 辞書一覧                     1/3
```

第 3 章

かな漢字変換サーバ

3.1 かな漢字変換サーバ

3.1.1 サーバクライアントモデル

本日本語入力システムはサーバクライアントモデルをベースにしています。

つまり、かな漢字変換が行われる時は漢字に変換したい読みをプロセス間通信でかな漢字変換サーバに送り、かな漢字変換サーバでかな漢字変換辞書を元に漢字に変換した後に変換した結果をアプリケーションに返すことによりかな漢字変換を成し遂げています。

すなわち、日本語入力デーモンや日本語入力を行うアプリケーションがかな漢字変換を行う時には、それらのプログラムが直接かな漢字変換辞書をアクセスしてかな漢字変換を行うのではなく、かな漢字変換サーバを介して漢字に変換しています。

このように、本システムではサーバクライアントモデルをベースとしたかな漢字変換が採用されています。

3.1.2 かな漢字変換サーバの指定

日本語入力を利用するプログラムとかな漢字変換サーバとの間はソケットを用いたプロセス間通信が行われます。このため、日本語入力を利用するプログラムと、かな漢字変換サーバは同一のマシン上に存在する必要はなく、かな漢字変換サーバとして利用するサーバを、ネットワーク上の任意のマシンから自由に選択して使用することができます。

たとえば、ディスク容量がたくさんあり、 CPU もある程度の性能を持つマシンに辞書を置き、かな漢字変換サーバを走らせておけば、ネットワーク上のどのマシン上からでも高性能のかな漢字変換エンジンを利用することが可能となります。ネットワーク上のどのマシンを利用している時も、常に同一のかな漢字変換サーバを利用することにより、新たに登録した単語や頻度情報を一元管理することができます。

- CANNAHOST

  どのマシンで動作しているかな漢字変換サーバを用いるかを指定するには環境変数 CANNAHOST で、利用したいかな漢字変換サーバが動作しているマシンのホスト名を指定します。たとえば、ホスト名が machine1 というマシンで動作しているかな漢字変換サーバを利用する場合は.login ファイルに次のように記述してください。

  (例)

```
% setenv CANNAHOST machine1
```

かな漢字変換サーバは、デフォルトでは同一マシン上で動作しているものが使われます。環境変数 CANNAHOST を設定することにより他のマシン上で動作しているかな漢字変換サーバを用いるようになります。

- $(CANNALIBDIR)/cannahost

  環境変数 CANNAHOST が設定されていないときには通常は同一マシン上のかな漢字変換サーバが使われますが、 /usr/lib/canna/cannahost というファイルに他のマシンのホスト名が記述されている場合は /usr/lib/canna/cannahost にて示されるマシン上のかな漢字変換サーバが用いられるようになります。すなわち、 /usr/lib/canna/cannahost に他のマシンのホスト名を記述しておくと、そのマシンを利用しているユーザは環境変数 CANNAHOST を用いて明示的にかな漢字変換サーバを指定している場合以外は、 /usr/lib/canna/cannahost にて指定されるサーバを用いるようにすることができます。

  たとえばディスクレスマシンのように、システム資源が足りない場合などは、 /usr/lib/canna/cannahost ファイルにディスクレスサーバマシンなどのホスト名を記述しておくことにより、他のマシンのかな漢字変換サーバを利用するように記述しておいたほうが良いでしょう。

  (例)

  ```
  % cat /usr/lib/canna/cannahost  machine1
  ```

  /usr/lib/canna/cannahost にて他のマシンを指定することにより、同一マシン上のかな漢字変換サーバは使われないようになります。その場合は、かな漢字変換サーバの起動は不要となりますので、 /etc/rc0.d/Kaacanna、 /etc/rc1.d/Kaacanna、 /etc/rc2.d/Saacanna を削除して cannaserver が起動しないようにしておくと良いでしょう。また、この場合、さらにかな漢字変換辞書等が不要となります。以下のファイルまたはディレクトリをそのマシンのハードディスクから削除することが可能となります。

  ```
  $(CANNALIBDIR)/dic
  $(CANNALIBDIR)/cannaserver
  $(CANNALIBDIR)/cannakill
  $(CANNABINDIR)/crxdic
  $(CANNABINDIR)/dicar
  $(CANNABINDIR)/dpbindic
  $(CANNABINDIR)/dpxdic
  $(CANNABINDIR)/forsort
  $(CANNABINDIR)/mkbindic
  ```

3.1.3 cannaserver の起動

日本語機能のかな漢字変換は、かな漢字変換サーバ cannaserver が行っています。 cannaserver はシステム起動時に /etc/rc0.d/Kaacanna、 /etc/rc1.d/Kaacanna により起動されます。

かな漢字変換の際、辞書は cannaserver を介してアクセスします。

/usr/lib/canna/cannahost でそのマシン上のユーザが使用するかな漢字変換サーバとして、他のホスト上のサーバを指定している場合は、 cannaserver の起動は不要となりますので、 /etc/rc0.d/Kaacanna、 /etc/rc1.d/Kaacanna、 /etc/rc2.d/Saacanna を削除して cannaserver が起動しないようにしておくと良いでしょう。

3.1.4 cannaserver のアクセス制御

cannaserver にアクセスできるホストとユーザを指定することができます。

- /etc/hosts.canna

  /etc/hosts.canna というファイルを作成することで、同一のホスト上の cannaserver が接続要求を受け付けるホストの制限を付けることができます。 cannaserver は同一のホスト、または /etc/hosts.canna に登録されたホストと登録されたユーザに対してのみ接続要求を受け付けます。 /etc/hosts.canna が存在しないか、ファイルの長さが 0 の場合はすべてのホストが cannaserver にアクセスすることができます。 (/etc/hosts.canna に不正な記述をすると、サーバにアクセスできなくなるおそれがあるので注意してください。) ホスト名のあとにコロン ":"を書き、その後にユーザ名を指定します。多数のユーザを指定する場合は、ユーザ名とユーザ名の間にカンマ ","を書きます。ホスト名のみが指定された場合は、そのホストのユーザ全員のアクセスを受け付けます。

  cannaserver が存在するホストの名前は,"unix"と記述してください。 /etc/hosts.canna ファイルが存在する場合、そのファイルの中に "unix"という記述がないと、そのホスト自身の cannaserver にアクセスすることができなくなります。

  (例)

  ```
  pdg3:hanafusa,root
  pbg1:
  pbg4
  pbg5
  unix
  ```

- cshost

  cshost(1M) でアクセス可能なホスト名とユーザ名のリスト (アクセスコントロールリスト) を参照することができます。

3.2 辞　　書

3.2.1 辞書と辞書ファイル

かな漢字変換に用いる「辞書」は、かなを漢字に変換するためのデータを格納しているものです。「辞書」が納めてあるファイルを「辞書ファイル」と呼びます。「辞書ファイル」には 1 個だけの「辞書」が納められている場合もあれば、複数の「辞書」が納められている場合もあります。「辞書ファイル」にはテキスト形式のものとバイナリ形式のものがあります。システムで最初から提供されている辞書をシステム辞書と呼びます。

3.2.2 システム辞書のディレクトリ

本システムではかな漢字変換辞書へのアクセスは単一のかな漢字変換サーバだけが行います。かな漢字変換辞書はかな漢字変換サーバが動作しているマシン上の $(CANNALIBDIR)/dicの下にあります。

- 辞書ホームディレクトリ

  $(CANNALIBDIR)/dic を辞書ホームディレクトリと呼びます。辞書ホームディレクトリの下にはさらに canna、 user、 group というディレクトリがあります。

- 辞書ディレクトリ

  $(CANNALIBDIR)/dic/canna、 $(CANNALIBDIR)/dic/user、 $(CANNALIBDIR)/dic/group、を辞書ディレクトリと呼びます。 canna 配下のファイルは、システム辞書ファイルと呼びます。 user 配下のディレクトリは、かな漢字変換の利用者ごとに使用する辞書を分けて管理するためのディレクトリです。ユーザ辞書ディレクトリ名はユーザ名と同じです (uid が異なる利用者でもユーザ名が同じ場合は同じ利用者とみなされますので注意してください)。また、 group 配下のディレクトリは、グループ名 (gid) が同じ利用者間で共通に使用する辞書を管理するためのディレクトリです。グループ辞書ディレクトリ名はグループ名と同じです。

- システム辞書ファイル

  $(CANNALIBDIR)/dic の下にあるディレクトリのうち、 canna という名前のディレクトリは、かな漢字変換を利用するすべてのユーザに共通に使用されます (このため canna というユーザ名をもつユーザによる辞書の作成等は禁止されています)。このディレクトリの下にある辞書ファイルのことをシステム辞書ファイルと呼びます。システム辞書ファイルには以下のものがあります。

  ```
  iroha.d
  fuzokugo.d
  hojomwd.t
  hojoswd.t
  ```

- ユーザ辞書ファイル

  かな漢字変換利用者はシステム辞書ファイル以外に各自で使用するための辞書ファイルを作成することができます。各自で用意した辞書ファイルは辞書ホームディレクトリの下にある user 配下にログイン名と同名のディレクトリを作成してその下に置いておきます。単語登録の際、かな漢字変換サーバが辞書ファイルに対して書き込みを行います。かな漢字変換サーバはオーナ / グループが bin/bin で動作していますので、辞書ファイルを作成する時はそのファイルのグループを bin にしてグループに対する書き込み権を開放しておいてください。

  例として辞書ホームディレクトリの下にある user 配下に、 user というディレクトリを準備しました。各自でディレクトリを作成する時の参考にしてください。 user の下には以下の辞書ファイルがあります。

  ```
  chimei.t
  katakana.t
  software.t
  user.t
  necgaiji.t
  ```

- グループ辞書ファイル

  かな漢字変換利用者のグループ名 (gid) が同じ利用者間では、共通の辞書ファイルを利用することができます。この辞書ファイルに単語を登録することによって、グループ名が同じ利用者は、その単語を利用することができますので、各自で登録する手間がはぶけ、ディスクスペースを削減することができます。単語登録の際は、ユーザ辞書ファイルと同様に、かな漢字変換サーバが辞書ファイルに対して書き込みを行いますので、辞書ファイルを作成する時には、そのファイルのグループを bin にしてグループに対する書き込み権を開放しておいてください。グループ辞書は、辞書ホームディレクトリの下にある group 配下に、グループ名と同じディレクトリが作成され、その下に置かれます。

  グループ辞書は、『かんな』 Version 3.2 より前のクライアントでは使用できません。

- 辞書サーチパス

  辞書を探し出す順序のことを辞書サーチパスといいます。

  (1) 『かんな』 Version 3.2 より前のクライアントの場合
  かな漢字変換に用いる辞書は、辞書ホームディレクトリの下のログイン名と同名のディレクトリの下の辞書をまず探して用いようとします。次にシステム辞書ファイルの中から指定された辞書を探して用います。

  (2) 『かんな』 Version 3.2 以降のクライアントの場合
  ユーザ辞書、グループ辞書、システム辞書という順で辞書を探して用いようとします。ユーザ辞書を探す際には、 user 配下にあるログイン名と同名のディレクトリの下の辞書を探して用いようとします。グループ辞書を探す際には、 group 配下にあるグループ名と同名のディレクトリの下の辞書を探して用いようとします。最後に、システム辞書ファイルの中から指定された辞書を探して用います。
  ユーザ辞書とグループ辞書に同じ名前のファイル名がある場合は、ユーザ辞書が優先されて用いられます。この場合は、単語を登録・削除する際にユーザ辞書に対して行いますので注意してください。

3.2.3 辞書ファイルの形式

かな漢字変換辞書ファイルには次の 2 つの形式があります。

- テキスト形式辞書

  通常のテキスト形式の辞書ファイルです。単語登録などはこの辞書ファイルに格納されている辞書に対して行われます。テキスト形式ですので通常のエディタでメンテナンスすることもできます。(ただし、サーバが使用中の辞書ファイルをエディトしても、その変更は無効となります)

  辞書ファイル名の拡張子は ”.t” でなければなりません。

- バイナリ形式辞書

  辞書の読み込みが高速になるような形式に変換された辞書ファイルです。辞書ファイル名の拡張子は ”.d” でなければなりません。辞書中にある不要な単語を削除することや、一度削除した単語を復活させることができます。辞書ユーティリティツールを使ってテキスト形式とバイナリ形式を相互変換することが可能です。また、複数のバイナリ形式の辞書ファイルを cat(1) または dicar(1) で 1 つのバイナリ形式辞書ファイルに結合することができます。 dicar(1) については **4 かな漢字変換ユーティリティ**を参照してください。

  辞書ユーティリティツールを使ってテキスト形式辞書ファイルをバイナリ形式に変換した場合、バイナリ形式辞書ファイルは、内部にもとのテキスト形式辞書ファイルのファイル名を保持しておきます。 cat(1) で単一のバイナリ形式辞書ファイルに結合した場合もこのテキスト形式辞書ファイル名は失われません。 cat(1) で辞書を結合した場合、そのバイナリ辞書ファイルに含まれている辞書を区別するのにテキスト形式辞書ファイル名が用いられます。

3.2.4 辞書の所有者・非所有者

辞書は、だれでも単語を登録したり、辞書の中身を見たりできる訳ではありません。辞書それぞれに制限があります。辞書に対しての制限は、所有者と非所有者によって変わります。辞書ディレクトリ `/usr/lib/canna/dic` 配下の辞書の所有者についての考え方を以下に示します。

	所有者	非所有者
user/* 配下	ログイン名がディレクトリと同じ名前の利用者	それ以外の利用者
group/* 配下	グループ名がディレクトリと同じ名前の利用者	それ以外の利用者
canna 配下	利用者全員	なし

ユーザ辞書は、/usr/lib/canna/dic/user 配下のディレクトリと同じログイン名の利用者に所有されます。またグループ辞書は、/usr/lib/canna/dic/group 配下のディレクトリと同じグループ名の利用者に所有されます。システム辞書は、利用者全員が所有者となっていますが、システム辞書への単語の登録・削除は行えません。

3.2.5 辞書の READ・WRITE 権

辞書の READ 権は、非所有者が辞書の中身を見たり (catdic(1))、辞書のコピー (cpdic(1)) をしたりできるかどうかの制限を与えます。

他人に辞書の中身を見られたくない場合には、READ 権をなくしておくことによって、他人に中身を見られないようにすることができます。

辞書の WRITE 権は、所有者が単語の登録・削除できるかどうかの制限を与えます。

この制限は主にグループ辞書を考慮したもので、WRITE 権をなくしておくことで、同じグループの利用者間での不用意な単語登録・削除を避けることができます。

	所有者	非所有者
READ 権	○	○ ／ ×
WRITE 権	○ ／ ×	×

3.2.6 辞書目録 (dics.dir)

辞書ホームディレクトリの下の各ディレクトリ配下にどのような辞書が存在するかは各ディレクトリの下の dics.dir という名前のファイルに記述されています。dics.dir の記述は以下のようになります。

```
#
# システム辞書
#
fuzokugo.d(fuzokugo.swd)        -fuzokugo---
fuzokugo.fq(fuzokugo.swd)       -fuzokugo---
iroha.d(iroha.mwd)      -iroha---
iroha.fq(iroha.mwd)     -iroha---
iroha.d(yuubin.mwd)     -yuubin---
yuubin.fq(yuubin.mwd)   -yuubin---
iroha.d(bushu.mwd)      -bushu---
bushu.fq(bushu.mwd)     -bushu---
hojomwd.t(.mwd)         -hojomwd---
hojoswd.t(.swd)         -hojoswd---
```

```
#
# ユーザ辞書
#
user1.t(.mwd) -user1--r-
user2.t(.mwd) -user2--rw-
user3.t(.mwd) -user3--w-
user4.t(.mwd) -user4---

#
# グループ辞書
#
user1.t(.mwd) -group1--r-
user2.t(.mwd) -group2--rw-
user3.t(.mwd) -group3--w-
user4.t(.mwd) -group4---
```

dics.dir では 1 行に 1 つの辞書についての記述を行います。まず、一番左に辞書ファイル名を記述します。次に () に挟まれて記述されるのが、その辞書ファイルの中に存在する辞書の子辞書名です。右側にハイフン (–) で囲って記述されているのがその辞書につけた辞書名です。この辞書名が辞書の指定に用いられます。辞書名はファイル名および子辞書名に依存せず、自由に命名することができます。 () 内に記述される子辞書名は、その辞書をテキスト形式からバイナリ形式に変換する際に命名されます。子辞書名の拡張名の拡張子は ”.mwd”または ”.swd”でなければなりません。また、テキスト辞書の場合、ベースネームを省略して ”.mwd”または ”.swd”を子辞書名とします。

辞書名を囲んでいるハイフン "-" の次にアクセス権についての記述を行います。非所有者に READ 権があるならば "r" を、所有者に WRITE 権があるならば "w" を、ハイフン "-"で囲んで記述します。 READ・WRITE 権がない場合には、ハイフンのみを続けて記述します。

dics.dir で # から改行まではコメントとして扱われます。

頻度別学習情報ファイルについては、 READ・ WRITE 権に関係なくそのファイルの所有者であれば、書き込みを行うことができます。

3.2.7　テキスト形式辞書の作り方

1 行に読み、品詞、単語を空白またはタブで区切って記述します。単語は空白またはタブで区切れば複数記述することができます。

辞書ファイル名は、バイナリ辞書と区別するために xxx.t という名前にしてください。

読み　品詞　単語　［単語］　…

(例)

```
あぷ        #T35   アプリケーション　アプリケーションプログラム
お          #KJ    お　御
かなかん    #T30   かな漢字変換
でふぉると  #T15   デフォルト
ふじいえ    #JN    藤家
   :
   :
   :
```

品詞を表わす記号については**付録 G** 品詞コード表を参照してください。

3.2.8 バイナリ形式辞書の作り方

テキスト形式辞書ファイルをバイナリ形式辞書ファイルに変換するには、 mkbindic(1) を使って行います。

xxx.t というテキスト辞書をバイナリ形式に変換すると、 xxx.mwd という名前が子辞書名になります。

(例)

```
% mkbindic foo.t
```

mkbindic(1) を実行しますと、読みをキーとしてテキスト形式辞書ファイルを辞書順にソートし、それをバイナリ化し、 xxx.d(例では、 foo.d) というバイナリ形式辞書を作成します。

複数の辞書ファイルをまとめて 1 つのバイナリ形式辞書にすることができます。

(例 1)

```
% cat foo.d >> bar.d
```

(例 2)

```
% dicar -r bar.d foo.d
```

3.2.9 バイナリ形式辞書ファイルとテキスト形式辞書ファイルの相互変換

辞書はテキスト形式でもバイナリ形式でも使用することができますが、単語登録用ユーザ辞書として使えるのはテキスト形式だけです。

テキスト形式とバイナリ形式は、相互に変換することができます。バイナリ形式への変換は mkbindic(1)、バイナリ形式からテキスト形式への変換は dpbindic(1) を使います。 mkbindic(1) と dpbindic(1) についての詳細は、 **4 かな漢字変換ユーティリティ**を参照してください。

3.2.10 ユーザ辞書の設定

単語登録を行うには、カスタマイズ機能によりユーザ辞書の設定を行う必要があります。カスタマイズ機能についての詳細は **2 カスタマイズ** を参照してください。

(1) .canna の作成

ホームディレクトリに .canna というファイルを作成し、以下のように書き込んでください。引数には辞書名を記述します。最初の 2 行は基本になる辞書の名前ですので必ず記述してください。

はじめは $(CANNALIBDIR)/sample の下の default.canna というファイルをコピーして用いると良いでしょう。

例)

```
(use-dictionary  "iroha"
                 "fuzokugo"
                 :user "dicname" )
```

(2) ユーザ辞書ファイルの作成

mkdic(1) によりユーザ辞書 (テキスト形式) の作成が可能です。

これらのコマンドにより辞書ディレクトリおよび辞書ファイルが作成され、必要に応じて dics.dir ファイルが更新あるいは作成されます。

(例)

```
% mkdic dicname
```

以下の操作により mkdic(1) と同じ処理を行うことができます。

(a) 辞書ディレクトリの作成

$(CANNALIBDIR)/dic/user/ の下に自分のユーザ名のディレクトリを作ってください。

(b) 辞書ファイルの作成

そのディレクトリに filename.t というファイルを作ります。 filename は任意の名前で、これを dics.dir に書き込みます。登録はテキストファイルに行いますので、必ず .t という拡張子を付けてください。ファイルには何も書き込む必要はありません。

このファイルのグループは bin とし、グループに対する書き込み権を許可しておいてください。これを行わないと単語登録が反映されない場合があります。

(c) dics.dir の作成

ユーザ辞書ファイルと同じディレクトリに dics.dir というファイルを作ってください。

dics.dir はそのディレクトリ下にある辞書の一覧を保持しているファイルです。このファイルには以下のように辞書ファイルの名前、辞書の種類、辞書名、辞書のアクセス権を書き込みます。

例)

```
        filename.t (.mwd)      -dicname-rw-
辞書ファイル名      ↓        辞書名とアクセス権
    子辞書名 (自立語辞書であることを指定)
```

注意 アクセス権 (-rw-) は、 r は READ 権あり、 w は WRITE 権ありを表します。

3.2.11 グループ辞書の設定

単語登録を行うには、カスタマイズ機能によりユーザ辞書の設定を行う必要があります。カスタマイズ機能についての詳細は **2 カスタマイズ** を参照してください。

(1) .canna の作成

ホームディレクトリに .canna というファイルを作成し、以下のように書き込んでください。引数には辞書名を記述します。最初の 2 行は基本になる辞書の名前ですので必ず記述してください。

はじめは $(CANNALIBDIR)/sample の下の default.canna というファイルをコピーして用いると良いでしょう。

例)

```
(use-dictionary  "iroha"
                 "fuzokugo"
```

```
                    :user "dicname" )
```

(2) グループ辞書ファイルの作成

mkdic(1) によりグループ辞書 (テキスト形式) の作成が可能です。

これらのコマンドにより辞書ディレクトリおよび辞書ファイルが作成され、必要に応じて dics.dir ファイルが更新あるいは作成されます。

(例)

```
% mkdic -G dicname
```

以下の操作により mkdic(1) と同じ処理を行うことができます。

(a) 辞書ディレクトリの作成

$(CANNALIBDIR)/dic/group/ の下に共有するグループ名のディレクトリを作ってください。

(b) 辞書ファイルの作成

そのディレクトリに filename.t というファイルを作ります。 filename は任意の名前で、これを dics.dir に書き込みます。登録はテキストファイルに行いますので、必ず .t という拡張子を付けてください。ファイルには何も書き込む必要はありません。

このファイルのグループは bin とし、グループに対する書き込み権を許可しておいてください。これを行わないと単語登録が反映されない場合があります。

(c) dics.dir の作成

グループ辞書ファイルと同じディレクトリに dics.dir というファイルを作ってください。

dics.dir はそのディレクトリ下にある辞書の一覧を保持しているファイルです。このファイルには以下のように辞書ファイルの名前、辞書の種類、辞書名、辞書のアクセス権を書き込みます。

例)

```
        filename.t (.mwd)      -dicname-rw-
辞書ファイル名     ↓       辞書名とアクセス権
      子辞書名 (自立語辞書であることを指定)
```

注意 アクセス権 (-rw-) は、 `r` は `READ` 権あり、 `w` は `WRITE` 権ありを表します。

3.2.12 Wnn対応

今までかな漢字変換システムとして Wnn を使用してきたユーザに『かんな』を有効に御使用していただくための簡易ツールを用意しました。

おそらく、 Wnn を使い込んできたユーザには自分なりの辞書を持っていることでしょう。そこで Wnn ユーザは Wnn での使い慣れた辞書を『かんな』の形式に変換することにより、本システムで今までどおりのかな漢字変換ができます。このために、 Wnn のテキスト形式辞書を『かんな』のテキスト形式辞書に変換するコマンド wtoc(1) があります。

また、『かんな』と Wnn の両方を使用しているユーザは『かんな』での辞書を Wnn と共有することができれば便利でしょう。そのために『かんな』のテキスト形式辞書を Wnn の形式に変換するコマンド ctow(1) があります。 wtoc と ctow により Wnn と『かんな』間でかな漢字変換辞書を相互に利用できます。

第 4 章

かな漢字変換ユーティリティ

本システムではかな漢字変換辞書のメンテナンスやかな漢字変換サーバのメンテナンスを行うための種々のユーティリティを提供しています。ユーティリティは辞書をメンテナンスするものとその他の 2 つに大きく分けられ、辞書をメンテナンスするものに関しては、かな漢字変換サーバを介して辞書のメンテナンスが可能なものと、辞書ファイルが存在するマシン上でなければメンテナンスができないものがあります。

この章ではこれらのユーティリティツールに関する説明をマニュアルページの形で示します。

辞書の変換や、新しい辞書の作成などのために以下のようなユーティリティを提供しています。

canvert	旧日本語入力システムのファイルを本日本語入力システムのファイルに変換する。
mkromdic	テキスト形式のローマ字かな変換テーブルをバイナリ形式に変換する。
mkbindic	テキスト形式の辞書をバイナリ形式の辞書に変換する。
dpbindic	バイナリ形式の辞書をテキスト形式の辞書に変換する。
dpromdic	バイナリ形式のローマ字かな変換テーブルをテキスト形式に変換する。
dicar	バイナリ形式の辞書のアーカイブを行う。

以下はかな漢字変換サーバを介して辞書のメンテナンスが可能なツールです。

mkdic	辞書の作成／アップロードを行う。
lsdic	辞書の一覧を出力する。
rmdic	辞書を削除する。
mvdic	辞書の名前を変更する。
cpdic	辞書をコピーする。
catdic	辞書のダウンロードを行う。
addwords	単語の一括登録を行う。
delwords	単語の一括削除を行う。
chmoddic	辞書のアクセス権の変更を行う。

また、初期状態のチェックや、かな漢字変換サーバに関する以下のようなコマンドを提供しています。

cshost	かな漢字変換システムのサーバ・アクセス制御をする。

cannacheck	日本語入力システム関連情報を表示する。
cannakill	かな漢字変換サーバ cannaserver を終了させる。
cannaserver	かな漢字変換のサービスを提供する。
cannastat	かな漢字変換サーバの情報を表示する。

カスタマイズファイルを自動的に作成修正するツールを提供しています。

ccustom	日本語入力システムのカスタマイズをする。

かな漢字変換システム Wnn とのテキスト辞書互換ツールを提供しています。

ctow	本日本語入力システムのテキスト形式の辞書を Wnn のテキスト形式の辞書に変換する。
wtoc	Wnn のテキスト形式の辞書を本日本語入力システムのテキスト形式の辞書に変換する。
splitword	複数のテキスト形式の辞書を１つにして、１行１品詞１候補にする。

端末での日本語環境設定のために以下のコマンドを提供しています。

jsetconv	入出力時の日本語のコード変換の設定を行う。
jset	日本語入出力環境のセットアップを行う。
cannad	端末でのかな漢字変換を行う。

(各コマンドのマニュアルページを出力して、この部分にはさんでください。)

第 5 章

かな漢字変換ライブラリ

5.1 ライブラリ概説

5.1.1 ライブラリと階層

『かんな』では、かな漢字変換ライブラリとして以下の 2 つのライブラリを提供しています。

- ユーザインタフェースライブラリ

  1 文字の入力、1 キーの入力に対応して、未確定文字列、確定文字列、注目文節位置、ステータス表示のための文字列、候補一覧のための文字列などを返還するライブラリです。低レベルかな漢字変換ライブラリに見られるような辞書のメンテナンスや、学習の制御などに関する機能は、このライブラリでは提供しません。日本語入力ユーザインタフェースの大まかなラインは規定されますが、表示部分に若干の工夫をする余地は残ります。X ウィンドウにおいて、図形と文字を同じウィンドウ内で入力 / 表示したいアプリケーションはこのアプリケーションインタフェースを用いることをお勧めします。

- RK ライブラリ

  辞書のアクセスや候補の選択を直接制御したいときに用います。このレベルのかな漢字変換ライブラリでは、かな漢字変換システムの利用開始、読みから漢字への変換、文節の切り分けの変更、次候補 / 前候補の取り出し、学習の制御、辞書のメンテナンスなどの処理に対する関数インタフェースを提供します。低レベルライブラリは中レベルアプリケーションインタフェースや高レベルアプリケーションインタフェースを構築するために用いられています。一般のユーザは用いないほうが良いでしょう。

5.1.2 ライブラリとヘッダファイルについて

本システムは、ライブラリ 1 つとヘッダファイル 2 つを提供しています。それぞれの詳細を以下に示します。

- **ライブラリ**

  (1) libcanna.∗ 日本語入力のためのライブラリです。

- **ヘッダファイル**

(1) jrkanji.h (/usr/include/canna) TTY レベルの日本語の入出力を行うときに必要なヘッダファイルです。

(2) RK.h (/usr/include/canna) RK ライブラリを使用するとき必要なヘッダファイルです。

下表の関数を利用する場合は、表に示すライブラリをリンクし、ヘッダファイルをインクルードしてください。

関　　数	ライブラリ	ヘッダファイル
jrKanjiString	libcanna.∗	canna/jrkanji.h
RK ライブラリ	libcanna.∗	canna/RK.h

注意 ライブラリ名の ”.∗” は、 ”.a”、 ”.so” などを表します。

5.1.3 libcanna 使用時の注意事項

libcanna.∗ をリンクするアプリケーションを作成する際には、以下の点に注意してください。

- libucb と libcanna は同時にリンクすることはできません。

  したがって、 `/usr/ucb/cc` を用いてコンパイルするようなアプリケーションや、 libucb を明示的にリンクするようなアプリケーションでは、 libcanna をリンクしないでください。

- libcanna の提供する関数はマルチスレッドに対応していません。

  したがって、 EWS-UX/V(Rel4.2MP), UP-UX/V(Rel4.2MP) においてマルチスレッド機能を用いる場合、 libcanna で提供する関数は単一のスレッドからのみ呼び出してください。

5.2 ユーザインタフェースライブラリ

ユーザインタフェースライブラリは、 1 文字の入力、 1 キーの入力に対応して、未確定文字列、確定文字列、注目文節位置、ステータス表示のための文字列、候補一覧のための文字列などを返還するライブラリです。アプリケーションプログラムは、返還された情報によって表示します。アプリケーションプログラムは、キー入力以外のトリガー (たとえばマウスによる選択など) によってモードの遷移を制御することもできます。

ユーザインタフェースライブラリは、 jrKanjiString などの汎用ベースの関数を提供します。

ユーザインタフェースライブラリの関数と、その関数を使用する際に必要なファイルを表 5.1に示します。

表 5.1: ユーザインタフェースライブラリ関数およびそのファイル一覧

用　途	文字コード	関　数	ライブラリ	ヘッダファイル
汎　用	EUC	jrKanjiString	libcanna.∗	canna/jrkanji.h
		jrKanjiControl		
		jrCloseKanjiContext		
	ワイド キャラクタ	wcKanjiString		
		wcKanjiControl		
		wcCloseKanjiContext		
	共　通	kanjiInitialize		
		kanjiFinalize		
		createKanjiContext		

注意 ライブラリ名の ”.∗” は、 ”.a”、 ”.so” などを表します。

5.2.1 提供機能

jrKanjiString	TTY からの入力、 X での入力、基本ウィンドウでの入力など一般的なキーボード入力に対して漢字文字列への変換を行います。漢字文字列は EUC コード扱いです。
jrKanjiControl	指定されたパラメータにより、 jiKanjiString での制御を行います。
wcKanjiStirng	jrKanjiString と基本的な機能は同じです。ほとんどの文字列の扱いがワイドキャラクタになっています。
wcKanjiControl	wcKanjiString での制御を指定されたパラメータにより行います。
kanjiInitialize	かな漢字変換の初期化を行います。
kanjiFinalize	プログラム終了などに伴うかな漢字変換終了処理を行います。
createKanjiContext	かな漢字変換に使用するコンテクストを作成します。
jrCloseKanjiContext	jrKanjiString でかな漢字変換に使用したコンテクストをクローズします。
wcCloseKanjiContext	wcKanjiString でかな漢字変換に使用したコンテクストをクローズします。

5.2.2 利用ガイド

ここではユーザインタフェースライブラリの使用法について説明します。

5.2.2.1 簡単な例

以下にユーザインタフェースライブラリを用いた簡単な例を示します。

```
 1  #include <stdio.h>
 2  #include <canna/jrkanji.h>
 3
 4  #define MAX_SIZE 1024
 5
 6  main()
 7  {
 8    int c, nbytes;
 9    unsigned char workbuf[MAX_SIZE];
10    jrKanjiStatus ks;
11
12    jrKanjiControl(0, KC_INITIALIZE, 0);
13    {
14      jrKanjiStatusWithValue ksv;
15
16      ksv.ks = &ks;
17      ksv.buffer = workbuf;
18      ksv.bytes_buffer = MAX_SIZE;
19      ksv.val = CANNA_MODE_HenkanMode;
20      jrKanjiControl(0, KC_CHANGEMODE, &ksv);
21    }
22    c = getchar();
23    while (c != EOF) {
24      nbytes = jrKanjiString(0, c, workbuf, MAX_SIZE, &ks);
25      if (nbytes > 0) {
26        workbuf[nbytes] = '\0';
27        printf("%s\n", workbuf);
28      }
29      c = getchar();
30    }
31    jrKanjiControl(0, KC_FINALIZE, 0);
32    exit(0);
33  }
```

このプログラムをコンパイルし、実行した結果は以下のようになります。以下の例で ␣ は半角のスペー

ス文字を表します。

```
% cc sample1.c -lcanna -o sample1
% sample1
koreha␣kantannna␣puroguramudesu.␣
これは
簡単な
プログラムです。
% 
```

ただし、 SVR4 系マシンでは、 -lsocket、 -lnsl を指定してください。

上記プログラムでは、 12 行目 〜 21 行目で初期化処理、 22 行目 〜 30 行目でローマ字かな変換 / かな漢字変換処理をしています。 31 行目は終了処理です。上記プログラムの詳細について以下で説明します。

5.2.2.2 ユーザインタフェースライブラリ

ユーザインタフェースライブラリでは、個々の入力に対して表示すべき文字列と確定した文字列を返します。たとえば、「kanjini<space>shimasu<return>」のような入力があったときに、ユーザインタフェースライブラリに対する入出力は以下のようになります。

入　力	出　　力	
	表　　示	確定文字
k	k	
a	か	
n	かn	
j	かんj	
i	かんじ	
n	かんじn	
i	かんじに	
<space>	漢字に	

入　力	出　　力	
	表　　示	確定文字
s	s	漢字に
h	sh	
i	し	
m	しm	
a	しま	
s	しまs	
u	します	
<return>		します

アプリケーションプログラムの処理は以下の 3 つから構成されます。

(1) 1 文字ごとの入力をユーザインタフェースライブラリに与える。

(2) その結果ユーザインタフェースライブラリが返す中間表示文字列を表示する。

(3) 確定文字列として返ってきた文字列を入力して取り扱う。

すなわち、アプリケーションプログラムは何らかの入力を 1 文字ごとにライブラリに与え、その結果得られる中間結果文字列を表示する処理を行います。もちろん、確定文字列として得られる文字列は自分の入力として扱います。

スペースキーが変換キーであるとか、矢印キーが右移動であるなどのユーザインタフェースにかかわる処理はライブラリ内部で行われるので、アプリケーションプログラムはそのような煩わしいことに関知することなく日本語入力が行えます。

上記で説明した中間結果および確定文字列を返す関数は jrKanjiString という関数です。前述のプログラムでは 24 行目に使われており、プログラム中で中心的な役割を行っています。

この関数を使うためには、まず、次のようにヘッダファイルをインクルードしなければなりません。

```
#include <canna/jrkanji.h>
```

jrKanjiString は以下のような関数仕様になっています。

```
jrkanjiString(context_id, ch, buffer_return,
              bytes_buffer, kanji_status_return)
int context_id;
int ch;
char *buffer_return;
int bytes_buffer;
jrKanjiStatus *kanji_status_return;
```

引数には *context_id*、 *ch*、 *buffer_return*、 *bytes_buffer*、 *kanji_status_return* があります。

context_id にはコンテクスト識別子を指定します。
コンテクストに関しては、 **5.2.2.3 コンテクスト** を参照してください。

ch は入力された文字です。

buffer_return には確定した文字列を返すためのバッファを指定します。このバッファはアプリケーションプログラムが準備します。確定文字列ができると jrKanjiString は確定文字列をこのバッファに格納して返します。

bytes_buffer は *buffer_return* のバッファ長です。 jrKanjiString は、どんなに長い文字列が確定文字列として発生した場合においても *bytes_buffer* で与えられた長さ以上の文字を返すことはありません。

kanji_status_return は中間表示のための情報、その他が入っている jrKanjiStatus 型の構造体です。
jrKanjiStatus 型については、 **5.2.2.6 中間表示** を参照してください。

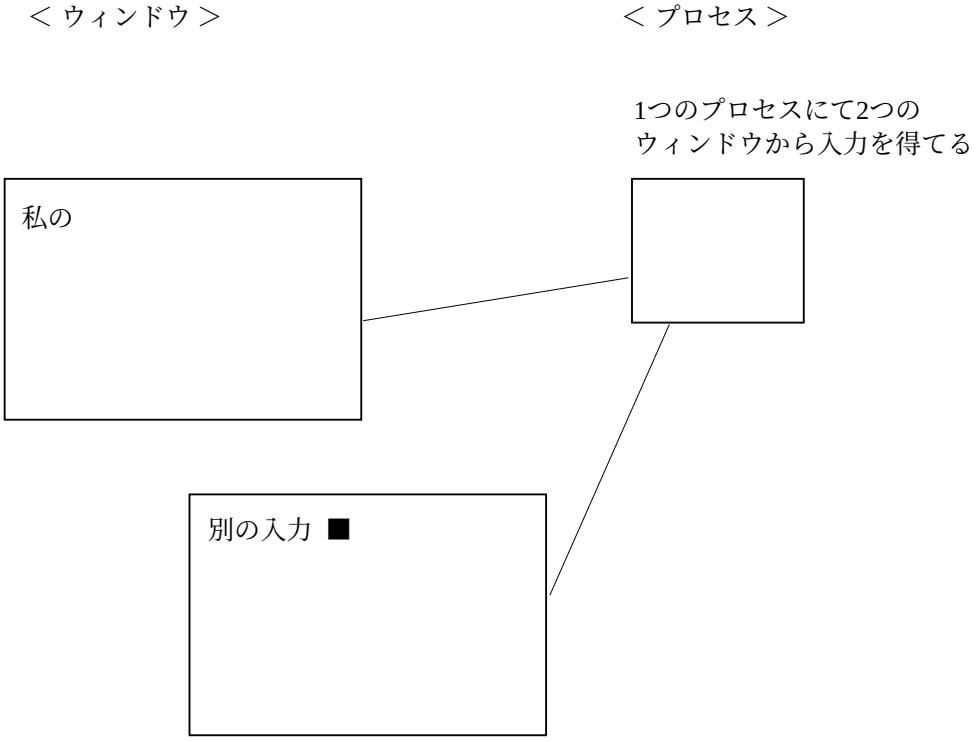
5.2.2.3 コンテクスト

ユーザインタフェースライブラリの呼び出しには必ず *context_id* を指定します。

さて、 *context_id* とは何でしょうか。

アプリケーションプログラムが 2 つのウィンドウを同時に表示しており、それぞれのウィンドウにおいて入力を受け付けている状況を考えてください。

図 5.1: コンテクスト説明図



2 つのウィンドウでの入力は独立して行われると考えるのが自然です。

すなわち、1 つのウィンドウで『わたしの ＜ 変換 ＞』まで入力したときに他のウィンドウで入力を始めても、最初のウィンドウでは『わたしの ＜ 変換 ＞』の状態のままであるのが望ましいわけです。

このような処理はかな漢字変換を行っているプロセスがウィンドウごとに別々であれば自然に行える処理ですが、単一のプロセスで複数の入力部を持つ場合には入力の切り分けをうまく行わなければなりません。

上記の場合、キーの入力がどちらのウィンドウのものであるかを区別する識別子が必要になります。その識別子が *context_id* です。 *context_id* はユーザインタフェースライブラリを呼び出すときに、どの入力ポートに関するものであるかを表すものです。すなわち、かな漢字変換の処理単位を示す識別子なのです。

context_id には、 int 型の値であり、複数の入力部間で重複することのない値であれば、どのような値を与えてもかまいません。たとえば、 open で返されるファイル記述子や fopen で返されるファイルポインタを int 型にキャストしたものでも良いのです。特に X ウィンドウのプログラミングでは Widget 構造体へのポインタを int 型にキャストして用いると良いでしょう。

コンテクストを切り分ける必要がない場合には *context_id* として 0 を指定します。

5.2.2.4 初期化処理

きちんとした初期化処理を行わずに jrKanjiString を呼び出した場合でも、 jrKanjiString の最初の呼び出し時に自動的に初期化処理が行われます。しかし、明示的に初期化処理を行う方が望ましいプログラミングスタイルといえます。

かな漢字変換処理を初期化するためには以下の処理を行います。

```
char **warn;

jrKanjiControl(context_id, KC_INITIALIZE, &warn);
```

jrKanjiControl はかな漢字変換処理に関する制御を行うための関数です。第 1 引数には *context_id* を与え、第 2 引数にはどのような制御を行うかを表すコントロールキーワードを与えます。第 3 引数には警告メッセージのポインタの格納場所を与えます。

初期化を行う場合には、第 2 引数に KC_INITIALIZE を指定します。前述のサンプルプログラムでは 12 行目で呼び出しています。

初期化時には以下の 4 つの処理が行われます。

(1) カスタマイズファイルの読み込み

(2) ローマ字かな変換テーブルの読み込み

(3) かな漢字変換サーバとの接続

(4) 辞書のマウント処理

初期化処理にはコンテクストは関係しないので、 *context_id* としては 0 を与えます。

第 3 引数には警告メッセージが返されます。初期化処理において警告が発生した場合、第 3 引数で指定された変数に警告メッセージへのポインタが格納されます。警告メッセージを標準エラー出力に表示したい場合には以下のような処理を行います。

```
char **warn;

jrKanjiControl(0, KC_INITIALIZE, &warn);
if (warn) {
  char **p;

  for (p = warn ; *p ; p++) {
    fprintf(stderr, "%s\n", *p);
  }
}
```

警告メッセージが必要でない場合は第 3 引数に 0 を指定します。

5.2.2.5 終了処理

かな漢字変換の終了処理を行う場合は、 jrKanjiControl の第 2 引数に KC_FINELIZE を指定します。前述のサンプルプログラムでは 31 行目で呼び出しています。

終了時には以下の処理が行われます。

(1) 学習情報の辞書への書き込み
(2) 辞書のアンマウント
(3) かな漢字変換サーバとの接続の切断
(4) ローマ字かな変換テーブルの解放
(5) 各種カスタマイズ設定のリセット
(6) かな漢字変換に使われていたメモリの解放

終了処理時の第 3 引数の取扱いは初期化処理時と同じになります。

終了処理は、プログラム終了時およびかな漢字変換終了時には必ず呼び出してください。

5.2.2.6 中間表示

中間表示情報は jrKanjiStatus 構造体で返されます。 jrKanjiStatus について説明する前に、中間表示にはどのようなものがあるかについて以下で説明します。

図 5.2: 中間表示図

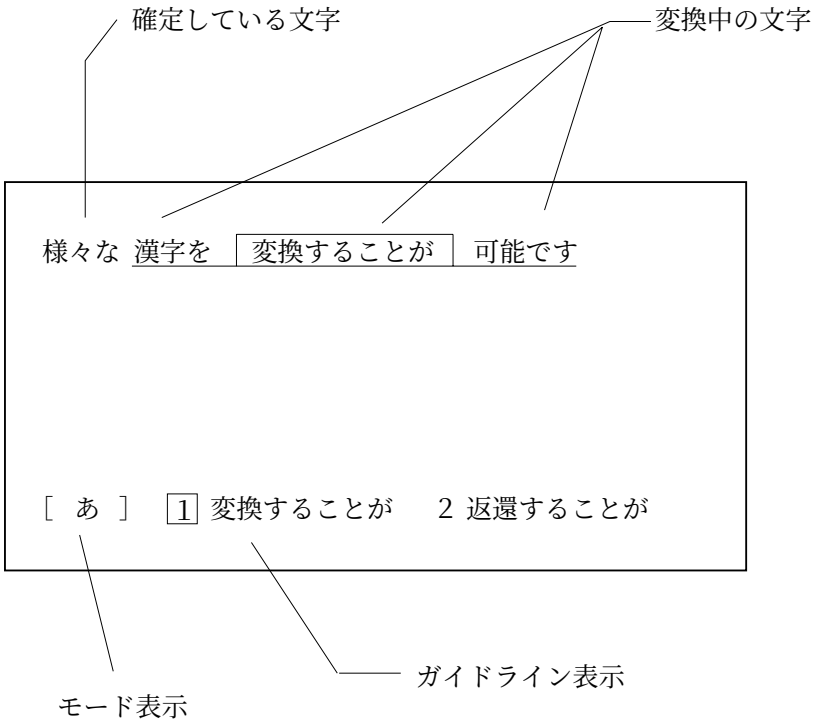


図 5.2に示されるように中間表示としては以下の 3 つがあります。

(1) 変換の様子を示す入力部

(2) モードを表すモード表示部

(3) 候補一覧など、その他の情報を表すガイドライン表示部

入力部およびガイドライン表示部は表示の一部が反転するといった強調表示が行われることがあります (図 5.2 において四角で囲まれている部分)。jrKanjiStatus 構造体では、このような強調表示される部分がどこであるのかといった情報も返されます。

中間表示のための情報は以下の構造体により返されます。

```
typedef struct {
  unsigned char *echoStr;  /* 入力部分の文字列 */
  int           length;    /* その長さ */
  int           revPos;    /* 強調表示の開始位置 */
  int           revLen;    /* 強調表示の長さ */
  int           info;      /* その他の情報。下を見よ */
  unsigned char *mode;     /* モード情報 */
  struct {
    unsigned char *line;   /* ガイドライン部分の文字列 */
    int           length;  /* その長さ */
    int           revPos;  /* 強調表示の開始位置 */
    int           revLen;  /* 強調表示の長さ */
  } gline;
} jrKanjiStatus;

/* information flags */
#define KanjiModeInfo   01
#define KanjiGLineInfo  02
```

(1) **入力部の情報**

入力部の情報は jrKanjiStatus 構造体の echoStr、length、revPos、revLen の各メンバに返されます。それぞれの意味は上記の構造体の定義中にコメントで表されています。

- **length が −1 のとき－入力部に変化がない**

  length が −1 のときは、入力部に変化がなかったことを表します。たとえば、かな漢字変換入力中にシフトキーなどの単発的な入力があってもかな漢字変換入力操作に影響を与えません。このような場合には length メンバに −1 が返ります。

- **length が 0 のとき－入力部がない**

  length メンバが 0 である場合には入力部がなくなったということを表します。−1 の場合と混同しないようにしてください。

- **revPos、 revLen －強調表示部**

  revPos、 revLen メンバは強調表示部の範囲を表します。入力部の表示に関しては、入力部全体に下線をほどこし、強調部のみを反転するという表示で良いでしょう。あるいは、入力部全体の色を確定しているテキストの色とは別の色にし、さらに強調部については別の色にするなども考えられます。表示の方法についてはアプリケーションプログラムの自由ですので、いろいろと工夫してみてください。

- **echoStr －入力部文字列**

  echoStr には中間表示文字列へのポインタが格納されますが、この文字列を格納するためのメモリ領域はライブラリ内部に取られています。この文字列バッファをアプリケーションプログラムが書き換えたり、 free したりしないでください。そのような場合には予測されない結果を生じることがあります。

(2) **モード表示**

モード表示に変化が生じた場合 (すなわちモードが変わった場合) には info メンバの KanjiModeInfo ビットが立ちます。したがって、モード表示に変化があったかどうかは以下のように判断することができます。

```
jrKanjiStatus ks;
int nbytes, ch;
.....
nbytes = jrKanjiString(0, ch, buf, bufsize, &ks);
.....

if (ks.info & KanjiModeInfo) { /* モードに変化がある */
  モード表示を変更する処理 ;
}
```

- **info －モードが変化したかどうか**

  info メンバに対して KanjiModeInfo ビットが立っている場合には、 mode メンバにモードを表す文字列が格納されています。アプリケーションプログラムは、このモード文字列をしかるべきモード表示領域に表示する必要があります。

- **mode －モード文字列**

  mode メンバで返される文字列用の領域もユーザインタフェースライブラリ内部で確保されていますので、文字列の内容を勝手に書き換えたり、 free したりしないでください。

(3) **ガイドライン表示部の情報**

ガイドラインの表示に変化が生じた場合には info メンバの KanjiGLineInfo ビットが立ちます。したがって、ガイドラインの表示に変化が生じたかどうかは、モード表示の変化と同様の方法で調べることができます。唯一の違いは、調べるビットが KanjiModeInfo ではなく、 KanjiGLineInfo であることです。

- **info −ガイドライン表示に変化があるかどうか**

  info メンバの KanjiGLineInfo ビットが立っていた場合にはガイドラインの情報が jrKanjiStatus 構造体を介して返されます。 jrKanjiStatus 構造体の gline メンバがそれに当たります。 gline メンバはさらに構造体の形になっており、内部メンバとして、 line、 length、 revPos、 revLen を持っています。これは、 line が echoStr に対応していると考えれば、 jrKanjiStatus 構造体の入力部分の情報を表している echoStr、 length、 revPos、 revLen と同じ意味を持ちます。

  すなわち、以下のようになります。

  – **line −ガイドライン文字列**

    line にはガイドラインに表示する文字列のポインタが返ります。

  – **length −ガイドライン文字列の長さ**

    length には line の長さ (バイト単位) が返ります。

  – **revPos、 revLen −強調表示部分**

    revPos、 revLen にはそれぞれ、 line のうち、強調表示する部分の位置と長さが返ります。

  入力部とは違い、ガイドライン情報については、 length として −1 が返ることはありません。

  入力部では、表示に変化がない場合には length に −1 が返りましたが、ガイドラインに関しては表示に変化がないことは、単に info メンバの KanjiGLineInfo が立たないことによって表されます。

  KanjiGLineInfo が立っていないということは、ガイドラインに表示する情報が変化していないということであり、ガイドラインに表示する情報が存在しないという意味ではありません。

5.2.2.7 初期化ふたたび

さて、ここまでの説明でひととおりかな漢字変換を伴うアプリケーションプログラムが記述できるようになったことと思います。

これより以降では、実使用に耐えるアプリケーションプログラムを作成するときに必ず守ってもらいたい事項や、このことを知っていればさらにいろいろな機能を活用できるようになる、といった事項について説明します。

まず始めは初期化のときの処理です。

初期化のときに初期化ファイルを読み込んだりかな漢字変換サーバに接続したりする処理が行われますが、どの初期化ファイルを読み込むかとか、どのかな漢字変換サーバに接続するかなどがアプリケーションプログラムから制御することができます。

また、初期化直後にアプリケーションの名前をサーバに通知する必要があります。

さらに、かな漢字変換を始めるに当たってモード表示領域やガイドライン表示領域を設けなければなりませんが、それがどのくらい必要であるかをあらかじめ知っておく必要があります。

典型的な初期化の処理は次のようになります。

	キ ー ワ ー ド	処 理 内 容	備 考
1	KC_SETINITFILENAME	初期化ファイルの指定	オプション
	↓		
2	KC_SETSERVERNAME	接続するかな漢字変換サーバの指定	オプション
	↓		
3	KC_INITIALIZE	かな漢字変換の初期化	
	↓		
4	KC_SETAPPNAME	アプリケーション名の通知	
	↓		
5	KC_QUERYMAXMODESTR	モード表示文字列の最大長の取得	
	↓		
6	KC_QUERYMODE	モード文字列の取得	
	↓		
7	KC_SETWIDTH	ガイドライン表示部分の大きさの設定	

上記の表でオプションとあるものについては実行してもしなくても構いません。オプションとあるもの以外 (3 〜 7) に関しては必ず実行しておくことを強くお勧めいたします。

以下では上記の各々について説明します。

5.2.2.7.1 初期化ファイルの指定 初期化ファイルとして、アプリケーション固有のファイルを用いるようにすることが、 jrKanjiControl を用いてできます。それには KC_SETINITFILENAME を用います。 KC_SETINITFILENAME の詳しい仕様についてはマニュアルページの方を参照してもらうことにして、ここではその使用例を示します。

KC_SETINITFILENAME は KC_INITIALIZE を行う前に設定します。なぜなら初期化ファイルの読み込みは KC_INITIALIZE ときに行われるからです。 KC_SETINITFILENAME を行うと、環境変数 CANNAFILE にいかなる値が設定されていようとも KC_SETINITFILENAME で指定されたものの方が優先されることになります。

```
{
  char *initfile = "app-own.canna";
  .....

  nbytes = jrKanjiControl(0, KC_SETINITFILENAME, initfile);
  .....

  jrKanjiControl(0, KC_INITIALIZE, &warn);
}
```

5.2.2.7.2 接続するかな漢字変換サーバの指定 接続するかな漢字変換サーバをアプリケーションプログラムから指定することができます。指定には KC_SETSERVERNAME を用います。この指定は環境変数 CANNAHOST よりも優先されます。

たとえばこれにより、アプリケーションプログラムのオプションとしてかな漢字変換サーバの指定を含めることができます。本日本語入力に関して提供されている多くのシステムコマンド (lsdic など) では接続するかな漢字変換サーバを接続するオプションが準備されており、”−cs *server* −*name*”で指定することができるようになっています。

本ライブラリを用いるアプリケーションプログラムで、かな漢字変換サーバを指定するオプションを備える場合、”−cs”オプションにするようお勧めいたします。

KC_SETSERVERNAME は KC_SETINITFILENAME と同様 KC_INITIALIZE よりも前に呼び出さなければなりません。次のようなイメージになります。

```
{
  char *servername = "app-own-server";
  .....

  nbytes = jrKanjiControl(0, SETSERVERNAME, servername);
  .....

  jrKanjiControl(0, KC_INITIALIZE, &warn);
}
```

5.2.2.7.3 アプリケーション名の通知 初期化処理が成功したときには、 KC_SETAPPNAME を用い、接続しているかな漢字変換サーバにアプリケーション名を通知します。

KC_SETAPPNAME で設定したアプリケーション名は KC_FINALIZE を行うまで有効です。したがって、 KC_INITIALIZE を行った場合には、そのつど実行する必要があります。

```
{
  char appname[CANNA_MAXAPPNAME];
  .....
  strcpy(appname, "cannad");
  .....
  jrKanjiControl(0, KC_INITIALIZE, &warn);
  .....
  jrKanjiControl(0, KC_SETAPPNAME, appname);
  .....
}
```

5.2.2.7.4　モード表示文字列の最大長の取得　モード表示文字列は利用者の初期化ファイルの設定にしたがって長さが大きく変わります。

KC_QUERYMAXMODESTR を用いればモード文字列のうち最大のコラム長を持つ文字列のコラム長を得ることができます。ここで、コラム長とは半角英数文字の文字幅を 1 としたときの文字幅になります。たとえば全角文字は 2 コラムとなります。

KC_INITIALIZE を呼び出したら、モード文字列表示領域としてどれだけの大きさが必要であるかを、KC_QUERYMAXMODESTR を呼び出して確認してください。

```
{
  int max_mode_columns;
  .....
  jrKanjiControl(0, KC_INITIALIZE, &warn);
  .....
  max_mode_columns = jrKanjiControl(0, KC_QUERYMAXMODESTR, 0);
  .....
}
```

5.2.2.7.5　モード文字列の取得　モード表示のための領域の幅を取得したら、次は最初に表示しておくべき文字列を取得します。

システムで提供しているデフォルトのユーザインタフェースでは、最初のモードはアルファベットモードであり、表示されるモード文字列は半角空白文字 6 文字となります。したがって真っ白い表示にしておけば良いことになります。

ただしユーザのカスタマイズのやり方しだいでいろいろなモードが最初のモードとなることがあり、モード文字列としても ””以外のものになることがあります。

アプリケーションプログラムはユーザのカスタマイズによってどのようなモードが最初のモードになっているかを、 jrKanjiControl の KC_QUERYMODE によって入手し、表示する必要があります。

それには、以下のような処理をすることになります。

```
{
  unsigned char currentMode[MAXMODEELEN];
  .....
  jrKanjiControl(0, KC_QUERYMODE, currentMode);
  .....
}
```

この処理により文字配列 currentMode に初期モード文字列が返されますので、アプリケーションプログラムはこのモード文字列をモード表示領域に表示してください。

5.2.2.7.6 ガイドライン表示部分の大きさの設定 ガイドラインには候補一覧などが表示されますが、アプリケーションプログラムの都合にしたがって、ガイドライン領域の幅を調整することができます。ユーザインタフェースライブラリはアプリケーションプログラムから指示された幅に収まるようにガイドライン表示を行います。たとえば候補一覧を表示する場合にはガイドラインに入る範囲で候補を表示します。

ガイドラインの幅を指定するには jrKanjiControl を KC_SETWIDTH を伴って呼び出します。

```
.....
jrKanjiControl(0, KC_INITIALIZE, &warn);
.....
jrKanjiControl(0, KC_SETWIDTH, 72);
.....
```

もし、ガイドラインがあまりにも小さすぎてユーザインタフェースでどう調整しても入り切らない場合には次のいずれかの処理が行われます。

(1) ガイドラインに表示し切れなかった文字列を入力部分に表示する。

(2) 表示を行わない。

KC_SETWIDTH は入力コンテクストごとに別々の値を設定することができます。

逆に言うと入力コンテクストを複数持っているアプリケーションプログラムはそれぞれのコンテクストに対して KC_SETWIDTH を行わなければならないことになります。

複数の入力コンテクストを利用されるかたはご注意ください。

5.2.2.8 再初期化

カスタマイズファイルを修正したとき、その修正を即座に反映させるために、再初期化処理が行えると便利です。もしアプリケーションプログラムが再初期化処理を備えていれば、ユーザがカスタマイズファイルを書き換えたときに、アプリケーションプログラムを再起動しなくてもカスタマイズの結果をアプリケーションプログラムに反映させることができます。

以下の項目を満たすようなアプリケーションプログラムでは再初期化処理を備えておくことをお勧め致します。

(1) 一度起動したらかなり長い間起動しておく

(2) 起動や終了の処理が比較的たいへん

(3) 日本語入力をよく使う

たとえば Nemacs などが上を満たす良い例になります。

このようなアプリケーションプログラムではプルダウンメニューなどに “かな漢字変換の再初期化” の項目を入れておくことをお勧めします。

再初期化処理は、 KC_FINALIZE を行った後、前節で説明した初期化関連の一連の処理を行うことによって達成されます。

5.2.2.9 かな漢字変換処理中の機能

かな漢字変換処理中にも、必要に応じて現在のモードを調べたり、かな漢字変換中の文字を強制的にアプリケーションプログラム側から確定させたりすることが可能です。

それぞれの機能は jrKanjiControl を対応するコントロールキーワードを伴って呼び出すことにより実行されます。実行できる機能には以下のものがあります。

表 5.2: jrKanjiControl 実現機能

	キ ー ワ ー ド	内 容 処 理	備考
1	KC_QUERYMODE	モードの調査	
2	KC_CHANGEMODE	モードの変更	
3	KC_KAKUTEI	入力中の文字列の確定	
4	KC_KILL	入力中の文字列の消去	
5	KC_SETUNDEFKEYFUNCTION	未定義キーの処理	
6	KC_CLOSEUICONTEXT	入力コンテクストのクローズ	

以下ではそれぞれの機能に対応するコントロールキーワードと、その使い方などについて説明します。詳細なアプリケーションインタフェース仕様については　jrKanjiControl の説明の項を参照してください。

5.2.2.9.1 モードの調査　現在の入力モードが何であるのかを前述した KC_QUERYMODE で調べることができます。ただし、以下の理由から KC_QUERYMODE の呼び出しだけでは今どのようなモードであるのか断定することができません。

(1) KC_QUERYMODE の返す値はデフォルトではモード文字列の形である

(2) モード文字列はユーザのカスタマイズにより変更できる。

実は、 KC_SETMODEINFOSTYLE によって KC_QUERYMODE の返す値を調査用に少し変更することができます。 KC_SETMODEINFOSTYLE ではモードとしてどのようなものを採用するかを制御できます。

SETMODEINFOSTYLE に与える値	0	1
表示されるモード	文字列	数値 +'@'(0x40)

KC_SETMODEINFOSTYLE でモードスタイルを 1 にすることにより KC_QUERYMODE で返される値はモードを表す数値に'@'(0x40) を足した値が 1 文字長の文字列として返されるようになります。この値はユーザによるカスタマイズの影響を受けません。モードを表す数値とはヘッダファイル <canna/jrkanji.h> から間接的にインクルードされる <canna/mfdef.h> の中で、 CANNA_MODE_xxx としてマクロ定義されている値です。たとえば、アルファベットモードは CANNA_MODE_AlphaMode というマクロで定義される値を持ちます。

KC_QUERYMODE によって返される値から'@' を引いたものを CANNA_MODE_AlphaMode などと比較することにより現在のモードを明確に判断することができます。

5.2.2.9.2 モードの変更 アプリケーションプログラムの都合でかな漢字変換のモードをたとえばカタカナ入力モードなどに強制的に変更する場合には KC_CHANGEMODE を用います。

5.2.2.9.3 入力中の文字列の確定 / 消去 何らかの都合で入力中の文字列をアプリケーションプログラム側から消去したり確定したりしたいことがあると思います。

たとえば、ダイアログボックスを表示していた場合、 Cancel ボタンなどがクリックされたときはそのダイアログボックス中のすべての入力ポートに対して入力状態をクリアする処理を行う必要があります。

このような場合には KC_KAKUTEI あるいは KC_KILL を行います。

それぞれの意味は以下のようになります。

KC_KAKUTEI	入力中の文字を強制的に確定させる
KC_KILL	入力中の文字を強制的に消去する

いずれの場合もかな漢字変換の状態は基底状態 (何も入力されていない状態) になります。

5.2.2.9.4 未定義キーの処理 かな漢字変換中にかな漢字変換には用いられないキー (未定義キー) が入力されたときの処理を指定することができます。デフォルトでは未定義キーは無視されるだけですが、以下のように設定することができます。

(1) ビープ音を鳴らす

(2) 未定義キーをアプリケーションプログラムに渡す。

この場合、以下の 3 つのうちいずれかを選択できます。

(a) 入力中の文字列を確定する。

(b) 入力中の文字列を消去する。

(c) 入力中の文字列は入力中のままにしておく。

5.2.2.9.5 入力中のコンテクストのクローズ ユーザインタフェースライブラリを用いてかな漢字変換を行う上ではさまざまな形でメモリを消費します。

たとえば入力ポートが増えればその分だけ余分にメモリを消費することになります。

入力にウィンドウを用いるアプリケーションプログラムによっては入力のためのウィンドウを次々と作っては壊していく場合があります。このような場合に jrKanjiString に対して次々と新しいコンテクストを与えて行くとどんどんメモリを消費してしまうことになります。

入力に用いられたウィンドウを破壊する場合にはそのウィンドウからの入力を jrKanjiString に与えるときに用いていたコンテクストをクローズした方が良いでしょう。

コンテクストをクローズすることにより、その入力コンテクストのために確保されていたメモリが解放されます。

コンテクストの解放には KC_CLOSEUICONTEXT を用います。

(各関数のマニュアルページを出力して、この部分にはさんでください。)

第 6 章

こんな症状のときには

日本語入力システムを使っていて、自分が設定しているはずの設定どおりに動かない場合があります。原因はいろいろありますが、それらのうちのいくつかは、なんらかの設定の誤りだったり勘違いから生じていることもあります。ここでは、問題になっている症状に対処するときの助力となるように、考えられる原因とその対処の方法について説明します。

6.1 cannacheck コマンド

日本語入力システムを使用している際に起こる、予期しない症状の原因を調べるための手段の 1 つとして cannacheck(1) コマンドを使用する方法があります。

6.1.1 cannacheck と irohacheck

cannacheck と irohacheck はともに日本語入力システム『かんな』の関連情報を表示するコマンドです。cannacheck は、『かんな』 Version 2.1 以降の日本語入力システムに関する情報、 irohacheck は、『かんな』 Version 1.2 以前の日本語入力システムに関する情報を表示します。

使用している『かんな』のバージョンは拡張機能の『環境設定』→『バージョン表示』にて、調べることができます。使用している『かんな』のバージョンに応じて、 cannacheck と irohacheck を使い分けるようにしてください。

cannacheck を行ったときに反応がない場合や、拡張機能のメニューに『環境設定』という項目がない場合は、『かんな』 Version 1.2 以前のものを使用しています。 irohacheck にて状況を調査してください。

cannacheck、 irohacheck の詳細な使用方法については、マニュアルページをご覧ください。

以下では、 cannacheck の機能についてのみ説明します。

6.1.2 cannacheck の機能

cannacheck コマンドは次の項目についてどのような状況になっているか調べます。

(1) どのカスタマイズファイルが使われるか

(2) カスタマイズファイルのシンタックスエラー

(3) ローマ字かな変換テーブルとしては何が使われるか

(4) どのかな漢字変換サーバが使われるか

(5) どの辞書が使われるか

上記の項目を調査し、問題があればその問題を報告します。 −v オプション (verbose) を指定することにより、問題がない場合でも報告が行われます。

問題となった症状の解析に cannacheck をご利用ください。

6.2 問題のレベル

日本語入力システムが思うように動かない場合に、 cannacheck(1) でどのような問題があるかを調べたとしても何の問題も報告されない場合があります。次のような場合は cannacheck(1) では問題は報告されません。

(1) **環境変数 CANNAFILE に設定したカスタマイズファイルが存在しない場合**

$HOME/.canna あるいは、システムデフォルトの default.canna が用いられ特に問題は報告されません。

(2) **カスタマイズファイルで指定したローマ字かな変換テーブルがカレントディレクトリに存在しないとき**

同名のファイルがホームディレクトリあるいは、システムのローマ字かな変換テーブルディレクトリ ($(CANNALIBDIR)/dic) に存在する場合はそれらの辞書ファイルが用いられ問題は報告されません。

反対に次のような場合はエラーが報告されます。

(1) 環境変数 CANNAHOST で示されるかな漢字変換サーバに接続できないとき。

(2) かな漢字変換辞書が利用できないとき。

(3) 指定したローマ字かな変換テーブルがシステムディレクトリにも存在しないとき。

以下の表を参考にして、各項目を参照してください。

6.3 一般に見られる症状

(1) **クライアントが ”かな漢字変換サーバと通信できません ”というメッセージを出す。**

- **かな漢字変換サーバが起動されていますか？**

  かな漢字変換サーバが起動されていない場合は、かな漢字変換サーバを起動してください。

  かな漢字変換サーバが起動されているのに、該当するかな漢字変換サーバにすべてのユーザクライアントが接続できないときには、 **6.3(3) かな漢字変換が行えない** も参照してください。また、かな漢字変換サーバを起動しようとしても起動できないときには、 **6.5(1) cannaserver が起動できない**を参照してください。

- **意図しないかな漢字変換サーバに接続しようとしていませんか？**

  自分が接続したいホスト以外のかな漢字サーバに接続しようとしていませんか。

  接続しようとしているかな漢字サーバは、環境変数 CANNAHOST、または、ファイル $(CANNALIBDIR)/cannahost によって決定されます。

表 6.1: 症状一覧

症　　状	対応章番号
”かな漢字変換サーバと通信できません”といわれる	6.3 (1)
ローマ字かな変換が行われない	6.3 (2)
かな漢字変換が行われない	6.3 (3)
学習の効果がない	6.3 (4)
”単語登録できませんでした”といわれる	6.3 (5)
部首変換ができない	6.3 (6)
逐次変換ができない	6.3 (7)
個人別学習の効果がない	6.3 (8)
キー入力ができない	6.3 (9)
カスタマイズしたとおりの動作をしない	6.4 (1)
ローマ字かな変換テーブルの効果がない	6.4 (2)
ユーザ辞書のマウントに失敗する	6.4 (3)
かな漢字変換サーバが起動できない	6.5

これらの設定が間違っている場合には、拡張機能のサーバの切り替え機能でサーバを切り替えるか、環境変数、または、ファイル $(CANNALIBDIR)/cannahost を変更してからクライアントを再起動してください。

- **指定したサーバにアクセス権がありますか？**

  接続しようとしているサーバのアクセス権は、 cshost(1) コマンドで確認できます。

  詳細はマニュアルページをご覧ください。

  かな漢字変換サーバの指定の仕方は、 **5.1.2　かな漢字変換サーバの指定** を参照してください。

- **サーバのクライアント数が多くありませんか？**

  『かんな』 Version 3.2 以上のサーバに接続しようとしている場合には、メモリの不足やシステム側で設定しているファイルディスクリプタ数をこえてしまっているために接続できくなっていることがあります。この場合には、未使用のクライアントは終了させるか、他のマシンのサーバに接続するようにしてください。

  『かんな』 Version 3.2 より前のサーバに接続しようとしている場合には、サーバ側で接続できるクライアントの数を制限しています。サーバのファイルディスクリプタ数の最大値は 64 個になっています (ファイルディスクリプタ数はクライアント 1 つに 1 個、バイナリ辞書ファイル 1 つにつき 1 個消費されます)。このため、 クライアント数が多い時にサーバに接続できなくなることがあります。未使用のクライアントは終了させるか、あるいはサーバを増やし 1 サーバに接続するクライアント数の比率を変え、システムの資源を節約する必要があります。

(2) **ローマ字かな変換が行われない**

- **ローマ字かな変換テーブルは存在しますか？**

  まず、 cannacheck でメッセージが出るかどうか確認してください。メッセージが次のような場合は指定されているローマ字かな変換テーブルが存在しません。

  ```
  % cannacheck
  ```

```
ローマ字かな変換テーブル (user.kp) がオープンできません。
%
```

カスタマイズファイルあるいは、ローマ字かな変換テーブルを確認してください。ローマ字かな変換テーブルの形式が正しくない場合にもこのエラーメッセージがでます。

(3) かな漢字変換が行えない

- **かな漢字変換サーバと接続できていますか？**

かな漢字変換サーバと接続できていないと、かな漢字変換が行えません。かな漢字変換サーバと接続できない理由として以下のものがあります。

(a) かな漢字変換サーバが動いていない。

自分が接続しようとしている、かな漢字変換サーバが動いていない場合があります。この原因としては、接続しようとしているマシンで、かな漢字変換サーバを起動するための処理を行っていないことが考えられます。接続しようとしているマシンのかな漢字変換サーバを起動してください。

(b) 意図しないかな漢字変換サーバに接続しようとしていた。

$(CANNALIBDIR)/cannahost や CANNAHOST などにより自分の意図とは別のかな漢字変換サーバに接続しようとしている場合があります。 cannacheck コマンドによって、どのマシンのかな漢字変換サーバに接続しようとしているのかをもう一度確認してください。

(c) UNIX ドメインでの通信ができない。

かな漢字変換サーバとして、同じマシンのサーバを用いるように指定してあり、しかも cannaserver が動いている場合は UNIX ドメインでの通信ができないことが考えられます。その原因としては、以下が考えられます。

(i) /tmp の容量不足

/tmp の容量が不足すると (/tmp の全体の容量の 9 割以上が使用されている場合がこれにあたります)、 UNIX ドメインを行うための /tmp/.iroha_unix/IROHA というファイルが作成されません。このファイルは UNIX ドメインでの通信を行うために cannaserver が作成する特殊ファイルです。 /tmp の容量が不足している場合は、 /tmp 下の領域が十分用意されるように調整してください。

(ii) /tmp/.iroha_unix/IROHA ファイルが存在しない

cannaserver が UNIX ドメインでの通信を行う際に使用する特殊ファイルが、何らかの原因で削除されています。このファイルが存在しないと、 UNIX ドメインではかな漢字変換が行えません。かな漢字変換サーバを一旦終了させ、再度立ち上げ直して下さい。

- **かな漢字変換辞書がマウントできていますか？**

かな漢字変換が全く行えないときは、その原因は基本辞書である ”iroha”あるいは ”fuzokugo”がマウントできないことにあります。

cannacheck −v でマウントしている辞書を表示してみてください。

```
% cannacheck -v
```

カスタマイズファイルとして "/uhome/user/.canna"を用います。
ローマ字かな変換辞書は "/uhome/user/user.kp"を用います。
システム辞書 "iroha" を指定しています。
システム辞書 "fuzokugo" を指定しています。
システム辞書 "hojomwd" を指定しています。
システム辞書 "hojoswd" を指定しています。
システム辞書 "yuubin" を指定しています。
システム辞書 "katakana" を指定しています。
システム辞書 "software" を指定しています。
部首辞書 "bushu" を指定しています。
サーバ "wink"に接続します。

上記のように ”iroha”および ”fuzokugo”が指定されていれば問題ありません。 ”iroha”および ”fuzokugo”が指定されていない場合は、カスタマイズファイルをチェックしてください。

(4) **学習が行われない**

- **(setq gakushu nil) となっていませんか？**

  カスタマイズファイルで ”(setq gakushu nil)”が指定されている場合は学習が行われません。学習を行いたい場合はカスタマイズファイルでのこの指定を取り消すか、 ”(setq gakushu t)”のように記述してください。

- **辞書の書き込み権がありますか？**

  たとえばユーザ登録した単語が頻度学習されない場合はユーザ辞書への書き込み権が cannaserver に対して解放されていないことが考えられます。 cannaserver はユーザ ID が bin(2)、グループ ID が bin(2) で動作していますのでユーザが作成した辞書については辞書のオーナを bin にしてオーナに対する書き込み権を解放するか、辞書のグループを bin にしてグループに対する書き込み権を解放するかしてください。ユーザ辞書はオーナをそのユーザ自身とし、グループを bin としてモードを 664 または 660 にするのが良いでしょう。

- **バイナリ辞書を更新したときに mkdic を行いましたか?**

  OS のバージョンアップなどを行ってバイナリ辞書が更新されたときには個人別学習も更新する必要があります。

  したがって、それまで個人別学習を行っていた場合、 OS のバージョンアップ後に個人別学習ファイルを mkdic コマンドによって作成し直さなかった場合には、学習が行われません。

(5) **単語が登録できない**

- **単語登録用の辞書は指定されていますか？**

  HELP キーを押して、辞書操作・単語登録を選択した時点で ”単語登録用辞書が指定されていません” または、 ”ユーザ辞書が指定されていません ” と警告メッセージが出る場合は単語登録用の辞書が指定されていません。

  カスタマイズファイルを用いて単語登録用辞書を指定してください。単語登録用辞書の指定の詳細は **2.2 使用する辞書の指定** を参照してください。

- **書き込めない辞書を単語登録用辞書として指定していませんか？**

  この場合は単語登録の最後で辞書を指定した後 ”単語登録できませんでした ”というメッセージが出ます。この場合は原因が 2 つ考えられます。

1 つは、かな漢字変換サーバとして『かんな』の Version 3.2 より前のかな漢字変換サーバを指定している場合に、単語登録用辞書としてバイナリ辞書を指定していた場合です。『かんな』の Version 3.2 より前のかな漢字変換サーバでは、単語登録用辞書として使用できるのはテキスト形式辞書だけですので注意してください。

もう 1 つ考えられるのは登録の対象辞書として指定された辞書の書き込み権が cannaserver に対して解放されていない場合です。辞書のモードに関しては学習ができない場合と同様に対処してください。

- **単語登録用辞書に WRITE 権はついていますか？**

  『かんな』 Version 3.2 から辞書のセキュリティ強化のため、辞書の READ・WRITE 権が設定できるようになっています。辞書に WRITE 権がない場合には単語登録ができませんので、chmoddic(1) で辞書に WRITE 権をつけてください。辞書の READ・WRITE 権の詳細については、**3 かな漢字変換サーバ** を参照してください。 chmoddic(1) の詳細については、**4 かな漢字変換ユーティリティ** を参照してください。

(6) **部首変換が行えない**

- **部首辞書を指定していますか？**

  部首変換が行えないのは部首変換辞書が指定されていないのが原因であると思われます。カスタマイズファイルでキーワード :bushu で部首辞書が指定されているかどうか確認してください。

(7) **逐次自動変換できない (連文節変換になってしまう)。**

拡張機能の変換方式で、逐次自動変換に切り替えてください。それでも、逐次自動変換にならない場合は以下を参考にしてください。

- **カスタマイズで (setq auto nil) になっていませんか？**

  カスタマイズファイルを書き換えて、クライアントを再起動してください。

- **古いサーバに接続していませんか？**

  古いかな漢字変換サーバ (irohaserver) では逐次自動変換はできません。

(8) **個人別学習ができない。**

- **バイナリ辞書を更新したときに mkdic を行いましたか?**

  OS のバージョンアップなどを行ってバイナリ辞書が更新されたときには個人別学習も更新する必要があります。

  したがって、それまで個人別学習を行っていた場合、 OS のバージョンアップ後に個人別学習ファイルを mkdic コマンドによって作成し直さなかった場合には、学習が行われません。

- **個人別学習ファイルがありますか？**

  mkdic にて個人別学習ファイルを作成してください。

- **個人別学習ファイルに書き込み権がありますか？**

  個人別学習ファイルにかな漢字変換サーバの書き込み権を解放してください。

- **古いサーバに接続していませんか？**

  古いかな漢字変換サーバ (irohaserver) では個人別学習はできません。

(9) **キーの入力ができない**

- **マルチシーケンスで割り当ててあるキーを押しませんでしたか？**

  マルチシーケンスで割り当ててある (カスタマイズしている) キーを押してしまうとキーシーケンス入力状態になってしまいます。誤ってこのような状態になってしまったときは、 quit(取りやめ) を割りあててあるキーを押してください。そうすれば、キーシーケンス入力状態から抜け出て一般のキー入力ができるようになります。デフォルトでは quit は CTRL + g となっています。

(10) **ローマ字かな変換の ”ん ”の変換が今までとちがう**

『かんな』 Version 2.1 からローマ字かな変換辞書の ”ん ”の変換するルールが変更になりました。従来の変換どおりにしたい場合は、使用しているローマ字かな変換テーブルを変更してください。

デフォルトのローマ字かな変換辞書を使用している場合は、デフォルトのローマ字かな変換テーブルのテキスト形式のもの ”$(CANNALIBDIR)/sample/src/default.kpdef” を変更して、バイナリに変換したものを使用してください。

詳細は、 **2.3 ローマ字かな変換テーブルの設定** 、 **2.4 ローマ字かな変換テーブルのカスタマイズ** を参照してください。

6.4 カスタマイズに伴う症状

(1) **カスタマイズファイルに書いたことに効果がない**

- **エディットしているカスタマイズファイルが実際に用いられるカスタマイズファイルと同じですか？**

  どのカスタマイズファイルが使われるかは cannacheck コマンド −v オプションを用いて調べてください。

  ```
  % cannacheck -v
  カスタマイズファイルとして "/uhome/user/.canna"を用います。
  ……
  ```

  ここで表示されるファイルとエディットしているファイルが同じものであるかどうか調べてください。

- **バージョンの違うカスタマイズファイルを使用していませんか？**

  カスタマイズファイルは『かんな』 Version 1.2 までは .iroha というファイルでしたが、『かんな』 Version 2.1 からは、.canna というファイルに変更になりました。

  使用している日本語入力システムのバージョンに応じたカスタマイズファイルを使用してください。

- **カスタマイズで設定した項目がカスタマイズファイルの後ろの方で打ち消されていませんか？**

  カスタマイズの効果が表れない別の原因としてカスタマイズファイルの後ろの方で設定を打ち消している場合があります。たとえば以下の記述を見てください。

  ```
  (set-key 'yomi-mode "\C-1" henkan)
    ・・・・
  (global-set-key "\C-1" forward)
  ```

このような場合、読み入力モードで設定した C−1 が後ろの方の設定で打ち消されてしまい結局 C−1 は読み入力モードでも右移動キーとして使われることになります。

すべてのモードに影響するキーのカスタマイズは最初の方に記述し、特定のモードに限定したキーのカスタマイズは後ろの方に記述するのが良いでしょう。

- **カスタマイズで設定したキーがアプリケーションなどで用いられていませんか？**

最後にアプリケーションによりキーが奪われてしまう例を示します。

たとえばカスタマイズファイルに次のような記述を入れたとします。

```
(global-set-key "\C-s" next)
(global-set-key "\C-q" previous)
```

これを cannad で使おうとしても TTY ドライバが先に C−s、 C−q を XON/XOFF のための文字と解釈し使ってしまうため cannad 内部のかな漢字変換部分にまでこれらのキーが伝わりません。

他の例では、アプリケーションによって HELP キーや INS キーに特別の処理を割り当てている場合にもかな漢字変換ではこれらのキーが使われないことになりますので注意してください。

(2) **ローマ字かな変換テーブルの効果がない**

- **ローマ字かな変換テーブルはバイナリ形式で与えていますか？**

ローマ字かな変換テーブルをエディットした後で mkromdic(1) によりローマ字かな変換テーブルをバイナリ形式に変換していますか？ローマ字かな変換テーブルはバイナリ形式のもののみ有効ですので注意してください。

- **エディットしているテーブルと実際に使われるテーブルが同じですか？**

エディットしているローマ字かな変換テーブルと実際に使われるローマ字かな変換テーブルが一致していない場合が考えられます。 cannacheck −v によりどのローマ字かな変換テーブルが使われるかを確認してください。

使用している日本語入力システムが『かんな』 Version 1.2 以前の場合は、旧形式のローマ字かな変換テーブルしか使用できません。

(3) **ユーザ辞書がマウントできない**

- **ユーザ辞書ディレクトリが存在しますか？**

ユーザ辞書は接続しているサーバが動作しているホストのユーザ辞書ディレクトリに存在しなければなりません。次のことをチェックしてください。

(a) cannacheck −v にて、自分がどのホストで動いているサーバを使っているのかをチェックしてください。

(b) cannastat にて、自分が接続しているサーバのバージョンをチェックしてください。

(c) そのホストにユーザ辞書ディレクトリが存在することを確認してください。ユーザ辞書ディレクトリは、『かんな』 Version 3.2 より前のバージョンでは、 ”$(CANNALIBDIR)/dic/ユーザ名 ” ですが、 Version 3.2 からは ”$(CANNALIBDIR)/dic/user/ ユーザ名 ” になっています。

(d) ユーザ辞書ディレクトリ配下の dics.dir に指定した辞書名が記述されていますか？

(4) グループ辞書がマウントできない

- **『かんな』のバージョンが 3.2 以上ですか？**

  グループ辞書は、『かんな』の Version 3.2 から使用できます。サーバ・クライアントともに『かんな』の Version 3.2 以上でなければなりません。

- **グループ辞書ディレクトリが存在しますか？**

  グループ辞書は接続しているサーバが動作しているホストのグループ辞書ディレクトリに存在しなければなりません。次のことをチェックしてください。

  (a) cannacheck −v にて、自分がどのホストで動いているサーバを使っているのかをチェックしてください。

  (b) そのホストにグループ辞書ディレクトリが存在することを確認してください。グループ辞書ディレクトリは、”$(CANNALIBDIR)/dic/group/ グループ名” になっています。

  (c) グループ辞書ディレクトリ配下の dics.dir に指定した辞書名が記述されていますか？

6.5 システム管理関係の症状

(1) **cannaserver が起動できない**

- **他の cannaserver が動いていませんか？**

  cannaserver は重複した起動をすることができません。起動させようとしている cannaserver が すでに起動していないか確認してください。

- **/tmp/.iroha_unix/IROHA ファイルがありませんか？**

  cannaserver は UNIX ドメインの通信用に、 /tmp/.iroha_unix/IROHA という特殊ファイルを作成しますが、なんらかの障害により cannaserver が終了しても特殊ファイルが残ったままになることがあります。

  特殊ファイルが残ったままだと、次にサーバを起動した時に特殊ファイルの作成に失敗してサーバが起動できませんので、特殊ファイルを削除してから cannaserver を再起動してください。

  特殊ファイルを削除する際に、誤って他の特殊ファイルを削除しますと、その特殊ファイルを使用していたサーバは、 UNIX ドメインで通信が行えなくなりますので、 ps コマンドで cannaserver の起動状態を十分に確認の上、起動していない cannaserver の特殊ファイルを削除して下さい。

- **cannaserver に正当なオーナ、グループ、パーミッションが与えてありますか？**

  cannaserver のオーナ、グループは **bin** です。

  cannaserver のパーミッションは `-r-sr-sr-x` でなくてはいけません。シンボリックリンクされている場合は実体を調べてください。

  なお、それぞれのディレクトリのパーミッションも不正の場合、うまく動作しないことがあります。

付録 A

デフォルトでの基本的なキーの割り当て

キー	かな漢字変換での機能
XFER	アルファベット入力モードと日本語入力モードとの相互切り替え、変換、次候補
Space	変換、次候補
RETURN, CTRL−m	確定
BS, CTRL−h	カーソル左文字削除、漢字を読みに戻す
INS	記号入力
HELP	拡張機能
CTRL−a	読みの左端へ移動、左端候補へ移動、左端文節選択
←, CTRL−b	カーソル左移動、左候補へ移動、左文節選択
CTRL−c	文節編集
CTRL−d	カーソル右文字削除
CTRL−e	読みの右端へ移動、右端候補へ移動、右端文節選択
→, CTRL−f	カーソル右移動、右候補へ移動、右文節選択
CTRL−g	取りやめ
CTRL−i	文節縮め
CTRL−k	カーソル右文字列削除、部分確定
CTRL−l	小文字変換
↓, CTRL−n	次文字種、次候補表示、次の候補一覧表示
CTRL−o	アルファベット入力モードと日本語入力モードとの相互切り替え、文節伸ばし
↑, CTRL−p	前文字種、前候補表示、前の候補一覧表示
CTRL−q	引用入力
CTRL−u	大文字変換
CTRL−w	候補一覧、部首一覧
CTRL−y	16 進コード変換

付録 B

デフォルトでの機能一覧表

アルファベット入力状態における機能

機　　能	キー操作	備　　考
日本語入力モードへの移行	XFER, C−o	

読み入力がない状態における機能

機　　能	キー操作	備　　考
アルファベットモードへの移行	XFER, C−o	
読みの入力	a, b, c … 0, 1 …	アルファベットや記号を入力することにより読みが入力された状態へ移行する。
記号入力モードへの移行	INS	
引用文字列の挿入	C−q	引用文字列の挿入を行う。引用文字列の挿入を開始するための文字の次の 1 文字をローマ字かな変換せず入力する。
拡張	HELP	拡張モードに入る。

読みを入力している状態における機能

機　　能	キー操作	備　　考
読みの削除	BS, C−h(左方向)	左端での左方向削除および右端での右方向削除は無効である。削除した結果読みが空になった場合には、読みがない状態に戻る。
	C−d(右方向)	
	C−k(カーソル以降)	

機　　能	キー操作	備　　考
カーソルの移動	C−a(左端)	
	C−e(右端)	
	←, C−b(左へ)	左端から左へ移動しようとすると右端へ移動する。
	→, C−f(右へ)	右端から右へ移動しようとすると左端に移動する。
読みを漢字に変換	Space, XFER	単候補表示状態に移る。
読みをそのまま確定	RETURN, NFER, C−m	読みを確定し、読みがない状態に戻る。
読みの字種変換	↓, C−n(順変換)	字種変換状態に移る。
	↑, C−p(逆変換)	
大文字小文字変換	C−u(大文字)	入力をローマ字に変換しさらに、大文字または小文字に変換する。
	C−l(小文字)	
読みの取り消し	C−g	読みを取り消し、読みがない状態に戻る。
引用文字列の挿入	C−q	引用文字列の挿入を行う。引用文字列の挿入を開始するための文字の次の 1 文字をローマ字かな変換せず入力する。
読みを 16 進コードとみなして変換	C−y	
読みを部首名とみなして変換	C-w	
マークをつける	C-Space,C-@	

単候補表示状態における機能

機　　能	キー操作	備　　考
次の読みの入力	a, b, c, … 0, 1 …	候補を確定し次の読みを入力する。読みを入力している状態となる。
確　　定	RETURN, NFER, C−m	候補を確定し、読みのない状態に戻る。

機　　能	キー操作	備　　考
部分確定	C−k	カーソル部分より左の文節を確定し、カーソルより後の部分は読みに戻る。
無変換状態への移行	BS, C−g, C−h	読みを入力している状態に戻る。
文節間の移動	C−a(左端)	
	C−e(右端)	
	←, C−b(左へ)	左端から左へ移動しようとすると右端へ移動する。
	→, C−f(右へ)	右端から右へ移動しようとすると左端に移動する。
文節の伸縮	TAB, C−i(文節縮め)	文節は 1 文字長以下には縮まない。
	C−o(文節伸ばし)	その文節より後の文節が空になった場合はそれ以上文節伸ばし操作は行えない。
次候補 / 前候補の表示	Space, XFER ↓, C−n(次候補)	最後の候補時に次候補を表示しようとすると、一番最初の候補に移動する。
	↑, C−p(前候補)	最初の候補時に前候補を表示しようとすると、一番最後の候補に移動する。
候補一覧表示	C−w	候補一覧表示状態へ移行する。
文節の編集	C−c	カレント文節を読みに戻す。
大文字小文字変換	C−u(大文字)	カレント文節をローマ字に変換し、さらに大文字または小文字に変換する。
	C−l(小文字)	

候補一覧表示状態における機能

機　　能	キー操作	備　　考
選　　択	RETURN, NFER, C−m	候補を選択して、単候補表示状態に移る。
次候補 / 前候補への移動	C−a(左端)	
	C−e(右端)	

機　　能	キー操作	備　　考
次候補 / 前候補への移動	←, C−b(前候補へ)	候補列の左端で前候補へ移動すると前候補列の右端に移動する。ただし、最初の候補列であった場合には、最後の候補列の左端に移動する。
	Space, XFER →, C−f(次候補へ)	候補列の右端で次候補へ移動すると次候補列の左端に移動する。ただし、最後の候補列であった場合には、最初の候補列の左端に移動する。
番号による移動	1 〜 9	対応する番号の候補に移動する。
次候補列 / 前候補列の表示	↓, C−n(次候補列)	最後の候補列で次候補列へ移動すると最初の候補列の同じ番号の候補に移動する。
	↑, C−p(前候補列)	最初の候補列で前候補列へ移動すると最後の候補列の同じ番号の候補に移動する。
候補一覧表示の取り消し	BS, C−g,C−h	単候補状態に戻る。

字種変換状態における機能

機　　能	キー操作	備　　考
次の読みの入力	a, b, c, … 0, 1 …	字種変換を確定し、次の読みを入力する。
確　　定	RETURN, NFER, C−m	字種変換を確定し、読みのない状態に戻る。
読みの字種変換	↓, C−n(順変換)	次の字種がひらがなで字種変換領域と未確定文字列が一致しているときは、字種変換モードから抜けて、読みを入力している状態になる。
	↑, C−p(逆変換)	
字種変換の取り消し	BS, C−g	字種変換をとりやめ、読みを入力している状態に戻る。

機　　能	キー操作	備　　考
字種変換領域の伸縮	C−o(領域伸ばし)	読み全体が字種変換の対象であるときに、領域伸ばしを行うと、左端の 1 文字のみが対象領域となる。
	TAB, C−i(領域縮め)	領域が 1 文字長の時に領域縮めを行うと、読み全体が字種変換の対象領域となる。
大文字 / 小文字変換	C−u(大文字変換)	
	C−l(小文字変換)	
漢字への変換	Space, XFER	

記号入力モードにおける機能

機　　能	キー操作	備　　考
確　　定	RETURN, NFER	候補を確定する。
次候補 / 前候補へ の移動	C−a(左端)	
	C−e(右端)	
	←, C−b(前候補へ)	候補列の左端で前候補へ移動すると前候補列の右端に移動する。ただし、最初の候補列であった場合には、最後の候補列の右端に移動する。
	Space, XFER →, C−f(次候補へ)	候補列の右端で次候補へ移動すると 次候補列の左端に移動する。ただし、最後の候補列であった場合には、最初の候補列の左端に移動する。
次候補列 / 前候補 列の表示	↓, C−n(次候補列)	最後の候補列で次候補列へ移動する と最初の候補列の同じ番号の候補に移動する。
	↑, C−p(前候補列)	最初の候補列で前候補列へ移動すると最後の候補列の同じ番号の候補に移動する。

機　　能	キー操作	備　　考
記号入力の取り消 し	BS, C−g	読みのない状態に戻る。 ただし、拡張モードから記号入力モードに入った場合、記号を 1 つも確定していないときは、1 つ前の画面に戻る。

付録 C

カスタマイズに用いる機能名一覧表

名　　前	デフォルト	機　　能
japanese-mode	Xfer, C−o	日本語モードに移行する
alpha-mode	Xfer, C−o	アルファベットモードに移行する
quoted-insert	C−q	次の一文字を無条件に入力する
henkan-nyuuryoku-mode	−	変換入力モードに移行する
self-insert	a 〜 z, A 〜 Z, 記号等....	その文字を入力する
henkan-or-self-insert	−	連文節変換時、ベースがひらがな以外のとき、その文字を入力する
henkan-or-do-nothing	−	逐次自動変換時、ベースがひらがな以外のとき、なにも行わない
hex-mode	−	コード入力モードに移行する
bushu-mode	−	部首入力モードに移行する
kigou-mode	Insert	記号入力モードに移行する
russian-mode	−	ロシア文字入力モードに移行する
greek-mode	−	ギリシャ文字入力モードに移行する
line-mode	−	罫線入力モードに移行する
chikuji-mode	−	逐次自動変換モードに移行する
renbun-mode	−	連文節変換モードに移行する

名　　前	デフォルト	機　　能
undefined	−	そのキーに割り当てられている機能を未定義に戻す
temporary	−	現在のモードをセーブし、次のモードの準備をする
forward	→, C−f	右移動
backward	←, C−b	左移動
next	↓, C−n	下移動
previous	↑, C−p	上移動
beginning-of-line	C−a	左端へ移動
end-of-line	C−e	右端へ移動
delete-next	C−d	次一文字消去
delete-previous	Backspace, C−h	前一文字消去
kill-to-end-of-line	C−k	行末まで消去
henkan	Space, Xfer	変換
kakutei	Nfer, Return, C−m, C−j	確定
extend	S− →, C−o	領域伸ばし
shrink	S− ←, C−i	領域縮め
shinshuku-mode	−	文節伸縮モードに移行する
quit	C−g	とりやめ
extend-mode	Help	メニュー
touroku-mode	−	単語登録
delete-dic-mode	−	単語削除
jisho-ichiran	−	辞書のマウント / アンマウント情報を表示する
sync-dictionary	−	辞書の書き込みを行う
show-sever-name	−	サーバを表示する
switch-server	−	サーバを切り替える
disconnect-server	−	サーバとの接続を切る
show-gakushu	−	学習状態を表示する

名　　前	デフォルト	機　　能
show-canna-version	–	かんなのバージョンを表示する
show-romkana-table	–	ローマ字かな変換テーブルを表示する
show-canna-file	–	カスタマイズファイルを表示する
convert-as-hex	C−y	16 進数とみなして変換
convert-as-bushu	C−w	部首名とみなして変換
kouho-ichiran	C−w	候補一覧表示
zenkaku	–	全角変換
hankaku	–	半角変換
to-upper	C−u	大文字に変換
capitalize	–	先頭だけを大文字に変換
to-lower	C–l	小文字に変換
hiragana	–	ひらがな変換
katakana	–	カタカナ変換
romaji	–	ローマ字変換
base-hiragana	–	入力ベースをひらがなにする
base-katakana	–	入力ベースをカタカナにする
base-kana	–	入力ベースをカナにする
base-eisu	–	入力ベースを英数にする
base-zenkaku	–	入力ベースを全角にする
base-hankaku	–	入力ベースを半角にする
base-kakutei	–	入力モードを確定モードにする
base-henkan	–	入力モードを変換モードにする
base-hiragana-katakana-toggle	–	入力ベースをひらがなとカタカナでトグルする

名　　前	デフォルト	機　　能
base-zenkaku-hankaku-toggle	–	入力ベースを全角と半角でトグルする
base-kana-eisu-toggle	–	入力ベースをカナと英数でトグルする
base-kakutei-henkan-toggle	–	入力モードを確定と変換でトグルする
base-rotate-forward	–	入力ベースを順番に切替える
base-rotate-backward	–	入力ベースを逆順に切替える
henshu	C-c	カレント文節を読みに戻す
mark	C-Space, C-@	マーク

付録 D

ローマ字かな変換表

	a	i	u	e	o
	ア	イ	ウ	エ	オ
k	カ	キ	ク	ケ	コ
s	サ	シ	ス	セ	ソ
t	タ	チ	ツ	テ	ト
n	ナ	ニ	ヌ	ネ	ノ
h	ハ	ヒ	フ	ヘ	ホ
m	マ	ミ	ム	メ	モ
y	ヤ	イ	ユ	イェ	ヨ
r	ラ	リ	ル	レ	ロ
w	ワ	ヰ	ウ	ヱ	ヲ
g	ガ	ギ	グ	ゲ	ゴ
z	ザ	ジ	ズ	ゼ	ゾ
d	ダ	ヂ	ヅ	デ	ド
b	バ	ビ	ブ	ベ	ボ
p	パ	ピ	プ	ペ	ポ

	a	i	u	e	o
ky	キャ	キィ	キュ	キェ	キョ
gy	ギャ	ギィ	ギュ	ギェ	ギョ
sy	シャ	シィ	シュ	シェ	ショ
zy	ジャ	ジィ	ジュ	ジェ	ジョ
ty	チャ	チィ	チュ	チェ	チョ
ny	ニャ	ニィ	ニュ	ニェ	ニョ
hy	ヒャ	ヒィ	ヒュ	ヒェ	ヒョ
by	ビャ	ビィ	ビュ	ビェ	ビョ
py	ピャ	ピィ	ピュ	ピェ	ピョ
my	ミャ	ミィ	ミュ	ミェ	ミョ
ry	リャ	リィ	リュ	リェ	リョ
ts	ツァ	ツィ	ツ	ツェ	ツォ
sh	シャ	シ	シュ	シェ	ショ
th	テャ	ティ	テュ	テェ	テョ
j	ジャ	ジ	ジュ	ジェ	ジョ
ch	チャ	チ	チュ	チェ	チョ
f	ファ	フィ	フ	フェ	フォ
v	ヴァ	ヴィ	ヴ	ヴェ	ヴォ
gw	グァ	グィ	グゥ	グェ	グォ
dy	ヂャ	ヂィ	ヂュ	ヂェ	ヂョ
dh	デャ	ディ	デュ	デェ	デョ
c	カ		ク		コ
cy	チャ	チィ	チュ	チェ	チョ
jy	ジャ	ジィ	ジュ	ジェ	ジョ
l	ラ	リ	ル	レ	ロ
ly	リャ	リィ	リュ	リェ	リョ
x	ァ	ィ	ゥ	ェ	ォ
xw	ヮ				
xt			ッ		
xts			ッ		
xy	ャ		ュ		ョ

n	ん
nn	ん
mn	ん
n’	ん
<space>	半角スペース (0x20)
@@	全角スペース (0x2121)
数字	1、2、3　　(全角)
!	！
"	”
#	＃
$	＄
%	％
&	＆
’	’
(	（　　(半角)
)	）　　(半角)
+	＋
-	ー
*	＊
=	＝
^	＾
‘	｀
\	¥
[	「
]	」
{	｛
}	｝
;	；
:	：

|	｜
@	＠
~	￣
,	、
<	＜
.	。
>	＞
/	／
?	？
_	＿
@	＠
@(	（
@)	）
@{	｛
@}	｝
@[	［
@]	］
@,	，
@.	．
@~	～
@\	＼
@/	・
@-	－
@2	‥
@3	…
@||	‖
@|	｜

付録 E

カスタマイズファイルの例

以下にカスタマイズファイルの例として unix.canna および just.canna を示します。それぞれのファイルは $(CANNALIBDIR)/sample の下に存在しますので実際に環境変数 CANNAFILE で指定するなどして試してみてください。

E.1 unix.cannaファイル

根っからの UNIX ユーザには default.canna よりも unix.canna の方が馴染みやすいと思われます。数字などのデフォルトを半角文字にしています。

```
;; 記号置き換え変換を利用するか。

(setq use-symbolic-definition t)

;;; ローマ字かな変換の設定
;
; ローマ字かな変換テーブルを指定します。ローマ字かな変換テーブルは
; (1) カレントディレクトリ
; (2) ホームディレクトリ
; (3) /usr/lib/canna/dic
; の順にサーチされます。
;

;(setq romkana-table "unix.kp")
(setq romkana-table "default.kp")

;; 外来語変換
;(setq english-table "canna/english.kp")


;;; 利用する辞書
;
; ただ単に書き並べているのは単語登録を行わない辞書でシステム辞書などが
; これに当たります。
```

```
;
; 部首辞書に関しては :bushu を先行させて記述します。
;
; 単語登録をしたい辞書に関しては :user を先行させて記述します。単語登
; 録をする辞書はテキスト形式辞書でなければなりません。単語登録させたい
; 辞書は mkdic user のようなコマンドを打つことでで簡単に作成することがで
; きます。
;
(use-dictionary
 "iroha"
 "fuzokugo"
 "hojomwd"
 "hojoswd"
 :bushu "bushu"
 :user  "user"
 )

;;; さまざまなパラメタの設定
;
; 真理値を設定するものについては真については t 、偽については nil を設
; 定します。

; 一番右で右に行こうとすると左端にいくようにするか
(setq cursor-wrap          t) ; default t

; 一覧表示で数字を打ったときにカーソル移動に留める (nil) か選択までして
; しまうか (t)
(setq select-direct        t) ; default t

; 一覧表示で数字で選択するか (t)。 nil の時は数字が表示されません
(setq numerical-key-select t) ; default t

; 候補表示時に文節の区切りに空白をいれるか
(setq bunsetsu-kugiri nil)      ; default nil

; 文字ベースの移動をするか (t)。 nil の時はローマ字かな変換の単位でカー
; ソル移動が行われます。
(setq character-based-move t) ; default t

; 入力中の読み全体をリバースするか
(setq reverse-widely       nil) ; default nil

; 頻度学習をするか
(setq gakushu              t) ; default t

; 一覧の最後までいったらいったん一覧表示をやめるか
(setq quit-if-end-of-ichiran nil); default nil
```

```
; 文節の右端で右移動をすると全体が確定してしまうか
(setq kakutei-if-end-of-bunsetsu nil) ; default nil

; 一覧選択後同じ文節にとどまるか (t)。 nil の時は一覧選択を行うと 1 つ先
; の文節に移動します。
(setq stay-after-validate  t) ; default t

; 読みを入力中にバックスペースを打つとローマ字までバラバラにするか
(setq break-into-roman     t) ; default nil

; 何回の変換キーで一覧表示を行うか。0 を入れると変換キーだけでは
; 一覧表示には移りません。その場合には C-w を入力して一覧表示して
; ください。
(setq n-henkan-for-ichiran 2) ; default 2

; 単語登録時に文法関連の質問をするか (t)。 nil の時は適当な品詞を付けてし
; まいます。
(setq grammatical-question  t) ; default t

; 候補が全体の何番目かを表示するか
(setq kouho-count t)            ; default t

; 逐次変換をするか
(setq auto nil)                 ; default nil

; 逐次変換をする時に何個の文節までを未確定で保持するか
(setq n-kouho-bunsetsu 16)      ; default 16

; ローマ字かな変換でイリーガルなローマ字を捨てるか
(setq abandon-illegal-phonogram nil)              ; default nil

; 一覧の時も次の入力を認めるか
(setq allow-next-input   t) ; default nil

; 一文字レベルのローマ字かな変換の補助テーブル
(if use-symbolic-definition
    (progn
      (defsymbol ?-  "ー" "-" "—" "‐" )
      (defsymbol ?/  "/"  "／"  "・")
      (defsymbol ?\\ "\\" "＼"  "¥")
      (defsymbol
?( "(" "（" "【"
?) ")" "）" "】" )
      (defsymbol
?[ "「" "『" "［" "〔" "["
?] "」" "』" "］" "〕" "]")
```

```
      (defsymbol
?. "。" ". " "."
?, "、" ", " ","  )
      (defsymbol
?0 "0" "０" "〇" "零"
?1 "1" "１" "一" "壱"
?2 "2" "２" "二" "弐"
?3 "3" "３" "三" "参"
?4 "4" "４" "四" "四"
?5 "5" "５" "五" "伍"
?6 "6" "６" "六" "六"
?7 "7" "７" "七" "七"
?8 "8" "８" "八" "八"
?9 "9" "９" "九" "九" )
      (defsymbol
?? "?" "？"
?! "!" "！")
      (defsymbol ?# "#" "＃")
      (defsymbol ?$ "$" "＄")
      (defsymbol ?% "%" "％")
      (defsymbol ?& "&" "＆")
      (defsymbol ?+ "+" "＋")
      (defsymbol ?* "*" "＊" "×")
      (defsymbol ?_ "_" "＿")
      (defsymbol ?’ "’" "’")
      (defsymbol ?‘ "‘" "‘")
      (defsymbol ?\" "\"" "”")
      ))

;; delete キーもバックスペースと同じ処理をさせる
(global-set-key "\Delete" 'delete-previous) ; necessary in using Emacs

;; Help がない人もいるので F1 でも同じ動作をさせる。
(global-set-key "\F1" 'extend-mode)
```

E.2 just.cannaファイル

一太郎[1]における日本語入力操作をシミュレートするカスタマイズファイルです。ローマ字かな変換テーブルに専用のテーブルを使っています。また、主に以下のような操作法を提供しています。

キー	機能
Space	変換
Return	確定
C-u	ひらがな変換
C-i	カタカナ変換
C-o	半角変換
C-p	ローマ字
C-n	右文節移動
→、C-l	右移動、文節伸ばし
←、C-k	左移動、文節縮め
↓	確定

;; **一太郎での入力の操作方法をシミュレートするためのカスタマイズファイル**

```
;(setq initial-mode 'han-alpha-kakutei-mode)

(setq romkana-table "just.kp")

(use-dictionary
 "iroha"
 "fuzokugo"
 "hojomwd"
 "hojoswd"
 :bushu "bushu"
 :user "user"
 )

(setq cursor-wrap            nil) ; default on
(setq select-direct          t)   ; default on
(setq numerical-key-select   t)   ; default on
(setq character-based-move   nil) ; default on
(setq reverse-widely         nil) ; default off
(setq break-into-roman       nil) ; default off
(setq stay-after-validate    nil) ; default on
(setq quit-if-end-of-ichiran t)   ; default off
(setq kakutei-if-end-of-bunsetsu t) ; default off
(setq n-henkan-for-ichiran   2)   ; default 2
(setq inhibit-list-callback  t)   ; default off
(setq kouho-count            t)   ; default off
```

[1] 一太郎はジャストシステム (株) の商標です

```
;;  setting for keymapping

(global-unbind-key-function 'quoted-insert)
(global-unbind-key-function 'extend)
(global-unbind-key-function 'shrink)
(global-unbind-key-function 'quoted-insert)
(global-unbind-key-function 'forward)
(global-unbind-key-function 'backward)
(global-unbind-key-function 'previous)
(global-unbind-key-function 'next)
(global-unbind-key-function 'beginning-of-line)
(global-unbind-key-function 'end-of-line)
(global-unbind-key-function 'kill-to-end-of-line)

;(global-unbind-key-function 'alpha-mode)
;(global-unbind-key-function 'japanese-mode)
;(global-unbind-key-function 'henkan)
;(global-unbind-key-function 'quit)
;(global-unbind-key-function 'deleteNext)

(global-set-key "\C-l"      'extend)
(global-set-key "\C-k"      'shrink)
(global-set-key "\Right"    'forward)
(global-set-key "\Left"     'backward)
(global-set-key "\Up"       'previous)
(global-set-key "\Down"     'next)
(global-set-key "\C-["      'quit)

(set-key 'empty-mode "\F10"       'kigo-mode)
(set-key 'kigo-mode  "\F10"       'henkan-nyuuryoku-mode)

(let ((mode 'yomi-mode))
  (set-key mode "\C-u" 'hiragana)
  (set-key mode "\F6"  'hiragana)
  (set-key mode "\C-i" 'katakana)
  (set-key mode "\F7"  'katakana)
  (set-key mode "\C-o" 'hankaku)
  (set-key mode "\F8"  'hankaku)
  (set-key mode "\C-p" 'romaji)
  (set-key mode "\F9"  'romaji)
  (set-key mode "\C-l" 'forward)
  (set-key mode "\Right" 'forward)
  (set-key mode "\C-k"   'backward)
  (set-key mode "\Left" 'backward)
  (set-key mode "\C-n"  'kakutei)
  (set-key mode "\Down" 'kakutei)
  )
```

```
(let ((mode 'tan-kouho-mode))
  (set-key mode "\C-u" 'hiragana)
  (set-key mode "\F6"  'hiragana)
  (set-key mode "\C-i" 'katakana)
  (set-key mode "\F7"  'katakana)
  (set-key mode "\C-o" 'hankaku)
  (set-key mode "\F8"  'hankaku)
  (set-key mode "\C-p" 'romaji)
  (set-key mode "\F9"  'romaji)
  (set-key mode "\C-n" 'forward)
  (set-key mode "\Down" 'forward)
  (set-key mode "\C-l" '(shinshuku-mode extend))
  (set-key mode "\C-k" '(shinshuku-mode shrink))
  (set-key mode "\Right" '(shinshuku-mode extend))
  (set-key mode "\Left"  '(shinshuku-mode shrink))
  (set-key mode "\Up"  'previous)
  )

(let ((mode 'mojishu-mode))
  (set-key mode "\C-u" 'hiragana)
  (set-key mode "\F6"  'hiragana)
  (set-key mode "\C-i" 'katakana)
  (set-key mode "\F7"  'katakana)
  (set-key mode "\C-o" 'hankaku)
  (set-key mode "\F8"  'hankaku)
  (set-key mode "\C-p" 'romaji)
  (set-key mode "\F9"  'romaji)
  (set-key mode "\C-l" 'extend)
  (set-key mode "\Right" 'extend)
  (set-key mode "\C-k"  'shrink)
  (set-key mode "\Left" 'shrink)
  (set-key mode "\C-n"  'kakutei)
  (set-key mode "\Down" 'kakutei)
  )

(let ((mode 'ichiran-mode))
  (set-key mode "\Xfer"  'next)
  (set-key mode "\C-l"   'forward)
  (set-key mode "\Right"  'forward)
  (set-key mode "\C-k"   'backward)
  (set-key mode "\Left"  'backward)
  (set-key mode "\C-n"   'next)
  (set-key mode "\Down"  'next)
  (set-key mode "\Up"    'previous)
;;  Please put off comment character below
;;  if you do not use Extend mode well.
```

```
;  (set-key mode "\C-n"   'kakutei)
;  (set-key mode "\Right" '(quit shinshuku-mode extend))
;  (set-key mode "\C-l"   '(quit shinshuku-mode extend))
;  (set-key mode "\Left"  '(quit shinshuku-mode shrink))
;  (set-key mode "\C-k"   '(quit shinshuku-mode shrink))
;  (set-key mode "\Up"    'backward)
  )
(let ((mode 'shinshuku-mode))
  (set-key mode "\C-u" '(henkan hiragana))
  (set-key mode "\F6"  '(henkan hiragana))
  (set-key mode "\C-i" '(henkan katakana))
  (set-key mode "\F7"  '(henkan katakana))
  (set-key mode "\C-o" '(henkan hankaku))
  (set-key mode "\F8"  '(henkan hankaku))
  (set-key mode "\C-p" '(henkan romaji))
  (set-key mode "\F9"  '(henkan romaji))
  )
```

付録 F

部首入力の部首名一覧表

画数	部首		部首キー
1	一	(いち)	いち
	ノ	(のかんむり)	の
2	口	(かんにょう)	うけばこ
		(うけばこ)	
	十	(じゅう)	じゅう
	卩	(ふしづくり)	せつ、ふし
	刀 (刈)	(りっとう)	かたな、りっとう
	力	(ちから)	か、ちから
	厂	(かんだれ)	がん
	勹	(つつみがまえ)	く、つつみ
	冂匚凵	(どうがまえ)	かまえ
		(はこがまえ)	
		(くにがまえ)	
	亠	(なべぶた)	なべ
	冫	(にすい)	に、ん
	人 (仁)	(にんべん)	にん、ひと
	又	(また)	ぬ、また
	几	(きにょう)	つくえ
	八	(はちがしら)	はち
	儿	(ひとあし)	ひとあし、る
	冖	(わかんむり)	わ
3	宀	(うかんむり)	う
	廴	(えんにょう)	えん
	郁	(おおざと)	おおざと
	己	(おのれ)	おのれ
	女	(おんな)	おんな
	彳	(ぎょうにんべん)	ぎょう

画数	部首		部首キー
3	口	(くちへん)	くち、ろ
	草	(くさかんむり)	くさ
	独	(けものへん)	けもの
	子	(こへん)	こ
	隋	(こざと)	こざと
	士	(さむらい)	さむらい
	江	(さんずい)	さん、し
	弋	(しきがまえ)	しき
	尸	(しかばね)	しゃく
	小／単	(しょう、つ)	しょう、つ
	辶	(しんにゅう)	しん
	寸	(すん)	すん
	大	(だい)	だい
	土	(つちへん)	つち、ど
	手	(てへん)	て
	巾	(はば)	はば
	广	(まだれ)	ま
	山	(やまへん)	やま
	夕	(ゆうべ)	ゆう
	弓	(ゆみへん)	ゆみ
	忙	(りっしんべん)	りっしん
4	欠	(あくび)	あくび、けつ
	歹	(がつへん)	いちた
	犬	(いぬ)	いぬ
	牛／牡	(うしへん)	うし
	片	(かたへん)	かた
	木	(きへん)	き
	気	(きがまえ)	きがまえ
	毛	(け)	け
	心	(したごころ)	こころ
	水	(みず)	すい、みず

画数	部首		部首キー
4	月	(つきへん)	つき
	爪	(つめかんむり)	つめ
	日	(ひへん)	ひ
	夕	(のぶん)	のぶん
	火	(ひ)	ひ
	方	(かたへん)	ほう
	戈	(ほこがまえ)	ほこ
	点	(れっか)	よつてん
	殳	(るまた)	るまた
5	穴		あな
	石	(いしへん)	いし
	玉	(たまへん)	おう、たま
	皮	(ひのかわ)	かわ
	瓦	(かわら)	かわら
	皿	(さら)	さら
	示／神	(しめすへん)	しめす、ね
	白	(しろ)	しろ
	田	(たへん)	た
	立	(たつへん)	たつ、りつ
	禾	(のぎへん)	のぎ
	目	(めへん)	め
	癶	(はつがしら)	はつ
	矢	(やへん)	や
	疔	(やまいだれ)	やまい
	四	(あみがしら)	よん
6	糸	(いとへん)	いと
	臼	(うす)	うす
	瓜		うり
	老	(おいかんむり)	おい、ろう
	缶	(ほどぎ)	かん
	衣／初	(ころもへん)	きぬ、ころも

画数	部首		部首キー
6	米	(こめへん)	こめ
	舌	(した)	した
	耒	(すきへん)	すき
	竹	(たけかんむり)	たけ
	血	(ち)	ち
	虎	(とらかんむり)	とら
	肉	(にく)	にく
	西	(か)	にし
	羽	(はね)	はね
	羊	(ひつじへん)	ひつじ
	聿	(ふでづくり)	ふで
	舟	(ふねへん)	ふね
	耳	(みみへん)	みみ
	虫	(むしへん)	むし
7	赤	(あかへん)	あか
	足 (疋)	(あしへん)	あし
	豕	(いのこへん)	いのこ
	臣		おみ
	貝	(かいへん)	かい
	辛	(からい)	からい
	車	(くるまへん)	くるま
	見	(みる)	けん、みる
	言	(ごんべん)	ごん
	酉	(さけづくり)	さけ、ひよみ
		(ひよみのとり)	
	走	(そうにょう)	そう、はしる
	谷	(たにへん)	たに
	角	(つのへん)	つの
	釆	(のごめへん)	のごめ
	麦	(ばくにょう)	ばく、むぎ
	豆	(まめへん)	まめ

画数	部首		部首キー
7	身	(みへん)	み
	豸	(むじな)	むじな
8	雨	(あめかんむり)	あめ
	非	(あらず)	あらず
	金	(かねへん)	かね
	門	(もんがまえ)	もん
	隹	(ふるとり)	ふるとり
9	頁	(おおがい)	おおがい、ページ
	音	(おと)	おと
	香	(かおり)	かおり、こう
	革	(かわへん)	かく
	風	(かぜ)	かぜ
	首	(くび)	くび
	食	(しょくへん)	しょく
	韋	(なめしがわ)	なめし
	面	(めん)	めん
10	馬	(うまへん)	うま
	鬼	(きにょう)	おに
	髟	(かみがしら)	かみ
	高	(たかい)	たかい
	鬥	(とうがまえ)	とう
	骨	(ほね)	ほね
11	魚	(うおへん)	うお
	亀		かめ
	鳥	(とりへん)	とり
	黒	(くろ)	くろ
12	鹿		しか
14	鼻	(はなへん)	はな
15	歯	(はへん)	は
		きごう	きごう
		そのた	そのた

付録 G

品詞コード表

品詞コード	品詞名	例
#T35	名詞 (一般的な名詞)	山、本
#T30	サ変名詞	努力、検査
#KK	団体・会社名	
#JN	人名	
#CN	地名	
#K5	動詞 (か行 5 段活用)	描く
#G5	動詞 (が行 5 段活用)	注ぐ
#S5	動詞 (さ行 5 段活用)	倒す
#T5	動詞 (た行 5 段活用)	絶つ
#N5	動詞 (な行 5 段活用)	死ぬ
#B5	動詞 (ば行 5 段活用)	転ぶ
#M5	動詞 (ま行 5 段活用)	住む
#R5	動詞 (ら行 5 段活用)	威張る
#W5	動詞 (わ (あ) 行 5 段活用)	言う
#KS	動詞 (上下一段活用)	降りる
#KX	動詞 (カ変活用)	来る
#ZX	動詞 (ザ変活用)	感ずる
#SX	動詞 (サ変活用)	関する
#K5r	動詞 (か行 5 段で連用形が名詞)	動く
#C5r	動詞 (か行 5 段で連用形が名詞特殊)	行く
#G5r	動詞 (が行 5 段で連用形が名詞)	急ぐ
#S5r	動詞 (さ行 5 段で連用形が名詞)	写す
#T5r	動詞 (た行 5 段で連用形が名詞)	勝つ
#B5r	動詞 (ば行 5 段で連用形が名詞)	遊ぶ
#M5r	動詞 (ま行 5 段で連用形が名詞)	歩む
#R5r	動詞 (ら行 5 段で連用形が名詞)	見張る

#W5r	動詞（わ（あ）行 5 段で連用形が名詞）	扱う
#KSr	動詞（上下 1 段，語幹が名詞）	生きる
#KY	形容詞	美しい、早い
#KYT	形容詞	古い
#T00	形容動詞（サ変名詞としても使う）	心配だ
#T05	形容動詞	幸運だ
#F04	副詞	
#F06	副詞	
#F12	副詞	
#F14	副詞	飽くまで
#KJ	単漢字	
#NN	数詞	何
#RT	連体詞	
#CJ	接続詞・感動詞	及び

付録 H

カタカナコード一覧表

code	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2	SP	!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	____
6	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
8																
9																
A		。	「	」	、	・	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
B	-	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
C	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
E																
F																

注意　SP は空白文字です。 DEL は消去文字です。

付録 I

16 進漢字コード一覧表

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
2120		‐	、	。	,	.	・	:	;	?	!	゛	°	´	`	¨
2130	^	¯	￣	ヽ	ゝ	ゞ	'	〃	仝	々	〆	〇	ー	—	-	／
2140	＼	～	‖	｜	…	‥	‘	’	“	”	（	）	〔	〕	［	］
2150	{	}	〈	〉	《	》	「	」	『	』	【	】	＋	－	±	×
2160	÷	＝	≠	＜	＞	≦	≧	∞	∴	♂	♀	°	′	″	℃	¥
2170	$	¢	£	%	#	&	*	@	§	☆	★	○	●	◎	◇	
2220		◆	□	■	△	▲	▽	▼	※	〒	→	←	↑	↓	〓	
2230											∈	∋	⊆	⊇	⊂	⊃
2240	∪	∩									∧	∨	¬	⇒	⇔	∀
2250	∃												∠	⊥	⌒	∂
2260	∇	≡		≪	≫	√	∽	∝	∵	∫	∬					
2270			Å	‰	♯	♭	♪	†	‡	¶					○	
2330	0	1	2	3	4	5	6	7	8	9						
2340		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
2350	P	Q	R	S	T	U	V	W	X	Y	Z					
2360		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
2370	p	q	r	s	t	u	v	w	x	y	z					
2420		ぁ	あ	ぃ	い	ぅ	う	ぇ	え	ぉ	お	か	が	き	ぎ	く
2430	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
2440	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
2450	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
2460	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
2470	ゐ	ゑ	を	ん												
2520		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
2530	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
2540	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
2550	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	ビ	ミ
2560	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
2570	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									
2620		A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O
2630	Π	P	Σ	T	Y	Φ	X	Ψ	Ω							
2640		α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
2650	π	ρ	σ	τ	υ	φ	χ	ψ	ω							
2720		А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
2730	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
2740	Ю	Я														

注意: 2121 は全角スペースです。

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
2750		а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
2760	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
2770	ю	я														
2820																
2830																
2840																
3020		亜	唖	娃	阿	哀	愛	挨	姻	逢	葵	茜	穐	悪	握	渥
3030	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
3040	粟	袷	安	庵	按	暗	案	闇	鞍	杏	以	伊	位	依	偉	囲
3050	夷	委	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	尉	胃
3060	萎	衣	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸
3070	稲	茨	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭	
3120		院	陰	隠	韻	吋	右	宇	鰻	姥	羽	雨	卯	鬩	窺	丑
3130	碓	臼	渦	嘘	唄	鬱	蔚	鰻	映	厩	浦	瓜	閏	噂	云	運
3140	雲	荏	餌	叡	営	嬰	影	映	曳	栄	永	泳	洩	瑛	盈	穎
3150	頴	英	衛	詠	鋭	液	疫	益	駅	悦	謁	越	閲	榎	厭	円
3160	園	堰	奄	宴	延	怨	掩	援	沿	演	炎	焔	煙	燕	猿	縁
3170	艶	苑	薗	遠	鉛	鴛	塩	於	汚	甥	凹	央	奥	往	応	
3220		押	旺	横	欧	殴	王	翁	襖	鴬	鴎	黄	岡	沖	荻	億
3230	屋	憶	臆	桶	牡	乙	俺	卸	恩	温	穏	音	下	化	仮	何
3240	伽	価	佳	加	可	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河
3250	火	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦	課	嘩	貨
3260	迦	過	霞	蚊	俄	峨	我	牙	画	臥	芽	蛾	賀	雅	餓	駕
3270	介	会	解	回	塊	壊	廻	快	怪	悔	恢	懐	戒	拐	改	
3320		魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階	貝	凱	劾
3330	外	咳	害	崖	慨	概	涯	碍	蓋	街	該	鎧	骸	浬	馨	蛙
3340	垣	柿	蛎	鈎	劃	嚇	各	廓	拡	撹	格	核	殻	獲	確	穫
3350	覚	角	赫	較	郭	閣	隔	革	学	岳	楽	額	顎	掛	笠	樫
3360	橿	梶	鰍	潟	割	喝	恰	括	活	渇	滑	葛	褐	轄	且	鰹
3370	叶	椛	樺	鞄	株	兜	竃	蒲	釜	鎌	噛	鴨	栢	茅	萱	
3420		粥	刈	苅	瓦	乾	侃	冠	寒	刊	勘	勧	巻	喚	堪	姦
3430	完	官	寛	干	幹	患	感	慣	憾	換	敢	柑	桓	棺	款	歓
3440	汗	漢	澗	潅	環	甘	監	看	竿	管	簡	緩	缶	翰	肝	艦
3450	莞	観	諌	貫	還	鑑	間	閑	関	陥	韓	館	舘	丸	含	岸
3460	巌	玩	癌	眼	岩	翫	贋	雁	頑	顔	願	企	伎	危	喜	器
3470	基	奇	嬉	寄	岐	希	幾	忌	揮	机	旗	既	期	棋	棄	
3520		機	帰	毅	気	汽	畿	祈	季	稀	紀	徽	規	記	貴	起
3530	軌	輝	飢	騎	鬼	亀	偽	儀	妓	宜	戯	技	擬	欺	犠	疑

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
3540	祇	義	蟻	誼	議	掬	菊	鞠	吉	吃	喫	桔	橘	詰	砧	杵
3550	黍	却	客	脚	虐	逆	丘	久	仇	休	及	吸	宮	弓	急	救
3560	朽	求	汲	泣	灸	球	究	窮	笈	級	糾	給	旧	牛	去	居
3570	巨	拒	拠	挙	渠	虚	許	距	鋸	漁	禦	魚	亨	享	京	
3620		供	侠	僑	兇	競	共	凶	協	匡	卿	叫	喬	境	峡	強
3630	彊	怯	恐	恭	挟	教	橋	況	狂	狭	矯	胸	脅	興	蕎	郷
3640	鏡	響	饗	驚	仰	凝	尭	暁	業	局	曲	極	玉	桐	粁	僅
3650	勤	均	巾	錦	斤	欣	欽	琴	禁	禽	筋	緊	芹	菌	衿	襟
3660	謹	近	金	吟	銀	九	倶	句	区	狗	玖	矩	苦	躯	駆	駈
3670	駒	具	愚	虞	喰	空	偶	寓	遇	隅	串	櫛	釧	屑	屈	
3720		掘	窟	沓	靴	轡	窪	熊	隈	粂	栗	繰	桑	啓	鍬	君
3730	薫	訓	群	軍	郡	卦	袈	祁	係	傾	刑	兄	桂	渓	圭	型
3740	契	形	径	恵	慶	慧	憩	掲	携	敬	景	桂	頚	鶏	畦	系
3750	経	継	繋	罫	茎	荊	蛍	計	詣	警	軽	結	血	芸	迎	鯨
3760	劇	戟	撃	激	隙	桁	傑	欠	決	潔	穴	結	懸	訣	月	件
3770	倹	倦	健	兼	券	剣	喧	圏	堅	嫌	建	憲	謙	拳	捲	
3820		検	権	牽	犬	献	研	硯	絹	県	肩	見	賢	軒	遣
3830	鍵	険	顕	験	鹸	元	原	厳	幻	弦	減	源	玄	現	絃	舷
3840	言	諺	限	乎	個	古	呼	固	姑	孤	己	庫	弧	戸	故	枯
3850	湖	狐	糊	袴	股	胡	菰	虎	誇	跨	鈷	雇	顧	鼓	五	互
3860	伍	午	呉	吾	娯	後	御	悟	梧	檎	瑚	碁	語	誤	護	醐
3870	乞	鯉	交	佼	侯	候	倖	光	公	功	効	勾	厚	口	向	
3920		后	喉	坑	垢	好	孔	孝	宏	工	巧	巷	幸	広	庚	康
3930	弘	恒	慌	抗	拘	控	攻	昂	晃	更	杭	校	梗	構	江	洪
3940	浩	港	溝	甲	皇	硬	稿	糠	紅	紘	絞	綱	耕	考	肯	肱
3950	腔	膏	航	荒	行	衡	講	貢	購	郊	酵	鉱	砿	鋼	閤	降
3960	項	香	高	鴻	剛	劫	号	合	壕	拷	濠	豪	轟	麹	克	刻
3970	告	国	穀	酷	鵠	黒	獄	墾	婚	恨	懇	昏	昆	根	混	痕
3a20		此	頃	今	困	坤	叉	催	嗟	左	差	査	沙	瑳	砂	詐	鎖
3a30	紺	艮	魂	些	佐	債	犀	砕	最	哉	塞	妻	宰	彩	才	採	栽
3a40	裟	坐	座	挫	債	催	砕	阪	祭	斎	細	菜	裁	載	際	剤	在
3a50	歳	済	災	采	坂	阪	堺	策	榊	肴	咲	崎	埼	碕	鷺	作	削
3a60	材	罪	財	冴	柵	窄	策	索	錯	桜	鮭	笹	崎	埼	冊	刷
3a70	咋	搾	昨	朔	擦	札	殺	薩	雑	皐	鯖	捌	算	崎	鮫	皿
3b20		察	拶	撮	擦	撒	散	桟	燦	珊	産	司	史	笹	蚕	晒
3b30	三	傘	参	山	惨	仕	仔	伺	使	刺	司	斯	施	埼	四	賛
3b40	酸	餐	斬	暫	残	師	志	思	指	支	孜	斯	施	旨	讃	始
3b50	姉	姿	子	屍	市	師	志	思	指	支	孜	斯	施	旨	士	止

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
3b60	死	氏	獅	祉	私	糸	紙	紫	肢	脂	至	視	詞	詩	試	誌
3b70	諮	資	賜	雌	飼	歯	事	似	侍	児	字	寺	慈	持	時	
3c20		次	滋	治	爾	璽	痔	磁	示	而	耳	自	蒔	辞	汐	鹿
3c30	式	識	鴫	竺	軸	宍	雫	七	叱	執	失	嫉	室	悉	湿	漆
3c40	疾	質	実	蔀	篠	偲	柴	芝	屡	蕊	縞	舎	写	射	捨	赦
3c50	斜	煮	社	紗	者	謝	車	遮	蛇	邪	借	勺	尺	杓	灼	爵
3c60	酌	釈	錫	若	寂	弱	惹	主	取	守	手	朱	殊	狩	珠	種
3c70	腫	趣	酒	首	儒	受	呪	寿	授	樹	綬	需	囚	収	周	
3d20		宗	就	州	修	愁	拾	洲	秀	秋	終	繍	習	臭	舟	蒐
3d30	衆	襲	讐	蹴	輯	週	酋	酬	集	醜	什	住	充	十	従	戎
3d40	柔	汁	渋	獣	縦	重	銃	叔	夙	宿	淑	祝	縮	粛	塾	熟
3d50	出	術	述	俊	峻	春	瞬	竣	舜	駿	准	循	旬	楯	殉	淳
3d60	準	潤	盾	純	巡	遵	醇	順	処	初	所	暑	曙	渚	庶	緒
3d70	署	書	薯	藷	諸	助	叙	女	序	徐	恕	鋤	除	傷	償	
3e20		勝	匠	升	召	哨	商	唱	嘗	奨	妾	娼	宵	将	小	少
3e30	尚	庄	床	廠	彰	承	抄	招	掌	捷	昇	昌	昭	晶	松	梢
3e40	樟	焦	沼	紹	渉	湘	焼	焦	照	症	省	硝	礁	祥	称	章
3e50	笑	粧	紹	肖	菖	蒋	蕉	衝	裳	訟	証	詔	詳	象	賞	醤
3e60	鉦	鍾	鐘	障	鞘	上	丈	丞	乗	冗	剰	城	場	壌	嬢	常
3e70	情	擾	条	杖	浄	状	畳	穣	蒸	譲	醸	錠	嘱	埴	飾	
3f20		拭	植	殖	燭	織	職	色	触	食	蝕	辱	尻	伸	信	侵
3f30	唇	娠	寝	審	心	慎	振	新	晋	森	榛	浸	深	申	疹	真
3f40	神	秦	紳	臣	芯	薪	親	診	身	辛	進	針	震	人	仁	刃
3f50	塵	壬	尋	甚	尽	腎	訊	迅	陣	靭	笥	諏	須	酢	図	厨
3f60	逗	吹	垂	帥	推	水	炊	睡	粋	翠	衰	遂	酔	錐	錘	随
3f70	瑞	髄	崇	嵩	数	枢	趨	雛	据	杉	椙	菅	頗	雀	裾	
4020		澄	摺	寸	世	瀬	畝	是	凄	制	勢	姓	征	性	成	政
4030	整	星	晴	棲	栖	正	静	斉	生	盛	精	聖	声	製	西	誠
4040	誓	請	逝	醒	青	脊	責	赤	税	隻	席	惜	戚	斥	昔	析
4050	石	積	籍	績	脊	舌	蝉	跡	脆	碩	切	拙	接	摂	折	設
4060	窃	節	説	雪	絶	浅	洗	仙	先	千	占	宣	専	尖	川	戦
4070	扇	撰	栓	栴	泉	船	薦	染	潜	煎	煽	旋	穿	箭	線	
4120		繊	羨	腺	舛	禅	繕	詮	賤	践	選	遷	銭	銑	閃	鮮
4130	前	善	漸	然	全	租	粗	膳	糎	噌	塑	岨	措	曾	曽	楚
4140	狙	疏	疎	礎	祖	租	粗	素	組	蘇	訴	阻	遡	鼠	僧	創
4150	双	叢	倉	喪	壮	奏	爽	宋	層	匝	惣	想	捜	掃	挿	掻
4160	操	早	曹	巣	槍	槽	漕	燥	争	痩	相	窓	糟	総	綜	聡
4170	草	荘	葬	蒼	藻	装	走	送	遭	鎗	霜	騒	像	増	憎	

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
4220		臓	蔵	贈	造	促	側	則	即	息	捉	束	測	足	速	俗
4230	属	賊	族	続	卒	袖	其	揃	存	孫	尊	損	村	遜	他	多
4240	太	汰	詑	唾	堕	妥	惰	打	柁	舵	楕	陀	駄	騨	体	堆
4250	対	耐	岱	帯	待	怠	態	戴	替	泰	滞	胎	腿	苔	袋	貸
4260	退	逮	隊	黛	鯛	代	台	大	第	醍	題	鷹	滝	瀧	卓	啄
4270	宅	托	択	拓	沢	濯	琢	託	鐸	濁	諾	茸	凧	蛸	只	
4320		叩	但	達	辰	奪	脱	巽	竪	辿	棚	谷	狸	鱈	樽	誰
4330	丹	単	嘆	坦	担	探	旦	歎	淡	湛	炭	短	端	箪	綻	耽
4340	胆	蛋	誕	鍛	団	壇	弾	断	暖	檀	段	男	談	値	知	地
4350	弛	恥	智	池	痴	稚	置	致	蜘	遅	馳	築	畜	竹	筑	蓄
4360	逐	秩	窒	鋳	茶	着	中	仲	宙	忠	抽	昼	柱	注	虫	衷
4370	註	酎	鋳	駐	樗	瀦	猪	苧	著	貯	丁	兆	凋	喋	寵	
4420		帖	帳	庁	弔	張	彫	徴	懲	挑	暢	朝	潮	勅	町	眺
4430	聴	脹	腸	蝶	調	諜	超	跳	銚	長	頂	鳥	勅	捗	直	朕
4440	沈	珍	賃	鎮	陳	津	墜	椎	槌	追	鎚	痛	通	塚	栂	掴
4450	槻	佃	漬	柘	辻	蔦	綴	鍔	椿	潰	坪	壷	嬬	紬	爪	吊
4460	釣	鶴	亭	低	停	偵	剃	錠	貞	堤	定	帝	底	庭	廷	弟
4470	悌	抵	挺	提	梯	汀	泥	碇	禎	程	締	艇	諦	蹄	逓	
4520		邸	鄭	釘	鼎	泥	摘	擢	敵	滴	的	笛	適	鏑	溺	哲
4530	徹	撤	轍	迭	鉄	典	填	天	展	店	添	纏	甜	貼	転	顛
4540	点	伝	殿	澱	田	電	兎	吐	堵	塗	妬	屠	徒	斗	杜	渡
4550	登	菟	賭	途	都	鍍	砥	砺	努	度	土	奴	怒	倒	党	冬
4560	凍	刀	唐	塔	塘	套	宕	島	嶋	悼	投	搭	東	桃	梼	棟
4570	盗	淘	湯	涛	灯	燈	当	痘	祷	等	答	筒	糖	統	到	
4620		董	蕩	藤	討	謄	豆	踏	逃	透	鐙	陶	頭	騰	闘	働
4630	動	同	堂	導	憧	撞	洞	瞳	童	胴	萄	道	銅	峠	鴇	匿
4640	得	徳	涜	特	督	禿	篤	毒	独	読	栃	橡	凸	突	椴	届
4650	鳶	苫	寅	酉	瀞	噸	屯	惇	敦	沌	豚	遁	頓	呑	曇	鈍
4660	奈	那	内	乍	凪	薙	謎	灘	捺	鍋	楢	馴	縄	暖	南	楠
4670	軟	難	汝	二	尼	弐	迩	匂	賑	肉	虹	廿	日	乳	入	
4720		如	尿	韮	任	妊	忍	認	濡	禰	祢	寧	葱	猫	熱	年
4730	念	捻	撚	燃	粘	乃	廼	之	埜	嚢	悩	濃	納	能	脳	膿
4740	農	覗	蚤	巴	把	播	覇	杷	波	派	琶	破	婆	罵	芭	馬
4750	俳	廃	拝	排	敗	杯	盃	牌	背	肺	輩	配	倍	培	媒	梅
4760	楳	煤	狽	買	売	賠	陪	這	蝿	秤	矧	萩	伯	剥	博	拍
4770	柏	泊	白	箔	粕	舶	薄	迫	曝	漠	爆	縛	莫	駁	麦	
4820		函	箱	硲	箸	肇	筈	櫨	幡	肌	畑	畠	八	鉢	滝	発
4830	醗	髪	伐	罰	抜	筏	閥	鳩	噺	塙	蛤	隼	伴	判	半	反

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
4840	叛	帆	搬	斑	板	氾	汎	版	犯	班	畔	繁	般	藩	販	範
4850	釆	煩	頒	飯	挽	晩	番	盤	磐	蕃	蛮	匪	卑	否	妃	庇
4860	彼	悲	扉	批	披	斐	比	泌	疲	皮	碑	秘	緋	罷	肥	被
4870	誹	費	避	非	飛	樋	簸	備	尾	微	枇	毘	琵	眉	美	
4920		鼻	柊	稗	匹	疋	髭	彦	膝	菱	肘	弼	必	畢	筆	逼
4930	桧	姫	媛	紐	百	謬	俵	彪	標	氷	漂	瓢	票	表	評	豹
4940	廟	描	病	秒	苗	錨	鋲	蒜	蛭	鰭	品	彬	斌	浜	瀕	貧
4950	賓	頻	敏	瓶	不	付	埠	夫	婦	富	冨	布	府	怖	扶	敷
4960	斧	普	浮	父	符	腐	膚	芙	譜	負	賦	赴	阜	附	侮	撫
4970	武	舞	葡	蕪	部	封	楓	風	葺	蕗	伏	副	復	幅	服	
4a20		福	腹	複	覆	淵	弗	払	沸	仏	物	鮒	分	吻	噴	墳
4a30	憤	扮	焚	奮	粉	糞	陛	米	返	頁	僻	壁	丙	癖	併	平
4a40	弊	柄	並	蔽	閉	編	辺	返	遍	便	勉	娩	弁	碧	別	蔑
4a50	偏	変	片	篇	補	輔	穂	募	墓	慕	戊	暮	母	簿	保	舗
4a60	圃	捕	歩	甫	奉	宝	峰	峯	崩	庖	抱	捧	放	方	訪	豊
4a70	俸	包	呆	報	砲	縫	胞	芳	萌	蓬	蜂	褒	忙	房	暴	望
4b20		法	泡	烹	亡	傍	剖	坊	妨	帽	忘	忙	房	暴	望	某
4b30	飽	鳳	鵬	乏	亡	傍	剖	貿	鉾	防	吠	頬	北	僕	卜	墨
4b40	棒	冒	紡	肪	膨	謀	貌	貿	鉾	防	吠	頬	北	僕	卜	墨
4b50	撲	朴	牧	睦	穆	釦	勃	没	殆	堀	幌	奔	本	翻	凡	盆
4b60	摩	磨	魔	麻	埋	妹	昧	枚	毎	哩	槙	幕	膜	枕	鮪	柾
4b70	鱒	桝	亦	俣	又	抹	末	沫	迄	侭	繭	麿	万	慢	満	
4c20		漫	蔓	味	未	魅	巳	箕	岬	密	蜜	湊	蓑	稔	脈	妙
4c30	粍	民	眠	務	夢	無	牟	矛	霧	鵡	椋	婿	娘	冥	名	命
4c40	明	盟	迷	銘	鳴	姪	牝	滅	免	棉	綿	緬	面	麺	摸	模
4c50	茂	妄	孟	毛	猛	盲	網	耗	蒙	儲	木	黙	目	杢	勿	餅
4c60	尤	戻	籾	貰	問	悶	紋	門	匁	也	冶	夜	爺	耶	野	弥
4c70	矢	厄	役	約	薬	訳	躍	靖	柳	薮	鑓	愉	愈	油	癒	
4d20		諭	輸	唯	佑	優	勇	友	宥	幽	悠	憂	揖	有	予	湧
4d30	涌	猶	猷	由	祐	裕	誘	遊	邑	郵	雄	融	夕	楊	余	与
4d40	誉	輿	預	傭	幼	妖	容	庸	揚	揺	擁	曜	洋	様	洋	溶
4d50	熔	用	窯	羊	耀	葉	蓉	要	謡	踊	遥	陽	養	慾	抑	欲
4d60	沃	浴	翌	翼	淀	羅	螺	裸	来	莱	頼	雷	洛	絡	落	酪
4d70	乱	卵	嵐	欄	濫	藍	蘭	覧	利	吏	履	李	梨	理	璃	
4e20		痢	裏	裡	里	離	陸	律	率	立	葎	掠	略	劉	流	溜
4e30	琉	留	硫	粒	隆	竜	龍	侶	慮	旅	虜	了	亮	僚	両	凌
4e40	寮	料	梁	涼	猟	療	瞭	稜	糧	良	諒	遼	量	陵	領	力
4e50	緑	倫	厘	林	淋	燐	琳	臨	輪	隣	鱗	麟	瑠	塁	涙	累

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
4e60	類	令	伶	例	冷	励	嶺	怜	玲	礼	苓	鈴	隷	零	霊	麗
4e70	齢	暦	歴	列	劣	烈	裂	廉	恋	憐	漣	煉	簾	練	聯	
4f20		蓮	連	錬	呂	魯	櫓	炉	賂	路	露	労	婁	廊	弄	朗
4f30	楼	榔	浪	漏	牢	狼	篭	老	聾	蝋	郎	六	麓	禄	肋	録
4f40	論	倭	和	話	歪	賄	脇	惑	枠	鷲	亙	亘	鰐	詫	藁	蕨
4f50	椀	湾	碗	腕												
5020		弌	丐	丕	个	丱	丶	丼	丿	乂	乖	乘	亂	亅	豫	亊
5030	舒	弍	于	亞	亟	亠	亢	亰	亳	亶	从	仍	仄	仆	仂	仗
5040	仞	仭	仟	价	伉	佚	估	佛	佝	佗	佇	佶	侈	侏	侘	佻
5050	佩	佰	侑	佯	來	侖	儘	俔	俟	俎	俘	俛	俑	俚	俐	俤
5060	俥	倚	倨	倔	倪	倥	倅	伜	俶	倡	倩	倬	俾	俯	們	倆
5070	偃	假	會	偕	偐	偈	做	偖	偬	偸	傀	傚	傅	傴	傲	
5120		僉	僊	傳	僂	僖	僞	僥	僭	僣	僮	價	僵	儉	儁	儂
5130	儖	儕	儔	儚	儡	儺	儷	儼	儻	儿	兀	兒	兌	兔	兢	竸
5140	兩	兪	兮	冀	冂	囘	册	冉	冏	冑	冓	冕	冖	冤	冦	冢
5150	冩	冪	冫	决	冱	冲	冰	况	冽	凅	凉	凛	几	處	凩	凭
5160	凰	凵	凾	刄	刋	刔	刎	刧	刪	刮	刳	刹	剏	剄	剋	剌
5170	剞	剔	剪	剴	剩	剳	剿	剽	劍	劔	劒	剱	劈	劑	辨	
5220		辧	劬	劭	劼	劵	勁	勍	勗	勞	勣	勦	飭	勠	勳	勵
5230	勸	勹	匆	匈	甸	匍	匐	匏	匕	匚	匣	匯	匱	匳	匸	區
5240	卆	卅	丗	卉	卍	凖	卞	卩	卮	夘	卻	卷	厂	厖	厠	厦
5250	厥	厮	厰	厶	參	簒	雙	叟	曼	燮	叮	叨	叭	叺	吁	吽
5260	呀	听	吭	吼	吮	吶	吩	吝	呎	咏	呵	咎	呟	呱	呷	呰
5270	咒	呻	咀	呶	咄	咐	咆	哇	咢	咸	咥	咬	哄	哈	咨	
5320		咫	哂	咤	咾	咼	哘	哥	哦	唏	唔	哽	哮	哭	哺	哢
5330	唹	啀	啣	啌	售	啜	啅	啖	啗	唸	唳	啝	喙	喀	咯	喊
5340	喟	啻	啾	喘	喞	單	啼	喃	喩	喇	喨	嗚	嗅	嗟	嗄	嗜
5350	嗤	嗔	嘔	嗷	嘖	嗾	嗽	嘛	嗹	噎	噐	營	嘴	嘶	嘲	嘸
5360	噫	噤	嘯	噬	噪	嚆	嚀	嚊	嚠	嚔	嚏	嚥	嚮	嚶	嚴	囂
5370	嚼	囁	囃	囀	囈	囎	囑	囓	囗	囮	囹	圀	囿	圄	圉	
5420		圈	國	圍	圓	團	圖	嗇	圜	圦	圷	圸	坎	圻	址	坏
5430	坩	埀	垈	坡	坿	垉	垓	垠	垳	垤	垪	垰	埃	埆	埔	埒
5440	埓	堊	埖	埣	堋	堙	堝	塲	堡	塢	塋	塰	毀	塒	堽	塹
5450	墅	墹	墟	墫	墺	壞	墻	墸	墮	壅	壓	壑	壗	壙	壘	壥
5460	壜	壤	壟	壯	壺	壹	壻	壼	壽	夂	夊	夐	夛	梦	夥	夬
5470	夭	夲	夸	夾	竒	奕	奐	奎	奚	奘	奢	奠	奧	奬	奩	
5520		奸	妁	妝	佞	侫	妣	妲	姆	姨	姜	妍	姙	姚	娥	娟
5530	娑	娜	娉	娚	婀	婬	婉	娵	娶	婢	婪	媚	媼	媾	嫋	嫂

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
5540	媽	嫣	嫗	嫦	嫩	嫖	嫺	嫻	嬌	嬋	嬖	嬲	嫐	嬪	嬶	嬾
5550	孃	孅	孀	孑	孕	孚	孛	孥	孩	孰	孳	孵	學	斈	孺	宀
5560	它	宦	宸	寃	寇	寉	寔	寐	寤	實	寢	寞	寥	寫	寰	寶
5570	寳	尅	將	專	對	尓	尠	尢	尨	尸	尹	屁	屆	屎	屓	
5620		屐	屏	孱	屬	屮	乢	屶	屹	岌	岑	岔	妛	岫	岻	岶
5630	岼	岷	峅	岾	峇	峙	峩	峽	峺	峭	嶌	峪	崋	崕	崗	嵜
5640	崟	崛	崑	崔	崢	崚	崙	崘	嵌	嵒	嵎	嵋	嵬	嵳	嵶	嶇
5650	嶄	嶂	嶢	嶝	嶬	嶮	嶽	嶐	嶷	嶼	巉	巍	巓	巒	巖	巛
5660	巫	已	巵	帋	帚	帙	帑	帛	帶	帷	幄	幃	幀	幎	幗	幔
5670	幟	幢	幤	幇	幵	并	幺	麼	广	庠	廁	廂	廈	廐	廏	
5720		廖	廣	廝	廚	廛	廢	廡	廨	廩	廬	廱	廳	廰	廴	廸
5730	廾	弃	弉	彝	彜	弋	弑	弖	弩	弭	弸	彁	彈	彌	彎	弯
5740	彑	彖	彗	彙	彡	彭	彳	彷	徃	徂	彿	徊	很	徑	徇	從
5750	徙	徘	徠	徨	徭	徼	忖	忻	忤	忸	忱	忝	悳	忿	怡	恠
5760	怙	怐	怩	怎	怱	怛	怕	怫	怦	怏	怺	恚	恁	恪	恷	恟
5770	恊	恆	恍	恣	恃	恤	恂	恬	恫	恙	悁	悍	惧	悃	悚	
5820		悄	悛	悖	悗	悒	悧	悋	惡	悸	惠	惓	悴	忰	悽	惆
5830	悵	惘	慍	愕	愆	惶	惷	愀	惴	惺	愃	愡	惻	惱	愍	愎
5840	慇	愾	愨	愧	慊	愿	愼	愬	愴	愽	慂	慄	慳	慷	慘	慙
5850	慚	慫	慴	慯	慥	慱	慟	慝	慓	慵	憙	憖	憇	憬	憔	憚
5860	憊	憑	憫	憮	懌	懊	應	懷	懈	懃	懆	憺	懋	罹	懍	懦
5870	懣	懶	懺	懴	懿	懽	懼	懾	戀	戈	戉	戍	戌	戔	戛	
5920		戞	戡	截	戮	戰	戲	戳	扁	扎	扞	扣	扛	扠	扨	扼
5930	抂	抉	找	抒	抓	抖	拔	抃	抔	拗	拑	抻	拏	拿	拆	擔
5940	拈	拜	拌	拊	拂	拇	抛	拉	挌	拮	拱	挧	挂	挈	拯	拵
5950	捐	挾	捍	搜	捏	掖	掎	掀	掫	捶	掣	掏	掉	掟	掵	捫
5960	捩	掾	揩	揀	揆	揣	揉	插	揶	揄	搖	搴	搆	搓	搦	搶
5970	攝	搗	搨	搏	摧	摯	摶	摎	攪	撕	撓	撥	撩	撈	撼	
5a20		據	擒	擅	擇	撻	擘	擂	擱	擧	舉	擠	擡	抬	擣	擯
5a30	攬	擶	擴	擲	擺	攀	擽	攘	攜	攅	攤	攣	攫	攴	攵	攷
5a40	收	攸	畋	效	敖	敕	敍	敘	敞	敝	敲	數	斂	斃	變	斛
5a50	斟	斫	斷	旃	旆	旁	旄	旌	旒	旛	旙	无	旡	旱	杲	昊
5a60	昃	旻	杳	昵	昶	昴	昜	晏	晄	晉	晁	晞	晝	晤	晧	晨
5a70	晟	晢	晰	暃	暈	暎	暉	暄	暘	暝	曁	暹	曉	暾	暼	
5b20		曄	暸	曖	曚	曠	昿	曦	曩	曰	曵	曷	朏	朖	朞	朦
5b30	朧	霸	朮	朿	朶	杁	朸	朷	杆	杞	杠	杙	杣	杤	枉	杰
5b40	枩	杼	杪	枌	枋	枦	枡	枅	枷	柯	枴	柬	枳	柩	枸	柤
5b50	柞	柝	柢	柮	枹	柎	柆	柧	檜	栞	框	栩	桀	桍	栲	桎

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
5b60	梳	栫	桙	档	桷	桿	梟	梏	梭	梔	條	梛	梃	檮	梹	桴
5b70	梵	梠	梺	椏	梍	桾	椁	棊	椈	棘	椢	椦	棡	椌	棍	
5c20		棔	棧	棕	椶	椒	椄	棗	棣	椥	棹	棠	棯	椨	椪	椚
5c30	椣	椡	棆	楹	楷	楜	楸	楫	楔	楾	楮	椹	楴	椽	楙	椰
5c40	楡	楞	楝	榁	楪	榲	榮	槐	榿	槁	槓	榾	槎	寨	槊	槝
5c50	榻	槃	榧	樮	榑	榠	榜	榕	榴	槞	槨	樂	樛	槿	權	槹
5c60	槲	槧	樅	榱	樞	槭	樔	槫	樊	樒	櫁	樣	樓	橄	樌	橲
5c70	樶	橸	橇	橢	橙	橦	橈	樸	樢	檐	檍	檠	檄	檢	檣	
5d20		檗	蘗	檻	櫃	櫂	檸	檳	檬	櫞	櫑	櫟	檪	櫚	櫪	櫻
5d30	欅	蘖	櫺	欒	欖	鬱	欟	欸	欷	盜	欹	飮	歇	歃	歉	歐
5d40	歙	歔	歛	歟	歡	歸	歹	歿	殀	殄	殃	殍	殘	殕	殞	殤
5d50	殪	殫	殯	殲	殱	殳	殷	殼	毆	毋	毓	毟	毬	毫	毳	毯
5d60	麾	氈	氓	气	氛	氤	氣	汞	汕	汢	汪	沂	沍	沚	沁	沛
5d70	汾	汨	汳	沒	沐	泄	泱	泓	沽	泗	泅	泝	沮	沱	沾	
5e20		沺	泛	泯	泙	泪	洟	衍	洶	洫	洽	洸	洙	洵	洳	洒
5e30	洌	浣	涓	浤	浚	浹	浙	涎	涕	濤	涅	淹	渕	渊	涵	淇
5e40	淦	涸	淆	淬	淞	淌	淨	淒	淅	淺	淙	淤	淕	淪	淮	渭
5e50	湮	渮	渙	湲	湟	渾	渣	湫	渫	湶	湍	渟	湃	渺	湎	渤
5e60	滿	渝	游	溂	溪	溘	滉	溷	滓	溽	溯	滄	溲	滔	滕	溏
5e70	溥	滂	溟	潁	漑	灌	滬	滸	滾	漿	滲	漱	滯	漲	滌	
5f20		漾	漓	滷	澆	潺	潸	澁	澀	潯	潛	濳	潭	澂	潼	潘
5f30	澎	澑	濂	潦	澳	澣	澡	澤	澹	濆	澪	濟	濕	濬	濔	濘
5f40	濱	濮	濛	瀉	瀋	濺	瀑	瀁	瀏	濾	瀛	瀚	潴	瀝	瀘	瀟
5f50	瀰	瀾	瀲	灑	灣	炙	炒	炯	烱	炬	炸	炳	炮	烟	烋	烝
5f60	烙	焉	烽	焜	焙	煥	煕	熈	煦	煢	煌	煖	煬	熏	燻	熄
5f70	熕	熨	熬	燗	熹	熾	燒	燉	燔	燎	燠	燬	燧	燵	燼	
6020		燹	燿	爍	爐	爛	爨	爭	爬	爰	爲	爻	爼	爿	牀	牆
6030	牋	牘	牴	牾	犂	犁	犇	犒	犖	犢	犧	犹	犲	狃	狆	狄
6040	狎	狒	狢	狠	狡	狹	狷	倏	猗	猊	猜	猖	猝	猴	猯	猩
6050	猥	猾	獎	獏	默	獗	獪	獨	獰	獸	獵	獻	獺	珈	玳	珎
6060	玻	珀	珥	珮	珞	璢	琅	瑯	琥	珸	琲	琺	瑕	琿	瑟	瑙
6070	瑁	瑜	瑩	瑰	瑣	瑪	瑶	瑾	璋	璞	璧	瓊	瓏	瓔	珱	
6120		瓠	瓣	瓧	瓩	瓮	瓲	瓰	瓱	瓸	瓷	甄	甃	甅	甌	甎
6130	甍	甕	甓	甞	甦	甬	甼	畄	畍	畊	畉	畛	畆	畚	畩	畤
6140	畧	畫	畭	畸	當	疆	疇	畴	疊	疉	疂	疔	疚	疝	疥	疣
6150	痂	疳	痃	疵	疽	疸	疼	疱	痍	痊	痒	痙	痣	痞	痾	痿
6160	痼	瘁	痰	痺	痲	痳	瘋	瘍	瘉	瘟	瘧	瘠	瘡	瘢	瘤	瘴
6170	瘰	瘻	癇	癈	癆	癜	癘	癡	癢	癨	癩	癪	癧	癬	癰	

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
6220		癩	癶	癸	發	皀	皃	皈	皋	皎	皖	皓	皙	皚	皰	皴
6230	皸	皹	皺	盂	盍	盖	盒	盞	盡	盥	盧	盪	蘯	盻	眈	眇
6240	眄	眩	眤	眞	眥	眦	眛	眷	眸	睇	睚	睨	睫	睛	睥	睿
6250	睾	睹	瞎	瞋	瞑	瞠	瞞	瞰	瞶	瞹	瞿	瞼	瞽	瞻	矇	矍
6260	矗	矚	矜	矣	矮	矼	砌	砒	礦	砠	礪	硅	碎	硴	碆	硼
6270	碚	碌	碣	碵	碪	碯	磑	磆	磋	磔	碾	碼	磅	磊	磬	
6320		磧	磚	磽	磴	礇	礒	礑	礙	礬	礫	祀	祠	祗	祟	祚
6330	祕	祓	祺	祿	禊	禝	禧	齋	禪	禮	禳	禹	禺	秉	秕	秧
6340	秬	秡	秣	稈	稍	稘	稙	稠	稟	禀	稱	稻	稾	稷	穃	穗
6350	穉	穡	穢	穩	龝	穰	穹	穽	窈	窗	窕	窘	窖	窩	竈	窰
6360	窶	竅	竄	窿	邃	竇	竊	竍	竏	竕	竓	站	竚	竝	竡	竢
6370	竦	竭	竰	笂	笏	笊	笆	笳	笘	笙	笞	笵	笨	笶	筐	
6420		筺	笄	筍	笋	筌	筅	筵	筥	筴	筧	筰	筱	筬	筮	箝
6430	箘	箟	箍	箜	箚	箋	箒	箏	筝	箙	篋	篁	篌	篏	箴	篆
6440	篝	篩	簑	簔	篦	篥	籠	簀	簇	簓	篳	篷	簗	簍	篶	簣
6450	簧	簪	簟	簷	簫	簽	籌	籃	籔	籏	籀	籐	籘	籟	籤	籖
6460	籥	籬	籵	粃	粐	粤	粭	粢	粫	粡	粨	粳	粲	粱	粮	粹
6470	粽	糀	糅	糂	糘	糒	糜	糢	鬻	糯	糲	糴	糶	糺	紆	
6520		紂	紜	紕	紊	絅	絋	紮	紲	紿	紵	絆	絳	絖	絎	絲
6530	絨	絮	絏	絣	經	綉	絛	綏	絽	綛	綺	綮	綣	綵	緇	綽
6540	綫	總	綢	綯	緜	綸	綟	綰	緘	緝	緤	緞	緻	緲	緡	縅
6550	縊	縣	縡	縒	縱	縟	縉	縋	縢	繆	繦	縻	縵	縹	繃	縷
6560	縲	縺	繧	繝	繖	繞	繙	繚	繹	繪	繩	繼	繻	纃	緕	繽
6570	辮	繿	纈	纉	續	纒	纐	纓	纔	纖	纎	纛	纜	缸	缺	
6620		罅	罌	罍	罎	罐	网	罕	罔	罘	罟	罠	罨	罩	罧	罸
6630	羂	羆	羃	羈	羇	羌	羔	羞	羝	羚	羣	羯	羲	羹	羮	羶
6640	羸	譱	翅	翆	翊	翕	翔	翡	翦	翩	翳	翹	飜	耆	耄	耋
6650	耒	耘	耙	耜	耡	耨	耿	耻	聊	聆	聒	聘	聚	聟	聢	聨
6660	聳	聲	聰	聶	聹	聽	聿	肄	肆	肅	肛	肓	肚	肭	冐	肬
6670	胛	胥	胙	胝	胄	胚	胖	脉	胯	胱	脛	脩	脣	脯	腋	
6720		隋	腆	脾	腓	腑	胼	腱	腮	腥	腦	腴	膃	膈	膊	膀
6730	膂	膠	膕	膤	膣	腟	膓	膩	膰	膵	膾	膸	膽	臀	臂	膺
6740	臉	臍	臑	臙	臘	臈	臚	臟	臠	臧	臺	臻	臾	舁	舂	舅
6750	與	舊	舍	舐	舖	舩	舫	舸	舳	艀	艙	艘	艝	艚	艟	艤
6760	艢	艨	艪	艫	舮	艱	艷	艸	艾	芍	芒	芫	芟	芻	芬	苡
6770	苣	苟	苒	苴	苳	苺	莓	范	苻	苹	苞	茆	苜	茉	苙	
6820		茵	茴	茖	茲	茱	荀	茹	荐	荅	茯	茫	茗	茘	莅	莚
6830	莪	莟	莢	莖	茣	莎	莇	莊	荼	莵	荳	荵	莠	莉	莨	菴

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
6840	萓	董	菎	菽	萃	菘	萋	菁	菷	萇	菠	菲	萍	萢	萠	莽
6850	萸	蔆	菻	葭	萪	萼	蕚	蒄	葷	葫	蒭	葮	蒂	葩	葆	萬
6860	葯	葹	萵	蓊	葢	蒹	蒿	蒟	蓙	蓍	蒻	蓚	蓐	蓁	蓆	蓖
6870	蒡	蔡	蓿	蓴	蔗	蔘	蔬	蔟	蔕	蔔	蓼	蕀	蕣	蕘	蕈	
6920		蕁	蘂	蕋	蕕	薀	薤	薈	薑	薊	薨	蕭	薔	薛	藪	薇
6930	薜	蕷	蕾	薐	藉	薺	藏	薹	藐	藕	藝	藥	藜	藹	蘊	蘓
6940	蘋	藾	藺	蘆	蘢	蘚	蘰	蘿	虍	乕	虔	號	虧	虱	蚓	蚣
6950	蚩	蚪	蚋	蚌	蚶	蚯	蛄	蛆	蚰	蛉	蠣	蚫	蛔	蛞	蛩	蛬
6960	蛟	蛛	蛯	蜒	蜆	蜈	蜀	蜃	蛻	蜑	蜉	蜍	蛹	蜊	蜴	蜿
6970	蜷	蜻	蜥	蜩	蜚	蝠	蝟	蝸	蝌	蝎	蝴	蝗	蝨	蝮	蝙	
6a20		蝓	蝣	蝪	蠅	螢	螟	螂	螯	蟋	螽	蟀	蟐	雖	螫	蟄
6a30	螳	蟇	蟆	螻	蟯	蟲	蟠	蠏	蠍	蟾	蟶	蟷	蠎	蟒	蠑	蠖
6a40	蠕	蠢	蠡	蠱	蠶	蠹	蠧	蠻	衄	衂	衒	衙	衞	衢	衫	袁
6a50	衾	袞	衵	衽	袵	衲	袂	袗	袒	袮	袙	袢	袍	袤	袰	袿
6a60	袱	裃	裄	裔	裘	裙	裝	裹	褂	裼	裴	裨	裲	褄	褌	褊
6a70	褓	襃	褞	褥	褪	褫	襁	襄	褻	褶	褸	襌	褝	襠	襞	
6b20		襦	襤	襭	襪	襯	襴	襷	襾	覃	覈	覊	覓	覘	覡	覩
6b30	覦	覬	覯	覲	覺	覽	覿	觀	觚	觜	觝	觧	觴	觸	訃	訖
6b40	訐	訌	訛	訝	訥	訶	詁	詛	詒	詆	詈	詼	詭	詬	詢	誅
6b50	誂	誄	誨	誡	誑	誥	誦	誚	誣	諄	諍	諂	諚	諫	諳	諧
6b60	諤	諱	謔	諠	諢	諷	諞	諛	謌	謇	謚	諡	謖	謐	謗	謠
6b70	謳	鞫	謦	謫	謾	謨	譁	譌	譏	譎	證	譖	譛	譚	譫	
6c20		譟	譬	譯	譴	譽	讀	讌	讎	讒	讓	讖	讙	讚	谺	豁
6c30	谿	豈	豌	豎	豐	豕	豢	豬	豸	豺	貂	貉	貅	貊	貍	貎
6c40	貔	豼	貘	戝	貭	貪	貽	貲	貳	貮	貶	賈	賁	賤	賣	賚
6c50	賽	賺	賻	贄	贅	贊	贇	贏	贍	贐	齎	贓	賍	贔	贖	赧
6c60	赭	赱	赳	趁	趙	跂	趾	趺	跏	跚	跖	跌	跛	跋	跪	跫
6c70	跟	跣	跼	踈	踉	跿	踝	踞	踐	踟	蹂	踵	踰	踴	蹊	
6d20		蹇	蹉	蹌	蹐	蹈	蹙	蹤	蹠	踪	蹣	蹕	蹶	蹲	蹼	躁
6d30	躇	躅	躄	躋	躊	躓	躑	躔	躙	躪	躡	躬	躰	軆	躱	躾
6d40	軅	軈	軋	軛	軣	軼	軻	軫	軾	輊	輅	輕	輒	輙	輓	輜
6d50	輟	輛	輌	輦	輳	輻	輹	轅	轂	輾	轌	轉	轆	轎	轗	轜
6d60	轢	轣	轤	辜	辟	辣	辭	辯	辷	迚	迥	迢	迪	迯	邇	迴
6d70	逅	迹	迺	逑	逕	逡	逍	逞	逖	逋	逧	逶	逵	逹	迸	
6e20		遏	遐	遑	遒	逎	遉	逾	遖	遘	遞	遨	遯	遶	隨	遲
6e30	邂	遽	邁	邀	邊	邉	邏	邨	邯	邱	邵	郢	郤	扈	郛	鄂
6e40	鄒	鄙	鄲	鄰	酊	酖	酘	酣	酥	酩	酳	酲	醋	醉	醂	醢
6e50	醫	醯	醪	醵	醴	醺	釀	釁	釉	釋	釐	釖	釟	釡	釛	釼

code	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
6e60	釵	鈎	鈞	釿	鈔	鈬	鈕	鈑	鉞	鉗	鉅	鉉	鉤	鉈	銕	鈿
6e70	鉋	鉐	銜	銖	銓	銛	鉚	鋏	銹	銷	鋩	錏	鋺	鍄	錮	
6f20		錙	錢	錚	錣	錺	錵	錻	鍜	鍠	鍼	鍮	鍖	鎰	鎬	鎭
6f30	鎔	鎹	鏖	鏗	鏨	鏥	鏘	鏃	鏝	鏐	鏈	鏤	鐚	鐔	鐓	鐃
6f40	鐇	鐐	鐶	鐫	鐵	鐡	鐺	鑁	鑒	鑄	鑛	鑠	鑢	鑞	鑪	鈩
6f50	鑰	鑵	鑷	鑽	鑚	鑼	鑾	钁	鑿	閂	閇	閊	閔	閖	閘	閙
6f60	閠	閨	閧	閭	閼	閻	閹	閾	闊	濶	闃	闍	闌	闕	闔	闖
6f70	關	闡	闥	闢	阡	阨	阮	阯	陂	陌	陏	陋	陷	陜	陞	
7020		陝	陟	陦	陲	陬	隍	隘	隕	隗	險	隧	隱	隲	隰	隴
7030	隶	隸	隹	雎	雋	雉	雍	襍	雜	霍	雕	雹	霄	霆	霈	霓
7040	霎	霑	霏	霖	霙	霤	霪	霰	霹	霽	霾	靄	靆	靈	靂	靉
7050	靜	靠	靤	靦	靨	勒	靫	靱	靹	鞅	靼	鞁	靺	鞆	鞋	鞏
7060	鞐	鞜	鞨	鞦	鞣	鞳	鞴	韃	韆	韈	韋	韜	韭	齏	韲	竟
7070	韶	韵	頏	頌	頸	頤	頡	頷	頽	顆	顏	顋	顫	顯	顰	
7120		顱	顴	顳	颪	颯	颱	颶	飄	飃	飆	飩	飫	餃	餉	餒
7130	餔	餘	餡	餝	餞	餤	餠	餬	餮	餽	餾	饂	饉	饅	饐	饋
7140	饑	饒	饌	饕	馗	馘	馥	馭	馮	馼	駟	駛	駝	駘	駑	駭
7150	駮	駱	駲	駻	駸	騁	騏	騅	駢	騙	騫	騷	驅	驂	驀	驃
7160	騾	驕	驍	驛	驗	驟	驢	驥	驤	驩	驫	驪	骭	骰	骼	髀
7170	髏	髑	髓	體	髞	髟	髢	髣	髦	髯	髫	髮	髴	髱	髷	
7220		髻	鬆	鬘	鬚	鬟	鬢	鬣	鬥	鬧	鬨	鬩	鬪	鬮	鬯	鬲
7230	魄	魃	魏	魍	魎	魑	魘	魴	鮓	鮃	鮑	鮖	鮗	鮟	鮠	鮨
7240	鮴	鯀	鯊	鮹	鯆	鯏	鯑	鯒	鯣	鯢	鯤	鯔	鯡	鰺	鯲	鯱
7250	鯰	鰕	鰔	鰉	鰓	鰌	鰆	鰈	鰒	鰊	鰄	鰮	鰛	鰥	鰤	鰡
7260	鰰	鱇	鰲	鱆	鰾	鱚	鱠	鱧	鱶	鱸	鳧	鳬	鳰	鴉	鴈	鳫
7270	鴃	鴆	鴪	鴦	鶯	鴣	鴟	鵄	鴕	鴒	鵁	鴿	鴾	鵆	鵈	
7320		鵝	鵞	鵤	鵑	鵐	鵙	鵲	鶉	鶇	鶫	鵯	鵺	鶚	鶤	鶩
7330	鶲	鷄	鷁	鶻	鶸	鶺	鷆	鷏	鷂	鷙	鷓	鷸	鷦	鷭	鷯	鷽
7340	鸚	鸛	鸞	鹵	鹹	鹽	麁	麈	麋	麌	麒	麕	麑	麝	麥	麩
7350	麸	麪	麭	靡	黌	黎	黏	黐	黔	黜	點	黝	黠	黥	黨	黯
7360	黴	黶	黷	黹	黻	黼	黽	鼇	鼈	皷	鼕	鼡	鼬	鼾	齊	齒
7370	齔	齣	齟	齠	齡	齦	齧	齬	齪	齷	齲	齶	龕	龜	龠	
7420		堯	槇	遙	瑤											

索引

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

X

Y

Z

あ

か

き

く

け

こ

さ

し

せ

そ

た

ち

て

と

に

は

ひ

ふ

へ

ま

め

も

ゆ

よ

れ

ろ