Rime_description.md

# Schema.yaml 詳解

## 開始之前

```
# Rime schema
# encoding: utf-8
```

## 描述檔

1. `name:` 方案的顯示名偁〔即出現於方案選單中以示人的，通常爲中文〕
2. `schema_id:` 方案內部名，在代碼中引用此方案時以此名爲正，通常由英文、數字、下劃線組成
3. `author:` 發明人、撰寫者。如果您對方案做出了修改，請保留原作者名，並將自己的名字加在後面
4. `description:` 請簡要描述方案歷史、碼表來源、該方案規則等
5. `dependencies:` 如果本方案依賴於其它方案〔通常來說會依賴其它方案做爲反查，抑或是兩種或多種方案混用時〕

6. `version:` 版本號，在發佈新版前請確保已陞版本號

   示例

```
schema:
  name: "蒼頡檢字法"
  schema_id: cangjie6
  author:
    - "發明人 朱邦復先生、沈紅蓮女士"
  dependencies:
    - luna_pinyin
    - jyutping
    - zyenpheng
  description: |
    第六代倉頡輸入法
    碼表由雪齋、惜緣和crazy4u整理
  version: 0.19
```

## 開關

通常包含以下五個：

1. `ascii_mode` 是中英文轉換開關。預設 `0` 爲英文， `1` 爲中文
2. `full_shape` 是全角符號 / 半角符號開關。注意，開啓全角時英文字母亦爲全角。 `0` 爲半角， `1` 爲全角
3. `extended_charset` 是字符集開關。 `0` 爲CJK基本字符集， `1` 爲CJK全字符集
   - 僅 `table_translator` 可用
4. `simplification` 是轉化字開關。一般情況下與上同， `0` 爲不開啓轉化， `1` 爲轉化。
5. `ascii_punct` 是中西文標點轉換開關， `0` 爲中文句讀， `1` 爲西文標點。

- 此選項名俱可自定義，亦可添加多套替換用字方案：

```
- name: zh_cn
  states: ["漢字", "汉字"]
  reset: 0
```

或

```
- options: [ zh_trad, zh_cn, zh_mars ]
  states:
    - 字型 → 漢字
    - 字型 → 汉字
    - 字型 → 蘿荸
  reset: 0
```

- `states:` 可不寫，如不寫則此開關存在但不可見，可由快捷鍵操作
- `reset:` 設定默認狀態〔 `reset` 可不寫，此時切換窗口時不會重置到默認狀態〕

示例

```
switches:
  - name: ascii_mode
    reset: 0
    states: ["中文", "西文"]
  - name: full_shape
    states: ["半角", "全角"]
  - name: extended_charset
    states: ["通用", "增廣"]
  - name: simplification
    states: ["漢字", "汉字"]
  - name: ascii_punct
    states: ["句讀", "符號"]
```

# 引擎

- 以下**加粗**項爲可細配者，*斜體*者爲不常用者

引擎分四組：

## 一、 processors

- 這批組件處理各類按鍵消息

1. `ascii_composer` 處理西文模式及中西文切
2. **`recognizer`** 與 `matcher` 搭配，處理符合特定規則的輸入碼，如網址、反查等 `tags`
3. **`key_binder`** 在特定條件下將按鍵綁定到其他按鍵，如重定義逗號、句號爲候選翻頁、開關快捷鍵等
4. **`speller`** 拼寫處理器，接受字符按鍵，編輯輸入
5. **`punctuator`** 句讀處理器，將單個字符按鍵直接映射爲標點符號或文字
6. `selector` 選字處理器，處理數字選字鍵〔可以換成別的哦〕、上、下候選定位、換頁
7. `navigator` 處理輸入欄內的光標移動
8. `express_editor` 編輯器，處理空格、回車上屏、回退鍵
9. *`fluid_editor`* 句式編輯器，用於以空格斷詞、回車上屏的【注音】、【語句流】等輸入方案，替換 `express_editor`
10. *`chord_composer`* 和絃作曲家或曰並擊處理器，用於【宮保拼音】等多鍵並擊的輸入方案

## 二、 segmentors

- 這批組件識別不同內容類型，將輸入碼分段並加上 `tag`

1. `ascii_segmentor` 標識西文段落〔譬如在西文模式下〕字母直接上屏
2. `matcher` 配合 `recognizer` 標識符合特定規則的段落，如網址、反查等，加上特定 `tag`
3. **`abc_segmentor`** 標識常規的文字段落，加上 `abc` 這個 `tag`
4. `punct_segmentor` 標識句讀段落〔鍵入標點符號用〕加上 `punct` 這個 `tag`
5. `fallback_segmentor` 標識其他未標識段落
6. **`affix_segmentor`** 用戶自定義 `tag`
    - 此項可加載多個實例，後接 `@` + `tag` 名

## 三、 `translators`

- 這批組件翻譯特定類型的編碼段爲一組候選文字

1. `echo_translator` 沒有其他候選字時，回顯輸入碼〔輸入碼可以 `Shift` + `Enter` 上屏〕
2. `punct_translator` 配合 `punct_segmentor` 轉換標點符號
3. **`table_translator`** 碼表翻譯器，用於倉頡、五筆等基於碼表的輸入方案
   - 此項可加載多個實例，後接 `@` +翻譯器名〔如： `cangjie` 、 `wubi` 等〕
4. **`script_translator`** 腳本翻譯器，用於拼音、粵拼等基於音節表的輸入方案
   - 此項可加載多個實例，後接 `@` +翻譯器名〔如： `pinyin` 、 `jyutping` 等〕
5. *`reverse_lookup_translator`* 反查翻譯器，用另一種編碼方案查碼

## 四、 `filters`

- 這批組件過濾翻譯的結果

1. **`simplifier`** 用字轉換
2. `uniquifier` 過濾重複的候選字，有可能來自 **`simplifier`**
3. `cjk_minifier` 字符集過濾〔用於 `script_translator` ，使之支援 `extended_charset` 開關〕
4. **`reverse_lookup_filter`** 反查濾鏡，以更靈活的方式反查，Rime1.0後替代 *`reverse_lookup_translator`*
   - 此項可加載多個實例，後接 `@` +濾鏡名〔如：`pinyin_lookup` 、 `jyutping_lookup` 等〕
5. **`single_char_filter`** 單字過濾器，如加載此組件，則屏敝詞典中的詞組〔僅 `table_translator` 有效〕

   示例

   cangjie6.schema.yaml

```
engine:
  processors:
    - ascii_composer
    - recognizer
    - key_binder
    - speller
    - punctuator
    - selector
    - navigator
    - express_editor
  segmentors:
    - ascii_segmentor
    - matcher
    - affix_segmentor@pinyin
    - affix_segmentor@jyutping
    - affix_segmentor@pinyin_lookup
    - affix_segmentor@jyutping_lookup
    - affix_segmentor@reverse_lookup
```

```
    - abc_segmentor
    - punct_segmentor
    - fallback_segmentor
  translators:
    - punct_translator
    - table_translator
    - script_translator@pinyin
    - script_translator@jyutping
    - script_translator@pinyin_lookup
    - script_translator@jyutping_lookup
  filters:
    - simplifier@zh_simp
    - uniquifier
    - cjk_minifier
    - reverse_lookup_filter@middle_chinese
    - reverse_lookup_filter@pinyin_reverse_lookup
    - reverse_lookup_filter@jyutping_reverse_lookup
```

---

# 細項配置

- 凡 `comment_format` 、 `preedit_format` 、 `speller/algebra` 所用之正則表達式，請參閱「Perl正則表達式」

引擎中所舉之加粗者均可在下方詳細描述，格式爲：

```
name:
  branches: configurations
```

或

```
name:
  branches:
    - configurations
```

## 一、 speller

1. alphabet: 定義本方案輸入鍵
2. initials: 定義僅作始碼之鍵
3. finals: 定義僅作末碼之鍵
4. delimiter: 上屏時的音節間分音符
5. algebra: 拼寫運算規則，由之算出的拼寫匯入 prism 中
6. max_code_length: 形碼最大碼長，超過則頂字上屏〔 number 〕
7. auto_select: 自動上屏〔 true 或 false 〕
8. auto_select_pattern: 自動上屏規則，以正則表達式描述，當輸入串可以被匹配時自動頂字上屏。
9. use_space: 以空格作輸入碼〔 true 或 false 〕

    - speller 的演算包含:

        ```
        xform --改寫〔不保留原形〕
        derive --衍生〔保留原形〕
        abbrev --簡拼〔出字優先級較上兩組更低〕
        fuzz --畧拼〔此種簡拼僅組詞，不出單字〕
        xlit --變換〔適合大量一對一變換〕
        erase --刪除
        ```

示例

luna_pinyin.schema.yaml

```
speller:
  alphabet: zyxwvutsrqponmlkjihgfedcba
  delimiter: " '"
```

```
    algebra:
      - erase/^xx$/
      - abbrev/^([a-z]).+$/$1/
      - abbrev/^([zcs]h).+$/$1/
      - derive/^([nl])ve$/$1ue/
      - derive/^([jqxy])u/$1v/
      - derive/un$/uen/
      - derive/ui$/uei/
      - derive/iu$/iou/
      - derive/([aeiou])ng$/$1gn/
      - derive/([dtngkhrzcs])o(u|ng)$/$1o/
      - derive/ong$/on/
      - derive/ao$/oa/
      - derive/([iu])a(o|ng?)$/a$1$2/
```

## 二、 segmentor

- segmentor 配合 recognizer 標記出 tag 。這裏會用到 affix_segmentor 和 abc_translator
- tag 用在 translator 、 reverse_lookup_filter 、 simplifier 中用以標定各自作用範圍
- 如果不需要用到 extra_tags 則不需要單獨配置 segmentor

1. tag: 設定其 tag
2. prefix: 設定其前綴標識，可不填，不填則無前綴
3. suffix: 設定其尾綴標識，可不填，不填則無尾綴
4. tips: 設定其輸入前提示符，可不填，不填則無提示符
5. closing_tips: 設定其結束輸入提示符，可不填，不填則無提示符
6. extra_tags: 爲此 segmentor 所標記的段落插上其它 tag

當 affix_segmentor 和 translator 重名時，兩者可併在一處配置，此處**1-5**條對應下面**19-23**條。 abc_segmentor 僅可設

```
extra_tags
```

示例

cangjie6.schema.yaml

```
reverse_lookup:
  tag: reverse_lookup
  prefix: "`"
  suffix: ";"
  tips: "【反查】"
  closing_tips: "【蒼頡】"
  extra_tags:
    - pinyin_lookup
    - jyutping_lookup
```

## 三、 translator

- 每個方案有一個主 `translator` ，在引擎列表中不以 `@` +翻譯器名定義，在細項配置時直接以 `translator:` 命名。以下加粗項爲可在主 `translator` 中定義之項，其它可在副〔以 `@` +翻譯器名命名〕 `translator` 中定義

1. **`enable_charset_filter:`** 是否開啓字符集過濾〔僅 `table_translator` 有效。啓用 `cjk_minifier` 後可適用於 `script_translator` 〕
2. **`enable_encoder:`** 是否開啓自動造詞〔僅 `table_translator` 有效〕
3. **`encode_commit_history:`** 是否對已上屏詞自動成詞〔僅 `table_translator` 有效〕
4. **`max_phrase_length:`** 最大自動成詞詞長〔僅 `table_translator` 有效〕

5. `enable_completion`：提前顯示尚未輸入完整碼的字〔僅 `table_translator` 有效〕
6. `sentence_over_completion`：在無全碼對應字而僅有逐鍵提示時也開啓智能組句〔僅 `table_translator` 有效〕
7. `strict_spelling`：配合 `speller` 中的 `fuzz` 規則，僅以畧拼碼組詞〔僅 `table_translator` 有效〕
8. `disable_user_dict_for_patterns`：禁止某些編碼錄入用戶詞典
9. `enable_sentence`：是否開啓自動造句
10. `enable_user_dict`：是否開啓用戶詞典〔用戶詞典記錄動態字詞頻、用戶詞〕
    - 以上選填 `true` 或 `false`
11. `dictionary`：翻譯器將調取此字典文件
12. `prism`：設定由此主翻譯器的 `speller` 生成的棱鏡文件名，或此副編譯器調用的棱鏡名
13. `user_dict`：設定用戶詞典名
14. `db_class`：設定用戶詞典類型，可設 `tabledb`〔文本〕或 `userdb`〔二進制〕
15. `preedit_format`：上屏碼自定義
16. `comment_format`：提示碼自定義
17. `spelling_hints`：設定多少字以內候選標註完整帶調拼音〔僅 `script_translator` 有效〕
18. `initial_quality`：設定此翻譯器出字優先級
19. `tag`：設定此翻譯器針對的 `tag`。可不填，不填則僅針對 `abc`
20. `prefix`：設定此翻譯器的前綴標識，可不填，不填則無前綴
21. `suffix`：設定此翻譯器的尾綴標識，可不填，不填則無尾綴
22. `tips`：設定此翻譯器的輸入前提示符，可不填，不填則無提示符

23. `closing_tips:` 設定此翻譯器的結束輸入提示符，可不填，不填則無提示符

示例

cangjie6.schema.yaml 蒼頡主翻譯器

```
translator:
  dictionary: cangjie6
  enable_charset_filter: true
  enable_sentence: true
  enable_encoder: true
  encode_commit_history: true
  max_phrase_length: 5
  preedit_format:
    - xform/^([a-z ])$/$1|\U$1\E/
    - xform/(?<=[a-z])\s(?=[a-z])//
    - "xlit|ABCDEFGHIJKLMNOPQRSTUVWXYZ|日月金木水火土竹z
  comment_format:
    - "xlit|abcdefghijklmnopqrstuvwxyz~|日月金木水火土竹
  disable_user_dict_for_patterns:
    - "^z.$"
  initial_quality: 0.75
```

---

cangjie6.schema.yaml 拼音副翻譯器

```
pinyin:
  tag: pinyin
  dictionary: luna_pinyin
  enable_charset_filter: true
  prefix: 'P' #須配合recognizer
  suffix: ';' #須配合recognizer
  preedit_format:
    - "xform/([nl])v/$1ü/"
    - "xform/([nl])ue/$1üe/"
    - "xform/([jqxy])v/$1u/"
```

```
  tips: "【漢拼】"
  closing_tips: "【蒼頡】"
```

pinyin_simp.schema.yaml 拼音・簡化字主翻譯器

```
translator:
  dictionary: luna_pinyin
  prism: luna_pinyin_simp
  preedit_format:
    - xform/([nl])v/$1ü/
    - xform/([nl])ue/$1üe/
    - xform/([jqxy])v/$1u/
```

luna_pinyin.schema.yaml 朙月拼音用戶短語

```
custom_phrase: #這是一個table_translator
  dictionary: ""
  user_dict: custom_phrase
  db_class: tabledb
  enable_sentence: false
  enable_completion: false
  initial_quality: 1
```

## 四、 reverse_lookup_filter

- 此濾鏡須掛在 translator 上，不影響該 translator 工作

1. tags: 設定其作用範圍
2. overwrite_comment: 是否覆蓋其他提示
3. dictionary: 反查所得提示碼之碼表
4. comment_format: 自定義提示碼格式

示例

cangjie6.schema.yaml

```
pinyin_reverse_lookup: #該反查濾鏡名
  tags: [ pinyin_lookup ] #掛在這個tag所對應的翻譯器上
  overwrite_comment: true
  dictionary: cangjie6 #反查所得爲蒼頡碼
  comment_format:
    - "xform/$/〕/"
    - "xform/^/〔/"
    - "xlit|abcdefghijklmnopqrstuvwxyz |日月金木水火土竹
```

---

## 五、 simplifier

1. `option_name:` 對應 `switches` 中設定的切換項名
2. `opencc_config:` 用字轉換配置文件
   - 位於： `rime_dir/opencc/` ，自帶之配置文件含：
     a. 繁轉簡〔默認〕： `t2s.json`
     b. 繁轉臺灣： `t2tw.json`
     c. 繁轉香港： `t2hk.json`
     d. 簡轉繁： `s2t.json`
3. `tags:` 設定轉換範圍
4. `tips:` 設定是否提示轉換前的字，可塡 `none` 〔或不塡〕、 `char` 〔僅對單字有效〕、 `all`
5. *`excluded_types:`* 取消特定範圍〔一般爲 *`reverse_lookup_translator`* 〕轉化用字

示例

修改自 luna_pinyin_kunki.schema

```
zh_tw:
  option_name: zh_tw
  opencc_config: t2tw.json
  tags: [ abc ] #abc對應abc_segmentor
  tips: none
```

## 六、 *chord_composer*

- 並擊把鍵盤分兩半，相當於兩塊鍵盤。兩邊同時擊鍵，系統默認在其中一半上按的鍵先於另一半，由此得出上屏碼

1. `alphabet:` 字母表，包含用於並擊的按鍵。擊鍵雖有先後，形成並擊時，一律以字母表順序排列
2. `algebra:` 拼寫運算規則，將一組並擊編碼轉換爲拼音音節
3. `output_format:` 並擊完成後套用的式樣，追加隔音符號
4. `prompt_format:` 並擊過程中套用的式樣，加方括弧

示例

combo_pinyin.schema.yaml

```
chord_composer:
  # 字母表，包含用於並擊的按鍵
  # 擊鍵雖有先後，形成並擊時，一律以字母表順序排列
  alphabet: "swxdecfrvgtbnjum ki,lo."
  # 拼寫運算規則，將一組並擊編碼轉換爲拼音音節
  algebra:
    # 先將物理按鍵字符對應到宮保拼音鍵位中的拼音字母
    - 'xlit|swxdecfrvgtbnjum ki,lo.|sczhlfgdbktpRiuVa
    # 以下根據宮保拼音的鍵位分別變換聲母、韻母部分
    # 組合聲母
    - xform/^zf/zh/
    - xform/^cl/ch/
    - xform/^fb/m/
```

```
    - xform/^ld/n/
    - xform/^hg/r/
    ……
    # 聲母獨用時補足隱含的韻母
    - xform/^([bpf])$/$1u/
    - xform/^([mdtnlgkh])$/$1e/
    - xform/^([mdtnlgkh])$/$1e/
    - xform/^([zcsr]h?)$/$1i/
  # 並擊完成後套用的式樣，追加隔音符號
  output_format:
    - "xform/^([a-z]+)$/$1'/"
  # 並擊過程中套用的式樣，加方括弧
  prompt_format:
    - "xform/^(.*)$/[$1]/"
```

---

## 七、其它

- 包括 `recognizer` 、 `key_binder` 、 `punctuator` 。**標點**、**快捷鍵**、**二三選重**、**特殊字符**等均於此設置

1. **`import_preset:`** 由外部統一文件導入
2. `recognizer:` 下設 `patterns:` 配合 `segmentor` 的 `prefix` 和 `suffix` 完成段落劃分、 `tag` 分配
   - `:` 前字段可以爲以 `affix_segmentor@someTag` 定義的 `Tag` 名，或者 `punct` 、 `reverse_lookup` 兩個內設的字段。其它字段不調用輸入法引擎，輸入即輸出〔如 `url` 等字段〕
3. `key_binder:` 下設 `bindings:` 設置功能性快捷鍵
   - 每一條 `binding` 可能包含： `accept` 實際所按之鍵、 `send` 輸出效果、 `toggle` 切換開關和 `when` 作用範圍〔 `send` 和 `toggle` 二選一〕
     - `toggle` 可用字段包含五個開關名
     - `when` 可用字段包含：

| | |
|---|---|
| paging | 翻篥用 |
| has_menu | 操作候選項用 |
| composing | 操作輸入碼用 |
| always | 全域 |

- accept 和 send 可用字段除A-Za-z0-9外，還包含以下鍵板上實際有的鍵：

| | |
|---|---|
| BackSpace | 退格 |
| Tab | 水平定位符 |
| Linefeed | 換行 |
| Clear | 清除 |
| Return | 回車 |
| Pause | 暫停 |
| Sys_Req | 印屏 |
| Escape | 退出 |
| Delete | 刪除 |
| Home | 原位 |
| Left | 左箭頭 |
| Up | 上箭頭 |
| Right | 右箭頭 |
| Down | 下箭頭 |
| Prior、Page_Up | 上翻 |
| Next、Page_Down | 下翻 |
| End | 末位 |
| Begin | 始位 |
| Shift_L | 左Shift |
| Shift_R | 右Shift |
| Control_L | 左Ctrl |
| Control_R | 右Ctrl |
| Meta_L | 左Meta |
| Meta_R | 右Meta |
| Alt_L | 左Alt |
| Alt_R | 右Alt |
| Super_L | 左Super |
| Super_R | 右Super |
| Hyper_L | 左Hyper |

```
Hyper_R 右Hyper
Caps_Lock       大寫鎖
Shift_Lock      上檔鎖
Scroll_Lock     滾動鎖
Num_Lock        小鍵板鎖
Select  選定
Print   列印
Execute 執行
Insert  插入
Undo    還原
Redo    重做
Menu    菜單
Find    蒐尋
Cancel  取消
Help    幫助
Break   中斷
space
exclam  !
quotedbl        "
numbersign      #
dollar  $
percent %
ampersand       &
apostrophe      '
parenleft       (
parenright      )
asterisk        *
plus    +
comma   ,
minus   -
period  .
slash   /
colon   :
semicolon       ;
less    <
equal   =
greater >
question        ?
at      @
```

```
bracketleft     [
backslash
bracketright    ]
asciicircum     ^
underscore      _
grave   `
braceleft       {
bar     |
braceright      }
asciitilde      ~
```

```
KP_Space        小鍵板空格
KP_Tab  小鍵板水平定位符
KP_Enter        小鍵板回車
KP_Delete       小鍵板刪除
KP_Home 小鍵板原位
KP_Left 小鍵板左箭頭
KP_Up   小鍵板上箭頭
KP_Right        小鍵板右箭頭
KP_Down 小鍵板下箭頭
KP_Prior、KP_Page_Up    小鍵板上翻
KP_Next、KP_Page_Down   小鍵板下翻
KP_End  小鍵板末位
KP_Begin        小鍵板始位
KP_Insert       小鍵板插入
KP_Equal        小鍵板等於
KP_Multiply     小鍵板乘號
KP_Add  小鍵板加號
KP_Subtract     小鍵板減號
KP_Divide       小鍵板除號
KP_Decimal      小鍵板小數點
KP_0    小鍵板0
KP_1    小鍵板1
KP_2    小鍵板2
KP_3    小鍵板3
KP_4    小鍵板4
KP_5    小鍵板5
KP_6    小鍵板6
KP_7    小鍵板7
```

| | |
|---|---|
| KP_8 | 小鍵板8 |
| KP_9 | 小鍵板9 |

4. editor 用以訂製操作鍵〔不支持 import_preset: 〕，鍵板鍵名同 key_binder/bindings 中的 accept 和 send，效果定義如下：

| | |
|---|---|
| confirm | 上屏候選項 |
| commit_comment | 上屏候選項備注 |
| commit_raw_input | 上屏原始輸入 |
| commit_script_text | 上屏變換後輸入 |
| commit_composition | 語句流單字上屏 |
| revert | 撤消上次輸入 |
| back | 按字符回退 |
| back_syllable | 按音節回退 |
| delete_candidate | 刪除候選項 |
| delete | 向後刪除 |
| cancel | 取消輸入 |
| noop | 空 |

5. punctuator: 下設 full_shape: 和 half_shape: 分別控制全角模式下的符號和半角模式下的符號，另有 use_space: 空格頂字〔 true 或 false 〕
   - 每條標點項可加 commit 直接上屏和 pair 交替上屏兩種模式，默認爲選單模式

示例

修改自 cangjie6.schema.yaml

```
key_binder:
  import_preset: default
  bindings:
```

```
    - {accept: semicolon, send: 2, when: has_menu} #分
    - {accept: apostrophe, send: 3, when: has_menu} #
    - {accept: "Control+1", select: .next, when: alwa
    - {accept: "Control+2", toggle: full_shape, when:
    - {accept: "Control+3", toggle: simplification, w
    - {accept: "Control+4", toggle: extended_charset,
editor:
bindings:
Return: commit_comment


punctuator:
import_preset: symbols
half_shape:
"'": {pair: ["「", "」"]} #第一次按是「，第二次是」
"(": ["〔", "［"] #彈出選單
.: {commit: "。"} #無選單，直接上屏。優先級最高


recognizer: import_preset: default patterns: email: "^[a-z]
[-_.0-9a-z]@.$" url: "^(www[.]|https?:|ftp:|mailto:).$"
reverse_lookup: "[a-z]*;?$" pinyin_lookup: "P[a-z];?$"
jyutping_lookup: "J[a-z]*;?$" pinyin: "(?&lt!)P[a-z';?$"
jyutping: "(?&lt!`)J[a-z';?$" punct: "/[a-z]*$" #配合
symbols.yaml中的特殊字符輸入
```

---

# 其它

- Rime還爲每個方案提供選單和一定的外觀訂製能力
- 通常情況下 `menu` 在 `default.yaml` 中定義〔或用戶修改檔 `default.custom.yaml` 〕， `style` 在 `squirrel.yaml` 或 `weasel.yaml` 〔或用戶修改檔 `squirrel.custom.yaml` 或 `weasel.custom.yaml` 〕

  示例

```
menu:
  alternative_select_keys: ASDFGHJKL #如編碼字符佔用數字
  page_size: 5 #選單每筆顯示個數
style: font_face: "HanaMinA, HanaMinB" #字體〔小狼毫得且僅得設一個字體；鼠鬚管得設多個字體，後面的字體自動補前面字體不含的字〕
font_point: 15 #字號 horizontal: false #橫 / 直排
line_spacing: 1 #行距 inline_preedit: true #輸入碼內嵌
```

---

# Dict.yaml 詳解

## 開始之前

```
# Rime dict
# encoding: utf-8
```
〔你還可以在這註釋字典來源、變動記錄等〕

## 描述檔

1. `name:` 內部字典名，也即 `schema` 所引用的字典名，確保與文件名相一致
2. `version:` 如果發佈，請確保每次改動陞版本號

   示例

```
name: "cangjie6.extended"
version: "0.1"
```

# 配置

1. `sort:` 字典**初始**排序，可選 `original` 或 `by_weight`
2. `use_preset_vocabulary:` 是否引入「八股文」〔含字詞頻、詞庫〕
3. `max_phrase_length:` 配合 `use_preset_vocabulary:`，設定導入詞條最大詞長
4. `min_phrase_weight:` 配合 `use_preset_vocabulary:`，設定導入詞條最小詞頻
5. `columns:` 定義碼表以 `Tab` 分隔出的各列，可設 `text` 【文本】、 `code` 【碼】、 `weight` 【權重】、 `stem` 【造詞碼】
6. `import_tables:` 加載其它外部碼表
7. `encoder:` 形碼造詞規則
    i. `exclude_patterns:`
    ii. `rules:` 可用 `length_equal:` 和 `length_in_range:` 定義。大寫字母表示字序，小寫字母表示其所跟隨的大寫字母所以表的字中的編碼序
    iii. `tail_anchor:` 造詞碼包含結構分割符〔僅用於倉頡〕
    iv. `exclude_patterns` 取消某編碼的造詞資格

示例

cangjie6.extended.dict.yaml

```
sort: by_weight
use_preset_vocabulary: false
```

```
import_tables:
  - cangjie6 #單字碼表由cangjie6.dict.yaml導入
columns: #此字典爲純詞典，無單字編碼，僅有字和詞頻
  - text #字 / 詞
  - weight #字 / 詞頻
encoder:
  exclude_patterns:
    - '^z.*$'
  rules:
    - length_equal: 2 #對於二字詞
      formula: "AaAzBaBbBz" #取第一字首尾碼、第二字首次尾碼
    - length_equal: 3 #對於三字詞
      formula: "AaAzBaYzZz" #取第一字首尾碼、第二字首尾碼、
    - length_in_range: [4, 5] #對於四至五字詞
      formula: "AaBzCaYzZz" #取第一字首碼，第二字尾碼、第三
  tail_anchor: "'"
```

---

## 碼表

- 以 `Tab` 分隔各列，各列依 `columns:` 定義排列。

  示例

  cangjie6.dict.yaml

  ```
  columns:
    - text #第一列字 / 詞
    - code #第二列碼
    - weight #第三列字 / 詞頻
    - stem #第四列造詞碼
  ```

  cangjie6.dict.yaml

| | | | |
|---|---|---|---|
| 個 | owjr | 246268 | ow'jr |
| 看 | hqbu | 245668 | |
| 中 | l | 243881 | |
| 呢 | rsp | 242970 | |
| 來 | doo | 235101 | |
| 嗎 | rsqf | 221092 | |
| 爲 | bhnf | 211340 | |
| 會 | owfa | 209844 | |
| 她 | vpd | 204725 | |
| 與 | xyc | 203975 | |
| 給 | vfor | 193007 | |
| 等 | hgdi | 183340 | |
| 這 | yymr | 181787 | |
| 用 | bq | 168934 | b'q |

雪齋

09-Nov-2013