**必知必会**

建议您在定制 Rime 输入法之前了解 Rime 输入方案的概念等基础知识。

[必知必会](http://code.google.com/p/rimeime/wiki/RimeWithSchemata#%E5%BF%85%E7%9F%A5%E5%BF%85%E6%9C%83)

[Rime 中的数据文件分布及作用](http://code.google.com/p/rimeime/wiki/RimeWithSchemata#Rime_%E4%B8%AD%E7%9A%84%E6%95%B8%E6%93%9A%E6%96%87%E4%BB%B6%E5%88%86%E4%BD%88%E5%8F%8A%E4%BD%9C%E7%94%A8)

**重新布署的操作方法**

* 【中州韵】点击输入法状态区上的 ⟲ (Deploy) 按钮

或：如果找不到状态区，在终端输入以下命令，可触发自动部署：

    rm ~/.config/ibus/rime/default.yaml; ibus-daemon -drx

* 【小狼毫】开始菜单→小狼毫输入法→重新布署；当开启托盘图标时，右键点选「重新布署」
* 【鼠须管】在系统语言文字选单中选择「重新布署」

对设置的修改于[重新布署](http://code.google.com/p/rimeime/wiki/CustomizationGuide" \l "%E9%87%8D%E6%96%B0%E4%BD%88%E7%BD%B2%E7%9A%84%E6%93%8D%E4%BD%9C%E6%96%B9%E6%B3%95)后生效。编译新的输入方案需要一段时间，此间若无法输出中文，请稍等片刻。

若部署完毕后，可以通过 Ctrl+` 唤出方案选单，输入方案却仍无法正常使用，可能是输入方案未部署成功。请[查看日志文件](http://code.google.com/p/rimeime/wiki/RimeWithSchemata" \l "%E9%97%9C%E6%96%BC%E8%AA%BF%E8%A9%A6)定位错误。

**查阅 DIY 处方集**

已将一些定制 Rime 的常见问题、解法及定制档链接俱收录于[〔DIY 处方集〕](http://code.google.com/p/rimeime/wiki/CustomizationGuide" \l "DIY_%E8%99%95%E6%96%B9%E9%9B%86)

**设定项速查手册**

[雪斋的文档](https://github.com/LEOYoon-Tsaw/Rime_collections/blob/master/Rime_discription.md) 全面而详细解释了输入方案及词典中各设定项的含义及用法。

**定制指南**

Rime 输入方案，将 Rime 输入法的设定整理成完善的、可分发的形式。 但并非一定要创作新的输入方案，才可以改变 Rime 的行为。

当用户需要对 Rime 中的各种设定做小幅的调节，最直接、但不完全正确的做法是：编辑用户文件夹中那些 .yaml 文檔。

这一方法有弊端：

* 当 Rime 软件升级时，也会升级各种配置文件、预设输入方案。用户编辑过的文文件会被覆写为更高版本，所做调整也便丢失了。
* 即使在软件升级后再手动恢复经过编辑的文件，也会因配置文件的其他部分未得到更新而失去本次升级新增和修复的功能。

因此，对于随 Rime 发行的配置文件及预设输入方案，推荐的定制方法是：

创建一个文件名的主体部份（「.」之前）与要定制的文件相同、次级扩展名（「.yaml」之前）为 .custom 的定制文档：

patch:  
  "一级设定项/二级设定项/三级设定项": 新的设定值  
  "另一个设定项": 新的设定值  
  "再一个设定项": 新的设定值

就是这样：patch 定义了一组「补靪」，以源文件中的设定为基础，写入新的设定项、或以新的设定值取代现有设定项的值。

不懂？那看我来示范。

**一例、定制每页候选数**

Rime 中，默认每页至多显示 5 个候选项，而允许的范围是 1〜9（个别 Rime 发行版可支持10个候选）。

设定每页候选个数的默认值为 9，在用户目录建立文文件 default.custom.yaml ：

patch:  
  "menu/page\_size": 9

**重新布署** 即可生效。

〔注意〕如果 default.custom.yaml 里面已经有其他设定内容，只要以相同的缩进方式添加 patch: 以下的部分，不可重复 patch: 这一行。

若只需要将独孤一个输入方案的每页候选数设为 9，以【朙月拼音】为例，建立文档 luna\_pinyin.custom.yaml 写入相同内容，**重新布署** 即可生效。

注：[重新布署的操作方法★](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E9%87%8D%E6%96%B0%E4%BD%88%E7%BD%B2%E7%9A%84%E6%93%8D%E4%BD%9C%E6%96%B9%E6%B3%95)

**一例、定制标点符号**

有的用家习惯以 / 键输入标点「、」。

仍以【朙月拼音】为例，输入方案中有以下设定：

# luna\_pinyin.schema.yaml  
# ...  
  
punctuator:  
  import\_preset: default

解释：

punctuator 是 Rime 中负责转换标点符号的组件。该组件会从设定中读取符号映像表，而知道该做哪些转换。

punctuator/import\_preset 是说，本方案要继承一组默认的符号映像表、要从另一个配置文件 default.yaml 导入。

查看 default.yaml ，确有如下符号表：

punctuator:  
  full\_shape:  
    # ……其他……  
    "/" : [ ／, "/", ÷ ]  
    # ……其他……  
  half\_shape:  
    # ……其他……  
    "/" : [ "/", ／, ÷ ]  
    # ……其他……

可见按键 / 是被指定到 "/", ／, ÷ 等一组符号了。 并且全角和半角状态下，符号有不同的定义。

欲令 / 键直接输出「、」，可如此定制 luna\_pinyin.custom.yaml:

patch:  
  punctuator/full\_shape:  
    "/" : "、"  
  punctuator/half\_shape:  
    "/" : "、"

以上在输入方案设定中写入两组新值，合并后的输入方案成为：

# luna\_pinyin.schema.yaml  
# ...  
  
punctuator:  
  import\_preset: default  
  full\_shape:  
    "/" : "、"  
  half\_shape:  
    "/" : "、"

含义是、在由 default 导入的符号表之上，覆写对按键 / 的定义。

通过这种方法，既直接继承了大多数符号的默认定义，又做到了局部的个性化。

**使用全套西文标点**

有些用户习惯在中文里使用ASCII标点，那么与其一个一个覆写，不如 **整套都换掉** 。

取得这份配置文件—— [Rime 别样设定，使用西文标点](http://gist.github.com/2334409) 在用户文件夹保存为 alternative.yaml ；

再将输入方案中的「导入 default 设定」通过打 patch 替换为「导入 alternative 设定」

# luna\_pinyin.custom.yaml  
  
patch:  
  'punctuator/import\_preset': alternative

就换上了自己习惯的一套标点！

**一例、定制简化字输出**

**注意：**

* 如果您只是需要 Rime 输出简化字，敲 Ctrl+` 组合键、从菜单中选择「汉字→汉字」即可！
* 本例说明了其中原理，以及通过配置文件修改默认输出字形的方法。

Rime 预设的词汇表使用传统汉字。 这是因为传统汉字较简化字提供了更多信息，做「繁→简」转换能够保证较高的精度。

Rime 中的过滤器组件 simplifier，完成对候选词的繁简转换。

# luna\_pinyin.schema.yaml  
# ...  
  
switches:  
  - name: ascii\_mode  
    reset: 0  
    states: [ 中文, 西文 ]  
  - name: full\_shape  
    states: [ 半角, 全角 ]  
  - name: simplification    # 转换开关  
    states: [ 汉字, 汉字 ]  
  
engine:  
  filters:  
    - simplifier  # 必要组件一  
    - uniquifier  # 必要组件二

以上是【朙月拼音】中有关繁简转换功能的设定。

在 engine/filters 中，除了 simplifier，还用了一件 uniquifier。 这是因为有些时候，不同的候选会转化为相同的简化字，例如「钟→钟」、「钟→钟」。 uniquifier 的作用是在 simplifier 执行转换之后，将文字相同的候选项合并。

该输入方案设有三个状态开关：中／西文、全／半角、繁简字。即 switches 之下三项。

每个开关可在两种状态（states）之间切换，simplifier 依据名为 simplification 的开关状态来决定是否做简化：

* 初始状态下、输出为传统漢字、〔方案选单〕中的开关选项显示为「漢字→汉字」。
* 选择该项后、输出为简化汉字、〔方案选单〕中显示「汉字→漢字」。
* Rime 会记忆您的选择，下次打开输入法时、直接切换到所选的字形。
* 亦可无视上次记住的选择，在方案中重设初始值：reset 设为 0 或 1，分别选中 states 列表中的两种状态。

如果日常应用以简化字为主:-(，则每每在〔方案选单〕中切换十分不便； 于是佛振献上默认输出简化字的配置文件：

# luna\_pinyin.custom.yaml  
  
patch:  
  switches:                   # 注意缩进  
    - name: ascii\_mode  
      reset: 0                # reset 0 的作用是当从其他输入方案切换到本方案时，  
      states: [ 中文, 西文 ]  # 重设为指定的状态，而不保留在前一个方案中设定的状态。  
    - name: full\_shape        # 选择输入方案后通常需要立即输入中文，故重设 ascii\_mode = 0；  
      states: [ 半角, 全角 ]  # 而全／半角则可沿用之前方案中的用法。  
    - name: simplification  
      reset: 1                # 增加这一行：默认启用「繁→简」转换。  
      states: [ 汉字, 汉字 ]

其实预设输入方案中就提供了一套【朙月拼音】的简化字版本，名为【简化字】，以应大家“填表”之需。 看他的代码如何却与上篇定制档写得不同：

# luna\_pinyin\_simp.schema.yaml  
# ...  
  
switches:  
  - name: ascii\_mode  
    reset: 0  
    states: [ 中文, 西文 ]  
  - name: full\_shape  
    states: [ 半角, 全角 ]  
  - name: zh\_simp           # 注意这里（※1）  
    reset: 1  
    states: [ 汉字, 汉字 ]  
  
simplifier:  
  option\_name: zh\_simp      # 和这里（※2）

前文说，simplifier 这个组件会检查名为 simplification 的开关状态； 而这款【简化字】方案却用了一个不同名的开关 zh\_simp，即 ※1 处所示； 并通过在 ※2 行设定 simplifier/option\_name 告知 simplifier 组件所需关注的开关名字。

何故？

还记得否，前文对「全／半角」这个开关的讨论—— 当切换方案时，未明确使用 reset 重置的开关，会保持之前设定过的状态。

【朙月拼音】等多数方案，并未重设 simplification 这个选项—— 因为用户换了一种输入编码的方式、并不意味着需要变更输出的字形。

而【简化字】这一方案不同，恰恰是表达变更输出字形的需求； 用户再从【简化字】切回【朙月拼音】时，一定是为了「回到」繁体输出模式。 所以令【简化字】使用独立命名的开关、而非方案间共享的 simplification 开关， 以避免影响其他输入方案的繁简转换状态。

**一例、定制方案选单**

在【小狼毫】方案选单设定接口上勾勾选选，就可以如此定制输入方案列表：

# default.custom.yaml  
  
patch:  
  schema\_list:  # 对于列表类型，现在无有办法指定如何添加、消除或单一修改某项，于是要在定制档中将整个列表替换！  
    - schema: luna\_pinyin  
    - schema: cangjie5  
    - schema: luna\_pinyin\_fluency  
    - schema: luna\_pinyin\_simp  
    - schema: my\_coolest\_ever\_schema  # 这样就启用了未曾有过的高级输入方案！其实这么好的方案应该排在最前面哈。

无有设定接口时，又想启用、禁用某个输入方案，手写这样一份定制档、[重新布署](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E9%87%8D%E6%96%B0%E4%BD%88%E7%BD%B2%E7%9A%84%E6%93%8D%E4%BD%9C%E6%96%B9%E6%B3%95)就好啦。

**一例、定制唤出方案选单的快捷键**

唤出方案选单，当然要用键盘。默认的快捷键为 Ctrl+` 或 F4。

不过，有些同学计算机上 Ctrl+` 与其他软件冲突，F4 甚至本文写作时在【鼠须管】中还不可用。又或者有的玩家切换频繁，想定义到更好的键位。

那么……

# default.custom.yaml  
  
patch:  
  "switcher/hotkeys":  # 这个列表里每项定义一个快捷键，使哪个都中  
    - "Control+s"      # 添加 Ctrl+s  
    - "Control+grave"  # 你看写法并不是 Ctrl+` 而是与 IBus 一致的表示法  
    - F4

按键定义的格式为「修饰符甲+修饰符乙+…+按键名称」，加号为分隔符，要写出。

所谓修饰符，就是以下组合键的状态标志或是按键弹起的标志：

* Release——按键被放开，而不是按下
* Shift
* Control
* Alt——Windows上 Alt+字母 会被系统优先识别为程序菜单项的快捷键，当然 Alt+Tab 也不可用
* 嗯，Linux 发行版还支持 Super, Meta 等组合键，不过最好选每个平台都能用的啦

按键的名称，大小写字母和数字都用他们自己表示，其他的按键名称参考这里 [<X11/keysymdef.h>](http://rimeime.googlecode.com/svn/trunk/librime/thirdparty/include/X11/keysymdef.h) 的定义，去除代码前缀 XK\_ 即是。

**一例、定制【小狼毫】字体字号**

虽与输入方案无关，也在此列出以作参考。

# weasel.custom.yaml  
  
patch:  
  "style/font\_face": "明兰"  # 字体名称，从记事本等处的系统字体对话框里能看到  
  "style/font\_point": 14     # 字号，只认数字的，不认「五号」、「小五」这样的

**一例、定制【小狼毫】配色方案**

注：这款配色已经在新版本的小狼毫里预设了，做练习时，你可以将文中 starcraft 换成自己命名的标识。

# weasel.custom.yaml  
  
patch:  
  "style/color\_scheme": starcraft    # 这项用于选中下面定义的新方案  
  "preset\_color\_schemes/starcraft":  # 在配色方案列表里加入标识为 starcraft 的新方案  
    name: 星际我争霸／StarCraft  
    author: Contralisk <contralisk@gmail.com>, original artwork by Blizzard Entertainment  
    text\_color: 0xccaa88             # 编码行文字颜色，24位色值，用十六进制书写方便些，顺序是蓝绿红0xBBGGRR  
    candidate\_text\_color: 0x30bb55   # 候选项文字颜色，当与文字颜色不同时指定  
    back\_color: 0x000000             # 底色  
    border\_color: 0x1010a0           # 边框颜色，与底色相同则为无边框的效果  
    hilited\_text\_color: 0xfecb96     # 高亮文字，即与当前高亮候选对应的那部份输入码  
    hilited\_back\_color: 0x000000     # 设定高亮文字的底色，可起到凸显高亮部份的作用  
    hilited\_candidate\_text\_color: 0x60ffa8  # 高亮候选项的文字颜色，要醒目！  
    hilited\_candidate\_back\_color: 0x000000  # 高亮候选项的底色，若与背景色不同就会显出光棒

效果自己看！

**DIY 处方集**

已将一些定制 Rime 的常见问题、解法及定制档链接收录于此。

建议您首先读完《定制指南》、通晓相关原理，以正确运用这些处方。

**初始设定**

**在方案选单中添加五笔、双拼**

<https://gist.github.com/2309739>

仿此例，可启用任一预设或自定义输入方案，如【粤拼】、【注音】等。（[详解](http://code.google.com/p/rimeime/wiki/CustomizationGuide" \l "%E4%B8%80%E4%BE%8B%E3%80%81%E5%AE%9A%E8%A3%BD%E6%96%B9%E6%A1%88%E9%81%B8%E5%96%AE)）

如果下载、自己制作了非预设的输入方案，将源文件复制到「用户文件夹」后，也用上面的方法将方案标识加入选单！

修改于[重新布署](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E9%87%8D%E6%96%B0%E4%BD%88%E7%BD%B2%E7%9A%84%E6%93%8D%E4%BD%9C%E6%96%B9%E6%B3%95)后生效。

**【小狼毫】外观设定**

使用横向候选栏、嵌入式编码行：

# weasel.custom.yaml  
patch:  
  style/horizontal: true      # 候选横排  
  style/inline\_preedit: true  # 内嵌编码（仅支持TSF）  
  style/display\_tray\_icon: true  # 显示托盘图标

[设定字体字号](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E4%B8%80%E4%BE%8B%E3%80%81%E5%AE%9A%E8%A3%BD%E3%80%90%E5%B0%8F%E7%8B%BC%E6%AF%AB%E3%80%91%E5%AD%97%E9%AB%94%E5%AD%97%E8%99%9F) [制作配色方案](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E4%B8%80%E4%BE%8B%E3%80%81%E5%AE%9A%E8%A3%BD%E3%80%90%E5%B0%8F%E7%8B%BC%E6%AF%AB%E3%80%91%E9%85%8D%E8%89%B2%E6%96%B9%E6%A1%88)

**【鼠须管】外观与键盘设定**

<https://gist.github.com/2290714>

〔注意〕鼠须管从 0.9.6 版本开始支持选择配色方案，并改用squirrel.custom.yaml 保存用户的设定。

[更换图标](http://lixiaolai.com/blog/2012/10/22/rime-mac-shang-de-shen-ji-shu-ru-fa/)

鼠须管的图标比较难看，我随便在网上找了一个适合放在菜单条上的黑白图标，用来替换显示在状态栏的输入法图标。

网友帮忙制作了几款不同式样的图标，有模仿系统自带输入法图标样式的，有换成其他图案的，有对原图标加工美化的，收集于此：

* <https://github.com/lotem/rime-artworks/tree/master/user-contributed-icons>
* <https://github.com/lotem/rime-artworks/blob/master/rime-logo-2010/zhung-original-16x16.tiff> （旧版鼠须管图标，重命名为 rime.icns）
* <https://github.com/lotem/rime-artworks/blob/master/rime-logo-2012/squirrel-icon-retina.icns> （2012版鼠须管图标）

用法是，将图标重名名为 rime.icns，替换 "/Library/Input Methods/Squirrel.app/Contents/Resources/rime.icns"

**在特定程序里关闭中文输入**

【鼠须管】0.9.9 开始支持这项设定：

在指定的应用程序中，改变输入法的初始转换状态。如在

* 终端 Terminal / iTerm
* 代码编辑器 MacVim
* 快速启动工具 QuickSilver / Alfred

等程序里很少需要输入中文，于是鼠须管在这些程序里默认不开启中文输入。

自定义 Mac 应用程序的初始转换状态，首先查看应用的 Info.plist 文件得到 该应用的 Bundle Identifier，通常是形如 com.apple.Xcode 的字符串。

例如，要在 Xcode 里面默认关闭中文输入，又要在 Alfred 里面恢复开启中文输入，可如此设定：

# example squirrel.custom.yaml  
patch:  
  app\_options/com.apple.Xcode:  
    ascii\_mode: true  
  app\_options/com.alfredapp.Alfred: {}

注：一些版本的 Xcode 标识为 com.apple.dt.Xcode，请注意查看 Info.plist。

【小狼毫】 0.9.16 亦开始支持这项设定。

例如，要在 gVim 里面默认关闭中文输入，可如此设定：

# example weasel.custom.yaml  
patch:  
  app\_options/gvim.exe:  # 程序名字全用小写字母  
    ascii\_mode: true

**输入习惯**

**使用Control键切换中西文**

<https://gist.github.com/2981316>

以及修改Caps Lock、左右Shift、左右Control键的行为，提供三种切换方式。 详见代码注释。

**方便地输入含数字的西文用户名**

通常，输入以小写拉丁字母组成的编码后，数字键的作用是选择相应序号的候选字。

假设我的邮箱地址是 rime123@company.com，则需要在输入rime之后上屏或做临时中西文切换，方可输入数字部分。

为了更方便输入我的用户名 rime123，设置一组特例，将 rime 与其后的数字优先识别西文：

<https://gist.github.com/3076166>

**以方括号键换页**

<https://gist.github.com/2316704>

添加 Mac 风格的翻页键 [ ] 。这是比较直接的设定方式。下一则示例给出了一种更系统、可重用的设定方式。

**使用西文标点兼以方括号键换页**

<https://gist.github.com/2334409>

详见上文 [使用全套西文标点](http://code.google.com/p/rimeime/wiki/CustomizationGuide#%E4%BD%BF%E7%94%A8%E5%85%A8%E5%A5%97%E8%A5%BF%E6%96%87%E6%A8%99%E9%BB%9E) 一节。

**以回车键清除编码兼以分号、单引号选字**

<https://gist.github.com/2390510>

适合一些形码输入法（如五笔、郑码）的快手。

**关闭逐键提示**

table\_translator 默认开启逐键提示。若要只出精确匹配输入码的候选字，可关闭这一选项。

以【仓颉五代】为例：

# cangjie5.custom.yaml  
patch:  
  translator/enable\_completion: false

**关闭用户词典和字频调整**

以【五笔86】为例：

# wubi86.custom.yaml  
patch:  
  translator/enable\_user\_dict: false

**关闭码表输入法连打**

注：这个选项仅针对 table\_translator，用于屏蔽仓颉、五笔中带有太极图章「☯」的连打词句选项，不可作用于拼音、注音、速成等输入方案。

以【仓颉】为例：

# cangjie5.custom.yaml  
patch:  
  translator/enable\_sentence: false

**关闭仓颉与拼音混打**

默认，给出仓颉与拼音候选的混合列表。

如此设定，直接敲字母只认作仓颉码，但仍可在敲 ` 之后输入拼音：

# cangjie5.custom.yaml  
patch:  
  abc\_segmentor/extra\_tags: {}

**空码时按空格键清空输入码**

首先需要关闭码表输入法连打（参见上文），这样才可以在打空时不出候选词。

然后设定（以五笔86为例）：

# wubi86.custom.yaml  
patch:  
  translator/enable\_sentence: false  
  key\_binder/bindings:  
    - {when: has\_menu, accept: space, send: space}  
    - {when: composing, accept: space, send: Escape}

**模糊音**

**【朙月拼音】模糊音定制模板**

<https://gist.github.com/2320943>

【明月拼音・简化字／台湾正体／语句流】也适用， 只须将模板保存到 luna\_pinyin\_simp.custom.yaml 、 luna\_pinyin\_tw.custom.yaml 或luna\_pinyin\_fluency.custom.yaml 。

对比模糊音定制模板与[【朙月拼音】方案原件](https://github.com/lotem/brise/blob/master/preset/luna_pinyin.schema.yaml)， 可见模板的做法是，在 speller/algebra 原有的规则中插入了一些定义模糊音的代码行。

类似方案如双拼、粤拼等可参考模板演示的方法改写 speller/algebra 。

**【吴语】模糊音定制模板**

<https://gist.github.com/2015335>

**编码反查**

**设定【速成】的反查码为粤拼**

<https://gist.github.com/2944320>

**设定【仓颉】的反查码为双拼**

<https://gist.github.com/2944319>

**在Mac系统上输入emoji表情**

参考 <https://gist.github.com/2309739> 把 emoji 加入输入方案选单；

切换到 emoji 输入方案，即可通过拼音代码输入图释。[查看符号表](http://rimeime.googlecode.com/svn/trunk/artworks/schema/emoji-chart.png)

输入 all 可以列出全部符号，符号后面的括号里标记其拼音代码。

若要直接在【朙月拼音】里输入图释，请按此文设定：

<http://gist.github.com/3705586>

**五笔简入繁出**

【小狼毫】用家请到[下载页](http://code.google.com/p/rimeime/wiki/Downloads)取得「扩展方案集」。

安装完成后，执行输入法设定，添加【五笔・简入繁出】输入方案。

其他版本请参考这篇说明：

<https://gist.github.com/3467172>

**修正不对称繁简字**

繁→简实时转换比简体转繁体要轻松许多，却也免不了个别的错误。

比如这一例，「干」字是一繁对多简的典型。由它组成的常用词组，opencc 都做了仔细分辨。但是遇到较生僻的词组、专名，就比较头疼：

<http://tieba.baidu.com/p/1909252328>

**活用标点创建自定义词组**

在【朙月拼音】里添加一些自定义文字、符号。可以按照上文设定「emoji表情」的方式为自定义词组创建一个专门的词典。

可是建立词典稍显繁琐，而活用自定义标点，不失为一个便捷的方法：

# luna\_pinyin.custom.yaml  
# 如果不需要 ` 键的仓颉反查拼音功能，则可利用 ` 键输入自定义词组  
patch:  
  recognizer/patterns/reverse\_lookup:  
  'punctuator/half\_shape/`':  
    - '佛振 <chen.sst@gmail.com>'  
    - 'http://code.google.com/p/rimeime/'  
    - 上天赋予你高的智商，教你用到有用的地方。

上例 recognizer/patterns/reverse\_lookup: 作用是关闭`键的反查功能。若选用其他符号则不需要这行。又一例：

patch:  
  'punctuator/half\_shape/+': '+\_+'

'punctuator/half\_shape/+' 因为字符串包含符号，最好用 **单引号** 括起来，避免符号的转义问题。

重定义「/」这个符号，无法用上面演示的路径连写方式，那就分开写：

patch:  
  punctuator/half\_shape:  
   '/': [ '/', '/hello', '/bye', '/\* TODO \*/' ]  
   '+': '+\_+'