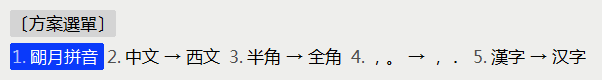
**序言**

很喜欢小狼毫输入法，喜欢他的简洁，美观以及超强悍的个人定制功能。关于 RIME输入法。大家都已经争执许多了，没有用户交互界面，导致许多想要使用 RIME 输入法的人最终放弃。吧里有人说， RIME 输入法只是用给程序使用的，因为它实在是太折腾了，貌似只有程序员才能使用。抑或者如 RIME 输入法是符合我们对什么的控制等等。RIME 输入法只是一款软件，因为它的种种优点我们使用它。我个人选择它的第一个原因就是它足够简洁。对于它的个人定制功能，我也只是在有限能力下对它做些修改，当然还是为了简洁。我个人认为，RIME 输入法并不是针对某个人群的，它只是按作者意图创造出来的一款软件，恰巧，这些意图也是我们想要的，我们从不同的角度去接受它，使用它。或是为了精简，或是为了个人定制，但没有要求你必须是什么什么人。限制 RIME 的不是 RIME 本身，而是我们自己的主观想法，想将它局限于电脑高手→\_→。

1. **繁简转换**

这是我们使用 RIME 碰上的第一个问题。设置方法：在已启动 RIME条件下，按Ctrl+grave（Tab上面那个键）进行方案选择



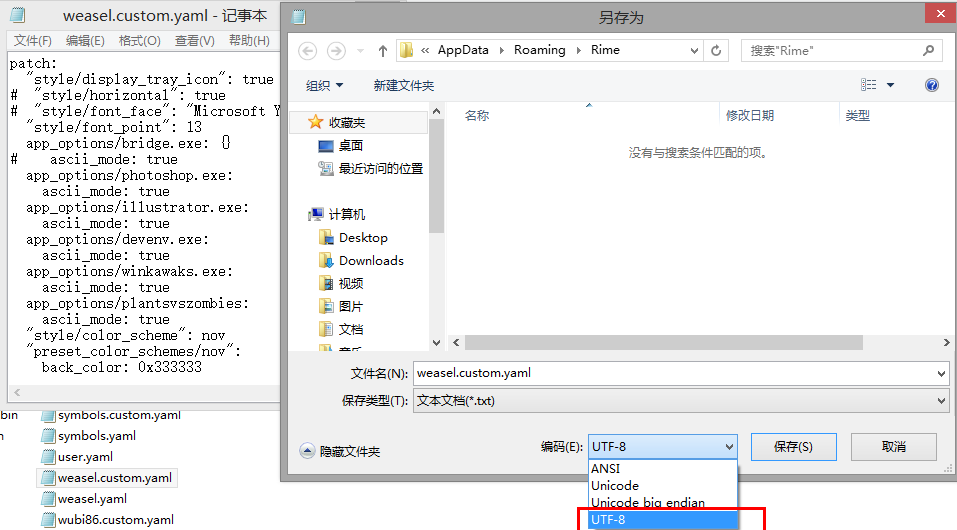
如图，直接按数字 5 就可以了。

1. **RIME 常规定制** （针对文件 weasel.yaml）
2. 托盘图标的显示左面红框圈住那个而不是右边篮框那个。

修改方法：

第一步：在“用户文件夹”里新建文件一文本文档，重命名为weasel.custom.yaml（注意扩展名也是要修改的，另外还要是UTF-8编码，如果不知道怎么做，就直接复制weasel.yaml文件，将里面的代码删除后进行自己的编辑

UTF-8如下图

）

三点解释：

* 1. 我们不直接修改weasel.yaml原因是wealsel.yaml文件是 RIME软件自解压出来的文件，在以后RIME 更新时，weasel.yaml将被重置为默认设置，那样你又要重新修改weasel.yaml文件了。而新建的weasel.custom.yaml是你自己建立的，不会修改。更新后只需点点击“重新部署”即可。
  2. weasel.custom.yaml文件会根据你的命令修改weasel.yaml，所以，其实最终修改的还是weasel.yaml文件。
  3. 重新部署命令就是 RIME 软件接受你所修改的命令，将命令安排部署下去。

重新部署命令在“开始→所有程序→小狼毫输入法”处可找到。

第二步：打开weasel.custom.yaml文件，写下如下命令

patch:

"style/display\_tray\_icon": true

注：patch即为补丁的意思，也就是为weasel.yaml文件打补丁。里面几个英文也好理解：style－样式风格，display－显示、展示，tray－盘、托盘，icon－图标。

另外patch顶格写，"style/display\_tray\_icon": true 是要空两格的。所有冒号后面均有一个空格（以下代码均需注意空格，只有patch是顶个写，其它或空两格或空四格）。

第三步：重修部署（在下面每步修改完代码并保存之后你都要重新部署，此部只说这一次，以下从略）

注意，在托盘图标显示后，可直接右击托盘图标对RIME进行一些基本操作



以后可在此处直接实现重新部署命令。

1. 修改横竖排显示后选项（如图第一张），显示字体和显示字号，命令如下

"style/horizontal": true #横竖排

"style/font\_face": "Microsoft YaHei Mono" #微软雅黑

"style/font\_point": 13 #字号13

注意：其中#字符只是一个注释说明的作用，#本身与后面的就是文字可同时去掉。

还有就是空格，与上文中的"style/display\_tray\_icon": true代码一样，需空两格。冒号后面即icon前面空一格。此处只这里说，以下从略。

1. 对特定程序程序指定默认输入法。

说明：在我们快乐的玩耍游戏是我是不是不希望出现这种情况的。

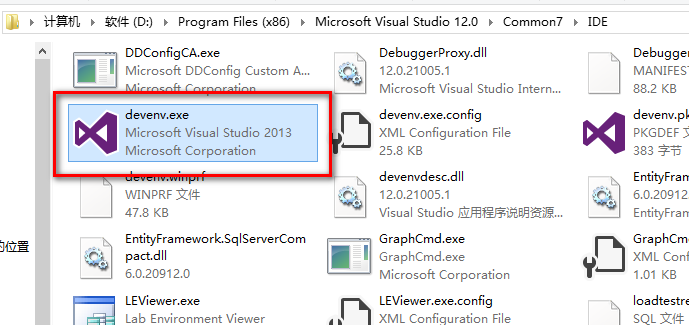
我按q键是想进入游戏，而不是打字，这种情况很恼人的。聪明的小狼毫可以设定即便在开启中文输入状态下，一旦进入特定程序也可以转变为英文输入状态（英文输入状态实际上就是关闭了输入法）。

So，我们要这样做，再上一条命令下继续写命令如下

app\_options/winkawaks.exe:

ascii\_mode: true

其中，app：应用，options：选项。二者连起来就是控制应用（中英文）选项之意。后面的winkawaks.exe是应用在安装目录下的名称。再如Visual Studio 2013安装目录下的名称为devenv.exe（注意将扩展名exe也是要写出来的）



这里如果想将多个程序设置为RIME中文输入状态下进入程序时默认切换为英文状态。那么，你只需将

app\_options/winkawaks.exe:

ascii\_mode: true

中的winkawaks.exe修改为你想要的变化的程序即可。例如你电脑上安装了ps，Ai程序，想做此修改，则，代码书写如下：

app\_options/photoshop.exe:

ascii\_mode: true

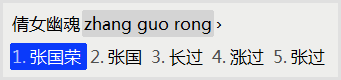
app\_options/illustrator.exe:

ascii\_mode: true

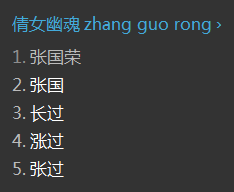
1. 界面颜色配置

小狼毫颜色定制比较麻烦的一点是，它的颜色不是按RGB颜色编码的，而是BGR。群里有个工具是改颜色的，我在这里画蛇添足，在说一些。上图

这是RIME默认的界面



这是我修改的界面

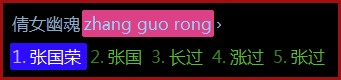


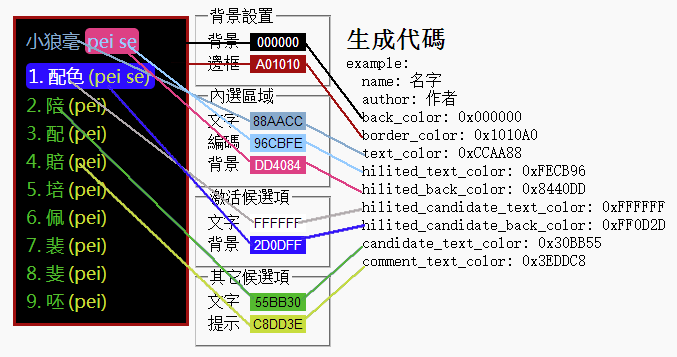
（下面这里请容许我复制粘贴一下）代码修改方式如下面这样做就可以了。

# 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。

  "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

效果图：



配置对应选项如图

直接将右侧的生成代码复制粘贴进weasel.custom.yaml这是针对 weasel.yaml 文件所做的修改。至此，我完成对weasel.custom.yaml文件的配置。

1. **RIME默认设置的修改**（针对文件default.yaml）

RIME的一些默认设置与我们平时所用输入法有些区别，造成用户不太适应，这些都有设置。

1. RIME默认设置每页后选项为五个词汇。将其修改为自己喜欢的个数很简单。

第一步：在用户文件夹新建文本文档，重命名default.custom.yaml文件，填写代码如下:

patch:

"menu/page\_size": 5

数字5既是你要修改的数值了，六七八九十均可（十的后选项为0）。

1. 据我所知，在某些游戏中F4键是大招的快捷键，当你想要发大招时RIME的方案选项出来了。这是一件痛苦的事情，所以有必要修改！！！代码如下：

"switcher/hotkeys":

- "Control+grave"

英文字母便就很清楚的告诉你，你是在修改热键，如此修改，之后便只有Control+grave（Tab上面那个键）可以呼出方案选项来了。

1. 左Shift键输入字符直接切换为英文上屏，输入状态变为英文。

RIME默认情况下，在输入中文时按下Shift键会临时切换到英文状态，回车输入英文后恢复中文输入状态。许多在搜狗、QQ、Google输入法上迁移过来的用户不习惯。而是Shift键直接上屏，之后切换为英文输入。有的或是用Control键进行切换。如此设置都可实现。代码如下：

ascii\_composer/good\_old\_caps\_lock: true

ascii\_composer/switch\_key:

Caps\_Lock: noop

Shift\_L: commit\_code

Shift\_R: inline\_ascii

Control\_L: clear

Control\_R: commit\_text

共七行代码，前两行代码不用管了，直接写就是了。解释下后面的。所有下面这些叙述是在你的RIME为中文输入状态下的操作。

* 1. Caps\_Lock: noop代码：

Caps\_Lock键就是大写键。noop是在告诉Caps\_Lock键，你丫只能打酱油了，神马事情都与你无关。当然大小写这种与系统有关而与RIME无关的事情它还是要管得。

* 1. Shift\_L: commit\_code代码：

其中L指Left，左的意思。commit-提交。code－代码。

当我们输入一段文字未上屏之前，按此键后字符将被将直接上屏，RIME切换为英文输入状态。再次按此键RIME切换回中文输入状态。

* 1. Shift\_R: inline\_ascii代码：

其中R指Right，右的意思。inline－直接插入。ascii－字符，美国信息交换标准码。

这个设置就是RIME的默认设置，在我们输入一段文字未上屏之前，按此键后，不上屏，字符转变为英文输入，按Enter键英文字符上屏，之后默认还是中文输入。

* 1. Control\_L: clear代码

其中clear为清除的意思。

在你输入一段字符后按此键，字符被清除，同时RIME输入状态切换为英文输入。在按此键，RIME输入切换回中文输入。

（告诉大家一下与clear类似但不转换输入状态的快捷键是Esc键，非常好用）

* 1. Control\_R: commit\_text代码

其中text为文本的意思。

在你输入一段拼音字符后，按此键，文字上屏，同时RIME切换为英文输入。在按此键，切换回来。

1. 对RIME快捷键的修改

RIME的默认快捷键设置在default.yaml文件中定义。

RIME默认Control+Shift+4切换繁简转换，我自己添加了Control+Shift+space快捷键也控制繁简转换。

方法是将default.yaml文件中定义的快捷键复制进default.custom.yaml文件中，在代码末尾添加自己的快捷键设置。

代码如下：

key\_binder:

bindings:

- {accept: "Control+p", send: Up, when: composing}

- {accept: "Control+n", send: Down, when: composing}

- {accept: "Control+b", send: Left, when: composing}

- {accept: "Control+f", send: Right, when: composing}

- {accept: "Control+a", send: Home, when: composing}

- {accept: "Control+e", send: End, when: composing}

- {accept: "Control+d", send: Delete, when: composing}

- {accept: "Control+k", send: "Shift+Delete", when: composing}

- {accept: "Control+h", send: BackSpace, when: composing}

- {accept: "Control+g", send: Escape, when: composing}

- {accept: "Control+bracketleft", send: Escape, when: composing}

- {accept: "Alt+v", send: Page\_Up, when: composing}

- {accept: "Control+v", send: Page\_Down, when: composing}

- {accept: ISO\_Left\_Tab, send: Page\_Up, when: composing}

- {accept: "Shift+Tab", send: Page\_Up, when: composing}

- {accept: Tab, send: Page\_Down, when: composing}

- {accept: minus, send: Page\_Up, when: has\_menu}

- {accept: equal, send: Page\_Down, when: has\_menu}

- {accept: comma, send: Page\_Up, when: paging}

- {accept: period, send: Page\_Down, when: has\_menu}

- {accept: "Control+Shift+1", select: .next, when: always}

- {accept: "Control+Shift+2", toggle: ascii\_mode, when: always}

- {accept: "Control+Shift+3", toggle: full\_shape, when: always}

- {accept: "Control+Shift+4", toggle: simplification, when: always}

- {accept: "Control+Shift+5", toggle: extended\_charset, when: always}

- {accept: "Control+Shift+exclam", select: .next, when: always}

- {accept: "Control+Shift+at", toggle: ascii\_mode, when: always}

- {accept: "Control+Shift+numbersign", toggle: full\_shape, when: always}

- {accept: "Control+Shift+dollar", toggle: simplification, when: always}

- {accept: "Control+Shift+percent", toggle: extended\_charset, when: always}

- {accept: "Shift+space", toggle: full\_shape, when: always}

- {accept: "Control+period", toggle: ascii\_punct, when: always}

# - {accept: "Control+Shift+space", toggle: simplification, when: always} #我添加的欲使其生效，只需将代码前面的#符号去掉即可

注：此处需将代码复制进default.custom.yaml文件中，而不是直接在default.custom.yaml文件中添加快捷键设置，这看起来更像是覆盖，而不是“打补丁”。

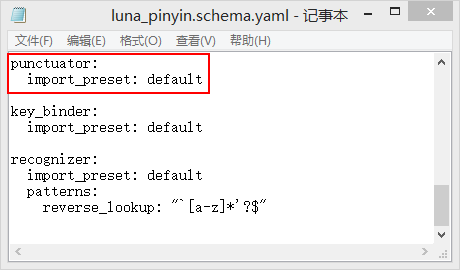
至此，完成我对default.custom.yaml文件的配置

♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋

注意对以上两文件的定制是全局的，也就是讲，无论你用的是什么输入方案，全拼、双拼或是五笔，此两文件对RIME的设置均起作用。而一下面要讲到的文件定制，则是针对不同的输入方案进行的。但大同小异，例如对标点的定制，只需修改一下文件名称就可以了。

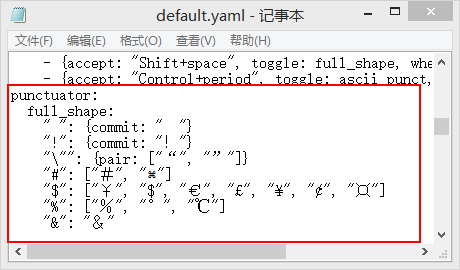
♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋♋

1. **RIME其它设置**（针对文件luna\_pinyin.custom.yaml；symbol.yaml）
2. 特殊符号与标点的定制
   1. 在说标点定制之前，先说明一下RIME**默认**的标点输入方式。

以明月拼音为例。打开luna\_pinyin.schema.yaml文件查看代码，有以下

punctuator为标点的意思。下面import为导入，preset为预设，两词连起来就是导入预设的意思。从哪里导入呢？就是后面的default.yaml文件。

打开未经配置的default文件，你会看到标点



标点是由luna\_pinyin.custom.yaml文件规定从哪里导入的。默认是从default.yaml文件中导入标点，那么我们可以为default.yaml打补丁。下面是复制粘贴：

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「、」，可如此编辑定制luna\_pinyin.custom.yaml:

patch:

punctuator/full\_shape:

"/" : "、"

punctuator/half\_shape:

"/" : "、"

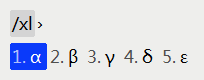
以上在输入方案设定中写入两组新值，合并后的输入方案成为：

# 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。

其它的关于标点的设置可参考官方给的教程，我这里只说我自己对标点等的设置。不过再说标点的设置之前，我先说对特殊表情符号的设置如下：

* 1. RIME没有软键盘，有些童鞋说，打特殊字符不方便，其实，那是因为你还未了解到RIME 强大的特殊字符输出能力。如按/（问号键）之后敲入xl字符，你便可以看到完整地希腊字符了。如图



这个主要是symbols.yaml文件在起作用，打开此文档，你就可以看见RIME的所有的特殊符号了。通过定制symbols.yaml文档，我们可以添加些我们自己常用的表情。输出这些字符操作步骤如下。

添加自己表情时，我们新建一个表情符号文档，随意命名，姑且叫它mysymbols好了（symbols文件不能向前面那样通过custom来定制）。我们要注意，RIME自带的表情符号我们还是要用的，所以，添加自己的表情之前，需先将symbols.yaml中的内容复制进mysymbols.yaml文件中（类似于上文中对快捷键的定制，是在覆盖而不是打补丁），然后进行表情的添加，具体怎么添加，你就看着symbols.yaml中的样式来就好了，此处不予赘述。

之后修改luna\_pinyin.custom.yaml文件中'punctuator/import\_preset':指向为mysymbols。

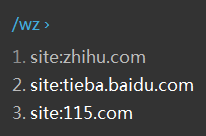
新建luna\_pinyin.custom.yaml文档，打开后，填写代码如下：

patch:

'punctuator/import\_preset': mysymbols

'recognizer/patterns/punct': "^/([a-z]+|[0-9])$"

重新部署即可。

当然你也可以导入默认的symbols.yaml中的标点文件配置。而我想做的是对它进行控制，添加自己常用的表情，比如这个→\_→。像一些网站我也可以添加进去，故自行命名mysymbols.yaml文件，以进行表情添加的操作。

* 1. 标点定制
     1. 如果注意观察上面给的那两行代码的话你会注意到'punctuator/import\_preset':这一行代码。它的意思是指，普通的标点符号也是在你指定的文件中导入的。打开symbols.yaml文件，你会注意到普通的标点符号。在上面操作中，即将symbols.yaml中的代码复制进mysymbols.yaml文件中时，你最好是全部复制进去，如果遗漏普通的标点复制工作，特殊符号表情也不会起作用的。就是讲'punctuator/import\_preset': mysymbols

'recognizer/patterns/punct': "^/([a-z]+|[0-9])$"

这两行代码共同决定了特殊符号的导入，而第一行代码同时又决定了普通标点的导入工作。（希望你懂了→\_→）

所以现在我们通过定制mysymbols.yaml文件中的普通标点来达到我对标点符号的控制。

* + 1. 我理想中的标点符号是在英文状态下半角输入（这个不用设置，一定是的）中文状态下也是半角输入（部分字符如逗号句号除外）。小狼毫自身提供的标点符号相当全面，可是大多数我用不到，每个字符都有好几个后选项，一定程度上影响了我的打字效率，本人并不是都用到，所以自己定制了标点输入。即，中文输入状态下为半角输入（逗号句号引号之类中文字符除外，因为word里面打文档，你是要正规的去打字符，要用全角的。）

说道半角全角，那么有必要解释一下了。半角字符主要是针对英文字符，如ABCDE,.!@#$等他们在二进制中占1B（不懂的话，简单的理解为占用一个空格）空间，而中文字符，如”这些汉字还有“”ＡＢＣＤ”。等是占2B空间的（占用两个空格）。

所以我们可直接直接编辑修改mysymbols.yaml文件中的punctuator部分。我对普通符号所做修改会在文末给出链接。

如此，便可实现我对标点的设置。

1. 词库导入设置

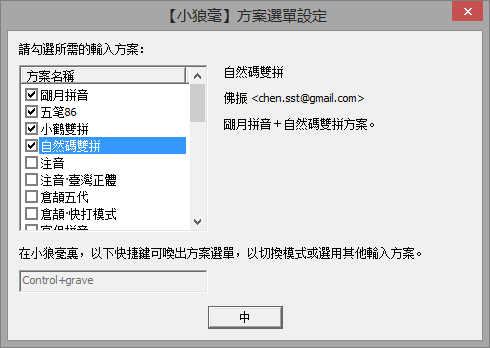
这个在贴吧置顶贴里面已经有详细设置了。我在这里只想说一些你第一次安装RIME时最基础的。详情请看置顶贴

（复制粘贴贴吧置顶贴。。。。。。）

* 1. RIME的词典由两部分组成。  
     一部分是由系统文本词库（一般以xxx.dict.yaml结尾），通过「重新部署/deploy」生成的固态词典（一般以xxx.table.bin结尾）。这部份词库因为在输入过程是固定不変的，所以存在用大量的词量，也不允许用户来直接删除。  
     另一部分就是记录我们用户输入习惯的用户词典（一般以xxx.userdb.kct）结尾。这部份词库的词量，正常情况下是由用户输入的时候随时生成的；其词量可以动态调整，数量理论上来说不会特别多，也允许用户自行删除（shift+delete）。
  2. 导入词库之前最好进行繁简转换

1. 模糊拼音定制
   1. 以“明月拼音”为例，可直接对luna\_pinyin.custom.yaml文件进行编辑。于此处<https://gist.github.com/2320943>取得文档，之后的就是根据自己的需求，将#符号去掉便可。
   2. 注：【明月拼音・简化字／台湾正体／语句流】也适用此文档， 只须将模板保存到luna\_pinyin\_simp.custom.yaml 、 luna\_pinyin\_tw.custom.yaml 或luna\_pinyin\_fluency.custom.yaml 。
   3. 【吴语】模糊音定制模板<https://gist.github.com/2015335>
2. 方案选择

RIME默认的输入方案没有双拼、五笔等。可自行添加



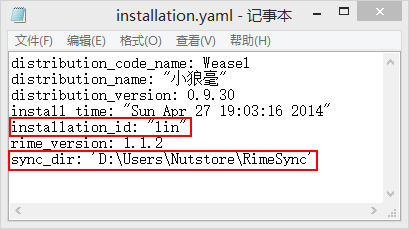
如图勾选输入方案，便可直接添加你想要的输入了。

1. **RIME个人词库配置同步**（针对文件installation.yaml，文件夹sync）

RIME本身不可以实现云同步，但我们可以借助第三方云端实现此功能。我本人使用的坚果云网盘，类似的像是百度，115均可。

第一步打开用户文件夹中的installation.yaml文件，修改sync\_dir指向位置（这个位置将是RIME的词典与配置存储的地方）

另外也可修改installation\_id为自己喜欢的名称（为字母下划线数字，也可以不修改，我的是修改为了lin）



第二部做好上述修改后，点击用户同步。同步完毕后，lin文件夹设置为同步文件夹。右击即可实现设置。

以后若在其他电脑上使用RIME可做如下步骤：

* 1. 安装好RIMR与你的坚果云客户端。
  2. 通过坚果云客户端下载好你的个人词库配置文件等（注意位置要与你在installation.yaml中修改的一致）
  3. 修改用户文件夹中installation.yaml文件中的sync\_dir指向，使其与坚果云下载好的个人配置等文件位置一致。
  4. 点击“用户资料同步”同步个人词库等。

关于同步在说明几点：

* + 1. 上述中的第四步同步是将从云端同步至本地的词库同步进你的RIME中，而个人配置没有同步进RIME“用户文件夹”中。个人配置需要手动将你设置好的几个文件手动拖入进“用户文件夹”，重新部署以实现配置的同步。
    2. 词库的同步对于RIME讲是一个双向的同步。

举例，第一步：你在甲电脑使用RIME积累了词汇ABCDE，而后坚果云同步网盘将ABCDE词汇同步进云端。打开乙电脑，坚果云将云端存储的词汇ABCDE同步进本地的“同步文件夹”（也就是sync\_dir指向的文件夹）。第二步：乙电脑在之前积累了词汇为EFGH，当乙点击“用户资料同步”后，RIME实现的是一个双向的同步：即乙电脑积累的词汇EFGH同步进“同步文件夹”，“同步文件夹”中的ABCDE同步进“用户文件夹”。

* + 1. 配置的同步是单向的同步，即由“用户文件夹”指向“同步文件夹”
    2. 原因也简单，如果不是这样，当“用户文件夹”与“同步文件夹”中的配置不一致时不知道采用哪个，那就乱了。

教程文档等在这里 <http://jianguoyun.com/p/DRylhFMQv_3jBRjxhwk>

我的个人配置等在这里<https://jianguoyun.com/p/DUK6VvAQv_3jBRj9zgk>

你可以下载下来直接使用，但我本人非常建议你一步一步去做。最后，我说话比较罗嗦，文中言语难免有糟粕之处，如有纰漏，望指摘。