1，relayer
语法:relayer(newlayernumber)
作用:改变套用该行模板所产生新行的层数,如果要使所产生行的层数都是一样的，可以只需在模板行设置一下模板行的层数，所有套用该模板行所产生的新行，会自动复制模板行的层数。
例子:
Comment: 0,0:00:00.00,0:00:00.00,Default,,0,0,0,template syl,!relayer(syl.i)!

2，restyle
语法:restyle(newstylename)
作用:改变套用该行模板所产生新行的样式名称，注意只是改名称，不是将原样式复制以后重新命名，同时line.styleref也将指向新样式，新改的样式名称需提前在样式库里设置好。在模板运行时改为新样式，模板所套用的大小与位置信息还是由原来的样式决定。
例子:无

3,maxloop
语法:maxloop(maxloopnumber)
作用:动态控制模板行的循环次数
例子:Comment: 0,0:00:00.00,0:00:00.00,Default,,0,0,0,template
syl,!maxloop(syl.width +
2*line.styleref.outline)!{\clip(!line.left+syl.left-line.styleref.outline+j-1!,0,!line.left+syl.left-line.styleref.outline+j!,!meta.res_y!)\an5\move(!line.left+syl.center!,!line.middle!,!line.left+syl.center!,!line.middle+math.random(-20,20)!,$start,$end)\shad0}

maxloop(syl.width + 2*line.styleref.outline)
控制循环次数，为syl的像素宽度与两倍边框值的和
\clip(!line.left+syl.left-line.styleref.outline+j-1!,0,!line.left+syl.left-line.styleref.outline+j!,!meta.res_y!)
做该字的像素宽度个矩形切割（矩形宽度为一，高度为视频的高度）
line.left+syl.left-line.styleref.outline 为该所有syl像素的最左边(为什么这里line.left还要加上syl.left因为syl.left是一个相对值，相对于line.left)
meta.res_y 视频纵向长度
\move(!line.left+syl.center!,!line.middle!,!line.left+syl.center!,!line.middle+math.random(-20,20)!,$start,$end)
纵向垂直随机移动

4，loopctl
语法:loopctl(newj, newmaxj)
作用：动态改变控制循环的变量j和maxj。j与maxj，为使用修饰符loop时，j为当前循环的次数，maxj为最大的循环次数也就是loop后面所跟的数字，一般用不到
例子:
1： Comment: 0,0:00:00.00,0:00:00.00,Default,,0,0,0,template syl noblank notext loop 10,!j! !maxj! !loopctl(j+1,maxj)!
2： Comment: 0,0:00:00.00,0:00:00.00,Default,,0,0,0,template syl noblank notext loop 10,!loopctl(j+1,maxj)! !j! !maxj!

两个例子都是将j加一以后重新赋值给j，事实上运行时，j的值会每次加2，因为本身执行时计数会回加1。注意上面两个结果的例子的结果是不一样的。

5，remember
语法:remember(name, value)
作用：存储给定的名字与相对应的值，并返回传入的value，如果想不返回返回值，可以参考附件例子里myremember的写法。
例子: 参见recall

6，recall
语法:
1，recall(name)
2，recall(name,defaultvalue)
作用：查找给定的名字与相对应的值。第二个设置了如果相对应名字不存在的时候返回的值。
例子:
Comment: 0,0:00:03.00,0:00:03.00,Default,,0,0,0,template
pre-line,!recall("last_line_start_time",0)!
!remember("last_line_start_time",line.start_time)!
recall得到上一次调用remember记住的行的时间值，remember记住该行的时间

7，remember_line
语法: remember_line(name, value)
作用：同remember,不过remember_line保存是单独对于每行来说的，remember是相对于全局来说的，所以实际上是每行有一个同名的保存了，行之间不相互影响，超了该行的范围后通过recall就取不出保存的值了。从syl,char 等能取到保存的值。为方便我在例子自己做了个reactor的函数，需将kara-templater.lua放入aegisub\automation\autoload文件下覆盖原文件，然后重启aegisub,如果你不打算这么做，可以把所有的reactor删掉。
例子: 见ass

8，remember_syl
这个我也没搞明白怎么用的，有可能是本身有问题

9，remember_basesyl
用法：remember_basesyl(name, value)
作用：在这里有两个变量先需要区分，basesyl与syl。
{\k5}明日{\k10}ま{\k7}た
在这里的使用template syl noblank 后会产生3个 basesyl，为方便表述只用text代表该syl
1，明日
2，ま
3，た
为了方便统一，上面3个basesyl也能看做3个syl

而使用 template char noblank 后会产生4个 syl
4，明
5，日
6，ま
7，た

1-3，4-7都是syl
但4-7是伪syl也就是说它是虚构出来的，这也是为什么我们在使用template char 时能使用syl变量。basesyl将访问1-3.

同样还有orgline与line，不过一般都能统一通过line访问

例子: 见ass。

10，remember_if
用法：remember_if(name, value, condition)
作用：跟remember类似，只是多了条件控制。