Contents

1 小闹钟

这是一个小工具，用来给我自己做点提醒的。 它的用途是设置一些循环提醒、闹钟提醒、倒计时提醒之类的。 当提醒的时间到的时候，它会给我的notification center发消息提醒。

1.1 场景

看看一些场景:
1. 每天吃药
2. 定时喝水
3. 定时休息

1.1.1 每天吃药

我设定一个闹钟，每天在特定的时间段给我提醒吃药。
每天，如果我吃过药了，那么我会去手动取消这次提醒， 而如果我一直置之不理，这个闹钟会按照我先前指定的频率重复提醒。

1.1.2 定时喝水

我设定了一个喝水闹钟，每隔五分钟就会提醒我喝点水，它会按照频率定时提醒。 不过有些时候，我是不希望被打扰的，所以我会提前设定喝水的提醒时段。

1.1.3 定时休息

我一工作起来就容易忘了时间，为了健康着想，我需要设定一个定时休息的闹钟。 每次我开始工作时，我都会启动这个闹钟，这个闹钟启动之后，会开始倒计时， 在倒计时结束时提醒我。如果我不理睬这个提醒，那么就会一直反复提醒，直到我取消。

1.2 设计

小闹钟有一个界面，用于显示当前正在执行的闹钟。
设定过的闹钟被放在闹钟库中，便于下次使用。
可以添加倒计时闹钟、定时闹钟等等。
设置界面要足够大，便于手指点击。

2a ⟨设置界面的定义 2a⟩≡

```
class AlarmSetting(wx.Panel):
    def __init__(self, parent):
        wx.Panel.__init__(self, parent)
        ⟨设定界面的初始布局 2b⟩
    ⟨设置界面的事件处理函数 5b⟩

app = wx.App(redirect=False)
f = wx.Frame(None)
f.SetSize((400, 800))
sz = wx.BoxSizer(wx.VERTICAL)
f.SetSizer(sz)
p = AlarmSetting(f)
sz.Add(p, proportion = 1, flag = wx.EXPAND|wx.ALL)
f.Show()
app.MainLoop()
```

This code is used in chunk 6c.

2b ⟨设定界面的初始布局 2b⟩≡

```
self.SetBackgroundColour(wx.Color(240, 240, 255))
mainSizer = wx.BoxSizer(wx.VERTICAL)
self.SetSizer(mainSizer)
⟨一个巨大的输入框，用于输入闹钟的标题 2c⟩
⟨我是水平分割线 6b⟩
⟨用来设置计时方式和时长的区域 3⟩
⟨我是水平分割线 6b⟩
⟨用来设置重复提醒的间隔的区域 (never defined)⟩
```

This code is used in chunk 2a.

2c ⟨一个巨大的输入框，用于输入闹钟的标题 2c⟩≡

```
te = wx.TextCtrl(self)
font = te.GetFont()
font.SetPointSize(24)
try:
    font.SetFaceName(u'Yahei Mono')
except Exception, e:
    pass
te.SetFont(font)
mainSizer.Add(te, proportion = 0, flag = wx.EXPAND | wx.LEFT | wx.RIGHT)
self.titleInput = te
```

This code is used in chunk 2b.

计时方法分两种：固定时间和倒计时。因此我们的时间设置区域也区分这两种情况。 区分的方法是，在区域的最上方设置类似segment/tab control，以区分计时方式。 在提示区域的下方就是时间设定。

关于闹钟重复的讨论：
1. 不重复
2. 重复
2.1. 每年，月，日，时，分，秒
2.2. 每个工作日
2.3. 每个满足条件的时间点
2.4. 如果今天是工作日，则在今天启用
2.5. 如果XXX，则启用
2.5.1. 如果今天是工作日，则启用
2.5.2. 如果今天没用过，则启用
2.5.3. 如果过了整点，则启用
2.5.4. 如果是每个月的第一天，则启用
2.5.5. 如果是某天，则启用
2.5.6. 如果程序启动，则启用

有一个闹钟设置器，它会反复检索启用条件，从模板库中根据闹钟模板生成新的闹钟。

当然，这种设置器并不是一直在检索条件，这不合理。怎样才是合理的？在恰当的时机，根据启用条件生成定时事件，在定时事件中启用闹钟？

我不想一直检索条件，是因为检索条件可能是非常复杂的操作，因此可能会阻塞，也会无谓地消耗计算资源。因此，反其道而行之，程序每次启动，会检索启用条件，生成闹钟。但是，假设程序一直运行，那么我们也需要一个时机来检索启用条件。什么时机呢？一个隐藏的时钟事件？这个事件的执行，一是激活一个闹钟，二是用于设置同等条件下的下一次时钟事件？当程序启动、重复条件更改时，都会启用这个隐藏的时钟事件。

1. 程序有闹钟模板库和激活闹钟库
2. 每次启动时，程序扫描闹钟模板库中的闹钟模板，根据模板内容生成激活闹钟
3. 每个激活的闹钟知道来自于哪个模板，当闹钟结束时，程序会再次根据闹钟的模板生成激活闹钟
4. 关于闹钟模板的状态：
4.1. 闹钟模板有两种状态，启用和停用，只有启用的闹钟模板才能用于生成激活的闹钟。

3 ⟨用来设置计时方式和时长的区域 3⟩≡
⟨计时方式提醒/切换区域 4a⟩
⟨时间设置区域 (never defined)⟩

This code is used in chunk 2b.

4a ⟨计时方式提醒/切换区域 4a⟩≡

```
headerSizer = wx.BoxSizer(wx.HORIZONTAL)
t1 = wx.StaticText(self, style = wx.BORDER_NONE,label = u"倒计时")
t1.SetFont(font)
headerSizer.Add(t1)
headerSizer.Add((20, 0))
t1.Bind(wx.EVT_LEFT_UP, self.onChooseCountDown)
self.ctHeader = t1

t1 = wx.StaticText(self, style = wx.BORDER_NONE,label = u"定时")
t1.SetFont(font)
t1.SetForegroundColour(wx.Color(128, 128, 128))
headerSizer.Add(t1)
t1.Bind(wx.EVT_LEFT_UP, self.onChooseAlarm)
mainSizer.Add(headerSizer)
self.amHeader = t1
p = wx.Panel(self)
p.SetBackgroundColour(wx.Color(240, 240, 255))
panelSizer = wx.BoxSizer(wx.VERTICAL)
p.SetSizer(panelSizer)
p.SetMinSize((100, 200))
⟨布局时长设置界面 4b⟩
mainSizer.Add(p, proportion = 0, flag = wx.EXPAND | wx.LEFT | wx.RIGHT)
```

This code is used in chunk 3.

4b ⟨布局时长设置界面 4b⟩≡

```
sz = p.GetSizer()
bsz = wx.BoxSizer(wx.HORIZONTAL)
imgPath =u"appbar.navigate.previous.png"
⟨添加时长的操作按钮 5d⟩
bsz.Add((32, 0))
imgPath =u"appbar.navigate.next.png"
⟨添加时长的操作按钮 5d⟩
bsz.Add((1, 0), proportion = 1, flag = wx.EXPAND| wx.LEFT| wx.RIGHT)
imgPath =u"appbar.add.png"
⟨添加时长的操作按钮 5d⟩
bsz.Add((32, 0))
imgPath =u"appbar.minus.png"
⟨添加时长的操作按钮 5d⟩
sz.Add(bsz, proportion = 0, flag = wx.EXPAND | wx.LEFT | wx.RIGHT)
sz.Add((0, 20))
⟨添加年月日时分秒 5a⟩
```

This code is used in chunk 4a.

5a ⟨添加年月日时分秒 5a⟩≡
```
gsz = wx.GridSizer(2, 3)
hint = u"年"
⟨添加时间编辑控件 5c⟩
hint = u"月"
⟨添加时间编辑控件 5c⟩
hint = u"日"
⟨添加时间编辑控件 5c⟩
hint = u"时"
⟨添加时间编辑控件 5c⟩
hint = u"分"
⟨添加时间编辑控件 5c⟩
hint = u"秒"
⟨添加时间编辑控件 5c⟩
sz.Add(gsz, proportion = 1, flag = wx.EXPAND | wx.LEFT | wx.RIGHT)
```
This code is used in chunk 4b.

5b ⟨设置界面的事件处理函数 5b⟩≡
```
def onEditKillFocus(self, te, hint, ev):
    v = te.GetValue().strip()
    if len(v) == 0:
        te.SetValue(hint)
```
This definition is continued in chunk 6a.
This code is used in chunk 2a.

5c ⟨添加时间编辑控件 5c⟩≡
```
te = wx.TextCtrl(p)
te.SetFont(font)
te.SetValue(hint)
te.SetForegroundColour(wx.Colour(220, 220, 220))
gsz.Add(te, proportion = 1, flag = wx.ALIGN_CENTER_HORIZONTAL)
te.Bind(wx.EVT_KILL_FOCUS, functools.partial(self.onEditKillFocus, te, hint))
```
This code is used in chunk 5a.

5d ⟨添加时长的操作按钮 5d⟩≡
```
b = wxTools.makeBitmapButton(
        p,
        (48, 48),
        imgPath)
b.SetBackgroundColour(wx.Color(240, 240, 255))
bsz.Add(b, proportion = 0)
```
This code is used in chunk 4b.

6a ⟨设置界面的事件处理函数 5b⟩+≡
```
def onChooseCountDown(self, evt):
    t1 = self.ctHeader
    t1.SetForegroundColour(wx.Color(0, 0, 0))
    t1 = self.amHeader
    t1.SetForegroundColour(wx.Color(128, 128, 128))
    self.Refresh()
def onChooseAlarm(self, evt):
    t1 = self.ctHeader
    t1.SetForegroundColour(wx.Color(128, 128, 128))
    t1 = self.amHeader
    t1.SetForegroundColour(wx.Color(0, 0, 0))
    self.Refresh()
```
This code is used in chunk 2a.

6b ⟨我是水平分割线 6b⟩≡
```
l = wx.StaticLine(self, wx.HORIZONTAL)
mainSizer.Add((0, 4))
mainSizer.Add(l, proportion = 0, flag = wx.EXPAND | wx.LEFT | wx.RIGHT)
mainSizer.Add((0, 4))
```
This code is used in chunk 2b.

2 文件

6c ⟨myAlarms.py 6c⟩≡
```
import wx
import wxTools
import functools
⟨设置界面的定义 2a⟩
⟨闹钟界面的定义 (never defined)⟩
⟨主界面的定义 (never defined)⟩
```
Root chunk (not used in this document).

6d ⟨tangle source codes 6d⟩≡
```
tangleSource myAlarms.py $file myAlarms.py
```
This code is used in chunk 7a.

3 action

6e ⟨action 6e⟩≡
```
⟨tangle_in_linux 7a⟩
⟨weave 8⟩
```
Root chunk (not used in this document).

3.1 tangle

7a ⟨tangle_in_linux 7a⟩≡

```
fileName=myAlarms
file=$fileName.nw
ltx_file=$fileName.ltx
aux_file=$fileName.aux
log_file=$fileName.log
function tangleSource
{

    echo '#-*- coding: utf-8 -*-' > $3
    ../pytangle.py -R"$1" -L'#line %L, %F%N' $2>> $3
    python ../literatePython/LineDirective.py $3>temp.py
    rm -rf $3
    mv temp.py $3
}
⟨tangle source codes 6d⟩
⟨tangle_windows_part 7b⟩
```

This code is used in chunk 6e.

7b ⟨tangle_windows_part 7b⟩≡

```
notangle -R"action\\_in\\_win" -t4 $file> action.bat
```

This code is used in chunk 7a.

7c ⟨action_in_win 7c⟩≡

```
@echo off
myAlarms.py
pause
exit 0
```

Root chunk (not used in this document).

3.2 weave

8 ⟨weave 8⟩≡

```
noweave -x $file| \
sed 's/\\usepackage{noweb}/\\usepackage[top=1.2in,bottom=1.2in,left=1.2in,right=1in]{geometry}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{fontspec, xunicode, xltxtra}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{listings}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage[120, ampersand]{easylist}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{paralist}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{color}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{hyperref}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{underscore}&/g'| \
sed 's/\\usepackage{noweb}/&\\noweboptions{longxref}/g'| \
sed 's/\\usepackage{noweb}/&\\noweboptions{smallcode}/g'| \
sed 's/\\usepackage{noweb}/&\\noweboptions{alphasubpage}/g'| \
sed 's/\\usepackage{noweb}/&\\noweboptions{longchunks}/g'| \
sed 's/\\usepackage{noweb}/&\\XeTeXlinebreaklocale "zh-cn"/g'| \
sed 's/\\usepackage{noweb}/&\\pagecolor{grayyellow}/g'| \
sed 's/\\usepackage{noweb}/&\\definecolor{grayyellow}{RGB}{255, 255, 200}/g'| \
sed 's/\\usepackage{noweb}/&\\XeTeXlinebreakskip = 0pt plus 1pt minus 0.1pt/g'| \
sed 's/\\usepackage{noweb}/&\\setmainfont[BoldFont={Adobe Heiti Std}]{Adobe Song Std}/g'| \
sed 's/\\usepackage{noweb}/&\\setmonofont[Color=0000FF99]{Microsoft YaHei UI Light}/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{amsmath}&/g'| \
sed 's/\\usepackage{noweb}/\\usepackage{amssymb}&/g'| \
sed 's/\\begin{document}/&\\tableofcontents/g'| \
sed 's/\\begin{document}/&\\setcounter{tocdepth}{7}/g'| \
sed 's/\\documentclass[11pt]/&[11pt]/g'|
sed 's/  /  /g'> $ltx_file &2|iconv -f utf-8 -t gbk
xelatex $ltx_file
xelatex $ltx_file
echo $ltx_file|sed 's/ltx$/aux/g'|xargs rm -rf
echo $ltx_file|sed 's/ltx$/toc/g'|xargs rm -rf
echo $ltx_file|sed 's/ltx$/out/g'|xargs rm -rf
rm -rf $ltx_file
rm -rf $aux_file
rm -rf $log_file
```

This code is used in chunk 6e.

9 ⟨declare of literate programming 9⟩≡

```
/*

    ****************************************
    *                  *
    *       注意事项        *
    *                  *
    ****************************************

  你看到的这份源码文件不是直接生成的,而是使用noweb工具，从*.nw文件中将代码抽取出来组织而成的。
  因此请不要直接编辑这些源文件，否则它们会被*.nw文件中的内容覆盖掉。
  如果想了解如何使用noweb工具抽取代码和生成pdf文档，请联系huangyangkun@gmail.com。
  noweb是一个"文学编程（literate programming）"工具。
  关于文学编程: http://zh.wikipedia.org/wiki/%E6%96%87%E5%AD%A6%E7%BC%96%E7%A8%8B
  关于noweb: http://en.wikipedia.org/wiki/Noweb
  */
```

Root chunk (not used in this document).

4 代码块列表