邮学小帮手

BUPT Students' Helper

详细设计文档

孙泽凯 2020212180

张扬 2020212185

王梓安 2020212212

2022 年 4 月-2022 年 6 月

目录

孙泽凯202021180张扬202021185王梓安202021212

1、系统概述

1.1、系统简介

目前，北邮大学生 PC 端还缺乏比较好用的课程表软件和 ddl 提醒软件，同学们在网课期间自己在家无人督促学习，普遍缺乏学习动力和时间观念；而且频繁使用浏览器登录北邮教务系统查看信息较为麻烦，缺乏一个直接查阅教务系统信息的手段；于是，为了辅助北邮学子的学习工作，我们团队设计了这个邮学小帮手；它可以帮助同学们更好地上课学习，更方便的获取教务系统信息、课程信息和网课链接，获取更充足的学习动力。

它的主要功能包括以下几个方面：

·使用爬虫获取教务系统课程表信息，并将爬取的信息存储在 xls 文件中，

·根据存储的 xls 文件生成课程表，并可以按照课程周数查看课程信息，

·用户可自己设置待办事项，读取写入删除代办事项并存储在 xls 文件中，

·课前自动弹窗提醒上课、待办事项到达时间后也会弹窗并播放音乐提醒，

·支持自定义课程表和待办事项背景图、更改课程信息字体颜色和提醒音等，

·可一键生成桌面小挂件，右键单击就可以在桌面上随时查看课程信息。

上图为邮学小帮手的主界面，点击对应的按钮就可以使用相应的功能，操作简单，使用逻辑清晰。

1.2、术语表

定义系统或产品中涉及的重要术语，为读者在阅读文档时提供必要的参考信息。

序号	术语或缩略语	说明性定义
1	小帮手	邮学小帮手，本项目名称简称。
2	PC	个人电脑，personal computer

1.3、系统运行环境

包括对硬件平台、操作系统、数据库系统、编程平台、网络协议等的描述。

本软件硬件方面使用普通配置的个人电脑即可完成编译开发与使用，需要使用 Windows 操作系统进行支持，未使用数据库系统，Pycharm 编译平台进行开发编译，Anaconda 解释器，python 第三方库（numpy、tkinter、PIL、selenium、time、datetime、pygame、_thread、os、re、openpyxl、csv、pyqt5 等），以及常用浏览器（chrome、edge 等）进行爬虫方面的功能实现。

软件运行时需要一台能够流畅运行的个人电脑，Windows7 或更高版本的操作系统，保证网络连接通畅，并最好使用北邮内网，如在北邮校外使用也提供了通过 VPN 进行爬取信息的使用方式。

1.4、开发环境

本程序完全使用 python 语言实现，PyCharm 编辑软件进行开发编译：

·PyCharm 2021.2.3 (Community Edition)
内部版本号 #PC-212.5457.59
运行时版本: 11.0.12+7-b1504.40 amd64
VM: OpenJDK 64-Bit Server VM，JetBrains s.r.o.
使用 Anaconda 解释器运行 python 代码：
·Anaconda Navigator 1.9.12

使用 edge 或 chrome 浏览器进行爬虫部分的实现：
·Microsoft Edge 版本 99.0.1150.55 (正式版本) (64 位)
·Google Chrome 版本 96.0.4664.93(正式版本)(64 位)

2、数据结构说明

2.1、常量

Main.py:

width = 690 主菜单界面窗口宽度
height = 455 主菜单界面窗口高度
window.config(background="#c0c0c0") 主菜单界面背景颜色
buptphoto = ImageTk.PhotoImage(file="beiyou.jpg") 主菜单背景图文件名称
window.iconbitmap('233.ico') 主菜单左上角北邮小图标文件名称
width1 = 300 开发人员信息窗口宽度
height1 = 200 开发人员信息窗口高度
kf.config(background="#8000ff") 开发人员界面窗口背景颜色
width2 = 500 个性化设置窗口宽度
height2 = 300 个性化设置窗口高度
wd1.config(background="#8000ff")个性化设置窗口背景颜色

Creep.py:

url = 'https://webvpn.bupt.edu.cn' 北邮 VPN 首页地址 url
Choose_back.py:
width = 1300 #图片尺寸修改默认宽度
height = 700 #图片尺寸修改默认高度
window.iconbitmap('233.ico') #窗口左上角北邮小图标文件名称
Choose_color.py:
findcolor = re.compile(r'', '(.*?)'") #颜色值匹配正则表达式

root.iconbitmap('233.ico')　#左上角北邮小图标文件名称
Choose_music.py:
window.iconbitmap('233.ico')　#左上角北邮小图标文件名称

show_calander.py

'课程表.xlsx' 储存课程表信息的文件

Todolist.py：

date1 = time.strftime("%Y-%m-%d", time.localtime())　获取当前日期从而推算出周数
date0 = '2022-02-28'　获取本学期开始日期
week_names = ["星期一", "星期二", "星期三", "星期四", "星期五", "星期六", "星期日"]　获取待办事项表星期数
class_times = ["1", "2", "3", "4", "5", "6"]　获取待办事项最多显示条数
y.geometry('270x600+500+100')　设置“读取”功能弹出窗口大小
z.geometry('270x150+500+300')　设置“写入”功能弹出窗口大小
window.geometry('400x500+500+100')　设置“清除”功能弹出窗口大小

desktopapp.py:

app_type='bupt'　读取名为 bupt 的目录下的挂件图标信息
ROOT_DIR = os.path.join(os.path.split(os.path.abspath(__file__))[0], 'resources') 获取文件位置信息
width = (screen_geo.width() - app_geo.width()) – 300
初始化小挂件的 x 轴位置信息
height = (screen_geo.height() - app_geo.height()) – 800初始化小挂件的 y 轴位置信息

2.2、变量

Main.py：

'课程表.xlsx' 储存课程表信息的文件
bg_path #用于暂时存储自定义背景图片文件路径
class_color #用于暂时存储自定义课表字体颜色的颜色值字符串

Creep.py:

week_path = str('/html/body/div/div/form[2]/div[2]/select[1]/option['+str(k+2)+']') #用于循环获取当前的课程表页面 xpath
classes_path= str('/html/body/div/div/form[2]/table[1]/tbody/tr['+str(j+2)+']/td['+str(i+1)+']/
div[2]') #用于循环获取当前课程信息 xpath

Choose_back.py:

Img #用于存储打开的图片文件
Filepath #用于存储打开的文件绝对路径
Filenewpath #用于存储希望将文件保存到的绝对路径
Choose_color.py:
kebiao_color #用于存储选定的颜色值字符串
Colorvalue #用于存储选择的颜色值详细信息
Choose_music.py:
Mark #用于标记是否已经选择了一个音频文件
Filepath #用于存储所选择的音频文件绝对路径

show_calander.py

1)root = Toplevel()（子窗口）在创建一个独立于主窗口之外的子窗口，位于主窗口的上一层，可作为其他控件的容器,，作为课程表显示系统的主要窗口

2)class_info = read_class_info() 调用 read_class_info()函数，读取课程表信息

3)workbook = openpyxl.load_workbook('课程表.xlsx') load_workbook 函数作用：打开给定的文件名并返回工作簿

4)sheet = workbook['课程表'] 选择 workbook 中的 sheet'课程表'

5）class_info_ 存储课程表信息

6）temp 临时储存单个课程信息

7）one = temp 的前半部分，以回车为分割

8）two = temp 的后半部分，以回车为分割

9) cv = Canvas(root, bg="#BFEFFF", width=1200, height=700) Canvas 控件具有两个功能，首先它可以用来绘制各种图形，比如弧形、线条、椭圆形、多边形和矩形等，其次 Canvas 控件还可以用来展示图片（包括位图），我们将这些绘制在画布控件上的图形，称之为“画布对象”。这里是课程表的主画布。

10）global t 全局变量 t，用于保存当前显示的课程文字对象，以便于界面刷新时删除。

11）pic_way 背景图片的路径

12）colour 课程文字的颜色

13）weeks_F = [] 生成每周课程信息文字对象的函数，均放在这个列表中，用于按钮的生成

14） 90 行 temp 用于记录循环次数的临时变量

15）33 行 temp 用于储存星期文字

16）class_times 用于储存节次信息

17）day 用于记录当前打印的日期

18）line 用于记录当前打印的行

19）week 用于记录当前的周数

20）t_new 用于储存新打印的课程文字的文字对象，便于下次删除

21）date0 学期开始的日期，格式为 XXXX-XX-XX

22）d0 学期开始日期的时间戳

23）date1 当前的日期，格式为 XXXX-XX-XX

24）d1 当前日期的时间戳

Alarm.py

1）root = Tk() 建立根窗口

2）music 提醒音乐文件的路径

3）screenwidth 存储屏幕的宽度的变量

4）screenheight 获取屏幕宽度和高度，并且在高度上考虑到底部的任务栏，为了是弹出的窗口在屏幕中间

5）frame 框架（容器）控件 定义一个窗体 (根窗口也是一个窗体），用于承载其他控件，即作为其他控件的容器

6) label = Label(frame, text=tex, font=('微软雅黑', 20, 'bold')) 窗口显示的文字、并设置字体、字号

7)button = Button(frame, text="我知道了", font="Cooper -25 bold", fg="purple", command=shut_down) “我知道了” 按钮\

8)workbook = openpyxl.load_workbook(' 课 程 表 .xlsx') load_workbook 函数作用：打开给定的文件名并返回工作簿

9)sheet = workbook['课程表'] 选择 workbook 中的 sheet'课程表'

10）class_info_ 存储课程表信息

11）temp 临时储存单个课程信息

12）one = temp 的前半部分，以回车为分割

13）two = temp 的后半部分，以回车为分割

14）date0 学期开始的日期，格式为 XXXX-XX-XX

15）d0 学期开始日期的时间戳

16）date1 当前的日期，格式为 XXXX-XX-XX

17）d1 当前日期的时间戳

18） class_info = read_class_info() 调用 read_class_info()函数，读取课程表信息

19）class_times 储存上课时间的数组

20）alarm_times 储存闹钟信息的数组

21）today 记录当天是该学期的第几天

22）ahead 记录设置的响铃提前的时间

23）to_do_info_ 储存读取文件获得的待办的时间信息和文字信息

24）alarm_times 储存待办和课前提醒的时间

25）alarm_things 储存待办和课前提醒的内容

26）ahead 储存待办和课前提醒的提前时间

2.3、数据结构

包括数据结构名称，功能说明，具体数据结构说明（定义、注释、取值）等。

Main.py：

window = Tk() #创建主菜单界面窗体
size_geo = '%dx%d+%d+%d' % (width, height, (screenwidth-width)/2,(screenheight-height)/2-50) #获取当前操作电脑的屏幕尺寸并存储
buptlabel = Label(window, image=buptphoto) #存储打开的北邮背景图作为标签
buptback = Canvas(window, bg="blue", height=450, width=690) #用于为背景图提供画布，以此将北邮背景图贴到主界面上。
kf = Toplevel() #创建一个开发人员上级窗口
l = tk.Label(kf,text=' 孙泽凯-2020212180\n\n 张扬-2020212185\n\n 王梓安-2020212212',bg='#8000ff',font=('宋体', 18, 'bold'),width=200,height=200, fg = '#ffffff',padx=10, pady=15, borderwidth=10, relief="sunken") #用于记录开发人员窗口相关信息
button0 = tk.Button(window,text='邮 学 小 帮 手 ',bg='#8000ff',font=('华文琥珀', 35, 'bold'),width=25,height=1, fg = '#ffffff',padx=10, pady=15, borderwidth=10, relief="sunken",command=kaifa) #用于记录开发人员按钮信息
bt1 = tk.Button(wd1, text=' 背 景 \n 更 换 ', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh1).pack(side = LEFT,expand = True) #记录个性化窗口中背景更换按钮信息
bt2 = tk.Button(wd1, text=' 颜 色 \n 选 择 ', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh2).pack(side = LEFT,expand = True) #记录个性化窗口中颜色选择按钮信息
bt3 = tk.Button(wd1, text=' 提 示 音 \n 更 换 ', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh3).pack(side = LEFT,expand = True) #记录个性化窗口中提示音更换按钮信息

bt4 = tk.Button(wd1, text='桌面\n挂件', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh4).pack(side = LEFT,expand = True) #记录个性化窗口中桌面挂件按钮信息

button1 = tk.Button(window, text='获取\n课程信息', activebackground='yellow', bg = '#9301fe',font = ('隶书',12),fg='#ffffff', cursor="watch",width=8, height=3, command=command_creep) #记录主窗口获取课程信息按钮信息

button2 = tk.Button(window, text = '显示\n课程表', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=
command_show_calendar) #记录主窗口显示课程表按钮信息

button3 = tk.Button(window, text = '设置\n待办事项', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=
command_todo) #记录主窗口设置待办事项按钮信息

button4 = tk.Button(window, text = '启用\n提醒系统', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=
command_alarm) #记录主窗口启用提醒系统按钮信息

button5 = tk.Button(window, text = '个性化\n设置', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=gexinghua) #记录主窗口个性化设置按钮信息

creep.py:

botton1 = byr_page.find_element('xpath','/html/body/div/div/div/div/div[3]/div/div[2]/div/
div[1]/div/div[1]/div/div[1]/a') #存储爬取到的按钮信息

botton2 =

byr_page.find_element('xpath','/html/body/div[2]/div[3]/div[2]/div/div[1]/a/p')　#存储第二个爬取到的按钮信息

button3 = byr_page.find_element('xpath','/html/body/div/div[2]/div[1]/div[2]/div/div/div[1]/

p')　#存储第三个爬取到的按钮信息

fm1 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe')　#存储爬取到的首个 frame 信息

fm2 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe[2]')　#存储爬取到的第二个 frame 信息

kebiao = []　#建立列表，存储爬取到的所有课程信息

class_data = []　#建立列表，存储爬取到的每周课程信息

day_data = []　#建立列表，存储爬取到的每天课程信息

Choose_back.py:

window = tk.Toplevel()　#记录新建的背景选择上级窗口数据结构

tk.Button(window, text='选择文件', command=open_img).grid(row=1, column=1, padx=5, pady=5)　#记录选择文件按钮信息

tk.Button(window, text='文件备份', command=save_png).grid(row=2, column=1, padx=5, pady=5)　#记录文件备份按钮信息

tk.Button(window, text='尺寸修改', command=fixed_size).grid(row=3, column=1, padx=5, pady=5)　#记录尺寸修改文件信息

Choose_color.py:

root = tk.Tk()　#记录新建的颜色选择上级窗口

lb=tk.Label(root,text='',font=('宋体',10))　#记录需要放置在窗口上的 label 标签

tk.Button(root, text="点击选择颜色", command=callback, width=14, bg='#808080')　#记录选择颜色按钮信息

tk.Button(root, text='设为课程字体颜色',command=kebiaocolor, width=14, bg='#808080')　#记录设置课程字体颜色按钮信息

Choose_music.py:

window = tk.Toplevel()　#记录新建的提示音选择上级窗口

tk.Button(window, text=' 选 择 音 频 ', command=open_music).grid(row=1, column=1, padx=5, pady=5) #记录选择音频按钮信息

tk.Button(window, text=' 设 置 音 频 ', command=save_music).grid(row=2, column=1, padx=5, pady=5) #记录设置音频按钮信息

Alarm.py

主要数据结构为列表和 tkinter 画布和 root 对象以及 pygame 对象，具体请参见变量列表。

show_calander.py

主要数据结构为列表和 tkinter 画布和 root 对象，具体请参见变量列表。

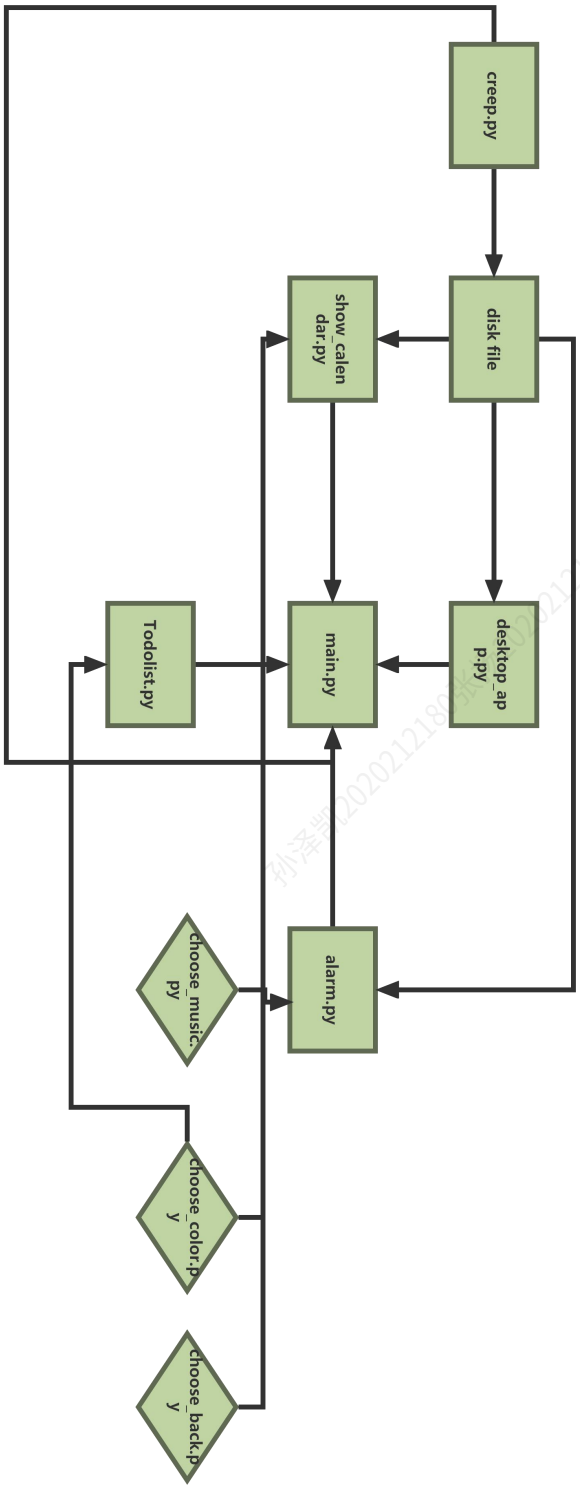
Todolist.py：

openpyxl.load_workbook('待办事项.xlsx') 读取待办事项所有信息

tkinter.StringVar() 以字符串的形式获取写入模块输入信息（包括待办事项的内容和时间）

3、模块设计

3.1、软件结构

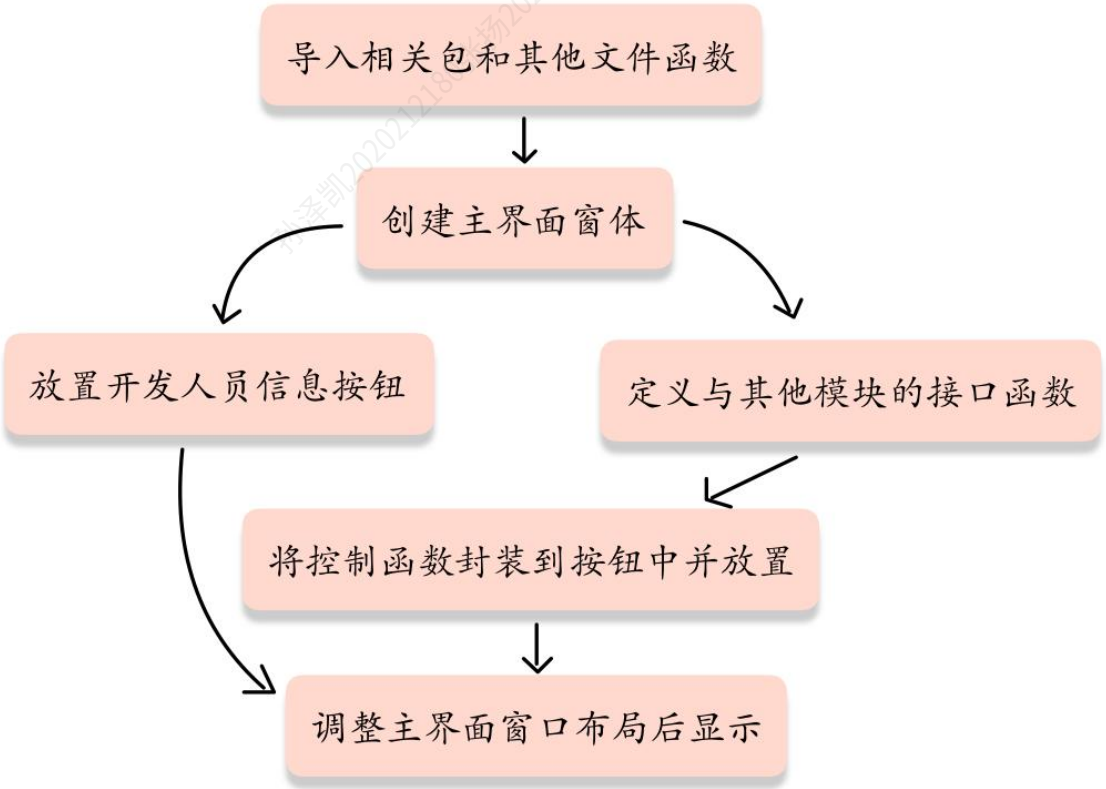


3.2、功能设计说明

邮学小帮手在设计过程中遵循了功能模块化、代码模块化的功能集成化的设计思路。具体来说，是将程序的总任务分解为一个个具体的小任务分别用代码文件实现，包括“爬取信息”、“课表生成”、“待办事项管理”、“个性化设置”、“待办事项提醒”、“桌面小挂件”六个主要功能文件，并把这些功能集成在主程序页面上供用户选择使用。其中每个主要功能之间存在一定的弱联系，如 3.1 图所示：“爬取信息”模块向“课表生成”模块会提供课程信息；“代办事项管理”模块为“待办事项提醒”模块提供代办信息；“个性化设置”模块分为平行的提示音乐、字体颜色、课表背景三个子模块，分别对不同的上游模块提供个性化服务

3.3、主界面模块（main.py）

3.3.1、设计图



3.3.2、功能描述

主界面模块作为本程序的核心模块，主要负责了将各个模块串联起来，将各个功能集合到一起，形成一个整体的任务。这一模块的功能主要有：显示主菜单界面、将各个模块的函数封装进按钮中，按下相应的按钮调用对应的功能函数、维持整个程序的正常运行等。

3.3.3、输入数据

用户输入：鼠标点击输入，当用户将鼠标移动到某一按钮上并左键单击时视为有效输入。

默认背景文件输入：从 py 文件的同目录下获取北邮背景图和小图标文件作为系统默认输入文件，程序运行时自动获取该输入文件。

3.3.4、输出数据

函数调用信号：当获得用户鼠标点击输入时，主程序界面产生一个对应的函数调用信号，表现为开始执行程序中对应函数的功能。

上层弹窗：当获得用户鼠标点击输入时，主程序也可根据对应选择产生一个弹窗，表现为在主程序窗口的上层弹出一个上层窗口。

图片显示：当获得默认背景文件输入时，模块产生一个画布数据结构并放置在主程序界面窗口上，表现为显示窗口的北邮背景图片。

3.3.5、数据设计

```
window = Tk()  #创建主菜单界面窗体
size_geo = '%dx%d+%d+%d' % (width, height, (screenwidth-width)/2,(screenheight-height)
/2-50)  #获取当前操作电脑的屏幕尺寸并存储
buptlabel = Label(window, image=buptphoto)  #存储打开的北邮背景图作为标签
buptback = Canvas(window, bg="blue", height=450, width=690)  #用于为背景图提供画布，以此将北邮背景图贴到主界面上。
kf = Toplevel()  #创建一个开发人员上级窗口
l = tk.Label(kf,text=' 孙 泽 凯 -2020212180\n\n 张 扬 -2020212185\n\n 王梓安-2020212212',
```

bg='#8000ff',font=('宋体', 18, 'bold'),width=200,height=200, fg = '#ffffff',padx=10, pady=15, borderwidth=10, relief="sunken") #用于记录开发人员窗口相关信息

button0 = tk.Button(window,text='邮学小帮手',bg='#8000ff',font=('华文琥珀', 35, 'bold'),width=25,height=1, fg = '#ffffff',padx=10, pady=15, borderwidth=10, relief="sunken",command=kaifa) #用于记录开发人员按钮信息

bt1 = tk.Button(wd1, text='背景\n更换', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh1).pack(side = LEFT,expand = True) #记录个性化窗口中背景更换按钮信息

bt2 = tk.Button(wd1, text='颜色\n选择', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh2).pack(side = LEFT,expand = True) #记录个性化窗口中颜色选择按钮信息

bt3 = tk.Button(wd1, text='提示音\n更换', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh3).pack(side = LEFT,expand = True) #记录个性化窗口中提示音更换按钮信息

bt4 = tk.Button(wd1, text='桌面\n挂件', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),width=6, height=2, command=gxh4).pack(side = LEFT,expand = True) #记录个性化窗口中桌面挂件按钮信息

button1 = tk.Button(window, text='获取\n课程信息', activebackground='yellow', bg = '#9301fe',font = ('隶书',12),fg='#ffffff', cursor="watch",width=8, height=3, command =command_creep) #记录主窗口获取课程信息按钮信息

button2 = tk.Button(window, text = '显示\n课程表', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff', cursor="watch",width=8,height=3,command= command_show_calendar) #记录主窗口显示课程表按钮信息

button3 = tk.Button(window, text = '设置\n待办事项', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff', cursor="watch",width=8,height=3,command=

command_todo)　#记录主窗口设置待办事项按钮信息

button4 = tk.Button(window, text = '启用\n提醒系统', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=

command_alarm)　#记录主窗口启用提醒系统按钮信息

button5 = tk.Button(window, text = '个性化\n设置', activebackground='yellow', bg = '#9301fe', font = ('隶书',12),fg='#ffffff',
cursor="watch",width=8,height=3,command=

gexinghua)　#记录主窗口个性化设置按钮信息

以上所有数据结构均不需要以文件形式长期存储，程序运行结束后所有占用空间均会被释放。

3.3.6、算法和流程

由于使用了 python 自带的 tkinter 函数库，我们将控制函数封装进了 tkinter 的 button 控制函数中，当获得用户鼠标输入后，系统就会产生函数调用信号，自动调用相应的函数。

Tkinter 函数库是 Python GUI 开发的一个标准库，我们编写的 Python 代码会调用内置的 Tkinter，Tkinter 封装了访问 Tk 的接口；而 Tk 是一个图形库，支持多个操作系统，使用 Tcl 语言开发；Tk 会调用操作系统提供的本地 GUI 接口，完成最终的 GUI。

所以，我们的代码只需要调用 Tkinter 提供的接口就可以了，并不需要额外的计算数据。

如果希望进一步了解 tkinter 可以参考以下文章：

https://blog.csdn.net/qq_40494873/article/details/123240187

http://c.biancheng.net/tkinter/

3.3.7、函数说明

Kaifa():

```
def kaifa():
    kf = Toplevel()
    kf.title('开发人员')
    kf.iconbitmap('233.ico')
    width = 300
    height = 200
    # 窗口居中，获取屏幕尺寸以计算布局参数，使窗口居屏幕中央
    screenwidth = kf.winfo_screenwidth()
    screenheight = kf.winfo_screenheight()
    size_geo = '%dx%d+%d+%d' % (width, height, (screenwidth - width) / 2, (screenheight - height) / 2 - 50)
    kf.geometry(size_geo)
    kf.resizable(0, 0)
    kf.config(background="#8000ff")
    l = tk.Label(kf,
    text='孙泽凯-2020212180\n\n张扬-2020212185\n\n王梓安-2020212212',bg='#8000ff',font=('宋体', 18, 'bold'),
    width=200,height=200, fg = '#ffffff',padx=10, pady=15, borderwidth=10, relief="sunken").pack()
```

该函数负责定义开发人员信息按钮的控制函数，格式如上，不需输入参数，访问全局变量 window，定义一个局部变量 kf 和上图中其他临时变量，无返回值，当按下对应按钮时调用该函数作为按钮的控制函数。

command_creep():

```
def command_creep():
    # 使用 curselection来选中文本
    try:
        creep.creeper()
    except Exception as e:
        e = '发现一个错误'
        messagebox.showwarning(e,'请按照正确步骤使用程序')
```

该函数负责定义获取课程信息按钮的控制函数，格式如上，不需要输入参数，不访问任何全局变量，定义一个局部变量 e 存储报错信息用于出错处理。无返回值，无数据计算，当按下对应按钮时调用该函数作为按钮的控制函数。

command_show_calendar():

```
def command_show_calendar():
    try:
        bg_path.set(choose_back.bg_change())
        class_color.set(choose_color.color_return())
        # pic_way = bg_path.get()
        show_calendar.show_calendar(class_color.get(),bg_path.get())
    except:
        show_calendar.show_calendar(class_color.get())
```

该函数负责定义显示课程表按钮的控制函数，格式如上，不需要输入参数，访问全局变量 bg_path 和 class_color，不定义任何局部变量，无返回值，无数据计算，当按下对应按钮时调用该函数作为按钮的控制函数。

command_todo():

```
def command_todo():
    try:
        bg_path.set(choose_back.bg_change())
        Todolist.todo_main(bg_path.get())
    except:
        Todolist.todo_main()
```

该函数负责定义设置待办事项按钮的控制函数，格式如上，不需要输入参数，不访问任何全局变量，不定义局部变量，无返回值，无数据计算，当按下对应按钮时调用该函数作为按钮的控制和函数。

command_alarm():

```
def command_alarm():
    try:
        alarm.alarm_system(0,choose_music.music_change())
    except:
        print('error')
```

该函数负责定义开启提醒系统按钮的控制函数，格式如上，不需要输入参数，不访问任何全局变量，不定义局部变量，无返回值，无数据计算，当按下对应按钮时调用该函数作为按钮的控制和函数。

Gexinghua():

```
def gexinghua():
    wd1 = Toplevel()
    width = 500
    height = 300
    # 窗口居中，获取屏幕尺寸以计算布局参数，使窗口居屏幕中央
    screenwidth = wd1.winfo_screenwidth()
    screenheight = wd1.winfo_screenheight()
    size_geo = '%dx%d+%d+%d' % (width, height, (screenwidth - width) / 2, (screenheight - height) / 2 - 50)
    wd1.geometry(size_geo)
    wd1.resizable(0, 0)
    wd1.iconbitmap('233.ico')
    wd1.config(background="#8000ff")
    wd1.title('个性化设置')
    def gxh1():
        choose_back.file_saves()
        wd1.quit()
    def gxh2():
        choose_color.color_choose()
        wd1.quit()
    def gxh3():
        choose_music.music_choose()
        wd1.quit()
    def gxh4():
        desktop_app.haha()
        wd1.quit()
    bt1 = tk.Button(wd1, text='背景\n更换', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),
            width=6, height=2, command=gxh1).pack(side = LEFT,expand = True)
    bt2 = tk.Button(wd1, text='颜色\n选择', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),
            width=6, height=2, command=gxh2).pack(side = LEFT,expand = True)
    bt3 = tk.Button(wd1, text='提示音\n更换', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),
            width=6, height=2, command=gxh3).pack(side = LEFT,expand = True)
    bt4 = tk.Button(wd1, text='桌面\n挂件', activebackground='yellow', bg = '#e2ad41', font = ('隶书',15),
            width=6, height=2, command=gxh4).pack(side = LEFT,expand = True)
    wd1.mainloop()
```

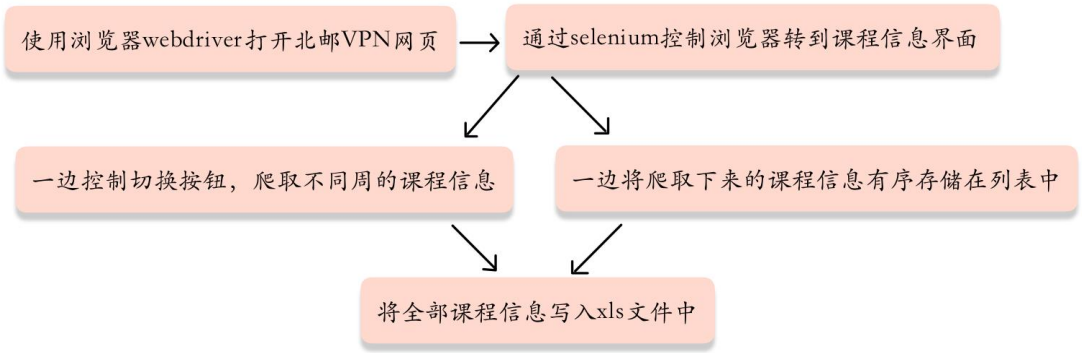
该函数负责定义个性化设置按钮的控制函数，当按下按钮时会弹出上层窗口，供用户选择需要的个性化设置功能；格式如上，访问全局变量 window，定义局部变量 wd1、bt1 等用作上层窗口的存储变量，无返回值，无数据计算，窗口的设置思路与主菜单界面设置思路一致，当按下个性化设置按钮时调用该函数作为按钮的控制函数。

3.3.8 全局数据结构与该模块的关系

实际上该文件中定义了三个全局数据结构，window 作为窗口存储变量、bg_path 和 class_color 作为背景图和课程颜色的信息存储变量，该模块主要依存 window 该窗口存储变量作为基础，所有的操作都需要建立在新建的窗口之上，所有的按钮布局操作和背景操作都是在针对 window 进行修改；而另外两个全局变量则是数据在别的模块接口中传输的中继变量，在本模块只起到中继作用。

3.4、课程信息获取模块（creep.py）

3.4.1、设计图



3.4.2、功能描述

顾名思义，本模块的主要功能是获取课程表信息；通过 python 的爬虫工具打开浏览器，通过 selenium 一边控制浏览器打开网页切换网页来定位课程表，一边把爬取下来的课程信息有序存储在列表中，最后把获取到的课程信息存储到一个 xls 文件中。

3.4.3、输入数据

用户输入：需要用户键盘输入北邮 VPN 和教务系统的账号密码，这一步需要用户自己在 selenium 打开的浏览器对应页面上在 60s 之内输入，用户输入信息并回车浏览器自动跳转到下一个页面视为有效输入。

3.4.4、输出数据

Xls 文件输出：模块正常运行完毕会产生一个 xls 文件输出，文件中以表格形式存储了用户本学期所有周的课程信息，包括上课时间、课程名称、授课教授、上课地点等。

3.4.5、数据设计

botton1 = byr_page.find_element('xpath','/html/body/div/div/div/div/div

[3]/div/div[2]/div/
div[1]/div/div[1]/div/div[1]/a') #存储爬取到的按钮信息
botton2 = byr_page.find_element('xpath','/html/body/div[2]/div[3]/div[2]/div/div[1]/a/p') #存储第二个爬取到的按钮信息
button3 = byr_page.find_element('xpath','/html/body/div/div[2]/div[1]/div[2]/div/div/div[1]/
p') #存储第三个爬取到的按钮信息
fm1 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe') #存储爬取到的首个 frame 信息
fm2 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe[2]') #存储爬取到的第二个 frame 信息
kebiao = [] #建立列表，存储爬取到的所有课程信息
class_data = [] #建立列表，存储爬取到的每周课程信息
day_data = [] #建立列表，存储爬取到的每天课程信息
以上数据结构中，kebiao 列表内的数据内容需要输出为 xls 文件保存，默认输出保存文件名为“课程表.xls”，默认保存在项目文件同目录下。

3.4.6、算法和流程

事实上，本模块的输入数据与输出数据没有直接的关系，输出数据并不是由输入数据计算或者做任何处理得来的，输入的用户名密码等数据只是保证了爬虫程序的正常运行，输出的课程表文件是爬虫正常运行结束的产物。

接下来介绍爬虫工具爬取课程信息的爬取流程：

导入爬虫所需要的函数包

```
import re
# import requests
import time
import selenium
from selenium import webdriver
import openpyxl
from openpyxl import Workbook
from openpyxl.reader.excel import load_workbook
import csv
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
```

1、使用 edgedriver 打开 edge 浏览器，打开北邮 VPN 网页后等待用户自行输入用户名密码，待正确输入后自动跳转到信息门户界面，再次等待用户自行输入信息门户用户名密码后自动跳转到教务系统登录界面，再次等待用户输入教务系统用户名密码后跳转到最后的课程表信息界面。

```
url = 'https://webvpn.bupt.edu.cn'
byr_page = webdriver.Edge()
byr_page.get(url)
byr_page.maximize_window()
WebDriverWait(byr_page,60,0.5).until(EC.presence_of_element_located((By.XPATH, '/html/body/div/div/div/div/div[3]/div/div[2]/div/div
botton1 = byr_page.find_element('xpath','/html/body/div/div/div/div/div[3]/div/div[2]/div/div[1]/div/div[1]/div/div[1]/a')
botton1.click()

byr_page.switch_to.window(byr_page.window_handles[1])
WebDriverWait(byr_page,60,0.5).until(EC.presence_of_element_located((By.XPATH, '/html/body/div[2]/div[3]/div[2]/div/div[1]/a/p')))
botton2 = byr_page.find_element('xpath','/html/body/div[2]/div[3]/div[2]/div/div[1]/a/p')
botton2.click()

byr_page.switch_to.window(byr_page.window_handles[2])
WebDriverWait(byr_page,60,0.5).until(EC.presence_of_element_located((By.XPATH, '/html/body/div[3]/div/div[2]/iframe')))

fm1 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe')
byr_page.switch_to.frame(fm1)

button3 = byr_page.find_element('xpath','/html/body/div/div[2]/div[1]/div[2]/div/div/div[1]/p')
button3.click()

byr_page.switch_to.window(byr_page.window_handles[2])
fm2 = byr_page.find_element('xpath','/html/body/div[3]/div/div[2]/iframe[2]')
byr_page.switch_to.frame(fm2)
time.sleep(5)
```

2、一边通过 selenium 控制浏览器翻页，分别翻到不同周的课程表；一边获取每一周的课程信息并分级存储到 day_data、class_data、kebiao 这三个列表中暂时存储。

```
kebiao = []
for k in range(16):
    week_path = str('/html/body/div/div/form[2]/div[2]/select[1]/option['+str(k+2)+']')
    week = byr_page.find_element('xpath', week_path)
    week.click()
    time.sleep(2)
    class_data = []
    for i in range(7):
        day_data = []
        for j in range(14):
            classes_path = str('/html/body/div/div/form[2]/table[1]/tbody/tr['+str(j+2)+']/td['+str(i+1)+']/div[2]')
            try:
                data = byr_page.find_element('xpath',classes_path)
                day_data.append(data.text)
            except:
                day_data.append(' ')
        class_data.append(day_data)
    kebiao.append(class_data)
```

3、读取 kebiao 列表中的所有信息，新建一个 xls 文件“课程表.xls”，将 kebiao 中的所有信息存储到该文件中，存储时每一列存储一天的课程表，每一行存储一节课的数据，共存储 14 行、7x16 列数据。

```
kecheng = openpyxl.Workbook()
kecheng.active.title_path = "课程表"
kecheng.create_sheet('课程表')
sheet1 = kecheng['课程表']

for k in range(len(kebiao)):
    for i in range(7):
        for j in range(14):
            sheet1.cell(j+1, i+1+(k*7)).value = kebiao[k][i][j]


kecheng.save("课程表.xlsx")
```

3.4.7、函数说明

整个文件中所有的代码被封装进了一个函数 creeper():中，这就使得当主程序在调用这一个函数的时候就可以直接执行本文件中的所有代码，有利于主程序与本模块之间接口的建立与函数调用。

该函数主要功能就是本模块的主要功能：获取课程信息并存储为 xls 文件，格式在上文已经给出，不需要传入任何参数，不调用任何全局变量，所有局部变量也在数据结构说明中给出，无返回值，算法流程在 3.4.6 中已说明，使用时需要在 60s 内输入用户名和密码，除此之外无任何使用约束。

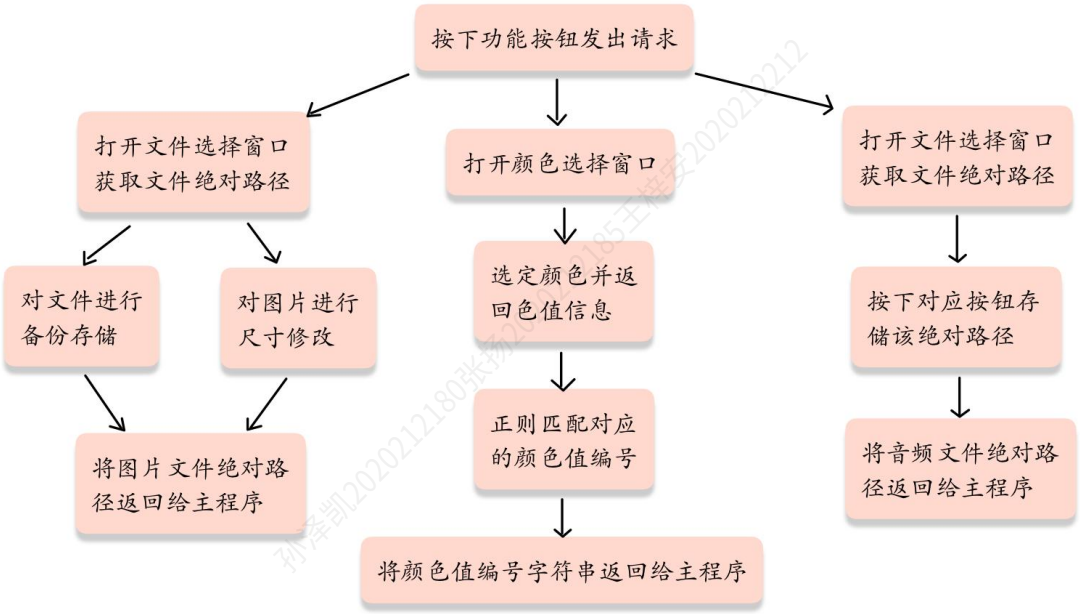
3.4.8 全局数据结构与该模块的关系

该模块未访问任何全局数据结构作为输入，但是本模块会长生一个新的全局数据结构，即爬取信息获得的“课程表.xls”，后续显示课程表和桌面小挂件模块都需要用到该数据结构作为输入文件信息。

3.5、自定义模块（choose_back.py,choose_color.py,choose_music.py）

自定义模块有三个功能，由于前三个功能思路较为相似，此处先行介绍。

3.5.1、设计图



3.5.2、功能描述

本部分的主要功能有三个：第一是更换显示课程表窗口的背景图片，用户选择本电脑上的文件就可以对该文件进行文件备份、尺寸修改和设为背景图等操作；第二是更换课程字体的颜色，打开颜色选择窗口选定一个颜色后将该颜色值编号传给主程序就可以更改课程字体颜色；第三是更换提示音，用户选择一个本地音频文件后就可以将该文件设为提示音，后续的上课提醒和 ddl 提醒就可以播放该音频文件。

3.5.3、输入数据

用户输入：鼠标点击输入，当用户将鼠标移动到某一按钮上并左键单击时视为有效输入。

文件输入：通过 tkinter 中 filedialog 函数包中封装好的 askopenfilename():函数就可以打开文件获取窗口，选定一个文件点击确定后就可以返回这个文件在电脑中的绝对路径。

3.5.4、输出数据

功能一输出：更换背景图功能的输出有两个方面：如果用户选择了文件备份功能，该模块就会把原文件复制一份后输出存储到用户选定的目录下；如果用户希望将该文件设为课程表背景图，该模块就会将选定文件的绝对路径返回给主程序，输出数据类型为 string 类型。

功能二输出：颜色选择功能的输出为一个颜色代码，用户选定一个颜色后模块就会自动将该颜色的颜色代码转化成一个字符串，并将该 string 类型的颜色代码返回给主程序。

功能三输出：更换提示音功能的输出为提示音文件的绝对路径，用户选择一个文件后模块就会将该文件的绝对路径返回给主程序模块，输出数据类型为 string 类型。

3.5.5、数据设计

window = tk.Toplevel()　#记录新建的背景选择上级窗口数据结构

tk.Button(window, text='选择文件', command=open_img).grid(row=1, column=1, padx=5, pady=5)　#记录选择文件按钮信息

tk.Button(window, text='文件备份', command=save_png).grid(row=2, column=1, padx=5, pady=5) #记录文件备份按钮信息

tk.Button(window, text='尺寸修改', command=fixed_size).grid(row=3, column=1, padx=5, pady=5)　#记录尺寸修改文件信息

root = tk.Tk()　#记录新建的颜色选择上级窗口

lb=tk.Label(root,text='',font=('宋体',10))　#记录需要放置在窗口上的 label 标签

tk.Button(root, text="点击选择颜色", command=callback, width=14, bg='#808080') #记录选择颜色按钮信息

tk.Button(root, text='设为课程字体颜色',command=kebiaocolor, width=14, bg='#808080') #记录设置课程字体颜色按钮信息

window = tk.Toplevel() #记录新建的提示音选择上级窗口

tk.Button(window, text='选择音频', command=open_music).grid(row=1, column=1, padx=5, pady=5) #记录选择音频按钮信息

tk.Button(window, text='设置音频', command=save_music).grid(row=2, column=1, padx=5, pady=5) #记录设置音频按钮信息

以上所有数据结构均不需要以任何文件形式长期存储，所占用空间会在程序运行结束后立即释放。

3.5.6、算法和流程

详细描述根据输入数据产生输出数据的算法和流程。

本部分也使用了 python 自带的 tkinter 函数库，我们同样将控制函数封装进了 tkinter 的 button 控制函数中，当获得用户鼠标输入后，系统就会产生函数调用信号，自动调用相应的函数。

而根据输入的文件数据来获取该文件绝对路径的算法流程也同样被封装进了 tkinter 的 filedialog.askopenfilename()这个函数中，所以综合上述情况，我们的代码只需要调用 Tkinter 提供的接口就可以了，并不需要额外的计算数据。

Tkinter 函数库是 Python GUI 开发的一个标准库，我们编写的 Python 代码会调用内置的 Tkinter，Tkinter 封装了访问 Tk 的接口；而 Tk 是一个图形库，支持多个操作系统，使用 Tcl 语言开发；Tk 会调用操作系统提供的本地 GUI 接口，完成最终的 GUI。

如果希望进一步了解 tkinter，了解该函数库根据输入数据获得输出数据的算法流程，可以参考以下文章：

https://blog.csdn.net/qq_40494873/article/details/123240187

http://c.biancheng.net/tkinter/

3.5.7、函数说明

具体说明模块中的各个函数，包括函数名称及其所在文件，功能，

格式，参数，全局变量，局部变量，返回值，算法说明，使用约束等。
Choose_back.py:

1、open_img():

```
def open_img():
    try:
        global img
        global filepath
        filepath = filedialog.askopenfilename()# 打开文件，返回该文件的完整路径
        filename.set(filepath)
        img = Image.open(filename.get())

    except Exception as e:
        print("您没有选择任何文件",e)
```

该函数可以打开文件选择界面并将用户选择的一个文件的绝对路径记录到全局变量 filepath 中，将打开的文件暂存到全局变量 img 中。格式如上，不需要传入参数，无局部变量，无返回值，按下相应按钮调用本函数作为按钮控制函数。

2、save_png():

```
def save_png():
    try:
        filetypes = [("PNG","*.png"), ("JPG", "*.jpg"), ("GIF","*.gif"),("txt files","*.txt"),('All files','*')]
        filenewpath = filedialog.asksaveasfilename(title='保存文件',
                                                   filetypes=filetypes,
                                                   defaultextension='.png',
                                                   initialdir='C:/Users/huawei/Desktop' )
        path_var.set(filenewpath)
        # 保存文件
        # print(str(path_var.get()))
        img.save(str(path_var.get()))
    except Exception as e:
        print(e)
```

该函数可以将打开的文件复制一个备份出来，并存储到用户选定的文件目录下。格式如上，不需要传入参数，需要访问全局变量 img，无局部变量，无返回值，按下相应按钮调用本函数作为按钮控制函数。

3、fixed_size():

```
def fixed_size():
    # im = Image.open(file)
    width = 1300
    height = 700
    out = img.resize((width, height), Image.ANTIALIAS)
    out.save(filepath)
```

该函数可以对打开的图片文件进行尺寸修改，默认修改尺寸为 1300x700 与显示课程表窗口大小对应。格式如上，不需要传入参数，需要访问全局变量 img，无局部变量，无返回值，按下相应按钮调用本函数作为按钮控制函数。

4、bg_change():

```
def bg_change():
    return filepath
```

如图，该函数功能较为简单，只是做了一个返回 filepath 值的功能，该函数主要由主程序进行调用。不传入参数，访问全局变量 filepath，无局部变量，返回值为 filepath 的值为 string 类型。

Choose_music.py:

1、open_music():

```
def open_music():
    try:
        global img
        global filepath
        filepath = filedialog.askopenfilename() # 打开文件，返回该文件的完整路径
        filename.set(filepath)
        global mark
        mark = 1
    except Exception as e:
        print("您没有选择任何文件",e)
        mark = 0
```

该函数功能同 open_img()相似，可以打开文件选择界面并将用户选择的一个文件的绝对路径记录到全局变量 filepath 中，将打开的文件暂存到全局变量 img 中。格式如上，不需要传入参数，无局部变量，无返回值，按下相应按钮调用本函数作为按钮控制函数。

2、save_music():

```
def save_music():
    try:
        if mark == 1:
            music_var.set('已设为提示音！')
        else:
            music_var.set('请选择一个文件！')
    except Exception as e:
        print(e)
```

该函数主要功能是控制窗口上的文本框显示不同内容，如果用户已经选择一个音频文件再按下按钮就会显示“已设为提示音”的文字提示，如果未选择文件再按按钮就会显示“请选一个文件”的文字提示。格式如上，不需要传入参数，无局部变量，无返回值，按下相应按钮调用本函数作为按钮控制函数。

3、music_change():

```
def music_change():
    return filepath
```

如图，该函数功能较为简单，只是做了一个返回 filepath 值的功能，该函数主要由主程序进行调用。不传入参数，访问全局变量 filepath，无局部变量，返回值为 filepath 的值为 string 类型。

Choose_color.py:

1、kebiaocolor():

```
def kebiaocolor():
    global kebiao_color
    kebiao_color = str(colorvalue)
    kebiao_color = re.findall(findcolor, kebiao_color)[0]
```

该函数功能为从字符串 colorvalue 中通过正则表达式匹配的方式截取出颜色代码来，并将颜色代码字符串赋值给全局变量 kebiao_color。不传入参数，访问全局变量 kebiao_color，无局部变量，无返回值，按下相应按钮调用该函数作为按钮控制函数。

2、callback():

```
def callback():
    # 打开颜色对话款
    global colorvalue
    colorvalue = tk.colorchooser.askcolor()
    # 在颜色面板点击确定后，会在窗口显示二元组颜色值
    lb.config(text='颜色值:'+ str(colorvalue))
```

该函数主要功能为唤出颜色选择窗口供用户选择一个颜色，并将该颜色详细信息赋值给全局变量 colorvalue。不传入参数，访问全局变量 colorvalue，无局部变量，无返回值，按下相应按钮调用该函数作为按钮控制函数。

3、color_return():

```
def color_return():
    return kebiao_color
```

如图，该函数功能也较为简单，只是做了一个返回 kebiao_color 值的功能，该函数主要由主程序进行调用。不传入参数，访问全局变量 kebiao_color，无局部变量，返回值为 string 类型。

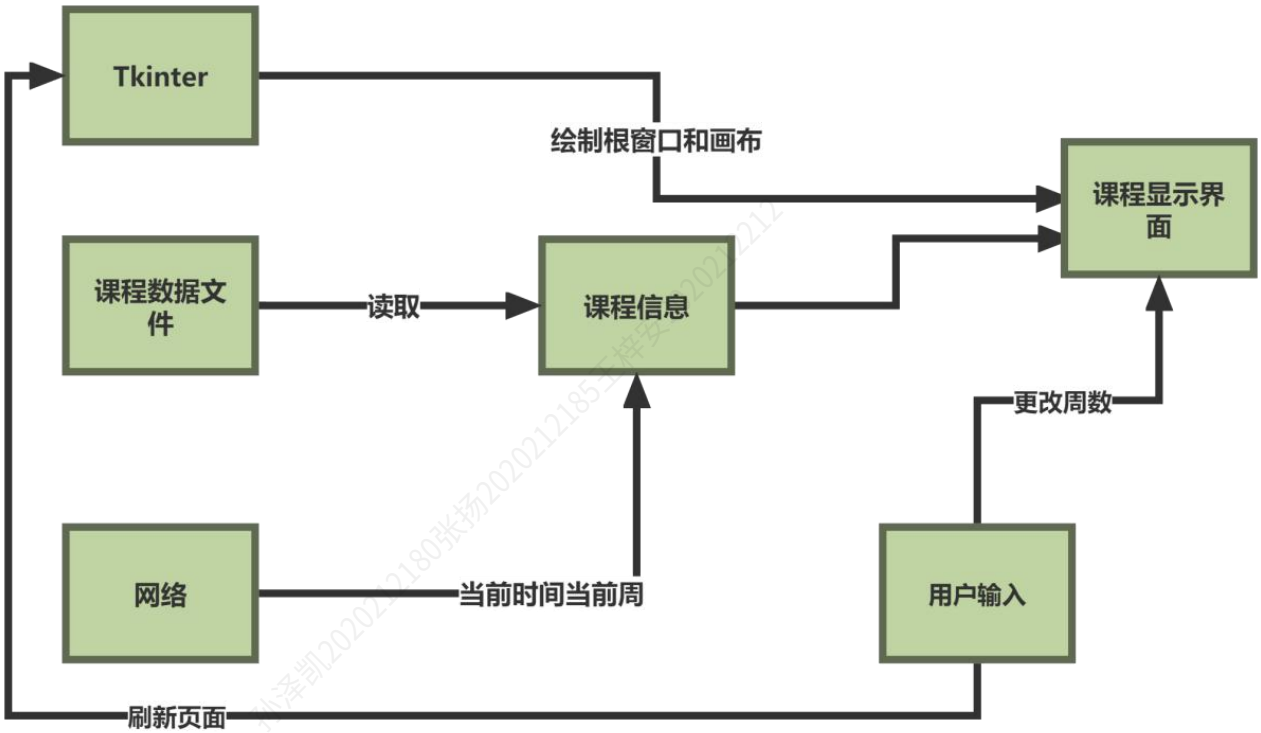
3.5.8 全局数据结构与该模块的关系

该模块不访问任何全局数据结构，但是主程序中 bp_path 和 class_color 这两个全局变量的值需要由该模块中 bg_change()和 color_return()两函数的返回值赋值。所以该模块不直接访问全局数据结构，而是通过函数的返回值来为全局数据结构赋值。

3.6 课程展示模块（show_calendar.py）

本模块的主要获取当前周数并显示当前周和任意周的课程信息，当启动课程表显示模块时，程序自动获取当前周数并显示当前周课程；点击右侧周数按钮，可以显示任意周的课程信息和跳转到当前周

3.6.1、设计图



3.6.2、功能描述

当启动课程表显示模块时，程序自动获取当前周数并显示当前周课程；点击右侧周数按钮，可以显示任意周的课程信息和跳转到当前周

3.6.3、输入数据

输入数据 1：通过网络获取当前时间，以及当前周信息
输入数据 2：通过读取文件获取课程表信息
输入数据 3：通过用户操作获取当前显示的周数

3.6.4、输出数据

各课程的上课时间、课程表信息

3.6.5、数据设计

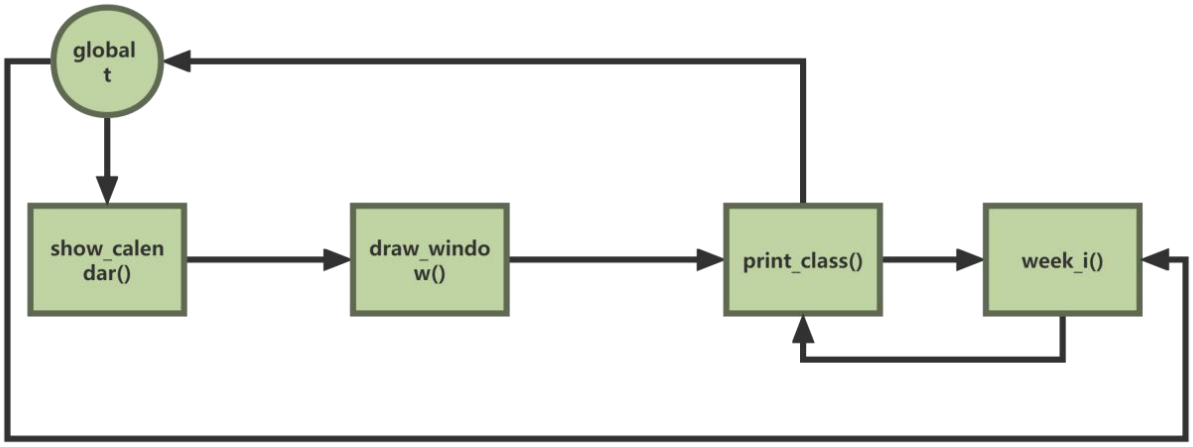
1)列表用于储存课程时间和课程信息

```
                "第1节/08:00-08:45"，"第2节/08:50-09:35"，"第3节/09:50-10:35"，"第4节/10:40-11:25"，
                "第5节/11:30-12:15"，"第6节/13:00-13:45"，"第7节/13:50-14:35"，"第8节/14:45-15:30"，
                "第9节/15:40-16:25"，"第10节/16:35-17:20"，"第11节/17:25-18:10"，"第12节/18:30-19:15"，
class_info = ["第1节/08:00-08:45"，"第2节/08:50-09:35"，"第3节/09:50-10:35"，"第4节/10:40-11:25"，"第5节/11:30-12:15"，
week = ["星期一"，"星期二"，"星期三"，"星期四"，"星期五"]
```

2)设计指针实现不同周界面的刷新

```
global t
if not week:
    t_new = print_class(get_current_week(), cv, class_info,color)
else:
    for each_ in t:
        cv.delete(each_)
    t_new = [cv.create_text(50, 30, text="第{}周".format(week), fill='#FF1493', font=('微软雅黑', 15, 'bold'))]
    for day in range(5):
        for line in range(14):
            t_new.append(
                cv.create_text(150 + day * 200, 80 + line * 45, fill=color,
                               text=class_info[line][(week - 1) * 7 + day],
                               font=('微软雅黑', 10, 'bold')))  # 课程文字
return t_new
```

3.6.6、算法和流程



如图，声明全局变量 t，调用 draw_window 函数绘制基本界面，第一次调用 print_class 为书写当前周课程信息，传回的指针赋给 t，之后定义 16 个函数 week_i，用于绘制各周的课程信息并植入到按钮中，按钮触发时可以获得 t 指向的文字信息并删除，之后把新的内容打印并将其地址传给 t，便于下次刷新时删除

3.6.7、函数说明

1）show_calendar(colour,pic_way=''):模块主题函数，封装了本模块的主要功能，可以直接调用，只需传入背景图路径与课程文字颜色字符串。

```
def show_calendar(colour,pic_way=''):
    # create window
    root = Toplevel()
    root.config(bg='#BFEFFF')
    root.title("邮学小帮手")
    root.geometry('1300x700')
    class_info = read_class_info()
    # root.iconbitmap('之后的图标路径')
    cv = Canvas(root, bg="#BFEFFF", width=1200, height=700)  # create canvas
    global t

    draw_window(pic_way, cv)
    t = print_class(0, cv, class_info,colour)

    weeks_F = []
    for i in range(17):
        exec('''def week_{}(cv=cv, class_info=class_info, color = colour):
                    global t
                    t = print_class({}, cv, class_info, color)'''.format(i, i))
        weeks_F.append("week_{}".format(i))
    temp = 1

    Button(root, text="当前周".format(temp), command=eval('week_0'), fg='#FF7256', activeforeground='#CD6600',
           font=('微软雅黑', 18, 'bold')).place(x=1205, y=15)
    for each in weeks_F[1:]:...

    cv.pack(anchor='w')
    root.mainloop()
```

2）draw_window(pic_way, cv)

绘制基本图形界面，包括背景图，课程时间信息和周次，星期信息。

```
def draw_window(pic_way, cv):
    if pic_way:
        img_open = Image.open(pic_way)
        global img_png
        img_png = ImageTk.PhotoImage(img_open)
        cv.create_image(600, 350, image=img_png)
    temp = ["星期一", "星期二", "星期三", "星期四", "星期五"]
    class_times = ["第1节\n08:00-08:45", "第2节\n08:50-09:35", "第3节\n09:50-10:35", "第4节\n10:40-11:25", "第5节\n11:30-12:15",
                   "第6节\n13:00-13:45", "第7节\n13:50-14:35", "第8节\n14:45-15:30", "第9节\n15:40-16:25", "第10节\n16:35-17:20",
                   "第11节\n17:25-18:10", "第12节\n18:30-19:15", "第13节\n19:20-20:05", "第14节\n20:10-20:55",
                   "第14节\n20:10-20:55"]
    for day in range(5):
        cv.create_text(150 + day * 200, 20, text=week_names[day], fill='#FF1493', font=('微软雅黑', 10, 'bold'))  # 星期文字
        for line in range(14):
            if day == 0:
                cv.create_text(50, 80 + line * 45, text=class_times[line], fill='#FF1493',
                               font=('微软雅黑', 10, 'bold'))  # 节次文字
```

3）read_class_info()

读取课程信息，对课程信息做一定的处理，以特定格式储存在多维列表中，便于之后使用。

```
def read_class_info():
    workbook = openpyxl.load_workbook('课程表.xlsx')
    sheet = workbook['课程表']
    class_info_ = [[]]
    for r in range(14):
        for c in range(112):
            temp = sheet.cell(row=r + 1, column=c + 1).value
            one = temp.find('\n')
            two = temp.find('\n', one + 1)
            if not one == -1:
                temp = temp[:two]
            class_info_[r].append(temp)
        class_info_.append([])
    return class_info_
```

4) get_current_week()

通过 datetime 库，获取当前日期并计算出当前周数，从而可以打印当前周信息。

```
def get_current_week():
    date0 = '2022-02-28'
    d0 = datetime.datetime.strptime(date0, '%Y-%m-%d')
    date1 = time.strftime("%Y-%m-%d", time.localtime())
    d1 = datetime.datetime.strptime(date1, '%Y-%m-%d')
    return int((d1 - d0).days / 7) + 1
```

5) print_class(week, cv, class_info,color)

打印课程文字信息，返回指向课程文字 Tkinter 对象的指针，访问当前课程文字指针 t 并清楚，从而完成刷新操作

```
def print_class(week, cv, class_info,color):
    global t
    if not week:
        t_new = print_class(get_current_week(), cv, class_info,color)
    else:
        for each_ in t:
            cv.delete(each_)
        t_new = [cv.create_text(50, 30, text="第{}周".format(week), fill='#FF1493', font=('微软雅黑', 15, 'bold'))]
        for day in range(5):
            for line in range(14):
                t_new.append(
                    cv.create_text(150 + day * 200, 80 + line * 45, fill=color,
                                   text=class_info[line][(week - 1) * 7 + day],
                                   font=('微软雅黑', 10, 'bold')))  # 课程文字
    return t_new
```

3.6.8 全局数据结构与该模块的关系

访问了全局数据结构课程信息表格。访问了主程序根窗口和按钮。

3.7、提醒模块(Alarm.py)

本模块主要功能是提供提醒功能，可以调用存储系统接口获得相关课程提醒和待办提醒数据，在到达相应的触发时间时，进行声音和弹窗的提醒功能，并支持自定义个性化提醒方式。

3.7.1、设计图

3.7.2、功能描述

到达课前几分钟或待办事项前几分钟的触发时间时，进行声音和弹窗的提醒功能，并支持自定义个性化提醒方式。

3.7.3、输入数据

通过待办事项模块获取的待办事项信息；
通过自定义系统获取的提前时间信息；
通过课程表展示模块获得的课程表信息。

3.7.4、输出数据

特定时间的弹窗、音乐播放、提醒文字信息。

3.7.5、数据设计

使用列表储存相关时间信息，多用到“XXXX-XX-XX”格式、

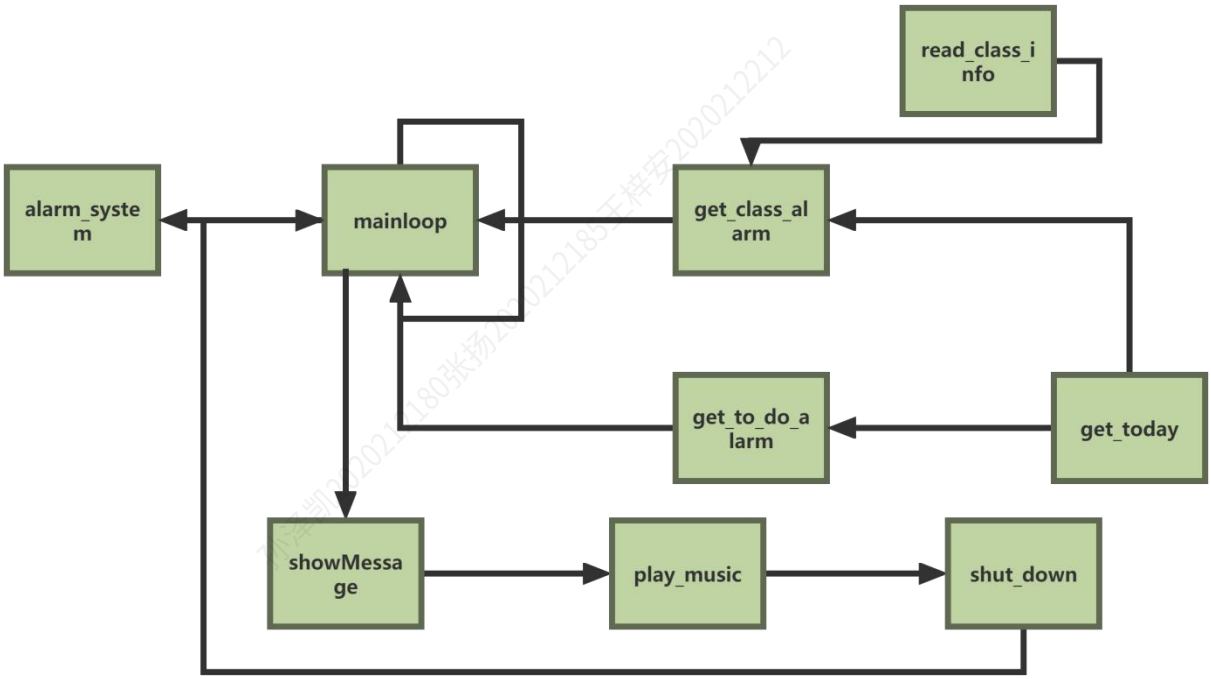
“XXXX-XX”格式的信息；以及 datetime 库中的时间戳类型信息和时间对象。

```
class_times = ["08:00", "08:50", "09:50", "10:40", "11:30", "13:00", "13:50", "14:45", "15:40", "16:35", "17:25",
               "18:30", "19:20", "20:10"]
alarm_times = []
today = get_today()
for i in range(14):
    if have_class[i][today]:
        alarm_times.append(class_times[i])

ahead = 60 * ahead
for i in range(len(alarm_times)):
    date = time.strftime("%Y-%m-%d", time.localtime())
    alarm_times[i] = time.mktime(time.strptime(date + ' ' + alarm_times[i], "%Y-%m-%d %H:%M")) - ahead

return alarm_times
```

3.7.6、算法和流程



如图，操作 alarm_system 时新建一个线程，后台进入进入 mainloop，每次循环，调用 get_class_alarm 和 get_to_do_alarm 获得当前需要提醒的事件，其中有 get_today 和 read_class_info 为上述两个函数提供支持，触发提醒事件时，进入 showMessage，开始播放音乐 play_music，按按钮时触发 shut_down 关闭窗口并再次进入 mainloop。

3.7.7、函数说明

1）def alarm_system(ahead, music):

新建一个后台线程，用于执行整个提醒系统，进入循环确认过程，传入 ahead 为提前提醒的时间，music 为播放音乐的路径。

```
def alarm_system(ahead, music):
    _thread.start_new_thread(mainloop, (ahead, music))
```

2）def mainloop(ahead, music):

提醒系统主循环函数，不断确认当前时间是否有需要提醒的事情，传入 ahead 为提前提醒的时间，music 为播放音乐的路径

```
def mainloop(ahead, music):
    class_alarms = get_class_alarm(ahead)
    to_do_info = get_to_do_alarm()
    to_do_alarms = to_do_info[0]
    to_do_things = to_do_info[1]
    print(class_alarms)
    print(to_do_alarms)
    while 1:
        time.sleep(1)
        for each in class_alarms:
            if each == int(time.time()):
                showMessage("现在距离上课还有{}分钟，请注意！！！！！".format(ahead), music)
        for i in range(len(to_do_alarms)):
            if to_do_alarms[i] == int(time.time()):
                showMessage("快到\n{}\n的时间了，请注意！！！！！".format(to_do_things[i]), music)
```

3）def get_class_alarm(ahead=0):

获得当天的课程情况，传入 ahead 为提前提醒的时间，输出 alarm_times 为绝对的时间戳，用于提醒确认时使用

```
def get_class_alarm(ahead=0):
    class_info = read_class_info()
    have_class = class_info
    for line in range(14):
        for row in range(len(class_info[0])):
            if class_info[line][row] == ' ':
                have_class[line][row] = 0
            else:
                have_class[line][row] = 1

    class_times = ["08:00", "08:50", "09:50", "10:40", "11:30", "13:00", "13:50", "14:45", "15:40", "16:35", "17:25",
                   "18:30", "19:20", "20:10"]
    alarm_times = []
    today = get_today()
    for i in range(14):
        if have_class[i][today]:
            alarm_times.append(class_times[i])

    ahead = 60 * ahead
    for i in range(len(alarm_times)):
        date = time.strftime("%Y-%m-%d", time.localtime())
        alarm_times[i] = time.mktime(time.strptime(date + ' ' + alarm_times[i], "%Y-%m-%d %H:%M")) - ahead

    return alarm_times
```

4）def get_today():

使用 datetime 库进行计算，返回当前是该学期的第几天。

```
def get_today():
    date0 = '2022-02-28'
    d0 = datetime.datetime.strptime(date0, '%Y-%m-%d')
    date1 = time.strftime("%Y-%m-%d", time.localtime())
    d1 = datetime.datetime.strptime(date1, '%Y-%m-%d')
    today = int((d1 - d0).days)
    return today
```

5）def read_class_info():

读取课程信息，对课程信息做一定的处理，以特定格式储存在多维列表中，便于之后使用。

```
def read_class_info():
    workbook = openpyxl.load_workbook('课程表.xlsx')
    sheet = workbook['课程表']
    class_info_ = [[]]
    for r in range(14):
        for c in range(112):
            temp = sheet.cell(row=r + 1, column=c + 1).value
            one = temp.find('\n')
            two = temp.find('\n', one + 1)
            if not one == -1:
                temp = temp[:two]
            class_info_[r].append(temp)
        class_info_.append([])
    return class_info_
```

6）def get_to_do_alarm(ahead=0):

读取当天待办事项信息，对信息做一定的处理，转换为时间戳和文字信息，以特定格式储存在多维列表中并返回，便于之后使用。返回格式[alarm_times, alarm_things]，前者为提醒时间，后者为提醒事件。

```
def get_to_do_alarm(ahead=0):
    workbook = openpyxl.load_workbook('待办事项.xlsx')
    sheet = workbook['待办事项']
    to_do_info_ = [[]]
    for r in range(20):
        for c in range(112):
            temp = sheet.cell(row=r + 1, column=c + 1).value
            to_do_info_[r].append(temp)
        to_do_info_.append([])

    alarm_times = []
    alarm_things = []
    today = get_today()
    for i in range(20):
        if not to_do_info_[i][today] == ' ':
            to_do_info_[i][today]=str(to_do_info_[i][today]).lstrip()
            print(to_do_info_[i][today])
            p = re.compile(r'\d\d:\d\d')
            temp = p.findall(to_do_info_[i][today])[0]
            date = time.strftime("%Y-%m-%d", time.localtime())
            alarm_times.append(time.mktime(time.strptime(date + ' ' + temp, "%Y-%m-%d %H:%M")))
            alarm_things.append(to_do_info_[i][today][6:])

    return [alarm_times, alarm_things]
```

7）def play_music(music):和 def shut_down():

开始播放和停止播放音乐，传入 music 为音乐文件的路径。

```
def play_music(music):
    pygame.mixer.init()
    pygame.mixer.music.load(music)
    pygame.mixer.music.play()


def shut_down():
    pygame.mixer.music.stop()
```

8）def showMessage(tex, music):

提醒事件触发时调用的函数，传入 tex 为提醒内容，music 为提醒音乐的路径。可以弹出提醒窗口并进行播放音乐。提供一个“我知道了”按钮，可以停止播放音乐。

该函数主要使用 tkinter 库进行操作

```python
def showMessage(tex, music):
    # show reminder message window
    root = Tk()  # 建立根窗口
    play_music(music)
    root.withdraw()  # hide window
    # 获取屏幕的宽度和高度，并且在高度上考虑到底部的任务栏，为了是弹出的窗口在屏幕中间
    screenwidth = root.winfo_screenwidth()
    screenheight = root.winfo_screenheight() - 100
    root.resizable(False, False)
    # 添加组件
    root.title("温馨提示")
    frame = Frame(root, relief=RIDGE, borderwidth=3)
    frame.pack(fill=BOTH, expand=1)  # pack() 放置组件若没有则组件不会显示
    # 窗口显示的文字、并设置字体、字号
    label = Label(frame, text=tex, font=('微软雅黑', 20, 'bold'))
    label.pack(fill=BOTH, expand=1)
    # 按钮的设置
    button = Button(frame, text="我知道了", font="Cooper -25 bold", fg="purple", command=shut_down)
    button.pack(side=BOTTOM)

    root.update_idletasks()
    root.deiconify()  # now the window size was calculated
    root.withdraw()  # hide the window again 防止窗口出现被拖动的感觉 具体原理未知？
    root.geometry('%sx%s+%s+%s' % (root.winfo_width() + 10, root.winfo_height() + 10,
                                   int((screenwidth - root.winfo_width()) / 2),
                                   int((screenheight - root.winfo_height()) / 2)))
    root.deiconify()
    root.mainloop()
```

3.7.8 全局数据结构与该模块的关系

访问了全局数据结构课程信息表格。访问了主程序根窗口和按钮。

3.8、待办事项管理模块（Todolist.py）

3.8.1、设计图

3.8.2、功能描述

待办事项管理模块的功能是给用户提供便捷的代办事项查看和编辑功能。其中展示功能以周为单位，展示一周内七天的简要待办事项（每天只展示前六条），完整的待办事项信息查看需要“读取”模块功能查看；编辑功能包括待办事项的写入和删除，写入功能包括录入用户待办事项的内容和时间，而删除功能可以对用户想要删除的待办事项进行多项选择。

3.8.3、输入数据

用户通过键盘和鼠标输入：待办事项内容信息和截止日期信息；需要删除的待办事项选择。

3.8.4、输出数据

无显性输出数据，待办事项的结果显示在展示模块。

3.8.5、数据设计

本模块外部唯一的数据是待办事项信息。虽然待办事项信息是稀疏的，但是，我们仍采用二维稀疏矩阵（xslx 文件）对其进行存储：一方面是因为待办事项的时间信息天然的包含在稀疏矩阵的位置信息里，大幅度减少了代码编写和数据结构设计的难度；另一方面，即便是稀疏矩阵存储信息，对空间的要求依然很小，在可执行范围内。

3.8.6、算法和流程

用户通过主界面按钮进入本模块后，本模块会读取待办事项信息，并通过 show_calendar 函数将待办事项信息表展示出来，用户可以选择点击每天的“查看详细”按钮进行当天的待办事项操作，如果选择了“写入”功能，本模块会获取用户的键盘输入信息，并更新待办事项信息存储表，“清除”功能同理。

3.8.7、函数说明

pop_up_box(w_num, d_num)函数:

功能：提供给用户输入待办事项信息的窗口

算法说明：通过调用 tkinter 包的 tkinter.StringVar 方法在子窗口上画出功能输入框，并加上标题，从而可以明确地获取用户的输入信息。

参数：当前周数 w_num　当前天数 d_num

使用限制：仅在 read_or_write 函数的写模块嵌套使用

read_or_write(w_num, d_num)函数:

功能：设置待办事项模块“读取”等功能，并集成于一个功能选择窗口，一共使用三个嵌套函数分别实现“读取”、“写入”、“删除”功能

算法说明：

cancel()函数：实现“删除”功能，通过 tkinter 包的 Checkbutton 方法获取用户选择的要删除的待办事项在“列表”中的位置信息，并更新列表达到删除的目的

read()函数：实现“读取”功能，通过输入的周数和天数信息，定位待办事项的列表在二维总表的位置，获取并打印至子窗口中。

write()函数：实现“写入功能”，一方面调用 pop_up_box 函数，

实现用户信息的获取，另一方面调用 tkinter.messagebox 包的 msgbox.showinfo 方法进行空输入的差错提醒。

参数：当前周数 w_num　当前天数 d_num

使用限制：仅在 draw_buttons_2 构造按键后，用户点击按键后生效。

read_info()函数:

功能：从“待办事项.xlsx”文件中读取待办事项信息

算法说明：通过调用 openpyxl 包的 openpyxl.load_workbook 方法遍历获取待办事项信息，并存入二维数组中。

参数：无

返回值：所有含有待办事项信息的二维数组

draw_window(pic_way)函数:

功能：构建单个本模块主题展示窗口及内容

算法说明：通过 Canvasr 方法经过 for 循环在窗口上分别打印出星期数和时间数（6 个空位）；在通过读取的待办事项信息，将信息分别填入每个展示窗口界面。

参数：背景图片链接 pic_way

使用限制：仅在 show_calendar 函数中调用

print_class(week)函数:

功能：为每个本模块主题展示窗口构建周数选择功能按钮

算法说明：通过调用 draw_buttons_2 和 draw_buttons 完成展示界面的所有功能性窗口的绘制，通过调用 get_current_week 函数来获取当前的周数。

参数：背景图片链接 pic_way

使用限制：仅在 show_calendar 函数中调用

get_current_week()函数:

功能：自动计算当前日期所在的周数

算法说明：通过纯数学的方法，将 time 包的 time.strftime 方法获取的日期进行简单的数学上的计算，从而获得当前的周数

参数：当前周数 week

使用限制：无

draw_buttons_2(week)函数:

功能：在展示窗口的星期数下建立“查看明细”按钮，按钮内封

装了“读取”等功能

算法说明：通过循环调用 tkinter 包的 Button 方法设置一系列建立按钮的函数，并将所有函数存储为字符串并列放入列表中。

参数：当前周数 week

使用限制：仅在 print_class 函数中使用 exac 方法调用

draw_buttons()函数:

功能：在展示窗口的星期数下建立查看周数的按钮，按钮内封装了切换展示界面的功能

算法说明：通过循环调用 tkinter 包的 Button 方法设置一系列建立按钮的函数，并将所有函数存储为字符串并列放入列表中。

参数：无

使用限制：仅在 print_class 函数中使用 exac 方法调用

show_calendar(pic_way)函数:

功能：综合上述几种函数，将本模块的主要展示届面完全绘制成功并展示

参数：背景图片链接 pic_way

孙泽割20202121180张扬20202121185王梓20202121212

3.9、桌面小组件模块（desktop_app.py）

3.4.1、设计图

3.4.2、功能描述

创建桌面小挂件，在方便移动和退出的同时，满足主界面的隐藏和再打开功能，并且不影响提醒模块的正常运行。

3.4.3、函数说明

所有的函数都封装到了一个两 class 类中——DesktopAPP(QWidget)和 bupt_study()，下面将介绍两个类中的主要功能函数。

LoadImages(self, name)函数:

功能：从本地导入一张或多张图片作为桌面小组件的图标和任务栏图标

参数：图片所在文件夹名称 name

mousePressEvent(self, event)函数:

功能：设置鼠标点击事件后的功能：单击右键打开主界面，单击左键开启图标跟随

参数：鼠标点击事件 event

使用限制：仅在检测到鼠标被点击时调用

mouseMoveEvent(self, event)函数:

功能：当图标跟随变量 self.is_follow_mouse 为 True 时，桌面组件跟随鼠标的拖动进行移动

参数：鼠标拖动事件 event

使用限制：仅在检测到鼠标左键被点击不释放后拖动时调用

mouseReleaseEvent(self, event)函数:

功能：当鼠标被释放将图标跟随变量 self.is_follow_mouse 设置为 False

参数：鼠标左键释放事件 event

使用限制：仅在检测到鼠标左键被释放时调用

execute(self, config={})函数:

功能：执行对应的小程序

算法说明：通过调用 PyQt5.QtWidgets 包的 QApplication(sys.argv)方法运行小程序

参数：参数 config（实际上为空）

孙泽凯202021180张扬202021185王梓安202021212

4、 接口设计

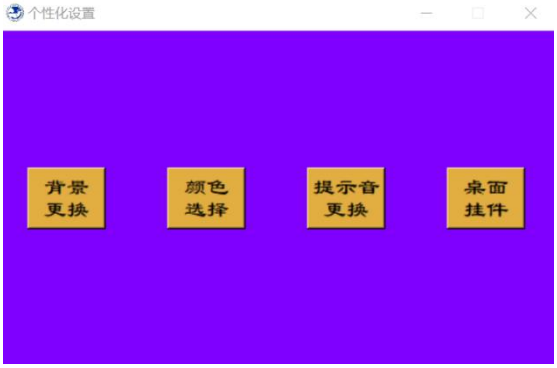
4.1、 用户接口

Main.py:

该部分为主菜单界面，包括了大部分与用户直接交互的按钮系统，用户通过点击窗口上的按钮作为用户接口来调用相应功能函数。

主要可用作交互的按钮接口有六个（邮学小帮手标题也是一个大的按钮）如下图所示：

当点击个性化设置按钮后会弹出上层窗口进行个性化功能选择，此处的新窗口也有四个按钮用作用户交互接口，如下图所示：

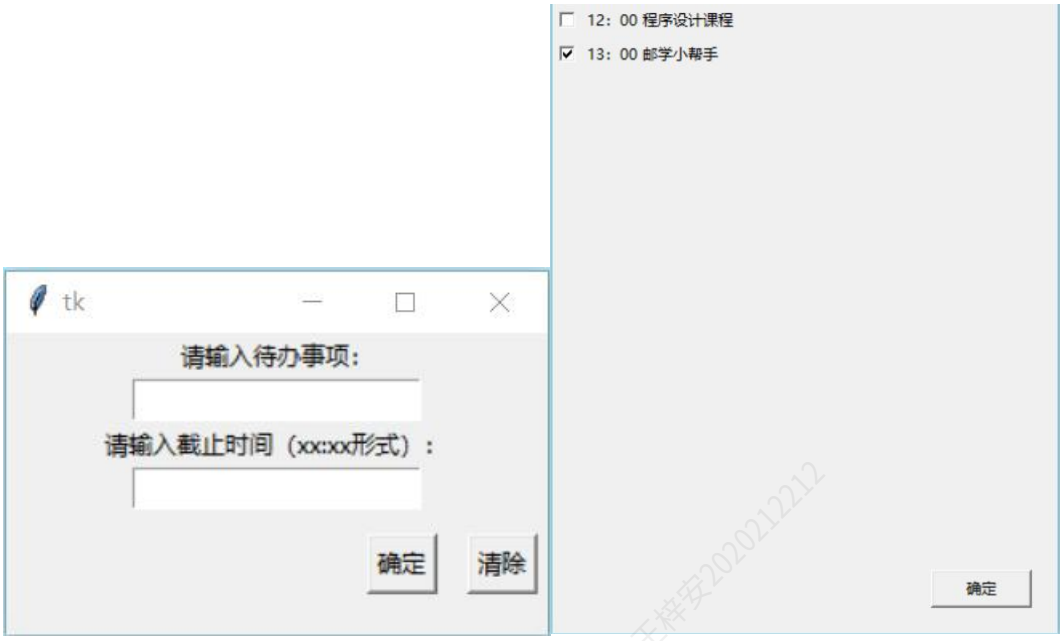


Creep.py:

该部分无向用户提供的接口，但是调用的浏览器中存在用户接口，主要表现为需要用户手动输入教务系统用户名密码等。

Todolist.py:

设计了与用户交互的按钮系统，包括：“写入”模块获取用户输入的待办事项信息；“清除”模块获取用户对待办事项的删除信息。



desktopapp.py:

mousePressEvent(self, event)方法获取用户鼠标的点击信息：左键长按移动插件、右键单击打开程序主界面，右键单击任务栏小图标可以选择退出程序。

4.2、 外部接口

Creep.py:

该模块需要使用到 python 的爬虫进行信息的爬取，当我们使用 selenium 来操纵浏览器时就需要用到与 edge 浏览器之间的外部接口。具体控制方式为调用 edgedriver.exe 来打开浏览器，后通过对 edgedriver 的操作来操纵浏览器进行信息的爬取。

```
byr_page = webdriver.Edge()
byr_page.get(url)
```

4.3、 内部接口

4.3.1、 接口说明

main 主模块通过调用子模块中的函数从 choose_back 模块取得

背景图片文件的绝对路径。

```
bg_path.set(choose_back.bg_change())
class_color.set(choose_color.color_return())
```

main 主模块通过调用子模块中的函数从 choose_music 模块取得音频文件的绝对路径，接口调用实例与 1 中的实例一致。

main 主模块通过调用子模块中的函数从 choose_color 模块取得用户所选择的颜色对应的颜色代码。接口调用实例与 1 中的实例一致。

4.3.2、 调用方式

接口 123 的调用方式思路一致，都是通过调用子模块中的函数进行调用。由于子模块中存在着函数可以直接返回所需要的字符串变量值，所以在主函数中调用接口时就只需要调用该子模块函数，并将返回值赋值给一个临时变量，再用该临时变量做下一步的处理就可以了。

5、数据库设计

系统不以数据库方式存储数据，故省略。

6、 系统出错处理

6.1、 出错信息

Todolist.py:

“写入”模块可能存在未能获取用户输入信息和用户未能正确输入信息的问题，从而导致错过待办事项。

6.2、 补救措施

Todolist.py:

“写入”模块设置了“提醒模块”：当用户点击率“确认写入”按钮后，如果软件无法检测到输入框写入有效信息，将弹出提醒窗口，由用户检查是否是自身输入产生问题。

7、 其他设计

略。

孙泽凯202021218O张扬202021185王梓安202021

全文共14257字