目录

[一、 安装mysql和python 2](#_Toc26048233)

[二、 基础配置 2](#_Toc26048234)

[1、python模块安装 2](#_Toc26048235)

[2、mysql数据库设置 2](#_Toc26048236)

[三、 设计结构 3](#_Toc26048237)

[1、 项目目录结构 3](#_Toc26048238)

[\_pycache\_层 4](#_Toc26048239)

[1、介绍说明 4](#_Toc26048240)

[2、 python文件说明 4](#_Toc26048241)

[Dao层 5](#_Toc26048242)

[1、 介绍说明 5](#_Toc26048243)

[2、 python文件说明 6](#_Toc26048244)

[四、 附录 8](#_Toc26048245)

[Main.py 8](#_Toc26048246)

[\_pycache\_层 8](#_Toc26048247)

[config.py 8](#_Toc26048248)

[model.py 9](#_Toc26048249)

[dao层 11](#_Toc26048250)

[enter.py 11](#_Toc26048251)

[OperationWind.py 14](#_Toc26048252)

[Add.py 17](#_Toc26048253)

[Delete.py 22](#_Toc26048254)

[Select.py 24](#_Toc26048255)

# 安装mysql和python

Mysql安装教程：<https://blog.csdn.net/bobo553443/article/details/81383194>

Sqlyog安装教程：<https://blog.csdn.net/qq_40938267/article/details/83894206>

Sqlyog是mysql的图形化管理软件

Python安装教程：https://blog.csdn.net/qq\_37702890/article/details/80508526

Pycharm安装教程: https://blog.csdn.net/qq\_15698613/article/details/86502371

Pycharm是python的编辑器

# 基础配置

## 1、python模块安装

键盘windows+R，输入cmd回车，依次输入

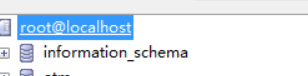
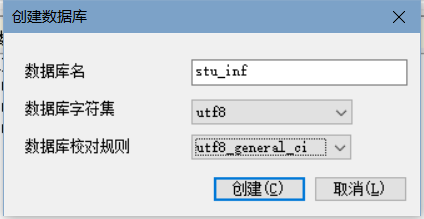
pip install wxPython

pip install pymysql

## 2、mysql数据库设置

建立一个stu\_inf的数据库(数据集为utf-8)

create database stu\_inf

右键选择创建数据库，

USE stu\_inf 进入数据库

建立管理员用户和学生信息的数据表

CREATE TABLE `users` (

`id` int(10) unsigned NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`user` varchar(8) UNIQUE NOT NULL,

`password` varchar(16) NOT NULL

);

CREATE TABLE `stu` (

`id` int unsigned NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`name` varchar(10) NOT NULL,

`age` tinyint(3) unsigned NOT NULL,

`sex` enum('男','女') NOT NULL,

`Sdept` varchar(5) NOT NULL,

`profession` varchar(3) NOT NULL,

`classid` tinyint(3) unsigned NOT NULL,

`grade` tinyint(3) unsigned NOT NULL

);

向管理员表插入一条管理员数据

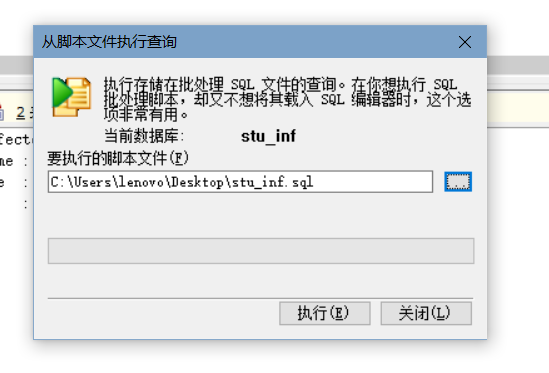
insert into `users`(`id`,`user`,`password`) values ( '814054','张三','123456');

学生数据库插入数据

insert into ` stu` values ( ' 201707010119','李金哲',20,’男’,'理学院',’信计’,1,1);

**或者用stu\_inf.sql进行数据恢复**

右键stu\_inf数据库，选择以SQL存储文件恢复数据库，选择stu\_inf.sql如图，然后执行



# 设计结构

## 项目目录结构

/student/

├── Main.py 程序入口

├── \_pycache\_ mysql连接和调用层

│ ├── \_\_init\_\_.py

│ ├── config.py mysql数据库配置

│ └── model.py 封装python中mysql的增删改查的类

├── dao 用户界面显示层

│ ├── \_\_init\_\_.py

│ ├── Add.py 添加学生信息的界面和insert的调用

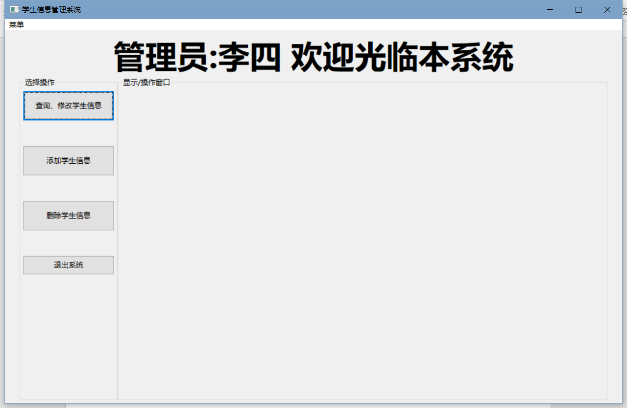
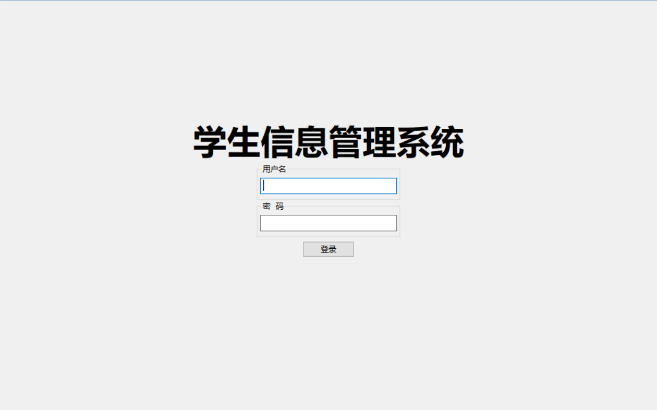
│ ├── Delete.py 删除学生信息的界面和delete的调用

│ ├── enter.py 学生管理系统的登录入口

│ ├── OperationWind.py 学生管理系统操作界面(调用界面父类

│ └── Select.py 查询全部学生信息的界面和select的调用

界面展示







## \_pycache\_层

### 1、介绍说明

本层是用于存放mysql的连接配置和实现其方法调用的python程序

### python文件说明

**说明：当前只展示重要代码，详细代码见附录，如果代码中含有部分的解析注释**

#### \_\_init\_\_.py

该文件是python调用此目录下其他python文件的访问路口，必须存在，内容为空即可。

#### config.py

存放了数据库链接的链接信息，将数据库的链接信息通过赋值，定义给变量。

#### model.py

存放了数据库的调用model方法。

**class** Model 是声明类

**def** \_\_init\_\_(self, table, config=config):

通过解析函数对类进行初始化，将连接数据库的内容放入其中，table是表需要连接的表名

self.link = pymysql.connect(host=config.HOST, user=config.USER, password=config.PASSSWD, db=config.DBNAME,port=config.PORT, charset=**'utf8'**)

调用pymysql里的connect方法连接mysql，配置文件为config.py里的变量。使用link对象，作为数据库连接对象的接受值。

self.cursor = self.link.cursor(pymysql.cursors.DictCursor)

将调用pymysql后获取的值转化为字典对象，赋值给cursor

**def** \_\_del\_\_(self):

析构函数，应用于函数结束的时候，将self.cursor和self.link的对象关闭并释放内存空间

**def** \_\_loadFields(self):

变量名字前+\_\_是定义当前变量或方法为私有。该方法是获取表内的字段信息，获得表的key字段信息

**def** findAll(self):

查询该表里的所有信息

sql = **'select** *\** **from %s'** % self.tab\_name

是封装sql语句，将需要查询的表名添加到sql字符串中

self.cursor.execute(sql)

是执行sql语句的方法

self.cursor.fetchall()

获取执行后的返回信息

其余函数原理同上

## Dao层

### 介绍说明

本层用于存放学生管理系统的界面构造的python文件

### python文件说明

#### enter.py

学生登录界面

**from** \_pycache\_.model **import** Model

从\_pycache\_文件夹中导入model类

**def** Button(self, event)

定义按钮的点击事件

mod = Model(**'users'**)

调用Mole类的构造函数，赋值给mod对象

userlist = mod.findAll()

调用model里的findAll函数

**from** .OperationWind **import** OperationWind

从当前目录下导入OperationWind.py文件中的OperationWind类

operation = OperationWind(**None**, title=**"学生信息管理系统"**, name=user, size=(1024, 668))

调用OperationWind函数的析构函数

operation.menu()  
operation.Show()

显示产生operation窗口

self.Close()

关闭当前enter窗口

#### OperationWind.py

**class** OperationWind(wx.Frame)

括号里的是继承的类

**def** \_\_init\_\_(self, \*args, \*\*kw):

super(OperationWind, self).\_\_init\_\_(\*args, \*\*kw)

调用父类wxpython的析构函数

self.Center()

设置窗口显示的位置为屏幕中央

self.pnl = wx.Panel(self)

创建窗口对象

**def** OperationInterface(self):

设定窗口组件的布局

self.vbox = wx.BoxSizer(wx.VERTICAL)

以箱式布局设置窗口

**由于添加组件的代码多有重复，在此仅举一例，以供参考**

title = wx.StaticText(self.pnl, label=**'管理员:'** + self.GetName() + **' 欢迎光临本系统'**)

设置文本，类型为wx里的StaticText，在pnl窗口里，label是显示的文本

font = title.GetFont()

设置显示的文字要求

font.PointSize += 30

设置显示的文字大小

font = font.Bold()

设置显示文字加粗

title.SetFont(font)

将文字要求放入title中

self.vbox.Add(title, proportion=0, flag=wx.FIXED\_MINSIZE | wx.TOP | wx.CENTER, border=5)

添加静态文本到箱式布局中，内容，相对坐标，位置，加粗

其余设置思路类似

self.Bind(wx.EVT\_BUTTON, self.ClickButton, id=10, id2=13)

设置绑定事件id是指定的事件源，id2是终止事件，wx.EVT\_BUTTON是事件类型

**def** menu(self):

设置菜单栏

思路同添加窗口组件

**def** ClickButton(self, event):

构造点击事件，思路同构建OperationWind的操作

**其余函数思路如同上两个，故不作过多说明**

# 附录

## Main.py

**from** dao.enter **import** User  
**import** wx  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 app = wx.App()  
 enter = User(**None**, title=**"学生信息管理系统"**, size=(1024, 668))  
 enter.Show()  
 app.MainLoop()

## \_pycache\_层

### config.py

# 数据库的链接信息  
# 主机名  
HOST = 'localhost'  
# 账号  
USER = 'root'  
# 密码  
PASSSWD = ''  
# 数据库名  
DBNAME = 'stu\_inf'  
# 端口  
PORT = 3306

### model.py

**import** pymysql  
**from** . **import** config  
  
  
**class** Model:  
 *"""单表信息操作类"""  
 # 表名* tab\_name = **None** *# 数据库连接对象* link = **None** *# 游标对象* cursor = **None** *# 表的主键名* pk = **None** *# 当前表的字段名列表* fields = []  
  
 **def** \_\_init\_\_(self, table, config=config):  
 *"""构造方法：初始化表名，连接数据库"""* **try**:  
 self.tab\_name = table  
 self.link = pymysql.connect(host=config.HOST, user=config.USER, password=config.PASSSWD, db=config.DBNAME,  
 port=config.PORT, charset=**'utf8'**)  
 *# 让返回结果为字典模式* self.cursor = self.link.cursor(pymysql.cursors.DictCursor)  
 *# 调用内部方法，加载当前表的字段信息* self.\_\_loadFields()  
 **except** Exception **as** err:  
 print(**"数据操作对象初始化失败，原因：%s"** % err)  
  
 **def** \_\_del\_\_(self):  
 *"""关闭对象和连接对象"""* **if** self.cursor **is not None**:  
 self.cursor.close()  
 **if** self.link **is not None**:  
 self.link.close()  
  
 **def** \_\_loadFields(self):  
 *"""加载当前的表的字段信息，内部私有方法"""* sql = **"show columns from %s"** % self.tab\_name  
 self.cursor.execute(sql)  
 dlist = self.cursor.fetchall()  
 *# 遍历所有字段信息* **for** v **in** dlist:  
 *# 收集每个字段名称* self.fields.append(v[**'Field'**])  
 *# 判断并收集表的主键名称* **if** v[**'Key'**] == **'PRI'**:  
 self.pk = v[**'Field'**]  
  
 **def** findAll(self):  
 *"""获取当前表的所有信息，没有信息返回[]"""* **try**:  
 sql = **'select** *\** **from %s'** % self.tab\_name  
 self.cursor.execute(sql)  
 **return** self.cursor.fetchall()  
 **except** Exception **as** err:  
 print(**"SQL查询错误,原因: %s"** % err)  
  
 **def** save(self, data={}):  
 *"""信息添加方法"""  
 # SQL语句：insert into 表名（字段列表） values（值列表）* **try**:  
 *# 组装SQL语句* keys = []  
 values = []  
 **for** k, v **in** data.items():  
 *# 获取有效字段* **if** k **in** self.fields:  
 keys.append(k)  
 values.append(v)  
 sql = **"insert into %s(%s) values(%s)"** % (self.tab\_name, **','**.join(keys), **','**.join([**'%s'**] \* len(values)))  
 self.cursor.execute(sql, tuple(values))  
 *# 提交事务* self.link.commit()  
 *# 返回行的自增id号* **return** self.cursor.rowcount  
 **except** Exception **as** err:  
 print(**"SQL执行错误,原因: %s"** % err)  
 **return** 0  
  
 **def** update(self, data={}):  
 *"""信息修改方法"""  
 # SQL语句：insert into 表名（字段列表） values（值列表）* **try**:  
 *# 组装SQL语句* values = []  
 **for** k, v **in** data.items():  
 *# 获取有效字段* **if** (k **in** self.fields) **and** (k != self.pk):  
 *# 判断其不是主键* values.append(**"%s='%s'"** % (k, v))  
 sql = **"update %s set %s where %s = '%s'"** % (self.tab\_name, **','**.join(values), self.pk, data.get(self.pk))  
 self.cursor.execute(sql)  
 *# 提交事务* self.link.commit()  
 *# 返回影响行数* **return** self.cursor.rowcount  
 **except** Exception **as** err:  
 print(**"SQL执行错误,原因: %s"** % err)  
 **return** 0  
  
 **def** delete(self, id=0):  
 *"""信息删除方法"""* **try**:  
 sql = **"delete from %s where %s = %s"** % (self.tab\_name, self.pk, id)  
 self.cursor.execute(sql)  
 self.link.commit()  
 **return** self.cursor.rowcount  
 **except** Exception **as** err:  
 print(**"SQL删除错误: %s"** % err)  
 **return** 0

## dao层

### enter.py

**import** wx  
**import** wx.grid  
**from** \_pycache\_.model **import** Model  
  
  
*# 创建学生信息管理系统登录界面类***class** User(wx.Frame):  
 *"""  
 登录界面  
 """  
  
 # 初始化登录界面* **def** \_\_init\_\_(self, \*args, \*\*kw):  
 super(User, self).\_\_init\_\_(\*args, \*\*kw)  
 *# 设置窗口屏幕居中* self.Center()  
 *# 创建窗口* self.pnl = wx.Panel(self)  
 *# 调用登录界面函数* self.Enter()  
  
 **def** Enter(self):  
 *# 创建垂直方向box布局管理器* vbox = wx.BoxSizer(wx.VERTICAL)  
 *# 创建静态文本，设置字体属性* title = wx.StaticText(self.pnl, label=**"学生信息管理系统"**)  
 font = title.GetFont()  
 font.PointSize += 30  
 font = font.Bold()  
 title.SetFont(font)  
  
 *# 添加静态文本到vbox布局管理器* vbox.Add(title, proportion=0, flag=wx.FIXED\_MINSIZE | wx.TOP | wx.CENTER, border=180)  
 *# 创建静态框* sb\_username = wx.StaticBox(self.pnl, label=**"用户名"**)  
 sb\_password = wx.StaticBox(self.pnl, label=**"密 码"**)  
 *# 创建水平方向box布局管理器* hsbox\_username = wx.StaticBoxSizer(sb\_username, wx.HORIZONTAL)  
 hsbox\_password = wx.StaticBoxSizer(sb\_password, wx.HORIZONTAL)  
 *# 创建用户名、密码输入框* self.id = wx.TextCtrl(self.pnl, size=(210, 25))  
 self.password = wx.TextCtrl(self.pnl, size=(210, 25), style=wx.TE\_PASSWORD)  
 *# 添加用户名和密码输入框到hsbox布局管理器* hsbox\_username.Add(self.id, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_password.Add(self.password, 0, wx.EXPAND | wx.BOTTOM, 5)  
 *# 将水平box添加到垂直box* vbox.Add(hsbox\_username, proportion=0, flag=wx.CENTER)  
 vbox.Add(hsbox\_password, proportion=0, flag=wx.CENTER)  
 *# 创建水平方向box布局管理器* hbox = wx.BoxSizer()  
 *# 创建登录按钮、绑定事件处理* login\_button = wx.Button(self.pnl, label=**"登录"**, size=(80, 25))  
 login\_button.Bind(wx.EVT\_BUTTON, self.Button)  
 *# 添加登录按钮到hbox布局管理器* hbox.Add(login\_button, 0, flag=wx.EXPAND | wx.TOP, border=5)  
 *# 将水平box添加到垂直box* vbox.Add(hbox, proportion=0, flag=wx.CENTER)  
  
 *# 设置面板的布局管理器vbox* self.pnl.SetSizer(vbox)  
  
 **def** Button(self, event):  
 *# 连接数据库* mod = Model(**'users'**)  
 *# 获取数据库中的全部管理用户* userlist = mod.findAll()  
 *# 定义标记对象* flag = **False** user = **None  
 for** data **in** userlist:  
 **if** str(data[**'id'**]) == self.id.GetValue() **and** data[**'password'**] == self.password.GetValue():  
 user = data[**'user'**]  
 flag = **True  
 break  
 if** flag **is False**:  
 print(**"用户名或密码错误！"**)  
  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'用户名或密码错误！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show()  
 app.MainLoop()  
  
 **else**:  
 print(**"登录成功！"**)  
 **from** .OperationWind **import** OperationWind  
 operation = OperationWind(**None**, title=**"学生信息管理系统"**, name=user, size=(1024, 668))  
 operation.menu()  
 operation.Show()  
 self.Close()

### OperationWind.py

*#!/usr/bin/env python  
# -\*- coding: utf-8 -\*-  
# @Time : 2019/11/21 22:31  
# @Author : 李金哲  
# @Site :  
# @File : OperationWind.py  
# @Software: PyCharm***import** wx  
**import** wx.grid  
  
  
**class** OperationWind(wx.Frame):  
 *"""  
 操作界面  
 """* **def** \_\_init\_\_(self, \*args, \*\*kw):  
 *# ensure the parent's \_\_init\_\_ is called* super(OperationWind, self).\_\_init\_\_(\*args, \*\*kw)  
  
 *# 设置窗口屏幕居中* self.Center()  
 *# 创建窗口* self.pnl = wx.Panel(self)  
 *# 调用操作界面函数* self.OperationInterface()  
  
 **def** OperationInterface(self):  
 *# 创建窗口* self.vbox = wx.BoxSizer(wx.VERTICAL)  
  
 *# 设置字体属性* title = wx.StaticText(self.pnl, label=**'管理员:'** + self.GetName() + **' 欢迎光临本系统'**)  
 font = title.GetFont()  
 font.PointSize += 30  
 font = font.Bold()  
 title.SetFont(font)  
 *# 添加静态文本到vbox布局管理器* self.vbox.Add(title, proportion=0, flag=wx.FIXED\_MINSIZE | wx.TOP | wx.CENTER, border=5)  
  
 *# 创建静态框* sb\_button = wx.StaticBox(self.pnl, label=**"选择操作"**)  
 *# 创建垂直方向box布局管理器* vsbox\_button = wx.StaticBoxSizer(sb\_button, wx.VERTICAL)  
 *# 创建操作按钮、绑定事件处理* check\_button = wx.Button(self.pnl, id=10, label=**"查询、修改学生信息"**, size=(150, 50))  
 add\_button = wx.Button(self.pnl, id=11, label=**"添加学生信息"**, size=(150, 50))  
 delete\_button = wx.Button(self.pnl, id=12, label=**"删除学生信息"**, size=(150, 50))  
 quit\_button = wx.Button(self.pnl, id=13, label=**"退出系统"**, size=(150, 50))  
 self.Bind(wx.EVT\_BUTTON, self.ClickButton, id=10, id2=13)  
  
 *# 添加操作按钮到vsbox布局管理器* vsbox\_button.Add(check\_button, 0, wx.EXPAND | wx.BOTTOM, 40)  
 vsbox\_button.Add(add\_button, 0, wx.EXPAND | wx.BOTTOM, 40)  
 vsbox\_button.Add(delete\_button, 0, wx.EXPAND | wx.BOTTOM, 40)  
 vsbox\_button.Add(quit\_button, 0, wx.EXPAND | wx.BOTTOM, 200)  
 *# 创建静态框* sb\_show\_operation = wx.StaticBox(self.pnl, label=**"显示/操作窗口"**, size=(800, 500))  
  
 *# 创建垂直方向box布局管理器* self.vsbox\_show\_operation = wx.StaticBoxSizer(sb\_show\_operation, wx.VERTICAL)  
  
 *# 创建水平方向box布局管理器* hbox = wx.BoxSizer()  
 hbox.Add(vsbox\_button, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hbox.Add(self.vsbox\_show\_operation, 0, wx.EXPAND | wx.BOTTOM, 5)  
 *# 将hbox添加到垂直box* self.vbox.Add(hbox, proportion=0, flag=wx.CENTER)  
  
 self.pnl.SetSizer(self.vbox)  
  
 **def** menu(self):  
 *"""设置菜单栏"""  
 # 创建一个菜单栏* menuBar = wx.MenuBar()  
 *# 创建一个菜单* fileMenu = wx.Menu()  
 *# 创建一个子菜单* editMenu = wx.Menu()  
 *# 创建三个子菜单的菜单项目* selectItem = wx.MenuItem(editMenu, id=10, text=**"查询"**, kind=wx.ITEM\_NORMAL)  
 insertItem = wx.MenuItem(editMenu, id=11, text=**"插入"**, kind=wx.ITEM\_NORMAL)  
 updateItem = wx.MenuItem(editMenu, id=11, text=**"修改"**, kind=wx.ITEM\_NORMAL)  
 delItem = wx.MenuItem(editMenu, id=12, text=**"删除"**, kind=wx.ITEM\_NORMAL)  
 editMenu.Append(selectItem)  
 editMenu.Append(insertItem)  
 editMenu.Append(updateItem)  
 editMenu.Append(delItem)  
  
 *# 把子菜单添加到菜单中* fileMenu.Append(wx.ID\_ANY, **"基础操作"**, editMenu)  
 *# 添加一行线* fileMenu.AppendSeparator()  
 *# 添加退出操作* quit = wx.MenuItem(fileMenu, id=13, text=**"Quit\tCtrl+Q"**, kind=wx.ITEM\_NORMAL)  
 fileMenu.Append(quit)  
  
 *# 将 fileMenu 菜单添加到菜单栏中* menuBar.Append(fileMenu, title=**'菜单'**)  
 *# 创建操作按钮、绑定事件处理* self.Bind(wx.EVT\_MENU, self.ClickButton)  
 self.SetMenuBar(menuBar)  
  
 **def** ClickButton(self, event):  
 num = event.GetId()  
 **if** num == 10:  
 print(**"查询操作！"**)  
  
 **from** .Select **import** Select  
 select\_wind = Select(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 select\_wind.menu()  
 select\_wind.Show()  
 self.Close()  
  
 **elif** num == 11:  
 print(**"添加操作！"**)  
 **from** .Add **import** Add  
 add\_wind = Add(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
  
 **elif** num == 12:  
 print(**"删除操作！"**)  
 **from** .Delete **import** Delete  
 add\_wind = Delete(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
  
 **elif** num == 13:  
 self.Close()

### Add.py

*#!/usr/bin/env python  
# -\*- coding: utf-8 -\*-  
# @Time : 2019/11/23 17:38  
# @Author : 李金哲  
# @Site :   
# @File : Add.py  
# @Software: PyCharm***from** .OperationWind **import** OperationWind  
**from** \_pycache\_.model **import** Model  
**import** wx  
**import** wx.grid  
  
  
**class** Add(OperationWind):  
 **def** \_\_init\_\_(self, \*args, \*\*kw):  
 *# ensure the parent's \_\_init\_\_ is called* super(Add, self).\_\_init\_\_(\*args, \*\*kw)  
 *# 创建添加学生信息输入框、添加按钮* self.stu\_id = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_name = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_age = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_gender = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_sde = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_pro = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_class = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.stu\_grade = wx.TextCtrl(self.pnl, size=(210, 20))  
 self.add\_button = wx.Button(self.pnl, label=**"添加"**, size=(80, 20))  
 self.update\_button = wx.Button(self.pnl, label=**"修改"**, size=(80, 20))  
 *# 为添加按钮组件绑定事件处理* self.add\_button.Bind(wx.EVT\_BUTTON, self.AddButton)  
 *# 为修改按钮组件绑定事件处理* self.update\_button.Bind(wx.EVT\_BUTTON, self.UpdateButton)  
  
 *# 创建静态框* input\_id = wx.StaticBox(self.pnl, label=**"学 号"**)  
 input\_name = wx.StaticBox(self.pnl, label=**"姓 名"**)  
 input\_age = wx.StaticBox(self.pnl, label=**"年 龄"**)  
 input\_gender = wx.StaticBox(self.pnl, label=**"性 别"**)  
 input\_sde = wx.StaticBox(self.pnl, label=**"学 院"**)  
 input\_pro = wx.StaticBox(self.pnl, label=**"专 业"**)  
 input\_class = wx.StaticBox(self.pnl, label=**"班 级"**)  
 inputu\_grade = wx.StaticBox(self.pnl, label=**"年 级"**)  
  
 *# 创建水平方向box布局管理器* hsbox\_id = wx.StaticBoxSizer(input\_id, wx.HORIZONTAL)  
 hsbox\_name = wx.StaticBoxSizer(input\_name, wx.HORIZONTAL)  
 hsbox\_age = wx.StaticBoxSizer(input\_age, wx.HORIZONTAL)  
 hsbox\_gender = wx.StaticBoxSizer(input\_gender, wx.HORIZONTAL)  
 hsbox\_sde = wx.StaticBoxSizer(input\_sde, wx.HORIZONTAL)  
 hsbox\_pro = wx.StaticBoxSizer(input\_pro, wx.HORIZONTAL)  
 hsbox\_class = wx.StaticBoxSizer(input\_class, wx.HORIZONTAL)  
 hsbox\_grade = wx.StaticBoxSizer(inputu\_grade, wx.HORIZONTAL)  
  
 *# 添加到hsbox布局管理器* hsbox\_id.Add(self.stu\_id, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_name.Add(self.stu\_name, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_age.Add(self.stu\_age, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_gender.Add(self.stu\_gender, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_sde.Add(self.stu\_sde, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_pro.Add(self.stu\_pro, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_class.Add(self.stu\_class, 0, wx.EXPAND | wx.BOTTOM, 5)  
 hsbox\_grade.Add(self.stu\_grade, 0, wx.EXPAND | wx.BOTTOM, 5)  
  
 *# 添加到vsbox\_show\_operation布局管理器* self.vsbox\_show\_operation.Add(hsbox\_id, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_name, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_age, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_gender, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_sde, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_pro, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_class, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(hsbox\_grade, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(self.add\_button, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.vsbox\_show\_operation.Add(self.update\_button, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
  
 self.menu()  
  
 **def** AddButton(self, event):  
 key\_word = {**'id'**: self.stu\_id.GetValue(), **'name'**: self.stu\_name.GetValue(), **'age'**: self.stu\_age.GetValue(),  
 **'sex'**: self.stu\_gender.GetValue(), **'Sdept'**: self.stu\_sde.GetValue(), **'profession'**: self.stu\_pro.GetValue(),  
 **'classid'**: self.stu\_class.GetValue(), **'grade'**: self.stu\_grade.GetValue()}  
 *# 连接数据库* mod = Model(**'stu'**)  
 *# 保存学生信息,获取保存结果* **if** mod.save(key\_word) > 0:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息添加成功！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **else**:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息添加失败！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **def** UpdateButton(self, event):  
 key\_word = {**'id'**: self.stu\_id.GetValue(), **'name'**: self.stu\_name.GetValue(), **'age'**: self.stu\_age.GetValue(),  
 **'sex'**: self.stu\_gender.GetValue(), **'Sdept'**: self.stu\_sde.GetValue(),  
 **'profession'**: self.stu\_pro.GetValue(),  
 **'classid'**: self.stu\_class.GetValue(), **'grade'**: self.stu\_grade.GetValue()}  
 *# 连接数据库* mod = Model(**'stu'**)  
 *# 保存学生信息,获取保存结果* **if** mod.update(key\_word) > 0:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息修改成功！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **else**:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息修改失败！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **def** ClickButton(self, event):  
 num = event.GetId()  
 **if** num == 10:  
 print(**"查询操作！"**)  
 **from** .Select **import** Select  
 select\_wind = Select(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 select\_wind.menu()  
 select\_wind.Show()  
 self.Close()  
  
 **elif** num == 11:  
 **pass  
  
 elif** num == 12:  
 print(**"删除操作！"**)  
 **from** .Delete **import** Delete  
 add\_wind = Delete(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
  
 **elif** num == 13:  
 self.Close()

### Delete.py

*#!/usr/bin/env python  
# -\*- coding: utf-8 -\*-  
# @Time : 2019/11/23 21:09  
# @Author : 李金哲  
# @Site :   
# @File : Delete.py  
# @Software: PyCharm***from** .Select **import** Select  
**from** \_pycache\_.model **import** Model  
**import** wx  
**import** wx.grid  
  
  
**class** Delete(Select):  
 **def** \_\_init\_\_(self, \*args, \*\*kw):  
 *# ensure the parent's \_\_init\_\_ is called* super(Delete, self).\_\_init\_\_(\*args, \*\*kw)  
 *# 创建删除信息输入框、删除按钮* self.del\_id = wx.TextCtrl(self.pnl, pos=(407, 78), size=(210, 25))  
 self.del\_button = wx.Button(self.pnl, label=**"删除"**, pos=(625, 78), size=(80, 25))  
 *# 为删除按钮组件绑定事件处理* self.del\_button.Bind(wx.EVT\_BUTTON, self.DelButton)  
  
 *# 创建静态框* sb\_del = wx.StaticBox(self.pnl, label=**"请选择需要删除的学生学号"**)  
 *# 创建水平方向box布局管理器* hsbox\_del = wx.StaticBoxSizer(sb\_del, wx.HORIZONTAL)  
 *# 添加到hsbox\_name布局管理器* hsbox\_del.Add(self.del\_id, 0, wx.EXPAND | wx.BOTTOM, 5)  
 *# 添加到vsbox\_show\_operation布局管理器* self.vsbox\_show\_operation.Add(hsbox\_del, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 5)  
 self.menu()  
  
 **def** DelButton(self, event):  
 *# 连接数据库* mod = Model(**'stu'**)  
 *# 获取删除学生信息* del\_id = self.del\_id.GetValue()  
  
 *# 删除学生信息* **if** mod.delete(str(del\_id)) > 0:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息删除成功！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **else**:  
 **class** App(wx.Frame):  
 *"""报错窗口"""* **def** \_\_init\_\_(self):  
 wx.Frame.\_\_init\_\_(self, **None**, -1, **'消息窗口'**, size=(250, 100))  
 panel = wx.Panel(self, -1)  
 wx.StaticText(panel, -1, **'学生信息删除失败！！'**, pos=(70, 20))  
 self.Center()  
  
 app = wx.App()  
 frame = App()  
 frame.Show(**True**)  
 app.MainLoop()  
  
 **def** ClickButton(self, event):  
 num = event.GetId()  
 **if** num == 10:  
 print(**"查询操作！"**)  
  
 **from** .Select **import** Select  
 select\_wind = Select(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 select\_wind.menu()  
 select\_wind.Show()  
 self.Close()  
  
 **elif** num == 11:  
 print(**"添加操作！"**)  
 **from** .Add **import** Add  
 add\_wind = Add(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
  
 **elif** num == 12:  
 print(**"删除操作！"**)  
 **pass  
  
 elif** num == 13:  
 self.Close()

### Select.py

*#!/usr/bin/env python  
# -\*- coding: utf-8 -\*-  
# @Time : 2019/11/22 8:26  
# @Author : 李金哲  
# @Site :   
# @File : Select.py  
# @Software: PyCharm***from** .OperationWind **import** OperationWind  
**from** \_pycache\_.model **import** Model  
**import** wx  
**import** wx.grid  
  
  
**class** Select(OperationWind):  
 **def** \_\_init\_\_(self, \*args, \*\*kw):  
 *# ensure the parent's \_\_init\_\_ is called* super(Select, self).\_\_init\_\_(\*args, \*\*kw)  
 *# 创建学生信息网格* self.stu\_grid = self.CreateGrid()  
 *# 添加到vsbox\_show\_operation布局管理器* self.vsbox\_show\_operation.Add(self.stu\_grid, 0, wx.CENTER | wx.TOP | wx.FIXED\_MINSIZE, 30)  
  
 **def** CreateGrid(self):  
 *# 连接数据库* mod = Model(**'stu'**)  
 *# 获取s表中的学生信息* stu\_data = mod.findAll()  
  
 column\_names = (**'学号'**, **'姓名'**, **'年龄'**, **'性别'**, **'院系'**, **'专业'**, **'班级'**, **'年级'**)  
 key\_word = (**'id'**, **'name'**, **'age'**, **'sex'**, **'Sdept'**, **'profession'**, **'classid'**, **'grade'**)  
  
 stu\_grid = wx.grid.Grid(self.pnl)  
 stu\_grid.CreateGrid(len(stu\_data), len(stu\_data[0]) - 1)  
 **for** row **in** range(len(stu\_data)):  
 *# 确保网格序列号与数据库学生的学号一直* stu\_grid.SetRowLabelValue(row, str(stu\_data[row][**'id'**]))  
 **for** col **in** range(1, len(column\_names)):  
 stu\_grid.SetColLabelValue(col - 1, column\_names[col])  
 stu\_grid.SetCellValue(row, col - 1, str(stu\_data[row][key\_word[col]]))  
 stu\_grid.AutoSize()  
 **return** stu\_grid  
  
 **def** ClickButton(self, event):  
 num = event.GetId()  
 **if** num == 10:  
 **pass  
 elif** num == 11:  
 print(**"添加操作！"**)  
 **from** .Add **import** Add  
 add\_wind = Add(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
 **elif** num == 12:  
 print(**"删除操作！"**)  
 **from** .Delete **import** Delete  
 add\_wind = Delete(**None**, title=**"学生信息管理系统"**, name=self.GetName(), size=(1024, 668))  
 add\_wind.Show()  
 self.Close()  
  
 **elif** num == 13:  
 self.Close()