实例 3：图书管理系统登录页面

登录与注册是程序中最基本的模块。用户只有登录成功后，才可以使用应用系统中的全部功能。若用户没有登录账号，可通过注册界面设置登录账号信息。某图书管理系统的登录窗口如图 1 所示。

图1 登录界面

图 1 的窗口中包含用户名、密码、验证码、登录、注册、退出。当用户输入正确的登录信息，点击“登录”按钮后，程序会弹出一个欢迎用户的对话框，如图 2 所示。

图2 欢迎对话框

用户点击“注册”按钮后，会弹出注册用户的窗口，如图 3 所示。

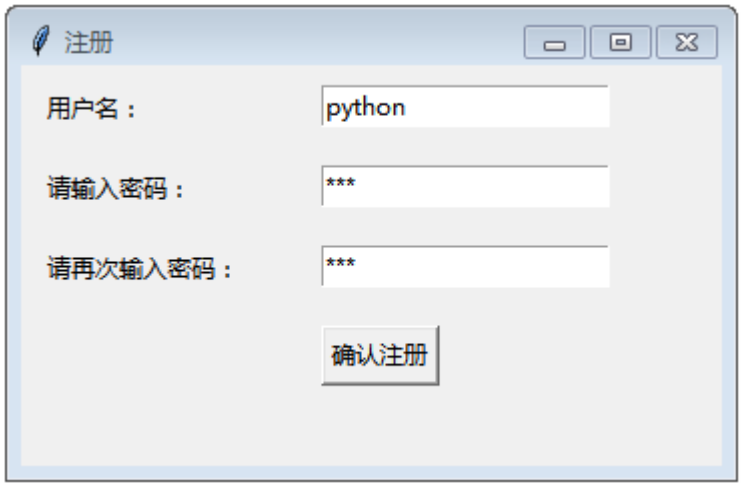

图3 注册窗口

用户填写完个人信息后，点击“确认注册”按钮，会记录用户的信息，并弹出“注册成功”对话框，如图 4 所示。

图4 注册成功对话框

本实例要求使用 tkinter，实现包含以上所示登录功能与注册功能的图形窗口。

实例目标

- 熟练地使用 tkinter 模块创建基础组件
- 熟练地使用 tkinter 模块创建消息对话框

实例分析

图书管理系统主要包含登录窗口和注册窗口，关于它们的介绍如下：

1. 登录窗口

图书管理系统的登录窗口主要包含以下元素：

- 根窗口：大小为 450*300 且不可拉伸。
- 窗口标题：内容为“图书管理系统”。
- 4 个标签组件：内容分别为“用户名”、“密码”、“验证码”和随机 6 位验证码。
- 3 个文本框：分别供用户输入“用户名”、“密码”和“验证码”

- 3 个按钮：按钮的标题分别为“登录”、“注册”、“退出”。

单击“登录”按钮后，若用户填写的密码有误，则弹出“密码错误”对话框；若用户填写的验证码有误，则弹出“验证码错误”对话框；若用户未填写用户名与密码，则弹出“用户名或密码不能为空”对话框；若用户填写的信息均无误，则弹出欢迎对话框。

如果用户并未注册过图书管理系统，则会弹出提醒用户先注册的对话框，通过选择对话框中包含的选项按钮可决定是否注册用户，或者可直接单击登录界面的“注册”按钮进入注册窗口。

2. 注册窗口

图书管理系统的注册窗口主要包含以下元素：

- 新窗口：初始大小为 350*200 且可拉伸。
- 窗口标题：内容为“注册”。
- 3 个标签组件：内容分别为“用户名：”、“请输入密码：”、“请再次输入密码：”。
- 3 个文本框：供用户输入账户、密码或确认密码。
- 1 个按钮：按钮的标题为“确认注册”

单击“确认注册”按钮后，若用户填写已注册过的名称，则弹出“用户名已存在”的对话框；若用户未填写用户名或密码，则弹出“用户名或密码不能为空”的对话框；若用户两次输入不同的密码，则弹出“前后密码不一致”的对话框；若用户填写的信息均无误，则弹出如图 4 所示的对话框。

实现本实例的过程比较简单，可直接先创建登录窗口，并为窗口中的每个按钮组件绑定与事件相关的函数。

代码实现

（1）定义一个随机产生 6 位验证码的函数 verifycode()，该函数返回随机生成的验证码，具体代码如下。

```
def verifycode():
    # 验证码
    code_list = ''
    # 每一位验证码都有三种可能（大写字母，小写字母，数字）
    for i in range(6):  # 控制验证码生成的位数
        state = random.randint(1, 3)
        if state == 1:
            first_kind = random.randint(65, 90)  # 大写字母
            random_uppercase = chr(first_kind)
            code_list = code_list + random_uppercase
        elif state == 2:
```

```
            second_kinds = random.randint(97, 122)  # 小写字母
            random_lowercase = chr(second_kinds)
            code_list = code_list + random_lowercase
        elif state == 3:
            third_kinds = random.randint(0, 9)
            code_list = code_list + str(third_kinds)
    return code_list
```

（2）定义一个处理用户登录业务的函数 usr_log_in ()，该函数中需先从登录窗口中获取用户输入的用户名、密码和验证码，其次尝试从本地磁盘中访问保存用户信息的文件，然后通过 if-else 语句判断用户名、密码与验证码是否匹配，并根据不同的匹配情况弹出相应的对话框，具体代码如下。

```
# 登录函数
def usr_log_in():
    # 输入框获取用户名、密码、验证码
    usr_name = var_usr_name.get()
    # 密码
    usr_pwd = var_usr_pwd.get()
    # 验证码
    var_vercode = var_usr_vercode.get()
    # 从本地字典获取用户信息，如果没有则新建本地数据库
    try:
        with open('usr_info.pickle', 'rb') as usr_file:
            usrs_info = pickle.load(usr_file)
    except FileNotFoundError:
        with open('usr_info.pickle', 'wb') as usr_file:
            usrs_info = {'admin': 'admin'}
            pickle.dump(usrs_info, usr_file)
    # 判断用户名和密码是否匹配
    if usr_name in usrs_info:
        if usr_pwd == usrs_info[usr_name]:
            if var_vercode == verification_Code:
```

```
                tk.messagebox.showinfo(title='welcome',
                    message='欢迎您：' + usr_name)
            else:
                tk.messagebox.showerror(message='验证码错误')
        else:
            tk.messagebox.showerror(message='密码错误')
    # 用户名密码不能为空
    elif usr_name == '' or usr_pwd == '':
        tk.messagebox.showerror(message='用户名或密码不能为空')
    # 不在磁盘中弹出是否注册的对话框框
    else:
        is_signup = tk.messagebox.askyesno('欢迎', '您还没有注册，是否现在注册')
        if is_signup:
            usr_sign_up()
```

（3）定义一个处理用户注册业务的函数 usr_sign_up ()，该函数中包括两个子业务，一是创建注册窗口，另一是处理注册业务，具体代码如下。

```
# 注册函数
def usr_sign_up():
    # 确认注册时的相应函数
    def signtowcg():
        # 获取输入框内的内容
        nn = new_name.get()
        np = new_pwd.get()
        npf = new_pwd_confirm.get()
        # 本地加载已有用户信息,如果没有则已有用户信息为空
        try:
            with open('usr_info.pickle', 'rb') as usr_file:
                exist_usr_info = pickle.load(usr_file)
        except FileNotFoundError:
            exist_usr_info = {}
```

```
            # 检查用户名已存在、密码为空、密码前后不一致
        if nn in exist_usr_info:
            tk.messagebox.showerror('错误', '用户名已存在')
        elif np == '' or nn == '':
            tk.messagebox.showerror('错误', '用户名或密码不能为空')
        elif np != npf:
            tk.messagebox.showerror('错误', '密码前后不一致')
        # 注册信息没有问题则将用户名、密码存入数据库
        else:
            exist_usr_info[nn] = np
            with open('usr_info.pickle', 'wb') as usr_file:
                pickle.dump(exist_usr_info, usr_file)
            tk.messagebox.showinfo('欢迎', '注册成功')
            # 注册成功关闭注册框
            window_sign_up.destroy()
    # 新建注册界面
    window_sign_up = tk.Toplevel(window)
    window_sign_up.geometry('350x200')
    window_sign_up.title('注册')
    # 用户名变量及标签、输入框
    new_name = tk.StringVar()
    tk.Label(window_sign_up, text='用户名：').place(x=10, y=10)
    tk.Entry(window_sign_up, textvariable=new_name).place(x=150, y=10)
    # 密码变量及标签、输入框
    new_pwd = tk.StringVar()
    tk.Label(window_sign_up, text='请输入密码：').place(x=10, y=50)
    tk.Entry(window_sign_up, textvariable=new_pwd,
        show='*').place(x=150, y=50)
    # 重复密码变量及标签、输入框
    new_pwd_confirm = tk.StringVar()
```

```
    tk.Label(window_sign_up, text='请再次输入密码：').place(x=10, y=90)
    tk.Entry(window_sign_up,
        textvariable=new_pwd_confirm, show='*').place(x=150, y=90)
    # 确认注册按钮及位置
    bt_confirm_sign_up = tk.Button(window_sign_up, text='确认注册',
        command=signtowcg)
    bt_confirm_sign_up.place(x=150, y=130)
```

（4）定义一个处退出图书管理系统的函数 usr_sign_quit()，具体代码如下。

```
# 退出的函数
def usr_sign_quit():
    window.destroy()
```

（5）创建登录窗口，并为该窗口中的按钮绑定前面定义的事件函数，具体代码如下。

```
# 验证码值
verification_Code = verifycode()
# 窗口
window = tk.Tk()
window.title('图书管理系统')
window.geometry('450x300')
# 设置登录窗口大小不可拉伸
window.resizable(width=False, height=False)
# 画布放置图片
canvas = tk.Canvas(window, height=300, width=500)
imagefile = ImageTk.PhotoImage(file=r'背景.png')
image = canvas.create_image(0, 0, anchor='nw', image=imagefile)
canvas.pack(side='top')
# 创建 4 个标签
tk.Label(window, text='用户名').place(x=100, y=110)
tk.Label(window, text='密　码').place(x=100, y=150)
tk.Label(window, text='验证码').place(x=100, y=190)
tk.Label(window, text=verification_Code).place(x=310, y=190)
```

```
# 用户名输入框
var_usr_name = tk.StringVar()
entry_usr_name = tk.Entry(window, textvariable=var_usr_name)
entry_usr_name.place(x=160, y=110)
# 密码输入框
var_usr_pwd = tk.StringVar()
entry_usr_pwd = tk.Entry(window, textvariable=var_usr_pwd, show='*')
entry_usr_pwd.place(x=160, y=150)
# 验证码输入框
var_usr_vercode = tk.StringVar()
var_usr_vercode = tk.Entry(window, textvariable=var_usr_vercode)
var_usr_vercode.place(x=160, y=190)
# 登录、注册、退出按钮
bt_login = tk.Button(window, text='登录', command=usr_log_in)
bt_login.place(x=140, y=230)
bt_logup = tk.Button(window, text='注册', command=usr_sign_up)
bt_logup.place(x=210, y=230)
bt_logquit = tk.Button(window, text='退出', command=usr_sign_quit)
bt_logquit.place(x=280, y=230)
# 主循环
window.mainloop()
```

代码测试

运行程序弹出登录窗口，单击该窗口中的“注册”按钮后弹出注册窗口，分别在输入框中输入“itcast”、“123”、“1234”，单击“确认注册”按钮之后的界面如图 5 所示。

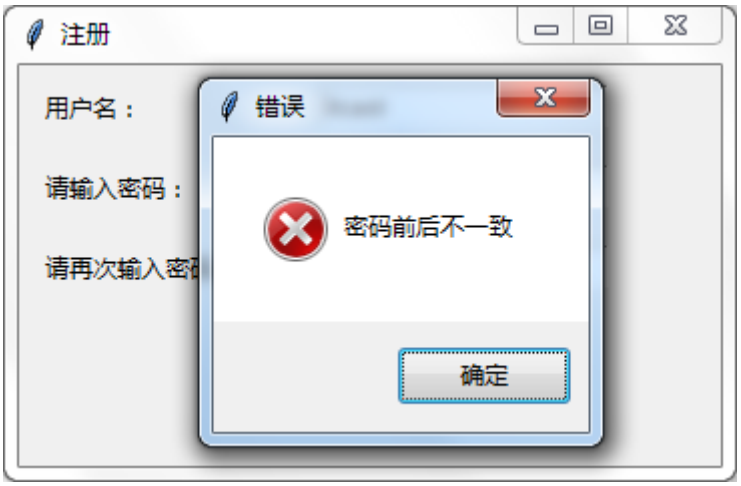

图5 密码不一致对话框

单击图 5 中的“确定”按钮返回注册窗口，删除再次输入密码文本框中的最后一个字符后，再次单击“确认注册”按钮之后的界面如图 6 所示。

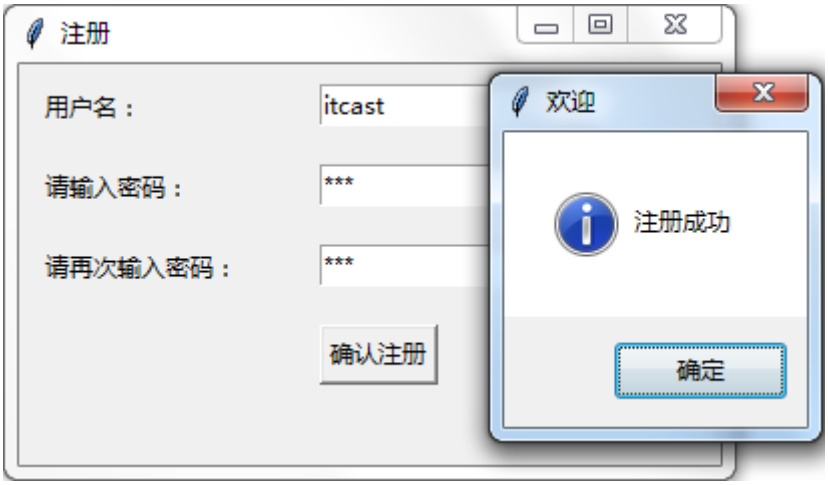

图6 注册成功对话框

单击图 6 中的“确定”按钮返回登录窗口，在该窗口中填写刚刚注册的用户信息，并按照提示输入验证码，单击“登录”按钮之后的界面如图 7 所示。

图7 登录成功对话框

单击图 7 的“确定”按钮即可关闭对话框。