

Python语言程序设计

Design and Programming of The Python Language

主讲教师：张小东

联系方式：z_xiaodong7134@163.com

答疑地点：宋健研究院514

第6章 图形用户界面

主要内容

- 产品规划设计
- 产品布局设计
- 产品功能实现

===产品设计规划===

● 产品设计规划—Python程序交互

交流软件需求

Tile

多文本—
数据展示

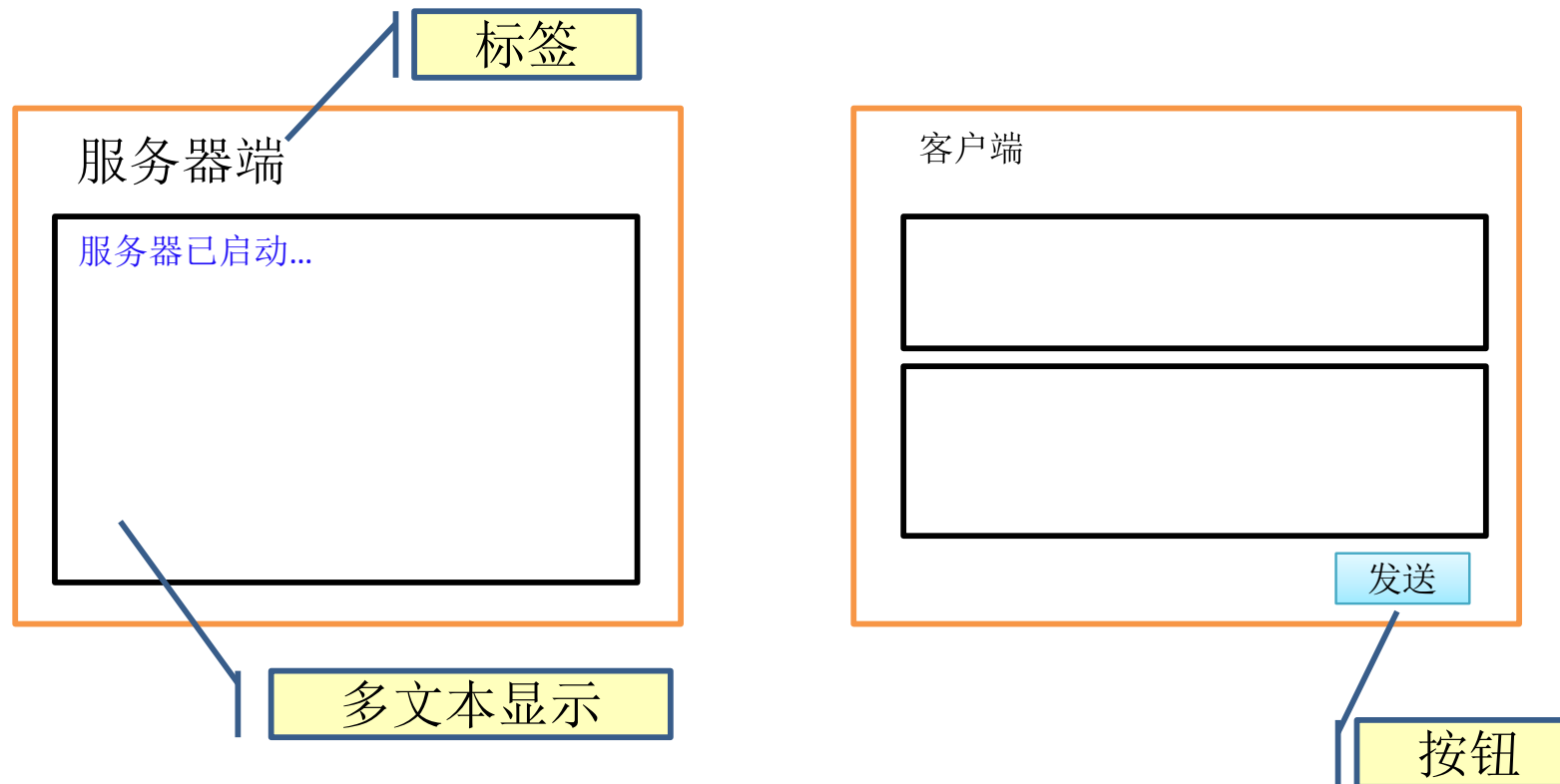
按钮
文本输入框

按钮



===产品设计规划===

- 产品设计规划—Python程序交互

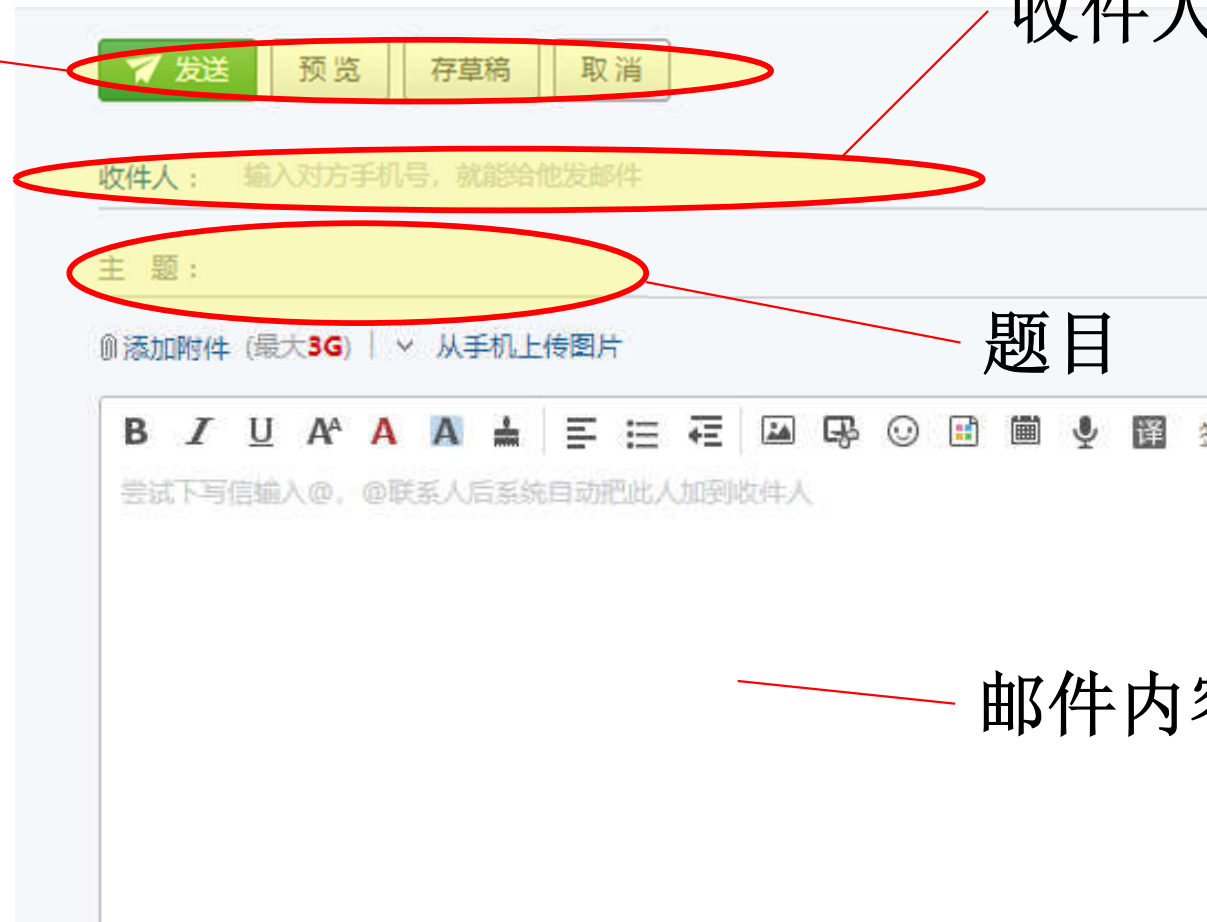


===产品设计规划===

- 产品设计规划—邮件之需求.发邮件

按钮

收件人



The image shows a typical email composition window. At the top, there are four buttons: '发送' (Send), '预览' (Preview), '存草稿' (Save Draft), and '取消' (Cancel). Below these are three input fields: '收件人:' (To:) with a hint '输入对方手机号, 就能给他发邮件' (Enter the other party's phone number, you can email him), '主题:' (Subject:), and a line for attachments with the text '添加附件 (最大3G) | 从手机上传图片'. Below the subject field is a rich text editor toolbar with icons for bold, italic, underline, text color, background color, bulleted list, numbered list, link, image, video, emoji, gallery, calendar, voice recording, and translation. The main body of the email is a large text area. Red lines and ovals highlight specific elements: a red oval around the '发送' button is labeled '按钮'; a red oval around the '收件人:' field is labeled '收件人'; a red oval around the '主题:' field is labeled '题目'; and a red line pointing to the main text area is labeled '邮件内容'.

题目

邮件内容

===产品设计规划===

● 邮件之需求界面设计

59

Tile

发送 Button 重置

Label

接收人 Entry

发送人 Entry

题目 Entry

邮件内容 Label

Text

===产品设计规划===

- 给一行文字加密码与解密界面设计

Tile

Label

加密解密Button

原文Entry

密文Entry

第6章 图形用户界面

主要内容

- 产品规划设计
- 产品布局设计
- 产品功能实现

===产品设计规划===

◆ 邮件组件创建与摆放

➤ Pack布局

组件.pack([属性名=值, 属性名=值,])

属性名	属性简析	取值	取值说明
fill	设置组件是否向水平或垂直方向填充	X、Y、BOTH 和NONE	fill = X (水平方向填充) fill = Y (垂直方向填充) fill = BOTH (水平和垂直) NONE 不填充
expand	设置组件是否展开, 当值为YES时, side选项无效。组件显示在父容器中心位置; 若fill选项为BOTH,则填充父组件的剩余空间。默认为不展开	YES、NO (1、0)	expand=YES expand=NO
side	设置组件的对齐方式	LEFT、TOP、RIGHT、BOTTOM	值为左、上、右、下
ipadx、ipady	设置x方向 (或者y方向) 内部间隙 (子组件之间的间隔)	可设置数值, 默认是0	非负整数, 单位为像素
padx、pady	设置x方向 (或者y方向) 外部间隙 (与之并列的组件之间的间隔)	可设置数值, 默认是0	非负整数, 单位为像素
anchor	锚选项, 当可用空间大于所需求的尺寸时, 决定组件被放置于容器的何处	N、E、S、W、NW、NE、SW、SE、CENTER (默认值为CENTER)	表示八个方向以及中心

===产品布局设计===

◆ 邮件组件创建与摆放

➤ Pack布局

组件.pack([属性名=值, 属性名=值,])

```
from tkinter import *  
root = Tk()  
Label(root, text="Red Sun", bg="red", fg="white").pack()  
Label(root, text="Green Grass", bg="green", fg="black").pack()  
Label(root, text="Blue Sky", bg="blue", fg="white").pack()  
mainloop()
```

```
w = Label(root, text="red", bg="red", fg="white")  
w.pack(padx=5, pady=10, side=LEFT)  
w = Label(root, text="green", bg="green", fg="black")  
w.pack(padx=5, pady=20, side=LEFT)  
w = Label(root, text="blue", bg="blue", fg="white")  
w.pack(padx=5, pady=20, side=LEFT)
```

【问题】凯撒密码是古罗马凯撒大帝用来对军事情报进行加密的算法，它采用了替换方式，将情报中的每一个英文字符循环替换为字母表序列中该字符后面第三个字符，对应关系如下：

原文：A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

密文：D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

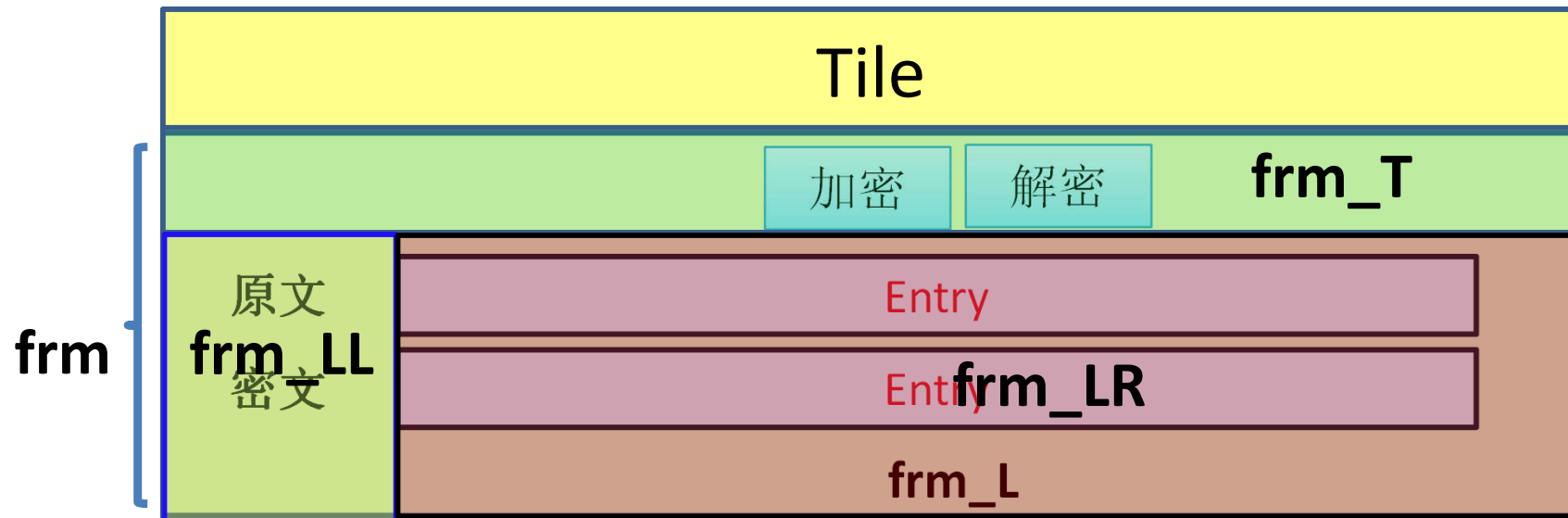
原文字符P，其密文字符C满足如下条件：

$$C = (P + 3) \bmod 26$$

$$P = (C - 3) \bmod 26$$

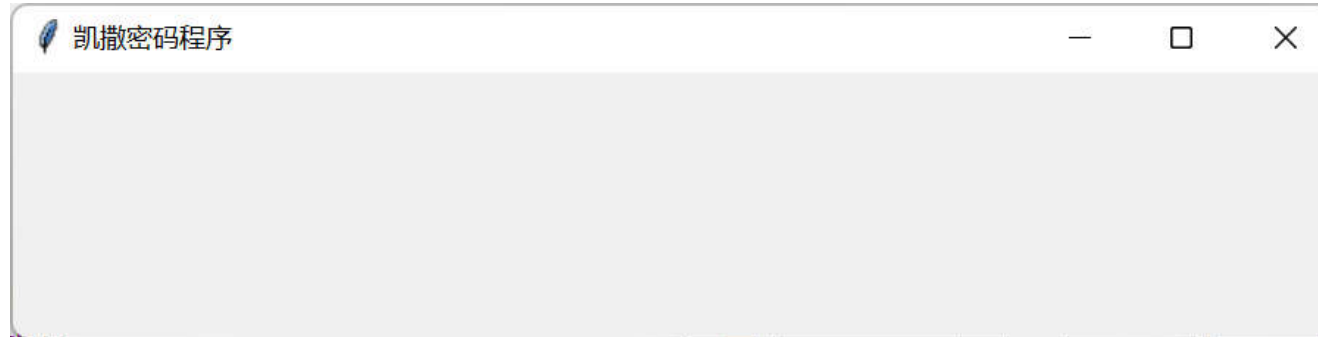
===产品设计规划===

【界面设计】



===产品设计规划===

【界面设计】



```
from tkinter import * #引入图形界面模块
def CaesarTF():
    root=Tk()
    root.title('凯撒密码程序')
    frm = Frame(root)
    root.geometry('600x120')
    frm.pack()
    root.mainloop()
CaesarTF()
```

===产品设计规划===

【界面设计】



```

from tkinter import * #引入图形界面模块
def CaesarTF():
    .....
    frm = Frame(root)
    frm_T=Frame(frm)
    frm_T.pack(side=TOP) #放置按钮
    Button(frm_T,text="加密",width = 8,height = 1).pack(side=LEFT)
    Button(frm_T,text="解密",width = 8,height = 1).pack(side=RIGHT)
    frm_T.pack(side=TOP)
    .....
    root.mainloop()
CaesarTF()

```

===产品设计规划===

【界面设计】



```

from tkinter import * #引入图形界面模块
def CaesarTF():
    .....
    frm_T.pack(side=TOP) #放置按钮
    frm_L = Frame(frm)
    frm_LL=Frame(frm_L) #放置左边的文本说明 label组件
    Label(frm_LL,text='原文',font=('Fangsong',18)).pack(side=TOP)
    Label(frm_LL,text='密文',font=('Fangsong',18)).pack(side=TOP)
    frm_LL.pack(side=LEFT)
    frm_RL=Frame(frm_L) #放置右边的单行输入框, Entry组件
    Entry(frm_RL,textvariable=org,width=24,font=('Fangsong',18)).pack()
    Entry(frm_RL,textvariable=cip,width=24,font=('Fangsong',18)).pack()
    frm_RL.pack(side=RIGHT)
    frm_L.pack(side=TOP)
    .....
    root.mainloop()
CaesarTF()

```

```

from string import ascii_lowercase
from string import ascii_uppercase
class Caesar():
    def __init__(self,txt):
        self.__txt = txt
    def setSource(self,txt):
        self.__txt = txt

```

```

def encryption(self):
    lst=[]
    for i in self.__txt:
        if i in ascii_lowercase: #小写字母
            a = ord(i)
            n = (a - 97 + 3) % 26
            s = chr(n + 97)
            lst.append(s)
        elif i in ascii_uppercase:#大写字母
            a = ord(i)
            n = (a - 65 + 3) % 26
            s = chr(n + 65)
            lst.append(s)
        else: #数字
            lst.append(i)#将列表转换为字符串
    return ''.join(lst)

```

```

def decryption(self):
    lst=[]
    for i in self.__txt:
        if i in ascii_lowercase:
            a = ord(i)
            n = (a - 97 - 3) % 26
            s = chr(n + 97)
            lst.append(s)
        elif i in ascii_uppercase:
            a = ord(i)
            n = (a - 65 - 3) % 26
            s = chr(n + 65)
            lst.append(s)
        else:
            lst.append(i)
    return ''.join(lst)

```

```

def __del__(self):
    print('文字解析完毕')

```


===产品功能实现===

◆ 功能实现.按钮响应

a=Caesar("")

Button(...,command=enCaesar,...)

```
def enCaesar():
    o = org.get()
    a.setSource(o)
    h=a.encryption()
    cip.set(h)
```



```
org=StringVar()
cip=StringVar()
```

command=deCaesar

Entry(...,textvariable=org, ...)

```
def deCaesar():
    c = cip.get()
    a.setSource(c)
    m=a.decryption()
    org.set(m)
```

===产品功能实现===

◆ 功能实现.按钮响应

```
frm_L = Frame(frm)
frm_RL=Frame(frm_L)
org = StringVar()
cip = StringVar()
Entry(frm_RL,width=24,textvariable=org,font=('Fangsong',18)).pack()
Entry(frm_RL,width=24,textvariable=cip,font=('Fangsong',18)).pack()
frm_RL.pack(side=RIGHT)
```

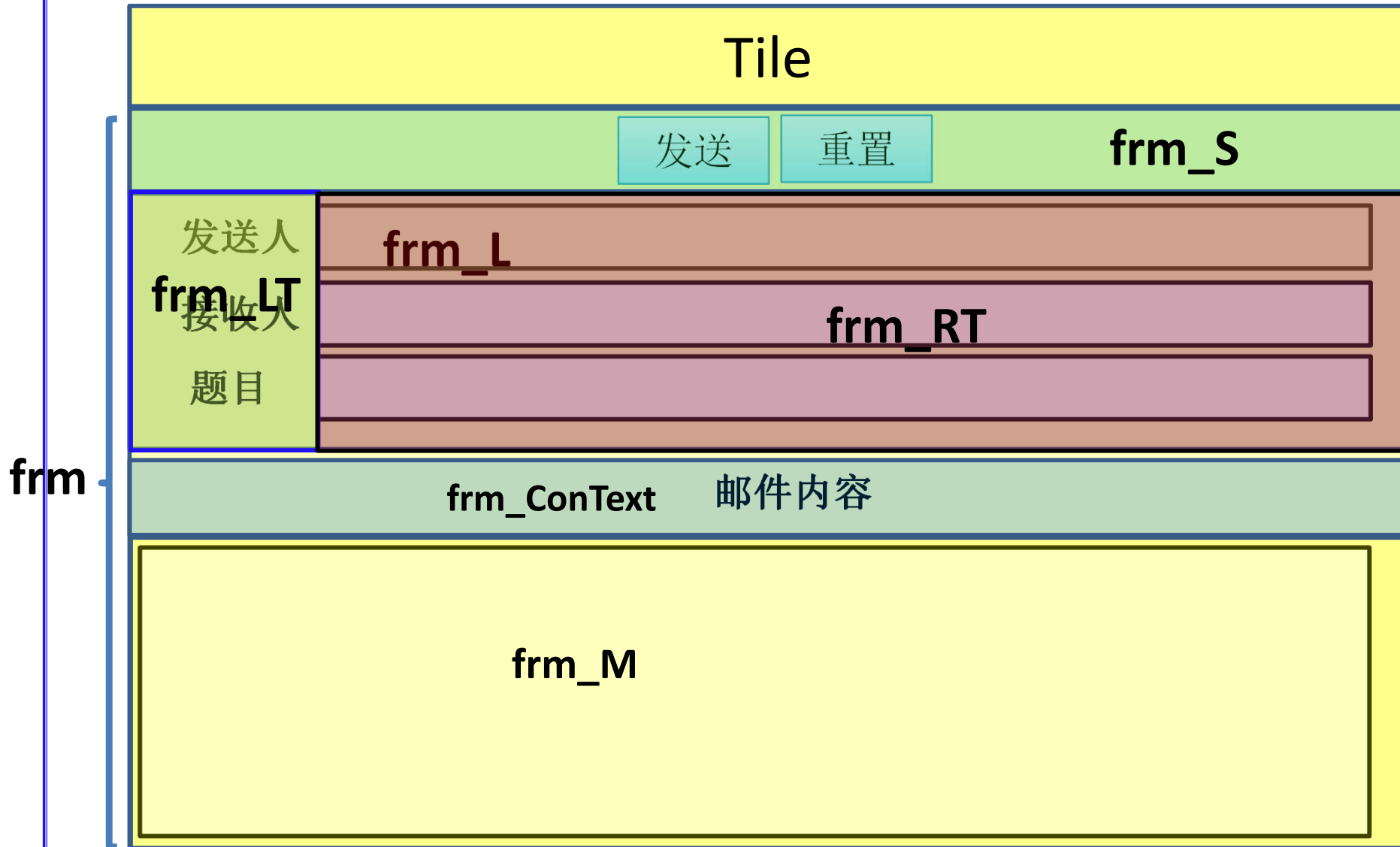
```
def enCaesar():
    o = org.get()
    a.setSource(o)
    h=a.encryption()
    cip.set(h)
```

```
def deCaesar():
    c = cip.get()
    a.setSource(c)
    m=a.decryption()
    org.set(m)
```

```
frm_T=Frame(frm)
frm_T.pack(side=TOP)
Button(frm_T,text="加密",command= enCaesar).pack(side=LEFT)
Button(frm_T,text="解密",command= deCaesar).pack(side=RIGHT)
frm_T.pack(side=TOP)
frm_LL=Frame(frm_L)
```

===产品设计规划===

◆ 邮件之需求.布局Frame+pack



===产品设计规划===

◆ 邮件组件创建与摆放

组件(根对象,[属性列表])

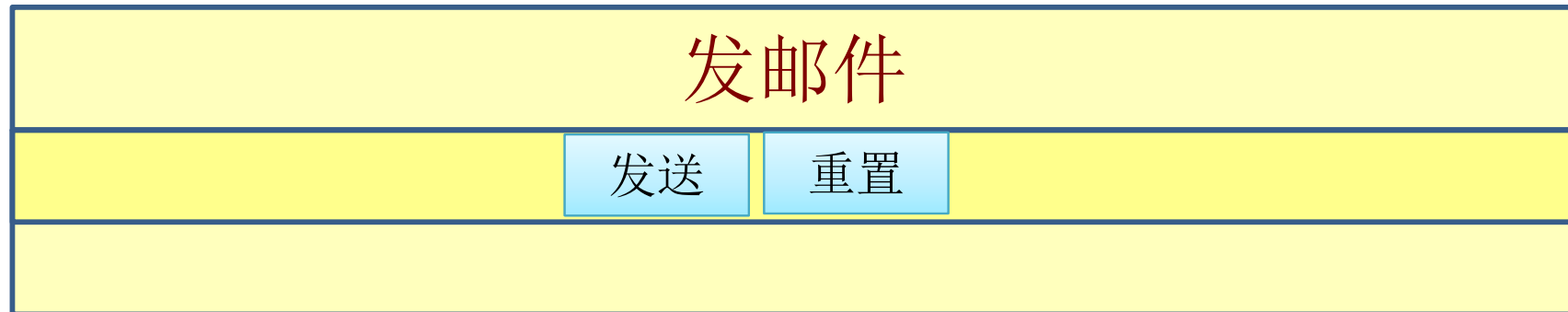
发邮件

➤ 窗口生成

```
from tkinter import * # 引用Tk模块
root = Tk()           # 初始化Tk()
root.title("发邮件")
frm = Frame(root)
root.geometry('470x280')
frm.pack()
root.mainloop()
```

===产品设计规划===

◆ 邮件组件创建与摆放



➤ 放置按钮

.....

```
frm = Frame(root)
```

```
frm_S = Frame(frm)
```

```
Button(frm_S, text="发送", width=6, height=1, font=('Arial', 10)).pack(side=LEFT)
```

```
Button(frm_S, text="重置", width=6, height=1, font=('Arial', 10)).pack(side=RIGHT)
```

```
frm_S.pack(side=TOP)
```

.....

```
frm.pack()
```

◆ 邮件组件创建与摆放

发邮件	
	<input type="button" value="发送"/> <input type="button" value="重置"/>
题目	<input type="text"/>
发送人	<input type="text"/>
接收人	<input type="text"/>

➤ 放置标签和输入框

```
frm_L = Frame(frm)
```

```
frm_LT = Frame(frm_L)
```

```
Label(frm_LT, text = '发件人', font = ('Arial',12)).pack(side=TOP)
```

```
Label(frm_LT, text = '接收人', font = ('Arial',12)).pack(side=TOP)
```

```
Label(frm_LT, text = ' 题目', font = ('Arial',12)).pack(side=TOP)
```

```
frm_LT.pack(side=LEFT)
```

```
frm_RT = Frame(frm_L)
```

```
var_sender = StringVar()
```

```
var_title = StringVar()
```

```
var_receiver = StringVar()
```

```
Entry(frm_RT, textvariable=var_sender, width = 30, font = ('Verdana',15)).pack()
```

```
Entry(frm_RT, textvariable= var_receiver, width = 30, font = ('Verdana',15)).pack()
```

```
Entry(frm_RT, textvariable= var_title, width = 30, font = ('Verdana',15)).pack()
```

```
frm_RT.pack(side=RIGHT)
```

```
frm_L.pack(side = TOP)
```

===产品设计规划===

◆ 邮件组件创建与摆放

➤ 放置标签和文本框

```
#third-label
frm_ConText = Frame(frm)
Label(frm_ConText, text = '发送内容', font = ('Arial',12)).pack(side=TOP)
frm_ConText.pack(side = TOP)
#forth-inputcontext
frm_M = Frame(frm)
t_show = Text(frm_M, width=34, height=6, font = ('Verdana',15))
t_show.insert('1.0', 'fdfd')
t_show.pack()
frm_M.pack()
```

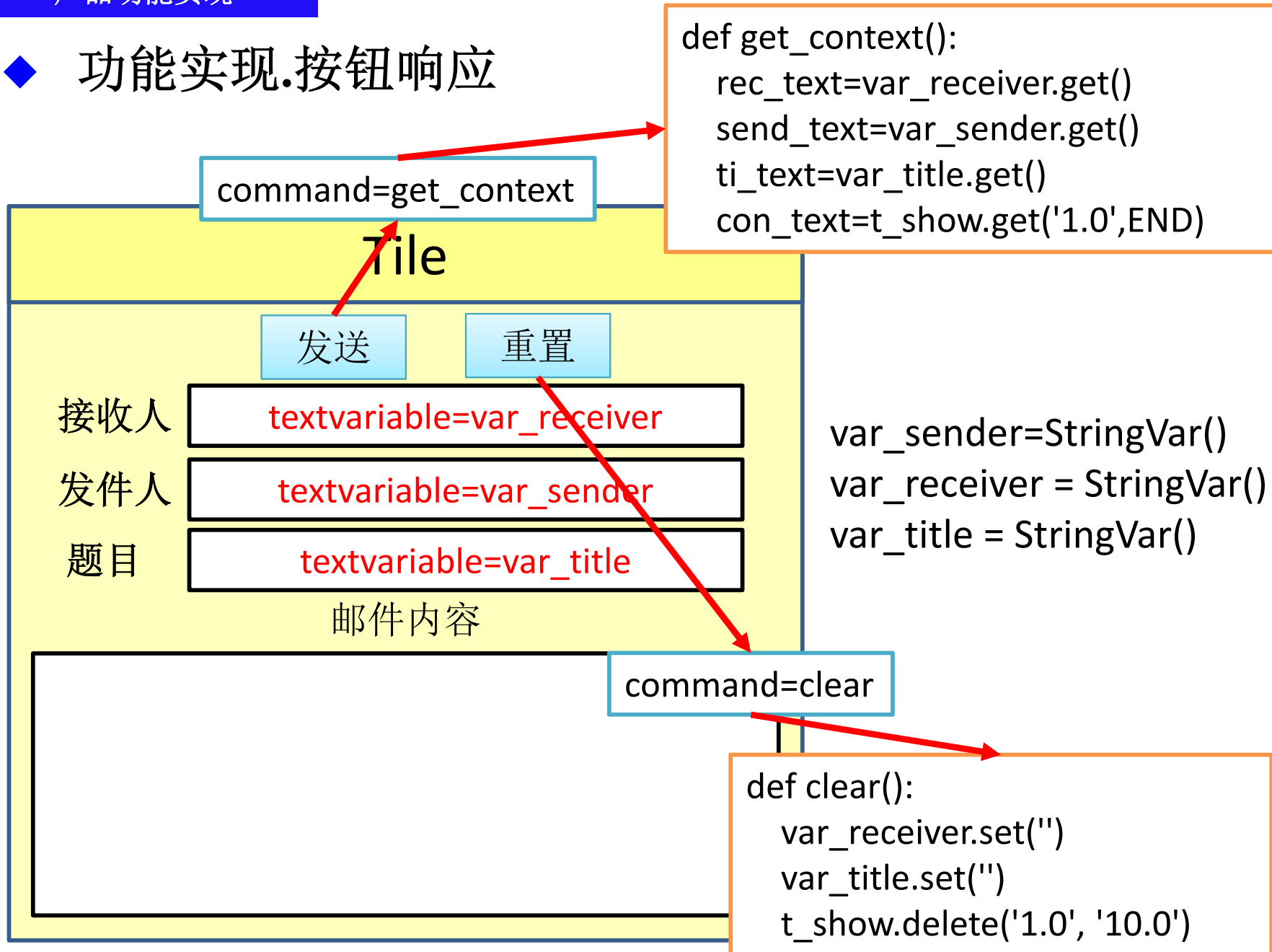
第6章 图形用户界面

主要内容

- 产品规划设计
- 产品布局设计
- 产品功能实现

===产品功能实现===

◆ 功能实现.按钮响应



===产品功能实现===

◆ 功能实现.邮件类

```
import smtplib
from email.header import Header
from email.mime.text import MIMEText
class SendMail:
    def __init__(self,smtp_host,mail_user,mail_pwd,toaddr,subject,context):
        #邮箱服务器信息
        self.__host=smtp_host
        self.__user=mail_user
        self.__pwd=mail_pwd
        #设置发送邮件内容
        self.__toaddr=toaddr.strip()
        self.__subject=subject.strip()
        self.__context=context.strip()
```

===产品功能实现===

◆ 功能实现.邮件类

```
def __del__(self):  
    print('bye')
```

```
def sendE(self):  
    print("开始发送邮件.....")  
    msg=MIMEText(self.__context, 'plain', 'utf-8') # 内容, 格式, 编码  
    msg['Subject']=self.__subject  
    msg['From']="{}".format(self.__user)  
    msg['To']=self.__toaddr #", ".join(self.__toaddr)  
    try:  
        server=smtplib.SMTP_SSL(self.__host, 465)  
        server.login(self.__user,self.__pwd)  
        server.sendmail(self.__user,self.__toaddr,msg.as_string())  
        print("Send successfully")  
        server.quit()  
    except smtplib.SMTPException as e:  
        print(e)  
        server.quit()  
        return False  
    return True
```

===产品功能实现===

◆ 功能实现.邮件类

```
from sendmailClassnew import SendMail
def get_context():
    send_text=var_sender.get()
    rec_text=var_receiver.get()
    if rec_text=="":
        tkinter.messagebox.showinfo('提示',"接收人不能为空")
        return
    ti_text=var_title.get()
    con_text=t_show.get('1.0',END)
    s=SendMail("smtp.163.com",send_text,'*****',rec_text,ti_text,con_text)
    if s.sendE():
        tkinter.messagebox.showinfo('提示',"发送成功")
    else:
        tkinter.messagebox.showinfo('提示',"发送失败")
```

===产品设计规划===

● 邮件之需求界面设计

Tile

接收

重置

Button

Label

邮箱地址

Entry

登录账号

Entry

登录密码

Entry

邮件编号

Entry

邮件内容

Label

Text

本章小结

- 产品设计规划
- 产品布局设计
- 产品功能实现