# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

**ФЕДЕРАЦИИ**

# Федеральное государственное автономное образовательное учреждение высшего образования

**«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

# Кафедра инфокоммуникаций Институт цифрового развития

**ОТЧЁТ**

# по лабораторной работе

Дисциплина: «Объектно – ориентированное программирование»

|  |
| --- |
| Выполнил студент группы  ИВТ-б-о-21-1 |
| Богдашов А.В. « » 20 г. |
| Подпись студента |
| Работа защищена « » 20 г. |
| Проверил доцент  Кафедры инфокоммуникаций, старший преподаватель  Воронкин Р.А.  (подпись) |

Ставрополь 2023

# Обработка событий и рисование в Tkinter

**Цель работы:** приобретение навыков улучшения графического интерфейса пользователя GUI с помощью обработки событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.

# Ход работы:

**Задание 1.** Решите задачу с использованием Tkinter: напишите программу, состоящую из двух списков Listbox . В первом будет,

например, перечень товаров, заданный программно. Второй изначально пуст, пусть это будет перечень покупок. При клике на одну кнопку товар должен переходить из одного списка в другой. При клике на вторую

кнопку – возвращаться (человек передумал покупать). Предусмотрите возможность множественного выбора элементов списка и их

перемещения.

Листинг программы:

#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
import tkinter as tk  
  
  
class ShoppingApp:  
 def \_\_init\_\_(self, roote):  
 self.root = roote  
 self.root.title("Список покупок")  
  
 # Список товаров  
 self.available\_items = ["Яблоко", "Банан", "Молоко", "Хлеб", "Яйца", "Сахар", "Чай", "Кофе"]  
  
 # Список покупок  
 self.shopping\_list = []  
  
 # Создаем Listbox для доступных товаров  
 self.available\_listbox = tk.Listbox(roote, selectmode=tk.MULTIPLE)  
 for item in self.available\_items:  
 self.available\_listbox.insert(tk.END, item)  
 self.available\_listbox.pack(padx=10, pady=10, side=tk.LEFT)  
  
 # Создаем Listbox для списка покупок  
 self.shopping\_listbox = tk.Listbox(roote, selectmode=tk.MULTIPLE)  
 self.shopping\_listbox.pack(padx=10, pady=10, side=tk.RIGHT)  
  
 # Кнопка для добавления товаров в список покупок  
 self.add\_button = tk.Button(roote, text="Добавить в покупки", command=self.add\_to\_shopping\_list)  
 self.add\_button.pack(pady=10)  
  
 # Кнопка для удаления товаров из списка покупок  
 self.remove\_button = tk.Button(roote, text="Убрать из покупок", command=self.remove\_from\_shopping\_list)  
 self.remove\_button.pack(pady=20)  
  
 def add\_to\_shopping\_list(self):  
 selected\_items = list(self.available\_listbox.curselection())  
 selected\_items.reverse()  
 for index in selected\_items:  
 item = self.available\_items[index]  
 if item not in self.shopping\_list:  
 self.shopping\_list.append(item)  
 self.shopping\_listbox.insert(tk.END, item)  
 self.available\_listbox.delete(index)  
  
 def remove\_from\_shopping\_list(self):  
 selected\_items = list(self.shopping\_listbox.curselection())  
 selected\_items.reverse()  
 for index in selected\_items:  
 item = self.shopping\_list[index]  
 self.shopping\_list.remove(item)  
 self.shopping\_listbox.delete(index)  
 self.available\_listbox.insert(tk.END, item)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 app = ShoppingApp(root)  
 root.mainloop()

Результат работы программы:

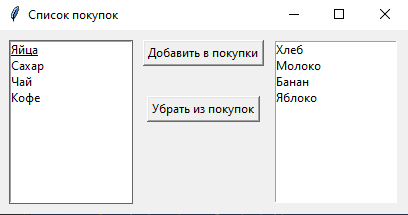


Рисунок 1. Пример работы программы к первому заданию

**Задание 2.** Решите задачу с использованием Tkinter: напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из него в список (экземпляр Listbox ). При двойном клике ( <Double-Button-1> ) по

элементу-строке списка, она должна копироваться в текстовое поле.

Листинг программы:

#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
import tkinter as tk  
  
  
def on\_enter(\_):  
 text = entry.get()  
 if text:  
 listbox.insert(tk.END, text)  
 entry.delete(0, tk.END)  
  
  
def on\_double\_click(\_):  
 selected\_index = listbox.curselection()  
 if selected\_index:  
 selected\_text = listbox.get(selected\_index)  
 entry.delete(0, tk.END)  
 entry.insert(0, selected\_text)  
  
  
root = tk.Tk()  
root.title("Программа с полем ввода и списком")  
  
entry = tk.Entry(root)  
entry.pack(pady=10)  
entry.bind("<Return>", on\_enter)  
  
listbox = tk.Listbox(root)  
listbox.pack(expand=True, fill=tk.BOTH)  
listbox.bind("<Double-Button-1>", on\_double\_click)  
  
root.mainloop()

Результат работы программы:

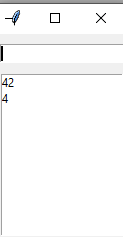


Рисунок 2. Пример работы программы ко второму заданию

**Задание 3.** Решите задачу с использованием Tkinter: напишите программу по описанию. Размеры многострочного текстового поля определяются значениями, введенными в однострочные текстовые поля. Изменение размера происходит при нажатии мышью на кнопку, а также при нажатии клавиши Enter. Цвет фона экземпляра Text светлосерый ( lightgrey ), когда поле не в фокусе, и белый, когда имеет фокус. Событие получения фокуса обозначается как <FocusIn> , потери – как <FocusOut>

. Для справки: фокус перемещается по виджетам при нажатии Tab,

Ctrl+Tab, Shift+Tab, а также при клике по ним мышью (к кнопкам последнее не относится).

Листинг программы:

Результат работы программы:

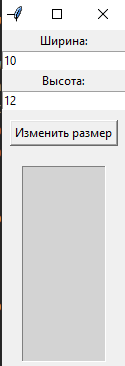


Рисунок 3. Пример работы программы к третьему заданию

**Задание 4.** Нарисуйте изображение, как на примере. Для отрисовки травы используйте цикл.

Листинг программы:

#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
import tkinter as tk  
import math  
  
  
def draw\_house(canvas):  
 # Рисуем дом  
 canvas.create\_rectangle(175, 150, 375, 300, fill="lightblue", outline="black")  
 canvas.create\_polygon(175, 150, 275, 50, 375, 150, fill="lightblue", outline="black")  
  
  
def draw\_sun(canvas):  
 # Рисуем солнце  
 canvas.create\_oval(400, 50, 450, 100, fill="orange", outline="yellow")  
  
  
def draw\_grass(canvas):  
 # Рисуем траву  
 for i in range(10, 400, 20):  
 y = 300 + 20 \* math.sin(i / 40) # Увеличиваем высоту травы  
 canvas.create\_line(i, 300, i + 10, y, fill="green")  
  
  
def main():  
 root = tk.Tk()  
 root.title("Домик с солнцем и травой")  
  
 canvas = tk.Canvas(root, width=500, height=400)  
 canvas.pack()  
  
 # Рисуем элементы  
 draw\_house(canvas)  
 draw\_sun(canvas)  
 draw\_grass(canvas)  
  
 root.mainloop()  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Результат работы программы:

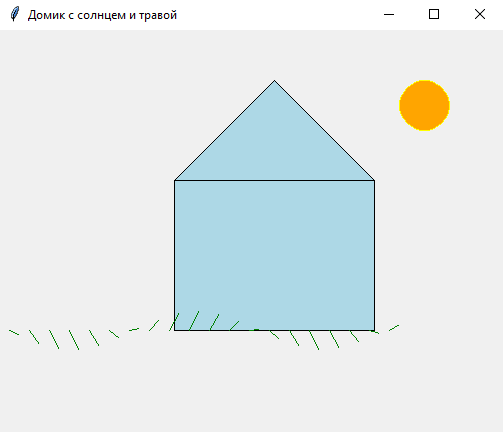


Рисунок 4. Пример работы программы к четвёртому заданию

# Задание 5.

Листинг программы:

#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
from tkinter import \*  
  
  
def motion(event):  
 current\_x, current\_y = c.coords(ball)[:2]  
 target\_x, target\_y = event.x, event.y  
  
 step\_x = 1 if target\_x > current\_x else -1  
 step\_y = 1 if target\_y > current\_y else -1  
  
 c.move(ball, step\_x, step\_y)  
  
 if (step\_x > 0 and current\_x < target\_x) or (step\_x < 0 and current\_x > target\_x) or \  
 (step\_y > 0 and current\_y < target\_y) or (step\_y < 0 and current\_y > target\_y):  
 root.after(10, motion, event)  
  
  
def on\_click(event):  
 motion(event)  
  
  
root = Tk()  
c = Canvas(root, width=300, height=200, bg="white")  
c.pack()  
  
ball = c.create\_oval(0, 100, 40, 140, fill='green')  
  
c.bind("<Button-1>", on\_click)  
  
root.mainloop()

Результат работы программы:

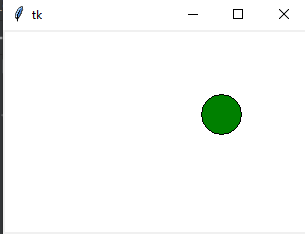


Рисунок 5. Пример работы программы к пятому примеру

# Ответы на контрольные вопросы:

1. **Каково назначение виджета ListBox?**

Виджет ListBox в Tkinter предназначен для отображения списка элементов, из которых пользователь может выбирать один или

несколько. Этот виджет предоставляет прокручиваемый список элементов.

# Каким образом осуществляется связывание событие или действие с виджетом Tkinter?

Связывание событий в Tkinter осуществляется с использованием метода bind(). Этот метод позволяет привязывать функции-обработчики к определенным событиям, таким как нажатие кнопки, перемещение

мыши и другие.

# Какие существуют типы событий в Tkinter? Приведите примеры.

Существует множество типов событий в Tkinter. Некоторые из них включают:

Button-1: Левая кнопка мыши.

Button-2: Средняя кнопка мыши (если она есть). Button-3: Правая кнопка мыши.

Motion: Движение мыши.

KeyPress: Нажатие клавиши на клавиатуре. KeyRelease: Отпускание клавиши на клавиатуре.

# Как обрабатываются события в Tkinter?

События обрабатываются путем привязки функций-обработчиков к определенным событиям с использованием метода bind() или через

метод after() для периодического выполнения действий.

# Как обрабатываются события мыши в Tkinter?

События мыши, такие как нажатие кнопок или движение мыши, обрабатываются путем привязки функций-обработчиков к

соответствующим событиям с использованием метода bind().

# Каким образом можно отображать графические примитивы в Tkinter?

Для отображения графических примитивов в Tkinter используется виджет Canvas. Этот виджет позволяет рисовать линии, прямоугольники, окружности и другие графические элементы.

# Перечислите основные методы для отображения графических примитивов в Tkinter.

Некоторые основные методы для работы с графическими

примитивами на холсте (Canvas): create\_line(): Создает линию.

create\_rectangle(): Создает прямоугольник. create\_oval(): Создает овал.

create\_text(): Создает текст.

# Каким образом можно обратиться к ранее созданным фигурам на холсте?

Каждая фигура на холсте имеет уникальный идентификатор, который возвращается методами создания фигур. Идентификаторы могут использоваться для обращения к ранее созданным фигурам.

# Каково назначение тэгов в Tkinter

Тэги в Tkinter используются для группировки и идентификации набора объектов на холсте. Они позволяют применять действия к определенным группам объектов. Тэги могут быть присвоены при

создании объекта на холсте или позднее с использованием метода addtag\_withtag().

Вывод: в ходе выполнения данной лабораторной работы были приобретены навыки улучшения графического интерфейса пользователя GUI с помощью обработки событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.