**📘 Методичка по компонентам tkinter**

**🔹 1. Основные компоненты**

| **Компонент** | **Назначение** |
| --- | --- |
| Tk() | Основное окно |
| Frame | Контейнер для других виджетов |
| Label | Текстовая метка |
| Button | Кнопка |
| Entry | Однострочное текстовое поле |
| Text | Многострочное текстовое поле |
| Checkbutton | Флажок (галочка) |
| Radiobutton | Переключатель |
| Listbox | Список |
| Canvas | Полотно для рисования |
| Scrollbar | Полоса прокрутки |
| Scale | Ползунок |
| Spinbox | Поле со стрелками (числовое) |
| Menu | Меню |
| Toplevel | Дополнительное окно |
| Message | Многострочная текстовая метка |
| PanedWindow | Панель с делением |
| LabelFrame | Рамка с заголовком |
| OptionMenu | Выпадающее меню |

**🔸 2. Общие свойства (опции)**

| **Свойство** | **Описание** |
| --- | --- |
| text | Текст на виджете |
| bg / background | Цвет фона |
| fg / foreground | Цвет текста |
| font | Шрифт (("Arial", 12, "bold")) |
| width, height | Размеры (в символах или пикселях) |
| padx, pady | Отступы внутри элемента |
| relief | Тип рамки (flat, raised, sunken) |
| state | Состояние (normal, disabled) |
| command | Функция-обработчик (например, кнопка) |
| variable | Привязка к переменной (для Check/Radio) |

Пример использования Объектов

import tkinter as tk

from tkinter import messagebox

# Обработчики событий

def on\_button\_click():

label.config(text="Нажата кнопка")

print("Button clicked")

def on\_check():

state = var\_check.get()

print("Checkbutton:", "Включен" if state else "Выключен")

def on\_radio():

print("Выбрано:", var\_radio.get())

def on\_entry\_change(event):

print("Entry:", entry.get())

def on\_list\_select(event):

selected = listbox.curselection()

if selected:

print("Выбрано из списка:", listbox.get(selected[0]))

def on\_scale(val):

print("Scale:", val)

def on\_spin():

print("Spinbox:", spin.get())

def on\_text\_change(event):

print("Text area:\n", text.get("1.0", tk.END))

def show\_menu\_action():

messagebox.showinfo("Меню", "Меню нажато")

# Главное окно

root = tk.Tk()

root.title("Tkinter Components")

root.geometry("600x600")

# Метка

label = tk.Label(root, text="Пример Label", bg="lightgray")

label.pack(pady=5)

# Кнопка

btn = tk.Button(root, text="Нажми меня", command=on\_button\_click)

btn.pack(pady=5)

# Поле ввода Entry

entry = tk.Entry(root)

entry.pack(pady=5)

entry.bind("<KeyRelease>", on\_entry\_change)

# Checkbutton

var\_check = tk.IntVar() # в эту переменную будет записываться true если установлена точка

check = tk.Checkbutton(root, text="Флажок", variable=var\_check, command=on\_check)

check.pack(pady=5)

# Radiobutton

var\_radio = tk.StringVar(value="A") #в зависимости от нажатия будет присвоено значение, по умолчанию будет галочка установлена в radio1

radio1 = tk.Radiobutton(root, text="Вариант A", variable=var\_radio, value="A", command=on\_radio)

# on\_radio это функция выше, то есть при нажатии выполнится функция

radio2 = tk.Radiobutton(root, text="Вариант B", variable=var\_radio, value="B", command=on\_radio)

radio1.pack()

radio2.pack()

# Listbox выпадающий список

listbox = tk.Listbox(root, height=4)

for item in ["Один", "Два", "Три"]:

listbox.insert(tk.END, item)

listbox.pack(pady=5)

listbox.bind("<<ListboxSelect>>", on\_list\_select)

# Scale (ползунок)

scale = tk.Scale(root, from\_=0, to=100, orient=tk.HORIZONTAL, command=on\_scale)

scale.pack(pady=5)

# Spinbox цифровое поле

spin = tk.Spinbox(root, from\_=0, to=10, command=on\_spin)

spin.pack(pady=5)

# Text

text = tk.Text(root, height=4, width=40)

text.pack(pady=5)

text.bind("<KeyRelease>", on\_text\_change)

# Scrollbar + Canvas (упрощённо) это полотно на котором можно рисовать

canvas\_frame = tk.Frame(root)

canvas\_frame.pack(pady=5)

canvas = tk.Canvas(canvas\_frame, width=200, height=100, bg="white")

scroll = tk.Scrollbar(canvas\_frame, orient="vertical", command=canvas.yview)

canvas.configure(yscrollcommand=scroll.set)

canvas.pack(side="left")

scroll.pack(side="right", fill="y")

# Меню

menubar = tk.Menu(root)

filemenu = tk.Menu(menubar, tearoff=0)

filemenu.add\_command(label="Открыть", command=show\_menu\_action)

filemenu.add\_separator()

filemenu.add\_command(label="Выход", command=root.quit)

menubar.add\_cascade(label="Файл", menu=filemenu)

root.config(menu=menubar)

# Показ доп. окна

def open\_toplevel():

win = tk.Toplevel(root)

win.title("Доп. окно")

tk.Label(win, text="Toplevel окно").pack()

btn\_top = tk.Button(root, text="Открыть окно", command=open\_toplevel)

btn\_top.pack(pady=5)

# LabelFrame

lf = tk.LabelFrame(root, text="Группа")

tk.Label(lf, text="Внутри LabelFrame").pack()

lf.pack(pady=5)

# OptionMenu

opt\_var = tk.StringVar(value="Выбери")

option = tk.OptionMenu(root, opt\_var, "Первый", "Второй", "Третий")

option.pack(pady=5)

# Панель с разделением

pw = tk.PanedWindow(root, orient=tk.HORIZONTAL)

left = tk.Label(pw, text="Левая панель", bg="lightblue")

right = tk.Label(pw, text="Правая панель", bg="lightgreen")

pw.add(left)

pw.add(right)

pw.pack(fill=tk.X, pady=5)

root.mainloop()

**🎨 Canvas в tkinter — полотно для рисования и сложных элементов интерфейса**

Canvas — мощный виджет, который позволяет рисовать **прямоугольники, линии, овалы, текст, изображения**, а также создавать **движущиеся объекты и интерактивные элементы**.

**📦 Создание**

#создает полотно, пустой квадрат залит белым фоном и определенным размером

canvas = tk.Canvas(parent, width=300, height=200, bg="white")

canvas.pack()

**🛠️ Основные методы рисования**

| **Метод** | **Описание** |
| --- | --- |
| create\_line() | Рисует линию |
| create\_rectangle() | Прямоугольник |
| create\_oval() | Эллипс/круг внутри прямоугольника |
| create\_text() | Вывод текста |
| create\_polygon() | Многоугольник |
| create\_arc() | Сектор окружности (дуга) |
| create\_image() | Отображение изображения (например PNG) |
| create\_window() | Встраивает другой виджет в Canvas |

## 🧱 Примеры

После создание полотна Canvas на нем можно рисовать различные геометрические фигуры

canvas.create\_line(10, 10, 200, 10, fill="blue", width=2)

canvas.create\_rectangle(50, 50, 150, 100, fill="red")

canvas.create\_oval(60, 60, 140, 140, outline="green", width=3)

canvas.create\_text(100, 180, text="Привет, Canvas!", font=("Arial", 12))

**⚙️ Полезные методы**

| **Метод** | **Описание** |
| --- | --- |
| coords(id) | Получить/установить координаты объекта |
| itemconfig(id, ...) | Изменение свойств |
| move(id, dx, dy) | Переместить |
| delete(id) | Удалить объект |
| tag\_bind(tag, ...) | Привязка событий по тегу |
| scale() | Масштабирование |
| bbox(id) | Границы объекта |
| find\_closest(x, y) | Найти ближайший объект |

**💡 Теги**

Можно задавать **теги** элементам и работать с группами:

canvas.create\_rectangle(10, 10, 100, 100, tags="box")

canvas.itemconfig("box", fill="green")

canvas.tag\_bind("box", "<Enter>", lambda e: print("Навели мышь"))

Позиционирование

**1️⃣ pack() — упрощённое автоматическое размещение**

📌 **Автоматически размещает виджеты один за другим** (сверху вниз или слева направо).

**🔧 Синтаксис:**

widget.pack(side=..., fill=..., expand=..., padx=..., pady=...)

**⚙️ Параметры:**

| **Параметр** | **Значение** |
| --- | --- |
| side | 'top' (по умолчанию), 'bottom', 'left', 'right' — куда «прижать» элемент |
| fill | 'x', 'y', 'both', None — заполнение по осям |
| expand | True/False — заполнять свободное пространство |
| padx/pady | Отступы по горизонтали и вертикали |

**✅ Пример:**

import tkinter as tk

root = tk.Tk()

tk.Label(root, text="Сверху", bg="red").pack(side="top", fill="x")

tk.Label(root, text="Слева", bg="green").pack(side="left", fill="y")

tk.Label(root, text="Справа", bg="blue").pack(side="right", fill="y")

tk.Label(root, text="Снизу", bg="yellow").pack(side="bottom", fill="x")

root.mainloop()

**2️⃣ grid() — табличная сетка**

📌 Размещает элементы **по строкам и столбцам**. Как в Excel.

**🔧 Синтаксис:**

widget.grid(row=..., column=..., rowspan=..., columnspan=..., padx=..., pady=..., sticky=...)

**⚙️ Параметры:**

| **Параметр** | **Значение** |
| --- | --- |
| row/column | номер строки и столбца |
| rowspan | сколько строк занять |
| columnspan | сколько столбцов занять |
| padx/pady | отступы |
| sticky | "n", "s", "e", "w" — «прилипание» к сторонам (направления) |

📌 sticky="nsew" — растянуть по всем направлениям.

**✅ Пример:**

import tkinter as tk

root = tk.Tk()

tk.Label(root, text="0,0", bg="lightgray").grid(row=0, column=0)

tk.Label(root, text="0,1", bg="lightblue").grid(row=0, column=1)

tk.Label(root, text="1,0", bg="lightgreen").grid(row=1, column=0)

tk.Label(root, text="1,1", bg="lightpink").grid(row=1, column=1)

root.mainloop()

💡 Можно сделать полноценную таблицу. Удобен для форм и формочек с подписями и полями.

**3️⃣ place() — абсолютное позиционирование**

📌 Полный контроль: указываешь **координаты вручную**.

**🔧 Синтаксис:**

widget.place(x=..., y=..., width=..., height=...)

**⚙️ Параметры:**

| **Параметр** | **Значение** |
| --- | --- |
| x / y | координаты относительно левого верхнего угла |
| width / height | размеры (можно задать вручную) |
| relx / rely | относительные координаты (0.0–1.0) |
| relwidth / relheight | относительные размеры |

**✅ Пример:**

import tkinter as tk

root = tk.Tk()

root.geometry("300x200")

tk.Label(root, text="Тут", bg="orange").place(x=50, y=50, width=100, height=30)

root.mainloop()

💡 Удобен для «свободного» расположения, но **нельзя адаптировать под разный размер окна**.

**🤔 Что выбрать?**

| **Менеджер** | **Когда использовать** |
| --- | --- |
| pack() | Простой интерфейс, вертикально/горизонтально |
| grid() | Формы, таблицы, структурированные блоки |
| place() | Максимальный контроль, пиксельная точность (редко) |

⚠️ Не **смешивай** pack() и grid() в **одном контейнере** — будет каша.

На картинке показан принцип позиционирования элементов различными способами,

Pack – распределяет по сторонам

Grid – использует сетку срока столбец

Place – позиционирует относительно верхнего левого угла по X,Y

