**Первая программа**

Создадим первую программу с использованием Tkinter. Для этого определим следующий скрипт:

```
from tkinter import *

root = Tk()     # создаем корневой объект - окно
root.title("Приложение на Tkinter")     # устанавливаем заголовок окна
root.geometry("300x250")    # устанавливаем размеры окна

label = Label(text="Hello world") # создаем текстовую метку
label.pack()    # размещаем метку в окне

root.mainloop()
```

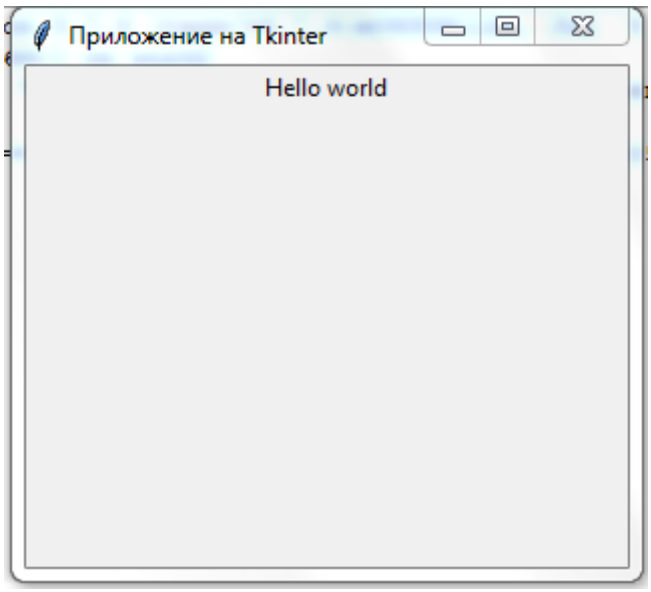
Для создания графического окна применяется конструктор **Tk()**, который определен в модуле tkinter. Создаваемое окно присваивается переменной root, и через эту переменную мы можем управлять атрибутами окна. В частности, с помощью метода **title()** можно установить заголовок окна.

С помощью метода **geometry()** - размер окна. Для установки размера в метод geometry() передается строка в формате "Ширина x Высота". Если при создании окна приложения метод geometry() не вызывается, то окно занимает то пространство, которое необходимо для размещения внутреннего содержимого.

Создав окно, мы можем разместить в нем другие графические элементы. Эти элементы еще называются виджетами. В данном случае мы размещаем в окне текстовую метку. Для это создаем объект класса **Label**, которые хранит некоторый текст. Затем для размещения элемента label в окне вызываем у него метод pack()

Для отображения окна надо вызвать у него метод **mainloop()**, который запускает цикл обработки событий окна для взаимодействия с пользователем.

Результат:




**Установка иконки**

Перед заголовком отображается иконка. По умолчанию это иконка пера. С помощью метода **iconbitmap()** можно задать любую другую иконку. Например, определим в одной папке с файлом приложения какой-нибудь файл с иконкой, допустип, он называется "favicon.ico" и используем его для установки иконки:

```
from tkinter import *

root = Tk()
root.title("Hello world")
root.iconbitmap(default="favicon.ico")
root.geometry("300x250")
root.mainloop()
```

В качестве альтернативы для установки иконки также можно было бы использовать метод **iconphoto()**

```
from tkinter import *

root = Tk()
root.geometry("250x200")

root.title("Hello world")
icon = PhotoImage(file = "icon2.png")
root.iconphoto(False, icon)
```

```
root.mainloop()
```

**Виджеты**

Ключевым строительным блоком в графическом приложении являются различные элементов управления, с которыми взаимодействует пользователь, как кнопки, метки, поля ввода. В Tkinter имеется богатая палитра различных элементов управления, которые называются **виджетами**. Основные из них:

- **Button**: кнопка
- **Label**: текстовая метка
- **Entry**: однострочное текстовое поле
- **Text**: многострочное текстовое поле
- **Checkbutton**: флажок
- **Radiobutton**: переключатель или радиокнопка
- **Frame**: фрейм, который организует виджеты в группы
- **Listbox**: список
- **Combobox**: выпадающий список
- **Menu**: элемент меню
- **Scrollbar**: полоса прокрутки
- **Treeview**: позволяет создавать древовидные и табличные элементы
- **Scale**: текстовая метка
- **Spinbox**: список значений со стрелками для перемещения по элементам
- **Progressbar**: текстовая метка
- **Canvas**: текстовая метка
- **Notebook**: панель вкладок

Tkinter предоставляет виджеты в двух вариантах: виджеты, которые располагаются непосредственно в пакете **tkinter**, и виджеты из

пакета **tkinter.ttk**. С одной стороны, оба пакета предоставляют практически одни и те же виджеты, например, виджет Button есть в обоих пакетах. Но с другой стороны, **ttk** предоставляет чуть больше функциональности по настройке виджетов, в частности, по их стилизации. И считается, что виджеты из **ttk** несколько современнее, чем стандартные, в то же время с ttk, возможно, чуть сложнее работать. Что именно использовать остается на выбор разработчика.

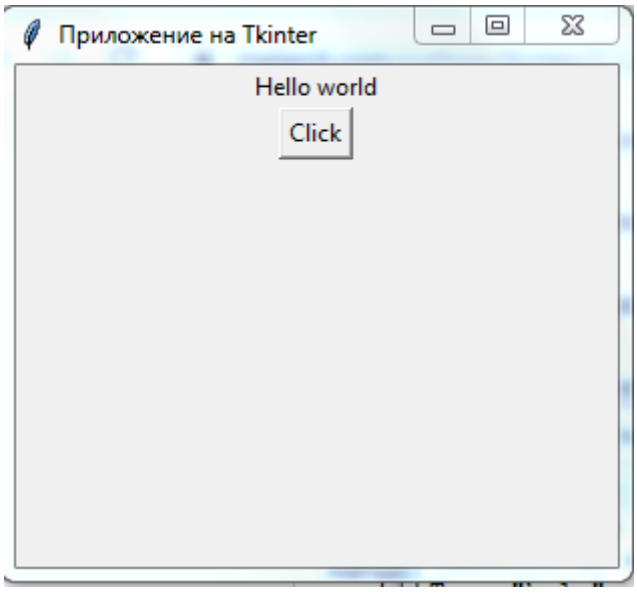
Сначала посмотрим на стандартный виджет Button из общего пакета **tkinter**:

```
from tkinter import *
root = Tk()     # создаем корневой объект - окно
root.title("Приложение на Tkinter")     # устанавливаем заголовок окна
root.geometry("300x250")    # устанавливаем размеры окна

label = Label(text="Hello world") # создаем текстовую метку
label.pack()    # размещаем метку в окне

btn = Button(text="Click") # создаем кнопку из пакета tkinter
btn.pack()    # размещаем кнопку в окне

root.mainloop()
```




Теперь посмотрим на примере кнопки из пакета **ttk**:

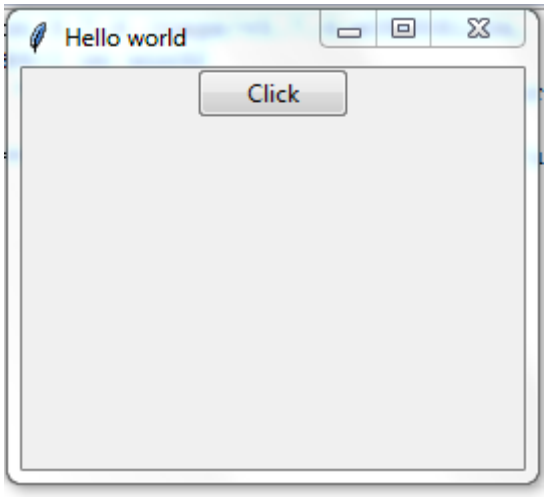
```
from tkinter import *
from tkinter import ttk     # подключаем пакет ttk

root = Tk()
```

```
root.title("Hello world")
root.geometry("250x200")

btn = ttk.Button(text="Click") # создаем кнопку из пакета ttk
btn.pack()    # размещаем кнопку в окне

root.mainloop()
```




**Параметры виджета**

Виджет обладает набором параметров, которые позволяют настроить его внешний вид и поведение. У каждого виджета свой набор параметров. Обычно параметры задаются через конструктор. Например, в примере выше у кнопки устанавливался параметр text, который задает текст на кнопке:

```
ttk.Button(text="Click") # устанавливаем параметр text
```

Но обращаться к параметрам можно и вне конструктора, используя имя переменной виджета и синтаксис словарей:

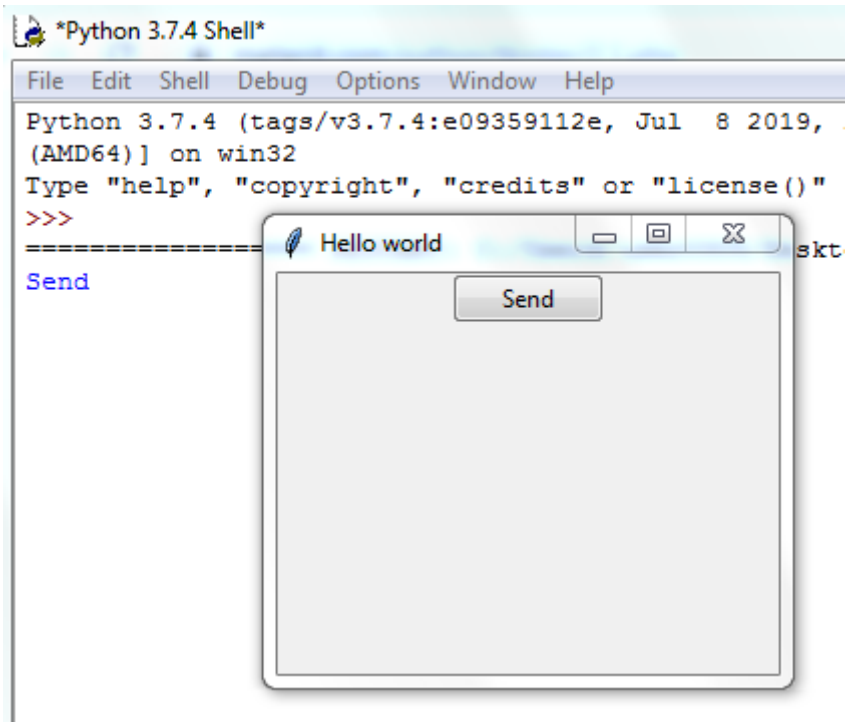
```
from tkinter import *
from tkinter import ttk

root = Tk()
root.title("Hello world")
root.geometry("250x150")

btn = ttk.Button()
btn.pack()
# устанавливаем параметр text
```

```
btn["text"]="Send"
# получаем значение параметра text
btnText = btn["text"]
print(btnText)

root.mainloop()
```




**Кнопки**

Одним из наиболее используемых компонентов в графических программах является кнопка. В tkinter кнопки представлены классом **Button**. Основные параметры виджета Button:

- **command**: функция, которая вызывается при нажатии на кнопку
- **compund**: устанавливает расположение картинки и текста относительно друг друга
- **cursor**: курсор указателя мыши при наведении на метку
- **image**: ссылка на изображение, которое отображается на метке
- **pading**: отступы от границ вилжета до его текста
- **state**: состояние кнопки
- **text**: устанавливает текст метки
- **textvariable**: устанавливает привязку к элементу StringVar
- **underline**: указывает на номер символа в тексте кнопки, который подчеркивается. По умолчанию значение -1, то есть никакой символ не подчеркивается
- **width**: ширина виджета

**Обработка нажатия на кнопку**

Для обработки нажатия на кнопку необходимо установить в конструкторе параметр command, присвоив ему ссылку на функцию, которая будет срабатывать при нажатии:

```
from tkinter import *
from tkinter import ttk

clicks = 0

def click_button():
    global clicks
    clicks += 1
    # изменяем текст на кнопке
    btn["text"] = f"Clicks {clicks}"

root = Tk()
root.title("Hello world")
root.geometry("250x150")

btn = ttk.Button(text="Click Me", command=click_button)
btn.pack()

root.mainloop()
```

Здесь в качестве обработчика нажатия устанавливается функция click_button. В этой функции изменяется глобальная переменная clicks, которая хранит число кликов. Кроме того, изменяем текст кнопки, чтобы визуально было видно сколько нажатий произведено. Таким образом, при каждом нажатии кнопки будет срабатывать функция click_button, и количество кликов будет увеличиваться:

**Отключение кнопки**

Для ttk-кнопки мы можем установить отключенное состояние с помощью метода state(), передав ему значение "disabled". С такой кнопкой пользователь не сможет взаимодействовать:

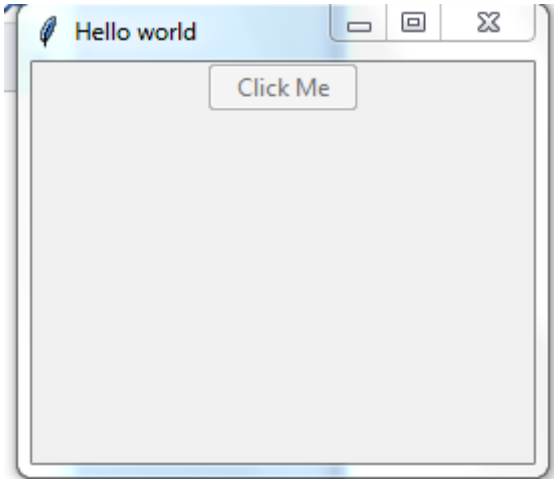
```
from tkinter import *
from tkinter import ttk     # подключаем пакет ttk

root = Tk()
root.title("Hello world")
root.geometry("250x200")

btn = ttk.Button(text="Click Me", state=["disabled"])
```

```
btn.pack()

root.mainloop()
```




**Позиционирование. Pack**

Для позиционирования виджетов в контейнере применяются различные способы. Один из них представляет вызов у виджета метода **pack()**. Этот метод принимает следующие параметры:

- **expand**: если равно True, то виджет заполняет все пространство контейнера.
- **fill**: определяет, будет ли виджет растягиваться, чтобы заполнить свободное пространство вокруг. Этот параметр может принимать следующие значения: NONE (по умолчанию, элемент не растягивается), X (элемент растягивается только по горизонтали), Y (элемент растягивается только по вертикали) и BOTH (элемент растягивается по вертикали и горизонтали).
- **anchor**: помещает виджет в определенной части контейнера. Может принимать значения n, e, s, w, ne, nw, se, sw, c, которые являются сокращениями от Noth(север - вверх), South (юг - низ), East (восток - правая сторона), West (запад - левая сторона) и Center (по центру). Например, значение nw указывает на верхний левый угол
- **side**: выравнивает виджет по одной из сторон контейнера. Может принимать значения: TOP (по умолчанию, выравнивается по верхней стороне контейнера), BOTTOM (выравнивание по нижней стороне), LEFT (выравнивание по левой стороне), RIGHT (выравнивание по правой стороне).
- **ipadx**: устанавливает отступ содержимого виджета от его границы по горизонтали.
- **ipady**: устанавливают отступ содержимого виджета от его границы по вертикали.
- **padx**: устанавливает отступ виджета от границ контейнера по горизонтали.

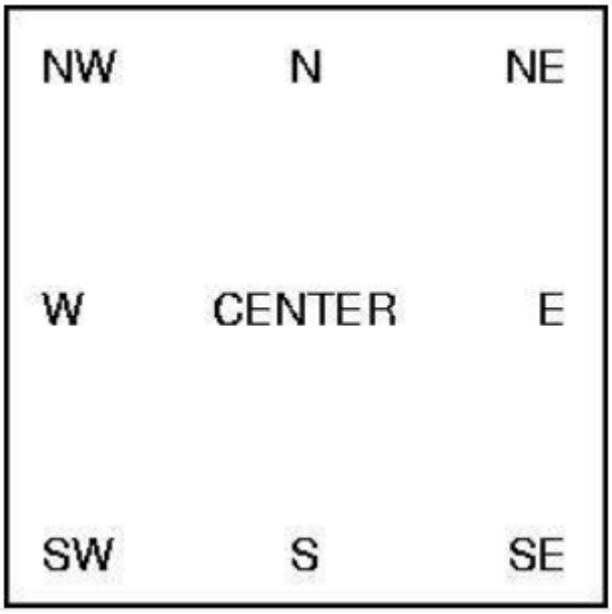
- **pady**: устанавливает отступ виджета от границ контейнера по вертикали.

**Anchor**

Параметр **anchor** помещает виджет в определенной части контейнера. Может принимать следующие значения:

- n: положение вверху по центру
- e: положение в правой части контейнера по центру
- s: положение внизу по центру
- w: положение в левой части контейнера по центру
- nw: положение в верхнем левом углу
- ne: положение в верхнем правом углу
- se: положение в нижнем правом углу
- sw: положение в нижнем левом углу
- center: положение центру

Схематически это выглядит следующим образом:




Стоит отметить, что значение в кавычках для параметра anchor передается в нижнем регистре, без кавычек - в верхнем регистре

```
btn.pack(anchor="nw")
btn.pack(anchor=NW)
```

**Заполнение контейнера**

Параметр **fill** позволяет заполнить пространство контейнер по горизонтали (значение X), по вертикали (значение Y) или по обеим сторонам (значение

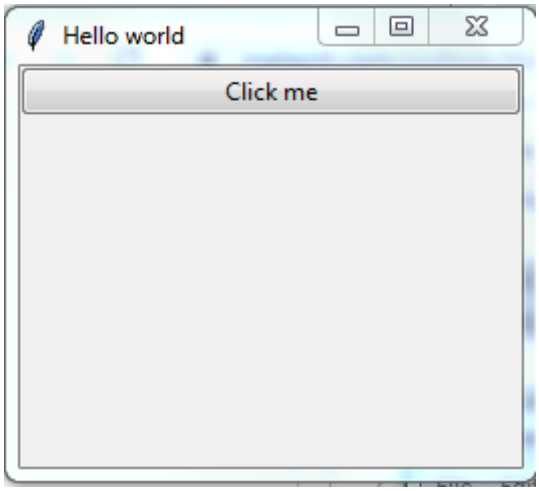
BOTH). По умолчанию значение NONE, при котором заполнение контейнера отсутствует. Например, заполним все пространство контейнера по горизонтали

```
from tkinter import *
from tkinter import ttk

root = Tk()
root.title("Hello world")
root.geometry("250x200")

btn = ttk.Button(text="Click me")
btn.pack(fill=X)

root.mainloop()
```




**Отступы**

Параметры **padx** и **pady** позволяют указать отступы виджета от границ контейнера:

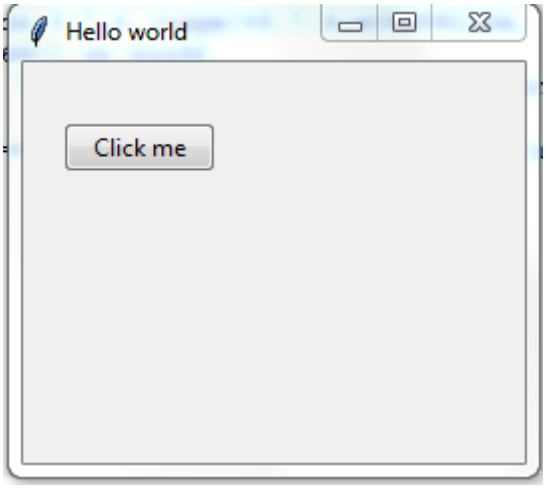
```
from tkinter import *
from tkinter import ttk

root = Tk()
root.title("Hello world")
root.geometry("250x200")

btn = ttk.Button(text="Click me")
btn.pack(anchor="nw", padx=20, pady=30)

root.mainloop()
```

Здесь кнопка смещена относительно верхнего левого угла на 20 единиц вправо и на 30 единиц вниз




Дополнительно можно ознакомиться с материалом здесь