Вставка текста с помощью метода холста *create_text*

Метод холста ***create_text*** размещает строку текста в точке с координатами *(x,y)* По умолчанию указанные координаты представляют центральную точку вывода текста.

```
объект_текст = холст.create_text(x,y, text=строка_текста)
```

Если же указать **параметр точки привязки *(anchor)***, то координаты *(x,y)* – это местоположение указанной точки привязки текста.

```
объект_текст = холст.create_text(x,y,
                                 text=строка_текста,anchor="значение")
```

Параметр *fill* задает цвет текста.

В качестве примера приведем полный текст программы:

```
from tkinter import *
tk = Tk()
tk.title('Текст')
tk.geometry('800x450+0+0')
tk.resizable(width=False, height=False)

s='Съешь ещё этих мягких французских булок, да выпей же чаю'
c = Canvas(tk, width=800, height=450)
c.pack()
c.create_text(10, 10, text=s)
c.create_text(400, 50, text=s)
c.create_text(0, 100, anchor=NW, text=s, fill='#0000ff')
c.create_text(0, 200, anchor=S, text=s, fill='#0000ff')
c.create_text(500, 300, anchor=NW, text=s, fill='#0000ff')

tk.update()
tk.mainloop()
```

Вы получите следующий результат:

Параметр *width* задает ширину для размещения текста в пикселах. Каждая строка текста будет разбита на более короткие строки, если это необходимо, или даже разбита на слова, чтобы соответствовать указанной ширине. Если не указать этот параметр, то текст будет помещен внутри прямоугольника, длина которого равна самой длинной строке. Давайте разберем, как работает этот параметр. Наберите этот текст:

```
from tkinter import *
tk = Tk()
tk.title('Текст')
tk.geometry('800x450+0+0')
tk.resizable(width=False, height=False)

s1='Съешь ещё этих мягких французских булок, да выпей же чаю'
with open('text1.txt', 'r') as f:
  s2 = ''.join(f.readlines())

c = Canvas(tk, width=800, height=450)
c.pack()

# черный текст
c.create_text(10, 0, anchor=NW, text=s1)
# синий текст
c.create_text(10, 50, anchor=NW, text=s2, fill='#0000ff')
# черный текст
c.create_text(10, 250, anchor=NW, text=s1, width=100)
# синий текст
c.create_text(250, 100, anchor=NW, text=s2, width=100, fill='#0000ff')
# красный текст
c.create_text(400, 150, anchor=NW, text=s1, width=400, fill='#ff0000')
# зеленый текст
c.create_text(400,200, anchor=NW, text=s2, width=400, fill='#00ff00')

tk.update()
tk.mainloop()
```

Проанализируйте получившийся результат:

Параметр *font* задает характеристики шрифта текста. Есть два варианта задания шрифта:

- в виде строки или кортежа (параметры, указанные через запятую, и взятые в круглые скобки)
- создав объект шрифта

Строка или кортеж в параметре *font*

1) первый элемент – название шрифта (гарнитура шрифта).
   Если название шрифта состоит из одного слова, то удобнее использовать строку, если из нескольких – кортеж
2) второй элемент – число, являющееся размером шрифта (кегль шрифта). Если число положительное, то размер задается в пунктах (1 пункт (1 пт) = 1/72 английского дюйма; 1 дюйм = 2,54 см). Если число отрицательное, то размер задается в пикселах (px)
3) третий элемент – строка, содержащая один или несколько модификаторов стиля:

   - *bold* – жирный шрифт
   - *italic* – курсивный шрифт
   - *underline* – подчеркнутый шрифт
   - *overstrike* – перечеркнутый шрифт

ВНИМАНИЕ! Не используйте подчеркивание для выделения текста. Если хотите выделить часть текста внутри предложения, то используйте курсив. Если же хотите выделить заголовок или подзаголовок, то, кроме увеличения размера шрифта, используйте жирный и курсивный шрифт (например, для заголовка – жирный, для подзаголовка – жирный курсивный).

Примеры задания значения параметра:

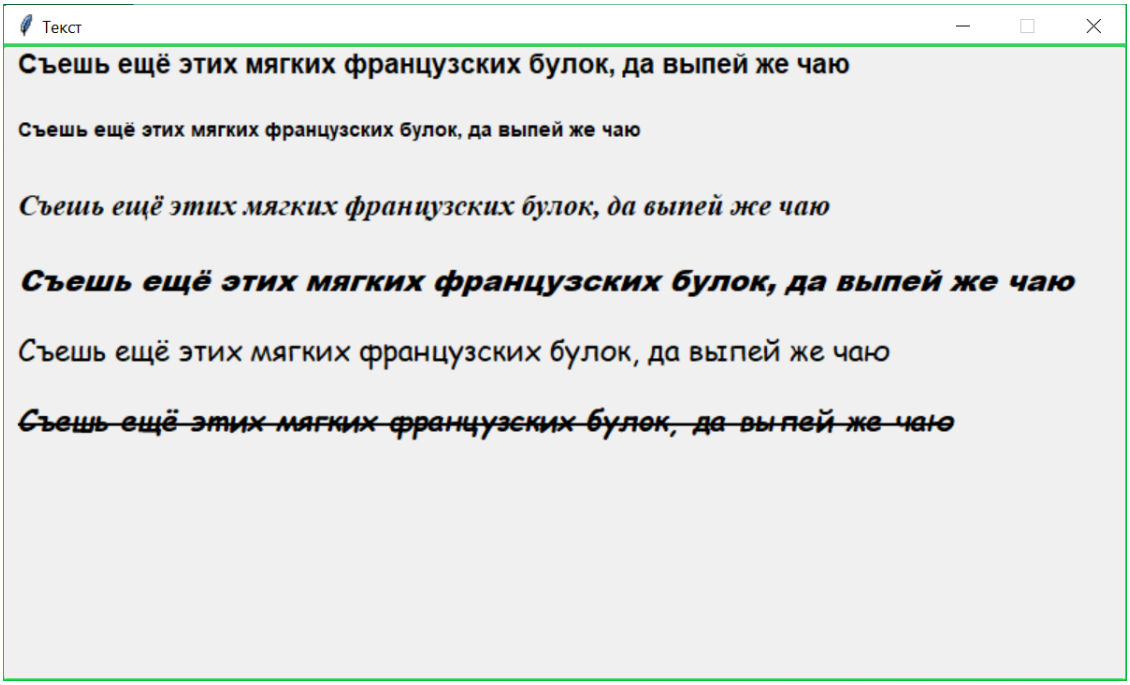
- обычный шрифт *Helvetica* 16 пт
  `'Helvetica 16'`
  `('Helvetica', '16')`
- жирный курсивный шрифт *Arial* размером 20 px
  `'Arial -20 bold italic'`
  `('Arial', -20, 'bold italic')`
- жирный курсивный шрифт *Times New Roman* 24 пт
  `('Times New Roman', '24', 'bold italic')`

Рассмотрим следующую программу:

```
from tkinter import *
tk = Tk()
tk.title('Текст')
tk.geometry('800x450+0+0')
tk.resizable(width=False, height=False)

s='Съешь ещё этих мягких французских булок, да выпей же чаю'
c = Canvas(tk, width=800, height=450)
c.pack()

c.create_text(10, 0, anchor=NW, text=s, font='Arial 14 bold')
c.create_text(10, 50, anchor=NW, text=s, font='Arial -14 bold')
c.create_text(10, 100, anchor=NW, text=s, font=('Times New Roman',
16,'bold italic'))
c.create_text(10, 150, anchor=NW, text=s, font=('Arial Black', 16 ,'bold
italic'))
```

```
c.create_text(10, 200, anchor=NW, text=s, font=('Comic Sans MS', 16))
c.create_text(10, 250, anchor=NW, text=s, font=('Comic Sans MS', 16,
'bold italic overstrike'))

tk.update()
tk.mainloop()
```

Она выдаст следующий результат:



Создание объекта шрифта для использования в параметре *font*

Чтобы не задавать каждый раз параметры шрифта как строку, можно создать объект шрифта и использовать его несколько раз. Но для создания объекта шрифта надо не забыть импортировать модуль *tkinter.font*

```
from tkinter import font
```

затем создаем объект шрифта:

```
шрифт = font.Font(параметры)
```

Параметры включают:

- ***family***
  название шрифта (гарнитура шрифта)
- ***size***
  размер шрифта (положительное число – в пунктах, отрицательное – в пикселах)
- ***weight***
  для обычного шрифта параметр может либо отсутствовать, либо *weight='normal'*
  для жирного шрифта *weight= 'bold'*
- ***slant (наклон)***
  для обычного (не курсивного) шрифта параметр может либо отсутствовать, либо *slant='roman'*
  для курсивного *slant='italic'*

- ***underline***
  для обычного (не подчеркнутого) шрифта параметр может либо отсутствовать, либо *underline=False*, либо *underline=0*
  для подчеркнутого либо *underline=True*, либо *underline=1*
- ***overstrike***
  для обычного (не зачеркнутого) шрифта параметр может либо отсутствовать, либо *overstrike=False*, либо *overstrike=0*
  для подчеркнутого либо *overstrike=True*, либо *overstrike=1*

Посмотрите, как можно задать этим способом предыдущий набор шрифтов (результат работы программы будет аналогичен предыдущему):

```
f1=font.Font(family= 'Arial', size=14, weight='bold', slant='roman',
underline=False, overstrike=False)
f2=font.Font(family= 'Arial', size=-14, weight='bold')
f3=font.Font(family= 'Times New Roman', size=16, weight='bold',
slant='italic')
f4=font.Font(family= 'Arial Black', size=16, weight='bold',
slant='italic')
f5=font.Font(family= 'Comic Sans MS', size=16)
f6=font.Font(family= 'Comic Sans MS', size=16, weight='bold',
slant='italic', overstrike=True)

c.create_text(10, 0, anchor=NW, text=s, font=f1)
c.create_text(10, 50, anchor=NW, text=s, font=f2)
c.create_text(10, 100, anchor=NW, text=s, font=f3)
c.create_text(10, 150, anchor=NW, text=s, font=f4)
c.create_text(10, 200, anchor=NW, text=s, font=f5)
c.create_text(10, 250, anchor=NW, text=s, font=f6)
```

Вставка текста с помощью виджета *Label (Метка)*

Мы использовали виджет *Label* для вставки изображения, но основное его предназначение – размещение текста.

Минимальные параметры:

```
имя = Label(контейнер, text=строка текста)
```

Для текста можно добавлять дополнительные параметры. Давайте их рассмотрим.

Параметр *font* был описан выше.

Параметр *width* задает ширину для виджета в СИМВОЛАХ

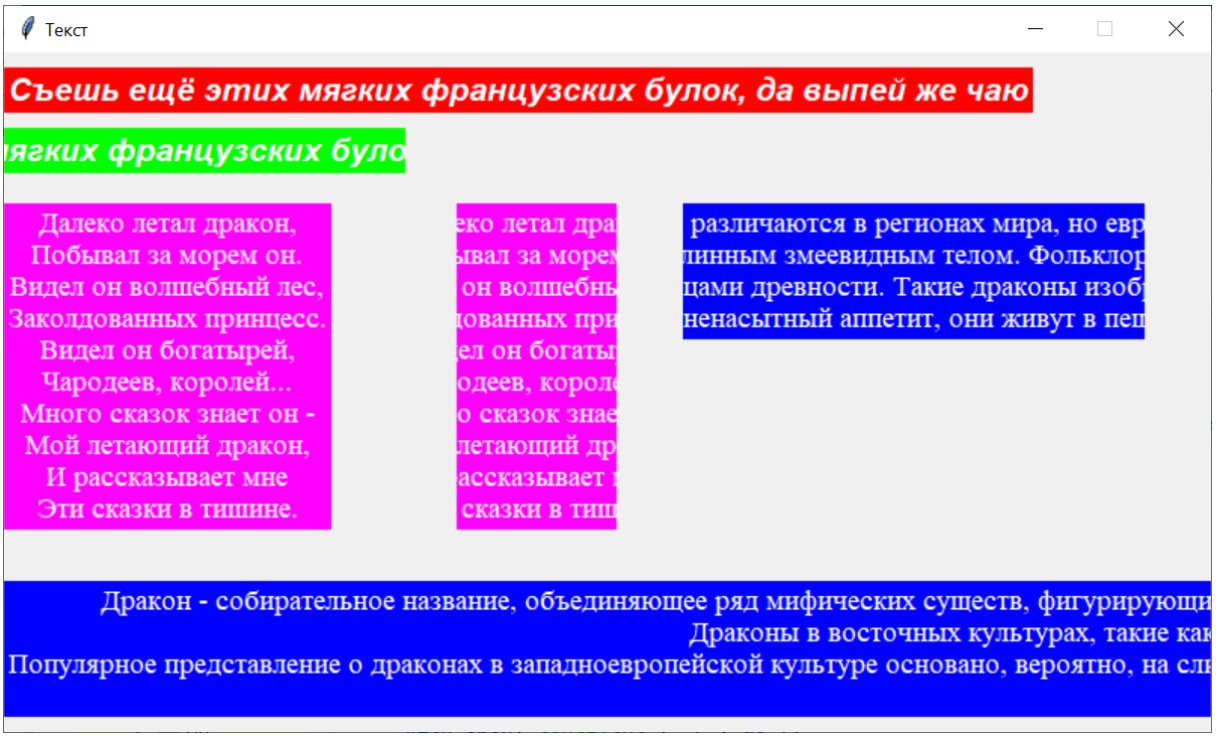
Рассмотрим работу параметра *width* на примере одной строки и двух многострочных текстов из файлов *text1.txt* и *text2.txt*.

Файл *text1.txt* содержит стихотворение из 10 коротких строк (использовано стихотворение В. Степанова https://allforchildren.ru/poetry/toys66.php).

Файл *text2.txt* содержит 4 длинных строки (текст из Википедии – https://ru.wikipedia.org/wiki/Дракон). Максимальная длина строки в этом файле – 374 символа.

Код программы:

```
from tkinter import *
from tkinter import font
tk = Tk()
tk.title("Текст")
tk.geometry("800x450+0+0")
tk.resizable(width=False, height=False)

s1="Съешь ещё этих мягких французских булок, да выпей же чаю"
with open("text1.txt", "r") as f:
  s2 = "".join(f.readlines())
with open("text2.txt", "r") as f:
  s3 = "".join(f.readlines())

f1=font.Font(family= "Arial", size=16, weight="bold",slant="italic")
f2=font.Font(family= "Times New Roman", size=14)

lbl1=Label(tk, text=s1,font=f1,fg="#ffffff",bg="#ff0000")
lbl2=Label(tk, text=s1,font=f1,fg="#ffffff",bg="#00ff00", width=20)
lbl3=Label(tk, text=s2,font=f2,fg="#ffffff",bg="#ff00ff")
lbl4=Label(tk, text=s2,font=f2,fg="#ffffff",bg="#ff00ff", width=10)
lbl5=Label(tk, text=s3,font=f2,fg="#ffffff",bg="#0000ff")
lbl6=Label(tk, text=s3,font=f2,fg="#ffffff",bg="#0000ff", width=30)

lbl1.place(x=0,y=10, anchor=NW)
lbl2.place(x=0,y=50, anchor=NW)
lbl3.place(x=0,y=100, anchor=NW)
lbl4.place(x=300,y=100, anchor=NW)
lbl5.place(x=0,y=350, anchor=NW)
lbl6.place(x=450,y=100, anchor=NW)
tk.update()
tk.mainloop()
```



Обратите внимание, что текст по умолчанию выравнивается по центру. Чтобы изменить выравнивание текстf, используется **параметр *justify***. Его значениями могут быть LEFT (по левому краю), CENTER (по умолчанию по центру), RIGHT (по правому краю).

Поменяйте в предыдущем коде эти строки и посмотрите на выравнивание левого маджентового текста и нижнего синего:

```
lbl3=Label(tk, text=s2,font=f2,fg='#ffffff',bg='#ff00ff', justify=RIGHT)
lbl5=Label(tk, text=s3,font=f2,fg='#ffffff',bg='#0000ff', justify=LEFT)
```

Исследуйте самостоятельно **параметр *height*** – он задает высоту виджета в СТРОКАХ

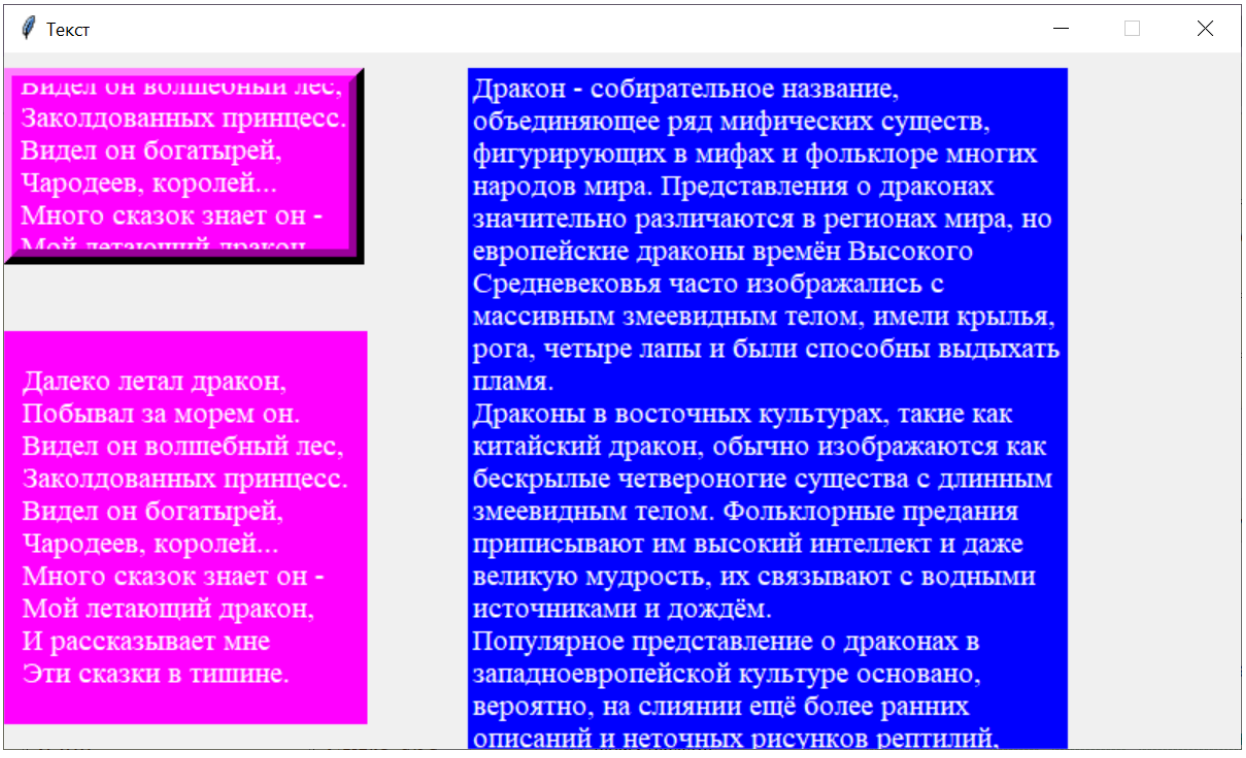
Параметры ***relief*** и ***bd*** были рассмотрены ранее при размещении изображений с помощью *Label*.

Для создания отступов текста от краев метки используются параметры ***padx*** и ***pady*** для отступов по горизонтали и вертикали соответственно.

Представляет интерес и параметр ***wraplength***. По умолчанию его задают в пикселах. При превышении заданной длины переноса строки текста будут переносится для вмещения в пространство виджета.

В предыдущем коде поменяйте часть кода на следующий:

```
lbl1 =Label(tk, text=s2,font=f2,fg='#ffffff',bg='#ff00ff', justify=LEFT,
bd=10, relief=RAISED, height=5)
lbl2 =Label(tk, text=s2,font=f2,fg='#ffffff',bg='#ff00ff', justify=LEFT,
padx=10, pady=20)
lbl3 =Label(tk, text=s3,font=f2,fg='#ffffff',bg='#0000ff', justify=LEFT,
wraplength=400)

lbl1.place(x=0,y=10, anchor=NW)
lbl2.place(x=0,y=180, anchor=NW)
lbl3.place(x=300,y=10, anchor=NW)
```



Обратите внимание на работу указанных параметров. Поэкспериментируйте с их разными значениями.

Дополнительная информация

Виджет *Label* может отображать и текст, и изображение. Для этого можно установить параметр ***compound***, который определяет положение текста по отношению к изображению с помощью одного из следующих значений:

- ***top***
  изображение выше текста
- ***bottom***
  изображение под текстом
- ***left***
  изображение слева от текста
- ***right***
  изображение справа от текста
- ***none***
  при наличии изображения отображается только изображение

Примеры показывают размещение логотипа Питона выше и правее текста. Код для первого варианта:

```
img = PhotoImage(file="python_logo.png")
f=font.Font(family= "Comic Sans MS", size=72)
lbl = Label(image=img,text="PYTHON", font=f, compound="top")
lbl.pack()
```