Віджети Button, Label, Entry

У цьому уроці розглянемо докладніше три найбільш простих і популярних віджета GUI - кнопку, мітку і однорядкове текстове поле. В `tkinter` об'єкти цих елементів інтерфейсу породжуються відповідно від класів `Button`, `Label` і `Entry`.

Властивості і методи віджетів бувають щодо загальними, характерними для багатьох типів, а також приватними, найчастіше зустрічаються тільки у якогось одного класу. У будь-якому випадку список параметрів, що властивостей великий. У цьому курсі ми будемо розглядати тільки ключові властивості і методи класів пакета `tkinter`.

У Tkinter існує три способи конфігурації властивостей віджетів:

- в момент створення об'єкту,
- за допомогою методу `config`, він же `configure`,
- шляхом звернення до властивості як до елементу словника.

Button - кнопка

Найважливішими властивостями віджета класу `Button` є `text`, за допомогою якого встановлюється напис на кнопці, і `command` для установки дії, тобто того, що буде відбуватися при натисканні на кнопку.

За замовчуванням розмір кнопки відповідає ширині і висоті тексту, однак за допомогою властивостей `width` і `height` ці параметри можна змінити. Одиницями виміру в даному випадку є знакоместа.

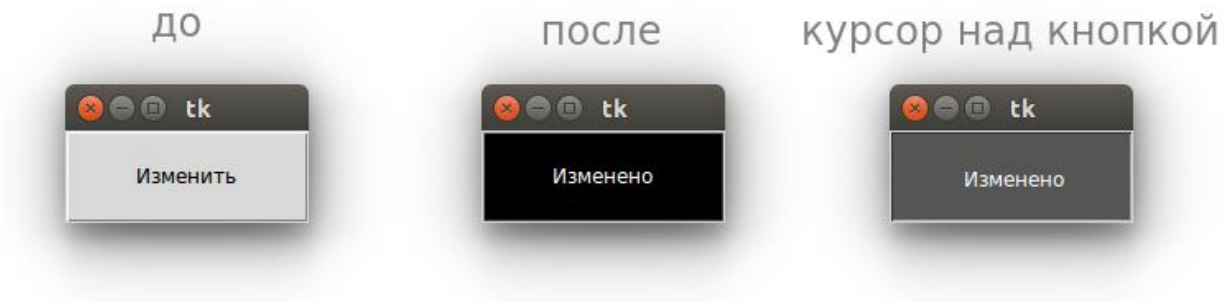
Такі властивості
як `bg`, `fg`, `activebackground` і `activeforeground` визначають відповідно колір фону і тексту, колір фону і тексту під час натискання та установки курсору миші над кнопкою.

```
from tkinter import *


def change ( ) :
    b1 [ 'text' ]  =  "Змінено"
    b1 [ 'bg' ]  =  '# 000000'
    b1 [ 'activebackground' ]  =  '# 555555'
    b1 [ 'fg' ]  =  '#ffffff'
    b1 [ 'activeforeground' ]  =  '#ffffff'
```

```
root = Tk ( )
b1 = Button ( text = "Змінити" ,
            width = 15 , height = 3 )
b1. config ( command = change )
b1. pack ( )
root. mainloop ( )
```

Тут властивість **command** встановлюється за допомогою методу **config**. Однак можна було зробити і так: `b1['command'] = change`. Ось так буде виглядати кнопка після запуску програми і після натискання на неї:



Label - мітка

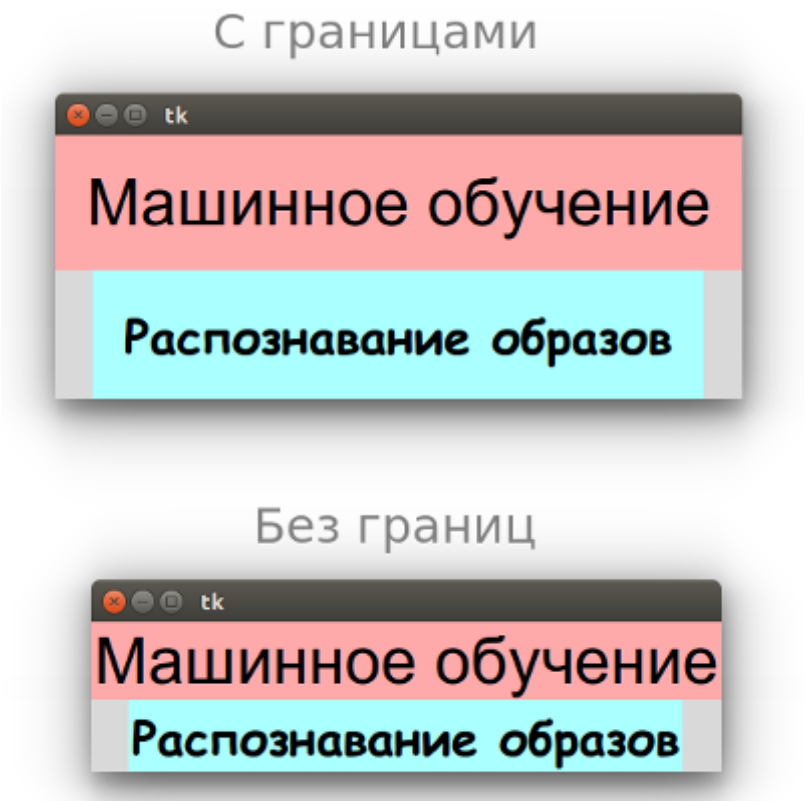
Віджет **Label** просто відображає текст у вікні і служить в основному для інформаційних цілей (висновок повідомлень, підпис інших елементів інтерфейсу). Властивості мітки багато в чому схожі з такими у кнопки. Однак у міток немає опції **command**. Тому зв'язати їх з подією можна тільки за допомогою методу **bind**.

На прикладі об'єкта типу **Label** розглянемо властивість **font** - шрифт.

```
from tkinter import *
root = Tk ( )
l1 = Label ( text = "Машинне навчання" ,
           font = "Arial 32" )
l2 = Label ( text = "Розпізнавання образів" ,
           font = ( "Comic Sans MS" ,
                 24 ,  "bold" ) )
l1. config ( bd = 20 , bg = '#ffaaaa' )
l2. config ( bd= 20 , bg = '#aaffff' )
l1. pack ( )
l2. pack ( )
root. mainloop ( )
```

Значення шрифту можна передати як рядок або як кортеж. Другий варіант зручний, якщо ім'я шрифту складається з двох і більше слів. Після назви шрифту можна вказати розмір і стиль.

Також як `font` властивість `bd` є не тільки у мітки. З його допомогою регулюється розмір кордонів (одиниця виміру - піксель):



Буває, що мітки і кнопки не привласнюють змінним, якщо потім до них в коді не доводиться звертатися. Їх створюють від класу і відразу розміщують:

```
from tkinter import *


def take ( ) :
    lab [ 'text' ]  =  "Видано"


root = Tk ( )
Label ( text = "Пункт видачі" ) . pack ( )
Button ( text = "Взяти" , command = take ) . pack ( )
lab = Label ( width = 10 , height = 1 )
lab. pack ( )
root. mainloop ( )
```

В даному прикладі тільки в однієї мітки є зв'язок зі змінною, так як одне з її властивостей може бути змінено в процесі виконання програми.

Entry - однорядкове текстове поле

Текстові поля призначені для введення інформації користувачем. Однак нерідко також для виведення, якщо передбачається, що текст з них буде скопійований. Текстові поля як елементи графічного інтерфейсу бувають однорядковими і багаторядковими. У `tkinter` другим відповідає клас `Text`, який буде розглянуто пізніше.

Властивості примірників `Entry` багато в чому схожі з двома попередніми віджетами. А ось методи - немає. З текстового поля можна взяти текст. За це дія відповідає метод `get`. У текстове поле можна вставити текст методом `insert`. Також можна видалити текст методом `delete`.

Метод `insert` приймає позицію, в яку треба вставляти текст, і сам текст.

такий код

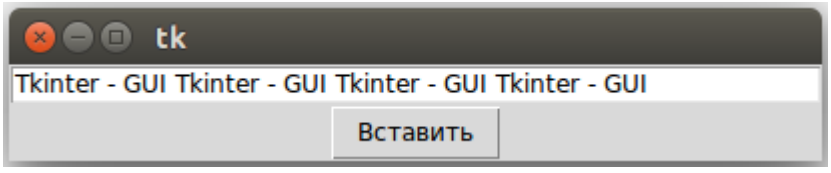
```
from tkinter import *

root = Tk ( )
e1 = Entry ( width = 50 )


def insert ( ) :
    e1. insert ( 0 ,  "Tkinter - GUI" )


but = Button ( text = "Вставити" , command = insert )
e1. pack ( )
but. pack ( )
root. mainloop ( )
```

призведе до того, що після кожного натискання на кнопку вставлятиметься нова фраза "Tkinter - GUI" перед вже існуючої в поле рядком.



Якщо 0 в `insert` замінити на константу `END`, то вставлятися буде в кінець. Можна вказати будь-яке число-індекс знакомісця, тоді вставка буде проводитися куди-небудь в середину рядка.

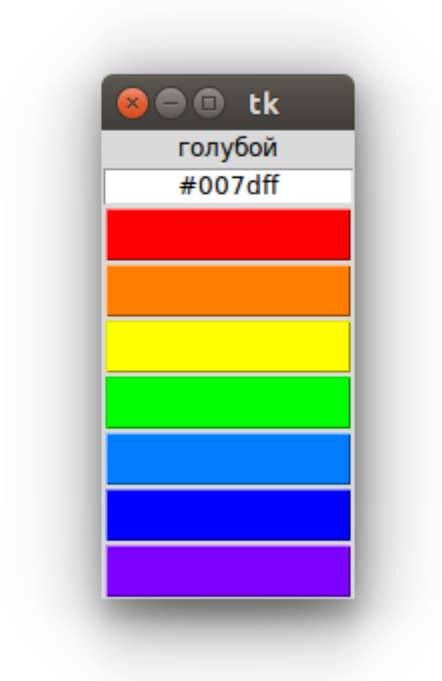
Метод `delete` приймає один або два аргументи. У першому випадку видаляється один символ у зазначеній позиції. У другому - зріз між двома зазначеними індексами, не включаючи останній. Якщо потрібно повністю очистити поле, то першим аргументом повинен бути 0, другим - `END`.

Практична робота

Напишіть програму, що складається з семи кнопок, кольори яких відповідають кольорам веселки. При натисканні на ту чи іншу кнопку в текстове поле повинен вставлятися код кольору, а в мітку - назва кольору.

Коди квітів в шістнадцятковій кодуванні: # ff0000 - червоний, # ff7d00 - помаранчевий, # ffff00 - жовтий, # 00ff00 - зелений, # 007dff - блакитний, # 0000ff - синій, # 7d00ff - фіолетовий.

Приблизно повинно вийти так:



Для вирівнювання рядка по центру в текстовому полі використовується властивість `justify` із значенням `CENTER`.