

BAB III

EXPLORASI

Bab ini berisikan hasil eksplorasi dari suatu topik telaahan yaitu mengenai Tkinter pada bahasa pemrograman python.

3.1 Pengenalan Pustaka Tkinter pada Python

Tkinter (Tk Interface) adalah suatu pustaka GUI (alat penghubung pemakai grafis) *widget* standar pembuatan Interface python untuk *Tk GUI Toolkit*. Tkinter merupakan pustaka yang secara langsung di 'bundle' di dalam python dan bekerja berdasarkan toolkit yang terdapat pada python itu sendiri. Tkinter merupakan antarmuka grafisnya dari **TCL** (Tool Command Language), yang memberikan kemudahan bagi para programmer dalam pembuatan programnya.

Tkinter merupakan pustaka grafis yang dapat memberikan kemudahan dalam pembuatan program berbasis grafis. Setiap GUI Toolkit menyediakan *widget*, yaitu objek user interface seperti *button*, *scrollbar*, *listbox*, *checkboxbutton*, *radiobutton*, *label text* dan lain sebagainya. Widget mengkapsulasi detail implementasi dan untuk setiap widget telah di definisikan perilaku *defaultya* sehingga mempermudah pemrograman GUI.

3.2 Penggunaan pustaka Tkinter pada Python

Pustaka Tkinter merupakan pustaka standar yang digunakan untuk membuat aplikasi berbasis GUI pada bahasa pemrograman python. Didalam membuat sebuah GUI biasanya terdapat berbagai objek – objek atau *widget*. Widget adalah objek yang ditampilkan dalam sebuah GUI untuk berinteraksi dengan user. Tkinter menyediakan suatu tombol widget (TkinterButton), suatu label widget (TkinterLabel), dan sebagainya. Kebanyakan widget diletakan di suatu induk widget. Toplevel widget adalah suatu widget yang khusus tanpa adanya induk. Toplevel widget diciptakan ketika memanggil Tkinter().

Untuk memeriksa pustaka Tkinter ini tersedia atau tidaknya dalam bahasa pemrograman python yang telah terpasang dalam komputer, dalam *IDLE* python tinggal mengetikan *skrip* sebagai berikut :

```
>>> from Tkinter import *
>>>
```

Bila tampilan pada python seperti di atas, berarti pustaka Tkinter tersedia pada python, dan jika tampilannya seperti di bawah ini :

```
>>> from Tkinter import *
Traceback (innermost last):
File "<stdin>", line 1, in ?
File "/usr/lib/python1.5/lib-tk/Tkinter.py",
line 8, in ?
import _tkinter # If this fails your Python may
not
be configured for Tk
ImportError: No module named _tkinter
```

Berarti pustaka Tkinter tidak tersedia di dalam python, agar Tkinter pada python tersedia maka python harus di instal ulang.

Dalam membuat suatu aplikasi berbasis GUI dengan menggunakan pustaka Tkinter, harus terlebih dahulu mengimpor pustaka Tkinter agar dapat diakses. Didalam pemrograman GUI, suatu objek di dalam sebuah tampilan merupakan bagian yang terpenting untuk memperindah tampilan, misalnya dalam sebuah tampilan terdapat label, teks, tombol, list box, combo box, dan lain – lain.

3.3 Tkinter Widget (komponen pada Tkinter)

Tkinter telah menyediakan sekitar 14 widget dasar, berikut ini tabel dari kelas widget pada Tkinter :

Table 3.1 Tkinter widget (komponen terkecil)

Widgets	Deskripsi
Button	Tombol sederhana, digunakan untuk mengeksekusi suatu perintah atau operasi lainnya
Canvas	Grafis terstruktur, widget dapat digunakan untuk menggambar grafik dan plot, membuat editor grafik, dan untuk mengimplementasikan perubahan widget
Checkbutton	Mempresentasikan sebuah variable yang dapat di pilih lebih dari dua nilai
Entry	Field untuk memasukan teks
Frame	Widget untuk wadah atau kotak, frame dapat diberi border dan background serta dapat digunakan untuk mengelompokan widget lainnya ketika membuat aplikasi atau layout dialog
Label	Menampilkan teks atau gambar
Listbox	Menampilkan sebuah daftar pilihan, listbox dapat dikonfigurasi untuk mendapatkan radiobutton atau checklist

Widgets	Deskripsi
Menubutton	Tombol menu, digunakan untuk mengimplementasikan pulldown menu
Message	Menampilkan sebuah teks, sama seperti widget label, tetapi teks dapat diatur tata letaknya secara otomatis
Radiobutton	Mempresentasikan suatu nilai dari variabel yang dapat memiliki satu atau banyak nilai, klik tombol tersebut artinya mengumpulkan nilai untuk variabel dan
Radiobutton (lanjutan)	menghapus semua radiobutton mengelompokan dengan variabel yang sama.
Scale	Menyarankan anda untuk mengeset nilai numerik dengan melakukan dragging pada panel
Scrollbar	Digunakan untuk menggulung canvas, entry, listbox, dan widget teks
Text	Memformat tampilan teks, menyarankan anda untuk menampilkan dan mengedit teks dengan gaya dan atribut, juga mendukung pemasangan image dan window
Toplevel	Sebuah widget wadah untuk menampilkan secara terpisah atas tingkatan window

3.4 Struktur pemrograman Tkinter

Dalam pembuatan aplikasi berbasis GUI dengan menggunakan Tkinter, harus mengikuti aturan – aturan dari Tkinter tersebut, termasuk pada struktur aplikasi Tkinter. Sebagai contoh untuk menjelaskan struktur dari aplikasi Tkinter, dalam sebuah window utama terdapat komponen – komponen (widgets) yang telah terpasang. Objek – objek tersebut di ciptakan oleh sebuah kelas Tk didalam pustaka Tkinter. Dengan perintah :

```
>>> from Tkinter import *  
>>> root = Tk()
```

Penulis membuat struktur pemrograman Tkinter sebagai berikut :

```
[import pustaka Tkinter]

#mengisikan pustaka Tk ke sebuah variabel
[variabel_tk] = [pustakaTk]

#pembuatan widget pada Tkinter
[variabel_widget] = [widget(vaiabel_tk)]
[option widget]
[variabel_widget].[geometry manager]

[start_event_loop]
```

Sebagai contoh struktur aplikasi dari Tkinter sebagai berikut :

File : hello.py

```
from Tkinter import *
root = Tk()

myframe = Frame(root)
myframe.master.title("Hello")
myframe.pack()

mylabel = Label(root, text = "Hello, Word")
mylabel.pack()

root.mainloop()
```

Tampilan window yang dihasilkan :



Penjelasan skrip diatas:

Baris pertama merupakan skrip untuk memanggil pustaka Tkinter.

```
from Tkinter import *
```

Baris kedua merupakan pendeklarasian widget tk yang disimpan pada variabel root

```
root = Tk()
```

baris berikutnya skrip untuk memanggil widget frame, frame akan membingkai window dengan mengatur border dan warna, dan membuat judul pada bar dengan memanggil kontrol yang tersedia pada frame.

```
myframe = Frame(root)
myframe.master.title("Hello")
myframe.pack()
```

Berikutnya memanggil widget label sebagai anak(*child*) dari window utama :

```
mylabel = Label(root, text = "Menampilkan Label
                    dengan\n"
                    "method pack()")
```

Ketika semua widget, termasuk window utama muncul di layar, maka akan terjadi pengulangan tanpa batas, pengulangan ini menunggu sebuah aksi dan memprosesnya, lalu akan kembali pada kondisi awal dan akan menunggu aksi berikutnya. Di dalam pustaka Tkinter untuk melakukan pengulangan seperti itu di perlukan suatu perintah yaitu :

```
root.mainloop()
```

Dari contoh struktur aplikasi Tkinter di atas pola yang saya dapat adalah sebagai berikut:

1. Mengimpor terlebih dahulu pustaka Tkinter

```
[from Tkinter import *]
```
2. Buat objek top level window, yang berisikan objek GUI pada sebuah aplikasi dalam contoh diatas adalah sebuah frame.
3. Buat seluruh widget GUI Tkinter diatas objek top level windows.
4. Masukkan event loop untuk mengulang semua kejadian

```
[mainloop]
```

3.5 Geometry Manager

Geometry manager diperlukan untuk menata letak dan penampilan widget di atas window. Ada macam – macam geometry manager, seperti *place()*, *grid()* dan *pack()*, namun yang paling banyak digunakan adalah *pack()*, karena cocok untuk kebutuhan sebagian besar aplikasi.

Untuk lebih jelasnya akan dijelaskan pada table Geometry manager dibawah ini :

Tabel 3.2 Geometry Manager

Manager	Diskripsi
Grid	Geometry manager grid digunakan untuk membantu membuat layout tabel dengan mengatur garis 2 dimensi, untuk menggunakan geometry manager ini menggunakan method : <i>grid()</i>
Pack	Geometry manager pack digunakan untuk membantu membuat layout dengan mengatur tata letak widget pada widget utama. Pemanggilannya dengan method : <i>pack()</i>

Place	Geometry manager place digunakan untuk mengetahui tempat secara eksplisit untuk menempatkan widget pada posisi yang tepat. Pemanggilannya dengan method : <i>place()</i>
-------	--

Dibawah ini penulis memberkan contoh – contoh penggunaan geometry manager pada Tkinter :

1. Contoh untuk geometry manager grid():

File : grid.py
<pre> from Tkinter import * root = Tk() Label(root, text="Nama").grid(row=0) Label(root, text="Alamat").grid(row=1) Entry(root, width=16).grid(row=0, column=1) Entry(root, width=16).grid(row=1, column=1) root.mainloop() </pre>
Tampilan window yang dihasilkan :


2. Contoh penggunaan geometry manager pack():

File : pack.py
<pre> from Tkinter import * root = Tk() Button(root, text="Left").pack(side=LEFT, expand=YES, fill=Y) Button(root, text="Top").pack(side=TOP, expand=YES, fill=BOTH) Button(root, text="Right").pack(side=RIGHT, expand=YES, fill=X) root.mainloop() </pre>

Tampilan window yang dihasilkan :



3. Contoh untuk geometry manager place():

File : place.py

```
from Tkinter import *  
root = Tk()  
Button(root, text="Place").place(x=10, y=10,  
    height=35)  
root.mainloop()
```

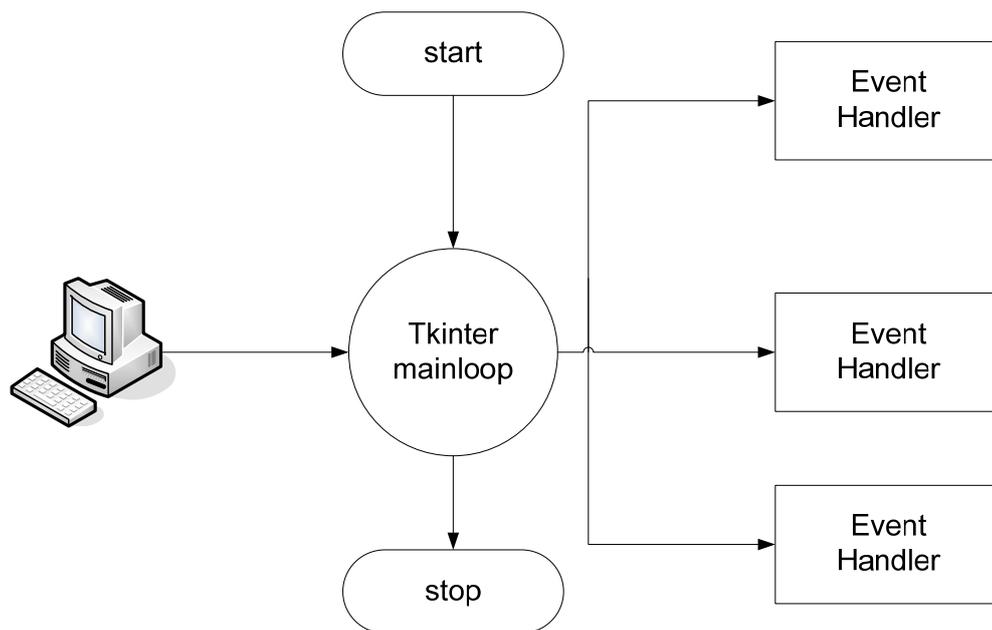
Tampilan window yang dihasilkan :



3.5.1 Event Bindings pada Tkinter

Aplikasi Tkinter banyak menyimpan event – event pada saat *mainloop* di panggil. Event bisa berulang berasal dari berbagai input, sebagai contoh event dari keyboard atau operasi mouse oleh user. Tkinter menyediakan mekanisme yang powerful untuk menangani event.

Salah satu aksi pada pemrograman GUI adalah *Event driven*, event driven adalah aksi menekan sebuah tombol, menggerakkan mouse, menekan tombol pada keyboard dan lain – lain, semua ini dikenal pada suatu sistem saat suatu kejadian dari awal hingga akhir suatu aplikasi berbasis GUI.



Gambar 3.1 event driven pemrograman GUI

3.6 Opsi Pada Tkinter Widget

Widget mempunyai opsi (atau atribut-atribut) untuk mengatur tampilan dan perilakunya. Beberapa opsi digunakan oleh banyak widgets. Sebagai contoh, kebanyakan widget mempunyai suatu opsi *background* yaitu menetapkan warna latar belakang widget yang normal. Opsi lain dikhususkan untuk jenis widget. Sebagai contoh, suatu tombol widget mempunyai suatu opsi *command*, yaitu suatu fungsi untuk memanggil (tanpa argumentasi-argumentasi) ketika tombol itu di-klik.

Tkinter Widget mempunyai masing – masing opsi pada masing – masing komponennya, dibawah ini macam – macam opsi yang dikelompokkan menjadi beberapa macam, diantaranya :

3.7.1 Opsi Packer (Paket)

Opsi ini merupakan opsi untuk menentukan posisi dari Tkinter widgets. Opsi packer pada Tkinter widgets diantaranya sebagai berikut :

a) side

Suatu opsi sisi untuk menempatkan widget di sisi yang ditetapkan di window. Nilainya adalah *left*, *right*, *top*, dan *bottom*. Nilai defaultnya adalah *top*. Jika terdapat dua widgets yang ditampilkan dalam satu window yang sama maka widget yang di panggil pertama disimpan / diletakan ditepi terdekat window yaitu sebelah kiri, selanjutnya untuk widget yang kedua akan disimpan di sebelah kanan widget pertama.

b) fill, expand

Opsi ini digunakan untuk mengatur suatu widget agar widget tersebut mengisi ruang sebuah window.

c) anchor

Opsi ini digunakan untuk menentukan posisi dimana letak dari objek disimpan, nilai dari opsi anchor ini adalah arah dari kompas (N, NW, W, SW, S, SE, E, MODA) dan center

d) padx,pady

Opsi ini memberi nilai pada posisi vertical dan horizontal, yaitu jarak pada sumbu X dan jarak pada sumbu Y.

(untuk lebih detail dari opsi ini terdapat pada lampiran)

3.7.2 Opsi Grid

Opsi ini merupakan opsi yang digunakan untuk mengatur tata letak objek – objek Tkinter yang akan disimpan pada suatu window. Opsi Grid pada Tkinter widget diantaranya sebagai berikut :

a) row, column

Opsi ini digunakan untuk menyimpan posisi widget pada kolom dan baris, nilai di isi mulai dari nol (0). Untuk menggunakan opsi ini harus menggunakan method Geometry manager yaitu method *grid()*.

b) `columnspan`, `rowspan`

- Opsi *columnspan* ini digunakan untuk melakukan *merge* pada kolom, misalnya kolom ke-1 pada baris ke1 di *merge* dengan kolom ke-2 pada baris ke1, maka kolom 1 dan 2 akan menyatu menjadi satu kolom.
- Opsi *rowspan* ini digunakan untuk melakukan *merge* pada baris, misalnya baris ke-1 di *merge* dengan baris ke-2, maka baris 1 dan 2 akan menyatu menjadi satu baris.

(untuk lebih detail dari opsi ini terdapat pada lampiran)

3.7.3 Opsi Common (Umum)

Opsi *common* ini biasanya ada pada setiap widget Tkinter, diantaranya :

a) Opsi warna

- `background (bg)`,
digunakan untuk memberikan warna pada latar sebuah widget Tkinter.
- `foreground (fg)`,
biasanya digunakan pada widget `text` pada sebuah widget Tkinter.

- `activebackground`,
digunakan untuk memberikan warna pada latar sebuah widget Tkinter dimana widget tersebut dalam keadaan aktif.
- `Activeforeground`,
digunakan untuk memberikan warna pada latar sebuah widget Tkinter dimana widget tersebut dalam keadaan aktif.
- `disabledforeground`,
digunakan untuk menonktikan warna pada latar sebuah widget Tkinter dimana widget tersebut dalam keadaan tidak aktif.
- `selectbackground`,
opsi biasanya digunakan oleh widget `entry`, `listbox`, `text`, `canvas` untuk melakukan pemilihan warna

untuk opsi warna Tkinter menyediakan dua cara untuk melakukan pengisian pada warna, yaitu :

- Berdasarkan nama dari sebuah warna
Misalnya : “red”, “blue”, “black”, dan sebagainya.
- Berdasarkan *colour string*
Misalnya :”RGB”, “RGBGGGBBG”

b) Opsi size

- `width`

Opsi ini untuk menentukan panjang

- height

Opsi ini untuk menentukan lebar

- pady, padx

opsi ini digunakan untuk menentukan panjang dari kordonat x (horizontal) dan y(vertical)

c) Appearance

- text

Opsi ini untuk memberikan teks pada suatu widget

- image

Opsi ini untuk memberikan sebuah gambar pada suatu widget

- relief

Opsi ini untuk memberikan border 3D pada suatu widget , nilainya FLAT, GROOVE, RAISED, RIDGED, SOLID, dan SUNKEN

- borderwidth

Opsi ini untuk memberikan ukuran border pada suatu widget

- font

Opsi ini untuk menentukan jenis tulisan yang akan digunakan

d) Behavior

- Command,

Opsi ini digunakan untuk memanggil fungsi

- State

Opsi ini digunakan untuk menentukan pada suatu widget apakah widget tersebut aktif atau tidak, nilai dari opsi state ini adalah ACTIF dan DISABLED

- Underline

Opsi ini digunakan untuk memberikan garis bawah yang digunakan sebagai Shortcut untuk pemanggilannya

- takefocus

Opsi ini digunakan pada untuk memberikan focus utamam pada widget

(untuk lebih detail dari opsi ini terdapat pada lampiran)

Catatan : untuk opsi lengkap perwidget bisa dilihat pada lampiran opsi widget

3.8 Perintah pemanggilan pustaka Tkinter widget

Dalam kasus ini saat memanggil Tkinter widget penulis mengkondisikan Tkinter disimpan pada variabel **root**. Dibawah ini akan diuraikan satu persatu widget yang dimiliki oleh Tkinter dan cara pemanggilannya :

1. Buton

Button widget adalah widget standar Tkinter yang digunakan untuk menerapkan berbagai macam dari tombol-tombol. Tombol dapat diisi dengan teks atau gambar, dan anda dapat menghubungkan suatu fungsi atau metoda dengan masing-

masing tombol. Ketika tombol itu ditekan, Tkinter secara otomatis memanggil fungsi atau metoda pada python.

- Skrip pemanggilan untuk *button*:

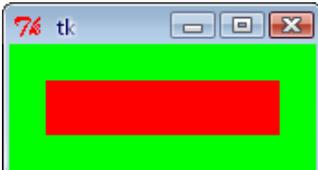
Skrip yang digunakan untuk pemanggilan dari widget *button* ini sebagai berikut :

Button (Tk(), [properti])
Contoh : <pre>>>> myTombol = Button(root, text="MyTombol", fg="white", bg="red", bd=8) >>> myTombol.pack()</pre>
Tampilan window yang dihasilkan :


2. Canvas

Widget canvas menyediakan fasilitas-fasilitas grafik tersusun untuk Tkinter. Canvas merupakan suatu widget yang sangat serbaguna yang digunakan untuk membuat grafik-grafik.

- skrip pemanggilan untuk canvas sebagai berikut :

Canvas(Tk(), [properti])
Contoh : <pre>>>> MyCanvas=Canvas(root,bg="red", bd=5, highlightbackground="green", highlightthickness=20) >>> MyCanvas.pack()</pre>
Tampilan window yang dihasilkan :


3. Checkbutton

Widget `checkbutton` digunakan untuk memilih suatu opsi dalam suatu kasus

- skrip pemanggilan untuk `checkbutton` sebagai berikut :

CheckButton(Tk(), properti)
Contoh : <pre>>>> MyCB = Checkbutton (root, text = "My CheckButton", bg="red", fg="WHITE", bd=3) >>> MyCB.pack()</pre>
Tampilan window yang dihasilkan :


4. Entry

Widget `entry` ini digunakan untuk memasukan teks, dan jika digabungkan dengan `geometry manager grid` akan menjadi sebuah form yang rapi.

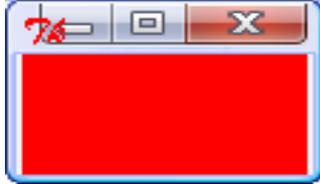
- skrip pemanggilan untuk `entry` sebagai berikut:

Entry(Tk(), [properti])
Contoh : <pre>>>> MyEntry = Entry(root, bg="red", fg="WHITE") >>> MyEntry.pack()</pre>
Tampilan window yang dihasilkan :


5. Frame

Widget frame digunakan untuk dasar atau 'wadah' untuk meletakkan widget Tkinter lainnya

skrip pemanggilan untuk frame sebagai berikut:

Frame(Tk(), [properti])
Contoh : >>> MyFrame = Frame(root, bg="red", bd=10, height = 200, width=200) >>> MyFrame.pack()
Tampilan window yang dihasilkan :


6. Label

Widget label ini digunakan untuk menampilkan teks

- skrip pemanggilan :

Label(Tk(), properti)
Contoh : >>> MyLabel = Label(root, text = "This is Label", bg="RED", fg="WHITE") >>> MyLabel.pack()
Tampilan window yang dihasilkan :


7. Listbox

Widget ini digunakan untuk menampilkan data

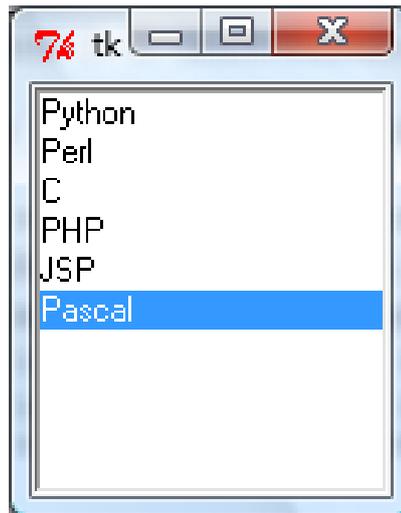
- skrip pemanggilan :

```
[variable] = Widget(Tk(), properti)
```

Contoh :

```
>>> MyListBox = Listbox(root)
MyListBox.insert(1, "Python")
MyListBox.insert(2, "Perl")
MyListBox.insert(3, "C")
MyListBox.insert(4, "PHP")
MyListBox.insert(5, "JSP")
MyListBox.insert(6, "Pascal")
>>> MyListBox.pack()
```

Tampilan window yang dihasilkan :



8. Menu

Widget menu ini digunakan untuk membuat menu pulldown

- skrip pemanggilan :

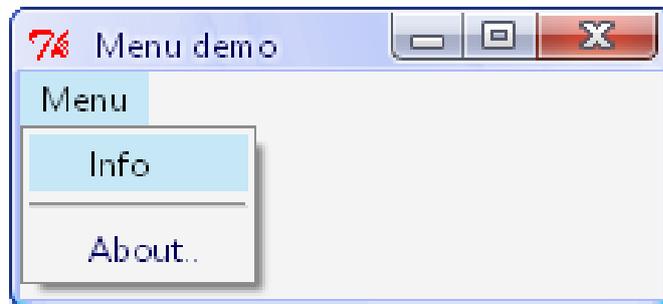
```
[variable] = Widget(Tk(), properti)
```

Contoh :

```
>>> masterMenu = Menu(root)
>>> menuinfo = Menu(masterMenu, tearoff=0)
>>> menuinfo.add_command(label="Info")
>>> menuinfo.add_separator()
```

```
>>> menuinfo.add_command(label="About..")
>>> masterMenu.add_cascade(label="Menu",
                             menu=menuinfo)
```

Tampilan window yang dihasilkan :



9. Message

Message ini digunakan untuk menyampaikan pesan singkat, yaitu dengan menggunakan *MessageBox*. Namun sebelumnya untuk memanggil module ini, harus melakukan pemanggilan terhadap module ini dengan cara memanggil *tkMessageBo*

- skrip pemanggilan :

Terdapat beberapa pesan yang dapat digubakan seperti info, *error*, *warning*, *yes/no*, dan *ok/cancel*. Untuk dapat memanggil pesan tersebut digunakan fungsi *show[nama_pesan]*. Fungsi yang dipakai untuk menampilkan pesan tersebut diantaranya :

showwarning, *showerror*, *askquestion*, *askokcancel*,
askyesno, *askretryignore*

```
Widget.showinfo("Caption untuk pesan",' Isi dari pesan)
```

Contoh :

```
>>> tkMessageBox.showinfo("Demo Messagebox","Ini
                             Menggunakan messagebox")
```

Tampilan window yang dihasilkan :

- Menggunakan fungsi showinfo



Skrip :

```
>>> tkMessageBox.showinfo("Demo Messagebox","Ini Menggunakan messagebox")
```

- Menggunakan fungsi showwarning



Skrip :

```
>>> tkMessageBox.showwarning("Demo Messagebox","Ini Menggunakan messagebox")
```

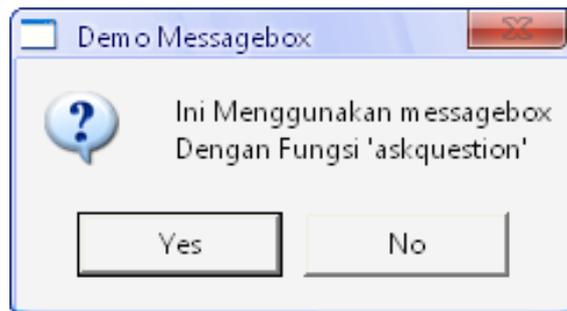
- Menggunakan fungsi showerror



Skrip :

```
>>> tkMessageBox.showwarning ("Demo Messagebox","Ini Menggunakan messagebox")
```

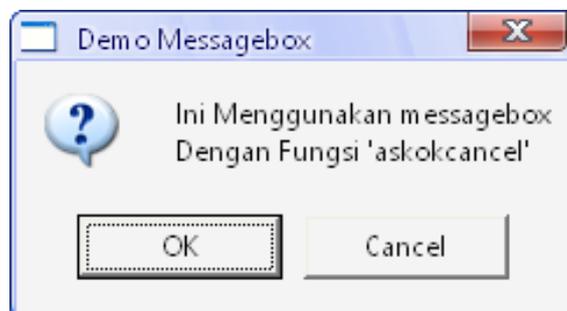
- Menggunakan fungsi askquestion

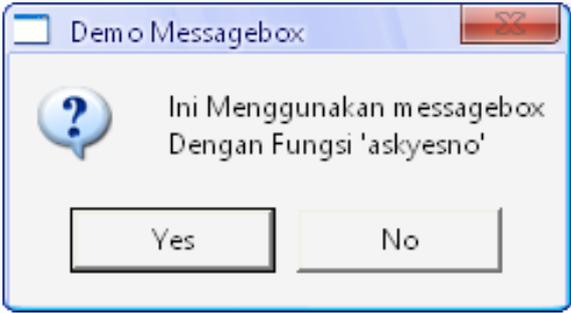


Skrip :

```
>>> tkMessageBox.askquestion ("Demo Messagebox","Ini Menggunakan messagebox")
```

- Menggunakan fungsi askokcancel



- Menggunakan fungsi askokcancel
Skrip :
>>> tkMessageBox. askokcancel ("Demo Messagebox","Ini Menggunakan messagebox")
- Menggunakan fungsi askyesno

Skrip :
>>> tkMessageBox. askyesno ("Demo Messagebox","Ini Menggunakan messagebox")

10. Radiobutton

Widget radiobutton digunakan untuk memilih suatu opsi pada suatu kasus, hanya salah satu pilihan saja yang dapat dipilih

- skrip pemanggilan :

[variable] = Widget(Tk(), properti)
Contoh :
>>> intTemp=IntVar
>>> a=Radiobutton(text="Tombol 1",variable=intTemp,value=1)
>>> b=Radiobutton(text="Tombol 2",variable=intTemp,value=2)
>>> intSelect=intTemp()
>>> a.configure(variable=intSelet)
>>> b.configure(variable=intSelect)
>>> intSelect.get()
>>> a.pack()
>>> b.pack()

Tampilan window yang dihasilkan :



11. Scale

- skrip pemanggilan :

`[variable] = Widget(Tk(), properti)`

Contoh :

```
>>> a=Scale(orient=HORIZONTAL,
            length=100,from_=0,to=10,tickinterval=5
            0)
>>> a.get()
>>> a.pack()
```

Tampilan window yang dihasilkan :



12. Scrollbar

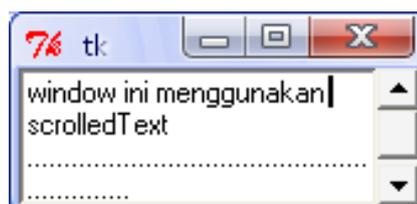
- skrip pemanggilan :

`[variable] = Widget(Tk(), properti)`

Contoh :

```
>>> from ScrolledText import ScrolledText
>>> ScrolledText().pack()
```

Tampilan window yang dihasilkan :



13. Text

- skrip pemanggilan :

```
[variable] = Widget(Tk(), properti)
```

Contoh :

```
>>> from ScrolledText import ScrolledText
>>> text=Text(root)
>>> text.insert(END, "ini adalah text, ")
>>> text.pack()
```

Tampilan window yang dihasilkan :

