Bölüm 19

Menu

19.1 Menu Nedir?

`Menü (menu)`, seçeneklerden oluşan bir listedir. Sosyal yaşamın her kesiminde uygulaması vardır. Örneğin, lokantada yemekler için seçim yapmaya yarayan liste bir menüdür. Bilgisayarda menü, ekranda seçenekleri sunan ve kullanıcının yazılımı yönetmesine olanak sağlayan bir (görsel) arayüzdür. Bir uygulama yazılımı için iyi düzenlenmiş menü, kullanıcıya büyük kolaylıklar sağlar. Örneğin, *MS Word*, *Excel* gibi yazılımların görsel menüleri, hiç programlama bilmeyenlerin o yazılımları kolayca kullanmalarını sağlar. Menu ile yönetilen yazılımların karşıtı *komut-yönetimli* (command-driven) yazılımlardır. Bu sistemde, istenilen işi yaptıran komut yazılır. İlgili programı bilenler için, komut-yönetimli yazılımları kullanmanın üstünlükleri vardır. Ayrıca, bazı yazılımlar için eksiksiz bir menu hazırlamak olanaksız ya da çok zordur. Örneğin, unix işletim sistemlerinde, çok sayıda komut ve her komutun onlarca seçeneği (option) vardır. Bütün seçenekleri içine alan bir menu hazırlamak hemen hemen olanaksızdır.

19.2 Menu Türleri

Menü türlerinin eksiksiz bir listesini vermek zordur. Üstelik, literatürde aynı işleve sahip menülere farklı adlar verilmektedir. Aşağıda, bilgisayarda yaygın kullanılan bazı menü türleri listelenmiştir.

19.2.1 Menu bar

Menu Çubuğu

Menü Çubuğu (menü şeridi, menu bar), genellikle menu penceresinin üstünde, ana seçenekleri yatay doğrultuda sunan dikdörtgenel bir çerçevedir. Menü şeridinde yer alan seçenekler birer düğme gibi görünebilir. Menu çubuğundaki seçeneklere üst-düzey (top-level) menüler denilir. Her birisine tıklandığında, varsa alt-menüleri açılır.

19.2.2 pop-up menu

Pop-up terimi, bilgisayarın belleğinde hazır duran ve *kısa-yol (hot-key)* tuşuna basılınca ekranda açılan pencerelere verilen addır. Ayrıca, bir web sayfası ziyaret edildiğinde, kullanıcının istemeden açılan ve reklam içerikli açılır pencereler de bu grubun içinde sayılır. Bilgisayarda *pop-up* menüsü, çoğunlukla, *drop-down* menüsüne eşanlamda kullanılır.

19.2.3 Drop-down menu

Pull-down menu ile aynıdır. Bir seçeneğe tıklandığında ona bağlı alt menüler, hemen altında beliren yeni bir pencerede açılır.

19.2.4 pull-down menu

Drop-down menu ile aynıdır.

19.2.5 tear-off menu

Sıyrılan menü (tear-offmenu), üzerine tıklandığında sıyrılıp yana açılan ya da ekranda istenilen yere taşınabilen pop-up menüdür.

19.3 tkinter'de Menu

Menu, *tkinter* içinde bir sınıftır, bir widget'tir. Adı üzerinde, gereksemeye göre, görünüşleri ve işlevleri farklı olan menüler yaratabilir.

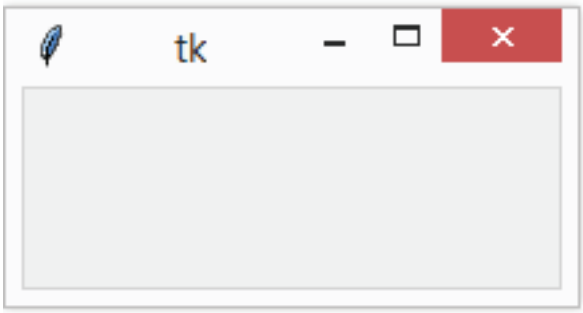
Menu sınıfı, işletim sisteminin doğal kodlarını kullanan ve kendi kendine yeterli olan bir sınıftır. O nedenle, menü yaratırken düğme ya da başka widgetleri işe karıştırmamak gerekir.

19.4 Görünüş

Menü aracının görüntüsü işletim sisteminin uygulamalarındaki menü görüntüsüne benzer. Windows isletim sisteminde Toplevel (üst düzey) menüler başlık satırının altında yer alır. Macintosh'ta ekranın üst satırında yer alır. Üst düzeyde yer alan seçenekler birer düğme gibi çalışır; sürekli görünürler. Birisine tıklayınca altmenüler açılır. Alt menüler, üst düzey menüye bağlıdır ve dört türlü olabilir.

Üst düzey menü yaratmak için, Menü sınıfından bir nesne türetilir. Bu nesne menü için anataşıyıcı olur. anataşıyıcı üzerine üst-düzey menü düğmeleri konulur. Sonra onlara altmenüler eklenir. Bunların nasıl yapıldığını örneklerle göstereceğiz.

Menu sınıfı tkinter içinde bir widget'tir. Onu, hep yaptığımız gibi *Tk()* ile yaratılan anataşıyıcı üzerine koyabiliriz. Ama istersek, onu tek başına da yaratabiliriz. Bunu yapmak için *Menu()* kurucusunu kullanırsak Şekil 19.1



Şekil 19.1: Basit Menu

Bu pencereyi oluşturan kodlar Module 19.1 ile verilmiştir.

Module 19.1.

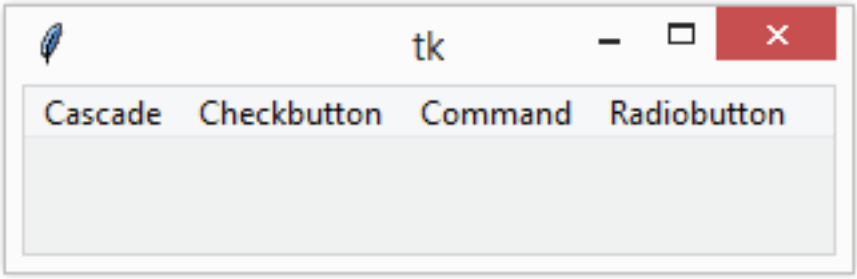
```
from tkinter import *

menu = Menu()

mainloop()
```

Açıklamalar:

İlk ve son satırın işlevlerini biliyoruz. Ortadaki satır *Menu()* kurucusu ile bir menü widgeti yaratıyor. Bu widget'in anataşıyıcıya benzerliğine dikkat ediniz.

19.5 Menü Yaratıcıları

tkinter ile dört tür menu yaratabiliriz: cascade, checkbutton, command, radiobutton. Bunları yaratan fonksiyonlar aşağıda listelenmiştir.



Şekil 19.2: Menu Yaratıcıları

add_cascade(coption,...) [1] Cascade *çağlayan* ya da *açılır menü* diye bilinir. Cascade düğmesi, menü çubuğunda sürekli görünür. Ona basılınca, bağlı olan alt menüler, açılan yeni pencerede alt alta dizilir. Bu düğme için coption (create option) listesine bakınız.

add_checkbutton(coption,...) Checkbutton widget'ini Bölüm 6'de görmüştük. Menüde checkbutton düğmesi, menü şeridinde süreki görünür. Ona basılınca, bağlı olan checkbutton düğmeleri, açılan yeni pencerede görünür. Bu düğme için coption (create option) listesine bakınız.

add_command(coption,...) Menüdeki komut düğmesi menü şeridinde sürekli görünür. Ona basılınca, bağlı komut çalışır. Bağlı komut, genellikle bir fonksiyon çağrısıdır. Bu düğme için coption (create option) listesine bakınız.

add_radiobutton(coption,...) Radiobutton widget'ini Bölüm 7'de görmüştük. Menüde radiobutton düğmesi, menü şeridinde süreki görünür. Ona basılınca, bağlı olan radiobutton düğmeleri, açılan yeni pencerede görünür. Bu düğme için coption (create option) listesine bakınız.

[1] coption= create option

19.6 Ayraç

Arayüzde, estetik nedenlerle, düğmeleri ya da etiketleri birbirlerinden bir çizgi ile ayırmak isteyebiliriz. Bunu yapmak için

```
add_seperator()
```

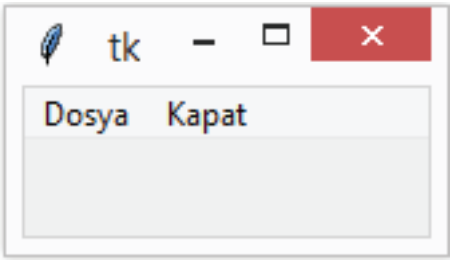
fonksiyonunu kullanırız. Bu fonksiyon, menü şeridinde yatay dizilen düğmeleri ayırmak için düşey bir çizgi oluşturur. Açılır (pop-up) penceresinde düşey sıralanan etiketleri ayırmak için yatay bir çizgi oluşturur.

Module 19.2 üst düzey menü yaratıyor.

Module 19.2.

```
from tkinter import *

win = Tk()
def DosyaBul():
    print ("Dosya bulunamadı!")

# üst düzey menü yarat
menubar = Menu(win)
menubar.add_command(label="Dosya", command=DosyaBul)
menubar.add_command(label="Kapat", command=win.quit)

# menüyü göster
win.config(menu=menubar)

mainloop()
```

Açıklamalar:



Şekil 19.3: Üstdüzey Menü Etiketlei

1.satır tkinter'in ad uzayındaki her şeyi çağırıyor.

3.satır, Tk() kurucusu ile bir pencere (anataşıyıcı, toplevel widget) yaratıyor ve pencereye `win` dını veriyor. anataşıyıcıya istediğniz adı verebilirsiniz, ama anataşıyıcı olduğunu algılatan bir ad tercih edilmelidir.

4.satır *DosyaBul* adlı bir fonksiyon tanımlıyor. Gerçekte, bu fonksiyon dosyayı aramadan *"Dosya bulunamadı"* iletisini veriyor. Şimdilik, basitçe, üst düzey menüye tıklandığında, bir fonksiyonun çağrılması örnekleniyor. İleride dizinlerimizde dosya arama eyleminin nasıl yapıldığını göreceğiz.

8.satırdaki `Menu(win)` kurucusu, yaratılan nesnenin win üzerine konulmasını istiyor. Ayrıca yaratılan nesneye *menubar* adını veriyor.

9.satırdaki `add_command()` fonksiyonu label parametresine atanan "Dosya" etiketini *menubar* üzerine koyuyor. Burada menubar, bir taşıyıcı rolünü üstleniyor. İkinci sıradaki *command* parametresi "DosyaBul" fonksiyonunu çağırıyor. tkinter'de bu eyleme *callback* deniliyor.

10.satır, benzer biçimde menubar üzerine "Kapat" etiketini koyuyor ve *quit* fonksiyonunu win penceresi için çağırıyor. Aksi söylenmediği için, menubar üzerine konulan etiketler soldan sağa doğru yerleşiyorlar. Bu yerleşim biçimi öntanımlıdır (default). Quit fonksiyonu *win* penceresini kapatıyor.

13.satır, win.config(menu= menubar) fonksiyonu, win anataşıyıcıya *config* fonksiyonunu uyguluyor. *config* fonksiyonu widgetleri isteğimize uyarlamak (configure) için kullanılır. Bu fonksiyonun çok sayıda seçimlik parametresi (options) vardır. *menu* parametresi onlardan birisidir. Windows işletim sisteminde *menu*= *menubar* değeri, menü şeridini pencerenin başlığı altına yerleştiriyor.

15.satırdaki mainloop() döngüsü, kapatılana kadar `win` penceresini görünür kılıyor.

Table 31-1. Menu Item Options

Module 19.3 sınıf kullanarak basit bir menü oluşturuyor. Menü'nün üst düzeyinde yalnızca Dosya var. Dosyaya tıklayınca Açılır menüde Çık sekmesi geliyor. Ona tıklayınca menü bütünüyle kapanıyor.

Modül çok basittir; ama yeni başlayanlar modülü kendi başlarına satır satır anlamaya çalışmalıdırlar. Gerekirse, modülün sonunda verilen açıklamalara bakılabilir. Ancak bu basit modüller anlaşılmazsa, sonrakileri anlamakta çok zorlanabilirsiniz.

Module 19.3.

```
"""
author: Jan Bodnar
last modified: December 2010
website: www.zetcode.com
"""

from tkinter import Tk, Frame, Menu
```

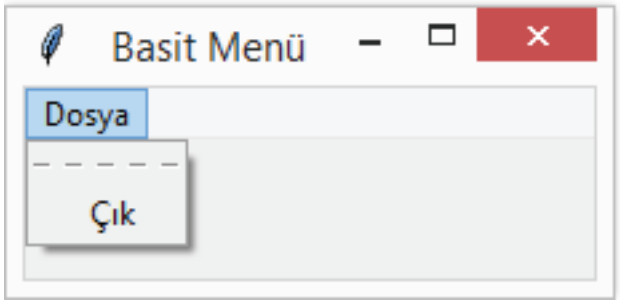
Option	Tip	Açıklama
activebackground	color	Seçili iken zemin rengi
activeforeground	color	Seçili iken yazı rengi
accelerator	string	
background	color	Zemin rengi
bitmap	bitmap	resim
columnbreak	flag	
command	callback	Fonksiyon çağırma
font	font	Font
foreground	color	Yazı rengi
hidemargin	flag	
image	image	Resim
indicatoron	flag	
label	string	Text
menu	widget	Menu pencere aracı
offvalue	value	
onvalue	value	
selectcolor	color	Renk seç
selectimage	image	Resim seç
state	constant	Sabit değer
underline	integer	Alt çizgili
value	value	Değer
variable	variable	Değişken

Tablo 19.1: Menu Seçimlik Parametreleri

```
10 class Kolay(Frame):

     def __init__(self, parent):
          Frame.__init__(self, parent)

15        self.parent = parent
          self.initUI()

     def initUI(self):

20        self.parent.title("Basit Menü")

          menubar = Menu(self.parent)
          self.parent.config(menu=menubar)

25        dosyaAç = Menu(menubar)
          dosyaAç.add_command(label="Çık", command=self.onExit)
          menubar.add_cascade(label="Dosya", menu=dosyaAç)


30   def onExit(self):
          self.quit()


def main():
```

```
35
    root = Tk()
    root.geometry("250x150+300+300")
    app = Kolay(root)
    root.mainloop()
40

if __name__ == '__main__':
    main()
```

Açıklamalar:



Şekil 19.4: tearoff = 1

1-5.satırlar Python'da açıklamadır (rem). Python derleyicisi onları işlemez.

7.satır, tkinter'den Tk, Frame ve Menu modüllerini çağırıyor.

10-16.satırlar, Frame'in altsınıfı olan *Kolay* adlı bir sınıf tanımlıyor. Dolayısıyla Frame'in bütün öğeleri ona kalıtsal geçer. Burada *self* Kolay sınıfıdır. 12.satır `__init__` fonksiyonunu tanımlamaya başlıyor. 13.satır sözkonusu fonksiyonun gövdesidir. Bu satır, 12.satırda tanımlanmaya başlanan `__init__` fonksiyonunu Frame'in `__init__` fonksiyonu olarak tanımlıyor. 15.satır *self.ata* değişkenini tanımlıyor ve ona *ata* değerini atıyor. 18.satır *self.initUI()* fonksiyonunu çağırıyor.

`__init__` fonksiyonu Kolay sınıfının atası (parent) olan Frame sınıfına ait bir self.ata adlı bir nesne türetiyor. Pencere araçları self.ata üzerine konulacaktır.

18-27.satırlar initUI() fonksiyonunu tanımlıyor. Bu fonksiyon 20.satırda *self.ata* penceresine "Basit Menü" başlığını veriyor. 22. satırda Menu() kurucu fonksiyonu ile *menubar* adlı menü şeridini yaratıyır ve onu *self.ata* üzerine koyuyor. 23.satırda self.ata için *config()* fonksiyonunu çalıştırıyor. Bu fonksiyon menu'ye *menubar* şeridinin eklenmesini istiyor. 25.satırdaki Menu(menubar) fonksiyonu o işi yapan nesneyi yaratıyor ve ona dosya adını

veriyor. Tabii, dosya yerine istenen başka bir ad da verilebilirdi. 26.satır *ad_ command()* metodunu dosya nesnesine uyguluyor. Bu metot, dosya nesnesine "Çık" etiketini koyuyor ve *command* parametresinin `onExit()` fonksiyonunu çağımasını söylüyor. "Çık" dümesine basıldığında `self.quit()` fonksiyonu çalışacak ve pencere bütünüyle kapanacaktır. 27.satır, menubar nesnesine ad_cascade() fonksiyonunu uyguluyor. Bu fonksiyon "Dosya" ya tıklandığında açılan açılır (cascede) bir pencere ekliyor.

30-31.satırlar çıkış (onExit) fonksiyonunu tanımlıyor.

34-39.satırlar asıl işi yapacak olan main() fonksiyonunu tanımlıyor.

42-43.satırlar programı çalıştırıyor ve menüyü oluşturuyor.

Module 19.4.

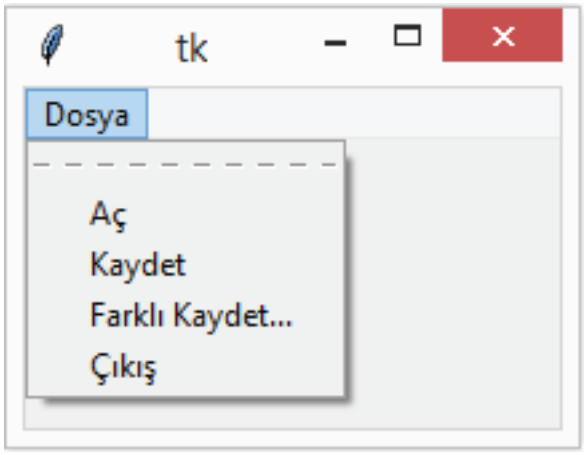
```
#!/usr/bin/env python
#-*-coding:utf-8-*-

from Tkinter import *

pencere = Tk()

menu = Menu(pencere)
pencere.config(menu=menu)
dosya = Menu(menu)
menu.add_cascade(label="Dosya",menu=dosya)
dosya.add_command(label="Aç")
dosya.add_command(label="Kaydet")
dosya.add_command(label="Farklı Kaydet...")
dosya.add_command(label="Çıkış",command=pencere.quit)

mainloop()
```

19.7 add_cascade()

Module 19.5, [6]'den uyarlanmıştır.

Module 19.5.

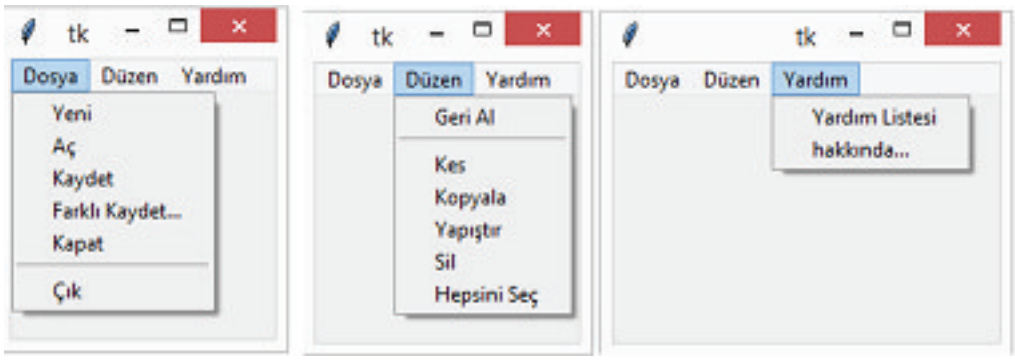
```
from tkinter import *
def yapma():
    filewin = Toplevel(pencere)
    düğme = Button(filewin, text="Bir şey yapma!")
    düğme.pack()

pencere = Tk()
menubar = Menu(pencere)

anaMenü = Menu(menubar, tearoff=0)
```



Şekil 19.5: Açılır Menü (cascade)

```
menubar.add_cascade(label="Dosya", menu=anaMenü)
anaMenü.add_command(label="Yeni", command=yapma)
anaMenü.add_command(label="Aç", command=yapma)
anaMenü.add_command(label="Kaydet", command=yapma)
anaMenü.add_command(label="Farklı Kaydet...", command=yapma)
anaMenü.add_command(label="Kapat", command=yapma)

anaMenü.add_separator()
anaMenü.add_command(label="Çık", command=pencere.quit)

düzenle = Menu(menubar, tearoff=0)
menubar.add_cascade(label="Düzen", menu=düzenle)
düzenle.add_command(label="Geri Al", command=yapma)
düzenle.add_separator()
düzenle.add_command(label="Kes", command=yapma)
düzenle.add_command(label="Kopyala", command=yapma)
düzenle.add_command(label="Yapıştır", command=yapma)
düzenle.add_command(label="Sil", command=yapma)
düzenle.add_command(label="Hepsini Seç", command=yapma)


yardım = Menu(menubar, tearoff=0)
menubar.add_cascade(label="Yardım", menu=yardım)
yardım.add_command(label="Yardım Listesi", command=yapma)
yardım.add_command(label="hakkında...", command=yapma)


pencere.config(menu=menubar)
pencere.mainloop()
```

Açıklamalar:



Şekil 19.6: Açılır Menu Cascade)

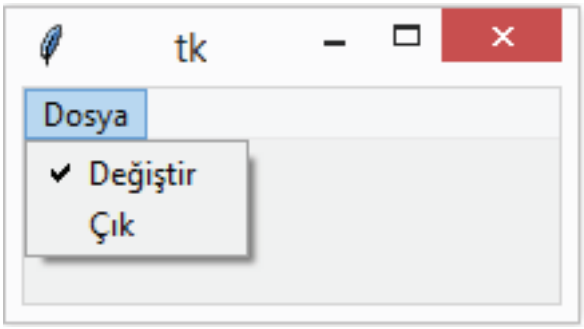
19.8 Checkbutton Menu

Module 19.6.

```
1  from tkinter import *

   class anaPencere:
     def __init__(self):
       self.pencere=Tk()
6      self.tıkla=IntVar()
       self.tıkla.set(1)
       self.createWidgets()
       return None

11   def createWidgets(self):
       menuBar=Menu(self.pencere)

       anaMenü=Menu(menuBar, tearoff=0)
       menuBar.add_cascade(label="Dosya",menu=anaMenü)
16     anaMenü.add_checkbutton(label="Değiştir",variable=self.tıkla)
       anaMenü.add_command(label="Çık", command=self.kapat)

       self.pencere.config(menu=menuBar)
       return None
21
     def kapat(self):
       sys.exit(0)

   def main():
26   anaPencere()
     mainloop()

   if __name__== '__main__':
     main()
```

Açıklamalar:

3.satır *anaPencere* adlı bir sınıf tanımlamaya başlıyor.



Şekil 19.7: Checkbutton

4.satır __init__ fonksiyonunu tanımalamaya başlıyor. *self* terimi burada anaPencereden türetilen nesneyi temsil ediyor.

5.satırda Tk() kurucusu bir anataşıyıcı yaratıyor, yaratılan nesneye *pencere* adı veriliyor.

6.satırda anaPencere sınıfının tıkla adlı değişkenine IntVar() fonksiyonu atanıyor. IntVar() tamsayı değerler alan bir fonksiyondur.

7.satırda set() metodu tıkla değişkenine 1 değerini atıyor. checkbutton için 1 değeri düğmemin seçilmiş, 0 değeri düğmenin seçilmemiş olduğu anlamına geliyor.

8.satır createWidgets() fonksiyonunu çağırıyor.

9.satır __init__ fonksiyonunun döndürdüğü bir değer olmadığını söylüyor.

11.satır, 8.satırda çağrılan fonksiyonu tanımlamaya başlıyor.

12.satır anataşıyıcı üzerine Menu() kurucusunun yarattığı menuBar adlı nesneyi koyuyor. Bu nesne ana taşıyıcı üzerine konulan ve *menuBar* adı verilen başka bir taşıyıcıdır. Çünkü Menu() kurucusunun görevi menü araçlarını taşıyacak taşıyıcılar yaratmaktır.

14.satır Menu() kurucusunun yarattığı *anaMenü* adlı nesneyi *menuBar* taşıyıcısı üzerine koyuyor. Böylece en altta *pencere*, onun üstünde *menuBar*, onun üstünde *anaMenü* nesnesi var. Arayüzün görünüşü açısından bakarsak, *pencere* arayüzün anataşıyıcısıdır. *menuBar*, pencere'nin başlığı altına yerleşen *menü çubuğu*'dur. Menü çubuğunun üzerinde *anaMenü* nesnesi var. *pencere* adlı anataşıyıcı dışındakilere kenar koymadığımız için, onlar arayüzde görünmezler, ama işlevlerini yaparlar. *anaMenü* üzerine seçmeli menü düğmeleri konulur.

Bu örnekte, basitliği sağlamak için, *anaMenü* üerine bir tek *Dosya* adlı üst düzey menü seçeneğini koyacağız. Benzer yolla, anaMenüye istenildiği kadar üst düzey menü seçeneği konulabilir.

Öntanımlı olarak, checkbutton düğmelerinden önce ilk satıra tearoff denilen ve - - - ile gösterilen sıyırma satırı yerleşir. Bu satırı önlemek için Menu() fonksiyonunun ikinci parametresi olan *tearoff* parametresine 0 değeri atanır. *anaMenü* üzerine konulan düğmeye tıklandığında ona bağlanacak *checkbutton* düğmeleri açılacaktır. Bu açılış sırasında *tearoff=0* değeri, menuBar ile *checkbutton* düğmeleri arasında boşluk bırakmaz. Eğer *tearoff=1* değeri konulursa, menuBar ile *checkbutton* düğmeleri arasında bir satırlık bir boşluk oluşur. Oluşan boşluğa *tearoff* (sıyırma) denilen - - - işaretli bir satır gelir.

15.satırda ad_cascade() metodu menuBar üzerine Dosya etiketli üst düzet menü seçeneğini koyuyor.

16.satır add|chack metodu ile Değiştir etiketli checkbutton düğmesini anaMenü üzerine koyuyor. İkinci parametre değişkene tıkla değerini atıyor. açılan checkbutton düğmesine tıklandığında seçili ise seçisiz, seçisiz ise seçili olmasını sağlar.

17.satır add_command() metodu ile anaMenü'ye Çık adlı bir komut düğmesi ekliyor. Düğmeye tıklandığında *cmmand=self.kapat* ataması pencereyi kapatıyor. *kapat* fonksiyonu 22.satırda tanımlanmıştır.

19.satırdaki config() fonksiyonu menubar çubuğunu pencere üzerine yerleştiriyor.

20.satır createWidget() fonksiyonunun döndürdüğü bir değer olmadığını söylüyor.

22.satır, sistemin exit() fonksiyonunu 0 değeri ile çağırıyor. exit(0) fonksiyonu işletim sisteminde açık uygulamayı kapatan fonkiyondur. Gerekiyorsa, programın başına import sys deyimini yazarak sys modülünü çağırınız.

25.satırda programa asıl işi yaptıracak main() fonksiyonu tanımlanmaya başlanıyor.

26.satır anaPencere sınıfının bir nesnesini türetiyor (instantiate).

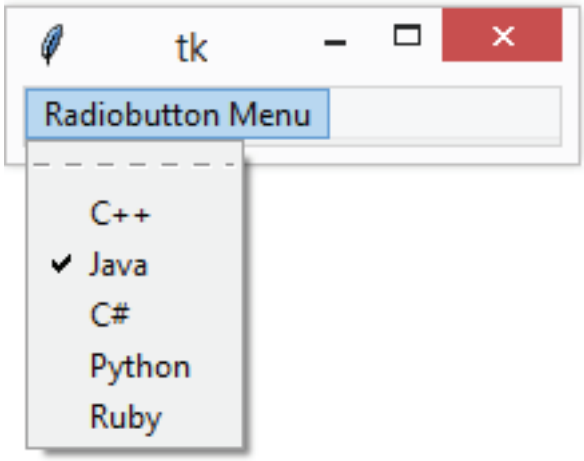
27.satırdaki mainloop() döngüsü, kapatılana kadar pencereyi ekranda görünür kılıyor.

19.9 Radyo Düğmeli Menu

Module 19.7.

```
from tkinter import *
from tkinter import *

class TestMenu:
    def __init__(self, master):
        self.master = master
        self.menubar = Menu(self.master)

        self.radyoDüğmesi = Menu(self.menubar)
        self.radyoDüğmesi.add_radiobutton(label='C++')
        self.radyoDüğmesi.add_radiobutton(label='Java')
        self.radyoDüğmesi.add_radiobutton(label='C#')
        self.radyoDüğmesi.add_radiobutton(label='Python')
        self.radyoDüğmesi.add_radiobutton(label='Ruby')

        self.menubar.add_cascade(label="Radiobutton Menu", menu=self.
         radyoDüğmesi)

        self.top = Toplevel(menu=self.menubar, width=200, relief=
         RAISED,
                            borderwidth=2)

def main():
    root = Tk()
    root.withdraw()
    app = TestMenu(root)
    root.mainloop()

if __name__ == '__main__':
    main()
```

Açıklamalar:



Şekil 19.8: radiobutton Menu