Bölüm 5

Frames

5.1 Frame Nedir?

Frame (çerçeve), arayüze konulan widgetleri gruplayıp taşıyan bir taşıyıcıdır, bir sınıftır, bir widget'tir. Basit arayüzlerde çerçeveye gerek duyulmayabilir, ama arayüz çok sayıda widget taşıyor ve onlar kendi aralarında gruplara ayrılıyorsa, her grup için bir çerçeve yaratmak algıyı kolaylaştırabilir.

Çoğunlukla çerçeveler (Frame nesneleri) anataşıyıcı üzerine konulur. Çerçeve üzerine çerçeve konulabilir. Böylece içiçe çerçeveler oluşturulabilir.

Frame sınıfından bir çerçeve yaratmak için Frame() kurucusu kullanılır. Sözdizmi Liste 5.1 gibidir.

Liste 5.1.

```
fr = Frame ( taban, seçimlik, ... )
```

Açıklamalar: Frame() kurucusunun ilk parametresi zorunlu olmak üzere, ötekileri seçimlik (optional) olan parametreleri vardır:

taban: çerçeveyi taşıyacak olan widget. Örneğimizde, bu, zorunlu olan ilk parametredir.

seçimlik,... : Çerçevenin boyutlarını, renklerini, rölyef biçemlerini belirleyen sabit parametreler şunlardır:

FLAT, RAISED, SUNKEN, GROOVE, RIDGE

Seçimlik parametreler, gereksemeye göre seçilebilir; seçiliş sırasının önemi

yoktur. Frame() kurucusunun başlıca seçimlik parametreleri Liste 5.2 ile verilmiştir.

Liste 5.2.

bg Çerçevenin zemin rengi.

bd Çerçevenin sınır çizgilerinin kalınlığı. Bir değer atanmamışsa, öndeğeri (default) 2 pixel'dir

cursor Fare işaretçisinin biçemini seçer. Fareyle imleç çerçevedeki checkbutton widgeti üzerine konulunca, seçilen imleç biemi görünür.

height Çerçevenin yüksekliği.

highlightbackground Fare ile üzerine gidildiğinde çerçevenin zemini aydınlanır.

highlightcolor fare ile üzerine gidildiğinde çerçevenin aydınlığı için renk seçer

highlightthickness Aydınlanmanın yoğunluğunu seçer.

relief FLAT, RAISED, SUNKEN, GROOVE RIDGE (düz, kabartma, oyma, oluk kenarlı, sırt kenarlı) biçemlerinden birisi seçilebilir. Öndeğer rölyef relief=FLAT olur. Rölyef seçenekleri checkbutton dışındaki bütün widgetlere uygulanabilir ancak, checkbutton kendi zemininden yukarıya çıkmaz.

width Çerçevenin genişliğini belirler. Çerçeve, üzerine konulan widgetlere göre kendi genişliğini belirler. checkbutton genişliği, üzerine konulan ikon ya da resimlere göre kendiliğinden ayarlanır.

5.2 Örnekler

Önce boş bir çerçeveden başlayarak, arayüzü adım adım geliştirelim.

Liste 5.3.

```
from tkinter import *

win = Tk()
frame1 = Frame(win)
frame1.pack()

win.mainloop()
```

Açıklamalar:

1.satır tkinter paketini çağırıyor. Bu çağrıdan sonra, arayüz için gerekli olan her şey programımıza girmiş oluyor.

3.satır Tk() kurucusu ile anataşıyıcıyı yaratıyor ve onu `win` pointeri ile işaret ediyor.

4.satır Frame() kurucusu ile `frame1` adlı (frame1 pointerinin işaret ettiği) bir Frame nesnesi (çerçeve) yaratıyor. Burada zorunlu olarak kullanılan ilk parametre `win` parametresidir. *frame1* widgetinin `win` adlı anataşıyıcı üzerine konulacağını gösteriyor. Başka seçimli parametre kullanılmadığı için, öndeğerler geçerli olur. Örneğin, üzerine bir şey konulmadığı için çerçevenin genişliği anataşıyıcının genişliğine etkimez. `win` penceresi kendi başlığının gerektirdiği genişliği alır.

5.satır `pack()` konuşlandırıcısını çalıştırıyor. Bu metot frame1 çerçevesini `win` üzerine koyuyor ve win'in genişliğini üstüne konulan widgetlere göre ayarlıyor. Burada frame1 boş olduğu için, win'in genişliğine etki emiyor. `win` anataşıyıcısı, kendi başlığının gerektirdiği genişliğe ayarlanıyor.

7.satır arayüzün, kapatılana kadar görünmesini sağlayan döngüdür.

Görüntü Şekil 5.1 gibidir.

Şekil 5.1: Boş bir Frame İçeren Arayüz

Şimdi `win` anataşıyıcısın üzerine bir frame daha koyalım. Liste 5.4 o işi yapıyor.

Liste 5.4.

```
from tkinter import *

win = Tk()
frame1 = Frame(win)
frame.pack()

frame2 = Frame(win)
frame2.pack( side = BOTTOM )

win.mainloop()
```

Açıklamalar:

7.satır Frame sınıfından `frame2` adlı bir çerçeve yaratıyor ve onu `win` anataşıyıcısı üzerine konulmasını istiyor.

8.satır, `frame2` için konuşlandırıcıdır. `pack()` konuşlandırıcısında *side* değeri olarak BOTTOM seçilmiştir. `side`, widgetin nereye konuşlanacağını belirtir. `side`'in değeri TOP, RIGHT, BOTTOM ve LEFT sabitlerinden ancak birisi olabilir. Bu sabitler `tkinter.constant` modülü içindedirler. Sırasıyla üst, sağ, alt ve sol kenara yanaşık olmayı ifade ederler. Konuşlandırmada iki kural işler:

1. Sözkonusu widget, taşıyıcıya konulan ilk widget ise, *side* değeri taşıyıcının kenarlarına göre değer alır.
2. Taşıyıcıya sözkonusu widget'ten önce konulan başka widgetler varsa, *side*'in değeri sözkonusu widgetten bir önceki widget'in kenarlarına göre değer alır.

Örneğimizde frame1 önce konulduğu için, side = BOTTOM ifadesi frame2'nin, frame1 nesnesinin altına konuşlanmasını sağlıyor.

`frame2` nesnesi üzerine de bir şey koymadığımız için, öncekinde olduğu gibi, bu da win'in genişkiğine etki etmeyecektir. Yarattığımız her iki frame boş olduğu için, arayüzümüz onları, `win` içinde Şekil 5.2'de olduğu gibi ince bir dikdörtgen olarak gösterecektir.

Şekil 5.2: İki Boş frame İçeren Arayüz

Şimdi *frame1* üzerine bir düğme (button) koyalım. Liste 5.5 o işi yapıyor.

Liste 5.5.

```
from tkinter import *

win = Tk()
frame1 = Frame(win)
frame1.pack()

frame2 = Frame(win)
frame2.pack(side = BOTTOM )

kırmızıDüğme = Button(frame1, text="Krmızı", fg="red")
```

```
kırmızıDüğme.pack( side = LEFT)

win.mainloop()
```

Açıklamalar:

Önceki programa 10.ve 11.satırlar eklenmiştir.

10.satır, Button() kurucu metodu ile Button sınıfından, *kırmızıDüğme* adlı bir nesne yaratıyor ve bunun *frame1* çerçevesi üzerine konulmasını istiyor. Yaratılan düğmenin üstündeki yazı text= "Kırmız" ile belirleniyor. Yazının rengi ise fg= "red" ataması ile kırmızı oluyor. Burada `fg`, foreground'un kısaltmasıdır; yazı rengini belirler.

11.satır, pack() konuşlandırıcısı ile *kırmızıDüğme* nesnesini *frame1* taşıyıcısı içine konuşlandırıyor. Buna paketleme deniyor. Düğmeyi, *frame1* çerçevesi içine ortalayarak yerleştiriyor.

Dördüncü adım olarak, *frame1* üzerine iki düğme daha koyalım. Liste 5.6 o işi yapıyor.

Liste 5.6.

```
from tkinter import *

win = Tk()
frame1 = Frame(win)
frame1.pack()

frame2 = Frame(win)
frame2.pack(side = BOTTOM )

kırmızıDüğme = Button(frame1, text="Kırmızı", fg="red")
kırmızıDüğme.pack( side = LEFT)

yeşilDüğme = Button(frame1, text="Yeşil", fg="green")
yeşilDüğme.pack( side = LEFT )

maviDüğme = Button(frame1, text="Mavi", fg="blue")
maviDüğme.pack( side = LEFT )

win.mainloop()
```

Açıklamalar:

13-17 satırlardaki deyimler kırmızıDüğme için yapılanların benzeridir. Yalnızca renkler yeşil ve mavi olmuştur. Burada, her üç düğme için pack() konşlndırıcısında `side = LEFT` değeri atanmıştır. Kırmızı düğme ilk konulan widgettir. Onu win taşıyıcısının sol kenarına yanaşık koydu. Böyle olduğunu görmek için

Şimdi fare1 ve frame2 üzerine düğmelet (button) koyalım. Liste 5.7 o işi yapıyor.

Liste 5.7.

```
from tkinter import *

win = Tk()
frame = Frame(win)
frame.pack()

bottomframe = Frame(win)
bottomframe.pack( side = BOTTOM )

redbutton = Button(frame, text="Red", fg="red")
redbutton.pack( side = LEFT)

greenbutton = Button(frame, text="Brown", fg="brown")
greenbutton.pack( side = LEFT )

bluebutton = Button(frame, text="Blue", fg="blue")
bluebutton.pack( side = LEFT )

blackbutton = Button(bottomframe, text="Black", fg="black")
blackbutton.pack( side = BOTTOM)

win.mainloop()
```

Açıklamalar: