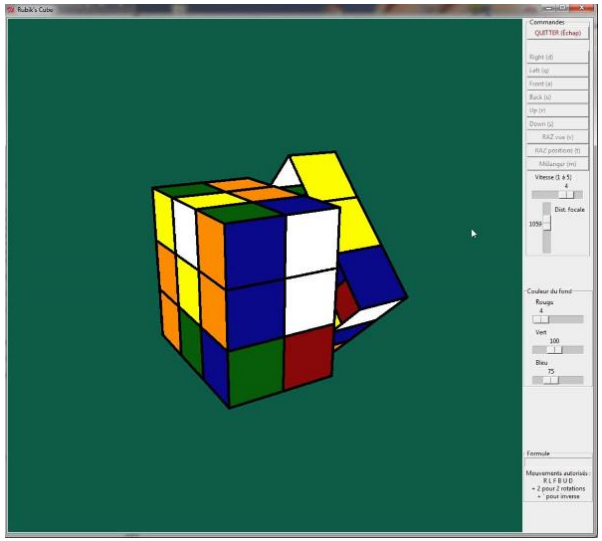
9장 tkinter를 이용한 GUI 프로그래밍

tkinter란?

- **tkinter는 파이썬에서 그래픽 사용자 인터페이스(GUI: graphical user interface)를 개발할 때 필요한 모듈**

tkinter의 유래

- **tkinter**는 예전부터 유닉스 계열에서 사용되던 **Tcl/Tk** 위에 객체 지향 계층을 입힌 것이다. **Tk**는 **John Ousterhout**에 의하여 **Tcl** 스크립팅 언어를 위한 **GUI** 확장으로 개발

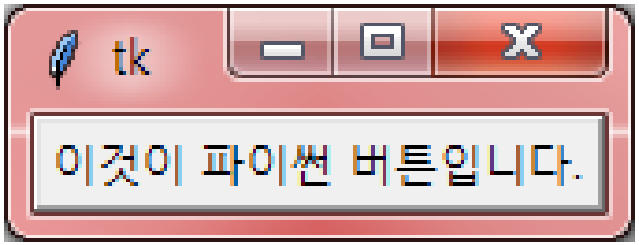


첫 번째 tkinter 프로그램

```
from tkinter import *

window = Tk()

label = Label(window, text="Hello World!")
label.pack()

window.mainloop()
```

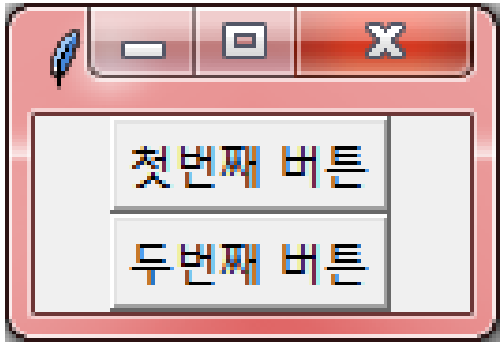
버튼과 이벤트 처리

```
from tkinter import *

window = Tk()

b1 = Button(window, text="이것이 파이썬 버튼입니다.")
b1.pack()

window.mainloop()
```



배치 관리자 소개

```
from tkinter import *

window = Tk()
b1 = Button(window, text="첫번째 버튼")
b2 = Button(window, text="두번째 버튼")
b1.pack()
b2.pack()

window.mainloop()
```

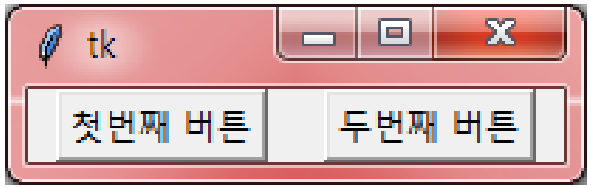


배치 관리자 소개

```
from tkinter import *

window = Tk()
b1 = Button(window, text="첫번째 버튼")
b2 = Button(window, text="두번째 버튼")
b1.pack(side=LEFT)
b2.pack(side=LEFT)

window.mainloop()
```

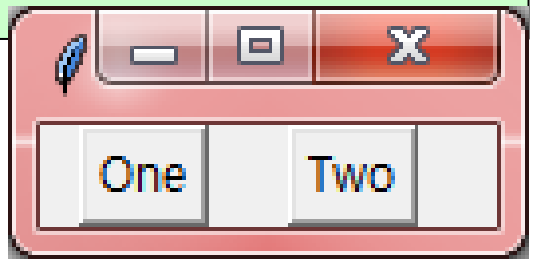
배치 관리자 소개

```
from tkinter import *

window = Tk()
b1 = Button(window, text="첫번째 버튼")
b2 = Button(window, text="두번째 버튼")
b1.pack(side=LEFT,padx=10)
b2.pack(side=LEFT,padx=10)

window.mainloop()
```



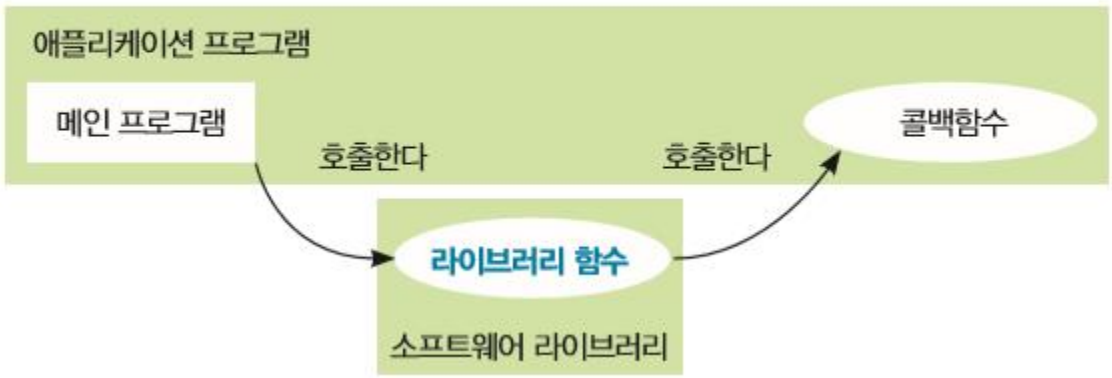
버튼의 텍스트 변경

```
from tkinter import *

window = Tk()
b1 = Button(window, text="첫번째 버튼")
b2 = Button(window, text="두번째 버튼")
b1.pack(side=LEFT,padx=10)
b2.pack(side=LEFT,padx=10)
b1["text"] = "One"
b2["text"] = "Two"

window.mainloop()
```



이벤트 처리 소개

- **tkinter** 프로그램은 이벤트에 기반을 두고 동작된다.

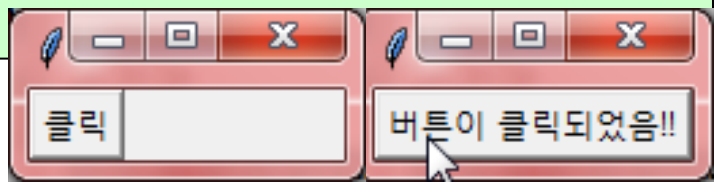


버튼의 이벤트를 처리하려면

전체적인 구조

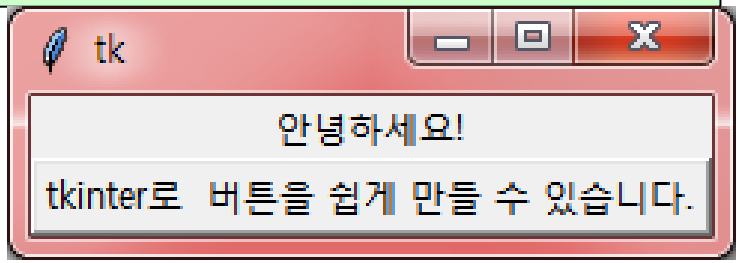
```
button = Button(window, text="one", command= 함수 이름 )
```

```
from tkinter import *

def callback():
    button["text"] ="버튼이 클릭되었음!"

window = Tk()
button = Button(window, text="클릭", command=callback)
button.pack(side=LEFT)

window.mainloop()
```



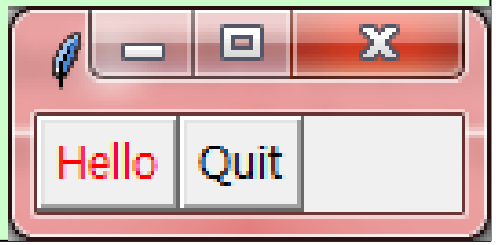
예제

```
from tkinter import *

window  = Tk()

label = Label(window, text="안녕하세요!")
label.pack()

button = Button(window, text="tkinter로  버튼을 쉽게 만들 수 있습니다.")
button.pack()

window.mainloop()
```



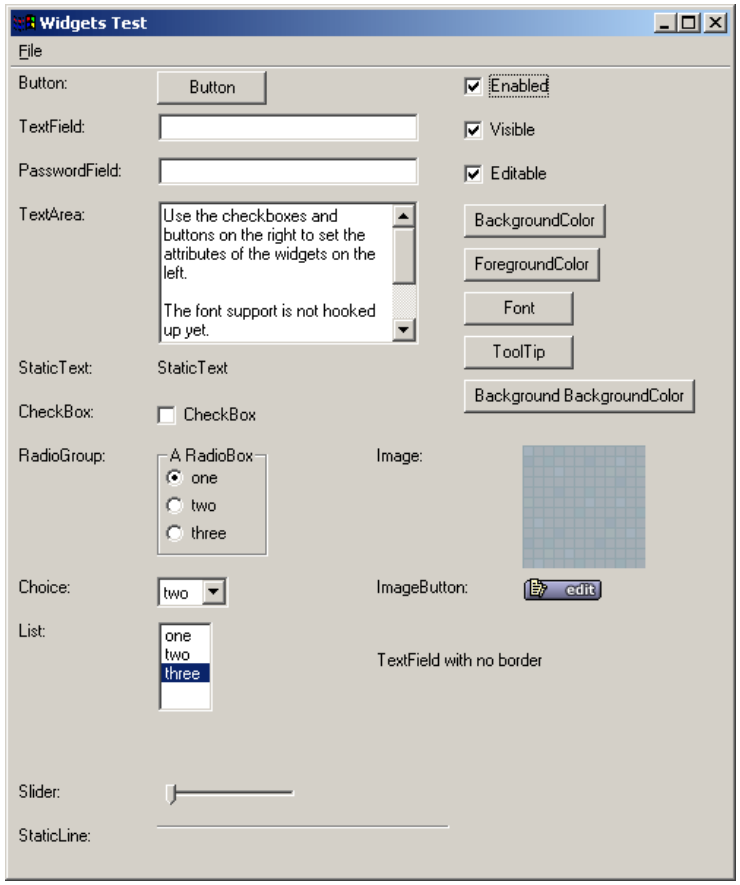
클래스로 프레임 감싸기

```
from tkinter import *

class App:
    def __init__(self ):
        window = Tk()
        helloB = Button(window, text="Hello", command=self.hello, fg="red")
        helloB.pack(side=LEFT)
        quitB = Button(window,  text="Quit", command=self.quit)
        quitB.pack(side=LEFT)
        window.mainloop()

    def hello(self):
        print("Hello 버튼이 클릭되었음!")

    def quit(self):
        print("Quit 버튼이 클릭되었음!")

App()
```

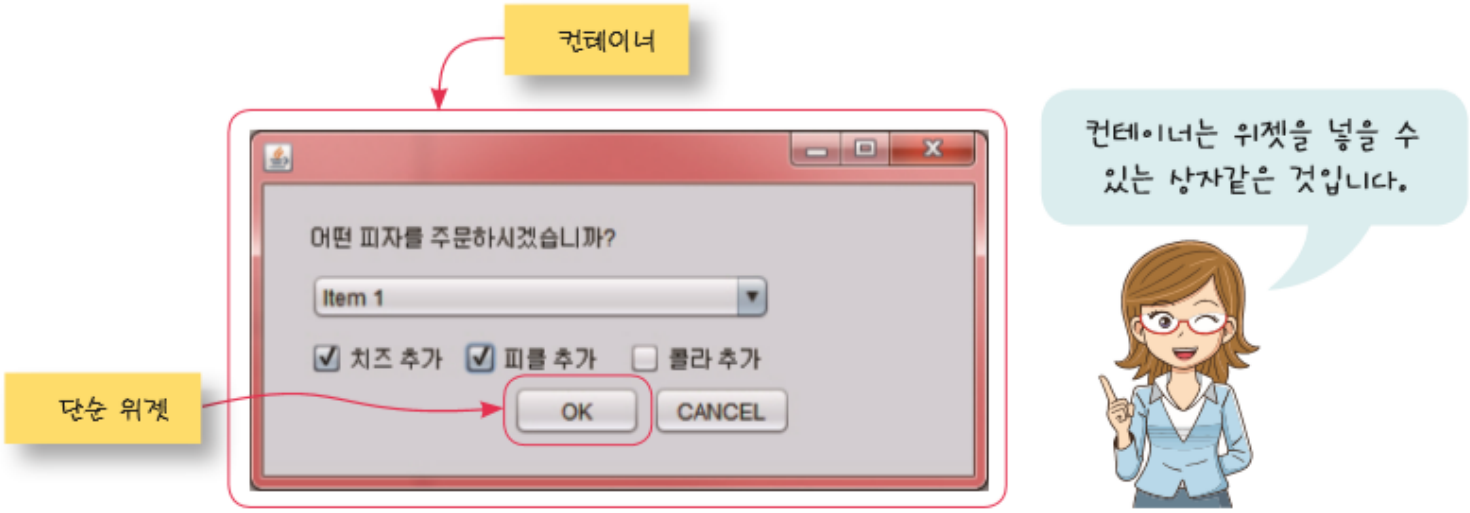


tkinter의 위젯들

- Button
- Canvas
- Checkbutton
- Entry
- Frame
- Label
- Listbox
- Menu
- Menubutton
- Message
- Radiobutton
- Scale
- Scrollbar
- Text
- Toplevel
- LabelFrame
- PanedWindow
- Spinbox



단순 위젯과 컨테이너 위젯



배치 관리자

- **Grid**
  - **격자 배치 관리자(grid geometry manager)**는 테이블 형태의 배치
- **Pack**
  - **압축 배치 관리자(pack geometry manager)**는 위젯들을 부모 위젯 안에 압축
- **Place**
  - **절대 배치 관리자(place geometry manager)**는 주어진 위치에 위젯을 배치

색상

- 색상: 부분의 위젯은 배경(**bg**)과 전경(**fg**) 변수를 사용하여 위젯 및 텍스트 색상을 지정

```
from  tkinter import *
window = Tk()
button = Button(window, text="버튼을 클릭하세요")
button.pack()
button["fg"] = "yellow"
button["bg"] = "green"
```

색상 대화상자

- 사용자에게 색상을 선택하게 한다.

```
from  tkinter import *
color=colorchooser.askcolor()
print(color)
```

폰트

- 폰트를 튜플로 지정할 수 있는데 여기에는 (폰트이름, 폰트의 크기, 폰트 스타일)과 같은 형식을 사용한다.

```
("Times", 10, "bold")
("Helvetica", 10, "bold italic")
("Symbol", 8)
```

- 문자열로도 지정

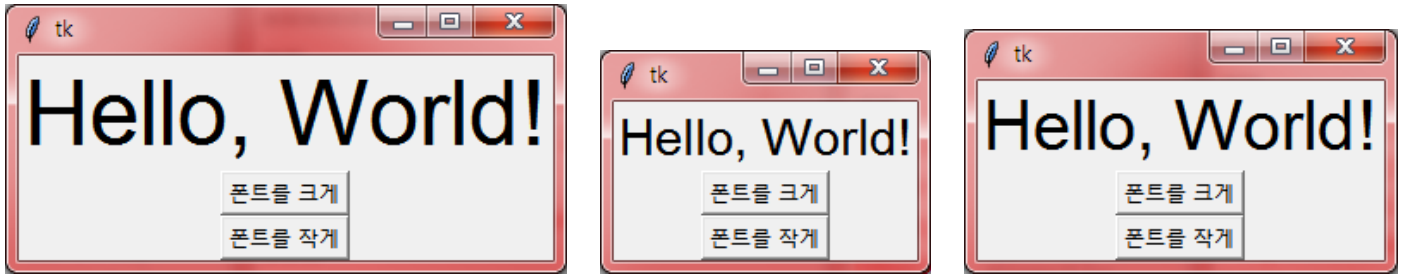
```
w = Label(master, text="Helvetica", font="Helvetica 16”)
```

예제

```
import tkinter as tk
import tkinter.font as font

class App:
    def __init__(self):
        root=tk.Tk()

        self.customFont = font.Font(family="Helvetica", size=12)

        buttonframe = tk.Frame()
        label = tk.Label(root, text="Hello, World!", font=self.customFont)
        buttonframe.pack()
        label.pack()

        bigger = tk.Button(root, text="폰트를 크게", command=self.BigFont)
        smaller = tk.Button(root, text="폰트를 작게", command=self.SmallFont)
        bigger.pack()
        smaller.pack()
```

예제

```
        root.mainloop()

    def BigFont(self):
        size = self.customFont['size']
        self.customFont.configure(size=size+2)

    def SmallFont(self):
        size = self.customFont['size']
        self.customFont.configure(size=size-2)

app=App()
```



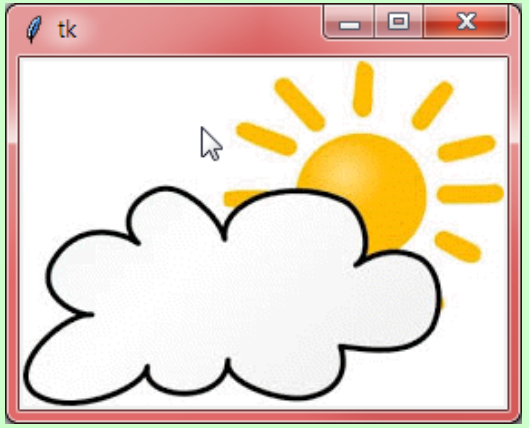
레이블

전체적인 구조

```
w = Label(parent , option, ... )
```

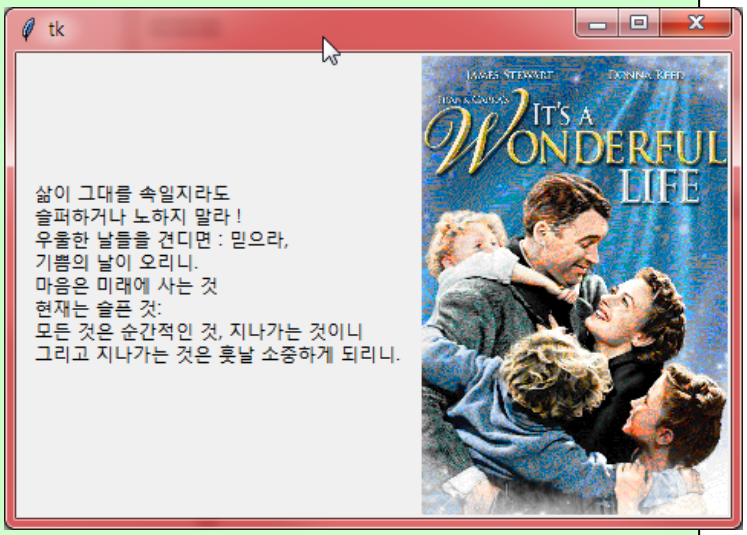
레이블로 화면에 이미지 표시하기

```
from tkinter import *

window = Tk()
photo = PhotoImage(file="a1.gif")
w = Label(window, image=photo)
w.photo = photo
w.pack()
window.mainloop()
```



레이블에 이미지와 텍스트를 동시에 나타내기

```
from tkinter import *

window = Tk()
photo = PhotoImage(file="wl.gif")
w = Label(window, image=photo).pack(side="right")
message= """삶이 그대를 속일지라도
슬퍼하거나 노하지 말라 !
우울한 날들을 견디면 : 믿으라,
기쁨의 날이 오리니.
마음은 미래에 사는 것
현재는 슬픈 것:
모든 것은 순간적인 것, 지나가는 것이니
그리고 지나가는 것은 훗날 소중하게 되리니.
"""
w2 = Label(window,
          justify=LEFT,
          padx = 10,
          text=message).pack(side="left")
window.mainloop()
```



레이블의 색상과 폰트 변경하기

```
from tkinter import *

window = Tk()

Label(window,
                    text="Times Font 폰트와 빨강색을 사용합니다.",
                    fg = "red",
                    font = "Times 32 bold italic").pack()
Label(window,
                    text="Helvetica 폰트와 녹색을 사용합니다.",
                    fg = "blue",
                    bg = "yellow",
                    font = "Helvetica 32 bold italic").pack()
window.mainloop()
```

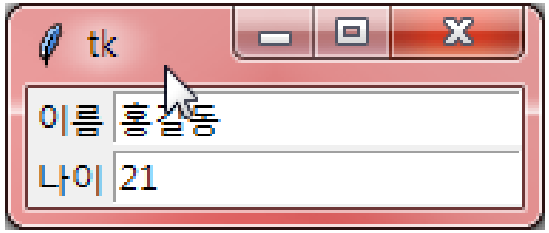
엔트리 위젯

전체적인 구조

```
w = Entry(parent , option, ... )
```

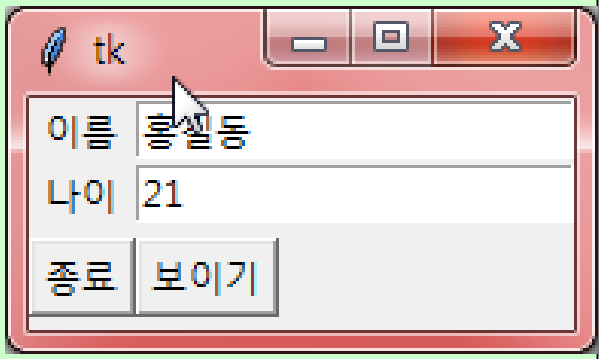
예제

```
from tkinter import *

window = Tk()
Label(window , text="이름").grid(row=0)
Label(window, text="나이").grid(row=1)

e1 = Entry(window)
e2 = Entry(window)

e1.grid(row=0, column=1)
e2.grid(row=1, column=1)

window.mainloop( )
```



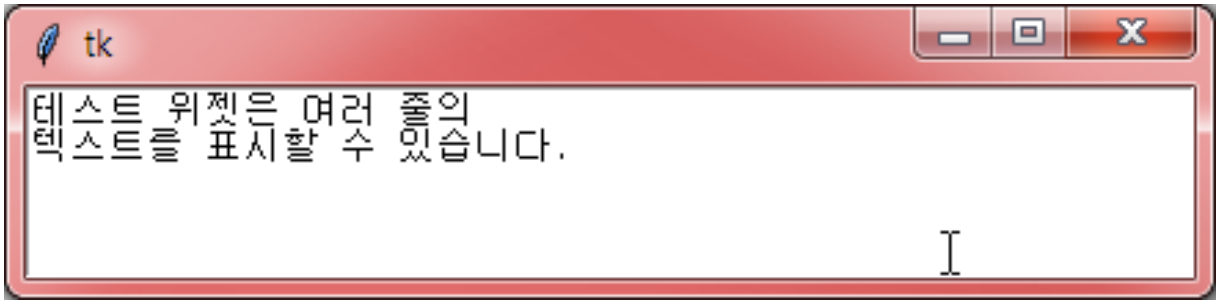
예제

```
from tkinter import *

def show():
   print("이름: %s\n나이: %s" % (e1.get(), e2.get()))

parent  = Tk()
Label(parent , text="이름").grid(row=0)
Label(parent, text="나이").grid(row=1)

e1 = Entry(parent)
e2 = Entry(parent)

e1.grid(row=0, column=1)
e2.grid(row=1, column=1)

Button(parent, text='보이기', command=show).grid(row=3, column=1,
sticky=W, pady=4)
Button(parent, text='종료', command=parent.quit).grid(row=3, column=0,
sticky=W, pady=4)

mainloop( )
```



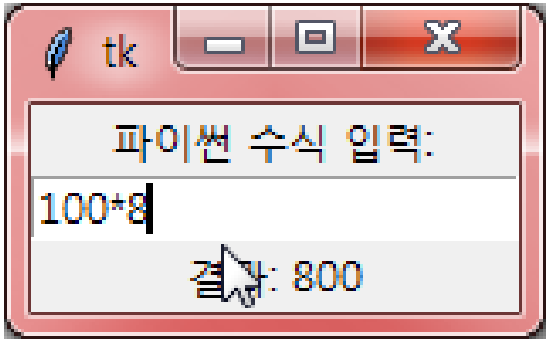
텍스트 위젯

```
from tkinter import *

window = Tk()
T = Text(window, height=5, width=60)
T.pack()
T.insert(END, "테스트 위젯은 여러 줄의\n텍스트를 표시할 수 있습니다.")
mainloop()
```



Lab: 계산기

- 수식을 텍스트로 입력하면 이것을 평가하고 그 결과를 출력할 수 있는 간단한 계산기를 작성하여 본다. 수식의 형식은 파이썬과 동일하여야 한다. **eval()** 함수를 사용하여 사용자가 입력한 수식을 계산할 수 있다.



Solution

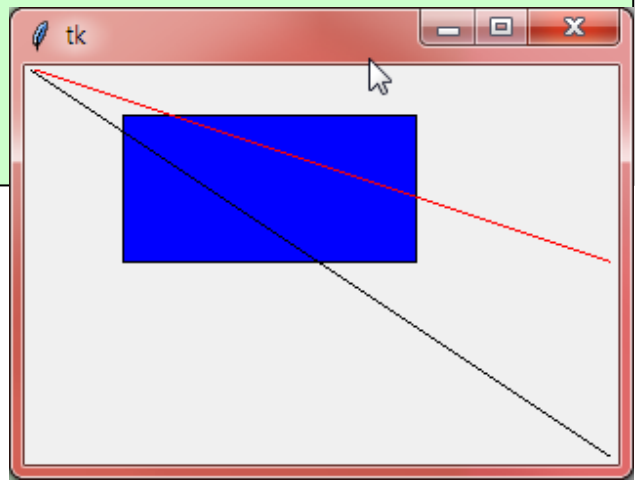
```
from tkinter import *
from math import *
def calculate(event):
    label.configure(text = "결과: " + str(eval(entry.get())))

window = Tk()

Label(window, text="파이썬 수식 입력:").pack()
entry = Entry(window)
entry.bind("<Return>", calculate)
entry.pack()

label = Label(window, text ="결과:")
label.pack()

w.mainloop()
```

tkinter를 이용한 그래픽

- **Canvas** 위젯을 이용하여 그래프를 그린다거나 그래픽 에디터를 작성할 수도 있고 많은 종류의 커스텀 위젯을 작성할 수 있다.

기본적인 구조

```
from tkinter import *

window = Tk()

w = Canvas(window, width=300, height=200)
w.pack()

w.create_rectangle(50, 25, 200, 100, fill="blue")
w.create_line(0, 0, 300, 200)
w.create_line(0, 0, 300, 100, fill="red")

mainloop()
```

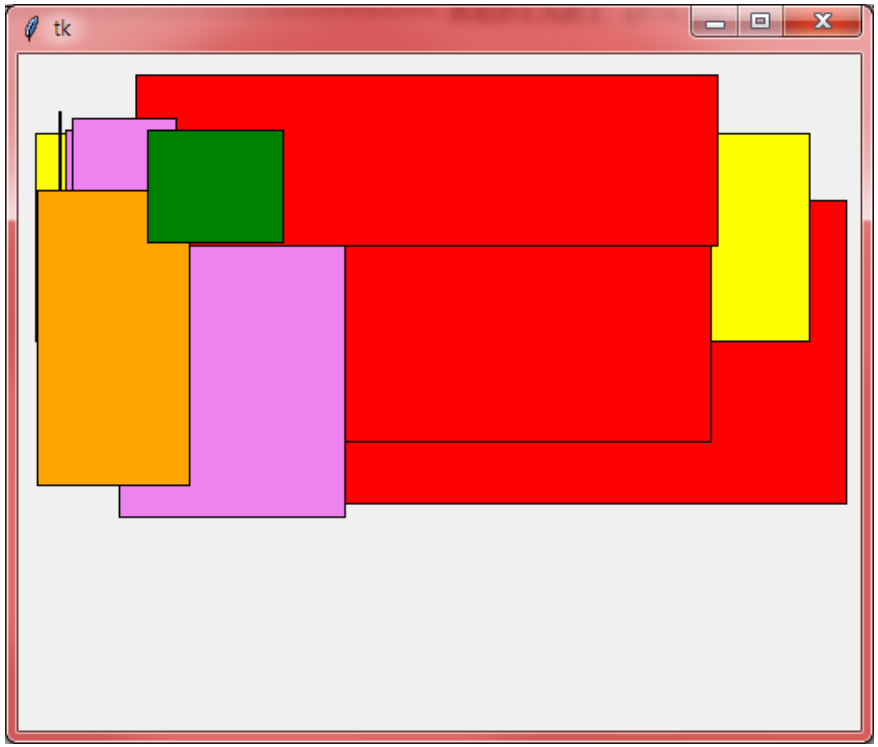


캔버스 항목

- 호(arc)
- 비트맵(bitmap, 내장 파일이나 XBM 파일 형식)
- 이미지(image, BitmapImage나 PhotoImage 객체)
- 직선(line)
- 타원(oval, 원이나 타원)
- 다각형(polygon)
- 사각형(rectangle)
- 텍스트(text)
- 윈도우(window)

Lab: 랜덤한 사각형 그리기

- 윈도우를 하나 만들고 여기에 랜덤한 크기의 사각형을 여러 개 그려보자. 위치도 랜덤이어야 하고 크기, 색상도 랜덤으로 하여 본다.

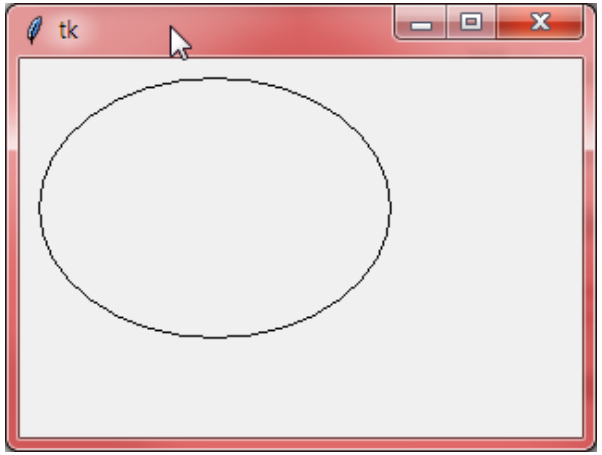


Solution

```
import random
from tkinter import *

window = Tk()
canvas = Canvas(window, width=500, height=400)
canvas.pack()
color = ["red", "orange", "yellow", "green", "blue", "violet"]

def draw_rect():
    x = random.randint(0, 500)
    y = random.randint(0, 400)
    w = random.randrange(100)
    h = random.randrange(100)
    canvas.create_rectangle(x, y, w, h, fill = random.choice(color))

for i in range(10):
    draw_rect()

window.mainloop()
```

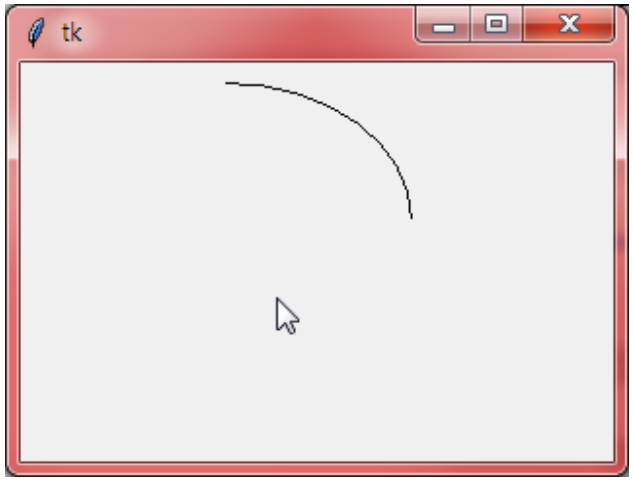
타원 그리기

```
from tkinter import *
window = Tk()
canvas = Canvas(window, width=300, height=200)
canvas.pack()
canvas.create_oval(10, 10, 200, 150)
window.mainloop()
```



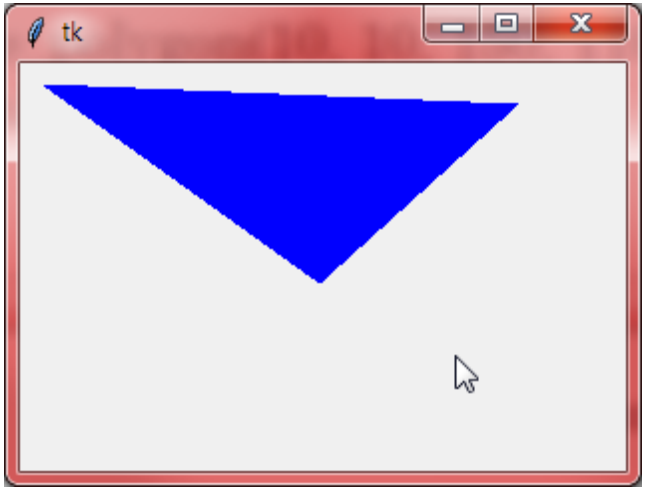
호 그리기

```
from tkinter import *
window = Tk()
canvas = Canvas(window, width=300, height=200)
canvas.pack()
canvas.create_arc(10, 10, 200, 150, extent=90, style=ARC)
window.mainloop()
```



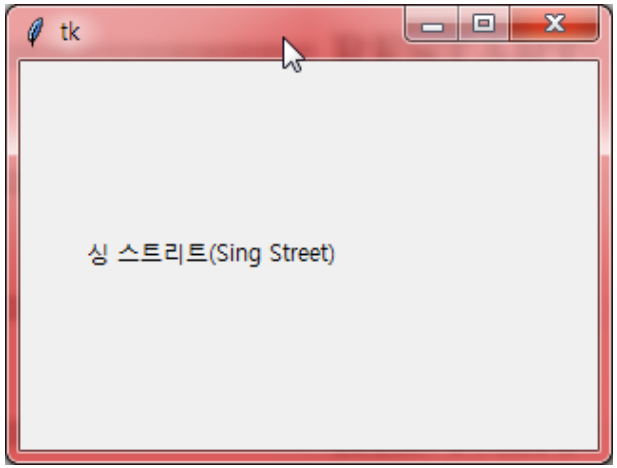
다각형 그리기

```
from tkinter import *
window = Tk()

canvas = Canvas(window, width=300, height=200)
canvas.pack()

canvas.create_polygon(10, 10, 150, 110, 250, 20, fill="blue")
window.mainloop()
```



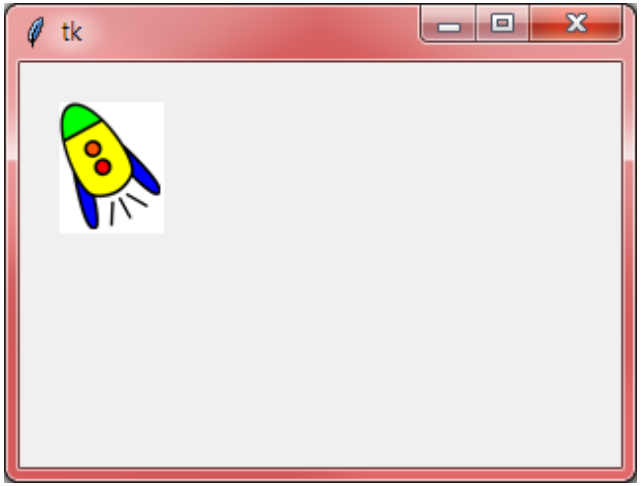
텍스트 그리기

```
from tkinter import *
window = Tk()
canvas = Canvas(window, width=300, height=200)
canvas.pack()
canvas.create_text(100, 100, text='싱 스트리트(Sing Street)')
mainloop()
```



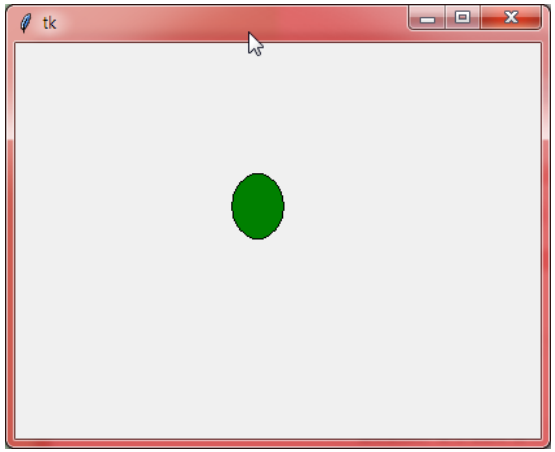
이미지 그리기

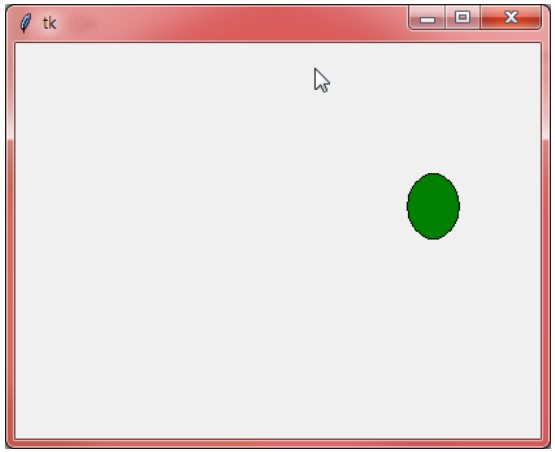
```
from tkinter import *
window = Tk()

canvas = Canvas(window,  width=300, height=200)
canvas.pack()

img = PhotoImage(file="D:\\starship.png")
canvas.create_image(20, 20, anchor=NW, image=img)

mainloop()
```



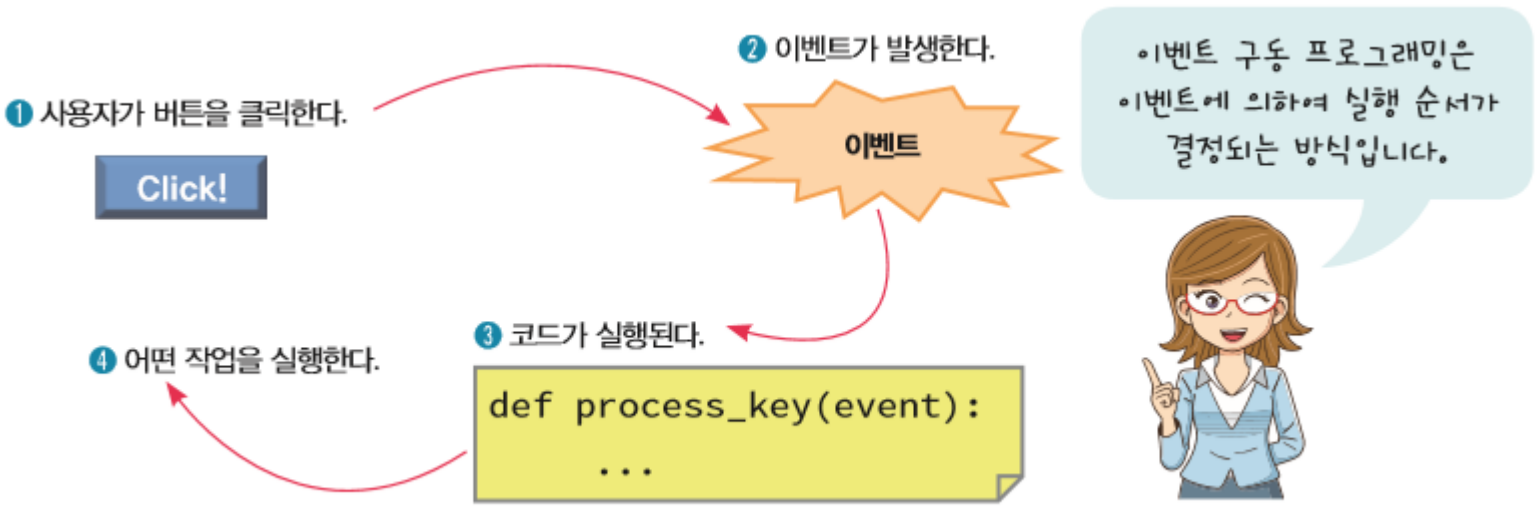
애니메이션을 만들어 보자.

```
import time
from tkinter import *
window = Tk()
canvas = Canvas(window, width=400, height=300)
canvas.pack()
id=canvas.create_oval(10, 100, 50, 150, fill="green")

for i in range(100):
   canvas.move(id, 3, 0)
   window.update()
   time.sleep(0.05)
```

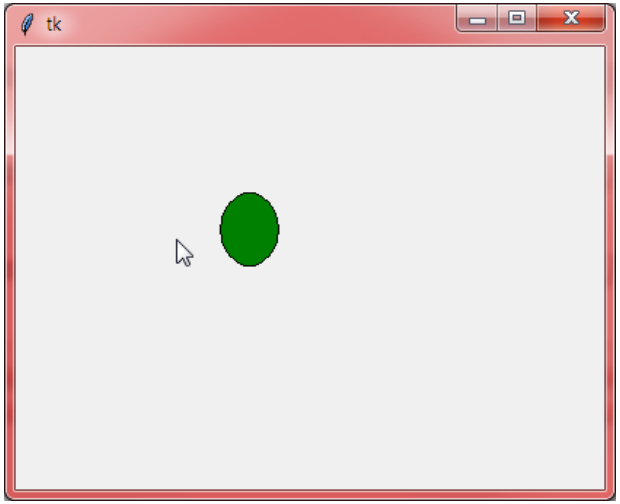


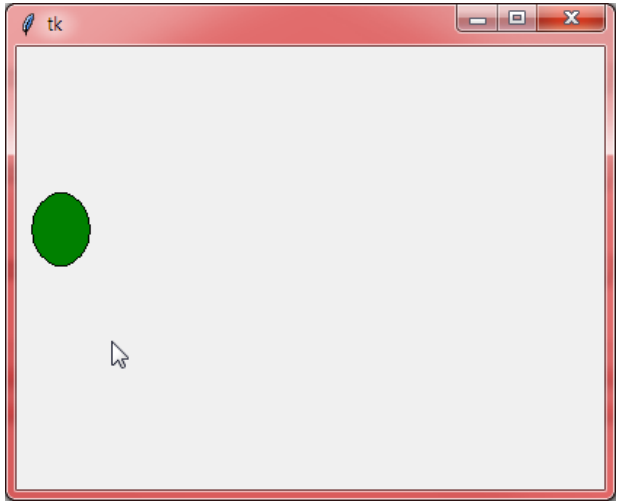


화살표 키로 공을 움직여 보자.



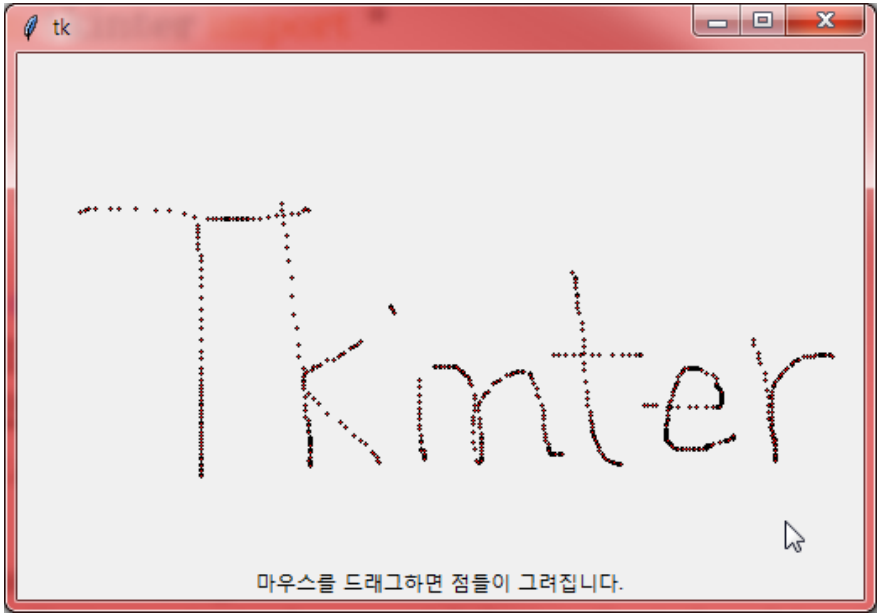
```
from tkinter import *
window = Tk()
canvas = Canvas(window, width=400, height=300)
canvas.pack()

id=canvas.create_oval(10, 100, 50, 150, fill="green")

def move_right(event):
    canvas.move(id, 5, 0)
canvas.bind_all('<KeyPress-Right>', move_right)
```





Lab: 마우스로 그림 그리기

- 다음과 같이 마우스를 움직여서 화면에 그림을 그리는 애플리케이션을 작성해보자.

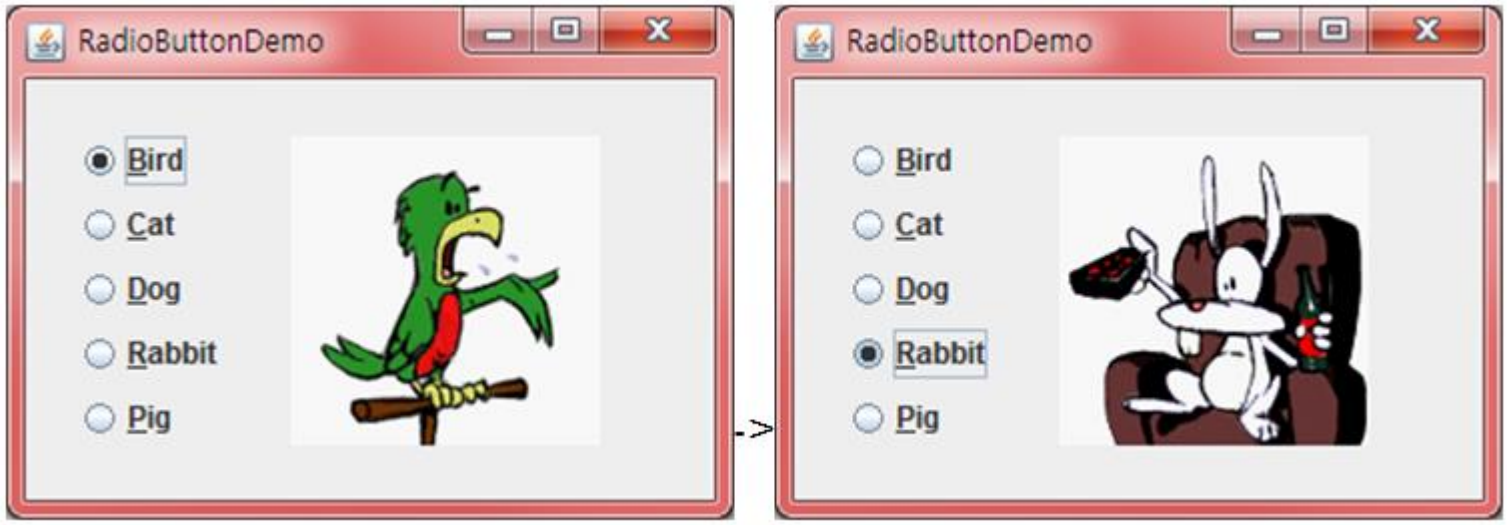


Solution

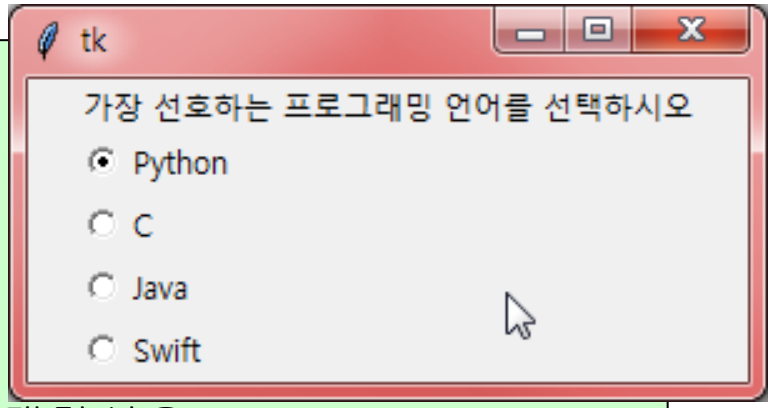
```
from tkinter import *

w = 500
h = 300

def drawDot( event ):
   x1, y1 = ( event.x - 1 ), ( event.y - 1 )
   x2, y2 = ( event.x + 1 ), ( event.y + 1 )
   canvas.create_oval( x1, y1, x2, y2, fill = "red" )

window = Tk()
canvas  = Canvas(window, width=w, height=h)
canvas.pack(expand = YES, fill = BOTH)
canvas.bind( "<B1-Motion>", drawDot )

message = Label( window, text = "마우스를 드래그하면 점들이 그려집니다."
)
message.pack( side = BOTTOM )

window.mainloop()
```

라디오 버튼

- 라디오 버튼(**radio button**)은 체크 박스와 비슷하지만 하나의 그룹 안에서는 한 개의 버튼만 선택할 수 있다는 점이 다르다.



라디오 버튼 예제

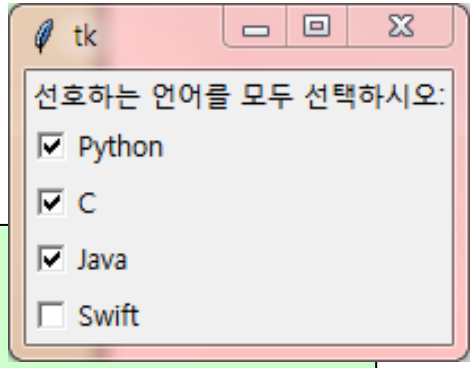



```
from tkinter import *

window = Tk()
choice = IntVar()

Label(window,
      text="가장 선호하는 프로그래밍 언어를 선택하시오",
      justify = LEFT,
      padx = 20).pack()
Radiobutton(window, text="Python", padx = 20, variable=choice,
value=1).pack(anchor=W)
Radiobutton(window, text="C", padx = 20, variable=choice,
value=2).pack(anchor=W)
Radiobutton(window, text="Java", padx = 20, variable=choice,
value=3).pack(anchor=W)
Radiobutton(window, text="Swift", padx = 20, variable=choice,
value=4).pack(anchor=W)

window.mainloop()
```

체크 박스

- 체크 박스(**check box**)란 사용자가 클릭하여서 체크된 상태와 체크되지 않은 상태 중의 하나로 만들 수 있는 위젯이다.

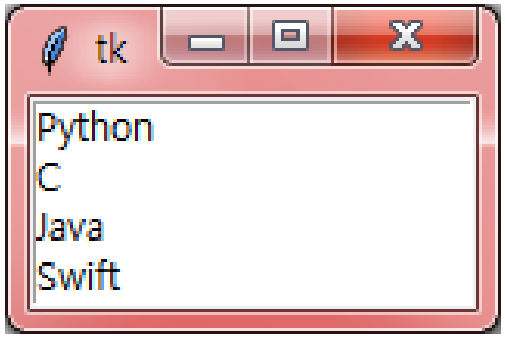
라디오 버튼 예제




```
from tkinter import *

window = Tk()
Label(window, text="선호하는 언어를 모두 선택하시오:").grid(row=0,
sticky=W)

value1 = IntVar()
Checkbutton(window, text="Python", variable=value1).grid(row=1, sticky=W)
value2 = IntVar()
Checkbutton(window, text="C", variable=value2).grid(row=2, sticky=W)
value3 = IntVar()
Checkbutton(window, text="Java", variable=value3).grid(row=3, sticky=W)
value4 = IntVar()
Checkbutton(window, text="Swift", variable=value4).grid(row=4, sticky=W)

window.mainloop()
```

리스트 박스

```
from tkinter import *

window = Tk()

lb = Listbox(window, height=4)
lb.pack()
lb.insert(END,"Python")
lb.insert(END,"C")
lb.insert(END,"Java")
lb.insert(END,"Swift")
```



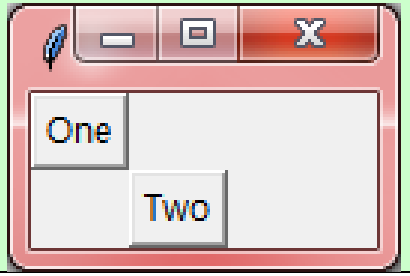
배치 관리자

- 배치 관리자는 컨테이너 안에 존재하는 위젯의 크기와 위치를 자동적으로 관리하는 객체이다.

격자 배치 관리자

- 격자 배치 관리자**(grid geometry manager)**는 위젯 (버튼, 레이블 등)을 테이블 형태로 배치한다.

```
from tkinter import *
window = Tk()

b1 = Button(window, text="One")
b2 = Button(window, text="Two")

b1.grid(row=0, column=0)
b2.grid(row=1, column=1)

window.mainloop()
```



압축 배치 관리자

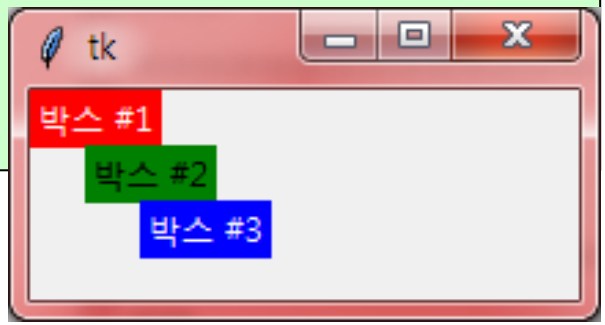
- 격자 배치 관리자(**grid geometry manager**)는 위젯 (버튼, 레이블 등)을 테이블 형태로 배치한다.

```
from Tkinter import *

window = Tk()

Label(window, text="박스 #1", bg="red", fg="white").pack()
Label(window, text="박스 #2", bg="green", fg="black").pack()
Label(window, text="박스 #3", bg="blue", fg="white").pack()

window.mainloop()
```

절대 위치 배치 관리자

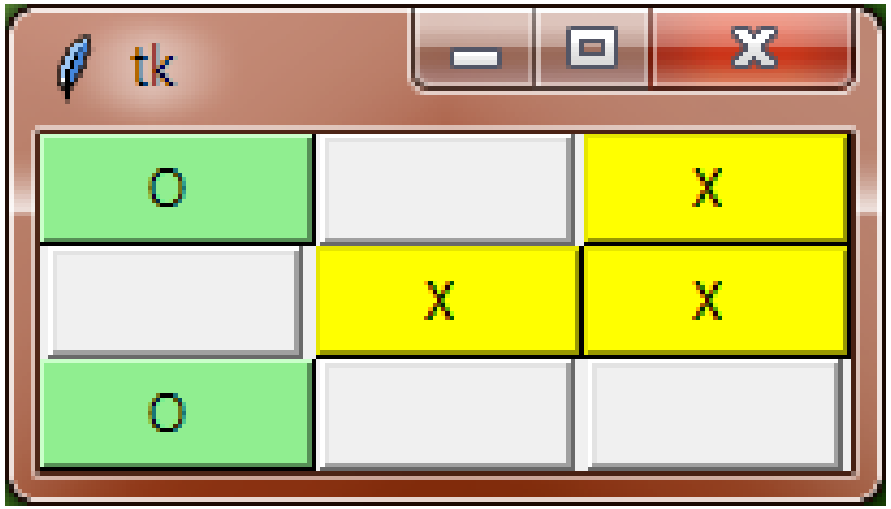
- 격자 배치 관리자(**grid geometry manager**)는 위젯 (버튼, 레이블 등)을 테이블 형태로 배치한다.

```
from tkinter import *

window = Tk()

w = Label(window, text="박스 #1", bg="red", fg="white")
w.place(x=0, y=0)
w = Label(window, text="박스 #2", bg="green", fg="black")
w.place(x=20, y=20)
w = Label(window, text="박스 #3", bg="blue", fg="white")
w.place(x=40, y=40)

window.mainloop()
```



Lab: 마우스로 그림 그리기

- **Tic-Tac-Toe**는 **3×3**칸을 가지는 게임판을 만들고, 경기자 **2**명이 동그라미 심볼**(O)**와 가위표 심볼**(X)**을 고른다. 경기자는 번갈아 가며 게임판에 동그라미나 가위표를 놓는다. 가로, 세로, 대각선으로 동일한 심볼을 먼저 만들면 승리하게 된다.



S

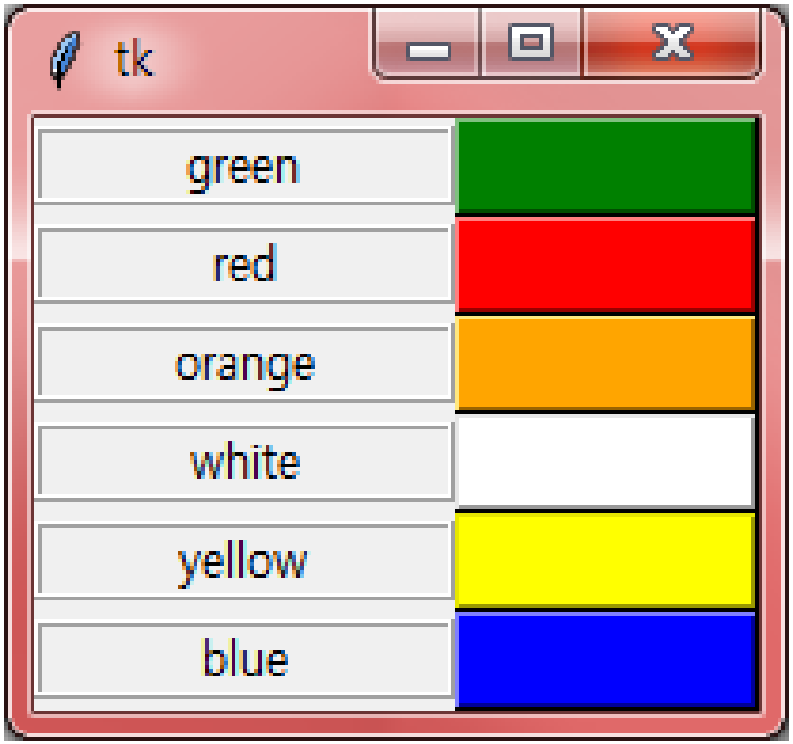
```
from tkinter import *

# i번째 버튼을 누를 수 있는지 검사한다. 누를 수 있으면 X나 O를 표시한다.
def checked(i):
    global player
    button = list[i]   # 리스트에서 I번째 버튼 객체를 가져온다.

                # 버튼이 초기상태가 아니면 이미 누른 버튼이므로 아무것도 하지
않고 리턴한다.
    if button["text"] != "            ":
        return
    button["text"] = "     " + player+"      "
    button["bg"] = "yellow"
    if player=="X":
        player = "O"
        button["bg"] = "yellow"
    else :
        player = "X"
        button["bg"] = "lightgreen"
```

Solution

```
window = Tk()                    # 윈도우를 생성한다.
player="X"                       # 시작은 플레이어 X이다.
list = []

# 9개의 버튼을 생성하여 격자 형태로 윈도우에 배치한다.
for i in range(9):
    b = Button(window, text="            ", command=lambda k=i: checked(k))
    b.grid(row=i//3, column=i%3)
    list.append(b)   # 버튼 객체를 리스트에 저장한다.

window.mainloop()
```

Lab: 마우스로 그림 그리기

- 격자 배치 관리자를 사용하여 레이블과 버튼을 배치한다. 색상의 개수만큼 반복하면서 레이블과 버튼을 생성하고 격자형태로 배치하면 된다.



Solution

```
from tkinter import *

약간의 3차원 효과를 낸다.

window = Tk()
colors = ['green', 'red', 'orange','white','yellow','blue']

r = 0
for c in colors:
    Label(window, text=c, relief=RIDGE, width=15).grid(row=r, column=0)
    Button(window, bg=c, width=10).grid(row=r, column=1)
    r = r + 1

window.mainloop()
```

Lab: 스톱워치 만들기

- 레이블을 사용하여 간단한 스톱워치를 작성하여 보자. 시작 버튼을 누르면 시작되고 중지 버튼을 누르면 스톱워치가 중지된다.

S

```
import tkinter as tk

def startTimer():
    if (running):
        global timer
        timer += 1
        timeText.configure(text=str(timer))
    window.after(10, startTimer)

def start():
    global running
    running = True

def stop():
    global running
    running = False

running = False
```

Solution

```
window = tk.Tk()

timer = 0

timeText = tk.Label(window, text="0", font=("Helvetica", 80))
timeText.pack()

startButton = tk.Button(window, text='시작', bg="yellow", command=start)
startButton.pack(fill=tk.BOTH)

stopButton = tk.Button(window, text='중지', bg="yellow", command=stop)
stopButton.pack(fill=tk.BOTH)

startTimer()
window.mainloop()
```

Lab: 스톱워치 만들기

- 파이썬을 이용하여 버튼을 가지는 계산기를 작성하여 보자. 적절한 배치 관리자를 선택하여 사용하라.

S
```
from tkinter import *

def click(key):
    if key == '=':                  # “=” 버튼이면 수식을 계산하여 결과를 표시한다.
        try:
            result = eval(entry.get())
            entry.delete(0, END)
            entry.insert(END, str(result))
        except:
            entry.insert(END, "오류!")
    elif key == 'C':
        entry.delete(0, END)
    else:
        entry.insert(END, key)

window = Tk()
window.title("간단한 계산기")
```

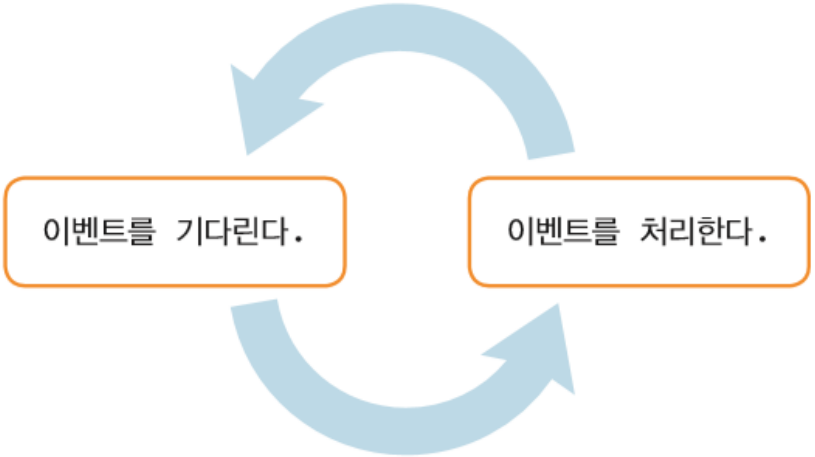
S

```
buttons = [
'7',  '8',  '9',  '+',  'C',
'4',  '5',  '6',  '-',  '  ',
'1',  '2',  '3',  '*',  '  ',
'0',  '.',  '=',  '/',  '   ' ]

# 반복문으로 버튼을 생성한다.
i=0
for b in buttons:
    cmd = lambda x=b: click(x)
    b = Button(window,text=b,width=5,relief='ridge',command=cmd)
    b.grid(row=i//5+1,column=i%5)
    i += 1

# 엔트리 위젯은 맨 윗줄의 5개의 셀에 걸쳐서 배치된다.
entry = Entry(window, width=33, bg="yellow")
entry.grid(row=0, column=0, columnspan=5)

window.mainloop()
```

이벤트 처리

전체적인 구조

위젯 .bind(이벤트 지정자 , 이벤트 처리 함수)



마우스 이벤트 처리

```
window = Tk()

def callback(event):
    print(event.x, event.y, "에서 마우스 이벤트 발생")

frame = Frame(window, width=100, height=100)
frame.bind("<Button-1>", callback)
frame.pack()

window.mainloop()
```

```
32 44 에서 마우스 이벤트 발생
6 52 에서 마우스 이벤트 발생
```

키보드 이벤트 처리

```
from tkinter import *

window = Tk()

def key(event):
    print ( repr(event.char), "가 눌렸습니다. ")

def callback(event):
    frame.focus_set()
    print(event.x, event.y, "에서 마우스 이벤트 발생")

frame = Frame(window, width=100, height=100)
frame.bind("<Key>", key)
frame.bind("<Button-1>", callback)
frame.pack()

window.mainloop()
```

```
66 31 에서 마우스 이벤트 발생
'k' 가 눌렸습니다.
```

이벤트 지정자

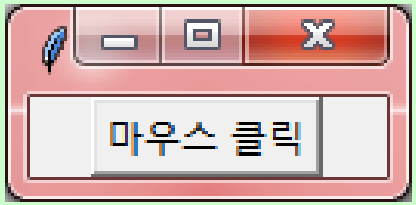
전체적인 구조

< 수식어 – 타입 – 세부사항 >

- *<Button-1>*
- *<B1-Motion>*
- *<ButtonRelease-1>*
- *<Double-Button-1>*
- *<Enter>*
- *<Leave>*
- *<FocusIn>*
- *<FocusOut>*
- *<Return>*
- *<Key>*
- *a*
- *<Shift-Up>*
- *<Configure>*

마우스 버튼 이벤트 처리

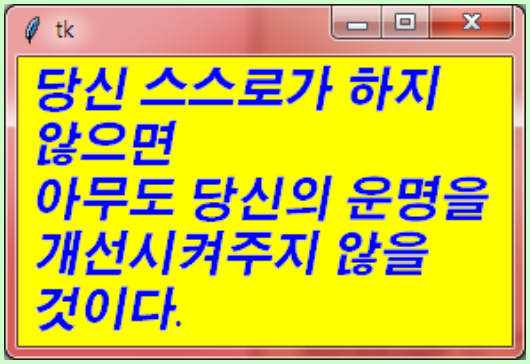
```
from tkinter import *

def sleft(event):
    print("단일 클릭, 왼쪽 버튼")
def dleft(event):
    print("더블 클릭, 왼쪽 버튼")

widget = Button(None, text='마우스 클릭')
widget.pack()

widget.bind('<Button-1>', sleft)
widget.bind('<Double-1>', dleft)

widget.mainloop()
```




```
단일 클릭, 왼쪽 버튼
단일 클릭, 왼쪽 버튼
더블 클릭, 왼쪽 버튼
```

마우스 모션 이벤트 처리

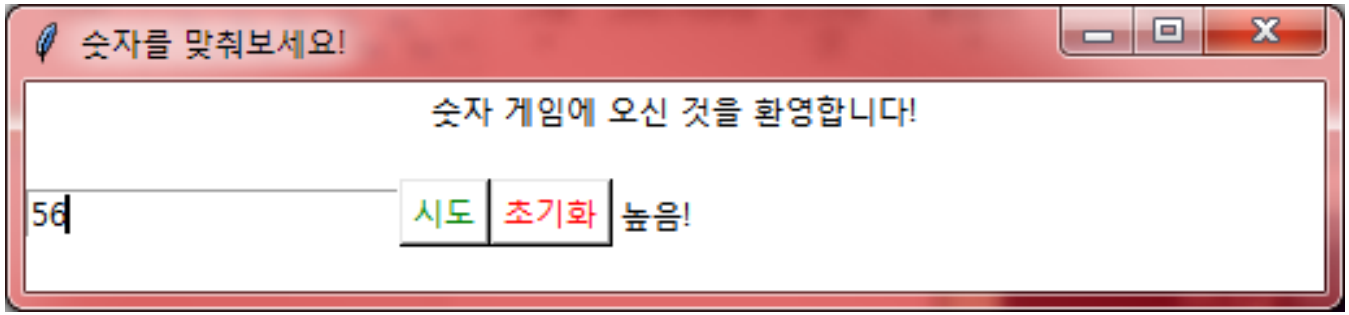
```
from tkinter import *

def motion(event):
  print("마우스 위치: (%s %s)" % (event.x, event.y))
  return

window = Tk()
message = """당신 스스로가 하지 않으면
아무도 당신의 운명을
개선시켜주지 않을 것이다. """

msg = Message(window, text = message)
msg.config(bg='yellow',fg='blue',  font="times 20 italic")
msg.bind('<Motion>',motion)
msg.pack()
window.mainloop()
```




```
마우스 위치: (274 45)
마우스 위치: (271 55)
마우스 위치: (270 69)
```

Lab: 숫자 추측 게임

- 사용자가 컴퓨터가 생성한 숫자(1부터 100사이의 난수)를 알아맞히는 게임을 그래픽 사용자 인터페이스를 사용하여 제작해보자.



Solution

```
from tkinter import *
import random

answer = random.randint(1,100)

def guessing():
        guess = int(guessField.get())

        if guess > answer:
                msg = "높음!"
        elif guess < answer:
                msg = "낮음!"
        else:
                msg = "정답!"

        resultLabel["text"] = msg
        guessField.delete(0, 5)
```

Solution

```
def reset():
        global answer
        answer = random.randint(1,100)
        resultLabel["text"] = "다시 한번 하세요!"

window = Tk()
window.configure(bg="white")
window.title("숫자를 맞춰보세요!")
window.geometry("500x80")

titleLabel = Label(window, text="숫자 게임에 오신 것을 환영합니다!",
bg="white")
titleLabel.pack()
```

Solution

```
guessField = Entry(window)
guessField.pack(side="left")
tryButton = Button(window, text="시도", fg="green", bg="white",
command=guessing )
tryButton.pack(side="left")

resetButton = Button(window, text="초기화", fg="red", bg="white",
command=reset)
resetButton.pack(side="left")
resultLabel = Label(window, text="1부터 100사이의 숫자를 입력하시오.",
bg="white")
resultLabel.pack(side="left")

window.mainloop()
```

헥심 정리

- **tkinter**에서는 먼저 루프 윈도우를 생성하고 레이블이나 버튼을 생성할 때 첫 번째 인수로 윈도우를 넘기면 된다.
- 파이썬은 **3**종류의 배치 관리자를 제공한다. 압축**(pack)** 배치 관리자, 격자**(grid)** 배치 관리자. 절대**(place)** 배치 관리자가 바로 그것이다.
- 위젯에 이벤트를 처리하는 함수를 연결하려면 **bind()** 메소드를 사용한다. 예를 들면 **widget.bind('<Button-1>', sleft)**와 같이 하면 된다.

Q & A