

[Python]



Python으로 배우는

소프트웨어 원리

Chapter 11. 그래픽 프로그래밍

목차

1. 사용자 인터페이스
2. GUI 프로그래밍
3. tkinter의 활용

01

사용자 인터페이스

01. 사용자 인터페이스

[사용자 인터페이스의 다양한 활용]

- 사람과 컴퓨터의 상호작용에는 유연한 인터페이스가 필요하다.
- 대표적인 인터페이스는 키보드와 마우스가 있으며 종류는 점점 다양해지고 있다.



01. 사용자 인터페이스

I. 사용자 인터페이스의 개념

- 사람과 컴퓨터가 상호작용하는 방법을 의미하며, 편리하고 효율적인 의사소통이 목적
- 사용자 인터페이스를 위해서는 키보드나 마우스, 스크린 등 다양한 도구와 방법이 필요
- 사용자 인터페이스는 원하는 기능을 쉽게 찾고 사용하기 쉬워야 하며, 간결하고 명확하게 명령이 전달되어야 함

01. 사용자 인터페이스

I. 사용자 인터페이스의 개념

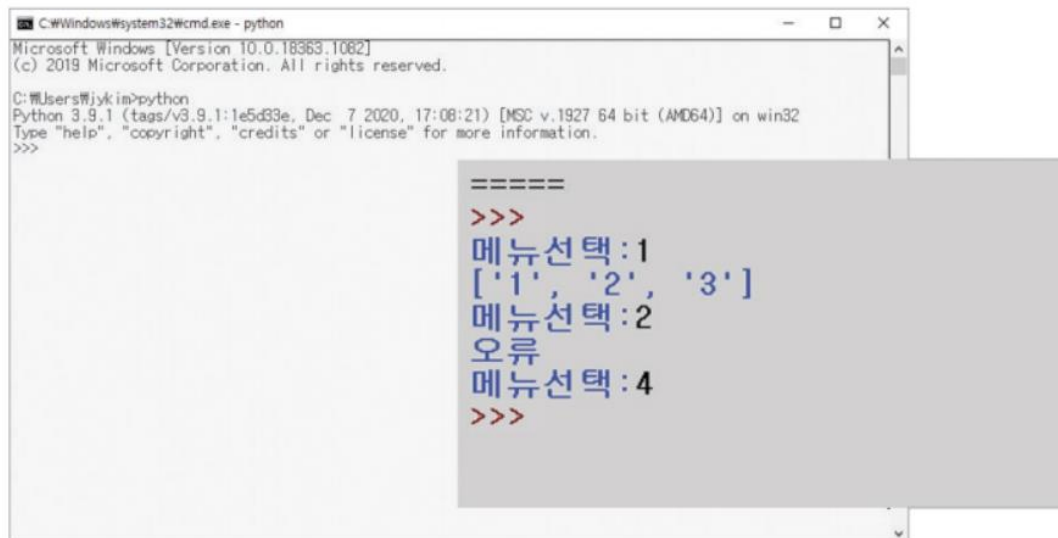
- 사람과 컴퓨터가 상호작용하는 방법을 의미하며, 편리하고 효율적인 의사소통이 목적
- 사용자 인터페이스를 위해서는 키보드나 마우스, 스크린 등 다양한 도구와 방법이 필요
- 사용자 인터페이스는 원하는 기능을 쉽게 찾고 사용하기 쉬워야 하며, 간결하고 명확하게 명령이 전달되어야 함

01. 사용자 인터페이스

I. 사용자 인터페이스의 종류

■ CUI(Character UI) 혹은 CLI(Command-line UI) 방식

- 키보드를 사용한 텍스트 입력 방식으로 명령을 전달하여 사용자가 원하는 동작이나 프로그램을 실행
- DOS 셸이나 프로그래밍에 활용하고 있는 파이썬 셸 등에서는 명령어를 직접 입력하는 CUI 방식을 사용



```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [Version 10.0.18363.1062]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\jykim>python
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
=====
>>>
메뉴선택:1
['1', '2', '3']
메뉴선택:2
오류
메뉴선택:4
>>>
```

그림 11-2 CUI 방식

01. 사용자 인터페이스

I. 사용자 인터페이스의 종류

▪ GUI(Graphic UI) 방식

- 화면의 특정 지점을 클릭하거나 드래그하는 등의 방식으로 사용자의 명령을 컴퓨터에게 전달
- 스마트폰에서는 스크린 터치와 같은 직관적이고 편리한 여러 가지 인터페이스 기술이 활용

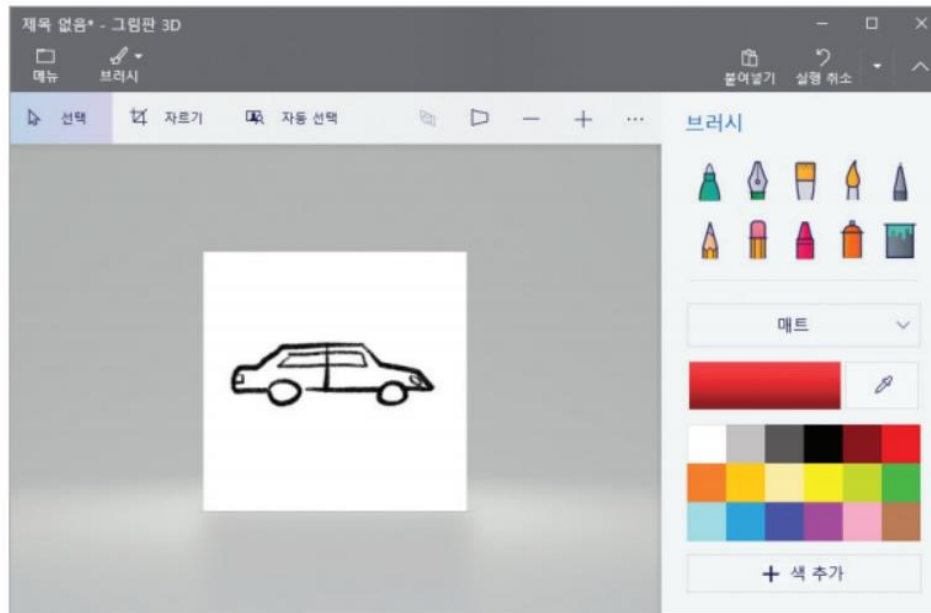


그림 11-3 GUI 방식

01. 사용자 인터페이스

I. 사용자 인터페이스의 종류

▪ NUI(Natural UI) 방식

- 사람의 음성과 표정, 몸이나 손가락의 움직임 등을 인식하여 사용자가 원하는 명령을 입력
- 실행한 결과를 표현할 때에도 사람의 자연스러운 표현 방법처럼 출력하는 방식
- 얼굴 표정, 심지어 뇌파나 맥박, 체온 같은 생체 정보를 측정하여 사람의 기분이나 상태에 맞추어 필요한 동작을 실행하는 진보된 형태의 인터페이스 기술

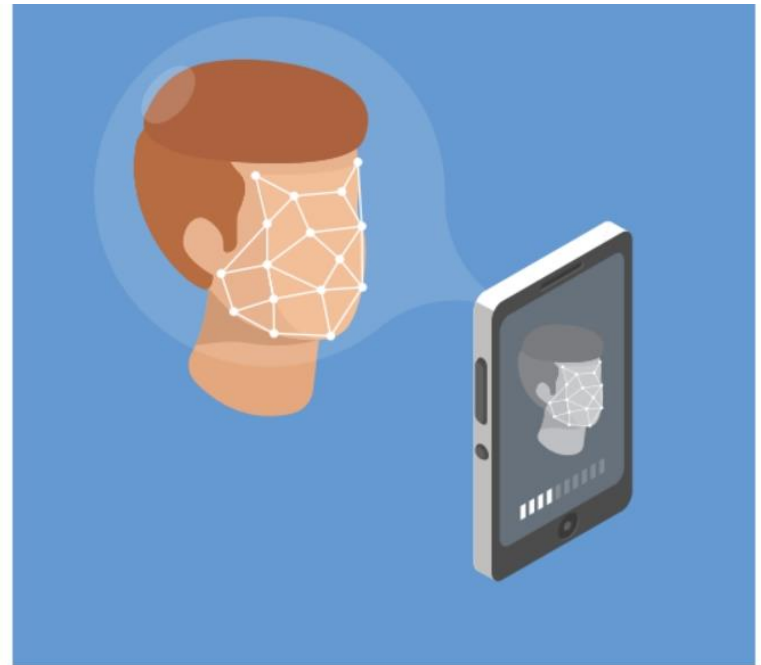


그림 11-4 NUI 방식

02

GUI 프로그래밍

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

- tkinter를 사용하면 GUI 프로그램을 작성할 수 있음

- ① 먼저 import 문으로 tkinter 모듈 사용을 선언하고, Tk() 객체를 생성하면 root라고 불리는 기본 윈도우가 생성

```
import tkinter # tkinter 모듈 사용 선언  
윈도우명 = tkinter.Tk() # 기본 윈도우 생성
```

=

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

```
>>> from tkinter import *  
>>> w = Tk()
```

→

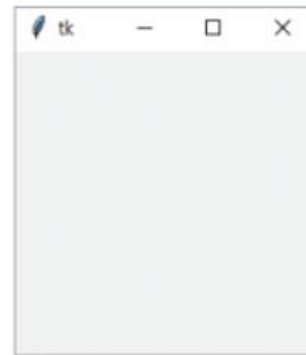


그림 11-5 기본 윈도우 생성

- Tk()는 윈도우 객체를 생성 후 핸들(handle) 값을 반환
- 핸들 값은 객체에 접근 할 수 있는 태그(tag) 또는 티켓의 일종

- ❖ Tkinter는 “Python interface to Tcl/Tk”로 Tk와 인터페이스(interface)의 합성어에서 유래
- ❖ Tkinter는 Tcl/Tk(Tool Command Language/Toolkit)를 파이썬에 사용할 수 있도록 한 Lightweight GIU 모듈
- ❖ Tk는 크로스 플랫폼에 사용하는 일종의 GUI 툴킷
- ❖ 크로스 플랫폼(영어: cross-platform) 또는 멀티 플랫폼(영어: multi-platform)은 컴퓨터 프로그램, 운영 체제, 컴퓨터 언어, 프로그래밍 언어, 컴퓨터 소프트웨어 등이 여러 종류의 컴퓨터 플랫폼에서 동작할 수 있다는 것을 뜻하는 용어

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

- ② 생성된 윈도우 안에 위젯이라고 불리는 구성 요소를 추가

```
>>> bt = Button(w, text="버튼을 누르세요.")  
>>> bt.pack()    # 윈도우에 버튼을 표시
```



그림 11-6 버튼 생성

- ③ 닫기()를 눌러 윈도우를 종료할 때까지 사용자의 키 입력이나 마우스 클릭 등의 이벤트 처리를 위해 대기

```
>>> w.mainloop()    # 이벤트 처리를 위한 대기
```

✓ **TIP** 현재 만든 버튼에는 이벤트 처리가 연결되지 않아 버튼을 눌러도 아무런 동작도 하지 않음

```
from tkinter import *  
w = Tk()  
bt = Button(w, text="눌러보세요.")  
bt.pack()  
bt.config(text="Press key to down")
```

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

실습 11-1

윈도우 창에 위젯 표시하기

code11-01.py

- 표시되는 순서대로 레이블(Label()), 엔트리(Entry()), 버튼(Button()) 위젯을 만들어 윈도우에 배치하고 이벤트 처리를 위해 대기

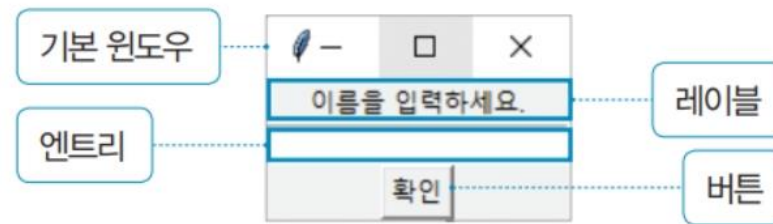


그림 11-7 실행 결과

- ① 모듈 사용을 선언하고 기본 윈도우를 생성

```
from tkinter import *
```

```
w = Tk()           # 기본 윈도우 만들기
```

- ② 레이블, 엔트리(입력 상자), 버튼 위젯을 생성하여 윈도우에 추가

```
lb = Label(w, text = "이름을 입력하세요.")   # 레이블 만들기
```

```
en = Entry(w)                                # 엔트리 만들기
```

```
bt = Button(w, text = "확인")                 # 버튼 만들기
```

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

실습 11-1

윈도우 창에 위젯 표시하기

code11-01.py

- ③ 위젯을 pack()으로 배치하면 기술한 순서대로 위에서부터 표시

```
lb.pack()           # 레이블 표시하기  
en.pack()           # 엔트리 표시하기  
bt.pack()           # 버튼 표시하기
```

- ④ 이벤트가 발생하면 처리하기 위해 대기

```
w.mainloop()        # 이벤트 처리를 위해 대기
```

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

실습 11-1

윈도우 창에 위젯 표시하기

code11-01.py

- ⑤ 단계별로 작성한 코드를 파이썬 파일로 저장하고 실행

```
01 from tkinter import *
02
03 w = Tk()                                # 기본 윈도우 만들기
04
05 lb = Label(w, text = "이름을 입력하세요.") # 레이블 만들기
06 en = Entry(w)                            # 엔트리 만들기
07 bt = Button(w, text = "확인")            # 버튼 만들기
08
09 lb.pack()                                # 위젯 표시하기
10 en.pack()
11 bt.pack()
12
13 w.mainloop()                            # 이벤트 처리를 위해 대기
```

02. GUI 프로그래밍

I. tkinter 모듈의 개념과 사용법

[실습] 윈도우 창 만들기

```
from tkinter import *  
  
w = Tk()                #기본 윈도우 객체 만들기  
  
w.title("Pythone Tkinter")    #윈도우 타이틀("제목")  
w.geometry("200x200+100+100")  #윈도우 크기 및 기준점("너비x높이+x좌표+y좌표")  
w.resizable(False, False)     #창 크기 조절 가능 여부(상하, 좌우)  
  
w.mainloop()             #이벤트 처리를 위해 대기
```


02. GUI 프로그래밍

II. 다양한 위젯의 사용

- 윈도우를 구성하는 레이블, 버튼, 엔트리나 메뉴 등이 위젯
- 메모장과 같은 응용 프로그램을 실행하면, 화면에 표시되는 윈도우 안에 많은 위젯이 포함

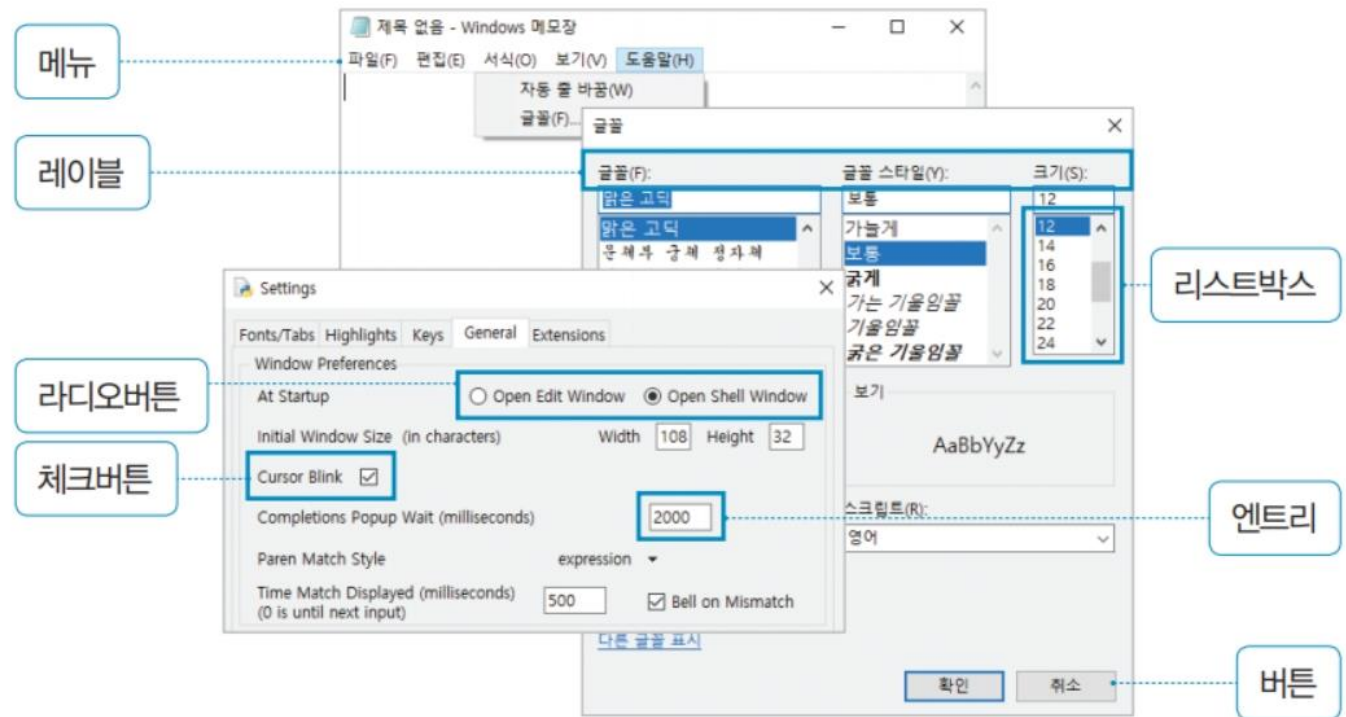


그림 11-8 윈도우에 표시되는 다양한 위젯

02. GUI 프로그래밍

II. 다양한 위젯의 사용

여기서 잠깐

위젯의 사용 가능한 속성 찾기

- 위젯 객체를 생성하고 `configure()` 또는 `config()` 메소드를 실행하면 사용할 수 있는 속성의 종류를 표시

```
>>> from tkinter import *
>>> bt = Button()
>>> bt.configure()                                # 버튼의 모든 속성 검색
{'activebackground': ('activebackground', 'activeBackground', 'Foreground', <string
object: 'SystemButtonFace'>, 'SystemButtonFace'), 'activeforeground':
('activeforeground', 'activeForeground', 'Background', ...
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

표 11-1 tkinter 모듈의 위젯 종류

위젯명	설명	위젯명	설명
Button	클릭 이벤트를 만들 수 있는 버튼	Menu	메뉴 항목을 구성할 때 사용
Canvas	선과 점으로 그림을 그리는 데 사용	Menubutton	기본 윈도우에 메뉴 버튼을 만들 때 사용
Checkbutton	여러 개의 항목을 동시에 선택할 수 있는 옵션에 사용	Message	크기 축소가 가능하고 레이블처럼 텍스트를 표시
Entry	한 줄로 텍스트를 입력하는 입력 상자	PanedWindow	위젯을 묶을 수 있는 컨테이너로 크기 조절이 가능
Frame	기본 윈도우 하위에서 위젯을 묶을 수 있는 컨테이너	Radiobutton	여러 개의 항목 중 하나를 선택하는 옵션에 사용
Label	텍스트나 이미지를 표시할 때 사용	Scale	숫자 조정을 위한 슬라이더를 표시
LabelFrame	제목을 추가한 프레임 위젯	Scrollbar	위젯에 스크롤바를 추가할 때 사용
ListBox	목록에서 하나를 선택하는 데 사용	Spinbox	화살표를 클릭하여 범위 내의 숫자를 선택하거나 입력
		Text	여러 줄의 텍스트를 표시할 때 사용
		Toplevel	새 윈도우를 추가로 만들 때 사용하는 컨테이너

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 속성

- 위젯에는 배경색이나 글자색, 크기와 커서 모양 등 여러 가지 속성 설정이 가능
- 위젯의 속성은 위젯을 생성할 때 설정할 수도 있고, 위젯을 생성하고 나서 필요할 때 `config()` 메소드로 변경할 수도 있음

```
# 위젯을 생성할 때 속성 설정
```

```
bt = Button(컨테이너, 속성1 = 값1, ...)
```

```
# 위젯을 생성한 후 속성 설정, 변경
```

```
bt.config(속성1 = 값1, 속성2 = 값2, ...)
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-2

위젯 속성 설정하기

code11-02.py



그림 11-9 실행 결과

① 프로그램에 필요한 위젯 속성

표 11-2 프로그램에 필요한 위젯 속성

속성	설명	사용 예
bd, border	테두리 두께	bd=5
bg, background	배경색	bg="gray"
cursor	커서 모양	cursor="circle"(arrow(기본), cross, pirate, plus)
fg, foreground	글자색	fg="white"
font	글꼴	font="Tahoma"
text	표시할 내용	text="확인"

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-2

위젯 속성 설정하기

code11-02.py

- ② 기본 윈도우를 생성하고, 속성을 설정한 위젯을 만들어 윈도우에 표시

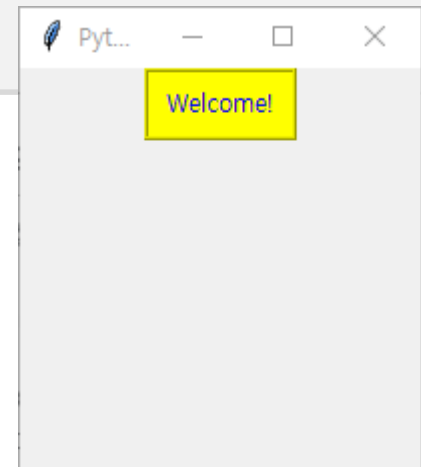
```
01 from tkinter import *
02
03 w = Tk()
04
05 lb = Label(w, text="이름을 입력하세요.", bg="gray", fg="white")      # 레이블 생성
06 en = Entry(w, bd = 5, cursor="circle")                             # 엔트리 생성
07 bt = Button(w, text="확인")                                         # 버튼 생성
08 bt.config(font="Tahoma", fg="blue")                                # 버튼 속성 변경
09 lb.pack()                                                           # 위젯 배치
10 en.pack()
11 bt.pack()
12
13 w.mainloop()
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

[실습] Label 만들기

```
from tkinter import *  
w = Tk()                #기본 윈도우 객체 만들기  
##중간 생략##  
  
lb1 = Label(w, text="Welcome!", width=10, height=2, bg="yellow", fg="blue", relief="ridge")  
lb1.pack()  
  
w.mainloop()           #이벤트 처리를 위해 대기
```



02. GUI 프로그래밍

II. 다양한 위젯의 사용

Label 파라미터

이름	의미	기본값	속성
text	라벨에 표시할 문자열	-	-
textvariable	라벨에 표시할 문자열을 가져올 변수	-	-
anchor	라벨안의 문자열 또는 이미지의 위치	center	n, ne, e, se, s, sw, w, nw, center
justify	라벨의 문자열이 여러 줄 일 경우 정렬 방법	center	center, left, right
wrlength	자동 줄내림 설정 너비	0	상수

이름	의미	기본값	속성
bitmap	라벨에 포함할 기본 이미지	-	info, warning, error, question, questhead, hourglass, gray12, gray25, gray50, gray75
image	라벨에 포함할 임의의 이미지	-	-
compound	라벨에 문자열과 이미지를 동시에 표시할 때 이미지의 위치	none	bottom, center, left, none, right, top
font	라벨의 문자열 글꼴 설정	TkDefaultFont	font
cursor	라벨의 마우스 커서 모양	-	커서 속성

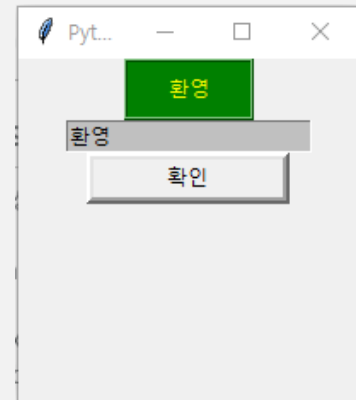
이름	의미	기본값	속성
width	라벨의 너비	0	상수
height	라벨의 높이	0	상수
relief	라벨의 테두리 모양	flat	flat, groove, raised, ridge, solid, sunken
borderwidth=bd	라벨의 테두리 두께	2	상수
background=bg	라벨의 배경 색상	SystemButtonFace	color
foreground=fg	라벨의 문자열 색상	SystemButtonFace	color
padx	라벨의 테두리와 내용의 가로 여백	1	상수
pady	라벨의 테두리와 내용의 세로 여백	1	상수

02. GUI 프로그래밍

II. 다양한 위젯의 사용

[실습] Entry, Button 만들기

```
from tkinter import *  
  
w = Tk()                #기본 윈도우 객체 만들기  
##중간 생략##  
  
def btn1_call():  
    instr = en1.get()  
    lb1.config(text=instr, bg="green", fg="yellow")  
    en1.config(bg="silver")  
  
en1 = Entry(w)  
en1.pack()  
bt1 = Button(w, text="확인", width=15, borderwidth=4, cursor="hand2")  
bt1.config(command=btn1_call)  
bt1.pack()
```



02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ Button 파라미터

이름	의미	기본값	속성
takefocus	Tab 키를 이용하여 위젯 이동 허용 여부	True	Boolean
command	버튼이 active 상태일 때 실행하는 메서드(함수)	-	메서드, 함수
repeatdelay	<u>버튼이 눌러진 상태에서 command 실행까지의 대기 시간</u>	0	상수(ms)
repeatinterval	<u>버튼이 눌러진 상태에서 command 실행의 반복 시간</u>	0	상수(ms)

이름	의미	기본값	속성
state	상태 설정	normal	normal , active, disabled
activebackground	active 상태일 때 버튼의 배경 색상	SystemButtonFace	color
activeforeground	active 상태일 때 버튼의 문자열 색상	SystemButtonText	color
disabledforeground	disabled 상태일 때 버튼의 문자열 색상	SystemDisabledText	color

이름	의미	기본값	속성
highlightcolor	버튼이 선택되었을 때 색상	SystemWindowFrame	color
highlightbackground	버튼이 선택되지 않았을 때 색상	SystemButtonFace	color
highlightthickness	버튼이 선택되었을 때 두께 (두께 설정)	0	상수

02. GUI 프로그래밍

II. 다양한 위젯의 사용

여기서 잠깐 위젯의 속성 변경하기

- 위젯을 생성하고 윈도우에 표시한 이후에도 config() 메소드를 사용하여 속성을 변경할 수 있음

```
...  
05 lb = Label(w, text="이름을 입력하세요.", bg="gray", fg="white") # 레이블 생성  
...  
09 lb.pack() # 레이블 표시  
...  
13 img = PhotoImage(file="coffee.gif") # 이미지 객체 생성  
14 lb.config(image=img) # 레이블에 이미지를 표시하도록 변경  
15 bt.config(text="커피 쿠폰 당첨") # 버튼의 텍스트 변경  
16  
17 w.mainloop()
```



그림 11-10 레이블과 버튼의 속성 변경

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

- 여러 항목 중 하나를 선택할 때 사용하는 라디오버튼과 슬라이더로 숫자를 입력할 수 있는 스케일 위젯을 이용한 프로그램을 작성

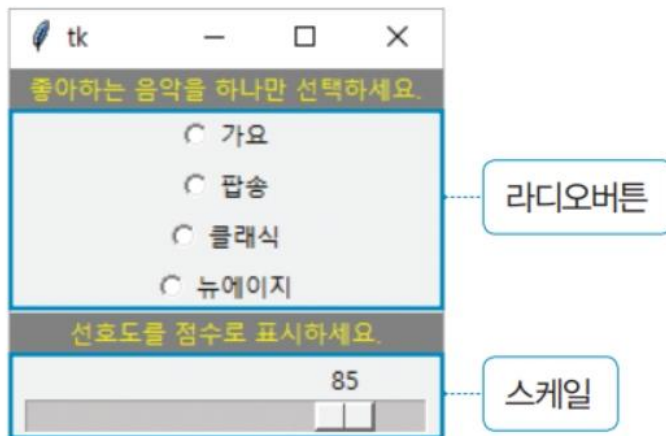


그림 11-11 실행 결과

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

① 필요한 위젯의 속성

표 11-3 라디오 버튼과 스케일 위젯 속성

위젯명	속성	설명	기본값
Radiobutton	value	항목별 값을 설정	-
	variable	value 값을 저장하는 변수	-
Scale	width	슬라이더의 너비	15
	length	슬라이더의 길이	100
	orient	슬라이더의 표시 방향	vertical
	tickinterval	수치 값 간격	0

② 기본 윈도우를 생성하고 레이블과 라디오버튼을 만들어 실행

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

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

- ② 기본 윈도우를 생성하고 레이블과 라디오버튼을 만들어 실행

레이블 만들기

위젯 객체 생성과 배치를 한 문장으로 실행

```
Label(w, text="좋아하는 음악을 하나만 선택하세요.", bg="gray", fg="yellow", width=30).pack()
```

라디오버튼 만들기

x = IntVar() # 정수형 변수 생성(라디오버튼에서 선택한 값을 저장하기 위한 변수)

```
Radiobutton(w, text="가요", variable=x, value=1).pack()
```

```
Radiobutton(w, text="팝송", variable=x, value=2).pack()
```

```
Radiobutton(w, text="클래식", variable=x, value=3).pack()
```

```
Radiobutton(w, text="뉴에이지", variable=x, value=4).pack()
```

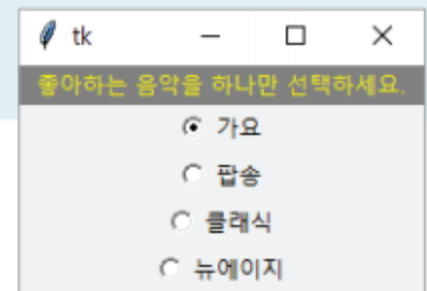


그림 11-12 라디오버튼 생성

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

- ③ 슬라이더의 설명을 위한 레이블과 스케일 위젯으로 만든 슬라이더를 윈도우에 추가

```
# 레이블 만들기
```

```
Label(w, text="선호도를 점수로 표시하세요.", bg="gray", fg="yellow", width=30).pack()
```

```
# 슬라이더 만들기
```

```
y = IntVar()
```

```
Scale(w, variable=y, orient="horizontal", length=200).pack()
```

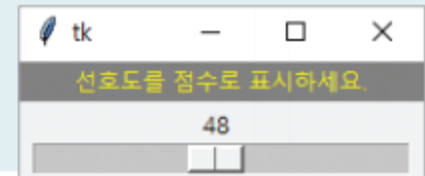


그림 11-13 슬라이더 생성

- ✓ **TIP** 슬라이더의 방향(그림 11-13 슬라이더 생성 orient)과 길이(length)의 기본값은 'vertical'과 100

02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

- ④ 이벤트가 발생하면 처리하기 위해 `mainloop()`을 마지막에 추가

```
01 from tkinter import *
02
03 w = Tk()
04
05 Label(w, text="좋아하는 음악을 하나만 선택하세요.",
06       bg="gray", fg="yellow", width=30).pack()
07
08 x = IntVar()
09 Radiobutton(w, text="가요", variable=x, value=1).pack()
10 Radiobutton(w, text="팝송", variable=x, value=2).pack()
11 Radiobutton(w, text="클래식", variable=x, value=3).pack()
12 Radiobutton(w, text="뉴에이지", variable=x, value=4).pack()
13
14 Label(w, text="선호도를 점수로 표시하세요.",
15       bg="gray", fg="yellow", width=30).pack()
16
17 y = IntVar()
```


02. GUI 프로그래밍

II. 다양한 위젯의 사용

실습 11-3

라디오버튼과 스케일 위젯 사용하기

code11-03.py

- ④ 이벤트가 발생하면 처리하기 위해 `mainloop()`을 마지막에 추가

```
18 Scale(w, variable=y, orient="horizontal", length=200).pack()  
19  
20 w.mainloop()
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

여기서 잠깐

tkinter 모듈에서의 변수 생성

- 정수형 변수를 만들 때는 `IntVar()`, 실수형은 `DoubleVar()`, 문자형에는 `StringVar()`을 사용
- 변수의 값을 확인하거나 변경할 때에는 다음 코드처럼 `get()`과 `set()` 메소드를 사용

```
from tkinter import *  
w = Tk()  
  
x = DoubleVar() # 실수형 변수 생성  
print(x.get())  # x의 값을 셸에 출력  
  
x.set(123.45)   # x의 값을 변경  
print(x.get())  # 변경된 값을 셸에 출력  
  
w.mainloop()
```



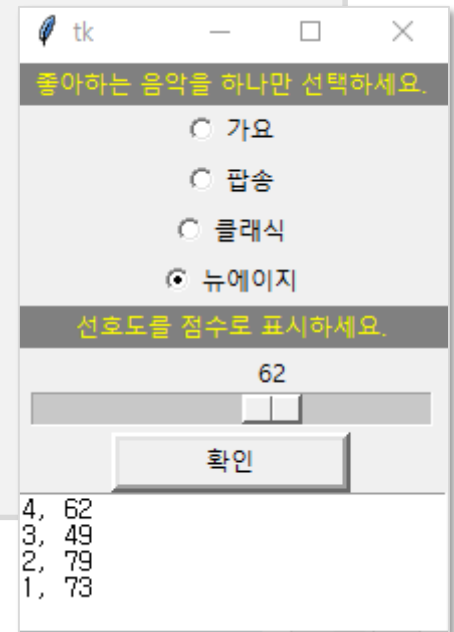
그림 11-14 `get()`, `set()` 메소드 사용

02. GUI 프로그래밍

II. 다양한 위젯의 사용

[실습] <실습 11-3> 업그레이드 (Text box에 결과 값 쓰기)

```
def btn1_call():  
    print(x.get(), y.get())          #객체 변수로부터 값 얻기  
    line = str(x.get()) + ', ' + str(y.get()) + '\n'  
    txt.insert(CURRENT, line)       #Text box 현재 위치에 문자열 쓰기  
  
bt1 = Button(w, text="확인", width=15, borderwidth=4, cursor="hand2")  
bt1.config(command=btn1_call)  
bt1.pack()  
  
txt = Text(w, width=50, height=10)  #Text box 생성  
txt.pack()
```



02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ 위젯의 배치

표 11-4 위젯을 배치하는 방법

배치 방법	설명	사용 예
Pack	위에서 아래 방향으로 순서대로 배치하고, 구역이나 방향을 설정	<code>bt.pack(side="left")</code>
Grid	행(row)과 열(column) 번호로 위젯을 배치하여 표(table) 모양으로 표시	<code>bt.grid(row=0, column=0)</code>
Place	위젯을 절대 좌표(x,y)로 배치. 윈도우의 왼쪽 상단이 (0,0)의 위치	<code>bt.place(x=10, y=10)</code>

실습 11-4

Pack으로 위젯 배치하기

code11-04.py

- 위에서부터 순서대로 위젯을 배치하는 Pack 방법을 사용

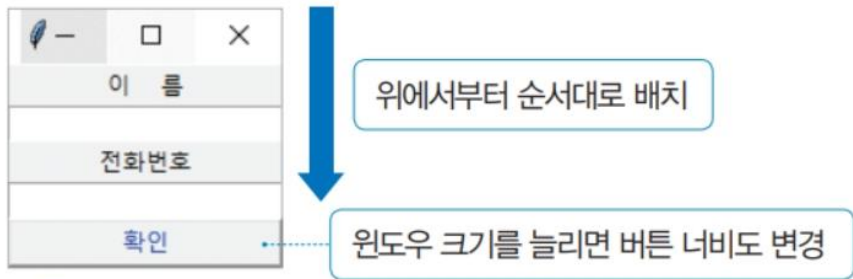


그림 11-15 실행 결과

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-4

Pack으로 위젯 배치하기

code11-04.py

- ① Pack의 여러 속성 가운데 자주 사용되는 기능은 다음과 같고, 버튼의 크기를 배치된 영역에 맞게 채우기 위해 fill 속성을 사용

표 11-5 Pack의 속성

메소드명	속성	설명	기본값
pack()	side	지정된 영역(top, bottom, left, right)에 배치	top
	anchor	배치된 영역 내의 특정 위치(center, n, e, s, w, ne, nw, se, sw)를 지정	center
	fill	위젯 크기를 배치된 영역 크기에 맞게 채우기(none, x, y, both)	-

- ② 기본 윈도우에 각 위젯을 생성하고 pack() 메소드를 사용해 순서에 맞게 배치

```
01 from tkinter import *
02
03 w = Tk()                # 윈도우 제목
04 w.title("회원 가입")
05
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-4

Pack으로 위젯 배치하기

code11-04.py

- ② 기본 윈도우에 각 위젯을 생성하고 pack() 메소드를 사용해 순서에 맞게 배치

```
06 Label(w, text="이름").pack()
07 nameEn = Entry(w)
08 nameEn.pack()
09
10 Label(w, text="전화번호").pack()
11 phoneEn = Entry(w)
12 phoneEn.pack()
13
14 Button(w, text="확인", fg="blue").pack(fill="x") # 버튼 너비를 윈도우에 맞게 채우기
15
16 w.mainloop()
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ 위젯의 배치

실습 11-5

Grid로 위젯 배치하기

code11-05.py

- 열과 행을 지정하는 Grid로 위젯을 배치



2 × 3 모양이고,
마지막 행은 열 병합

그림 11-16 실행 결과

- ① Grid의 여러 속성 가운데 자주 사용되는 기능

표 11-6 Grid 속성

메소드명	옵션명	설명	기본값
grid()	column	열 번호	0
	row	행 번호	0
	columnspan	병합하는 열의 개수	1
	rowspan	병합하는 행의 개수	1
	sticky	셀 내에서 위치 조정(n, e, s, w, nw, ne, sw, se)	-

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-5

Grid로 위젯 배치하기

code11-05.py

- ① Grid의 여러 속성 가운데 자주 사용되는 기능

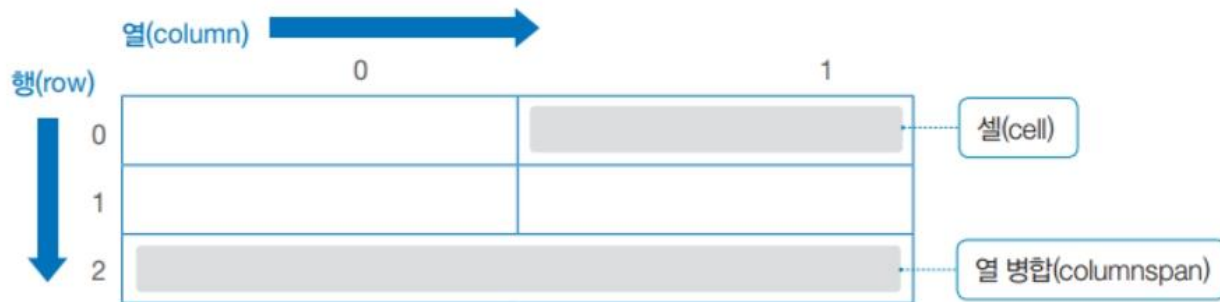


그림 11-17 표의 구성

- ② 기본 윈도우에 각 위젯을 생성하고 `grid()` 메소드를 사용해 열과 행으로 배치

```
01 from tkinter import *
02
03 w = Tk()
04 w.title("회원 가입")
05
```


02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-5

Grid로 위젯 배치하기

code11-05.py

- ② 기본 윈도우에 각 위젯을 생성하고 grid() 메소드를 사용해 열과 행으로 배치

```
06 Label(w, text="이름").grid(row=0, column=0)    # 첫 번째 행
07 nameEn = Entry(w)
08 nameEn.grid(row=0, column=1)
09
10 Label(w, text="전화번호").grid(row=1, column=0) # 두 번째 행
11 phoneEn = Entry(w)
12 phoneEn.grid(row=1, column=1)
13
14 Button(w, text="확인", fg="blue").grid(row=2, column=0, columnspan=2) # 열 병합
15
16 w.mainloop()                                # 이벤트 처리를 위해 대기
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ 위젯의 배치

실습 11-6

Place로 위젯 배치하기

code11-06.py

- ① Place에서는 x, y 좌표와 위젯의 크기 등을 지정하는 속성을 사용할 수 있음

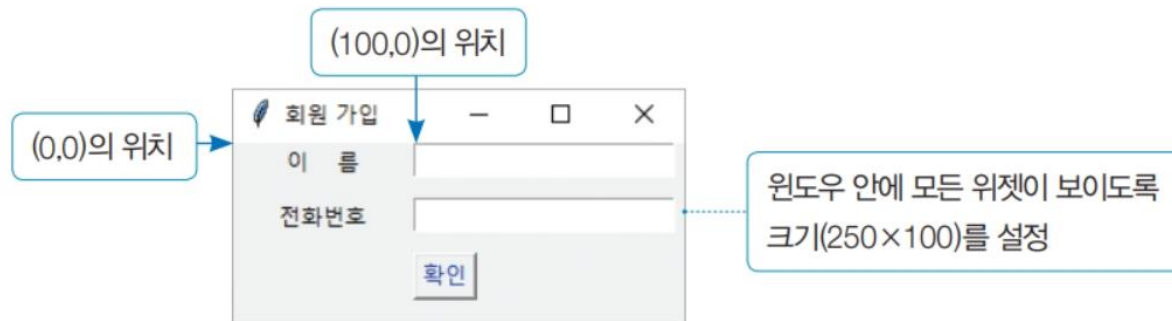


그림 11-18 실행 결과

표 11-7 Place 속성

메소드명	속성	설명	기본값
place()	x	가로 방향 좌표	-
	y	세로 방향 좌표	-
	width	위젯의 너비	-
	height	위젯의 높이	-

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-6

Place로 위젯 배치하기

code11-06.py

- ② 기본 윈도우와 위젯을 생성하고, place() 메소드로 위젯이 표시될 위치를 지정하는 전체 프로그램을 작성하고 실행

```
01 from tkinter import *
02
03 w = Tk()
04 w.title("회원 가입")
05 w.geometry("250x100") # 윈도우 크기("가로x세로") 설정
06
07 Label(w, text="이름").place(x=0, y=0, width=100) # 레이블의 위치와 너비 지정
08 nameEn = Entry(w)
09 nameEn.place(x=100, y=0)
10
```

#위의 line 07을 아래 2줄로 하여 label 객체명 사용 가능

```
lb1 = Label(w, text="이름")
lb1.place(x=0, y=0, width=100)
lb1.config(text="Name")
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 배치

실습 11-6

Place로 위젯 배치하기

code11-06.py

- ② 기본 윈도우와 위젯을 생성하고, place() 메소드로 위젯이 표시될 위치를 지정하는 전체 프로그램을 작성하고 실행

```
11 Label(w, text="전화번호").place(x=0, y=30, width=100)
12 phoneEn = Entry(w)
13 phoneEn.place(x=100, y=30)
14
15 Button(w, text="확인", fg="blue").place(x=100, y=60)
16
17 w.mainloop()
```

이벤트 처리를 위해 대기

02. GUI 프로그래밍

II. 다양한 위젯의 사용

여기서 잠깐 윈도우의 크기와 위치 설정하기

- Tk.geometry()를 사용하면 실행 윈도우의 크기뿐 아니라, 윈도우가 나타나는 스크린의 위치도 설정할 수 있음
- geometry() 메소드의 인수는 문자열 형식이며, 가로, 세로의 크기와 x, y 좌표값으로 사용

표 11-8 geometry 메소드

메소드명	옵션명	설명
geometry()	w	윈도우 너비(픽셀 단위)
	h	윈도우 높이(픽셀 단위)
	±x	x 좌표(+ 값은 스크린 왼쪽 끝에서의 거리, - 값은 오른쪽 끝에서의 거리)
	±y	y 좌표(+ 값은 스크린 위쪽 끝에서의 거리, - 값은 아래쪽 끝에서의 거리)

```
from tkinter import *
```

```
w = Tk()
```

```
w.geometry("250x100+0+0")
```

```
w.mainloop()
```

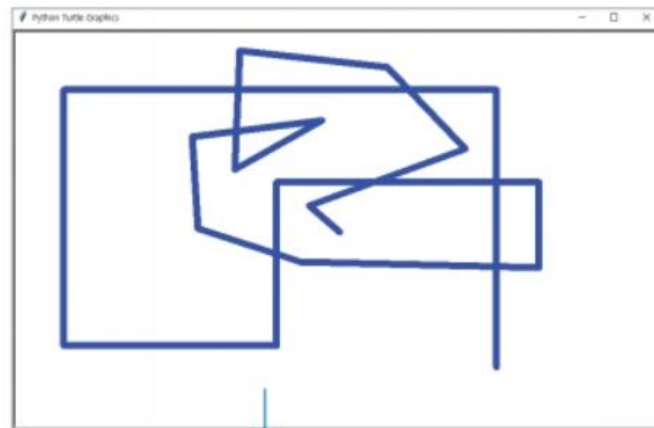
윈도우의 크기는 250×100, 윈도우의 위치는 스크린 왼쪽 상단(+0+0)에 나타나도록 설정

02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ 위젯의 이벤트 처리

- 버튼 클릭이나 키보드 입력 등의 이벤트가 감지되면 그에 알맞은 동작을 만들어야 함
- 이벤트 처리는 함수를 이용하여 필요한 동작을 정의



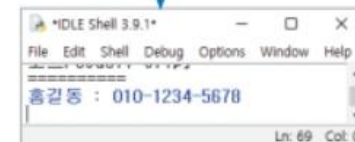
[실습 10-10]의 터틀 그림판 만들기
(마우스 클릭과 방향키에 대한 이벤트 처리)

터틀 그래픽스

그림 11-19 이벤트 처리 비교



버튼 클릭에 대한 이벤트 처리



tkinter위젯 이벤트

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 이벤트 처리

- 버튼 위젯의 command 속성에 이벤트 처리 함수를 지정

```
def 함수명() :  
    이벤트 처리 동작  
  
bt = tkinter.Button(컨테이너, 속성1 = 값1, ..., command = 함수명)
```

- 버튼 위젯의 bind() 메소드로 이벤트 처리 함수를 지정

```
def 함수명(event) :  
    이벤트 처리 동작  
  
bt = tkinter.Button(컨테이너, 속성1 = 값1, ...)  
bt.bind('<이벤트명>', 함수명)
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

■ 위젯의 이벤트 처리

실습 11-7

버튼 위젯에 이벤트 처리 기능 추가하기

code11-07.py

- 이름과 전화번호를 입력하고 버튼을 누르면 입력한 데이터를 셸에 출력

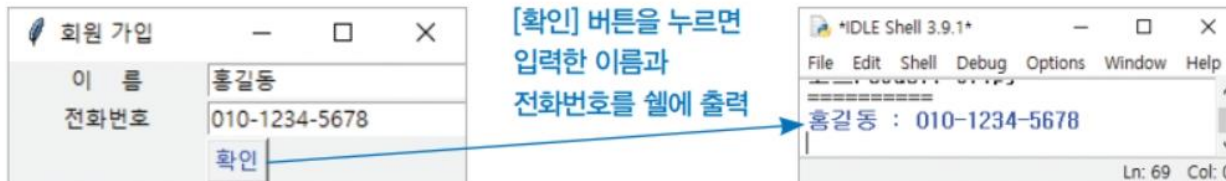


그림 11-20 실행 결과

- 엔트리에 입력된 값은 `get()` 메소드로 추출

```
def bt_click():  
    name = nameEn.get()           # 이름을 가져와 name에 저장  
    phone = phoneEn.get()         # 전화번호를 가져와 phone에 저장  
    print(name, ":", phone)       # 이름과 전화번호를 셸에 출력
```


02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 이벤트 처리

실습 11-7

버튼 위젯에 이벤트 처리 기능 추가하기

code11-07.py

- ① 엔트리에 입력된 값을 get() 메소드로 추출

표 11-9 get 메소드

메소드명	동작	예시
get()	위젯에 입력된 텍스트를 추출	entry.get()

- ② 이벤트 처리 함수를 위젯의 command 속성에 연결한 전체 프로그램

```
01 from tkinter import *
02
03 def bt_click() :
04     name = nameEn.get()      # 이름을 가져와 name에 저장
05     phone = phoneEn.get()    # 전화번호를 가져와 phone에 저장
06     print(name, ":", phone)  # 이름과 전화번호를 셸에 출력
07
08 w = Tk()
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 이벤트 처리

실습 11-7

버튼 위젯에 이벤트 처리 기능 추가하기

code11-07.py

- ② 이벤트 처리 함수를 위젯의 command 속성에 연결한 전체 프로그램

```
09 w.title("회원 가입")
10
11 Label(w, text="이름", width=15).grid(row=0, column=0)
12 nameEn = Entry(w)
13 nameEn.grid(row=0, column=1)
14
15 Label(w, text="전화번호", width=15).grid(row=1, column=0)
16 phoneEn = Entry(w)
17 phoneEn.grid(row=1, column=1)
18
19 bt = Button(w, text="확인", fg="blue", command=bt_click) # 이벤트 처리 함수 연결
20 bt.grid(row=2, column=0, columnspan=2)
21
22 w.mainloop()
```

02. GUI 프로그래밍

II. 다양한 위젯의 사용

▪ 위젯의 이벤트 처리

실습 11-7

버튼 위젯에 이벤트 처리 기능 추가하기

code11-07.py

- ③ bind()로 이벤트 처리 함수를 연결하려면 프로그램을 다음과 같이 수정

```
01 from tkinter import *
02
03 def bt_click(event):          # 함수의 매개변수 추가, 매개변수명은 원하는 대로 작성
04 ~ 18행은 동일
19 bt = Button(w, text="확인", fg="blue")
20 bt.bind('<Button-1>', bt_click) # 이벤트 처리 함수 연결
21 bt.grid(row=2, column=0, columnspan=2)
22
23 w.mainloop()
```

03

tkinter의 활용

01. tkinter의 활용

실습 11-9

GUI 방식의 사칙연산기 만들기

code11-09.py

- 2개의 숫자를 입력하고 연산의 종류를 선택하면 결과를 출력

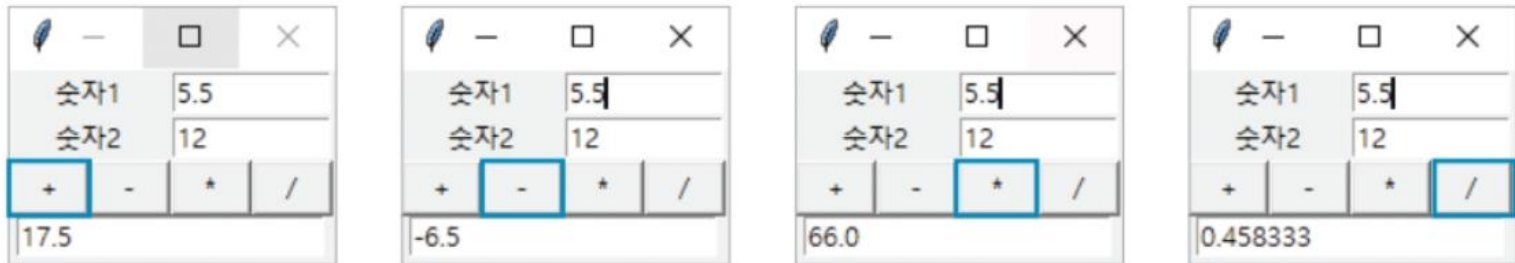


그림 11-24 실행 결과

- ① 기본 윈도우와 필요한 위젯을 모두 생성한 다음, Grid로 열과 행 번호를 지정하여 배치

```
01 from tkinter import *
02
03 # 이벤트 처리 함수를 정의해 넣을 부분
04
05 w = Tk()
06 w.title("사칙연산기")
07 l1 = Label(w, text="숫자1")
08 l2 = Label(w, text="숫자2")
09 e1 = Entry(w, width=10)
10 e2 = Entry(w, width=10)
```

01. tkinter의 활용

실습 11-9

GUI 방식의 사칙연산기 만들기

code11-09.py

- ① 기본 윈도우와 필요한 위젯을 모두 생성한 다음, Grid로 열과 행 번호를 지정하여 배치

```
11 bAdd = Button(w, text="+", width=4)
12 bSub = Button(w, text="-", width=4)
13 bMul = Button(w, text="*", width=4)
14 bDiv = Button(w, text="/", width=4)
15 eResult = Entry(w)                # 결과를 표시하는 엔트리
16
17 l1.grid(row=0, column=0, columnspan=2) # Grid 배치
18 e1.grid(row=0, column=2, columnspan=2)
19 l2.grid(row=1, column=0, columnspan=2)
20 e2.grid(row=1, column=2, columnspan=2)
21 bAdd.grid(row=2, column=0)
22 bSub.grid(row=2, column=1)
23 bMul.grid(row=2, column=2)
24 bDiv.grid(row=2, column=3)
25 eResult.grid(row=3, column=0, columnspan=4)
26
27 w.mainloop()                    # 이벤트 처리를 위해 대기
```

01. tkinter의 활용

실습 11-9

GUI 방식의 사칙연산기 만들기

code11-09.py

- ② 2개의 숫자 엔트리에 입력된 값을 추출하여 계산하는 4개의 함수를 정의

```
def f_add() :                                # 덧셈 버튼 클릭 이벤트 처리
    num1 = float(e1.get())                   # 엔트리에 입력된 숫자 가져오기
    num2 = float(e2.get())
    eResult.delete(0, END)                  # 기존에 표시한 결과를 모두 지우기
    eResult.insert(0, "%s" % (num1+num2))    # 계산 결과를 표시하기
```

```
def f_sub() :                                # 뺄셈 버튼 클릭 이벤트 처리
    num1 = float(e1.get())
    num2 = float(e2.get())
    eResult.delete(0, END)
    eResult.insert(0, "%s" % (num1-num2))
```

01. tkinter의 활용

실습 11-9

GUI 방식의 사칙연산기 만들기

code11-09.py

- ② 2개의 숫자 엔트리에 입력된 값을 추출하여 계산하는 4개의 함수를 정의

```
def f_mul() :                                # 곱셈 버튼 클릭 이벤트 처리
    num1 = float(e1.get())
    num2 = float(e2.get())
    eResult.delete(0, END)
    eResult.insert(0, "%s" % (num1*num2))
```

```
def f_div() :                                # 나눗셈 버튼 클릭 이벤트 처리
    num1 = float(e1.get())
    num2 = float(e2.get())
    eResult.delete(0, END)
    eResult.insert(0, "%.6f" % (num1/num2)) # 자릿수를 소수점 이하 6자리로 표시
```

표 11-11 엔트리의 값을 삭제, 추가하는 메소드

메소드명	동작	예시	인수 설명
delete()	지정한 범위의 텍스트를 삭제	entry.delete(0, "end")	(시작 위치, 끝 위치)
insert()	지정한 위치에 텍스트를 추가	entry.insert(0, "안녕")	(추가할 위치, "문자열")

01. tkinter의 활용

실습 11-9

GUI 방식의 사칙연산기 만들기

code11-09.py

- ③ 하나의 프로그램으로 (1) 위젯 만들기와 (2) 함수 정의 코드를 저장

```
01 from tkinter import *
02
03~26 # 단계 2에서 정의한 4개의 함수를 넣는 부분
27~32 # 단계 1의 05~10행과 동일
33 bAdd = Button(w, text="+", width=4, command=f_add)
34 bSub = Button(w, text="-", width=4, command=f_sub)
35 bMul = Button(w, text="*", width=4, command=f_mul)
36 bDiv = Button(w, text="/", width=4, command=f_div)
37~46 # 단계 1의 15~27행과 동일
```

버튼별 함수 연결

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py



그림 11-25 실행 결과

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ① 그림판의 모양을 먼저 만들고 처리할 동작을 함수로 만들어 위젯에 연결



그림 11-26 그림판 작성 순서

- ② 기본 윈도우를 생성하고, 그림을 그리기 위해 캔버스 위젯을 추가

```
from tkinter import *  
w = Tk()  
w.title("그림판")  
cv = Canvas(w, width=800, height=600, background="lightblue")  
cv.pack()
```

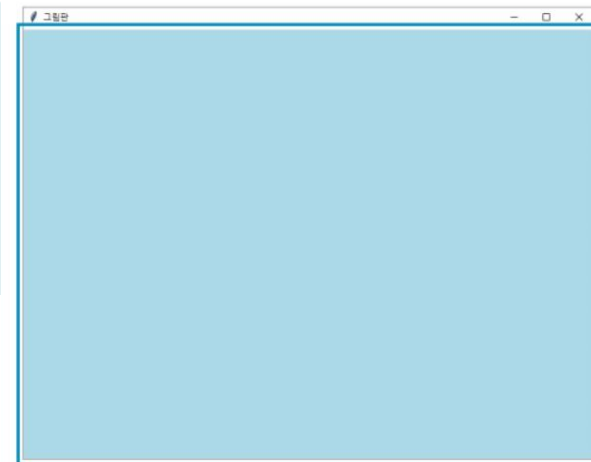


그림 11-27 윈도우에 캔버스 생성

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ③ 그림을 그릴 때 필요한 버튼을 배치하기 위해 프레임을 만들고 캔버스 아래에 추가

```
fr = Frame(w)
fr.pack()
```

- ④ 버튼 4개를 만들어 프레임에 추가하고, 행과 열 번호를 지정하여 Grid로 배치

```
bThick = Button(fr, text="두껍게", font="Tahoma")
bThin = Button(fr, text="가늘게", font="Tahoma")
bClear = Button(fr, text="지우기", font="Tahoma")
bExit = Button(fr, text="나가기", font="Tahoma")
bThick.grid(row=0, column=0)
bThin.grid(row=0, column=1)
bClear.grid(row=0, column=2)
bExit.grid(row=0, column=3)

w.mainloop()
```



그림 11-28 프레임과 버튼 추가

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ⑤ 각 버튼을 클릭할 때 실행할 동작을 다음과 같이 함수로 정의

```
penWidth = 5                # 펜 두께 설정

def f_thick():               # 두껍게 버튼을 눌렀을 때
    global penWidth
    penWidth = 10

def f_thin():                # 가늘게 버튼을 눌렀을 때
    global penWidth
    penWidth = 5

def f_clear():               # 지우기 버튼을 눌렀을 때
    cv.delete("all")

def f_exit():                # 나가기 버튼을 눌렀을 때
    w.destroy()
```

표 11-12 캔버스의 delete()와 destroy() 메소드

메소드명	동작	예시
delete()	캔버스(cv) 위의 특정 항목(타원, 사각형, 선, 호, 글자 등)이나 모든 항목을 삭제	cv.delete('항목명') cv.delete("all")
destroy()	기본 윈도우(w) 닫기	w.destroy()

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ⑥ 각 버튼을 생성하는 문장에 command 속성을 추가해서 이벤트 처리 함수를 연결

```
bThick = Button(fr, text="두껍게", font="Tahoma", command=f_thick)
bThin = Button(fr, text="가늘게", font="Tahoma", command=f_thin)
bClear = Button(fr, text="지우기", font="Tahoma", command=f_clear)
bExit = Button(fr, text="나가기", font="Tahoma", command=f_exit)
```

문장 수정

- ⑦ 캔버스 위에서 그림을 그리는 함수를 정의

```
def f_draw(event) :
    global penWidth
    x = event.x                # 마우스의 현재 좌표
    y = event.y
    cv.create_oval(x, y, x+1, y+1, width=penWidth)  # width는 타원의 선 두께
```

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ⑧ 캔버스 위에서 마우스 왼쪽 버튼을 눌렀을 때(), 처리할 동작으로 f_draw() 함수를 연결(cv.bind())

```
cv.bind("<B1-Motion", f_draw)      # 마우스 왼쪽 버튼 클릭을 함수 f_draw에 연결
```

- ⑨ 작성한 전체 프로그램

```
01 from tkinter import *
02
03 penWidth = 5                      # 펜 두께 설정(전역변수로 사용)
04
05 def f_thick():
06     global penWidth
07     penWidth = 10
08
09 def f_thin():
10     global penWidth
11     penWidth = 5
12
13 def f_clear():
```

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

⑨ 작성한 전체 프로그램

```
14     cv.delete("all")
15
16     def f_exit() :
17         w.destroy()
18
19     def f_draw(event) :
20         global penWidth
21         x = event.x                # 마우스의 현재 좌표
22         y = event.y
23         cv.create_oval(x, y, x+1, y+1, width=penWidth) # width는 타원의 선 두께
24
25     w = Tk()
26     w.title("그림판")
27     cv = Canvas(w, width=800, height=600, background="lightblue")
28     cv.pack()
29     cv.bind("<B1-Motion>", f_draw) # 마우스 왼쪽 버튼 클릭을 함수 f_draw에 연결
30     fr = Frame(w)
```

01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

⑨ 작성한 전체 프로그램

```
31 fr.pack()                                # 프레임을 윈도우에 추가
32 bThick = Button(fr, text="두껍게", font="Tahoma", command=f_thick)
33 bThin = Button(fr, text="가늘게", font="Tahoma", command=f_thin)
34 bClear = Button(fr, text="지우기", font="Tahoma", command=f_clear)
35 bExit = Button(fr, text="나가기", font="Tahoma", command=f_exit)
36 bThick.grid(row=0, column=0)
37 bThin.grid(row=0, column=1)
38 bClear.grid(row=0, column=2)
39 bExit.grid(row=0, column=3)
40
41 w.mainloop()
```


01. tkinter의 활용

실습 11-10

GUI 방식의 그림판 만들기

code11-10.py

- ⑧ 캔버스 위에서 마우스 왼쪽 버튼을 눌렀을 때(), 처리할 동작으로 f_draw() 함수를 연결(cv.bind())

```
cv.bind("<B1-Motion>", f_draw)      # 마우스 왼쪽 버튼 클릭을 함수 f_draw에 연결
```

Bind Event

Button

이름	의미
<Button-1>	마우스 왼쪽 버튼을 누를 때
<Button-2>	마우스 휠 버튼을 누를 때
<Button-3>	마우스 오른쪽 버튼을 누를 때
<Button-4>	스크롤 업
<Button-5>	스크롤 다운
<MouseWheel>	마우스 휠 이동

01. tkinter의 활용

Motion

이름	의미
<Motion>	마우스가 움직일 때
<B1-Motion>	마우스 왼쪽 버튼을 누르면서 움직일 때
<B2-Motion>	마우스 휠 버튼을 누르면서 움직일 때
<B3-Motion>	마우스 오른쪽 버튼을 누르면서 움직일 때

Release

이름	의미
<ButtonRelease-1>	마우스 왼쪽 버튼을 땄 때
<ButtonRelease-2>	마우스 휠 버튼을 땄 때
<ButtonRelease-3>	마우스 오른쪽 버튼을 땄 때

01. tkinter의 활용

Double Click

이름	의미
<Double-Button-1>	마우스 왼쪽 버튼을 더블 클릭할 때
<Double-Button-2>	마우스 휠 버튼을 더블 클릭할 때
<Double-Button-3>	마우스 오른쪽 버튼을 더블 클릭할 때

Widget Operation

이름	의미
<Enter>	위젯 안으로 마우스 포인터가 들어왔을 때
<Leave>	위젯 밖으로 마우스 포인터가 나갔을 때
<FocusIn>	위젯 안으로 Tab 키를 이용하여 들어왔을 때
<FocusOut>	위젯 밖으로 Tab 키를 이용하여 나갔을 때
<Configure>	위젯의 모양이 수정되었을 때

01. tkinter의 활용

Key Input

이름	의미
<Key>	특정 키가 입력되었을 때
<Return>	Enter 키가 입력되었을 때
<Cancel>	Break 키가 입력되었을 때
<Pause>	Pause 키가 입력되었을 때
<BackSpace>	백스페이스 키가 입력되었을 때
<Caps_Lock>	캡스 락 키가 입력되었을 때
<Escape>	이스케이프 키가 입력되었을 때
<Home>	Home 키가 입력되었을 때
<End>	End 키가 입력되었을 때
<Print>	Print 키가 입력되었을 때
<Insert>	Insert 키가 입력되었을 때
<Delete>	Delete 키가 입력되었을 때
<Prior>	Page UP 키가 입력되었을 때
<Up>	위쪽 방향키가 입력되었을 때
<Down>	아랫쪽 방향키가 입력되었을 때
<Right>	오른쪽 방향키가 입력되었을 때
<Left>	왼쪽 방향키가 입력되었을 때

01.. tkinter의 활용

[실습] 카페 메뉴 주문 받기를 윈도우 방식으로 처리 (1)

◆ 메뉴 딕셔너리(menu) 기반 주문 처리 윈도우를 구성하시오.

- 메뉴 신청을 메뉴별 개별 Button을 구성하여 버튼 클릭으로 주문 처리
- 주문내역은 orders 딕셔너리로 구성 (중복되는 메뉴 주문은 수량 증가로 처리)
- 주문 내역이 추가될 때마다 Text box에 주문내역 출력

```
menu = {'COFFEE': [['에스프레소', 3.0], ['아메리카노', 3.0], ['카페라떼', 4.0], ['카프치노', 4.0],  
'LATTE': [['말차라떼', 4.0], ['초코라떼', 4.0], ['카페라떼', 4.0],  
'TEA': [['청귤차', 4.0], ['자몽차', 4.0], ['레몬차', 4.0],  
'ADE': [['자몽에이드', 4.5], ['레몬에이드', 4.5], ['청포도', 4.5],  
'JUICE': [['망고', 4.5], ['바나나', 4.5], ['딸기', 4.5], ['키위', 4.5],  
'SMOOTHIE': [['청귤스무디', 4.5], ['요거트스무디', 4.5],  
'MILK_TEA': [['흑당밀크티', 4.5], ['달고나밀크티', 4.5]]
```

```
menugroup = {} #대분류 메뉴
```

```
orders = {} #주문내역 딕셔너리
```

```
#def new_menugroup(): #대분류 메뉴표 딕셔너리 생성
```

```
#def win_text_insert(): #Text박스에 주문내역 갱신 출력
```

```
#def btn_menu(i, j): #주문 메뉴 orders 딕셔너리에 추가 (i:클릭 버튼 행 번호, j:클릭 버튼 열번호)
```



01. . tkinter의 활용

[실습] 카페 메뉴 주문 받기를 윈도우 방식으로 처리 (2)

```
##### Main #####
new_menugroup()    #menu(소분류) 딕셔너리로부터 menugroup(대분류) 딕셔너리 구성

#기본 윈도우 생성
w = Tk()
w.title("부천대학 컴퓨터소프트웨어과 Dreammer Cafe")    #윈도우 타이틀("제목")
w.geometry("700x500+100+100")    #윈도우 크기 및 기준점("너비x높이+x좌표+y좌표")
w.resizable(1, 1)    #창 크기 조절 가능 여부(상하, 좌우)

#menu 딕셔너리 기반 주문 버튼 생성
ix, iy = (0, 0)
xp, yp = (0, 0)
for menugrp, mlist in menu.items():
    xp = 10;
    yp = iy * 30
    Label(w, text=menugrp, bg="maroon", fg="yellow", relief="ridge").place(x=xp, y=yp, width=100)
    ix = 0
    for mmenu, price in mlist:
        xp = 120 + ix * 140
        menutxt = mmenu + ' ' * int(14 - len(mmenu) * 1.0) + str(price)
        #Button 클릭 시 호출 함수로 버튼 위치(행, 열) 전달 >> lambda 함수 호출 방식 적용
        Button(w, text=menutxt, command=lambda i=ix, j=iy: btn_menu(i, j), anchor='w', bg="lightgray", fg="navy",
        relief="raised",
            overrelief="sunken").place(x=xp, y=yp, width=130)
        ix += 1
    iy += 1

#Text box 생성 (주문내역 표시)
txt1 = Text(w)    #Text 위젯 생성
txt1.place(x=120, y=yp+40, width=550, height=250)

w.mainloop() # 이벤트 처리를 위해 대기
#####
```

01. . tkinter의 활용

[실습] 카페 메뉴 주문 받기를 윈도우 방식으로 처리 (3)

```
def win_text_insert(): #Text박스에 주문내역 갱신 출력
    global orders
    global txt1
    txt1.delete(1.0, 100.100) #이전 내용 삭제
    totqty, totwon = (0, 0)
    seq = 0
    for mmenu, olist in orders.items():
        seq += 1
        price, qty, totprice = olist
        line = "[{0:>2}] {1:<10} {2:>5} {3:>3} {4:>8}".format(seq, mmenu, price, qty, totprice)
        txt1.insert(CURRENT, line+'\n')
        totwon += totprice
        totqty += qty
    line = "{}".format("-"*42)
    txt1.insert(CURRENT, line+'\n')
    line = "주문 합계: {0:>16}잔 {1:>10}원".format(totqty, totwon)
    txt1.insert(CURRENT, line+'\n')

def btn_menu(i, j): #주문 메뉴 orders 딕셔너리에 추가 (i:클릭 버튼 행 번호, j:클릭 버튼 열번호)
    global menu
    mkey = menugroup[j]
    mlist = menu.get(mkey)
    mmenu = mlist[i][0]; price = mlist[i][1]; qty = 1; totprice = mlist[i][1] * 1000;
    # 주문내역 orders 딕셔너리에 추가
    if mmenu in orders.keys(): #중복되는 주문 메뉴는 수량만 증가 처리
        orderlist = orders.get(mmenu)
        price, qty, totprice = orderlist
        qty += 1
        totprice = price * 1000 * qty
    orders[mmenu] = [price, qty, totprice] #주문 메뉴 갱신 또는 추가 처리
    win_text_insert() #주문 버튼 생성
```

Thank You !

[Python]